SOFTWARE ENGINEERING LABORATORY SEL-90-003

A STUDY OF THE PORTABILITY OF AN ADA SYSTEM IN THE SOFTWARE ENGINEERING LABORATORY (SEL)

JUNE 1990

(NASA-TM-103413) A STUDY OF THE PORTABILITY OF AN Ada SYSTEM IN THE SOFTWARE ENGINEERING LABORATORY (SEL) (NASA) 72 p CSCL 09B

N91-15771

Unclas

G3/61 0326319

National Aeronautics and
Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771

SOFTWARE ENGINEERING LABORATORY

SEL-90-003

A STUDY OF THE PORTABILITY OF AN ADA SYSTEM IN THE SOFTWARE ENGINEERING LABORATORY (SEL)

JUNE 1990

National Aeronautics and
Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771

FOREWORD

The Software Engineering Laboratory (SEL) is an organization sponsored by the National Aeronautics and Space Administration/Goddard Space Flight Center (NASA/GSFC) and created for the purpose of investigating the effectiveness of software engineering technologies when applied to the development of applications software. The SEL was created in 1977 and has three primary organizational members:

NASA/GSFC, Systems Development Branch

The University of Maryland, Computer Sciences Department

Computer Sciences Corporation, Systems Development Operation

The goals of the SEL are (1) to understand the software development process in the GSFC environment; (2) to measure the effect of various methodologies, tools, and models on this process; and (3) to identify and then to apply successful development practices. The activities, findings, and recommendations of the SEL are recorded in the Software Engineering Laboratory Series, a continuing series of reports that includes this document.

The major contributors to this document are

Linda O. Jun (GSFC)

Susan Ray Valett (GSFC)

Single copies of this document can be obtained by writing to

Systems Development Branch
Code 552
Goddard Space Flight Center
Greenbelt, Maryland 20771

ABSTRACT

This document discusses a particular porting effort and gives various statistics on analyzing the portability of Ada and the total staff months (overall and by phase) required to accomplish the rehost, and compares this effort to past experiments on the rehosting of FORTRAN systems. The discussion includes an analysis of the types of errors encountered during the rehosting, the changes required to rehost the system, experiences with the Alsys IBM Ada compiler, the impediments encountered, and the lessons learned during this study.

Table of Contents

Executive Summary xi

Section 1—Introduction 1-1

1.1 Objectives and Scope 1-1

1.2 Background of Rehosted Ada System 1-1

Section 2—Rehost Overview 2-1

2.1 Questionnaire 2-1

2.1.1 Description 2-1

2.1.2 Results From the Questionnaire 2-2

2.2 Findings From the System Overview 2-4

2.3 Description of Rehost Phases 2-5

Section 3—Results 3-1

3.1 Findings of Each Rehost Phase 3-1

3.1.1 Software Preparation 3-1

3.1.2 Compilation 3-1

3.1.3 Linkage 3-1

3.1.4 Execution 3-2

3.1.5 Testing 3-2

3.2 Problem Areas and Solutions 3-2

3.2.1 VAX-Specific Features 3-3

3.2.2 Anomalies With Alsys Compiler 3-3

3.2.3 Compiler Size Limitation 3-3

3.2.4 Adaptation to IBM/MVS Environment 3-4

3.2.5 Rehosted Software Errors 3-4

3.2.6 Compiler Implementation-Dependent Features 3-4

3.3 Rehost Statistics 3-4

3.3.1 Total Effort for Each Phase 3-5

3.3.2 Breakdown of Total Effort by Activity 3-5

3.3.3 Breakdown of Effort by Phase 3-7

3.3.4 Components Changed/Added by Phase 3-7

Table of Contents (Cont'd)

Section 3 (Cont'd)

3.3.5 Components Changed/Added in Each Problem Area 3-7

3.3.6 Time Spent in Each Problem Area 3-7

3.4 Summary Results 3-10

Section 4—Conclusion 4-1

4.1 The Rehost Effort Was a Success 4-1

4.2 ALSYS IBM Ada Compiler Was Not Yet Mature 4-1

4.3 Debugging and Recompilations Were Difficult and Costly 4-1

4.4 VAX-Dependent Features Caused Fewer Problems Than Expected 4-2

4.5 Comparing Ada Rehosting With Other Languages 4-2

4.6 Some Ada Features Promote Portability 4-4

Appendix A—Ada Compiler Characteristics

Appendix B—Developing Ada Systems for IBM/MVS

Appendix C—Questionnaire for GOESIM Rehosting

Appendix D—Monthly Status Reports

Glossary

References

Standard Bibliography of SEL Literature

List of Illustrations

Figure

2-1 Respondents' Levels of Experience 2-1
2-2 Effort Estimates 2-3
2-3 Estimates of Percent of Modules Changed 2-3
2-4 Estimated Problem Areas 2-4
2-5 Estimated Difficulty of Each Phase 2-4
2-6 Estimated Ada Problem Areas 2-5
3-1 Effort To Complete Each Phase 3-5
3-2 Rehost Schedule 3-6
3-3 Total Effort by Activity 3-6
3-4 Components Changed or Added 3-8
3-5 Components Changed 3-8
3-6 Components Added 3-9
3-7 Effort Spent in Each Problem Area 3-9

List of Tables

Table

3-1 Staff Effort by Phase 3-7
4-1 Comparative Effort to Rehost FORTRAN and Ada Telemetry Simulators 4-3

EXECUTIVE SUMMARY

E.1 INTRODUCTION

One of the major goals of Ada was to eliminate the proliferation of language dialects by standardizing a defined Ada syntax. The defined syntax was expected to greatly reduce the level of effort required to port an Ada system from one environment to another. To better understand the issues of portability, a rehost study was conducted in the Software Engineering Laboratory (SEL) at the National Aeronautics and Space Administration/Goddard Space Flight Center (NASA/GSFC). An operational software system consisting of 90,000 lines of code was ported from a VAX 8810 to an IBM 4341.

E.2 OBJECTIVE AND SCOPE

The rehost study was meant to be a concise, self-contained experiment on porting an Ada system. It had the following three objectives:

- Determine the effort required to rehost an operational Ada system
- Quantify the effort characteristics of each element of the rehost effort
- Identify and document subjective impressions, problems, and lessons learned

E.3 RESULTS

The following statements summarize the results of the rehost study:

- The study took a total of 133 staff-days over a 10-month period.
- Compilation took 38 percent of the total effort, and testing took 29 percent.
- At the end of the study, 6.4 percent of the total components were new, and 18 percent of the the total components were changed.
- Ten test cases were executed, and the majority of them passed successfully. (The test failures were due to compiler anomalies.)
- The majority of the problems encountered during the effort were due to the immaturity of the particular compiler employed on the target system.

E.4 CONCLUSION

The following statements summarize the conclusions drawn from the rehost study:

- Subjectively, the effort required to rehost an Ada system seemed to be less than that required to rehost a similar FORTRAN system.
- The rehost effort was considered a success in that all objectives were met.

- The Ada compiler employed was not yet mature enough to support the development of large-scale flight dynamics software systems.
- Debugging was difficult and expensive due to time-consuming recompilations and the lack of a debugging tool.
- Vendor-provided features caused fewer problems than expected.
- User-defined data types were an Ada feature that made the rehost effort easier.

SECTION 1—INTRODUCTION

One of Ada's primary goals was to eliminate the proliferation of both new languages and the numerous dialects of existing languages by standardizing Ada's defined syntax. It was anticipated that a "standard" common language could be defined and made resident on all software platforms. Most studies of Ada have been concerned with whether Ada improves the levels of productivity, reliability, and maintainability; only a few studies have been conducted on the issue of portability (Branyan, 1987; Engle, 1988). To better understand the issues of portability of an Ada system, a rehost study was conducted in the National Aeronautics and Space Administration/Goddard Space Flight Center (NASA/GSFC) Flight Dynamics Division's (FDD) Software Engineering Laboratory (SEL).

1.1 OBJECTIVES AND SCOPE

The study had three objectives: (1) to determine the effort required to rehost an operational Ada system from one environment to another; (2) to quantify the effort characteristics of each element of the rehost effort (file transfer, compiling, linking, execution, and accuracy); and (3) to identify and document subjective impressions, general problems, and "lessons learned" to use for future rehost projects. No performance issues were considered, and no environment comparisons were made. The study was meant to be a concise, self-contained experiment on porting an Ada system.

1.2 BACKGROUND OF REHOSTED ADA SYSTEM

GOESIM, the telemetry simulator for the Geostationary Operational Environmental Satellite (GOES), was the system chosen for rehosting. The simulator models attitude and sensor information from a satellite and transforms it into various output formats to be used to test other software systems. Initially developed on a VAX 8810 in Digital Equipment Corporation (DEC) Ada, the system was ported to a National Advanced Systems (NAS) 8063 (comparable to the IBM 3081) with an Alsys IBM Ada compiler, Version 3.6. The system contains approximately 90 thousand source lines of code (KSLOC) (physical lines), about 50 percent of which were blanks or comments. The system comprised the following 550 components:

- Generic package specifications and bodies: 41
- Package instantiations: 14
- Package specifications and bodies: 116
- Separately compilable subprograms: 373
- FORTRAN routines: 6

Development on the VAX required 72 staff-months over a period of 23 months. Effort expended by secretaries, managers above task manager, and librarians was not included in this figure and is not used in this document. The relative complexity of the system was considered average. No plans for rehosting had been factored into any aspect of the development. Consequently, the simulator made use of some VAX-specific features: for instance, the Direct_Mixed_IO package was used for input/output (I/O).

SECTION 2—REHOST OVERVIEW

This section describes the groundwork that was completed before the actual rehost began. This work included creating and analyzing the questionnaire, meeting with development personnel to overview GOESIM, and planning the steps necessary for the rehost.

2.1 QUESTIONNAIRE

Before the rehosting effort began, a questionnaire was distributed to 31 software developers, researchers, and managers across the flight dynamics development/ research environment. The questionnaire was intended to capture software personnel's general expectations regarding the "portability" characteristics of Ada software. A copy of the questionnaire is provided as Appendix C.

2.1.1 Description

The questionnaire included a precise description of what constituted a successful/ completed rehosting effort. The activities listed in the questionnaire included file transfer, compilation, linkage, execution, testing, and a draft of the final report.

The 24 people who responded ranged from junior programmers with less than 5 years' experience to senior managers with 15 years' experience. Some respondents had Ada experience and some had none, but all were familiar with the flight dynamics environment. Figure 2-1 shows the level of experience of the questionnaire respondents. Approximately 54 percent of the respondents had little or no experience with Ada, 95 percent had a good knowledge of software development, 82 percent had a good knowledge of the VAX, and about 59 percent had little or no experience with the IBM environment.

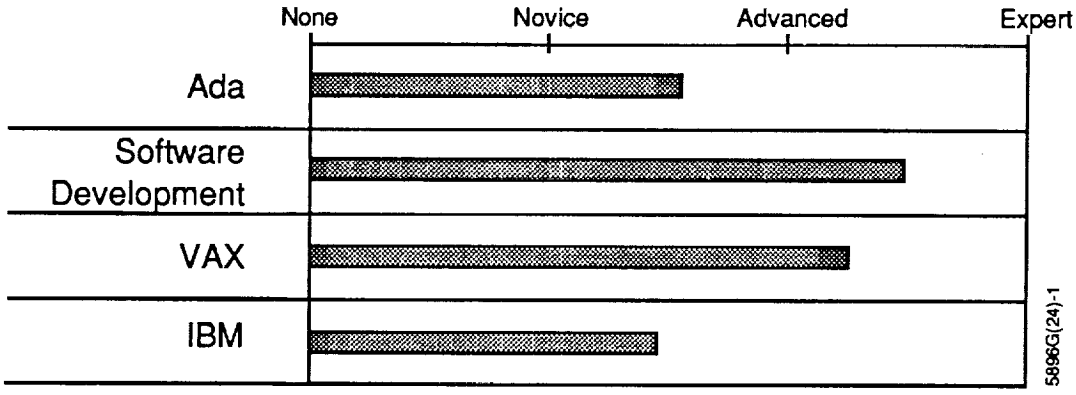


Figure 2-1. Respondents' Levels of Experience

The rehost effort was carried out by two people working half time. The team had worked well together in the past. The first team member's level of experience at the start of the rehost is characterized as follows:

Ada	Advanced
S/W Development	Expert
VAX Environment	Expert
IBM Environment	Expert

The other team member's experience is characterized as follows:

Ada	Novice
S/W Development	Advanced
VAX Environment	Advanced
IBM Environment	Advanced

2.1.2 Results From the Questionnaire

The respondents were asked about their expectations concerning major problems, effort expenditure for each phase of the rehost effort, and percentage of modules requiring modification. Of the 31 questionnaires distributed, 24 were completed and returned.

2.1.2.1 ESTIMATES OF THE TOTAL EFFORT

The average estimated total effort was 168 staff-days, and estimates ranged from 41 to 580 staff-days (Figure 2-2). The majority of respondents thought that preparing software and compilation would take more than 50 percent of the total effort. People with more knowledge in Ada (advanced or expert) estimated a higher total effort (240 days) than people with less knowledge in Ada (none or novice: 108 days).

2.1.2.2 PERCENT OF MODULES CHANGED

The questionnaire asked what percentage of modules would require changes (Figure 2-3). The average response estimated that close to one-third (29 percent) of the software modules would need to be changed. People with more experience in Ada estimated a lower percentage of module changes than people with less experience in Ada (24 percent versus 36 percent).

2.1.2.3 EXPECTED PROBLEM AREAS

The respondents were asked to estimate the level of difficulty for four possible problem areas. All four were estimated to present medium or major problems

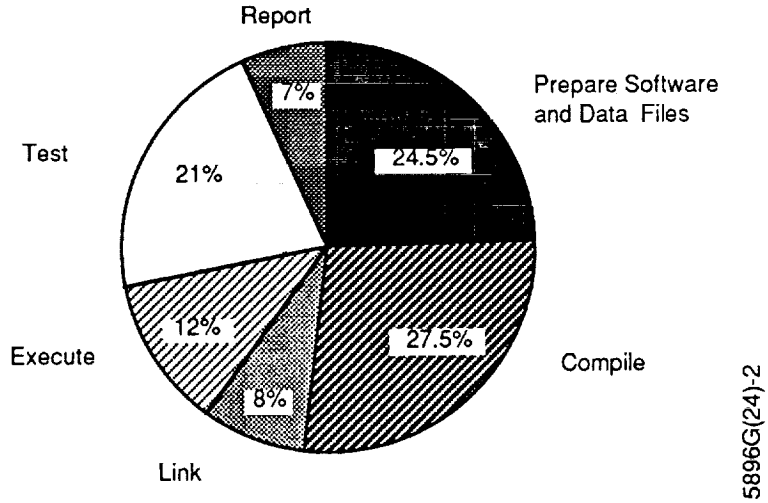


Figure 2-2. Effort Estimates

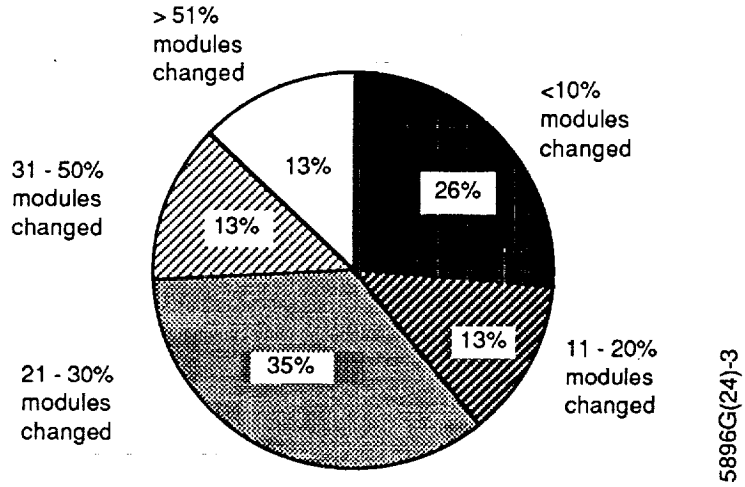


Figure 2-3. Estimates of Percent of Modules Changed

(Figure 2-4). The majority of the respondents thought that VAX-specific Ada features would cause major problems in rehosting the Ada system.

2.1.2.4 RELATIVE DIFFICULTY OF EACH PHASE

Most respondents believed that testing and analysis would be the most difficult phase and linking would be the least difficult phase (Figure 2-5). Many people

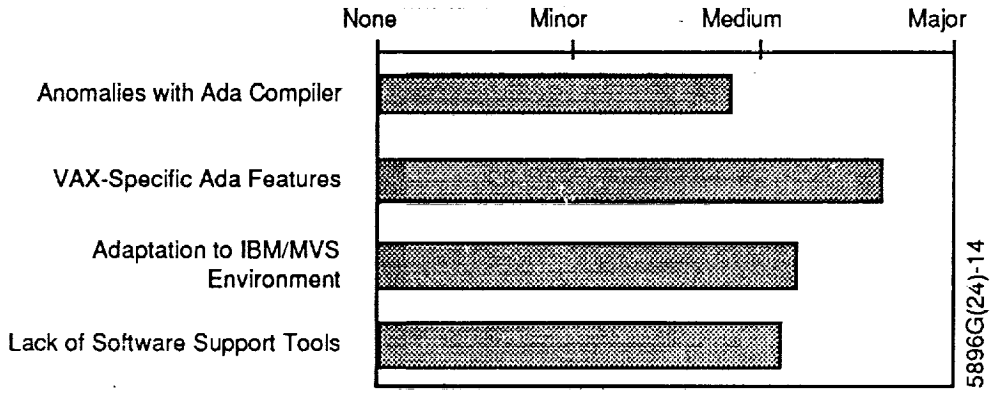


Figure 2-4. Estimated Problem Areas

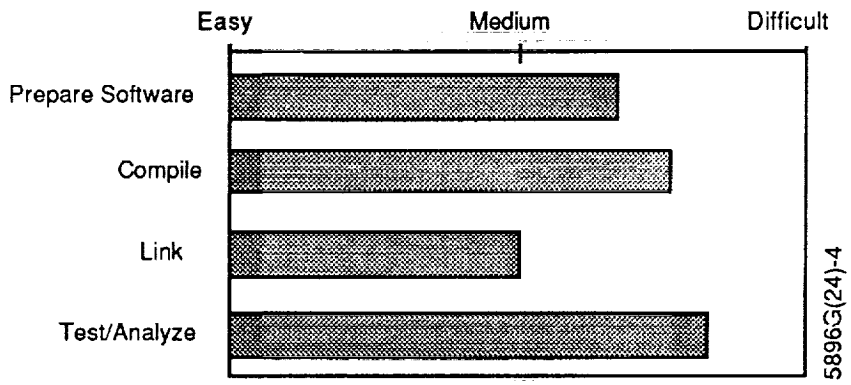


Figure 2-5. Estimated Difficulty of Each Phase

thought that the lack of a debugging tool would cause a major problem during testing and analysis.

2.1.2.5 TYPES OF ADA PROBLEMS

The questionnaire asked what specific types of problems were most expected (Figure 2-6). The average response showed that external I/O and compiler implementation-dependent Ada features would be major problems.

2.2 FINDINGS FROM THE SYSTEM OVERVIEW

The rehost team met with the GOESIM development personnel to get a general overview of the project. During this time, the team identified any VAX-dependent

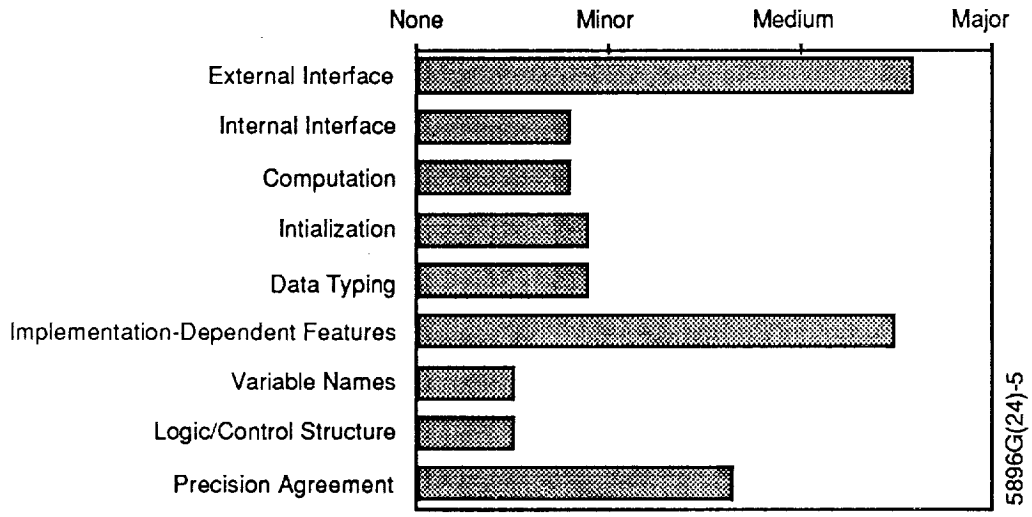


Figure 2-6. Estimated Ada Problem Areas

Ada features, and DEC provided packages and their possible solutions, reviewed the design, examined the ACS library on the VAX to establish a compilation order, listed the component names and their correspondent source file names (since the IBM limited file names to eight characters), and listed the external input data files needed for execution.

During the general overview of GOESIM, the team identified four VAX-dependent features that required special handling during the rehosting: Direct_Mixed_IO, VAX hardware-oriented types and functions, Math_Lib, and pragma INTERFACE to FORTRAN. GOESIM used Direct_Mixed_IO for reading and writing mixed type records from and to a direct access file. For instance, the engineering data file consisted of a header record, an index record, and data records, which were all different record types; GOESIM was required to read or write seven different direct access files. GOESIM used VAX hardware-oriented types and functions implemented in the DEC SYSTEM package for the bit manipulations necessary to generate simulated telemetry data. It used the DEC-provided Math_Lib generic package for trigonometric functions and other transcendental functions. And GOESIM included FORTRAN routines to compute nutation and precession of the equinox and to access direct-access files for obtaining solar and lunar positions.

2.3 DESCRIPTION OF REHOST PHASES

Rehosting comprised six phases: software preparation on the IBM, compilation, linkage, execution, testing/result analysis, and report preparation. The

questionnaire asked for estimates of efforts for each phase, and the "expected" effort was compared to the "actual" effort expended in the rehosting process.

The following activities were performed in each phase:

- Phase 1: Software Preparation
  1. Obtained general overview of GOESIM
  2. Established compilation order
  3. Created source libraries (partitioned data set and PANVALET)
  4. Transferred source files
  5. Moved source files from partitioned data set to controlled PANVALET library
  6. Created an Alsys Ada program library
  7. Implemented VAX hardware-oriented types and functions implemented in SYSTEM package (e.g., Bit_Array types and bit manipulation functions)
- Phase 2: Compilation
  1. Created CLIST to submit compilation in right compilation order
  2. Compiled/recompiled/maintained Alsys Ada program library
  3. Replaced VAX-dependent Direct_Mixed_IO with standard Direct_IO
  4. Implemented Math_Lib (coded TAN, ASIN, ACOS, LOG2) because Math_Lib was not provided by Alsys
  5. Converted six FORTRAN routines to Ada (Alsys compiler did not provide pragma INTERFACE to FORTRAN routines)
- Phase 3: Linkage
  1. Bound and linked to create executable load module
- Phase 4: Execution
  1. Converted two binary input data files to IBM format
  2. Resolved problems that occurred during execution, including compiler problems
  3. Repeated code change/recompilation/linkage
  4. Executed first test without exception or abends

- Phase 5: Testing/Results Analysis
  1. Ran 10 test cases selected from VAX system test cases
  2. Analyzed results
  3. Resolved problems and reran test cases as needed
- Phase 6: Report Preparation
  1. Gathered all information recorded during rehosting
  2. Wrote report describing the experience

SECTION 3—RESULTS

This section presents the experiences of the rehost team with each rehost phase and explores the problem areas and solutions, including workarounds developed.

3.1 FINDINGS OF EACH REHOST PHASE

3.1.1 Software Preparation

During the software preparation phase, an electronic transfer utility was used to move GOESIM source files to the IBM from the VAX. Source files were moved into members of a partitioned data set. Later, these members were moved to a controlled PANVALET library.

Before a compilation of the 540 Ada components was attempted, the team encountered problems with compilation order. Component listings from the VAX "ACS show program/portability" command were very helpful in determining the compilation order. Nevertheless, several attempts were made before all the package specifications were successfully compiled.

3.1.2 Compilation

At the beginning of the compilation phase, the team repeatedly encountered occurrences of library corruption due to abends caused by insufficient central processing unit (CPU) or disk space specified. This in turn required the team to recreate the library and recompile the source units. In the middle of the compilation phase, the Ada program library had to be reinitialized with a larger size, since more components were compiled into the library.

Estimating the disk space required for the program library was difficult. The library was initially allocated to 12,000 blocks; ultimately, 25,000 blocks were needed to complete the compilation successfully. The team tried to avoid overestimating the size of the program library, since the size affected the compilation speed. They first attempted to create multiple libraries with smaller sizes, since the larger the library, the longer compilation took, but they were not successful due to problems with the Alsys Ada unit manager. Since the Alsys IBM Ada compiler Version 3.6 did not provide automatic recompilation, submitting jobs for recompilation for each unit was time consuming and tedious. Later, this recompilation effort was reduced by creating a CLIST to submit compilation in the right compilation order and by routinely backing up the Alsys Ada program library. Normally, one compilation completed within the range of 4 to 160 CPU seconds, or 3 to 140 minutes wall-clock time, depending on the size of the compilation unit and the system workload.

3.1.3 Linkage

Binding and linking were attempted during the linkage phase, and went extremely well once all the components were successfully compiled. Normally, binding took

most of the CPU time (approximately 65 CPU seconds), while linking was relatively fast (about 2 CPU seconds).

3.1.4 Execution

Execution was hindered by the compiler problems and GOESIM software errors. The first attempt to execute the rehosted system was terminated abnormally by a problem in the initialization of record type variables. After a great effort to isolate it, the problem was found in the compiler. Some uninitialized variables in the software caused delays in successfully executing the system. The execution phase was terminated by executing the first test case without analysis of the output results. Before starting the testing phase, the team had to update the GOESIM software, since the source files were moved to the IBM before its completion on the VAX. This effort was necessary in order to compare results from the IBM to results from the VAX; however, it was not included in the actual rehosting effort.

3.1.5 Testing

During the testing phase, 10 test cases were selected and executed from the 30 acceptance test cases on the VAX. These test cases covered the testing of the major functions of the system and all input and output files. The output results were compared against the results from the VAX, and they matched to an acceptable level of accuracy. The problems encountered during testing were caused by Direct_IO and bit manipulation types and functions. The replacement of VAX-dependent Direct_Mixed_IO by standard Direct_IO worked well, in general, except for one input file. Due to problems with Alsys Direct_IO, the rehosted system was not able to read the file correctly even with the use of a record representation clause. The bit manipulation functions that the team replicated on the IBM gave incorrect results in the simulated telemetry data because the most significant bit and least significant bit were reversed from the implementation on the VAX.

3.2 PROBLEM AREAS AND SOLUTIONS

This section describes each problem area, the problem resolutions, and the number of components that were affected by problems during this particular porting effort. All the changes made to the software are categorized into the following problem areas:

- VAX-specific features
- Anomalies with the Alsys compiler
- Size limitation of the compiler
- Adaptation to the IBM/MVS environment
- Rehosted software errors
- Compiler implementation-dependent features

3.2.1 VAX-Specific Features

Use of VAX-specific features required major component changes. VAX-specific features included Direct_Mixed_IO, pragma INTERFACE to FORTRAN routines, F-/D-/G-floating point, VAX hardware-oriented types and functions provided in the package SYSTEM, and a predefined Math_Lib package. Twenty-seven components using Direct_Mixed_IO had to be changed to use Direct_IO instantiations, and 19 components were added for instantiating Direct_IO with appropriate file types. Direct_Mixed_IO was replaced with Direct_IO because replacement was thought to be a straightforward implementation that would require less effort than implementing the Direct_Mixed_IO package body, which contains 15 subprograms. An alternative method of implementing Direct_Mixed_IO would be to use Unchecked Type Conversions. This method would reduce the number of Direct_IO instantiations to five, but it could cause more difficulties in debugging and could give erroneous results. All other VAX-dependent features were replicated on the IBM, although some took longer to implement than expected due to compiler errors (Math_Lib) and reversed bit order of VAX hardware-oriented type (Bit_Array_8), a hidden implementation of the DEC Ada compiler. Six FORTRAN routines were converted to Ada, since Alsys supported INTERFACE to ASSEMBLER only.

3.2.2 Anomalies With Alsys Compiler

The rehost team found a total of five problems with the Alsys IBM Ada compiler Version 3.6 during rehosting. Four were found during execution and one was found during testing. Most of these problems were very difficult to isolate, as the team had to find the problems within the code. In all, 17 components were changed due to these compiler anomalies.

The following are the Alsys IBM Ada compiler Version 3.6 problems found during rehosting:

- Incorrectly evaluates "IF A >=1.0" or "IF A <= 0.0"
- Raises exception "SPURIOUS_ERROR" due to branch to odd address when initializing variant record type variables
- Incorrectly raises "NUMERIC_ERROR" in INTEGER computation
- Skips first record when reading Direct Access file written by non-Ada program
- Incorrectly reads Direct Access file starting with INTEGER, even with record representation clause

3.2.3 Compiler Size Limitation

The rehost team encountered compiler size limitation problems when compiling two packages. One package used more than the allowable size (1,024 bytes) for

each variant part of the record type. This problem was resolved by changing the maximum size and adding a new variant part for the 3x3 matrix. The other package body contained three instantiations of various generic packages that caused STORAGE_ERROR. This problem was resolved by making each instantiation a separate library unit.

3.2.4 Adaptation to IBM/MVS Environment

Several changes were necessary to adapt to the IBM/MVS environment. Several components, such as RECORD_FORMAT and RECORD_LENGTH in the "FORM" parameter, were changed to comply with the IBM/MVS file attributes. A change was also made to resolve NAME_ERRORs when creating files with duplicate names. (The VAX allows the creation of files with duplicate names; it will create the new file with a different version number.) Ephemeris_Type package had to be changed to declare an Extended Binary Coded Decimal Interchange Code (EBCDIC) string instead of an American Standard Code for Information Interchange (ASCII) one in order to read the IBM-resident GOES ephemeris file. Some of the "/=" test had to be rewritten to add a tolerance range to resolve precision problems when comparing double precision numbers to single precision numbers.

3.2.5 Rehosted Software Errors

Several changes were made to incorporate error corrections in GOESIM software. One software error that should have been caught by the VAX DEC Ada Compiler during compilation was an array aggregate with an OTHERS choice for an unconstrained array. Some other problems encountered were due to variables not being initialized, and CONSTRAINT_ERRORs raised during the writing of debug output or long error messages.

3.2.6 Compiler Implementation-Dependent Features

Problems with the Alsys compiler implementation-dependent characteristics were minor. Changes required due to different predefined floating-point type names between the VAX and the IBM were localized to only three components because GOESIM declared its own floating-point data types and used them consistently throughout the system. Use of pragma PACK was ignored by the Alsys compiler, which eliminated some component changes. One component changed due to a minimum size requirement of 16 bits on the IBM for the type "UNSIGNED_BYTE IS RANGE 0..255", which required only 8 bits on the VAX.

3.3 REHOST STATISTICS

During the rehosting, the rehost team kept detailed statistics of the entire effort, including problems. These statistics were gathered monthly and distributed in

status reports (see Appendix D). The total statistics were gathered when the testing phase was complete and were rearranged to better present the rehosting effort. The subsequent sections present detailed statistics on effort, schedule, components changed/added, and time spent on problem areas, by phase. It should be noted that the compilation phase also included other activities, such as resolving VAX-dependent features and creating CLISTs to ease recompilation.

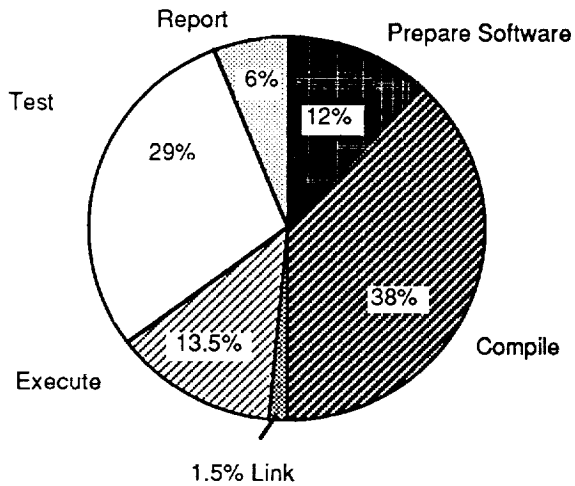
3.3.1 Total Effort for Each Phase

The following charts show the total effort required in staff-days to successfully complete each phase (Figure 3-1) and the schedule during the rehost study (Figure 3-2). It took a total of 133 staff-days over a 10-month period to complete the rehost up to the first draft of the report. As shown in Figure 3-1, 51 staff-days (38 percent of the total effort) were spent in the compilation phase and 38 staff-days (29 percent) were spent in testing.

3.3.2 Breakdown of Total Effort by Activity

A breakdown of the effort spent in each activity is shown in Figure 3-3. The CODE activity included

- Conversion of FORTRAN to Ada
- Implementation of Math_Lib, VAX hardware-oriented types, and functions
- Code changes and new code



Phase	Staff Days
Prepare S/W	16
Compile	51
Link	2
Execute	18
Test	38
Draft Report	8
Total	133

5896G(24)-6

Figure 3-1. Effort To Complete Each Phase

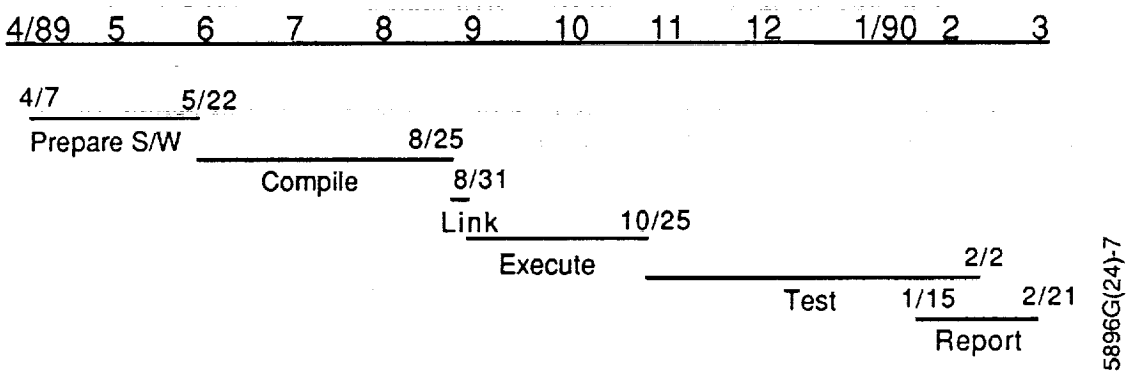


Figure 3-2. Rehost Schedule

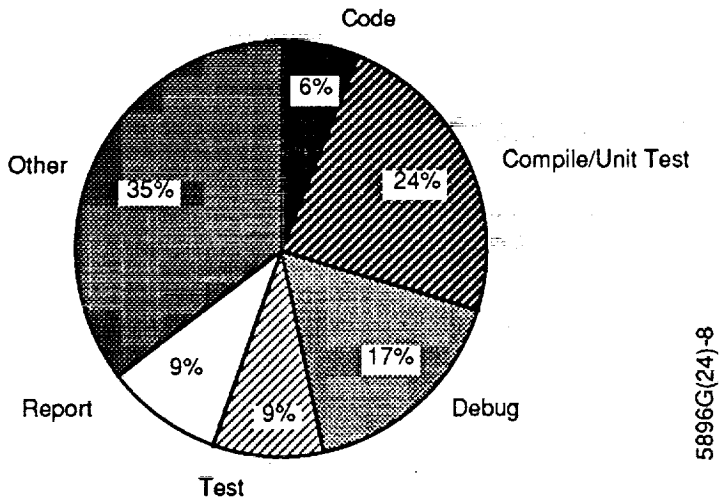


Figure 3-3. Total Effort by Activity

The OTHER activity included

- Transfer of source files and data files
- JCL and CLIST generation
- Input data file conversion
- Monthly status reports and SEL forms
- Ada library maintenance

The TESTING activity included both the integration testing and system testing, and DEBUGGING included all the debugging activity involved throughout testing.

3.3.3 Breakdown of Effort by Phase

Table 3-1 shows the breakdown of each activity for each particular phase.

Table 3-1. Staff Effort by Phase

	Prepare Software	Compile	Link	Execute	Test	Report	Total
Code	16.4	36.3		1.9	10.9		65.5
Compile/Test	30.9	154.6	12.1	33.7	20.5		251.8
Debug		50.0		44.2	85.8		180.0
Test				7.4	86.1		93.5
Report					34.4	62.2	96.6
Other	77.4	164.8	3.6	55.2	70.2	3.3	374.5
Staff-Hours	124.7	405.7	15.7	142.4	307.9	65.5	1,061.9

5896G(24)-9

3.3.4 Components Changed/Added by Phase

At the end of the study, GOESIM consisted of 564 components; 102 were changed, and 45 were added (9 components out of the 45 were made separate compilable units to ease debugging). Of the 102 components changed, 66 percent were changed during compilation, 18 percent were changed during execution, and 16 percent were changed during testing. Of the 45 new components, 75.5 percent were from compilation, 15.5 percent were from execution, and 9 percent were from testing (Figure 3-4).

3.3.5 Components Changed/Added in Each Problem Area

Figure 3-5 and 3-6 show the components changed and added as a result of resolving various types of problems. As shown in the figure, major changes were required due to VAX-specific features. A total of 44 components were changed, and 32 components were added to resolve VAX-dependent features. To ease debugging of the system, nine components were separated from the package body, which reduced the recompilation effort.

3.3.6 Time Spent in Each Problem Area

Figure 3-7 shows the effort spent in each problem area during rehosting. A total of 27 staff-days were spent fixing problems and eliminating VAX-dependent

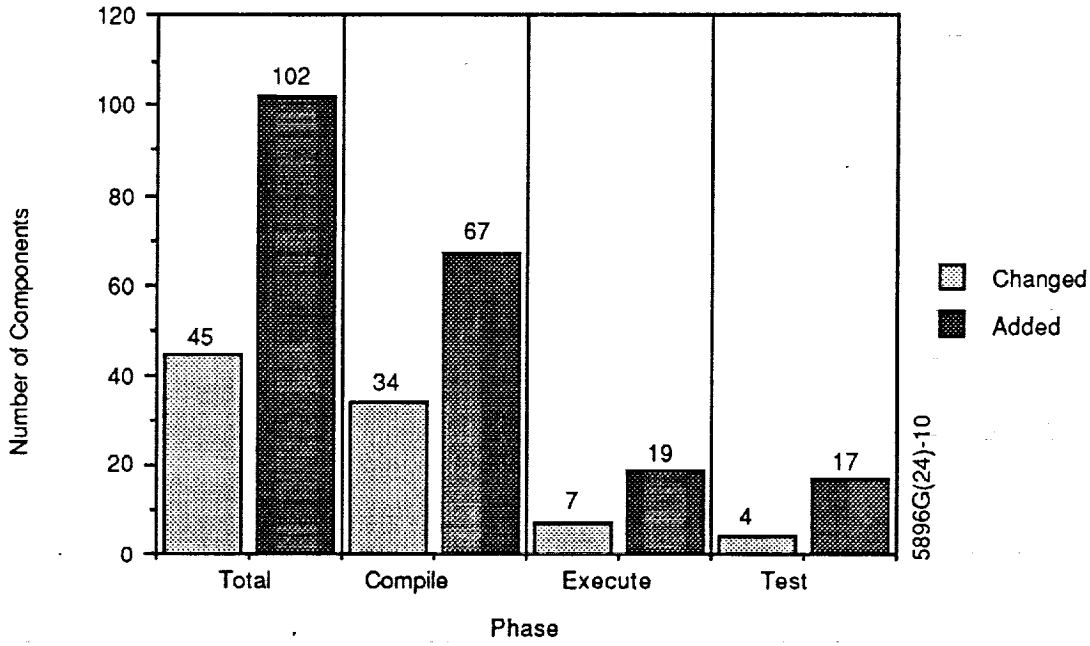


Figure 3-4. Components Changed or Added

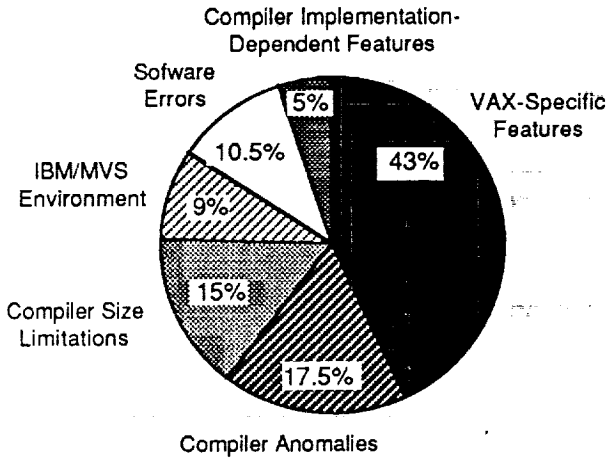


Figure 3-5. Components Changed

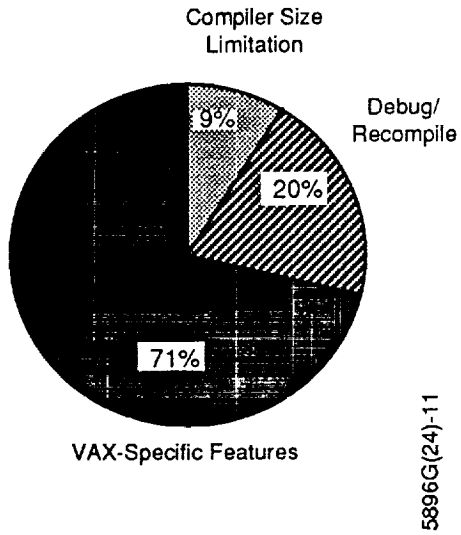


Figure 3-6. Components Added

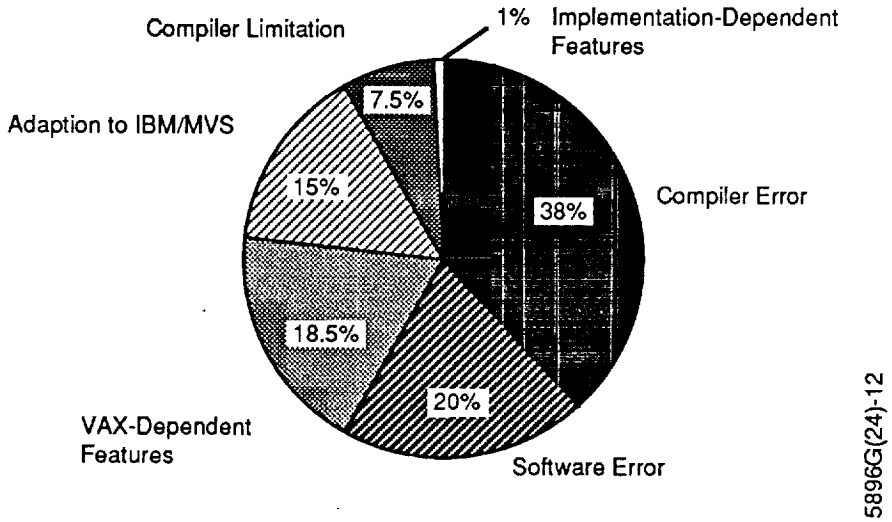


Figure 3-7. Effort Spent in Each Problem Area

features. As shown in the figure, more time was spent fixing unexpected problems, such as compiler errors and software errors, than on known problems, such as removing VAX-dependent features and adapting to the IBM/MVS environment.

3.4 SUMMARY RESULTS

Except for working around the problems with the Ada program library, the first three phases of the rehosting went smoothly. Transfer of source files and data files went without difficulty via the electronic file transfer utility. Compilation itself went rather smoothly, as 99 percent of the system was successfully compiled after correcting two components with size limitation errors. Linkage went extremely well; the system was linked during the first attempt. However, execution and testing were rather difficult due to errors with the software and the compiler.

The statistics collected during this experience are summarized below according to the phases in which they occurred.

- Phase 1: Source and Data File Transfer
  - Source files transferred: 540 (some files were not used on IBM)
  - Input data files converted to IBM format: 2
  - Input data files transferred: 35
  - Major problems: None
- Phase 2: Compilation
  - Components compiled: 564
  - New components added: 36 (6.4%, 1,970 LOC)
  - Components separated from the package body: 9 (1.6%)
  - Components changed: 102 (18%)
  - Average compilation speed: 1,295 lines/minute
  - Major problems: Library corruption, compiler limitation
- Phase 3: Linkage
  - Attempts made to successfully link: 1
  - Average bind/link speed: 64.20 CPU seconds
  - Major problems: None
- Phase 4: Execution
  - Number of attempts to successfully execute first test: 26
  - Major problems: Compiler errors, software errors

- Phase 5: Testing
  - Number of test cases executed: 10
  - Number of input files successfully read: 4 out of 5
  - Number of output files successfully written: 10 out of 10
  - Execution speed (2-minute simulation)
    - -- With Direct_IO only: 69.46 CPU seconds
    - -- With Text_IO only: 121.41 CPU seconds
    - -- With both IOs: 169.29 CPU seconds
    - -- Without any IO: 17.87 CPU seconds
  - Accuracy of results compared to VAX: Matched to 4-5 digits

    Major problems: Compiler error, file interface

SECTION 4—CONCLUSION

4.1 THE REHOST EFFORT WAS A SUCCESS

This particular porting effort was considered a success, since the majority of the problems encountered during the effort were due to the immaturity of the particular compiler employed and not to the Ada language itself. Transfer of source files and data files was made easy through the use of an electronic file transfer utility. Compilation of the Ada code was smooth: 99 percent of the system was successfully compiled after resolving the problems with size limitation. However, the Alsys Ada program library was immature and caused many problems during the compilation phase until the rehost team gained more experience. Appendix B describes some lessons learned during this phase. Execution and testing were rather difficult, due to compiler problems and the recompilation necessary for debugging.

4.2 ALSYS IBM ADA COMPILER WAS NOT YET MATURE

Corruption of the Ada program library was one of the major problems during rehosting. The library became corrupted when compilation terminated abnormally due to CPU timeout or disk space shortage. Occasionally, the library was deleted when a fatal error occurred during compilation. Another major problem was interfacing with Random_Access file (Direct_IO). The rehosted system was not able to read one of the input files correctly without any changes to the code. It was determined that the particular file type package specification had to be rewritten to include Record Representation Clause. The Alsys IBM Ada compiler automatically rearranges record components if no component clause is given. However, this was not attempted, since massive recompilations would have been required and it would have taken a fair amount of time to write Record Representation Clause for the record. It was later determined during unit testing that this fix would not eliminate the problem. Other compiler errors described earlier had a severe impact on the rehosting.

4.3 DEBUGGING AND RECOMPILATIONS WERE DIFFICULT AND COSTLY

Aside from the compiler problems, debugging was one of the most difficult and costly activities of the rehost. Debugging was difficult due to a lack of knowledge about the rehosted software and costly because of the time-consuming recompilations that were necessary. For instance, the Minor_Frame package had 72 subunits; if that package body was recompiled, all 72 subunits were automatically deleted and therefore had to be recompiled. This problem was most severe when a type package that was "withed" in other package specifications was recompiled, because all dependent package bodies and subunits were deleted. To solve

compiler errors and to update the software, several type packages had to be recompiled. Debugging caused several package bodies to be recompiled. Developing an Ada system is a high-cost proposition: in order to enforce the language rules, the compiler or compiling environment has to maintain information on each compilation unit in the Ada program library, which greatly increases the system workload. Therefore, Ada systems must be designed and implemented in a way that reduces the compilation cost. The following are some suggestions:

1. "With" in library units at the lowest unit (subunit) possible
2. Group a smaller number of functions into library packages rather than using one big general package
3. Make subunits instead of implementing within the package body

Other factors that made debugging more difficult were the lack of a debugging tool for the IBM and insufficient error messages generated by the rehosted software system. For instance, some exceptions were not handled properly at the lower level unit. The exception was propagated up to the main program and "irrecoverable error" was written out. Debugging would have been easier if GOESIM had written detailed error messages when an exception was raised at the lower level unit, before propagating it to the higher level unit.

4.4 VAX-DEPENDENT FEATURES CAUSED FEWER PROBLEMS THAN EXPECTED

The rehost team expected that the VAX-dependent features—Direct_Mixed_IO, predefined Math_Lib, hardware-oriented bit manipulation types and functions, etc.—would cause major problems. However, the bit manipulation in the Simulate Telemetry Data went smoothly, although it generated double-sized records due to the implementation of the Alsys compiler. As described earlier, nonstandard Direct_Mixed_IO was replaced with standard Direct_IO, and the rehost team successfully tested four of five input files. It was a surprise to find that the Alsys compiler did not provide a predefined Math_Library containing transcendental functions; some of the functions had to be implemented by the rehost team.

4.5 COMPARING ADA REHOSTING WITH OTHER LANGUAGES

It is not clear from the results of this rehost effort whether a system written in Ada is more portable than a system written in any other language. A similar system written in another language, like FORTRAN, should be rehosted for comparison. Nonetheless, porting this particular Ada system was not too difficult. The relative ease of rehosting is based on the software characteristics of the rehosted system. The rehosted system was a telemetry simulator (which was essentially data transformation) and required a simple, noninteractive user interface, which makes a

porting effort less difficult. A large interactive user interface is generally believed to be less portable because of hardware dependencies.

Although the SEL has never collected data on a software system rehost, the FDD has rehosted FORTRAN systems in the past. From this experience an attempt can be made to estimate the effort involved in rehosting a system like GOESIM that is developed in FORTRAN. Table 4-1 illustrates the SEL's estimated effort to rehost a FORTRAN telemetry simulator and compares it to the actual results from the GOESIM rehost. (The subjectivity of this comparison is acknowledged. However, in the absence of a similar empirical comparison to draw from, the authors felt that the inclusion of a subjective comparison would benefit the reader.)

Table 4-1. Comparative Effort to Rehost FORTRAN and Ada Telemetry Simulators

	FORTRAN Estimated	Ada Actual
Total Effort (Staff-days)	198 - 220	125*
Percent of Effort to Complete:		
Prepare Software	20 %	13 %
Compile	20 %	41 %
Link	5 %	2 %
Execute	10 %	14 %
Test	45 %	30 %

5896G(24)-13

*This number does not include effort spent on the report.

The FORTRAN estimates were derived in the following way:

1. The SEL has observed a ratio of approximately 3:1 between Ada and FORTRAN SLOC. (LOC in this case refers to the physical lines.)

2. The SEL also has an average FORTRAN productivity of 5 KSLOC per staff-year. Therefore, 6 to 7 staff-years would have been the expected time required to complete a comparable FORTRAN system.

3. Finally, in past FORTRAN rehostings, the SEL has documented a ratio of 6:1 for the number of years to develop a FORTRAN system compared to the years to rehost one. Therefore, the SEL would expect a rehost effort of 12 to 14 staff-months. However, this 6:1 ratio is based on 1984 technology. In 1989, FORTRAN compilers on the VAX and the IBM were more similar to each other than in 1984. Hence, the SEL would

expect a 25 percent improvement in the effort to rehost, and the rehost effort estimate would drop to 9 to 10 months.

The breakdown of the effort required for each phase is a result of the experience gained from past FORTRAN rehostings, updated to reflect the advantages (i.e., recently completed system to rehost, access to developers, documentation, etc.) found in this rehost.

4.6 SOME ADA FEATURES PROMOTE PORTABILITY

Aside from the software characteristics issue, it is clear from the study that some Ada features promote portability. During the rehost, the most appealing features were abstract data types, packages, and descriptive names. Use of user-defined data types was one of the features that made the rehost effort easier. For instance, instead of using predefined floating-point types, user-defined real types were used consistently throughout the system. For example

```
type REAL is digits 6

type REAL_LONG is digits 15
```

This was beneficial, because the compiler could choose the proper implementation.

The use of packages did not make the system more portable but aided in comprehending the physical structure of the system. In fact, use of packages increased compilation effort because of Ada visibility rules. Use of enumeration types and descriptive names for types, packages, subprograms, and objects made code more readable, and consequently less time was spent referring to documentation or other units, since code review was essential during debugging.

APPENDIX A—ADA COMPILER CHARACTERISTICS

DEC Ada Compiler Version V5 versus Alsys IBM Ada Compiler Version 3.6

	VAX (DEC)	IBM (Alsys)
Predefined Types		
SHORT_FLOAT	No	6 digit (32 bit)
FLOAT	6 digits (32 bit)	15 digit (64 bit)
LONG_FLOAT	9, 15 digit (64 bit)	18 digit (128 bit)
LONG_LONG_FLOAT	33 bit	No
SHORT_SHORT_INTEGER	8 bit	No
SHORT_INTEGER	16 bit	16 bit
INTEGER	32 bit	32 bit
Predefined Pragma		
CONTROLLED	Yes	No
INTERFACE	Assembler/FORTRAN*	Assembler
MEMORY_SIZE	Yes	No
OPTIMIZE	Yes	No
PACK*	Yes	No
STORAGE_UNIT	Yes	No
SYSTEM_NAME	Yes	No
Predefined Packages		
Direct_Mixed_IO *	Yes	No
Record_IO	No	Yes
Math_Lib*	Yes	No
ASCII_To_EBCDIC	No	Yes
EBCDIC_To_ASCII	No	Yes
Utilities		
Ada Program Library Facility	Yes	Yes
Symbolic Debugger	Yes	No

* Used in GOESIM on the VAX but not available on the IBM.

APPENDIX B—DEVELOPING ADA SYSTEMS FOR IBM/MVS

The following list contains some suggestions for developing Ada systems in an IBM/MVS environment using the Alsys IBM Ada Compiler Version 3.6:

1. Document the compilation order. This is essential to an Ada system.

2. Create a CLIST to compile the whole system as early as possible, since recompilation is often necessary.

3. Create an Ada Program Library with ample size to avoid redundant compilation effort due to space shortages. (An Ada system of 92 KSLOC with 564 components needed 25,000 blocks.)

4. Use another, smaller Program Library for unit testing. Units in the main Program Library can be referenced by the ACQUIRE command from the Ada Unit Manager. This will be a more efficient way of unit testing, since only the testing unit or the unit being tested is required to be in the other Program Library.

5. The main Program Library should be backed up routinely to restore a corrupted library.

6. The following workaround is recommended when generating Ada code to avoid some compiler problems:

   a. Code "A >= B" as "A > B or A = B" for floating-point number types

   b. When creating a record format for a random access file, all FLOAT items should be placed before items of type INTEGER

7. If an Ada system needs to read a random access file written by a non-Ada program, rewrite the file with the first record duplicated, since Direct_IO skips the first record of a non-Ada file.

APPENDIX C—QUESTIONNAIRE FOR GOESIM REHOSTING

Name: _____

1. What is your level of experience with the following?

	Ada	S/W * Development	VAX * Environment	IBM * Environment
None	☐	☐	☐	☐
Novice	☐	☐	☐	☐
Advanced	☐	☐	☐	☐
Expert	☐	☐	☐	☐

* Independent of Ada

2. Please estimate total effort in staff-days required to complete the following:

Preparing S/W on the IBM:	☐	Staff-days
Successful compilation:	☐	Staff-days
Successful linkage:	☐	Staff-days
Successful execution:	☐	Staff-days
Successful testing:	☐	Staff-days
Documentation – A short report	☐	Staff-days
Total:	☐	Staff-days

(Successful means no diagnostics/error for that step)

3. What will be the major area of problems in rehosting?
(Check one box for each area of problems.)

	Major	Medium	Minor	None
Anomalies with Alsys Compiler on the IBM	☐	☐	☐	☐
DEC VAX-specific Ada features used in the simulator	☐	☐	☐	☐
IBM/MVS environment for Ada (JCL, EBCDIC, data files, etc.)	☐	☐	☐	☐
Availability of S/W support library on IBM (e.g. Math_Lib)	☐	☐	☐	☐
Other: _____ (Specify)	☐	☐	☐	☐

5896G(23)-01

4. What will be the relative difficulty of each phase?
(Check one box for each phase.)

	Easy	Medium	Very Difficult
Preparing S/W on IBM (setting up files, etc.)	☐	☐	☐
Compilation	☐	☐	☐
Linking	☐	☐	☐
Testing and analysis	☐	☐	☐

5. What types of Ada problems would you expect to have?
(Check one box for each type.)

	Major	Medium	Minor	None
Interface (external I/O)	☐	☐	☐	☐
Interface (internal – module to module)	☐	☐	☐	☐
Computational	☐	☐	☐	☐
Initialization	☐	☐	☐	☐
Data typing	☐	☐	☐	☐
Implementation-dependent Ada features (Pragma, Attributes, Exception, etc.)	☐	☐	☐	☐
Variable names	☐	☐	☐	☐
Logic/control structure	☐	☐	☐	☐
Precision agreement (VAX to IBM)	☐	☐	☐	☐
Other: _____ (Specify)	☐	☐	☐	☐

6. Please estimate percentage of modules that will need changing. Changes include any changes other than commentary.

7. Please describe briefly any assumptions you made in the estimates.

5896G(23)-02

APPENDIX D—MONTHLY STATUS REPORTS

This appendix presents Monthly Status Reports submitted for the Ada rehosting study for the period from June 1989 through March 1990.

June 5, 1989

This memorandum is to give you current status of Rehosting of GOES Telemetry Simulator from VAX/VMS environment to the IBM/MVS environment.

o Received 24 questionnaires out of 31 Average estimated effort: 168 staff days	
o Project started	4/17/89
o Recoded 2 FORTRAN routines in Ada (6 routines were available from other source and modified)	4/25/89
o Set up Source Library (PANVALET)	5/02/89
o Coded and unit tested system dependent package Bit_OPS_System on the IBM	5/17/89
o Transferred GOESIM source files via BFX (was not successful in transferring files using tape due to incompatibility between files on the VAX and IBM)	5/22/89
o Compiled 37 units with 1 error which caused by Alsys Compiler restriction	5/26/89

Total effort expended to 6/2: 23 staff days

July 13, 1989

ATTACHMENT

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR
IN ADA FROM VAX TO IBM

1. Source and Data File Transfer :

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	545	-	-	545
Data Files	3	2	1	0

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	15	15	27	19 (70%)
Math_Lib	2	1	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	2	2	3	3

3. Compilation on IBM :

No. Components on IBM : 562
No. Component compiled : 354 (63%)
No. Errors : 2
No. Errors corrected : 2
No. Components affected by errors : 7

4. Linkage on IBM : None to date

5. Execution on IBM : None to date

6. Testing on IBM : None to date

7. Key Problems/Points :

* There are unexpected problems occurring in the Alsys compiler and Program Library Manager which cause Ada Program Library to be corrupted. Alsys has been contacted.

5396

July 13, 1989

* The size for data objects allocated in the static part of the static frame is limited to 4k bytes.

* There is no automatic recompilation provided. Obsolete units are deleted and must be recompiled individually.

* There is no mechanism for multiple file compilation.

* There is no bit level implementation for any of the representation clauses.

* The minimum storage size associated with a type is 16 bits. Function written in Assembly Language to extract and store lower 8 bits from the simulated telemetry word(16 bits on IBM) must be provided.

Total time spent to 6/30 : 40 staff days

August 7, 1989

ATTACHMENT

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

1. Source and Data File Transfer :

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	540	-	-	540
Data Files	3	2	1	0

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	19	19	27	27
Math_Lib	2	1	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	9	9	3	3
Ephemeris	3	3	-	-
Database_Types	-	-	13	13
Other	-	-	14	14

3. Compilation on IBM :
   No. Components on IBM : 553
   No. Components compiled: 544 (98%)
   No. Components changed : 59 (11%)
   No. New Components : 33 (1830LOC)
   No. Errors : 3
   No. Errors corrected : 3
   No. Components affected by errors: 8

4. Linkage on IBM : None to date

5. Execution on IBM : None to date

6. Testing on IBM : None to date

5896

August 7, 1989

7. Key Problems/Points :

* Problems occurring in the Alsys compiler and Program Library Manager (reported last month) have caused additional problems this month.

* Intermittant problem in unit testing (possible alsys compiler problem).

Total time spent to 8/04 : 57 staff days

ATTACHMENT

September 7, 1989

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

1. Source and Data File Transfer :

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	540	-	-	540
Data Files	3	2	2	0

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	19	19	27	27
Math_Lib	2	2	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	9	9	3	3
Ephemeris	3	3	3	3
Database_Types	-	-	13	13
Other	-	-	14	14

3. Compilation on IBM : (Completed)
   No. Components on IBM : 553
   No. Components compiled: 553
   No. Components changed : 62
   No. New Components : 33 (1830 LOC)
   No. Errors : 3
   No. Errors corrected : 3
   No. Components affected by errors: 11

4. Linkage on IBM : Successfully Linked (Completed)

5. Execution on IBM : Attempts Initiated 8/28
   6 Execution to Date
   All have abended

6. Testing on IBM : None to date

5896

September 7, 1989

7. Key Problems/Points :

* Unusual Abends in Attempting Execution:

- Return Code 4095 (Nothing explained in Manual)
  Increased Space Seemed to Resolve this

- Return Code 15 ('Spurious Error' - Unexpected Branch to Odd Address)

Total time spent to 9/01 : 71 staff days

5896

ATTACHMENT

October 5, 1989

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

1. Source and Data File Transfer :

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	540	-	-	540
Data Files	2	2	2	2
Namelist Files	33	-	-	33

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	19	19	27	27
Math_Lib	2	2	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	9	9	3	3
Ephemeris	3	3	3	3
Database_Types	-	-	16	16
Other	-	-	20	20

3. Compilation on IBM : (Completed)
   No. Components on IBM : 553
   No. Components compiled: 553
   No. Components changed : 71
   No. New Components : 38 (1830 LOC)
   No. Errors : 4
   No. Errors corrected : 4
   No. Components affected by errors: 12

4. Linkage on IBM : Successfully Linked (Completed)

5896

October 5, 1989

5. Execution on IBM : 17 Executions Attempted
None Successful - All Have Abended
See Key Problems

6. Testing on IBM : None to date

7. Key Problems/Points :

Problems	Solutions	Compiler Problem?
* "Unexpected Branch to Odd Address" Problem occurred in array aggregates	Replaced DVECTOR with DSCALAR when dimension is one	Yes
* "Constraint_Error" from Abstract_Calendar.Split	Corrected local variable type in Abstract_Calendar.Split	No
* "Constraint_Error" from Debug_Collector.Write	Properly handled "Constraint_Error"	No
* Failed to check "End_Of_File" when reading Namelist file	Modified to stop reading Namelist file when blank name returnd	Possible
* Failed to initialize Spacecraft Ephemeris	Being investigated	-

- Total time spent to 9/29 : 83 staff days

5896

ATTACHMENT

November 6, 1989

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

1. Source and Data File Transfer :

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	540	-	-	540
Data Files	2	2	2	2
Namelist Files	33	-	-	33

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	19	19	27	27
Math_Lib	2	2	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	9	9	3	3
Ephemeris	3	3	3	3
Database_Types	-	-	16	20
Other	-	-	20	28

3. Compilation on IBM : (Completed)
   No. Components on IBM : 561
   No. Components compiled: 561
   No. Components changed : 83
   No. New Components : 41 (1830 LOC)
   No. Errors : 7
   No. Errors corrected : 7
   No. Components affected by errors: 16

4. Linkage on IBM : Successfully Linked (Completed)

5896

November 6, 1989

5. Execution on IBM : Successfully Executed After 26 Attempts
See Key Problems

6. Testing on IBM : Started Test 1.1
Simulated Telemetry Output Does Not
Match with Output From the Vax

7. Key Problems/Points :

Problems	Solutions	Compiler Problem?
* Failed to initialize Spacecraft Ephemeris - Problem occurred due to incorrect evaluation of "IF A <= 0.0"	Rewrote IF statement	Yes
* Incorrectly raised "Numeric_Error" in INTEGER computation	Computed in REAL and converted back to INTEGER	Yes
* Failed to recompile subunit which has been previously compiled	Recompiled from the Body	Yes

Total time spent to 11/03/89 : 92 staff days

5896

ATTACHMENT

December 6, 1989

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

1. Source and Data File Transfer : (Completed)

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	540	-	-	540
Data Files	2	2	2	2
Namelist Files	33	-	-	33

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	19	19	27	27
Math_Lib	2	2	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	9	9	3	3
Ephemeris	3	3	3	3
Database_Types	-	-	16	22
Other	-	-	20	30

3. Compilation on IBM : (Completed)
   No. Components on IBM : 562
   No. Components compiled: 562
   No. Components changed : 87
   No. New Components : 42 (1830 LOC)
   No. Errors : 7
   No. Errors corrected : 7
   No. Components affected by errors: 16

4. Linkage on IBM : Successfully Linked (Completed)

5. Execution on IBM : Successfully Executed After 26 Attempts

5896

December 6, 1989

6. Testing on IBM : Completed 4 out of 10 selected test cases. 4 out of 5 output reports tested so far match Vax reports. Simulated Telemetry Reports partially matches Vax reports - being investigated

Input Files	Tested on the IBM?	Match Vax Output?
Namelist File	Yes	Yes
Attitude History File	No	-
Dynamics Engineering File	No	-
GOES Ephemeris File	No	-
SLP file	No	-
Output Reports		
Initialization Report	Yes	Yes
Telemetry Record Report	Yes	Yes
Simulated Telemetry Report	Yes	No (Partially)
Engineering Data Report	Yes	Yes
Proc. Eng. Data Report	Yes	Yes
Simulation Result Report	No	-
Bit Flip Report	No	-
Output Files		
Simulated Telemetry File	Yes	No (Partially)
Engineering Data File	Yes	Being verified
Proc. Eng. Data File	Yes	Being verified

7. Key Problems/Points :

Problems	Solutions	Compiler Problem?
Data_Error was raised when writing to Engineering Data File (Direct_IO)	Made same record length for Header, Index, and Data Record	No

8. Status

```
          Staff Days        20        40        60        80       100
                   ----|----|----|----|----|----|----|----|----|----|->
                            ^                      ^^         ^
Complete                    |                      ||         |
   Source Transfer ---------+                      ||         |
   Compilation     ---------------------------------+|         |
   Linkage         ----------------------------------+         |
   Execution       ---------------------------------------------+
```

Total time spent to 12/01/89 : 103 staff days

ATTACHMENT

January 5, 1990

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

1. Source and Data File Transfer : (Completed)

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	540	-	-	540
Data Files	2	2	2	2
Namelist Files	33	-	-	33

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	19	19	27	27
Math_Lib	2	2	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	9	9	3	3
Ephemeris	3	3	3	3
Database_Types	-	-	16	23
Other	-	-	20	34

3. Compilation on IBM : (Completed)
   No. Components on IBM : 562
   No. Components compiled: 562
   No. Components changed : 92
   No. New Components : 42 (1830 LOC)
   No. Errors : 7
   No. Errors corrected : 7
   No. Components affected by errors: 16

4. Linkage on IBM : Successfully Linked (Completed)

5. Execution on IBM : Successfully Executed After 26 Attempts

5896

January 5, 1990

6. Testing on IBM : - Completed 7 out of 10 selected test cases
- Successfully read SLP file
- All output looks good except DSS data
- Problem with Ephemeris File - getting Incorrect_Coord_Exception

Input Files	Tested on the IBM?	Match Vax Output?
Namelist File	Yes	Yes
Attitude History File	No	-
Dynamics Engineering File	No	-
GOES Ephemeris File	Yes	No
SLP file	Yes	Yes
Output Reports		
Initialization Report	Yes	Yes
Telemetry Record Report	Yes	Yes
Simulated Telemetry Report	Yes	Yes
Engineering Data Report	Yes	Yes*(Except DSS data)
Proc. Eng. Data Report	Yes	Yes*
Simulation Result Report	Yes	Yes*
Bit Flip Report	No	-
Output Files		
Simulated Telemetry File	Yes	Yes
Engineering Data File	Yes	Yes*
Proc. Eng. Data File	Yes	Yes*

7. Key Problems/Points :

Problems	Solutions	Compiler Problem?
* -Incorrect results of telemetry record(POCC) - Order of MSB and LSB was reversed in the type BIT_ARRAY_8 on the IBM. BIT_ARRAY_8 is predefined VAX Hardware-oriented type	Modified two routines to incorporate the difference in the type definition	No
* Name_Error was raised when opening simulation results report file	Modified one routine to correctly extend the file name	No

5896

January 5, 1990

* INCORRECT_COORD_SYSTEM is raised when initializing Ephemeris variables	being investigated	?
* Wrong DSS data	being investigated	?

8. Status

```
                                                                  Average
                                                                  Estimate
                                                                   (168)
                                                                     |
                                                                     |
   Staff Days            30        60        90       120       150    v
              ----|----|----|----|----|----|----|----|----|----|----|----|---->
                   ^                ^^     ^       ^
Complete           |                ||     |       |
   Source Transfer --+               ||     |       |
   Compilation     -------------------+|     |       |
   Linkage         --------------------+     |       |
   Execution       --------------------------+       |
   7 tests         ----------------------------------+
```

Total time spent to 12/29/89 : 113 staff days

9. Performance

		(NAS 8063) IBM	(VAX 8810) VAX
Speed of CPU (MIPS)		8.65	6.36
Compilation (Lines/Min)	:	1295	3618
Binding/Linking (CPU Sec)	:	64.20	12.84 *
Execution (CPU Sec)			
2 minute simulation			
with Direct_IO only	:	69.46	89.68
with Text_IO only	:	121.41	63.87
with both IOs	:	169.29	159.00
without any IO	:	17.87	34.67

* Computed based on the previous statistics that it takes 5 times more CPU times on the IBM than on the VAX

5896

ATTACHMENT

February 7, 1990

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

1. Source and Data File Transfer : (Completed)

	On VAX	Needs Data Conversion	Data File Converted	Transferred to IBM
Source Files	540	-	-	540
Data Files	2	2	2	2
Namelist Files	33	-	-	33

2. Function Requiring Rewrite or Change (VAX dependent) :

Function	# of New Components Needed	# of New Components Completed	# of Components Need to be Changed	# of Components Changed
Direct_IO	19	19	27	27
Math_Lib	2	2	-	-
Bit_Ops_System	-	-	2	2
Conversion of Fortran to Ada	9	9	3	3
Ephemeris	4	4	3	3
Database_Types	-	-	23	23
Other	11	11	40	40

3. Compilation on IBM : (Completed)
   No. Components on IBM : 564
   No. Components compiled: 564
   No. Components changed : 98
   No. New Components : 45 (1970 LOC)
   No. Errors : 7
   No. Errors corrected : 7
   No. Components affected by errors: 16

4. Linkage on IBM : Successfully Linked (Completed)

5. Execution on IBM : Successfully Executed After 26 Attempts

5896

February 7, 1990

6. Testing on IBM : - Completed all 10 selected test cases
- Successfully read GOES Ephemeris File
- Have problems with Dynamics Engineering File

Input Files	Tested on the IBM?	Match Vax Output?
Namelist File	Yes	Yes
Attitude History File	Yes	Yes
Dynamics Engineering File	Yes	No
GOES Ephemeris File	Yes	Yes
SLP file	Yes	Yes
Output Reports		
Initialization Report	Yes	Yes
Telemetry Record Report	Yes	Yes
Simulated Telemetry Report	Yes	Yes
Engineering Data Report	Yes	Yes
Proc. Eng. Data Report	Yes	Yes
Simulation Result Report	Yes	Yes
Bit Flip Report	Yes	Partially (Due to RANDOM number)
Output Files		
Simulated Telemetry File	Yes	Yes
Engineering Data File	Yes	Yes
Proc. Eng. Data File	Yes	Yes

7. Key Problems/Points :

Problems	Solutions	Compiler Problem?
* INCORRECT_COORD_SYSTEM is raised when reading IBM resident GOES Ephemeris File	Replaced STRING with EBCDIC.STRING	No
* Incorrectly Reads Dynamics Engineering File Header and Data records - Have a problem with reading records starting with INTEGER or SHORT_FLOAT(4 bytes) followed by FLOAT(8 bytes)	-	Yes

5896

8. Status

February 7, 1990

```
                                                                                  Average
                                                                                  Estimate
                                                                                   (168)
                                                                                     |
                                                                                     |
   Staff Days  0         30        60        90        120       150       v
               |----|----|----|----|----|----|----|----|----|----|----|----|----|---->
                         ^                ^^    ^                   ^
Complete                 |                ||    |                   |
   Source Transfer  --+                   ||    |                   |
   Compilation      -------------------+|    |                   |
   Linkage          --------------------+    |                   |
   Execution        --------------------------+                   |
   10 tests         ----------------------------------------------+
```

Total time spent to 2/2/90 : 128 staff days

5896

ATTACHMENT March 6, 1990

STATUS OF REHOSTING GOES TELEMETRY SIMULATOR IN ADA FROM VAX TO IBM

A. Source and Data File Transfer

```
Source files transferred            : 540 (some files were not used)
Data files converted to IBM format :   2
Input data files transferred        :  35
Major problems                      : NONE
```

B. Compilation on IBM

```
# of components compiled  :  564
# of components changed   :  102
# of new components added :   45 (1970 LOC)
Average compilation speed : 1295 lines/minute
Major problem             : library corruption, compiler limitation
```

C. Linkage

```
Attempts made to successfully link : 1
Average bind/link speed            : 64.2 cpu seconds
Major problems                     : NONE
```

C. Execution

```
Number of attempts made to
  successfully execute TEST 1    : 26
Number of compiler problem found :  7
Major problems                   : compiler errors, software errors
```

E. Testing

```
# of test cases executed                : 10
# of input files successfully read      : 4 out of 5
# of output files successfully written : 10 out of 10
Execution speed in cpu seconds (2 minute simulation) :
          with Direct_IO only :  69.46
          with Text_IO only   : 121.41
          with both IO        : 169.29
          with any IO         :  17.87
Accuracy of results compared to VAX : matched to 4 - 5 digits
Major problems : compiler error, file interface
```


5896

March 6, 1990

F. Compiler problems found during rehosting

1. CPU timeout caused Ada Program Library to be corrupted
2. Incorrectly evaluated "if A >= 1.0"
3. Incorrectly raised "Numeric_Error" in INTEGER computation
4. Raised "Spurious_Error" due to branch to odd address
5. Skipped first record when reading direct access file written by non-Ada program
6. Incorrectly read direct access file starting with INTEGER followed by FLOAT even with record representation clause

G. Staff days to complete each phase

```
                                                                  Average
                                                                  Estimate
                                                                   (168)
                                                                     |
                                                                     |
   Staff Days  0         30        60        90       120       150    v
               |----|----|----|----|----|----|----|----|----|----|----|----|---->
                     ^                ^^     ^            ^ ^
Complete             |                ||     |            | |
   Source Transfer --+                ||     |            | |
   Compilation     -------------------+|     |            | |
   Linkage         --------------------+     |            | |
   Execution       --------------------------+            | |
   IO tests        ----------------------------------------+ |
   Draft Report    ------------------------------------------+
```

Congratulations to the following people who gave the best estimates on the rehosting effort.

Actual effort including report draft : 133 staff days

F. McGarry/552	130
E. Booth/CSC	130
S. Watson/552	137
D. Wood/CSC	144

5896

GLOSSARY

ASCII	American Standard Code for Information Interchange
CPU	central processing unit
DEC	Digital Equipment Corporation
EBCDIC	Extended Binary Coded Decimal Interchange Code
FDD	Flight Dynamics Division
GOES	Geostationary Operational Environmental Satellite
GOESIM	GOES telemetry simulator
GSFC	Goddard Space Flight Center
I/O	input/output
KSLOC	thousand SLOC
NAS	National Advanced Systems
NASA	National Aeronautics and Space Administration
SEL	Software Engineering Laboratory
SLOC	source lines of code

REFERENCES

Branyan, E. R., and L. Brown-Aepli, "Application of Ada Software Development Environments to the Production of Time Critical Software. A Preliminary Report," in *Proceedings of the Twelfth Annual Software Engineering Workshop*. Greenbelt, Maryland: National Aeronautics and Space Administration, Goddard Space Flight Center, 1987

Engle, C., Jr., *The Myth of Portability in Ada*. Pittsburgh, Pennsylvania: Software Engineering Institute, Carnegie Mellon University, 1988

STANDARD BIBLIOGRAPHY OF SEL LITERATURE

The technical papers, memorandums, and documents listed in this bibliography are organized into two groups. The first group is composed of documents issued by the Software Engineering Laboratory (SEL) during its research and development activities. The second group includes materials that were published elsewhere but pertain to SEL activities.

SEL-ORIGINATED DOCUMENTS

SEL-76-001, *Proceedings From the First Summer Software Engineering Workshop*, August 1976

SEL-77-002, *Proceedings From the Second Summer Software Engineering Workshop*, September 1977

SEL-77-004, *A Demonstration of AXES for NAVPAK*, M. Hamilton and S. Zeldin, September 1977

SEL-77-005, *GSFC NAVPAK Design Specifications Languages Study*, P. A. Scheffer and C. E. Velez, October 1977

SEL-78-005, *Proceedings From the Third Summer Software Engineering Workshop*, September 1978

SEL-78-006, *GSFC Software Engineering Research Requirements Analysis Study*, P. A. Scheffer and C. E. Velez, November 1978

SEL-78-007, *Applicability of the Rayleigh Curve to the SEL Environment*, T. E. Mapp, December 1978

SEL-78-302, *FORTRAN Static Source Code Analyzer Program (SAP) User's Guide (Revision 3)*, W. J. Decker and W. A. Taylor, July 1986

SEL-79-002, *The Software Engineering Laboratory: Relationship Equations*, K. Freburger and V. R. Basili, May 1979

SEL-79-003, *Common Software Module Repository (CSMR) System Description and User's Guide*, C. E. Goorevich, A. L. Green, and S. R. Waligora, August 1979

SEL-79-004, *Evaluation of the Caine, Farber, and Gordon Program Design Language (PDL) in the Goddard Space Flight Center (GSFC) Code 580 Software Design Environment*, C. E. Goorevich, A. L. Green, and W. J. Decker, September 1979

SEL-79-005, *Proceedings From the Fourth Summer Software Engineering Workshop*, November 1979

SEL-80-002, *Multi-Level Expression Design Language-Requirement Level (MEDL-R) System Evaluation*, W. J. Decker and C. E. Goorevich, May 1980

SEL-80-003, *Multimission Modular Spacecraft Ground Support Software System (MMS/GSSS) State-of-the-Art Computer Systems/Compatibility Study*, T. Welden, M. McClellan, and P. Liebertz, May 1980

SEL-80-005, *A Study of the Musa Reliability Model*, A. M. Miller, November 1980

SEL-80-006, *Proceedings From the Fifth Annual Software Engineering Workshop*, November 1980

SEL-80-007, *An Appraisal of Selected Cost/Resource Estimation Models for Software Systems*, J. F. Cook and F. E. McGarry, December 1980

SEL-81-008, *Cost and Reliability Estimation Models (CAREM) User's Guide*, J. F. Cook and E. Edwards, February 1981

SEL-81-009, *Software Engineering Laboratory Programmer Workbench Phase 1 Evaluation*, W. J. Decker and F. E. McGarry, March 1981

SEL-81-011, *Evaluating Software Development by Analysis of Change Data*, D. M. Weiss, November 1981

SEL-81-012, *The Rayleigh Curve as a Model for Effort Distribution Over the Life of Medium Scale Software Systems*, G. O. Picasso, December 1981

SEL-81-013, *Proceedings From the Sixth Annual Software Engineering Workshop*, December 1981

SEL-81-014, *Automated Collection of Software Engineering Data in the Software Engineering Laboratory (SEL)*, A. L. Green, W. J. Decker, and F. E. McGarry, September 1981

SEL-81-101, *Guide to Data Collection*, V. E. Church, D. N. Card, F. E. McGarry, et al., August 1982

SEL-81-104, *The Software Engineering Laboratory*, D. N. Card, F. E. McGarry, G. Page, et al., February 1982

SEL-81-107, *Software Engineering Laboratory (SEL) Compendium of Tools*, W. J. Decker, W. A. Taylor, and E. J. Smith, February 1982

SEL-81-110, *Evaluation of an Independent Verification and Validation (IV&V) Methodology for Flight Dynamics*, G. Page, F. E. McGarry, and D. N. Card, June 1985

SEL-81-205, *Recommended Approach to Software Development*, F. E. McGarry, G. Page, S. Eslinger, et al., April 1983

SEL-82-001, *Evaluation of Management Measures of Software Development*, G. Page, D. N. Card, and F. E. McGarry, September 1982, vols. 1 and 2

SEL-82-004, *Collected Software Engineering Papers: Volume I*, July 1982

SEL-82-007, *Proceedings From the Seventh Annual Software Engineering Workshop*, December 1982

SEL-82-008, *Evaluating Software Development by Analysis of Changes: The Data From the Software Engineering Laboratory*, V. R. Basili and D. M. Weiss, December 1982

SEL-82-102, *FORTRAN Static Source Code Analyzer Program (SAP) System Description (Revision 1)*, W. A. Taylor and W. J. Decker, April 1985

SEL-82-105, *Glossary of Software Engineering Laboratory Terms*, T. A. Babst, F. E. McGarry, and M. G. Rohleder, October 1983

SEL-82-806, *Annotated Bibliography of Software Engineering Laboratory Literature*, M. Buhler and J. Valett, November 1989

SEL-83-001, *An Approach to Software Cost Estimation*, F. E. McGarry, G. Page, D. N. Card, et al., February 1984

SEL-83-002, *Measures and Metrics for Software Development*, D. N. Card, F. E. McGarry, G. Page, et al., March 1984

SEL-83-003, *Collected Software Engineering Papers: Volume II*, November 1983

SEL-83-006, *Monitoring Software Development Through Dynamic Variables*, C. W. Doerflinger, November 1983

SEL-83-007, *Proceedings From the Eighth Annual Software Engineering Workshop*, November 1983

SEL-83-106, *Monitoring Software Development Through Dynamic Variables (Revision 1)*, C. W. Doerflinger, November 1989

SEL-84-001, *Manager's Handbook for Software Development*, W. W. Agresti, F. E. McGarry, D. N. Card, et al., April 1984

SEL-84-003, *Investigation of Specification Measures for the Software Engineering Laboratory (SEL)*, W. W. Agresti, V. E. Church, and F. E. McGarry, December 1984

SEL-84-004, *Proceedings From the Ninth Annual Software Engineering Workshop*, November 1984

SEL-85-001, *A Comparison of Software Verification Techniques*, D. N. Card, R. W. Selby, Jr., F. E. McGarry, et al., April 1985

SEL-85-002, *Ada Training Evaluation and Recommendations From the Gamma Ray Observatory Ada Development Team*, R. Murphy and M. Stark, October 1985

SEL-85-003, *Collected Software Engineering Papers: Volume III*, November 1985

SEL-85-004, *Evaluations of Software Technologies: Testing, CLEANROOM, and Metrics*, R. W. Selby, Jr., May 1985

SEL-85-005, *Software Verification and Testing*, D. N. Card, C. Antle, and E. Edwards, December 1985

SEL-85-006, *Proceedings From the Tenth Annual Software Engineering Workshop*, December 1985

SEL-86-001, *Programmer's Handbook for Flight Dynamics Software Development*, R. Wood and E. Edwards, March 1986

SEL-86-002, *General Object-Oriented Software Development*, E. Seidewitz and M. Stark, August 1986

SEL-86-003, *Flight Dynamics System Software Development Environment Tutorial*, J. Buell and P. Myers, July 1986

SEL-86-004, *Collected Software Engineering Papers: Volume IV*, November 1986

SEL-86-005, *Measuring Software Design*, D. N. Card, October 1986

SEL-86-006, *Proceedings From the Eleventh Annual Software Engineering Workshop*, December 1986

SEL-87-001, *Product Assurance Policies and Procedures for Flight Dynamics Software Development*, S. Perry et al., March 1987

SEL-87-002, *Ada Style Guide (Version 1.1)*, E. Seidewitz et al., May 1987

SEL-87-003, *Guidelines for Applying the Composite Specification Model (CSM)*, W. W. Agresti, June 1987

SEL-87-004, *Assessing the Ada Design Process and Its Implications: A Case Study*, S. Godfrey, C. Brophy, et al., July 1987

SEL-87-008, *Data Collection Procedures for the Rehosted SEL Database*, G. Heller, October 1987

SEL-87-009, *Collected Software Engineering Papers: Volume V*, S. DeLong, November 1987

SEL-87-010, *Proceedings From the Twelfth Annual Software Engineering Workshop*, December 1987

SEL-88-001, *System Testing of a Production Ada Project: The GRODY Study*, J. Seigle, L. Esker, and Y. Shi, November 1988

SEL-88-002, *Collected Software Engineering Papers: Volume VI*, November 1988

SEL-88-003, *Evolution of Ada Technology in the Flight Dynamics Area: Design Phase Analysis*, K. Quimby and L. Esker, December 1988

SEL-88-004, *Proceedings of the Thirteenth Annual Software Engineering Workshop*, November 1988

SEL-88-005, *Proceedings of the First NASA Ada User's Symposium*, December 1988

SEL-89-002, *Implementation of a Production Ada Project: The GRODY Study*, S. Godfrey and C. Brophy, September 1989

SEL-89-003, *Software Management Environment (SME) Concepts and Architecture*, W. Decker and J. Valett, August 1989

SEL-89-004, *Evolution of Ada Technology in the Flight Dynamics Area: Implementation/Testing Phase Analysis*, K. Quimby, L. Esker, L. Smith, M. Stark, and F. McGarry, November 1989

SEL-89-005, *Lessons Learned in the Transition to Ada From FORTRAN at NASA/ Goddard*, C. Brophy, November 1989

SEL-89-006, *Collected Software Engineering Papers: Volume VII*, November 1989

SEL-89-007, *Proceedings of the Fourteenth Annual Software Engineering Workshop*, November 1989

SEL-89-008, *Proceedings of the Second NASA Ada Users' Symposium*, November 1989

SEL-89-101, *Software Engineering Laboratory (SEL) Database Organization and User's Guide (Revision 1)*, M. So, G. Heller, S. Steinberg, K. Pumphrey, and D. Spiegel, February 1990

SEL-90-001, *Database Access Manager for the Software Engineering Laboratory (DAMSEL) User's Guide*, M. Buhler and K. Pumphrey, March 1990

SEL-90-002, *The Cleanroom Case Study in the Software Engineering Laboratory: Project Description and Early Analysis*, S. Green et al., March 1990

SEL-90-003, *A Study of the Portability of an Ada System in the Software Engineering Laboratory (SEL)*, L. O. Jun and S. R. Valett, June 1990

SEL-RELATED LITERATURE

4Agresti, W. W., V. E. Church, D. N. Card, and P. L. Lo, "Designing With Ada for Satellite Simulation: A Case Study," *Proceedings of the First International Symposium on Ada for the NASA Space Station*, June 1986

2Agresti, W. W., F. E. McGarry, D. N. Card, et al., "Measuring Software Technology," *Program Transformation and Programming Environments*. New York: Springer-Verlag, 1984

1Bailey, J. W., and V. R. Basili, "A Meta-Model for Software Development Resource Expenditures," *Proceedings of the Fifth International Conference on Software Engineering*. New York: IEEE Computer Society Press, 1981

7Basili, V. R., *Maintenance = Reuse-Oriented Software Development*, University of Maryland, Technical Report TR-2244, May 1989

1Basili, V. R., "Models and Metrics for Software Management and Engineering," *ASME Advances in Computer Technology*, January 1980, vol. 1

7Basili, V. R., *Software Development: A Paradigm for the Future*, University of Maryland, Technical Report TR-2263, June 1989

Basili, V. R., *Tutorial on Models and Metrics for Software Management and Engineering*. New York: IEEE Computer Society Press, 1980 (also designated SEL-80-008)

3Basili, V. R., "Quantitative Evaluation of Software Methodology," *Proceedings of the First Pan-Pacific Computer Conference, September 1985*

1Basili, V. R., and J. Beane, "Can the Parr Curve Help With Manpower Distribution and Resource Estimation Problems?," *Journal of Systems and Software*, February 1981, vol. 2, no. 1

1Basili, V. R., and K. Freburger, "Programming Measurement and Estimation in the Software Engineering Laboratory," *Journal of Systems and Software*, February 1981, vol. 2, no. 1

3Basili, V. R., and N. M. Panlilio-Yap, "Finding Relationships Between Effort and Other Variables in the SEL," *Proceedings of the International Computer Software and Applications Conference*, October 1985

4Basili, V. R., and D. Patnaik, *A Study on Fault Prediction and Reliability Assessment in the SEL Environment*, University of Maryland, Technical Report TR-1699, August 1986

2Basili, V. R., and B. T. Perricone, "Software Errors and Complexity: An Empirical Investigation," *Communications of the ACM*, January 1984, vol. 27, no. 1

1Basili, V. R., and T. Phillips, "Evaluating and Comparing Software Metrics in the Software Engineering Laboratory," *Proceedings of the ACM SIGMETRICS Symposium/Workshop: Quality Metrics*, March 1981

Basili, V. R., and J. Ramsey, *Structural Coverage of Functional Testing*, University of Maryland, Technical Report TR-1442, September 1984

3Basili, V. R., and C. L. Ramsey, "ARROWSMITH-P—A Prototype Expert System for Software Engineering Management," *Proceedings of the IEEE/MITRE Expert Systems in Government Symposium*, October 1985

Basili, V. R., and R. Reiter, "Evaluating Automatable Measures for Software Development," *Proceedings of the Workshop on Quantitative Software Models for Reliability, Complexity, and Cost. New York: IEEE Computer Society Press*, 1979

5Basili, V., and H. D. Rombach, "Tailoring the Software Process to Project Goals and Environments," *Proceedings of the 9th International Conference on Software Engineering*, March 1987

5Basili, V., and H. D. Rombach, "T A M E: Tailoring an Ada Measurement Environment," *Proceedings of the Joint Ada Conference*, March 1987

5Basili, V., and H. D. Rombach, "T A M E: Integrating Measurement Into Software Environments," University of Maryland, Technical Report TR-1764, June 1987

6Basili, V. R., and H. D. Rombach, "The TAME Project: Towards Improvement-Oriented Software Environments," *IEEE Transactions on Software Engineering*, June 1988

7Basili, V. R., and H. D. Rombach, *Towards A Comprehensive Framework for Reuse: A Reuse-Enabling Software Evolution Environment*, University of Maryland, Technical Report TR-2158, December 1988

2Basili, V. R., R. W. Selby, Jr., and T. Phillips, "Metric Analysis and Data Validation Across FORTRAN Projects," *IEEE Transactions on Software Engineering*, November 1983

3Basili, V. R., and R. W. Selby, Jr., "Calculation and Use of an Environments's Characteristic Software Metric Set," *Proceedings of the Eighth International Conference on Software Engineering. New York: IEEE Computer Society Press*, 1985

Basili, V. R., and R. W. Selby, Jr., *Comparing the Effectiveness of Software Testing Strategies*, University of Maryland, Technical Report TR-1501, May 1985

3Basili, V. R., and R. W. Selby, Jr., "Four Applications of a Software Data Collection and Analysis Methodology," *Proceedings of the NATO Advanced Study Institute*, August 1985

4Basili, V. R., R. W. Selby, Jr., and D. H. Hutchens, "Experimentation in Software Engineering," *IEEE Transactions on Software Engineering*, July 1986

5Basili, V. and R. Selby, Jr., "Comparing the Effectiveness of Software Testing Strategies," *IEEE Transactions on Software Engineering*, December 1987

[2]Basili, V. R., and D. M. Weiss, *A Methodology for Collecting Valid Software Engineering Data*, University of Maryland, Technical Report TR-1235, December 1982

[3]Basili, V. R., and D. M. Weiss, "A Methodology for Collecting Valid Software Engineering Data," *IEEE Transactions on Software Engineering*, November 1984

[1]Basili, V. R., and M. V. Zelkowitz, "The Software Engineering Laboratory: Objectives," *Proceedings of the Fifteenth Annual Conference on Computer Personnel Research*, August 1977

Basili, V. R., and M. V. Zelkowitz, "Designing a Software Measurement Experiment," *Proceedings of the Software Life Cycle Management Workshop*, September 1977

[1]Basili, V. R., and M. V. Zelkowitz, "Operation of the Software Engineering Laboratory," *Proceedings of the Second Software Life Cycle Management Workshop*, August 1978

[1]Basili, V. R., and M. V. Zelkowitz, "Measuring Software Development Characteristics in the Local Environment," *Computers and Structures, August 1978, vol. 10*

Basili, V. R., and M. V. Zelkowitz, "Analyzing Medium Scale Software Development," *Proceedings of the Third International Conference on Software Engineering*. New York: IEEE Computer Society Press, 1978

[5]Brophy, C., W. Agresti, and V. Basili, "Lessons Learned in Use of Ada-Oriented Design Methods," *Proceedings of the Joint Ada Conference*, March 1987

[6]Brophy, C. E., S. Godfrey, W. W. Agresti, and V. R. Basili, "Lessons Learned in the Implementation Phase of a Large Ada Project," *Proceedings of the Washington Ada Technical Conference*, March 1988

[2]Card, D. N., "Early Estimation of Resource Expenditures and Program Size," Computer Sciences Corporation, Technical Memorandum, June 1982

[2]Card, D. N., "Comparison of Regression Modeling Techniques for Resource Estimation," Computer Sciences Corporation, Technical Memorandum, November 1982

[3]Card, D. N., "A Software Technology Evaluation Program," *Annais do XVIII Congresso Nacional de Informatica*, October 1985

[5]Card, D., and W. Agresti, "Resolving the Software Science Anomaly," *The Journal of Systems and Software*, 1987

[6]Card, D. N., and W. Agresti, "Measuring Software Design Complexity," *The Journal of Systems and Software*, June 1988

Card, D. N., V. E. Church, W. W. Agresti, and Q. L. Jordan, "A Software Engineering View of Flight Dynamics Analysis System," Parts I and II, Computer Sciences Corporation, Technical Memorandum, February 1984

4Card, D. N., V. E. Church, and W. W. Agresti, "An Empirical Study of Software Design Practices," *IEEE Transactions on Software Engineering*, February 1986

Card, D. N., Q. L. Jordan, and V. E. Church, "Characteristics of FORTRAN Modules," Computer Sciences Corporation, Technical Memorandum, June 1984

5Card, D., F. McGarry, and G. Page, "Evaluating Software Engineering Technologies," *IEEE Transactions on Software Engineering*, July 1987

3Card, D. N., G. T. Page, and F. E. McGarry, "Criteria for Software Modularization," *Proceedings of the Eighth International Conference on Software Engineering*. New York: IEEE Computer Society Press, 1985

1Chen, E., and M. V. Zelkowitz, "Use of Cluster Analysis To Evaluate Software Engineering Methodologies," *Proceedings of the Fifth International Conference on Software Engineering*. New York: IEEE Computer Society Press, 1981

4Church, V. E., D. N. Card, W. W. Agresti, and Q. L. Jordan, "An Approach for Assessing Software Prototypes," *ACM Software Engineering Notes*, July 1986

2Doerflinger, C. W., and V. R. Basili, "Monitoring Software Development Through Dynamic Variables," *Proceedings of the Seventh International Computer Software and Applications Conference*. New York: IEEE Computer Society Press, 1983

5Doubleday, D., "ASAP: An Ada Static Source Code Analyzer Program," University of Maryland, Technical Report TR-1895, August 1987 (NOTE: 100 pages long)

6Godfrey, S., and C. Brophy, "Experiences in the Implementation of a Large Ada Project," *Proceedings of the 1988 Washington Ada Symposium*, June 1988

Hamilton, M., and S. Zeldin, *A Demonstration of AXES for NAVPAK*, Higher Order Software, Inc., TR-9, September 1977 (also designated SEL-77-005)

Jeffery, D. R., and V. Basili, *Characterizing Resource Data: A Model for Logical Association of Software Data*, University of Maryland, Technical Report TR-1848, May 1987

6Jeffery, D. R., and V. R. Basili, "Validating the TAME Resource Data Model," *Proceedings of the Tenth International Conference on Software Engineering*, April 1988

5Mark, L., and H. D. Rombach, *A Meta Information Base for Software Engineering*, University of Maryland, Technical Report TR-1765, July 1987

6Mark, L., and H. D. Rombach, "Generating Customized Software Engineering Information Bases From Software Process and Product Specifications," *Proceedings of the 22nd Annual Hawaii International Conference on System Sciences*, January 1989

5McGarry, F., and W. Agresti, "Measuring Ada for Software Development in the Software Engineering Laboratory (SEL)," *Proceedings of the 21st Annual Hawaii International Conference on System Sciences*, January 1988

7McGarry, F., L. Esker, and K. Quimby, "Evolution of Ada Technology in a Production Software Environment," *Proceedings of the Sixth Washington Ada Symposium (WADAS)*, June 1989

3McGarry, F. E., J. Valett, and D. Hall, "Measuring the Impact of Computer Resource Quality on the Software Development Process and Product," *Proceedings of the Hawaiian International Conference on System Sciences*, January 1985

National Aeronautics and Space Administration (NASA), *NASA Software Research Technology Workshop (Proceedings)*, March 1980

3Page, G., F. E. McGarry, and D. N. Card, "A Practical Experience With Independent Verification and Validation," *Proceedings of the Eighth International Computer Software and Applications Conference*, November 1984

5Ramsey, C., and V. R. Basili, *An Evaluation of Expert Systems for Software Engineering Management*, University of Maryland, Technical Report TR-1708, September 1986

3Ramsey, J., and V. R. Basili, "Analyzing the Test Process Using Structural Coverage," *Proceedings of the Eighth International Conference on Software Engineering*. New York: IEEE Computer Society Press, 1985

5Rombach, H. D., "A Controlled Experiment on the Impact of Software Structure on Maintainability," *IEEE Transactions on Software Engineering*, March 1987

6Rombach, H. D., and V. R. Basili, "Quantitative Assessment of Maintenance: An Industrial Case Study," *Proceedings From the Conference on Software Maintenance*, September 1987

6Rombach, H. D., and L. Mark, "Software Process and Product Specifications: A Basis for Generating Customized SE Information Bases," *Proceedings of the 22nd Annual Hawaii International Conference on System Sciences*, January 1989

7*Rombach, H. D., and B. T. Ulery*, Establishing a Measurement Based Maintenance Improvement Program: Lessons Learned in the SEL, University of Maryland, Technical Report TR-2252, May 1989

5Seidewitz, E., "General Object-Oriented Software Development: Background and Experience," *Proceedings of the 21st Hawaii International Conference on System Sciences*, January 1988

6Seidewitz, E., "General Object-Oriented Software Development with Ada: A Life Cycle Approach," *Proceedings of the CASE Technology Conference*, April 1988

6Seidewitz, E., "Object-Oriented Programming in Smalltalk and Ada," *Proceedings of the 1987 Conference on Object-Oriented Programming Systems, Languages, and Applications*, October 1987

[4]Seidewitz, E., and M. Stark, "Towards a General Object-Oriented Software Development Methodology," *Proceedings of the First International Symposium on Ada for the NASA Space Station*, June 1986

[7]Stark, M. E. and E. W. Booth, "Using Ada to Maximize Verbatim Software Reuse," *Proceedings of TRI-Ada 1989*, October 1989

Stark, M., and E. Seidewitz, "Towards a General Object-Oriented Ada Lifecycle," *Proceedings of the Joint Ada Conference*, March 1987

[7]Sunazuka. T., and V. R. Basili, *Integrating Automated Support for a Software Management Cycle Into the TAME System*, University of Maryland, Technical Report TR-2289, July 1989

Turner, C., and G. Caron, *A Comparison of RADC and NASA/SEL Software Development Data, Data and Analysis Center for Software*, Special Publication, May 1981

Turner, C., G. Caron, and G. Brement, *NASA/SEL Data Compendium, Data and Analysis Center for Software*, Special Publication, April 1981

[5]Valett, J., and F. McGarry, "A Summary of Software Measurement Experiences in the Software Engineering Laboratory," *Proceedings of the 21st Annual Hawaii International Conference on System Sciences*, January 1988

[3]Weiss, D. M., and V. R. Basili, "Evaluating Software Development by Analysis of Changes: Some Data From the Software Engineering Laboratory," *IEEE Transactions on Software Engineering*, February 1985

[5]Wu, L., V. Basili, and K. Reed, "A Structure Coverage Tool for Ada Software Systems," *Proceedings of the Joint Ada Conference*, March 1987

[1]Zelkowitz, M. V., "Resource Estimation for Medium Scale Software Projects," *Proceedings of the Twelfth Conference on the Interface of Statistics and Computer Science. New York:* IEEE Computer Society Press, 1979

[2]Zelkowitz, M. V., "Data Collection and Evaluation for Experimental Computer Science Research," *Empirical Foundations for Computer and Information Science* (proceedings), November 1982

[6]Zelkowitz, M. V., "The Effectiveness of Software Prototyping: A Case Study," *Proceedings of the 26th Annual Technical Symposium of the Washington, D. C., Chapter of the ACM*, June 1987

[6]Zelkowitz, M. V., "Resource Utilization During Software Development," *Journal of Systems and Software*, 1988

Zelkowitz, M. V., and V. R. Basili, "Operational Aspects of a Software Measurement Facility," *Proceedings of the Software Life Cycle Management Workshop*, September 1977

NOTES:

[1]This article also appears in SEL-82-004, *Collected Software Engineering Papers: Volume I*, July 1982.

[2]This article also appears in SEL-83-003, *Collected Software Engineering Papers: Volume II*, November 1983.

[3]This article also appears in SEL-85-003, *Collected Software Engineering Papers: Volume III*, November 1985.

[4]This article also appears in SEL-86-004, *Collected Software Engineering Papers: Volume IV*, November 1986.

[5]This article also appears in SEL-87-009, *Collected Software Engineering Papers: Volume V*, November 1987.

[6]This article also appears in SEL-88-002, *Collected Software Engineering Papers: Volume VI*, November 1988.

[7]This article also appears in SEL-89-006, *Collected Software Engineering Papers: Volume VII*, November 1989.