AD-A068 635 WOLF COMPUTER CORP CAMBRIDGE MA F/G 9/2
RADC SOFTWARE SPECIFICATION SURVEY.(U)
MAR 79 W M WOLF F30602-77-C-0173
UNCLASSIFIED RADC-TR-79-25 NL

1 OF 2
AD A068635

LEVEL II

RADC-TR-79-25
Final Technical Report
March 1979

RADC SOFTWARE SPECIFICATION SURVEY

Wolf Computer Corp.

William M. Wolf

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

RADC SOFTWARE SPECIFICATION SURVEY

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-79-25 has been reviewed and is approved for publication.

APPROVED: Michael Landes

MICHAEL LANDES
Project Engineer

APPROVED: Wendall C Bauman

WENDALL C. BAUMAN, Col, USAF
Chief, Information Sciences Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (ISIE) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RADC-TR-79-25	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): RADC SOFTWARE SPECIFICATION SURVEY	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report. Sep 77 - Sep 78
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): William M. Wolf	8. CONTRACT OR GRANT NUMBER(s): F30602-77-C-0173
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Wolf Computer Corp. 238 Main Street Cambridge MA 02142	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62702F 55811801
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (ISIE) Griffiss AFB NY 13441	12. REPORT DATE: March 1979
	13. NUMBER OF PAGES: 147
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

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

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Michael Landes (ISIE)

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

software engineering
structured programming
modern software engineering
software specification

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

This report contains the results of a survey conducted under contract to determine the attitude of contractors and government agencies to the use of an existing software specification in the contractual procurement of computer software. In addition, the survey elicits constructive criticism from respondees. Suggestions received were subsequently used to update and revise the document, (RADC Software Development Specification #CP0787796100D dated 31 Mar 78) upon which the survey was based.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

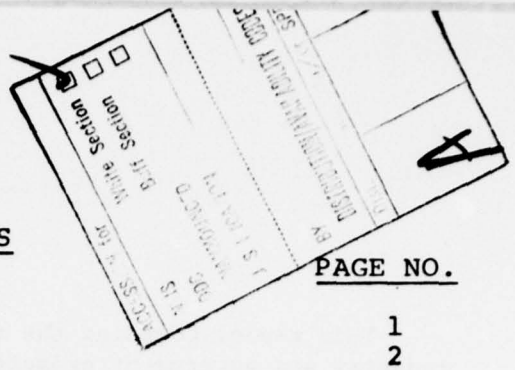
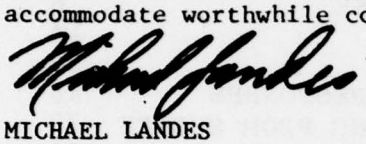


TABLE OF CONTENTS

EVALUATION

This report contains the results of a survey conducted throughout industry and government agencies. The survey solicited reactions to applying the RADC Software Development Specification to contracts involving the procurement of computer software. Questionnaires, copies of the Specification, and its Users Guide were mailed to participants in the survey and the results were tabulated and summarized in this report. In addition to the questionnaires, personal interviews were conducted with selected organizations to discuss issues in depth. As a result of this survey, the existing Specification will be modified to accommodate worthwhile comments and suggested additions.



MICHAEL LANDES
Project Engineer

INTRODUCTION

It was recently estimated that the Department of Defense is spending over three billion dollars annually on weapons systems software. This estimate was said to be conservative and did not include general purpose ADP. The current distribution of these dollars is estimated to be

* 68% system development

* 32% operations/maintenance.

The fact that this software is a fundamental and critical component in major defense systems is well-known. It is a disturbing recognition of today's contracting environment that the rules for purchasing this software are inconsistent and inadequate. A clearly defined specification, intelligently used, is long overdue.

It was an attempt to satisfy this need that prompted the USAF Rome Air Development Center (RADC) technical staff, principally Michael Landes, to write the - "RADC Computer Software Development Specification No. CP0787796100D". This Specification has been successfully used at RADC for a few years and will continue to be in use in the future. In response to an obvious need, a Users Guide to the Specification was introduced in 1977 as a supplement to the Specification.

The success of these documents at RADC and the serious need in other agencies strongly suggests that consideration be given to their broader use. As an integral stage in this evolutionary development, it is both necessary and desirable to obtain industry and other government agency feedback.

Thus, Contract No. F30602-77C-0173 was written with Wolf Computer Corp. to survey organizations affected by the broader application of this Specification. The objective of the Statement of Work is repeated - "The objective of this effort is to provide engineering services to conduct a national survey of the Computer Program Software Specification to determine its degree of acceptance by industry and Government on software development procurements and to increase the degree of acceptance to contractors by inviting them to submit recommendations and modifications to the specification."

This final report under this contract describes the approach taken to attain that objective and the results obtained.

SECTION I - APPROACH

A top down approach was employed to elicit response to the Specification and Users Guide from government and industry. That is, the name and address of the Chief Executive Officer (Chairman, President, Commander, etc.) were sought and a letter was sent to him requesting the organization's participation. In addition, the letter requested the name and address of the technical person within that organization to whom the specifics should be sent. This approach worked very well to get top-level support of the response. The sources of the initial names and addresses will be described below.

A. GOVERNMENTAL SOURCES

The principal source of names and addresses of government organizations was the "U.S. Government Manual 1977/1978" published by the U.S. Department of Commerce. A secondary source of government organizations was the "Directory of Federal R&D Installations" 1969; National Science Foundation; Washington, DC.

B. INDUSTRIAL SOURCES

The following libraries were researched to determine those companies and groups of companies (associations) who should be approached:

Massachusetts Institute of Technology - Sloan School Library; Baker Engineering Library; Dewey Library;
Harvard University - Baker Library;
Boston Public Library.

From these libraries the following references were obtained:

ACM "Roster of Members" 1977; Association of Computing Machinery; New York, NY;
"The Computer Directory and Buyers' Guide" 1976; Berkeley Enterprises; Newtonville, MA;
"Encyclopedia of Associations" 1977; Gale Research Co.; Detroit, MI;
"Guide to Reference Sources in the Computer Sciences" 1974; Macmillan Information; New York, NY;
"Worldwide Directory of Computer Companies" 1973-1974; Academic Media; Orange, NJ.

These references were purchased and served as sources for companies to be surveyed.

An additional industrial source was the June 1976 DATAMATION listing of the 50 largest EDP companies.

C. DEPARTMENT OF DEFENSE LISTING

The "Defense Department list of 500 Contractors Receiving Largest Dollar Volume of Prime Contract Awards for Research, Development, Test & Evaluation (FY 1976)." was also used as a source. The cutoff in this list was the first 100 organizations, both Profit and Non-Profit institutions. The sum of the gross sales from profit-making contractors represented by this truncation is $5,616,250,000 out of $6,057,451,000 or 93%. The non-profit institutions represent $555,839,000 out of $681,296,000 or 81%. Combined, the representation is $6,172,089,000 out of $6,738,747,000 or 92%. This is felt to be a significant representation of the volume of business performed by government contractors.

D. INTEGRATION OF SOURCES

Obviously, there was duplication from the sources described above. This duplication was eliminated and the sources were then compiled into two master lists - government and industry. The principal officers of each organization were obtained from the libraries. These lists may be found in Appendix A - Government Sources and Appendix B - Industrial Sources.

The sum of all the organizations, both government and industrial, is 377 organizations. All applicable government organizations were represented as well as over 90% (measured in gross sales) of DoD contractors engaged in Research, Development, Test and Evaluation. Included are 21 associations which are presented in Appendix B.

E. FIRST ITERATION MAILING

As mentioned previously, a letter was sent to each Chief Executive Officer. A sample of this letter is reproduced in Figure 1. It was decided, with the contract monitor, that the letter would have a greater impact and be more official if it were sent from RADC and signed by the contract monitor. This was a correct decision with respect to the industrial sector, measured by the response.

However, this approach was unfortunately interpreted as a breach of protocol with respect to the government sector.

DEPARTMENT OF THE AIR FORCE
HEADQUARTERS ROME AIR DEVELOPMENT CENTER (AFSC)
GRIFFISS AIR FORCE BASE, NEW YORK 13441

ISIM, M Landes(315-330-2672) 5 Jan 78

Software Development Specification Survey

Mr. Roderic Muir Fredrickson, Director
The RAND Computation Center
1700 Main St.
Santa Monica, CA 90406

Dear Sir,

The USAF Rome Development Center has developed and is using a computer software development specification in its procurement process. This document is entitled "RADC Computer Software Development Specification No. CP0787796100C." A Users guide to this Specification has also been written.

These documents are the result of a continuing effort to decrease the high cost of computer software development while assuring software which accurately reflects mission requirements. As a preliminary step to expanding the use of this Specification, we wish to obtain the reaction of companies who are expected to be affected by its use by conducting a survey. This survey will provide an opportunity for your organization to provide input to the Specification before it becomes integrated into the Government procurement regulations.

The purpose of this letter is to invite your organization to participate in the survey. If you wish to participate, please send us the name(s) of the appropriate person(s) in your organization to whom we should send our questionaire and copies of the Specification and Users Guide.

Yours truly,

Figure 1

Michael Landes, Project Mgr.

It was felt to be inappropriate for the contract monitor to write to the top levels of government requesting their participation in the survey. A retraction letter was sent to mend this breach.

It is interesting to note, in defense of this approach, that the software problem is so serious that the response demonstrates the high degree of interest of other government agencies.

For example:

"This command is interested in participating in a survey which may decrease the high cost of computer software development while assuring the applications software accurately reflects mission requirements."

William A. Myers III
Rear Admiral
U.S. Navy

"This office has long recognized the importance of utilizing computer software development specifications in the procurement process and would be most willing to participate in your survey."

Orville L. Smiley, Dir.
Automated Systems Off.
Asst. Sec. of Defense

"It is with enthusiasm that we at Air Force Studies and Analysis accept your invitation to review your document, "RADC Computer Software Development Specification No. CP0787796100C." We are in the process of developing a computer software development specification for use in the procurement of our software and have been giving this subject a great deal of consideration lately."

Richard L. Blair
Lt. Col., USAF; Chief
Computer App. Gp.
ACS/Studies & Analysis

The response from the first iteration mailing was enthusiastic. Selected copies of the letters from other government agencies are included in Appendix C. Of the total of 377 letters mailed, the following responses were received:

	Mailed	Responded	Percentage Response
Government	147	49	33%
Industry	209	87	41%
Associations	21	5	23%
Total	377	141	37%

F. SECOND ITERATION MAILING

Following the response from the first iteration mailing, a questionnaire was developed and mailed with a copy of the Specification and Users Guide. A copy of this questionnaire is included in Section II. A copy of the response from the Chief Executive Officer was also included. Figure 2 presents a copy of the covering letter sent with the second iteration.

The response from the second iteration mailing is listed below:

	Response
Government	12
Industry	39
Associations	1
Total	52

Although the final response (52 out of an initial 377) may appear to be less than expected, it will be seen that the respondents gave a great deal of careful thought and study to their constructive criticisms. Thus, they are very valuable to the effort and a much more useful Specification will result following the incorporation of the recommended changes in this report which meet with RADC approval. Further elaboration on these responses will be presented in Section II.

G. INTERVIEWS

In addition to the mailings described above, personal interviews were conducted at selected industrial organizations by Mr. Michael Landes (Specification author) and Mr. William Wolf (Principal Investigator on this contract). These interviews will be described in Section IV.

WOLF COMPUTER CORP.

238 MAIN STREET · CAMBRIDGE · MASSACHUSETTS 02142 · (617) 547-4909

13 May 1978

Dear Sir,

We appreciate your willingness to participate in the RADC software development survey.

The Wolf Computer Corp., under contract to the USAF RADC, has been assigned the role of distributing copies of the Questionnaire; Software Development Specification No. CP 0787796100D; and the Users Guide to this Specification. The Wolf Corp. will also receive and compile the responses. Therefore, please direct your response to:

Mr. William Wolf
Wolf Computer Corp.
238 Main Street
Cambridge, MA 02142

It should be understood that your participation in this survey is voluntary and is provided at no cost to the government or the Wolf Computer Corporation.

We need and value your input and feedback. This is a valuable opportunity to influence this Specification as it evolves and matures into wider acceptance in the government-industrial contracting environment.

Very truly yours,

WMWolf

William M. Wolf
President

Figure 2

SECTION II - QUESTIONNAIRE AND ITS RESPONSE

After a number of questionnaires were written, considered, and discussed, it was decided that a simple, strainghtforward, open-ended one would be best suited for the desired return. The principal reason for keeping it short is that the Specification and Users Guide might seem overwhelming to the first time reader. Thus, a five page questionnaire would make the task of responding seem less onerous. On the other hand, general as well as specific response was invited. As will be seen, this approach was useful in producing a broader spectrum of results than could have been solicited by specific questions and answers.

The questionnaire as structured provides a valuable opinion poll of the modern programming practices. Each respondent was asked for his opinion of the utility of a practice on a score of 0 to 10 (where 0 = of no use, 10 = most useful). These scores will be tabulated for comparative purposes in the pages to follow, wherein the responses are tabulated.

The five page questionnaire is reproduced in Figure 3.

QUESTIONNAIRE

This questionnaire relates to two documents, the Specification No. CP 0787796100D and the Users Guide to the Specification. Both these documents were written and are in use at the USAF Rome Air Development Center. They are being considered for more extensive use on government procurements involving software development. As an important part of this consideration, the views of other government agencies and industry are being surveyed. Therefore, it is important that careful consideration be given to these documents. We would greatly appreciate it if you would answer the questions herein and mail your response to:

Mr. William Wolf
Wolf Computer Corp.
238 Main Street
Cambridge, MA 02142

The Wolf Corp. is administering this survey for the USAF/RADC. The questionnaire is in three parts. The first details the extent to which your organization is involved in computer software development. Since our purpose is to gain general background, reasonable approximations will suffice. The second part seeks your experience with tools and procedures of modern programming technology. The third part asks for specific constructive criticisms of the Specification and Users Guide.

Please feel free to call Mr. William Wolf at (617)547-4909 or Mr. Michael Landes at (315)330-2672 if you have any questions or comments not covered by these documents. Thank you for your cooperation.

Figure 3

I RESPONDENT DATA

NAME _____ TEL. NO. _____

ORGANIZATION _____

ADDRESS: Street _____

City _____

State _____ Zip _____

DOLLARS SPENT ANNUALLY ON COMPUTER SOFTWARE DEVELOPMENT

	less than 100K	100K to 500K	500K to 1Mil	1Mil to 5Mil	more than 5Mil
INTERNAL DEVELOPMENT	____	____	____	____	____
EXTERNAL DEVELOPMENT	____	____	____	____	____
FOR GOVERNMENT USE	____	____	____	____	____
FOR INDUSTRIAL USE	____	____	____	____	____
TOTAL	____	____	____	____	____

Figure 3 (cont.)

II MODERN PROGRAMMING USAGE

The Specification relies upon the use of modern programming technology. Please check off on the listing below which of the following tools and procedures are employed on your software development efforts. Also, along with your checks, please score on a scale of 0 to 10 the usefulness of each of these tools or procedures that you have used (where 0 = of no use, 10 = most useful).

		SCORE
HIGHER ORDER LANGUAGE (Please Specify Which)	____	____
SOFTWARE DEVELOPMENT TOOLS		
CODE AUDITOR	____	____
SOFTWARE VERIFIER	____	____
TEXT EDITOR	____	____
OTHER (Please Specify)	____	____
PROGRAM DESIGN LANGUAGE (Please Specify)	____	____
DESIGN STANDARDS		
TOP DOWN PROGRAM DESIGN	____	____
TOP DOWN SYSTEM DESIGN	____	____
CODING STANDARDS		
STRUCTURED CODING	____	____
STRUCTURED CODING WITH PREPROCESSOR/PRECOMPILER	____	____
STRUCTURED PROGRAMMING LANGUAGE	____	____
CODING CONVENTIONS		
BASIC CODING CONVENTIONS	____	____
CODING CONVENTIONS WITH CODE READING	____	____
CODING CONVENTIONS WITH DESIGN/CODE REVIEW	____	____
CODING CONVENTIONS WITH INSPECTIONS	____	____

Figure 3 (cont.)

		SCORE
CHARACTER SET STANDARDS	_____	_____
SOFTWARE PROJECT RESOURCE REQUIREMENTS ESTIMATION	_____	_____
PROGRAMMING SUPPORT LIBRARY (PSL)		
MANUAL PSL	_____	_____
BASIC PSL	_____	_____
FULL PSL WITH MANAGEMENT DATA COLLECTION AND REPORTING	_____	_____
SOFTWARE RELIABILITY MEASUREMENT	_____	_____
PERSONNEL ORGANIZATION		
MODIFIED PROGRAMMER TEAM	_____	_____
PROGRAMMER TEAM	_____	_____
QUALITY ASSURANCE	_____	_____
SOFTWARE SECURITY PROCEDURES	_____	_____
DOCUMENTATION STANDARDS	_____	_____
OPERATING SYSTEMS AND UTILITIES SERVICES (OSS)		
OSS BASIC REQUIREMENTS	_____	_____
OSS WITH INTERACTIVE CAPABILITY	_____	_____
OSS WITH SECURITY REQUIREMENTS	_____	_____
OTHER TOOLS AND/OR PROCEDURES (Please Specify):		
_____		_____
_____		_____

Figure 3 (cont.)

III SPECIFICATION AND USERS GUIDE COMMENTS

Please examine Specification No. CP 0787796100D and the Users Guide and provide us with constructive criticism based upon your experience and opinion.

RADC experience has shown that the employment of the modern programming practices specified has led to more reliable software which is less expensive over the entire life cycle.

Do you agree? Yes _____ No _____

If yes, what experiences of yours corroborate this view? (Please discuss below)

If no, why?

Do you agree in some areas but not in others?

We are interested in knowing of techniques specified in the Specification which may be contradictory to your procedures. Also, are there omissions? How do you feel we can improve the Specification and the Users Guide?

If you choose to comment specifically, for particular items, this would be preferred. If you wish to respond in generalities, we welcome that response as well.

Figure 3 (cont.)

A. GOVERNMENT RESPONSE

Twelve response were received from the following government agencies:

Department of Commerce
Department of Transportation - Federal Highway Admin.
Department of Transportation - U.S. Coast Guard
NASA - George C. Marshall Space Flight Center
NASA - Lewis Research Center
National Center for Atmospheric Research
National Technical Information Service
NAVY - David Taylor Model Basin (2)
NAVY - Naval Intelligence
U.S. Postal Service
Veterans Administration

The comment part of the questionnaires is reproduced in Appendix D.

The scoring part of the responses is summarized in the table in Figure 4. It should be noted that three additional columns are added to the right:

* Total Score (SUM) - This column contains the total of the opinion scores assigned to each practice by the respondent.

* Average Score (AVG) - This column is the simple average of the opinion scores showing how highly they were rated. It is the sum of the scores divided by the number using that practice.

* Percent Usage (PER) - This column contains the percentage of the respondents who have reported on and use a particular practice.

	1	2	3	4	5	6	7	8	9	10	11	12	SUM	AVG	PER
HIGHER ORDER LANGUAGE	10	10	10	10	10	10	8	8	10	6	7	10	109	9.1	100
SOFTWARE DEVELOPMENT TOOLS															
Code Auditor	-	-	-	-	-	-	-	5	-	-	-	-	5	5.0	8
Software Verifier	-	-	-	-	-	10	-	1	-	-	5	-	16	5.3	25
Text Editor	-	-	-	10	10	10	-	9	10	2	8	8	67	8.4	67
PROGRAM DESIGN LANGUAGE	-	-	-	-	4	-	-	-	-	-	-	-	4	4.0	8
DESIGN STANDARDS															
Top Down Program Design	-	10	8	7	9	-	-	7	5	8	10	8	73	8.1	75
Top Down System Design	-	10	-	-	9	-	-	7	5	7	-	9	47	7.8	50
CODING STANDARDS															
Structured Coding	-	10	8	-	9	10	8	-	10	9	6	9	79	8.8	75
Struct.Coding with Preprc/Precomp.	-	-	-	7	2	10	-	-	-	-	-	-	19	6.3	25
Structured Prog. Language	-	-	-	-	-	2	-	-	-	-	-	-	2	2.0	8
CODING CONVENTIONS															
Basic Cod.Cnvtns.	-	-	7	10	8	5	5	8	10	6	10	8	77	7.7	83
Cod.Convtns. with Code Reading	-	-	-	-	-	-	4	-	-	-	4	9	17	5.7	25
Cod.Convtns. with Des/Code Review	-	-	-	-	7	-	7	-	-	5	4	9	32	6.4	42
Cod.Convtns. with Inspections	-	-	-	-	7	-	-	-	-	7	2	9	25	6.3	33
CHAR.SET STANDS.	10	10	9	7	7	-	7	-	10	-	-	7	67	8.4	67
SOFTWARE REQTS. ESTIMATION	8	10	-	5	8	-	6	-	-	3	2	-	42	6.0	58
PROG. SUPPORT LIBRARY															
Manual PSL	9	10	-	-	8	-	5	-	-	8	5	8	53	7.6	58
Basic PSL	-	-	-	8	-	-	-	-	-	-	-	10	18	9.0	17
Full PSL	-	-	-	-	8	-	-	-	-	-	-	-	8	8.0	8
SOFTWARE REL.MEAS.	8	-	-	-	2	-	-	-	5	-	2	-	15	3.8	33
PERSONNEL ORGANIZATION															
Mod.Prog.Team	9	10	8	5	9	-	7	-	-	-	8	7	54	6.8	67
Programmer Team	9	-	-	2	-	-	-	-	-	-	-	-	11	5.5	17
QUALITY ASSURANCE	9	10	8	-	2	-	-	-	-	-	1	8	38	6.3	50
SOFTWARE SECURITY	-	10	10	8	8	-	-	-	-	4	1	8	49	7.0	58
DOCUMENT.STANDS.	10	10	10	8	6	5	8	7	10	7	10	10	101	8.4	100
OPERATING SYSTEMS															
OSS Basic	10	-	-	-	8	-	8	-	-	-	-	8	34	8.5	33
OSS Interactive	10	0	7	9	9	-	7	7	-	-	10	9	68	8.5	67
OSS with Security	-	-	-	8	5	-	-	7	-	-	-	9	29	7.3	33

Figure 4

The next three tables, Figures 5, 6, and 7, present the ranking (from high to low) of the practices three ways - by total score, by average score, and by percent usage.

RANKING OF GOVERNMENT RESPONSE BY OPINION SCORE TOTALS

PRACTICE	SCORE
HIGHER ORDER LANGUAGE	109
DOCUMENTATION STANDARDS	101
CODING STANDARDS - Structured Coding	79
CODING CONVENTIONS - Basic Coding Conventions	77
DESIGN STANDARDS - Top Down Program Design	73
OSS with Interactive Capability	68
SOFTWARE DEVELOPMENT TOOLS - Text Editor	67
CHARACTER SET STANDARDS	67
PERSONNEL ORGANIZATION - Modified Programmer Team	54
PROGRAMMING SUPPORT LIBRARY (PSL) - Manual PSL	53
SOFTWARE SECURITY PROCEDURES	49
DESIGN STANDARDS - Top Down System Design	47
SOFTWARE PROJECT RESOURCE REQUIREMENTS ESTIMATION	42
QUALITY ASSURANCE	38
OSS Basic Requirements	34
CODING CONVENTIONS with Design/Code Review	32
OSS with Security Requirements	29
CODING CONVENTIONS with Inspections	25
CODING STANDARDS - Structured Coding with Preproc./Precomp.	19
PROGRAMMING SUPPORT LIBRARY (PSL) - Basic PSL	18
CODING CONVENTIONS with Code Reading	17
SOFTWARE DEVELOPMENT TOOLS - Software Verifier	16
SOFTWARE RELIABILITY MEASUREMENT	15
PERSONNEL ORGANIZATION - Programmer Team	11
PROGRAMMING SUPPORT LIBRARY (PSL) - Full PSL	8
SOFTWARE DEVELOPMENT TOOLS - Code Auditor	5
PROGRAM DESIGN LANGUAGE	4
CODING STANDARDS - Structured Programming Language	2

The modern programming practices are ranked above in the order of the total scores of their usefulness. That is, the SUM column of Figure 4 is presented above in numerical order from Higher Order Language with a total of 109 to Coding Standards - Structured Programming Language with a total of 2.

Figure 5

RANKING OF GOVERNMENT RESPONSE BY AVERAGE SCORE

PRACTICE	SCORE
HIGHER ORDER LANGUAGE	9.1
PROGRAMMING SUPPORT LIBRARY - Basic PSL	9.0
CODING STANDARDS - Structured Coding	8.8
OSS Basic Requirements	8.5
OSS with Interactive Capability	8.5
SOFTWARE DEVELOPMENT TOOLS - Text Editor	8.4
CHARACTER SET STANDARDS	8.4
DOCUMENTATION STANDARDS	8.4
DESIGN STANDARDS - Top Down Program Design	8.1
PROGRAMMING SUPPORT LIBRARY (PSL) - Full PSL	8.0
DESIGN STANDARDS - Top Down System Design	7.8
CODING CONVENTIONS - Basic Coding Conventions	7.7
PROGRAMMING SUPPORT LIBRARY (PSL) - Manual PSL	7.6
OSS with Security Requirements	7.3
SOFTWARE SECURITY PROCEDURES	7.0
PERSONNEL ORGANIZATION - Modified Programmer Team	6.8
CODING CONVENTIONS with Design/Code Review	6.4
CODING STANDARDS - Structured Coding w. Preproc/Precomp.	6.3
CODING CONVENTIONS with Inspections	6.3
QUALITY ASSURANCE	6.3
SOFTWARE PROJECT RESOURCE REQUIREMENTS ESTIMATION	6.0
CODING CONVENTIONS with Code Reading	5.7
PERSONNEL ORGANIZATION - Programmer Team	5.5
SOFTWARE DEVELOPMENT TOOLS - Software Verifier	5.3
SOFTWARE DEVELOPMENT TOOLS - Code Auditor	5.0
PROGRAM DESIGN LANGUAGE	4.0
SOFTWARE RELIABILITY MEASUREMENT	3.8
CODING STANDARDS - Structured Programming Language	2.0

The programming practices are ranked above by their average score. The average score is defined as the sum of all the scores for a given practice divided by the number of scores.

Figure 6

RANKING OF GOVERNMENT RESPONSE BY PERCENT USAGE

PRACTICE	PERCENT USAGE
HIGHER ORDER LANGUAGE	100
DOCUMENTATION STANDARDS	100
CODING CONVENTIONS - Basic Coding Conventions	83
DESIGN STANDARDS - Top Down Program Design	75
CODING STANDARDS - Structured Coding	75
SOFTWARE DEVELOPMENT TOOLS - Text Editor	67
CHARACTER SET STANDARDS	67
PERSONNEL ORGANIZATION - Modified Programmer Team	67
OSS with Interactive Capability	67
SOFTWARE PROJECT RESOURCE REQUIREMENTS ESTIMATION	58
PROGRAMMING SUPPORT LIBRARY (PSL) - Manual PSL	58
SOFTWARE SECURITY PROCEDURES	58
DESIGN STANDARDS - Top Down System Design	50
QUALITY ASSURANCE	50
CODING CONVENTIONS with Design/Code Review	42
CODING CONVENTIONS with Inspections	33
SOFTWARE RELIABILITY MEASUREMENT	33
OSS Basic Requirements	33
OSS with Security Requirements	33
SOFTWARE DEVELOPMENT TOOLS - Software Verifier	25
CODING STANDARDS - Structured Coding w. Preproc/Precompiler	25
CODING CONVENTIONS with Code Reading	25
PROGRAMMING SUPPORT LIBRARY (PSL) - Basic PSL	17
PERSONNEL ORGANIZATION - Programmer Team	17
SOFTWARE DEVELOPMENT TOOLS - Code Auditor	8
PROGRAM DESIGN LANGUAGE	8
CODING STANDARDS - Structured Programming Language	8
PROGRAMMING SUPPORT LIBRARY (PSL) - Full PSL	8

In the above ranking, the percent usage is determined by dividing the total number of respondents reporting on a given practice by the total number responding (12).

Figure 7

B. INDUSTRY RESPONSE

Since only one response was received from an association (IEEE), it is included with those from industry. Thus, forty responses were received from the following organizations:

ADP Network Services, Inc.
Aerospace Corp.
AIL (Cutler-Hammer) (3)
Argonne National Laboratory
Battelle - Columbus Laboratories
Bendix Corp. (2)
Bunker Ramo Corp.
CALSPAN Corp.
Compuserve Inc.
Computer Data Systems Inc.
Computer Sciences Corp.
Control Data Corp.
Delco Electronics
General Dynamics Corp.
General Electric Co.
Harris Corp.
Honeywell Information Systems, Inc.
IEEE - Computer Society Subcommittee on Software Engineering Standards
Jet Propulsion Laboratory
Keystone Computer Associates, Inc.
Lockheed-California Company (2)
Litton Data Systems
Litton Mellonics System Development
Massachusetts Computer Associates, Inc.
Mohawk Data Sciences Corp.
NCR Corp.
Rapidata
Scientific Time Sharing Corp.
Systems Research Laboratories, Inc.
TRW Corp.
University Computing Corp.
Western Electric (3)
Xerox Corp.

These responses formed the basis for another set of tables similar to those prepared for the government response. These are presented in the tables in Figures 8, 9, 10, and 11. The comment parts of some of the industrial responses are presented in Appendix E.

		1	2	3	4	5	6	7	8	9	10
HIGHER ORDER LANGUAGE		10	9	5	10	10	10	9	8	7	10
SOFTWARE	Code Auditor	-	9	-	10	8	2	5	-	6	6
DEVEL.	Software Verifier	-	-	8	10	8	-	5	5	3	5
TOOLS	Text Editor	10	8	2	10	10	8	8	7	3	8
PROGRAM DESIGN LANGUAGE		10	7	-	-	-	3	-	8	8	8
DESIGN	Top Down Program Design	-	7	-	5	8	8	8	9	5	10
STANDS.	Top Down System Design	-	8	5	10	10	8	7	9	8	10
CODING	Structured Coding	10	6	5	3	2	9	8	8	6	8
STANDS.	Structured Coding Preproc./Precompiler	10	7	-	2	5	4	8	-	1	10
	Structured Prog. Language	-	8	-	2	10	8	7	8	9	9
CODING	Basic Coding Conventions	-	7	8	10	6	8	7	-	5	10
CONVS.	Coding Conventions with Code Reading	-	8	10	2	8	5	7	-	6	6
	Coding Conventions with Design/Code Review	-	7	-	3	5	8	8	6	7	8
	Coding Conventions with Inspections	10	-	-	3	5	6	6	-	8	10
CHARACTER SET STANDARDS		-	7	-	-	8	-	6	-	3	10
SOFTWARE PROJECT RESOURCE REQUIREMENTS ESTIMATION		-	6	10	10	10	8	4	9	4	10
PROGRAMMING	Manual PSL	-	4	6	-	5	6	-	4	3	5
SUPPORT	Basic PSL	-	6	7	-	8	7	-	-	5	5
LIBRARY(PSL)	Full PSL	-	8	3	-	5	8	-	-	7	7
SOFTWARE RELIABILITY MEASUREMENT		-	-	-	-	2	-	-	-	5	5
PERSONNEL	Modified Programmer Team	-	6	8	-	-	9	7	6	6	8
ORGANIZ.	Programmer Team	-	-	-	-	-	-	7	-	6	10
QUALITY ASSURANCE		--	7	-	-	-	3	7	-	7	10
SOFTWARE SECURITY PROCEDURES		--	6	-	-	-	-	-	-	5	8
DOCUMENTATION STANDARDS		-	6	7	-	-	10	7	5	6	9
OPERATING	OSS Basic Requirements	-	7	5	10	10	-	6	-	8	10
SYSTEMS & UTILITIES (OSS)	OSS with Interactive Capability	10	9	-	-	10	-	9	-	0	10
	OSS with Security Reqts.	10	5	-	-	5	-	-	-	0	10

Figure 8

11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
5	9	9	6	8	9	8	10	9	10	10	10	7	10	7	10	10	10	8	9	10
-	-	1	-	6	4	5	-	8	-	5	-	-	4	-	-	5	-	-	-	-
-	-	-	-	6	4	4	-	-	3	-	-	-	7	-	-	5	8	-	-	-
8	6	8	7	6	8	7	5	9	10	10	8	5	10	7	8	10	6	8	7	10
-	6	-	-	7	8	2	-	9	-	4	5	2	8	-	-	9	-	-	9	-
7	8	7	10	8	9	8	5	9	10	8	8	7	5	7	4	8	8	10	7	3
8	-	8	10	8	9	8	5	10	10	7	-	7	3	7	-	9	8	10	7	3
6	-	7	10	7	5	7	3	8	5	7	8	-	6	7	2	7	8	7	5	5
-	5	-	-	5	7	4	-	-	-	7	8	10	6	-	1	3	8	-	-	8
-	-	9	-	-	10	9	-	10	-	-	-	-	6	-	-	10	-	-	9	5
5	10	5	6	7	9	8	6	10	10	9	5	4	2	10	-	4	8	4	8	5
-	10	-	7	7	8	5	8	9	-	1	-	-	5	-	-	3	-	8	8	-
-	-	-	8	8	8	6	-	8	5	5	-	-	7	-	5	5	8	9	8	-
-	5	10	-	9	8	5	-	9	5	3	-	5	-	-	-	8	-	-	5	-
5	8	2	5	5	10	8	-	10	10	10	10	3	-	10	-	-	-	-	5	8
7	-	-	-	8	8	4	3	8	10	3	-	-	5	5	-	8	9	-	9	-
8	-	-	2	6	7	6	3	9	5	8	-	-	5	--	-	-	2	7	8	-
-	8	-	5	7	8	7	6	8	-	8	-	7	-	10	-	3	-	8	-	-
-	-	-	-	8	9	7	-	6	-	9	-	-	-	5	-	9	10	-	-	-
-	-	-	-	5	8	2	8	-	4	2	-	-	-	-	-	2	5	-	8	-
8	7	7	7	10	4	6	5	8	10	6	5	-	5	6	-	5	8	3	-	-
-	-	-	-	10	3	2	-	6	-	3	-	8	5	-	6	8	-	-	9	5
10	7	4	10	10	6	6	8	5	5	7	-	-	6	7	-	7	10	6	3	5
6	5	3	-	7	8	5	3	-	5	7	-	-	7	5	-	4	-	-	3	-
10	7	5	6	7	10	7	4	9	10	5	10	5	7	7	3	3	9	3	10	5
8	5	7	6	7	7	6	7	10	5	9	8	-	7	10	10	-	5	-	6	7
8	7	9	8	9	9	7	9	9	-	9	-	7	9	10	10	9	-	8	8	10
8	-	10	-	8	10	8	5	-	-	9	-	-	7	10	-	-	-	-	-	5

32	33	34	35	36	37	38	39	40	SUM	AVG	PER
10	10	5	8	7	7	5	9	8	341	8.5	100
-	-	-	5	-	-	-	-	-	89	5.6	40
-	-	-	6	-	-	-	-	4	91	5.7	40
10	8	-	6	8	-	3	10	7	289	7.6	95
-	9	-	8	-	6	-	-	-	136	6.8	50
5	9	8	10	5	-	7	10	6	276	7.5	93
5	9	8	10	8	8	7	10	6	283	7.9	90
-	-	5	7	-	-	6	-	5	208	6.3	83
-	-	-	8	-	-	1	-	6	134	5.8	53
-	-	-	9	-	8	6	-	7	159	8.0	50
-	9	8	8	10	-	7	-	6	244	7.2	85
-	8	-	10	-	8	-	10	5	172	6,9	63
-	8	7	10	-	9	-	-	-	176	7.0	63
-	8	-	10	-	-	-	-	7	145	6.9	53
-	10	-	6	-	10	-	8	-	177	7.4	60
-	7	9	6	8	10	-	8	-	206	7.4	70
-	10	7	5	8	-	-	10	-	149	6.0	63
-	10	-	9	-	-	-	-	7	149	7.1	53
-	10	-	8	-	10	-	-	-	129	7.6	43
5	-	-	8	8	10	7	-	-	94	5.5	43
-	-	-	2	6	-	-	-	6	174	6.4	68
5	-	8	2	-	-	-	-	5	108	6.0	45
7	-	-	8	8	10	-	-	7	197	7.0	70
7	-	-	10	7	10	-	-	4	125	6.0	53
7	-	5	9	7	9	6	8	6	249	6.9	90
-	-	6	7	8	-	9	-	5	221	7.4	75
10	-	-	9	6	9	8	10	8	263	8.5	78
8	-	-	10	-	-	-	-	4	132	7.3	45

RANKING OF INDUSTRY RESPONSE BY OPINION SCORE TOTALS

PRACTICE	SCORE
HIGHER ORDER LANGUAGE	341
SOFTWARE DEVEL. TOOLS - Text Editor	289
DESIGN STANDARDS - Top Down System Design	283
DESIGN STANDARDS - Top Down Program Design	276
OSS with Interactive Capability	263
DOCUMENTATION STANDARDS	249
CODING CONVENTIONS - Basic	244
OSS Basic Requirements	221
CODING STANDARDS - Structured Coding	208
SOFTWARE PROJECT RESOURCE REQTS. ESTIMATION	206
QUALITY ASSURANCE	197
CHARACTER SET STANDARDS	177
CODING CONVENTIONS - with Design/Code Review	176
PERSONNEL ORGANIZATION - Modified Programmer Team	174
CODING CONVENTIONS - with Code Reading	172
CODING STANDARDS - Structured Programming Language	159
PROGRAMMING SUPPORT LIBRARY - Manual PSL	149
PROGRAMMING SUPPORT LIBRARY - Basic PSL	149
CODING CONVENTIONS - with Inspections	145
PROGRAM DESIGN LANGUAGE	136
CODING STANDARDS - Struct. Cod. with Preproc./Precomp.	134
OSS with Security Requirements	132
PROGRAMMING SUPPORT LIBRARY - Full PSL	129
SOFTWARE SECURITY PROCEDURES	125
PERSONNEL ORGANIZATION - Programmer Team	108
SOFTWARE RELIABILITY MEASUREMENT	94
SOFTWARE DEVEL. TOOLS - Software Verifier	91
SOFTWARE DEVEL. TOOLS - Code Auditor	89

Figure 9

RANKING OF INDUSTRY REPONSE BY AVERAGE SCORE

PRACTICE	SCORE
HIGHER ORDER LANGUAGE	8.5
OSS with Interactive Capability	8.5
CODING STANDARDS - Structured Programming Language	8.0
DESIGN STANDARDS - Top Down System Design	7.9
SOFTWARE DEVEL. TOOLS - Text Editor	7.6
PROGRAMMING SUPPORT LIBRARY - Full PSL	7.6
DESIGN STANDARDS - Top Down Program Design	7.5
CHARACTER SET STANDARDS	7.4
SOFTWARE PROJECT RESOURCE REQTS. ESTIMATION	7.4
OSS Basic Requirements	7.4
OSS with Security Requirements	7.3
CODING CONVENTIONS - Basic	7.2
PROGRAMMING SUPPORT LIBRARY - Basic PSL	7.1
CODING CONVENTIONS - with Design/Code Review	7.0
QUALITY ASSURANCE	7.0
CODING CONVENTIONS - with Code Reading	6.9
CODING CONVENTIONS - with Inspections	6.9
DOCUMENTATION STANDARDS	6.9
PROGRAM DESIGN LANGUAGE	6.8
PERSONNEL ORGANIZATION - Modified Programmer Team	6.4
CODING STANDARDS - Structured Coding	6.3
PROGRAMMING SUPPORT LIBRARY - Manual PSL	6.0
PERSONNEL ORGANIZATION - Programmer Team	6.0
SOFTWARE SECURITY PROCEDURES	6.0
CODING STANDARDS - Struct. Cod. with Preproc./Precomp.	5.8
SOFTWARE DEVEL. TOOLS - Software Verifier	5.7
SOFTWARE DEVEL. TOOLS - Code Auditor	5.6
SOFTWARE RELIABILITY MEASUREMENT	5.5

Figure 10

RANKING OF INDUSTRY RESPONSE BY PERCENT USAGE

PRACTICE	PERCENT USAGE
HIGHER ORDER LANGUAGE	100
SOFTWARE DEVEL. TOOLS - Text Editor	95
DESIGN STANDARDS - Top Down Program Design	93
DESIGN STANDARDS - Top Down System Design	90
DOCUMENTATION STANDARDS	90
CODING CONVENTIONS - Basic	85
CODING STANDARDS - Structured Coding	83
OSS with Interactive Capability	78
OSS Basic Requirements	75
SOFTWARE PROJECT RESOURCE REQTS. ESTIMATION	70
QUALITY ASSURANCE	70
PERSONNEL ORGANIZATION - Modified Programmer Team	68
CODING CONVENTIONS - with Code Reading	63
CODING CONVENTIONS - with Design/Code Review	63
PROGRAMMING SUPPORT LIBRARY - Manual PSL	63
CHARACTER SET STANDARDS	60
CODING STANDARDS - Struct. Cod. with Preproc./Precomp.	58
CODING CONVENTIONS - with Inspections	53
PROGRAMMING SUPPORT LIBRARY - Basic PSL	53
SOFTWARE SECURITY PROCEDURES	53
PROGRAM DESIGN LANGUAGE	50
CODING STANDARDS - Structured Programming Language	50
PERSONNEL ORGANIZATION - Programmer Team	45
OSS with Security Requirements	45
PROGRAMMING SUPPORT LIBRARY - Full PSL	43
SOFTWARE RELIABILITY MEASUREMENT	43
SOFTWARE DEVEL. TOOLS - Code Auditor	40
SOFTWARE DEVEL. TOOLS - Software Verifier	40

Figure 11

C. COMBINED GOVERNMENT AND INDUSTRY RESPONSES

The responses from government and industry were summed and ranked by the three measures to present an integrated view of the responses. These are presented in Figures 12, 13, and 14.

RANKING OF COMBINED RESPONSE BY OPINION SCORE TOTALS

PRACTICE	SCORE
HIGHER ORDER LANGUAGE	450
SOFTWARE DEVEL. TOOLS - Text Editor	356
DOCUMENTATION STANDARDS	350
DESIGN STANDARDS - Top Down Program Design	349
OSS with Interactive Capability	331
DESIGN STANDARDS - Top Down System Design	330
CODING CONVENTIONS - Basic	321
CODING STANDARDS - Structured Coding	287
OSS Basic Requirements	255
SOFTWARE PROJECT RESOURCE REQTS. ESTIMATION	248
CHARACTER SET STANDARDS	244
QUALITY ASSURANCE	235
PERSONNEL ORGANIZATION - Modified Programmer Team	228
CODING CONVENTIONS - with Design/Code Review	208
PROGRAMMING SUPPORT LIBRARY - Manual PSL	202
CODING CONVENTIONS - with Code Reading	189
SOFTWARE SECURITY PROCEDURES	174
CODING CONVENTIONS - with Inspections	170
PROGRAMMING SUPPORT LIBRARY - Basic PSL	167
CODING STANDARDS - Structured Programming Language	161
CODING STANDARDS - Struct. Cod. with Preproc./Precomp.	153
OSS with Security Requirements	141
PROGRAM DESIGN LANGUAGE	140
PROGRAMMING SUPPORT LIBRARY - Full PSL	137
PERSONNEL ORGANIZATION - Programmer Team	119
SOFTWARE RELIABILITY MEASUREMENT	109
SOFTWARE DEVEL. TOOLS - Software Verifier	107
SOFTWARE DEVEL. TOOLS - Code Auditor	94

Figure 12

RANKING OF COMBINED RESPONSE BY AVERAGE SCORE

PRACTICE	SCORE
HIGHER ORDER LANGUAGE	8.5
OSS with Interactive Capability	8.5
CODING STANDARDS - Structured Programming Language	8.0
DESIGN STANDARDS - Top Down System Design	7.9
SOFTWARE DEVEL. TOOLS - Text Editor	7.6
PROGRAMMING SUPPORT LIBRARY - Full PSL	7.6
DESIGN STANDARDS - Top Down Program Design	7.5
CHARACTER SET STANDARDS	7.4
SOFTWARE PROJECT RESOURCE REQTS. ESTIMATION	7.4
OSS Basic Requirements	7.4
OSS with Security Requirements	7.3
CODING CONVENTIONS - Basic	7.2
PROGRAMMING SUPPORT LIBRARY - Basic PSL	7.1
CODING CONVENTIONS - with Design/Code Review	7.0
QUALITY ASSURANCE	7.0
CODING CONVENTIONS - with Inspections	6.9
DOCUMENTATION STANDARDS	6.9
PROGRAM DESIGN LANGUAGE	6.8
PERSONNEL ORGANIZATION - Modified Programmer Team	6.4
CODING STANDARDS - Structured Coding	6.3
PROGRAMMING SUPPORT LIBRARY - Manual PSL	6.0
PERSONNEL ORGANIZATION - Programmer Team	6.0
SOFTWARE SECURITY PROCEDURES	6.0
CODING STANDARDS - Struct. Cod. with Preproc./Precomp.	5.8
SOFTWARE DEVEL. TOOLS - Software Verifier	5.7
SOFTWARE DEVEL. TOOLS - Code Auditor	5.6
SOFTWARE RELIABILITY MEASUREMENT	5.5

Figure 13

RANKING OF COMBINED RESPONSE BY PERCENT USAGE

PRACTICE	PERCENT USAGE
HIGHER ORDER LANGUAGE	100
SOFTWARE DEVEL. TOOLS - Text Editor	95
DESIGN STANDARDS - Top Down Program Design	93
DESIGN STANDARDS - Top Down System Design	90
DOCUMENTATION STANDARDS	90
CODING CONVENTIONS - Basic	85
CODING STANDARDS - Structured Coding	83
OSS Interactive Capability	78
OSS Basic Requirements	75
SOFTWARE PROJECT RESOURCE REQTS. ESTIMATION	70
QUALITY ASSURANCE	70
PERSONNEL ORGANIZATION - Modified Programmer Team	68
CODING CONVENTIONS - with Code Reading	63
CODING CONVENTIONS - with Design/Code Review	63
PROGRAMMING SUPPORT LIBRARY - Manual PSL	63
CHARACTER SET STANDARDS	60
CODING STANDARDS - Struct. Cod. with Preproc./Precomp.	58
CODING CONVENTIONS - with Inspections	53
PROGRAMMING SUPPORT LIBRARY - Basic PSL	53
SOFTWARE SECURITY PROCEDURES	53
PROGRAM DESIGN LANGUAGE	50
CODING STANDARDS - Structured Programming Language	50
PERSONNEL ORGANIZATION - Programmer Team	45
OSS with Security Requirements	45
PROGRAMMING SUPPORT LIBRARY - Full PSL	43
SOFTWARE RELIABILITY MEASUREMENT	43
SOFTWARE DEVEL. TOOLS - Code Auditor	40
SOFTWARE DEVEL. TOOLS - Software Verifier	40

Figure 14

SECTION III RECOMMENDED CHANGES RESULTING FROM SURVEY

As a result of the responses to the survey, the following changes to the Specification are recommended:

1) The Section 5.4.4 entitled **Personnel** organization requirements should be modified to relieve the stringent programmer team organization requirements.

It is clear from viewing the low score given 5.4.4.2 Programmer team that few respondents think highly of the strict enforcement of rigid rules with respect to the programmer team construction - number of members, etc. This view was often repeated in the interviews. It was clear that the imposition of this section was felt to be an overbearing intrusion of the government into the contractor's domain and traditional area of responsibility. Even though the less-stringent 5.4.4.1 Modified programmer team scored a somewhat higher 228 (Figure 13), this is a watered-down version of 5.4.4.2 which has a high degree of objection associated with it.

2) Delete 5.4.3 Software Reliability Measurement.

This recommendation is made due to the lack of interest and use of this practice, reflected, in part, by its low score - 109 (Figure 13). The section was also questioned during our interviews and thus might more appropriately be mention in the Users Guide.

3) Modify Section 5.2.1 to reflect the unpopularity of the Software Verifier and Code Auditor as Software Development Tools.

This section should be rewritten to encourage the contractor to use software tools to the fullest. The Text Editor, which is very popular (Figure 13) and very useful (Figure 14) should be mentioned specifically but the other two should be relegated to a lesser role and mentioned as an additional category for selection, where applicable, by the contractor.

Many more changes will be presented in the next section (Section IV) as a result of the interviews conducted by Mr. Landes and Mr. Wolf.

SECTION IV INTERVIEWS

In addition to the formal response to the questionnaire, personal interviews were conducted with a selected number of companies. These companies were selected for their diversity in software development application. The companies were:

Aerospace Corp.
Boeing Corp.
ESL Corp.
General Dynamics
Harris Corp.
Texas Instruments
TRW Corp.
Tymshare Corp.
Vought Corp.
Western Electric

In addition, the designated representative of IEEE was interviewed, and a subcommittee of EIA was addressed.

These interviews were conducted by the Principal Investigator, Mr. William Wolf, and by Mr. Michael Landes, author of the Specification and RADC technical monitor for this contract. The interviews were very beneficial in that they provided the critical feedback that was sought.

Written comments from the questionnaires are presented in Appendix D and E. Interview comments are presented in the following pages in two ways:

1) General Comments in which overall comments pertaining to the Specification are listed.

2) Specific Comments on a section-by-section basis. Since these comments relate to particular paragraphs, the Specification is reproduced up to and including the paragraph about which a specific comment is made. To distinguish the comment from the Specification text, the title "COMMENT" is interposed and the comment itself is asterisked on a line-by-line basis.

A. GENERAL COMMENTS

* The Specification is a start, but it should be backed up by education. Before using the Specification, one should pass a yet-to-be-defined training course. Training of contracting officers and project managers is important.

* The Specification is a good document, but many years before its time.

* The government is presuming contractor prerogatives. The government should be interested in the product, not the process. One should go slowly in, for instance, the area of the organization of the project management teams.

* It will taka a few years for industry to tool up to use the Specification.

* To make the Users Guide more helpful, the Specification should be printed in a different font and imbedded in the Users Guide. Thus, one could readily follow the Specification and the Users Guide without placing them side by side.

* Examples in the Users Guide should be given that show where criticality, security, etc., would dictate what is used. Size is not the only discriminator.

* Specific companies and their products should not be mentioned.

* It would be useful if relative dollar amounts are suggested in the Users Guide that might be related to the various alternatives within a section. That is, as the options of increasing rigor are selected within a section, there usually is a corollary increase in cost. The recognition of this increase may be helpful in checking the unbridled use of more stringent options.

* The Specification is massive overkill.

B. SPECIFIC COMMENTS

1. SCOPE

1.1 Purpose. This specification provides computer software development engineering practices oriented toward modern programming techniques for use in project planning and procurement. The overall intent of this specification is to assure that performance requirements will be achieved with a minimum of cost/schedule/design/technical risk but is not intended to preclude exceeding the minimum techniques and procedures mandated herein. In addition, justifiable negotiated waivers or substitutions of functionally equivalent devices or solutions shall be entertained.

COMMENT

* The business of evaluating a contractor's functional
* equivalencies could be a really big problem. There must
* be a strong procedure allowing latitude here. Possibly,
* waiver forms could be inserted to emphasize the fact
* that it can be done. Some sort of mechanism that could
* be invoked whereby a contractor would have some feedback
* and/or complaint mechanism was suggested.

1.2 Use. This specification shall be used during the acquisition and development of computer programs to mandate use of software techniques, tools, and procedures to enhance software reliability and maintainability. The specification is organized in sections any of which may be selected or omitted for a particular project. Depending on size, cost, criticality, and other considerations of the subject effort, project engineers shall select appropriate sections and list them in the Statement of Work (See Fig. 2) or as input to Sec 30.5 in Mil Std 483. Section 30.5 appears in Appendix III of Mil Std 483 and describes an addition, Sec 3.3.8., to Appendix I of Mil Std 490.

Some sections offer varying levels of rigor in particular areas. The design of those areas is based on a matrix (See Figure 1) which depicts the various elements of disciplined programming technology partitioned into logical units. Units in each column of the matrix express increasing degrees of rigor from the bottom up.

A Users Guide has been produced which contains explanations and examples of how this specification is used. Included in this Users Guide is a complete list of Software Acquisition Management Guidebooks which provide further guidance in software acquisition.

Contractor proposal responses to sections in this specification are expected to be placed in the Computer Program Development Plan (CPDP), or in Sec. 3.3.8 if a System Specification is required under Mil Std 490.

Program procurement personnel shall assure that all data (technical data and computer software) required herein to be delivered to the Government is also listed on a DD 1423, Contract Data Requirements List (CDRL) of the solicitation/contract.

COMMENT

* It was felt that the parenthesized definition of data -
* (technical data and computer software) - was inappropri-
* ate.
* A conflict was pointed out between the requirement that
* the software be listed in the CDRL (as required by the
* previous paragraph) and that the software be a contract
* line item (as required in the following paragraph). It
* was felt that it should be one or the other, but not both.

Additionally, delivered computer software should be listed in the Contract/Solicitation Schedule as a separate contract line item or subline item, (AFSC ASPR Sup 9-603(a) and ASPR 7-104.9(n)).

COMMENT

* It was pointed out that the term ASPR (Armed Services
* Procurement Regulation) is in the process of being re-
* placed by the term DAR (Defense Acquisition Regulation).

3. DEFINITIONS. The definitions included in the referenced documents listed in Section 2 shall apply. In addition, the following clarification is included for convenience.

COMMENT

* It was observed that the definitions in the following
* (taken from the ASPR) are poor and inconsistent. How-
* ever, this is typical of the field today. For instance,
* Systems Command Revision to 483 has definitions which do
* not match AF Regulation 814.

3.5 Structured Programming Definitions. The following list of terms and their definitions apply to this specification:

a. Program design language (PDL) - A design tool used to facilitate the translation of system functional requirements into the elements of the program design hierarchy. The snytax and semantics of the resultant PDL listings may take either a highly formal or informal form and depict the software system design to varying levels. The level to which the PDL is applied is dependent on the particular application environment and the complexity of the programming task.

b. Programming support library (PSL) - A repository for programs and data necessary for the orderly development of computer software, especially those developed using structured programming technology. The repository provides two fundamental capabilities: programs and data are stored in machine readable form for computer operation and the identical data is stored in hard copy form. In addition, this repository contains management data. A PSL implementation also includes the necessary computer and office procedures for PSL operation and use.

COMMENT

* One contractor felt that the last sentence was too harsh.
* Another contractor uses software tools to perform these
* functions.

c. Structured segment - consists of a compilable set of executable statements with the following constraints. The code within a structured segment to be limited to the following structured programming constructs: sequence of two operations, conditional branch to one of two operations and return, conditional branch to a specified set of locations, operation repeated while a stated condition is true, and repeated operation until a stated condition is true. The structured programming constructs may be nested within a structured segment. A structured segment has one entry and one exit. In addition to executable code, a structured segment may contain non-executable instructions such as data declarations and descriptive commentary, i.e. annotation.

d. Structured code - A logically related set of code consisting of one or more structured constructs.

e. Structured programming - A general term used to depict the use of structured code (see above) and other modern programming techniques. These techniques apply to all phases of software development and are concerned with computer programming (design, coding, integration, test, etc.), software development management, and documentation activities.

COMMENT

* It was suggested that structured programming is not
* equivalent to structured design is not equivalent to
* structured implementation. This definition, as well as
* the others, should be reconsidered.

f. Structured programming technology - A term which collectively references:

Top down program design and implementation

Programming support library

Programmer teams and structured code

g. Structured source code listing - A listing for a computer program produced by the necessary software resulting in the following sections:

Section 1 contains the first executable structured segment listing (commonly referred to as the top level segment) as coded in the source programming language.

Section 2 contains all remaining structured segment listings. The structured segments appear in alphabetical order of their segment names. As in Section 1 each structured segment is represented as coded in the source programming language.

Section 3 contains the executing sequence among the structured segments where it is straightforward. Otherwise, where sequences are a function of input, alternate possible paths are indicated.

COMMENT

* This definition is said to be too restrictive for some
* languages - DOD-1, for instance, which is a PASCAL de-
* rivative. One contractor felt that this section should
* be deleted. The material of this section should be in
* the documentation of the software system. It is not
* part of the source code listings.

h. Top down program design and implementation - The concept of implementing in hierarchical sequence a detailed design, coding, integration and testing as concurrent operations.

COMMENT

* It was felt that this is a good definition but it was
* pointed out that it is contradictory to MIL-STD-483.

i. Top down structured program (TDSP) - A program composed of structured code with the additional characteristic of the source code being logically, but not necessarily physically, segmented in a hierarchical manner and only dependent on code already written. Control of execution between segments is restricted to transfers to the level immediately above or below the segment in question. Skipping levels is not permitted.

COMMENT

* It was felt that this definition is too easy to violate
* in ordinary programming. Common subroutines would not
* work since the levels were defined to be too specific.
* The last two sentences should be changed.

j. Program stub - A coding device used in a TDSP in which uncoded segments are set up with just the interface arguments which allow the control segment to call the stub and return. Stubs are generally used to allow preliminary testing of upper echelon segments before the entire program is coded.

3.6 "Unlimited rights" in technical data or computer software shall be those unlimited rights as the term is defined in the "Rights in Technical Data and Computer Software" clause, ASPR 7-104.9(a) of this solicitation/contract.

COMMENT

* The whole data rights issue is a problem which trans-
* scends this effort but was mentioned in the interviews.

5. DETAILED REQUIREMENTS FOR A DISCIPLINED SOFTWARE DEVELOPMENT ENVIRONMENT. This section prescribes the means and techniques whereby the programmer, analysts, management personnel, and others engaged in this effort are required to interact with the programming environment and facilities. Procedures, rules and constraints defined herein shall be used for program design, program structure and data structures which enhance the readability and reliability of programs and which minimize rework during the programming process. Only those sections defined below which are specified in the statement of work or Sec. 3.3.8 of Mil Std 490 shall be mandated in this effort.

COMMENT

* It was strongly stated that the Government should not
* specify the means and techniques but should specify the
* requirements. This was stated often in the interviews.
* One contractor felt that the following should be append-
* ed:
* "In this specification sub-sections within a
* section are arranged so that each successive
* sub-section within a section increases the
* requirements. Moreover, the successive sub-
* sections include the requirements of all the
* sub-sections preceding it. Therefore,
* whenever the appropriate degree of rigor has
* been determined and the sub-section selected,
* the preceding sub-sections do not need to be
* explicitly included."

5.1 Programming languages

5.1.1 Higher order language selection procedures. Selection procedures for higher order languages to be implemented on subject effort shall be in accordance with accepted standard Air Force languages. JOVIAL, FORTRAN, COBOL, or a limited use of PL/I shall be used unless specific waivers are obtained in accordance with standard Air Force procedures. The use of PL/I shall be limited to non-embedded computer applications which eliminates its use for all weapons system software. Language versions used on the

subject effort shall be limited to the ANSI X 3.9 - 1966 and X 3.10 - 1966 Standard for FORTRAN, ANSI X 3.53 - 1976 Standard for PL/I, ANSI X 3.23 - 1974 Standard for COBOL, and ANSI-approved modifications and additions to these languages. JOVIAL shall be limited to JOVIAL (J3) as specified by MIL-S-1588 (AF), and JOVIAL/J73 as defined in MIL-S-1589 (AF). Operating system requirements related to higher order language compilers are specified in 5.7.1 of this standard.

COMMENT

* It was noted that PL/1 is not mentioned in the Users
* Guide and it should be. Also, DOD-1 should be mentioned
* in the Users Guide as a future language.
* It was requested that consideration be given to a
* 5.1.1.1 section in which a discussion of the use of im-
* bedded assembly language code be presented.

5.1.2 Higher order language compiler validation procedures. Provision for validation and approval of compilers implemented on subject efforts shall be made. Higher order language compilers shall be validated and approved according to standard Air Force procedures. The contractor shall identify any additions or extensions to the basic compiler according to the language standards referenced in 5.1.1 before submission of the compiler for validation and approval.

COMMENT

* This was felt to be too restrictive. It was felt that,
* if one uses vendor-supplied compilers, one should not
* have to validate them.

5.1.3 JOVIAL Compiler Implementation. Where JOVIAL compiler implementation is included in the procurement, the JOVIAL Compiler Implementation Tool (JOCIT) shall be used.

COMMENT

* A definition and availability of JOCIT was requested.
* It was suggested that for real-time software where time
* and memory are critical, portions should be allowed in
* Assembly Language.
* One contractor suggested adding the following paragraph:
* "5.1.4 Assembly Languages. Programs which must be
* written in assembly language shall use the techniques
* of structured programming to the extent possible."

5.2 Software Development Tools and Procedures

5.2.1 Software development aids. Software tools (i.e. editors, computer aided requirements analysis packages, software verifiers, code auditors, and debugging aids) shall be used where feasible in the software production environment. The contractor shall submit a list of software aids over and above sections cited in sections under 5.2.1 which he will use on the subject effort [for approval to the procuring Government agency].

COMMENT

* It was suggested that this last sentence should be re-
* written to qualify what is meant.

Software tools (and their associated documentation) developed by the contractor under this or another government contract shall be delivered with unlimited data rights. (See Sec. 3.6)

COMMENT

* More than one contractor suggested that the clause -
* "...shall be delivered with unlimited data rights" -
* should be relaxed if necessary.
* It was stated that a contractor would have difficulty
* complying with this provision with respect to delivering
* tools and data rights.
* Great concern was expressed that legal hassles be avoid-
* ed. The case should be covered where a company (soft-
* ware subcontractor, for example) will not sell unlimited
* rights.

The following sections under 5.2.1 do not follow this document's convention of proceeding from less to more rigor, nor does each succeeding item include its predecessor. Any or all of these independent subdivisions may be chosen for a solicitation/contract.

5.2.1.1 Code Auditor. A code auditor which performs the following functions as a minimum shall be employed on code developed under this contract.

The code auditor shall enforce the following:

a. Documentation Standards - defining quantity and placement of commentary.

b. Format Standards - identifying physical placement and grouping of code elements on the source code listing.

c. Design Standards - limiting segment size and placing restrictions on the use of certain instructions with the end result of providing an optimization of code relative to execution time.

COMMENT

* It was felt that the work "optimization" was too strong.
* One contractor suggested that the code auditor should
* not limit the size of a segment to an arbitrary number
* of lines. The length of a segment should be determined
* by its function. The phrase "limiting segment size"
* should be deleted.

d. Structural Standards - requiring the use of rules for the top-down design and implementation of a system of programs and the requirement that the components adhere to a hierarchical form.

COMMENT

* It was suggested that this section - 5.2.1.1 - is a soft-
* ware Quality Assurance function.

5.2.1.2 Software Verifier. A software verifier shall be employed on code developed under this contract which has, as a minimum, the following functional capabilities:

a. source text structural analysis, identification of logical execution paths and instrumentation such that test case identification, module invocation and logical paths executed are recorded during test execution.

b. identification of program paths not executed.

c. provision of data to assist in the development of additional test cases appropriate to improvement of testing coverage.

COMMENT

* It was stated that it is better to have a man perform
* this function.

d. analysis of data flow monitoring the results of assignment and exchange statements.

e. automatic generation of computer program documentation.

5.2.1.3 Text Editor. A text editor shall be used in the development of software under this contract which contains, as a minimum, the following functional capabilities:

a. provision for generating textual information with and without special control characters. A capability for printing and/or displaying the text.

b. support of remote devices such as electromechanical and CRT based terminals for input, output, and editing functions.

c. provisions for creating, deleting, updating, renaming, moving, and copying text files where such files may contain computer programs, data, or management information.

d. settable page margins, numbering, headers, footers.

COMMENT

* Some Text Editors don't have all these features.
* One contractor felt that to require thes is a severe
* imposition of the Government on a contractor.

5.2.2 Program design language requirements

COMMENT

* One contractor felt that the program design language is
* immature at this time. It does not adequately replace
* the existing techniques of structured analysis and de-
* sign and directed flowgraphs. This paragraph should be
* deleted, in his opinion.

5.2.2.1 Program design language (PDL). An approved design language shall be implemented in the software design of the subject effort with the exceptions noted in Section 4. This design language is not required to be automated for use in the computer. One design language is described in Appendix 40. Designs written in a PDL for this contract shall be delivered with unlimited data rights (See Sec. 3.6).

COMMENT

* A feeling was expressed that PDL is just another lan-
* guage. It was also felt to be immature in the field.
* It was felt that data rights would be a problem.

5.2.2.2 Automated design language. An approved automated design language capable of being implemented on the development computer shall be employed in the software design of the subject effort except as noted in Section 4. (Note: This section applies to an item that is not presently state of the art for general application and is included only for future reference).

COMMENT

* Doubts as to its existence were expressed.

5.3 Programming standards and conventions

5.3.1 Design standards

5.3.1.1 Top down program design and implementation. Computer programs consisting of a logically related set of segments shall be arranged as follows: Programs excluding those excepted in Section 4 shall be designed and implemented in a top down hierarchical manner, where the levels of the hierarchy correspond to levels of control of the tasks performed by the program. The essential characteristic of a top down hierarchical design is that each level of detail of the program is logically complete in itself. It should be noted that some segments may appear on more than one level in order to retain logical completeness at each level. Where useful, for early testing or for other reasons, a segment may be coded out of turn after the design hierarchy has been established. The decision(s) to code critical segments at subsumed levels of the software system hierarchy shall be fully justified in accordance with approved software configuration management procedures.

COMMENT

* This section was described as difficult to read and comprehend. It was questioned whether or not there were unnecessary assumptions.
* Another comment was that multitasking was a problem.

5.3.2 Language and coding standards

5.3.2.1 Structured coding. All segments shall be designed with one entry and one exit except for that software noted in Section 4 and shall be written using only the following control constructs. These constructs may be built using logically equivalent language simulations (See Figure 3); i.e., code in the language used shall follow the flowcharts depicted in Figure 3 without explicitly using the names of the constructs in the code. [Any expansion or exceptions to these constructs must be negotiated with and approved by the procuring agency.]

CONTROL CONSTRUCTS

a.	SEQUENCE	Sequence of two or more operations
b.	IFTHENELSE	Conditional branch to one of two operations and return
c.	DOWHILE	Operation repeated while a condition is true. Test is before operation.
d.	DOUNTIL	Operation repeated until a condition becomes true but test is after operation.
e.	CASE	To select one of many possible cases.

COMMENT

* It was suggested that other constructs be considered
* such as:
* FOR
* WHILEDO
* UNTILDO.
* It was mentioned that there will be a generalized looping construct in DOD-1.
* One contractor stated that exception handling and error handling should be mentioned. "What does one do when an error occurs(division overflow, for example)? This should be discussed."
* There needs to be some work done in this area.
* In the module design, the test conditions for the module should be specified. The single entry, single exit provision creates a problem if there is no error exit.
* There should be an exception in here for error condition handling, both real time and others.
* They are handled differently.

5.3.2.2 Structured coding with preprocessor/precompiler. All code written in the course of this effort except for that noted in Section 4 shall conform to the constraints in 5.3.2.1 except that language simulations to form the constructs shall not be allowed. A language preprocessor/precompiler shall be implemented if the language used does not contain Structured Programming control logic to process the constructs cited in Sec. 5.3.2.1. The preprocessor/precompiler will check the source code syntax and translate it into compilable code prior to invoking the source compiler. Preprocessor/precompilers and their related documentation developed under this or any other government contract shall be delivered with unlimited data rights. (See Sec. 3.6)

COMMENT

* This was felt to be a software Quality Engineering
* function, not to be done by a developing unit.
* "If one can't police it, it should be removed."
* One contractor opposed the use of precompilers. He felt
* that the language should be used in its natural state.
* Another contractor felt that the clause "...shall be de-
* livered with unlimited data rights" should be relaxed
* if necessary.
* Another ocntractor felt that this section is too re-
* strictive.

5.3.2.3 Structured programming language. All code written in the course of this effort except for that noted in Section 4 shall conform to the constraints presented in 5.3.2.1 except that language simulations to form the constructs shall not be allowed and the code shall be written in a language which compiles the constructs directly without using an intervening preprocessor/precompiler.

(NOTE: Since none of the Higher Order Languages on the approved list (See Sec. 5.1.1.) contain verbs such as DOWHILE, IFTHENELSE, etc., do not apply Sec 5.1.1. and the above section on the same effort.)

COMMENT

* It was observed that JOVIAL contains IFTHENELSE.
* One contractor felt that this section is too restric-
* tive.

5.3.3 Coding conventions

COMMENT

* One contractor observed that nothing was said in this
* section about Data Typing.
* He felt that one needs to specify standards on data
* types.

5.3.3.1 Basic coding conventions. All code written in the course of this effort except for that noted in Section 4 shall conform to the following conventions:

a. All segments of actual code exclusive of annotation shall be one standard printout page or less in length.

COMMENT

* One contractor felt that the requirement that all seg-
* ments exclusive of annotations be no longer than one
* page in length should be deleted. The length of a mod-
* ule should be determined by its function. To arbitrar-
* ily specify modules should be no longer than one page
* long creates unnecessary interfaces and a fragmented
* system.

b. All code shall be indented to clearly denote logical levels of constructs.

c. All segments shall have sufficient annotation, i.e. comments, to explain inputs, outputs, branches and other items not obvious in the code itself. Explanatory notes shall be uniformly indented and offset by blank lines.

d. Statements (i.e., Format) shall be grouped in each area in the segment to simplify debugging and maintenance.

COMMENT

* It was felt that this was unclear and language-dependent.

e. Data declarations shall be grouped and arranged in a meaningful order in the code, e.g. columnar rather than a horizontal string.

f. Data names and procedure labels shall be meaningful.

COMMENT

* It was felt that one could not always do this.

g. Each line of source code shall contain one statement only.

COMMENT

* A suggested change to this statement was - "Individual
* syntactical elements shall not be split across multiple
* lines."

5.3.3.2 Coding conventions with code reading. All items in 5.3.3.1 shall be implemented with the addition of a procedure for code reading where each page of code shall be submitted to another programmer for review and suggested revisions and corrections.

COMMENT

* This was thought to be too vague. The suggestion was
* made that one should insert how often - prior to the
* first compilation, after a clean compilation, etc.?
* It was thought by another contractor to be a software
* Quality Engineering function.

5.3.3.3 Coding conventions with design/code review. All items in 5.3.3.2 shall be implemented with the addition of scheduled reviews during the design and coding phases where groups working on the subject project meet and review the design or code completed. The intent of such meetings is to generate constructive comment and information.

5.3.3.4 Coding conventions with inspections. All items specified in Section 5.3.3.1 shall be implemented with the addition of inspections for error detection at the conclusion of the design and coding phases. Inspectors use lists of common types of errors to focus on. (See Appendix 70 for an example of common errors and a description of the specialized inspection team)

COMMENT

* One contractor objected to the fact that informal was
* meant but not stated.

5.4 Management aids and quality control data

5.4.1 Software Project Resource Requirements Estimation. The contractor shall prepare and submit a Project Resource Requirements Estimation document appropriate for this project. Resources, as a minimum, shall consist of cost, core space required and execution time. This document is required in the definition phase of the subject effort.

COMMENT

* It was suggested that the contractor does not want to
* give this information to the customer.
* One contractor asked - "Why does the Air Force need
* core requirements when most companies use virtual mem-
* ory?"

5.4.2 Programming Support Library (PSL). The PSL to be used throughout the software development project shall be built as specified herein [unless an existing PSL or an alternate design is specifically agreed upon as an acceptable functional equivalent by the procuring agency and the contractor]. The PSL and its related documentation shall be delivered to the government with unlimited data rights. (See Sec. 3.6)

COMMENT

* One contractor felt that this was applicable only for
* large scale jobs, and that it would be very costly.
* Another felt that the current state of the art for
* interactive programming does not include automating

* auditing trail capabilities.
* One contractor felt that the word "unlimited" should be
* "negotiated".
* Another suggested that the clause "...shall be delivered
* to the government with unlimited data rights" should be
* relaxed if necessary.

5.4.3 Software Reliability Measurement. Error data shall be collected by the contractor in the course of subject software development and delivered to the government with unlimited rights. (See Sec.3.6) [Formats and other details of this collection process shall be proposed to, and approved by, the government.] Appendix 60 contains sample content and schema for reference.

COMMENT

* One contractor stated that the error data is too diffi-
* cult to collect.
* Another suggested that this should be collected auto-
* matically. Otherwise, it can not practically and reli-
* ably be done. The consensus from those interviewed was
* that this section should be omitted from the Specifica-
* tion.

5.4.4 Personnel organization requirements

5.4.4.1 Modified programmer team. Programmer personnel employed in the subject effort shall be organized in programmer teams consisting of six programmers, one of which is designated the leader. Where required personnel exceed one team, an overall chief programmer shall be designated to coordinate the work. In addition, one person shall be designated as librarian. [Modifications to the above shall be proposed to the government for approval.]

5.4.4.2 Programmer team. Requirements specified in 5.4.4.1 shall be implemented except that the contractor shall not be allowed to make any modifications. Programming personnel shall consist of programmer teams, team leaders, and a full-time librarian.

COMMENT

* Contractors were practically unanimous in their opinion

* that the Government is being too restrictive in telling
* the contractor how to organize his programming effort.
* Typical comments were -
* "The number six should be taken out."
* "Why six members on programmer teams? Why not leave
* it up to the contractors?"
* "The new, approved CPDP asks a lot of questions about
* how a contractor manages his programming teams."
* "Staffing and organization as specified is a problem."
* "We are skeptical that the IBM way (as represented
* in this Specification) is really the best way."
* Another contractor stated that -
* "the requirement that a programming team shall con-
* sist of six programmers is much too restrictive.
* The size, composition, and organization of the pro-
* gramming teams must be flexible to suit the scope
* of the software project and the contractor personnel
* resources."

5.4.o Computer Firmware (Microprocessors and Microprogramming Requirements). Computer programs or microprograms that are loaded in a class of memory (read-only-memory (ROM), programmable ROM (PROM) or writeable control store that cannot be dynamically modified by the computer during processing shall be considered firmware. Since firmware consists of a composite of programs and memory, documentation of the memory shall be included in the hardware specification to which it pertains and the computer program or microprogram shall be included in a computer program specification and shall be subject to all software constraints and requirements. Design decisions to implement computer programs and computer data bases in firmware shall be subject to approval by the Government.

COMMENT

* The maintainability and documentation of firmware was
* questioned.
* It was suggested that the Users Guide should comment on
* the current state of affairs in microprocessor technol-
* ogy (No Higher Order Language, etc.)

a. Microprogramming. For the purpose of this effort, the Government considers microprogramming to be a method of implementing the control functions of a processor (microprocessor or large scale processor). Modifications or additions to the control functions (microprogramming) shall be subject to review and approval by the Government prior to implementation. All microprogramming shall be compliant with requirements concerning maintainability, reliability, and documentation.

COMMENT

* One contractor stated that the requirement that micro-
* programming be subject to all the documentation and
* maintainability standards of operational software is ex-
* cessive and should be relaxed. The function performed
* by the microcode is different from operational software,
* the customer typically does not need to change the
* microcode, and the requirements of complete documenta-
* tion and update capabilities would be very costly.

b. Computer Programs for Microprocessors. Computer programs (software) intended for execution on a microprocessor shall be considered to have the same requirements as computer programs intended for execution on large scale processors. Such computer program implementations shall be subject to all software constraints and requirements. Design decisions to implement computer programs and computer data bases in microprocessors shall be subject to approval by the Government. All microprocessor implementations shall be compliant with requirements concerning maintainability, reliability, and documentation.

COMMENT

* One contractor pointed out that computer programs in-
* tended for execution on microprocessor systems typically
* are not written in an approved higher order language.
* Consequently, all the requirements for a higher order
* language program on a large scale processor may not be
* applicable for the microprocessor program. The stated
* requirement should be relaxed.
* The observation was made that firmware and microelec-
* tronics are not associated with each other. Also, Field
* Programmable Logic Arrays should be included.

5.5 Software security procedures. Wherever applicable, software security procedures as presented in DOD 5200.28-M shall be implemented. The multilevel security requirement and the associated requirements for insuring multilevel security operation of the system, as stated in Appendix 80 to this specification, are hereby incorporated into this paragraph.

COMMENT

* One contractor pointed out that security requirements
* should be changed to encompass possible use of the
* National Bureau of Standards encryption algorithm.

5.6 Documentation standards

As a minimum, the contractor shall be required to document the software development according to MIL-STD-483 or DOD 7935.1-S, modified. Required modifications to the above documents are presented in full in Appendix 10 of this specification. The principal thrust of the modifications is to permit graphic representations such as logic diagrams and process charts in lieu of low level conventional flow charts.

COMMENT

* The observation was made that MIL-STD-483 is very ex-
* pensive. Also, it is not really appropriate and may not
* be desirable .
* Compilers must be documented to MIL-STD-483 standards.
* Newly created ones must conform, but it was felt that
* old ones should not be required to have MIL-STD-483
* documentation as a minimum.
* The suggestion was made that if it's already in opera-
* tion, then MIL-STD-483 should not be required.
* Another contractor stated that the requirement of MIL-
* STD-483 as the minimum documentation standards is much
* too restrictive. In many instances (support software,
* diagnostics, tools, and test drivers) MIL-STD-483 is not
* required or even desirable. Contractor format documen-
* tation should be included as an acceptable format in
* these instances.

5.7 Operating systems and utilities services (OSS). An operating system for the computer upon which effort software will be developed, and if different, the computer upon which the software will be maintained for the subject effort shall include, as a minimum, the following:

COMMENT

* The observation was made that not all operating systems
* have all these features. However, they are required
* herein as a minimum.

5.7.1 OSS Basic Requirements. If subject effort is implemented using a higher order language, OSS shall include as a minimum the following features:

a. Processor management

b. Memory management

c. Device management

d. Hardware configuration management

e. Information management

COMMENT

* One contractor suggested that the above features are not
* consistent with the characteristics of Higher Order
* Languages.

Utility and support functions for the OSS shall include the following capabilities when feasible:

a. Compilation

b. Assembly which produces relocatable object code

c. Linking type loader

COMMENT

* It was mentioned that the above three features are soft-
* ware tools.

5.7.3 OSS With Security Requirements. OSS, in addition to all requirements cited in Sec. 5.7.1., shall provide the constraints and controls consistent with DoD military security procedures and the multilevel security requirement of Appendix 80. This section may be used with or without Sec. 5.7.2.

COMMENT

* It was observed that the Specification does not allow use
* of the "NBS Encryption Algorithm" when it becomes avail-
* able. One contractor thought it was excluded because it
* is not included.

5.8 National Software Works (NSW). The contractor shall provide a list of NSW facilities he will employ on this effort for approval by the government. Use of National Software Works facilities require that the OSS shall provide access to the Advanced Research Projects Agency (ARPA) network.

COMMENT

* It was suggested that the AUTODIN II network is replacing
* the ARPANET.
* It was felt that inclusion of the NSW is too early. That
* is, contractors won't be using it for some time. There-
* fore, the suggestion was made to delete it from the Spec-
* ification and move it to the Users Guide for future use.

SECTION V SUGGESTIONS FOR FUTURE WORK

Three areas of potential future work are suggested below:

1) Rewrite Specification and Users Guide to produce version E, incorporating government and industry feedback subject to RADC approval.

The Specification and Users Guide have historically changed as they have been exposed to new viewers and views and these opinions fed back to the responsible RADC technical personnel. The work reported from this current contract represents the broadest exposure to date of these important documents. Thus, the feedback should continue the evolutionary pattern of change for improvement. Obviously, not all of the changes proposed by government and industry will be acceptable to RADC. Some of them are contradictory. It would make a great deal of sense to work with RADC personnel in discussing and thence changing the Specification and Users Guide to produce version E.

2) Help to introduce the Specification to other areas of government.

From those branches of the government which did reply to this survey, it is obvious that the need is great for a version of the Specification and Users Guide to be used in other government agencies. One would hope that there would not be an unbridled proliferation of these documents with inconsistent and careless local modifications. However, tailored versions which preserve the integrity and content of the Specification's proven philosophy and procedures do appear to be much-needed.

3) Develop training programs to educate other government agencies as well as contractors in the use of the Specification.

One of the most often-expressed fears by industry is that these documents will be subjected to uninformed usage by overenthusiastic contract procurement specialists who will overspecify a project. The results would be unnecessary expense and unhappiness on both sides of the contracting table. These fears are not unfounded and the most reasonable solution to the potential problem is training. Training programs must be established for both the contracting personnel and technical personnel. These programs, properly authorized and produced, should go far toward helping these important documents achieve their objectives of helping to provide computer software which reflects system requirements within programmed costs.

APPENDIX A

LISTING OF GOVERNMENT SOURCES

GOVERNMENT

DEPARTMENT OF THE AIR FORCE

Secretary of the Air Force

Assistant Secretary of the Air Force
(Financial Management)

Assistant Secretary of the Air Force
(Installations and Logistics)

Assistant Secretary of the Air Force
(Research & Development)

Acting Assistant Secretary of the Air Force
(Manpower and Reserve Affairs)

Assistant Chief of Staff, Communications and
Computer Resources

Assistant Chief of Staff, Studies & Analysis

Deputy Chief of Staff, Plans and Operations

Deputy Chief of Staff, Programs and Resources

Deputy Chief of Staff, Research and Development

Deputy Chief of Staff, Systems and Logistics

Director of Command, Control, and Communications

Director of Data Automation

Director of Space Systems

Commander, Aerospace Defense Command

Commander, Air Force Logistics Command

Commander, Air Force Systems Command

Commander, Military Airlift Command

Commander, Tactical Air Command

DEPARTMENT OF THE AIR FORCE (Cont.)

Commander, Strategic Air Command

Commander, Air Force Communications Service

Commander, Air Force Data Automation Agency

Director, Air Force Cambridge Research Laboratories

Director, SAMTEC

DEPARTMENT OF AGRICULTURE

Secretary of Agriculture

Director, Office of Automated Data Systems

DEPARTMENT OF THE ARMY

Secretary of the Army

Assistant Secretary of the Army
(Civil Works)

Assistant Secretary of the Army
(Financial Management)

Acting Assistant Secretary of the Army
(Installations and Logistics)

Acting Assistant Secretary of the Army
(Manpower & Researve Affairs)

Assistant Secretary of the Army
(Research & Development)

Armament Research & Development Command

Depot System Command

Electronics Command

International Logistics Command

Missile Material Readiness Command

Missile Research & Development Command

DEPARTMENT OF THE ARMY (Cont.)

Tank-Auto Research & Development Command

Test and Evaluation Command

DEPARTMENT OF COMMERCE

Secretary of Commerce

Assistant Secretary for the Administration

Assistant Secretary for Science and Technology

Bureau of Census

Chief Economist of the Department

Commissioner of Patents and Trademarks

National Bureau of Standards

National Oceanic and Atmospheric Administration

National Technical Information Service

Office of ADP Management

Office of Telecommunications

DEPARTMENT OF DEFENSE

Secretary of Defense

Assistant Secretary of Defense
(Atomic Energy)

Assistant Secretary of Defense
(Communications, Command, Control, and Intelligence)

Assistant Secretary of Defense
(Manpower, Reserve Affairs, and Logistics)

Assistant Secretary of Defense
(Program Analysis and Evaluation)

DEPARTMENT OF DEFENSE (Cont.)

Acting Deputy Assistant Secretary of Defense (Procurement)

Armed Services Procurement Regulation Committee

Director of Defense Research & Engineering

Defense Advanced Research Projects Agency

Defense Communications Agency

Defense Logistics Agency

Defense Mapping Agency

Defense Nuclear Agency

Defense Security Assistance Agency

ENERGY RESEARCH & DEVELOPMENT ADMINISTRATION

Acting Administrator

ENVIRONMENTAL PROTECTION AGENCY

Administrator

FEDERAL ENERGY ADMINISTRATION

Administrator

GENERAL SERVICES ADMINISTRATION

Administrator

Automated Data and Telecommunications Service

Office of Data Systems

DEPARTMENT OF HEALTH, EDUCATION, AND WELFARE

Secretary of Health, Education, and Welfare

Division of ADP and Telecommunications Resource

Division of Management Information Systems

DEPARTMENT OF HOUSING AND URBAN DEVELOPMENT

Secretary of Housing and Urban Development

DEPARTMENT OF THE INTERIOR

Secretary of the Interior

Office of ADP and Telecommunications Management

U.S. Geological Survey

DEPARTMENT OF LABOR

Secretary of Labor

Assistant Secretary of Labor

Departmental Computer Center

DEPARTMENT OF JUSTICE

Attorney General of the United States

OFFICE OF MANAGEMENT AND BUDGET

Director

DEPARTMENT OF THE NAVY

Secretary of the Navy

Assistant Secretary (Manpower, Reserve Affairs, & Logistics)

Assistant Secretary (Research & Development)

DEPARTMENT OF THE NAVY (Cont.)

Center for Naval Analysis

Command, Control, and Communications Programs

Chief of Naval Operations

Commander in Chief, U.S.Naval Forces Europe

Commander in Chief, U.S.Atlantic Fleet

Commander in Chief, U.S.Pacific Fleet

Chief of Naval Material

Military Sealift Command

Naval Air Systems Command

Naval Electronic Systems Command

Naval Facilities Engineering Command

Naval Intelligence Command

Naval Sea Systems Command

Naval Security Group Command

Naval Supply Systems Command

Naval Telecommunications Command

Commandant of the Marine Corps

Atlantic Undersea Test and Evaluation Center

Naval Research Laboratory

Naval Ship R&D Center

Naval Ship R&D Laboratory

Naval Training Devices Center

Naval Weapons Center

DEPARTMENT OF THE NAVY (Cont.)

Naval Weapons Laboratory

Office of Naval Research

Pacific Missile Range

NATIONAL INSTITUTES OF HEALTH

National Cancer Institute

National Heart and Lung Institute

Division of Computer Research & Technology

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Administrator

Ames Research Center

George C. Marshall Space Flight Center

Goddard Space Flight Center

Hugh L. Dryden Flight Research Center

John F. Kennedy Space Center

Langley Research Center

Lewis Research Center

Lyndon B. Johnson Space Center

Wallops Flight Center

DEPARTMENT OF STATE

Secretary of State

DEPARTMENT OF TRANSPORTATION

Secretary of Transportation

DEPARTMENT OF TRANSPORTATION (Cont.)

Federal Aviation Administration

United States Coast Guard

DEPARTMENT OF THE TREASURY

Secretary of the Treasury

UNITED STATES POSTAL SERVICE

Chairman of the Board

VETERANS ADMINISTRATION

Administrator of Veterans Affairs

OTHER AGENCIES, CENTERS, AND LABORATORIES

Argonne National Laboratory

Brookhaven National Laboratory

Draper Laboratories

Jet Propulsion Laboratory

Johns Hopkins University

Lincoln Laboratory

Mitre Corporation

National Center for Atmospheric Research

National Security Agency

National Space Technology Laboratories

Nuclear Rocket Development Station

Oak Ridge National Laboratory

Pacific Northwest Laboratory

OTHER AGENCIES, CENTERS, AND LABORATORIES (Cont.)

Savannah River Laboratory

Smithsonian Astrophysical Observatory

APPENDIX B

LISTING OF INDUSTRIAL SOURCES

INDUSTRY

Accounting Corp. of America

ACTS Computing Corp.

Advanced Computer Techniques Corp.

Aerojet General Corp.

Aerospace Corp.

American Information Services

American Management Systems, Inc.

AMPEX Corporation

The Analysts, Inc.

Applied Data Research, Inc.

ARO, Inc.

Automated Information, Inc.

Automatic Data Processing, Inc.

Automation Industries, Inc.

AVCO Corp.
Greenwich, Ct.

AVCO Everett Research Laboratory
Everett, MA.

Battelle Memorial Institute

BDM Corp.

R.W.Beck & Assocs.

The Bendix Corp.
Teterboro, NJ

The Bendix Corp.
Southfield, MI

Boeing Company
Dover, NJ

Boeing Company
Seattle, WA

Bolt, Beranek, & Newman

Bowne Time Sharing

Brandon Applied Systems, Inc.

Bunker Ramo Corp.
Trumbull, CT

Bunker Ramo Corp.
Hinsdale, IL

Bunker Ramo Corp.
Oak Brook, IL

Burroughs Corporation

California Computer Products, Inc.

Calldata Systems, Inc.

Calspan Corp.

Centronics Data Computer Corporation

CHI Corp.

Chrysler Corp.

CIG Computer Products, Inc.

Citizens National Bank

Collins Radio Corp.

Commercial Computer Service, Inc.

Comptek Research, Inc.

Compu-Serv Network, Inc.

Computer Communications, Inc.

The Computer Co.

Computer Data Systems, Inc.

Computer Dynamics, Inc.

Computer Horizons Corp.

Computer Network Corp.

Computer Usage Co.

Comserv Corp.

COMSHARE, Inc.

COMTEN, Inc.

Control Data Corporation

Cutler Hammer, Inc.

Cutler-William, Inc.

The Cyphernetics Corp.

Data General Corporation

Data 100 Corporation

Datapoint Corporation

Data Systems

Data Systems Analysts, Inc.

Data Technology Industries

Datatronic Systems Corp.

Decision Data Computer Corp.

Deltak, Inc.

Digital Equipment Corp.

Diversified Computer Applications

Dynalectron Corp.

Ebasco Services, Inc.

EDP Consultants

Edutronics Systems International

EG&G, Inc.

Electronic Data Processing Corp.

Electronic Data Systems Corp.

Electronic Memories & Magnetics Corp.

ESL, Inc.

E Systems, Inc.

Evans & Sutherland Computer Corp.

Fairchild Industries, Inc.

Federal Electric Corp.

Florida Software Services, Inc.

FMC Corp.

Foxboro Co.

Garrett Corp.

General Automation, Inc.

General Dynamics Corp.

General Electric Company

General Instrument Corporation

General Motors Corp.

General Research Corp.

GEOCOM, Inc.

Gilbert Assocs., Inc.

Goodyear Aerospace Corp.

Greyhound Computer Corporation

Grumman Aerospace Corp.

GTE Corporation

Harris Corporation

Hazeltine Corp.

Hercules, Inc.

Hewitt Assocs.

Hewlett-Packard Company

Honeywell Incorporated
Minneapolis, MN

Honeywell Information Systems
Waltham, MA

ICM Computer Corp.

Inforex, Incorporated

Information Science, Inc.

INSCO Systems Corp.

Integrated Data Corp.

International Business Machines Corporation

International Computaprint Corp.

International Telephone & Tel. Corp.

International Timesharing Corp.

Itel Corporation

Keane Assocs., Inc.

Keydata Corp.

Keystone Computer Assocs., Inc.

Lear Siegler, Inc.

Litton Systems, Inc.

Lockheed Aircraft Corp.

Lockheed Electronics Co., Inc.

Mainstem Corp.

Management and Computer Services, Inc.

Management Assistance, Inc.

Management Information Service

Manpower Data Services

Manufacturing Data Systems, Inc.

Martin Marietta Data Systems

Martin Marietta Corp.
Rockville, Md.

McDonnell Douglas Automation Co.

McDonnell Douglas Corporation
St. Louis, MO

Memorex Corporation

Mohawk Data Sciences Corp.

3M Company

Motorola, Inc.

MRI Systems Corp.

Mylee Digital Sciences, Inc.

National Data Communications, Inc.

NCR Corporation

NLT Computer Services Corp.

Northrop Corp.

OPOC Computing, Inc.

On-Line Systems, Inc.

Pansophic Systems, Inc.

Perkin Elmer Corp.

Pertec Corporation

Physics International Co.

Pinkerton Computer Consultants

Planning Research Corp.

POLA Data Assocs., Inc.

RAND Computation Center
Santa Monica, CA

RAND Corporation
Santa Monica, CA

Rapidata, Inc.

Raven Data Processing, Inc.

Raytheon Company

RCA Corp.

Recognition Equipment, Inc.

REHAB Computers, Inc.

Rockwell International Corp.

ROHR Industries, Inc.

RTW Management Systems

SAID, Inc.

Sanders Associates, Inc.

SC Data Center, Inc.

Science Applications, Inc.

Scientific Time Sharing Corp.

The Service Bureau Co.

Singer Co.

Sperry Rand Corporation

Standard Computer Corp.

Standard Information Systems, Inc.

Stanford Research Institute

STAT:COM

STC Systems, Inc.

Storage Technology Corporation

Sycor, Inc.

Syntonic Technology, Inc.

Systems Development Corp.

Systems Consultants, Inc.

Systems Engineering Computer Co.

Systems Research Laboratories, Inc.

Tektronix, Inc.

Teledyne Industries, Inc.

Teletype Corporation

Telex Corporation

TESDATA Systems Corp.

Total Typography, Inc

Transport Data Communications, Inc.

Trilog Assocs., Inc.

TRW Data Systems
Hawthorne, CA

TRW Incorporated
Cleveland, OH

Texas Instruments, Inc.

Textron, Inc.

Tymshare, Inc.

United Computing Corp.

United Computing Systems, Inc.

United Technologies Corp.

University Computing Co.

Varian Associates

Volt Information Sciences, Inc.

Vought Corp.

Wang Laboratories, Inc.

Weiland Computer Group

Western Electric Co.

Western Union Data Services Co.

Westinghouse Eletric Corp.

Wyly Corporation

Xerox Corporation

ASSOCIATIONS

American Association for the Advancement of Science (AAAS)

American Consulting Engineers Council (ACEC)

American Federation of Information Processing Societies (AFIPS)

American Institute of Aeronautics and Astronautics (AIAA)

American Mathematical Society (AMS)

American National Standards Institute (ANSI)

Association for Computing Machinery (ACM)

Association of Computer Programmers and Analysts (ACPA)

Association of Data Processing Service Organizations (ADAPSO)

Association of Time-Sharing Users (ATSU)

Computer and Business Equipment Manufacturers Association

Data Processing Management Association (DPMA)

Digital Equipment Users Society (DECUS)

Guidance for Users of Integrated Data Processing Equipment

Institute of Electrical and Electronics Engineers (IEEE)

International Federation for Information Processing (IFIP)

National Society of Professional Engineers (NSPE)

National Technical Services Association (NTSA)

Operations Research Society of America (ORSA)

Society for Industrial and Applied Mathematics (SIAM)

Society of Data Processing Consultants

APPENDIX C

INITIAL LETTERS FROM GOVERNMENT AGENCIES

DEPUTY CHIEF OF STAFF
TO
COMMANDER IN CHIEF UNITED STATES ATLANTIC FLEET
NORFOLK, VIRGINIA 23511

7 March 1978

Mr. M. Landes
Headquarters Rome Air Development Center (AFSC)
Griffiss Air Force Base, New York 13441

Dear Mr. Landes:

Admiral KIDD asked me to answer your letter of 6 January 1978 regarding your development of a computer software specification for use in the USAF Rome Development Center's procurement process.

This command is interested in participating in a survey which may decrease the high cost of computer software development while assuring the applications software accurately reflects mission requirements.

I would appreciate your forwarding the questionaire and copies of the Specification and Users Guide to:

Commander in Chief U. S. Atlantic Fleet
Attn: Mr. James F. Moran (022T)
Norfolk, Virginia 23511

Sincerely,

Wm. A. Myers III

William A. Myers III
Rear Admiral, U. S. Navy

DEPARTMENT OF THE AIR FORCE
HEADQUARTERS UNITED STATES AIR FORCE
WASHINGTON, D.C.

16 FEB 1978

SA

Software Development Specification Survey

HQ Rome Air Development Center (AFSC)
(Mr. Michael Landes)
Griffiss AFB, N.Y. 13441

1. It is with enthusiasm that we at Air Force Studies and Analysis accept your invitation to review your document, "RADC Computer Software Development Specification No. CP0787796100C." We are in the process of developing a computer software development specification for use in the procurement of our software and have been giving this subject a great deal of consideration lately.

2. Please send your questionnaire and copies of the Specification and Users guide to Captain Russell C. Hammerud who will coordinate our inputs.

RICHARD L. BLAIR, Lt Col, USAF
Chief, Computer Applications Group
AFS/Studies and Analysis

OFFICE OF THE ASSISTANT SECRETARY OF DEFENSE

WASHINGTON, D. C. 20301

MANPOWER,
RESERVE AFFAIRS
AND LOGISTICS

23 February 1978

Mr. Michael Landes
USAF Rome Air Development Center
Griffiss Air Force Base
New York 13441

Dear Mr. Landes:

The Assistant Secretary of Defense (Manpower, Reserve Affairs and Logistics) has requested that I respond to your letter of 6 January 1978 subject: Software Development Specification Survey.

This office has long recognized the importance of utilizing computer software development specifications in the procurement process and would be most willing to participate in your survey.

The questionnaire and copies of the Specification and Users Guide should be forward to me at the address indicated below.

Mr. Orville Smiley, Director
Automated Systems Office
MRA&L(PD) Pentagon - Room 3C771
Washington, D. C. 20301

Sincerely,

ORVILLE L. SMILEY, Director
Automated Systems Office
MRA&L(PD)

NATIONAL SECURITY AGENCY
CENTRAL SECURITY SERVICE
FORT GEORGE G. MEADE, MARYLAND 20755

Serial: N 0144

14 FEB 1978

SUBJECT: Software Development Specification Survey

TO: Commander
Headquarters Rome Air Development Center (AFSC)
ATTN: Mr. Michael Landes, Project Manager
Griffiss Air Force Base, New York 13441

1. As mentioned in your letter, subject as above, dated 6 January 1978, the National Security Agency would be pleased to participate in your survey of Government Agencies to obtain reaction to your Computer Software Development Specification. We have previously obtained a portion of your Specification and would be very interested in reviewing the entire Specification and Users' Guide.

2. The National Security Agency is also concerned about improving the software acquisition process and has a small effort underway to establish and implement policies, procedures and standards for software development. The studies and manuals that RADC and other AFSC Organizations have published on modern programming practices and software acquisition are very useful inputs to this effort.

3. Please send your questionnaire and copies of the Specification and Users' Guide to Mr. Donald H. Johnson. His mailing address is:

Director
National Security Agency/Chief, Central Security Service
ATTN: Mr. Donald H. Johnson, T303
Fort George G. Meade, Maryland 20755

Mr. Johnson may be reached by telephone on 301-688-7691.

for Kenneth H. Sperman

E. S. INCE
Rear Admiral, U. S. Navy
Deputy Director for Telecommunications
and Computer Services

DEPARTMENT OF THE NAVY
HEADQUARTERS NAVAL MATERIAL COMMAND
WASHINGTON, D. C 20360

IN REPLY REFER TO
09Y/JDC
Ser 60

FEB 15 1978

Mr. Michael Landes
Code ISIM
Headquarters, Rome Air Development Center
Griffiss Air Force Base, NY 13441

Dear Mr. Landes,

Thank you for your letter of 6 January 1978 inviting the Naval Material Command to participate in the survey of the RADC Computer Software Development Specification. The Chief of Naval Material concurs with your perception of the need to reduce the high cost of software development and maintenance. Such efforts as the RADC Computer Software Development Specification and MIL-STD-1679 (Navy), a proposed military standard for tactical software development, are promising steps toward this goal.

The Naval Material Command will be pleased to participate in your survey. Five copies of the questionnaire and other material would be appreciated to allow a wider review. The CNM representative will be:

CDR A. A. Pedersen
HQ Naval Material Command
Washington, D.C. 20360
Attn: MAT 09Y
AUTOVON 222-8615

Please consider this letter as the response to your 6 January 1978 letters to the Assistant Secretary of the Navy (M,RA&L), The Chief of Naval Operations, and the Director, Command and Control and Communications Programs (OP 094), as well as the letter to the Chief of Naval Material.

Sincerely,

E J Reiher

E. J. Reiher
Captain U. S. Navy
By direction of the
Chief of Naval Material

DEPARTMENT OF THE NAVY
NAVAL INTELLIGENCE COMMAND
2461 EISENHOWER AVENUE
ALEXANDRIA, VA. 22331

IN REPLY REFER TO
Ser 00Q/38

21 FEB 1978

From: Commander, Naval Intelligence Command
To: Department of the Air Force
Headquarters Rome Air Development Center (AFSC)
Griffiss Air Force Base, New York 13441

Subj: Software Development Specification Survey

Ref: (a) RADC (AFSC) ltr ISIM, dtd 6 Jan 1978

1. Reference (a) invited the Naval Intelligence Command to participate in a review of the RADC Computer Software Development Specification No. CP0787796100C.

2. This command is pleased to have the opportunity to provide input to the specification with a view toward reducing the cost of software development and acquisition through the use of a viable software procurement guidance tool.

3. Points of contact for the survey are Messrs. Lawrence E. Towner and Ronald K. Gray, NIPSSA-51 and 52, respectively. Both may be contacted on 8-221-0760 (AUTOVON) or 202-325-0760.

F. A. HULL
By direction

National Aeronautics and
Space Administration

George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama
35812

NASA

DM01

FEB 16 1978

Mr. Michael Landes
Project Manager, ISIM
Headquarters Rome Air Development Center (AFSC)
Griffiss Air Force Base, New York 13441

Dear Mr. Landes:

Marshall Space Flight Center shares in the concern for the high cost of computer software development and wishes to participate in the survey. The questionnaire and copies of the Specification and Users Guide may be addressed to:

Mr. Bobby C. Hodges
Huntsville Computer Complex, AH31
Computer Services Office
George C. Marshall Space Flight Center
Marshall Space Flight Center, AL 35812

Sincerely,

John S. Potate
Associate Director (Management)

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

February 14, 1978

Mr. Michael Landes, Project Manager
Software Development Specification Survey
Department of the Air Force
Headquarter Rome Air Development Center
Griffiss Air Force Base, New York 13441

Ref: USAF RDC memorandum to U.S. Patent and Trademark Office, January 5, 1978

Dear Mr. Landes:

Acting Commissioner of Patents and Trademarks, Lutrelle F. Parker, has asked my office to respond to your January 5 memorandum. The U.S. Patent and Trademark Office would be happy to participate in your survey. The following individual will act as the Patent and Trademark Office interface for the survey:

William L. Spittle
Manager, Systems Development Division
Office of ADP Administration

Address: U.S. Patent and Trademark Office
Office of ADP Administration
CP2 - Room 5D05
Washington, D.C. 20231

Telephone: 557-3646

We appreciate the opportunity to contribute to the resolution of the high cost software development problem.

Sincerely,

George S. Vaveris

George S. Vaveris, Director
Office of ADP Administration

APPENDIX D

GOVERNMENT RESPONSE TO COMMENT SECTION OF QUESTIONNAIRE

I would tend to agree that the programming practices specified in the specification lead to more reliable software which is less expensive over the entire life cycle. I wish I could comment in more detail and share with you some experiences in using the program tools and programming practices which would support my opinion. Unfortunately, we are just beginning to implement standards and procedures requiring the use of most of these techniques. As you can see, we currently use a limited number. However, the techniques which we have installed so far have greatly enhanced our ability to maintain the continuity and integrity of software design and to minimize life-cycle costs. We plan on installing most of the tools and procedures listed in Part II of the questionnaire.

The use of a PDL is fairly new to me. I find the concept, however, most interesting. However, it is not entirely clear to me in its application what the real relationships are between system specifications and design of programs, and the normal activities associated with these functions. It would appear some duplication of effort is involved. However, it may be a more natural form of expression to programming problems.

In regards to the specification, this is the first specification of this type I am aware of devoted exclusively to computer software development procurements. It is a well conceived package and will definitely serve to strengthen the area of costly software development projects. Its use has tremendous benefit potential for both the government and the contractor in the purchase of software. The requirements of the specification are such that it should alleviate the problems in software procurements associated with misunderstandings and misinterpretations which become extremely costly.

COMMENTS:

The employment of the following modern programming practices are in use at the USAF Rome Air Development Center (RADC).

a. High order programming languages
b. Software development aids
c. Programming standards and conventions
d. Management aids and quality control data
e. Software security procedures.
f. Documentation standards
g. National software works facilities

I concur with the idea that good system development practices such as listed above lead to more reliable software and in the long run is less expensive than systems developed without a well planned, developed and controlled approach.

In fact, we are using and have used with the exception of National Software works facilities all of the above listed program development practices both for in-house or contract program developments.

ADDITIONAL COMMENTS:

1. FIPS PUB 42--Guidelines for Benchmarking ADP Systems in the Competitive Procurement Environment--, and FIPS PUB 38--Guidelines for Documentation of Computer Programs and Automated Data Systems, have not been mentioned as applicable source documents on page 3. Are these documents in line with the above practices in context or format and which one of the practices has priority?

2. It is also felt that the details of Appendices 30 and 50 may distract from the main idea of the specification. They should be included in a separate specification or in a "FIPS PUB" publication.

3. It is desired that more specific selection parameters of suitable paragraphs, such as size of program, cost or time factors be provided to the program manager. This type of information is not included in the specification.

ADDITIONAL COMMENTS: (Continued)

4. Discrepancies noted in the specification:

   a. Page 1, par. 1.2 - Figure 1 should be discussed prior to discussing Figure 2.
   b. Page 3, par. 2 - The table column headings typed at the bottom of Page 3 should be typed on the top of page 4.
   c. Page 23 - There are two (2) page 23-s with different text.
   d. Page 25 - Figure 2 should be called Example 1 and discussed as such on page 1.

5. Discrepancies noted in the Users Guide:

   a. Page 4, 21, 39, 72 and 77 - The paragraph headings typed at the bottom of these pages should be typed on the top of the next pages above the appropriate text.

AD-A068 635 WOLF COMPUTER CORP CAMBRIDGE MA F/G 9/2
RADC SOFTWARE SPECIFICATION SURVEY.(U)
MAR 79 W M WOLF F30602-77-C-0173
UNCLASSIFIED RADC-TR-79-25 NL

2 OF 2
AD A068635

END
DATE FILMED
6-79
DDC

The Computer Services Division at Lewis Research Center has been extending a structured programing language precompiler (originally developed by Jet Propulsion Labs) for the past two to three years. This language is based on FORTRAN, and is called SFTRAN (for Structured FORTRAN). The precompiler has been developed for use on Lewis' two large-scale computer systems--the IBM 360/67 system operating under TSS/360, and the UNIVAC 1100/42 system operating under 1100 EXEC. A copy of the users' manual for the 360 version is enclosed for your information.

While we don't have a great deal of experience to use as a basis for our opinion, we believe that the use of structured programing concepts and procedures in our in-house development efforts has produced software that is more reliable, easier to develop, modify and maintain, and is self documenting to a great extent.

We do not feel that any of the techniques in the Specification documents are necessarily contradictory to our procedures. We just have not carried our procedures to the extent indicated in this document. Lewis does much of its own software development in-house. Most of the programs done under contract are by-products of larger research and development efforts. Lewis does not employ a software support services contractor at this time.

The approach advocated in the Specification documents is reasonable for any organization that intends to contract for much of their software. Such a rigorous approach will significantly improve the chances for acquiring good, properly constructed and operating software. The standards, documentation, etc. required by such an approach will likely add significantly to the cost of the software. However, this cost is probably justified.

The general approach advocated is not significantly different from the approach being advocated by NASA on its major programs that require software development.

COMMENTS:

Design Phase

The implementation of a Program Design Language (PDL) is not recommended for all software development activities. The use of HIPO's, Warnier diagrams, and module architecture charts are just as useful in some applications. The specifications seem to assume that clear system functional requirements are available (paragraph 3.5.a). Since this is not always true, a functional design phase must be addressed as a significant part of the complete software design phase. The documentation resulting from the functional design should be carried into the detail software design phase in a manner which provides continuity.

Design, Code, and Test Inspections

The basic contents of Appendix 70 are good, but could be greatly expanded. The concept of formal inspections with a moderator is excellent. However, the type and number of people attending an inspection should vary depending upon the type of inspection (design, code, test) and the nature of the project. Management personnel should not always attend these inspections, but should be aware of their occurrence. To assure that inspections are conducted properly, documentation of the inspections must be required. Appendix 70 does not adequately cover the documentation aspects.

Personnel Organization

From our experiences there is no one best way to organize personnel to develop software. Therefore, a particular method such as the Programmer Team should not be emphasized. The Programmer Team is probably excellent for large projects, but a one man effort, combined with clear goals and formal inspections, is more suitable for small projects such as our internal software projects.

Structured Code

Too much emphasis has been placed on code. Good coding practices should originate with good design and well educated programmers. Properly conducted code walkthroughs should be combined with automated code auditors and strict standards. Coding must, however, be readable and the coding conventions listed in paragraph 5.3.3.1 are very good and should be emphasized.

Experiments at MSFC in the use of precompilers/preprocessors have not been very successful.

Program Support Library

The material presented on the Program Support Library (PSL) is excellent. At MSFC, the use of PSL has proved very effective.

Documentation Standards

The importance of good documentation cannot be over emphasized. Documentation standards must consider the environment in which the software will be used and maintained. A highly interactive user-oriented system should provide online documentation, as well as offline usage and maintenance documentation.

Software Verification and Testing

This very important area has not been adequately covered in the specification. More emphasis has been placed on verifying that structured code has been employed than on verifying that the software performs according to the problem specifications.

Security Requirements

Security considerations must be clearly represented in the functional and detail software design documentation.

In general, the Specification and Users Guide and related applicable documents are overwhelming in size and detail. Their use implies considerable investments in man-hours both by the government agency and by any contractors, to be sure all applicable requirements are understood and met.

For relatively small installations, a much scaled down, self-contained document would be much more useful.

We have developed specifications similar to yours to ensure portability, reliability and maintainability of FORTRAN programs. To date, only one contract has been realeased which incorporates these specifications. The contractor's comments upon using the specifications reinforced our own belief - that the specification allow the contractor to better design the program as well as to produce code which is much easier to read and understand.

In general, I feel that both the Specification No. CP0787796100D and the Users Guide are extremely well done and very valuable. We have developed, at DTNSRDC, a set of nine specifications and three documentation requirements, entitled CASDAC Software Specifications, which are similar to your documents. I feel that your Users Guide helps tie the Specifications together and I wish we had developed a Users Guide to go with the CASDAC Software Specifications. It should be recommended to all contractors that the Users Guide be read prior to the Specifications. The Users Guide is much easier to read than the Specifications and serves as a good introduction to the Specifications.

However, I feel your specification make it too easy to get waivers for certain requirements. Our Specifications require the contractor to show why it is impossible for him to meet our requirements before we allow him to propose alternatives to the requirements.

On page 16 of Specification No. CP0787796100D, you talk about coding conventions. I have the following comments:

1) The term "segments" is used many times. Is it defined anywhere in the Users Guide or Specifications?

2) Our experience has been that indenting lines of code causes problems when modifying the code.

3) In our CASDAC Software Specifications, we have both qualitative and quantitative requirements concerning comment cards. We require that one comment must appear for every five lines of executable code. We require Comment cards at the following places in the code: before all branches, at the place in the code where the branch goes to, after each continue statement (if the continue is the terminal statement of a DO, the comment must explain what condition is true when the DO is satisfied), and preceding each arithmetic function definition explaining purpose. I prefer this approach to saying that comments should explain "inputs, outputs, branches and other items not obvious in the code itself."

4) When you say data names and procedure names should be meaningful, you should give some examples of both meaningul and non-meaningful names.

5) Some of the coding requirements in your specifications are the same as the ANSI requirement (e.g., loop variables shall not be altered during loop execution.) We found that most ANSI requirements for FORTRAN programs were very useful in trying to attain portable programs. I would suggest enforcing the ANSI specification for both FORTRAN and COBOL programs. When you differ with a specific ANSI requirement, the page and paragraph number of the ANSI publication can be noted, and appropriate modification made to it.

We generally agree that there appears to be a plausible correlation between the use of modern programming practices and the reliability and cost of software. As a labor intensive organization wherein labor costs account for more than 85 percent of our budget, the Postal Service must of necessity explore all avenues of potential cost avoidance. Recognizing that the planning, design, development, and implementation of software represents a considerable investment in human, financial and equipment resources, we continually employ various measures which seek to optimize our return on such investments. Among such measures are comprehensive documentation and operations standards based upon an awareness that the successful development of software does not in itself guarantee economical implementation or utilization. As a result of the personnel turnover, multi-vendor hardware and software procurements, and the geographical dispersion associated with our operations, standardized development methodology and documentation procedures are essential to our successful implementation and effective maintenance of software.

In reviewing your specification and users guide, we found some techniques which the Postal Service does not currently employ, such as the program design language, but found no contradictory procedures within the context of the basic specification (referenced documents not included). There are, however, distinct differences between your specification and some of our documentation standards. For example, our standards include installation and application dependent naming conventions and precisely defined requirements for operational systems. Further, in our statements of work we seek to state our requirements as fully, clearly, and precisely as possible but at the same time to not overspecify. We therefore avoid incorporating extraneous material and requirements. With due consideration of the apparent inherent flexibility provided by your specification, it still appears to be appreciably and understandably geared toward Air Force procurement in particular and Department of Defense procurement in general. As a result, Postal Service use of your specification would probably precipitate a revision which would include those salient features deemed most relevant to our requirements. The resulting adaption of your specification would likely be appreciably geared toward Postal Service procurement and computer systems requirements and practices; as we have done with applicable parts of other promulgated standards.

VETERANS ADMINISTRATION
OFFICE OF DATA MANAGEMENT AND TELECOMMUNICATIONS
WASHINGTON, D.C. 20420

JUL 10 1978

IN REPLY REFER TO: 301/31

Mr. William Wolf
Wolf Computer Corporation
238 Main Street
Cambridge, Massachusetts 02142

Dear Mr. Wolf:

We appreciate your consideration in allowing the Veterans Administration the opportunity to review and participate in the Software Development Specification Survey (No. CP 0787796100D).

Our overall evaluation of your specifications reveals that the data is complete and very detailed. We concur that adherence to these standardization techniques might well lead to a "better" means of doing the job. Nothing in your specifications is contrary to those used by the VA, but in some areas, such as structured programing, enforcement of programing support library, and software utility measurements, your notations are more encompassing and complete.

As an agency standard, the Veterans Administration uses COBOL whenever possible; in the past, we have used the techniques and methods presented in your documentation to varying degrees, dependent upon the objectives and environment of a particular installation. A combination of these techniques have been used on every major VA project for the past several years.

Our only reservation with the content of the material involved the inclusion of microprograming/microprocessor (firmware) specifications. It would seem that these specifications might be more effectively presented in a separate document. The exceptions to firmware which were presented within your specifications are so varied that they themselves warrant an independent treatment.

Our general understanding is that just as an assembly language is closer to the actual processor logic (Hardware) than Fortran/COBOL, microcode is still closer -- requiring a greater in-depth understanding of all the functions and limitations of a central processor. This degree of complexity required by the code in use of proper interrelated commands, timing, and data path considerations, lead us to regard the firmware as an integral portion of the hardware which should be under documentational constraint of the same.

SUBJ: Mr. William Wolf PAGE 2

I hope that I have given you some worthwhile feedback. If I can be of any further assistance in this project, please contact me.

Sincerely yours,

H. SEIFERT
Deputy Assistant Administrator for
Data Management and Telecommunications

Enclosures

APPENDIX E

INDUSTRY RESPONSE TO COMMENT SECTION OF QUESTIONNAIRE

Specification and Users Guide Comments

My only specific objection to the spec is the orientation toward IBM and a card/batch environment. I notice that they are based on the IBM Structured Programming Series. Many of the aspects of the program librarian function and the PSL seem to reflect a card/batch environment. In our time-sharing environment the program librarian function is supported by the assignment of project and programmer numbers where the programmers within a project have different privileges within the project. The program librarian is a programmer with the privilege of accessing the program library directory and programmers must go through the librarian to enter new software into the different development libraries.

Program style tends to be different in a time-sharing environment than a card environment. For example, tabs are used, instead of card columns, editing is done differently, line(card) id and sequence numbers come at the beginning of a line(card) instead of the end(cuts down on space and typing time).

The spec should encourage the use of automated documentation aids. In this way documentation can be built from entered text and program generated text. Data entered as documentation can be mapped into a program if standards are adhered to. Documentation entered for one purpose early in the project can evolve into more detailed documentation as the project progresses.

COMMENTARY
SPECIFICATION NUMBER CP 0787796100D
RADC COMPUTER SOFTWARE DEVELOPMENT SPECIFICATION

GENERAL

1. A table of contents would be helpful

2. Make the Users' Guide an Appendix to insure the Specification and Guide remain associated.

SPECIFICATION

1.1 The text of the third sentence starting "In addition, justifiable negotiated..." is more appropriately a part of the contract such as the Statement of Work on the Special Provisions.

1.2 (a) Put the third sentence of first paragraph starting "Depending on size,..." in the Users' Guide. Since this is a specification, administrative guidance is not appropriate in the principal text.

(b) Modify text of third paragraph to be consistent with General comment 2 (Users' Guide as an Appendix).

(c) The third, fourth and fifth paragraphs (all on page 2) are more appropriately part of the Guide.

(d) Data Requirements in the sixth paragraph may need modification or amplification. SAMSO has developed some special clauses on data rights which go in Section J, Special Provisions. These are under review at AFSC and may (should) become the AFSC standard.

2.1 Include RADC Series in Applicable Documents. This specification cannot stand totally on its own since many references are made directly to the Series for clarification or further detail.

3.5a Typo - syntax

3.5h The reader of the specification would be helped if this definition were expanded. It could set the entire tone of the subsequent text.

5. MIL-STD-490 does not reference a Section 3.3.8. Is this a typo?

5.1.3 Do you want to put a comment here about when JOCIT will be available?

5.2.1 (a) Calls for data delivery (list of software aids). Is a Data Item in preparation?

(b) The fourth sentence starting with "Software tools..." is not appropriate here. It should be included in Section L or J of the contract.

5.2.2.1 (a) What is the mechanism of arriving at an "approved" PDL as stated in the first sentence? Text is needed to clarify.

(b) The last sentence should be included in Section L or J of the contract.

5.3.2.2 The last sentence should be included in Section L or J of the contract.

5.4.1 Calls for data delivery (Project Resource Requirements Estimation). Is a Data Item in preparation?

5.4.2 The last sentence should be included in Section L or J of the contract.

5.4.3 Calls for data delivery (error data). Is a Data Item in preparation? Also the legal words about unlimited rights belong in Section L or J of the contract.

5.4.4.1 This paragraph should be tailorable. Six-programmer teams is too rigid a requirement.

5.8 The first encounter with NSW is too abrupt. More words are needed here or an earlier discussion, say in the Definitions section.

APPENDIX 10

Appendix 10, which suggests revision to the documentation standards, does not go into enough detail in the requirements for the computer program listing if it will indeed replace flow charts. Where a structured programming listing is not required, there is still a requirement for detailed flow charts. A more effective method may be to replace this requirement with functional and logic overview charts, each of which is no longer than one page in length.

APPENDIX 20

Cover Page and Line 1: spell out PSL.

APPENDIX 30

The format of this Appendix is not consistent with either the balance of the specification or with general specifications practices. It is a contractor report and the work "report" appears a number of times. The Appendix should be revised to standard format.

3.1.0 Change 4120.17-M to 7935.1-S

APPENDIX 40

This Appendix needs to be rewritten in a clear convincing manner to sell the concept. The PDL technique for improving communication during the design and development of computer software is recommended by a number of people; but others feel somewhat neutral as to its usefulness and so would not require its use on a contract. However, the technique shows promise and much can be learned as PDL is used. Therefore, the text should be modified to encourage use. As is, the purpose of PDL and its use are difficult to follow in the document because of the cumbersome language used. For example the following sentence is taken from 40.3.1, "The not inconsiderable choice left to the program designer here should be based on the goal of maximum understandability for transfer of technical information between diverse groups of managers, programmers, and users." Sections 40.1.0, 40.3.2, and 40.4.0 are particularly in need of rewrite. Finally, the text has not been completely freed of contractor report text. The word "paper" shows up in the third paragraph of 40.1.0, and the first bullet at the top of page A-40-2-2 needs to say "... Section 40.2.2".

The Table of Contents page numbering is incorrect.

40.2.3 The terms "one-in/one-out" and "standard indentation convention of two columns" will not be understood by all readers and should therefore be expressed in everyday language.

40.2.5 What is meant by "structured data"?

40.2.7 Give a reference or provide more information on the "structured theorem program".

APPENDIX 50

This Appendix shows the same symptoms of a contractor's report rather than a specification with the use of the word "report" and with incomplete paragraph referencing.

50.2.1.3 last paragraph: It is agreed that the intent should be to give the user flexibility in choosing procedures and tools. However, there are instances where this freedom allows the user to circumvent desirable requirements. Although it "is not a function of the precompiler to limit coding by preventing the use of language statements which violate structured programming conventions," precautions should be included in the specification to prevent excessive use of the INCLUDE statement to bring unstructured code into the program.

50.3.2 In the second line of text reference is made to a COBOL manual, which is then described in the references on Page A-50-A-1. From a specification integrity point of view, the standalone organization of the Appendix should be modified by moving the references to Section 2.2 (Page 5) and then referencing that Section in the text. This problem occurs several times in this Appendix. And in 50.5.1.2 is some duplication with other portions of the specification.

50.5.4.6 The reference to Figure 5.1 in the third line, although correct, is inconsistent with figure numbering in the other Appendices. The reference, the figure and the diagrams should use consistent Appendix notation.

APPENDIX 60

60.2 Two reports are identified (problem and closure). Are Data Items in preparation?

60.3 Refers to Table 60-1 which was not included in the specification.

APPENDIX 70

This appendix has the same figure labeling problem identified for Appendix 50.

GUIDE

1.2 Last paragraph, Page 9: The guidance promised was not provided, in general.

1.3.3 The most recent Data Item for the CPDP is DI-S-30567A.

5.1.1 This section should cover guidance on degree of assembly language which is acceptable, what to do with subsets and supersets of any HOL, and how the specification will relate to DOD-1 and the new ANSI FORTRAN.

5.1.2 The comment about the ESD guidebook is slightly erroneous. There are two guidebooks, one on Verification and one on Validation/Certification.

Page 25, Top incomplete paragraph: The role of QA needs clarification here. If a module is under test using stubs, how does QA "calibrate" the stubs so that buy-off can occur? An incorrect stub could lead to an incorrect module being placed under library control.

5.6 Can THREADS be considered in the public domain and be specified in this document?

Page 74 ESDP 800-4 is product division peculiar and cannot be considered an acceptable AFSC SOW preparation guide. Recommend this document be removed.

Section F - This section could be removed if the Guide is included as an Appendix.

Comments on Specification CP0787796100D:

(1) The specification appears to be oriented towards serving two distinct purposes:
(a) Defining the system development phases for the contractor, and
(b) Defining the actual procedures to be followed within each phase.
Unfortunately, there is an "uneveness" associated with the topics addressed in CP0787796100D.

(2) The actual phases of developing a large software system are not uniformly defined. There needs to be clearer presentation of the structure of the process. (I have attached the Table of Contents for SDM/70 for your review.)

(3) Once the functions of the phases are more clearly understood, actual procedures or standards can be defined for each one (preferably in separate documents).

(4) Much of Specification CP0787796100D is available commercially for $25K or less.

(5) No consideration is given to an approach which would make significant use of data base management technology. Software development is much different when using DBMS.

GENERAL COMMENTS

RADC Software Specification

The document itself is forbiddingly large--perhaps referencing the appendices would get it to a more manageable size.

The multi-paragraph selection option approach is excellent, as is the overall philosophy and execution of the document.

The specific standards are sometimes carried to an absurd and dangerous level of detail, with examples that are inappropriate to the specification.

Care should be exercised in dictating all the traditional encutrements of structured programming. As a general philosophy, software design appears to benefit from a top down approach. Optimal implementation depends upon many project characteristics however and, particularly for multi-person efforts, often (perhaps usually) benefits from bottom-up development.

USERS GUIDE

The philosophy and specifics of this Users Guide are excellent. The realities of off-the-shelf, modified, and small project software are especially well presented. The small/medium/large distinction, illustrated by example, is admirable.

SPECIFIC COMMENTS AND ERRATA

RADC Software Specification

Section	Page	Commentary
		The Table of Contents and Figures are missing.
5.3.2.1	14	Exceptions to the single entry/exit standard should be allowed for clusters of similar functions operating on a common algorithm, i.e., SIN/COS routine.
5.3.2.2	15	Structured languages and compilers that operate on them should be acceptable as alternatives to preprocessors. Preprocessors often add additional complexity and cost to developmental efforts.
5.3.3.1c	16	Suggestion of indentation and/or blank line offsets, or other technique that promotes clarity is as far as the specification should go. The requirement is for clarity and sufficiency, not for indentation.
5.3.3.1g	16	The objective of clarity and continuity is often enhanced by placing compact, functionally-related statements on a single line.

5.3.3.1i 16	Not only should patching be "rigidly controlled", but inhibited as well.
5.3.3.1(1) 17	Self-modifying (adaptive) code should be avoided whenever possible. On computers that use link instructions which jump to n+1 and set the location of the jump into n, it may be unavoidable.

Section Page	Commentary
Appendix 10	The provision for media other than flow charts for logic representations is commendable.
Appendix 60	Holding error categorization to a relatively small but critical number of entries is highly practical.
Appendix 70 Figs. 2 and 3	The examples are ill-chosen inasmuch as they are far too specific, and appear to be assembler rather than HOL oriented.

Users Guide

Page 4 last line	This heading should begin Page 5.
Page 6 heading	Typo error-- Section . . .
Page 9 3rd par.	Paragraph 2 seems to have established a procedure which should be followed in Para. 3.
Sec. 5.1.3 P19	Second sentence typo--Jovial
page 19 3rd paragraph	First sentence-- The list of software development aids in question are GFE?
Page 22, last paragraph	Doesn't read properly.
B.5.3.2.1 24	The restriction to one statement of code is undesirable. More readable code can be achieved by placing compact functionally related statements on a single line. This may interefere somewhat with identation structures.
B.5.3.2.1 25	The reference to "variable values, indirect addressing, and so forth" appears vague and misleading, and is probably an assembler consideration rather than directed toward HOLs.
B.5.3.2.3. 27	The reference to "interpreters to read in structured programming language directly" pertains to computers whose hardware interprets a given (structured) language?

Example	50	The senior programmer count for the 1st option year is wrong.
2nd para.	66	First sentence typo-- . . . to realize . . .
4th line	82	. . . Programming support library
1st sent.	83	Prefer that "top down structured programming" be concerned with methodology, display mechanisms and language constructs, but avoid including the coding heirarchy in the definition.

"Life cycle consciousness" is the key to effective application of these practices. The Government sponsors many efforts in which software is developed but not delivered (largely because the software is used as a tool to support analyses, which are the main product). Guidelines are needed for applying appropriate standards such as parts of your CP07877961001) to these situations in a way that ensures the Government will get what it wants (a reliable analysis) but will not pay for something it does not need or cannot use (a library of ad hoc analysis/simulation computer programs).

1. General Comments on the Specifications

   a. The RADC approach to produce a generalized specification that can be tailored to specific needs (by selection of alternative paragraphs) is especially useful.

   b. The suggested changes to Government specifications (Appendix 10) show recognition of the quandary that the contractor is in when he is told to use TDSP (etc.) and then required to deliver documents in a format ill-suited to TDSP. Some positive steps have been taken.

      However, your revision of DoD 7935.1-S 2.1.2 barely scratches the surface of a larger problem, which is also present in MIL-STD-1521A, AFR 300-12, etc. The SDR-PDR-CDR sequence needs to be treated more carefully so that, at least in certain circumstances, a true top-down implementation can be accommodated along with top-down design. The current specifications seem to leave no options for a level-by-level staged design/integration.

COMMENTS

The modern programming practices documented in Specification No. CP 0787796100D are quite complete and represent the most comprehensive collection of such procedures known to our review team. In our support of a number of Federal clients, including several DOD customers, we have utilized most of these procedures, though not necessarily in the exact form documented in the specifications. Almost without exception we agree with RADC's statement that these procedures result in "more reliable software which is less expensive over the entire life cycle." An example that we cite to our prospective clients in the benefits accrued from the use of the Logical Construction of Programs (LCP) methodology is as follows:

1) System containing approximately 40,000 lines of code taken from initial requirements analysis to acceptance testing in 7 months.

2) Reporting system containing approximately 130 programs taken from program specifications to acceptance testing at average rate of 20.5 hours per program.

3) Set of programs taken from program specifications to correct unit test at average rate of 24 lines of code per hour, with an average of 2 1/2 compilations and 2 test runs per programs.

The one procedure identified by our review team as having the smallest benefit was the Hierarchical-Input-Process-Output charting technique. The review team was unanimous in the view that, though the charts provide a higher level of program detail than conventional charting methods, they are not as useful as either Chapin or Warnier charts. Further, revisions to HIPO charts are difficult, usually requiring rewriting of all affected charts and, therefore, are expensive to maintain and update.

The open-ended approach adopted in compiling the procedures seem to insure that, with care, the specification can be maintained with up-to-date techniques and methodologies. Our review team felt that the User Guide was an excellent vehicle for directing a potential user to that part of the specification needed for a given element of the system development life cycle.

COMMENTS TO SPECIFICATION CP 0787796100D

No.	Paragraph	Comment
1	3.5c	Definition of structured segment is vague. It reads as though it is a complete routine coded exclusively with structured constructs. If so, it should be so stated.
2	3.5d	Structured constructs should be defined. Note: I see this was done in 5.3.2. The reader should be referred here.
3	3.5e	From your definition I see no difference between structured programming and modern programming practices (MPP). Personally, I define structured programming synonomous with structured coding.
4	3.5h	The concept of top down design and top down implementation (development) are different and should be distinguished. See Attachment A.
5	5.6	I hope it is the intention of this paragraph to allow a PDL listing in lieu of a detailed flowchart. If so, it is not clear from the paragraph.

COMMENTS

1. The answers to the questionnaire reflect what is being done in Control Data as a whole rather than only what has been implemented in the responding division. This does not imply, however, that any item checked is standard practice, uniformly implemented throughout Control Data but only that is is practiced in at least one organization.

2. We agree with the principle of the specification but not with all of its specific provisions. We believe that the specification is meant to illustrate the concepts discussed and is not meant to be an industry standard. Any specific provisions of the specification that are intended to be industry standards should first be approved by the National Bureau of Standards. As an example, research by several universities indicates that PASCAL is better suited for operating system development than the higher order languages mentioned in the specification. Further study is needed to relate benefit derived and cost of utilizing the specific software development techniques being suggested.

3. Refer to section 5.4.4 of the specification. It is felt that six man programmer teams are not appropriate for all development efforts. Software development projects producing about 2000 lines of code or less, and many enhancement efforts are examples of situations in which six man teams may be questionable. Chief programmer teams require a certain mix of personality types that may be impossible to assemble for all development projects. Also, it may not be possible to find enough "chief programmers" to meet requirements.

4. Refer to Appendix 70. We feel that the assignment of development duties on the basis of "designers" and "coders" is, in general, a mistake. Such a philosophy provides professional duties only for the designers and hinders the coders from becoming competent designers due to lack of design experience. Designers responsible for a program's general design, i.e. design to the "black box" level in which the interfaces and functional description of each module are specified, and programmers responsible for the detailed design, code and test of the modules from the general design specifications provide a better basis for assignment of development duties. In this way, each member of the development team has professional duties and challenge according to experience, ability and work preference.

5. Refer to Appendix 70. We feel that, in general, a four man inspection team is too large. In many cases a two man team consisting of a senior technical person as reviewer and the responsible programmer is more effective.

SUMMARY OF RADC REVIEW

- ENDORSE OVERALL INTENT OF RADC SPECIFICATION
  - •• HIGHER ORDER LANGUAGES
  - •• SOFTWARE DEVELOPMENT TOOLS
  - •• TOP DOWN DESIGN
  - •• STRUCTURED PROGRAMMING
  - •• PROGRAM SUPPORT LIBRARY
- DESIRE INCREASED CONTRACTOR FLEXIBILITY IN SOME AREAS
  - •• PROGRAMMER TEAM ORGANIZATION
  - •• PROGRAM DESIGN LANGUAGE
  - •• SOFTWARE SECURITY
  - •• MIL-STD-483 DOCUMENTATION
  - •• UNLIMITED RIGHTS FOR NON-CONTRACTOR DEVELOPED SOFTWARE
  - •• MICROPROCESSOR SYSTEMS
- CONSIDER THE IMPACTS ON SOFTWARE DEVELOPMENT AND MAINTENANCE COSTS OF USING THE SPECIFICATION
  - •• CONTRACTOR MUST PERFORM TASKS NOT TRADITIONALLY INCLUDED IN A SOFTWARE DEVELOPMENT
  - •• CUSTOMER'S MAINTENANCE PROCEDURES MAY NOT ADVANTAGEOUSLY USE CONTRACTOR'S DELIVERED SOFTWARE
  - •• SPECIFICATION OF EXCESSIVELY RIGOROUS REQUIREMENTS "JUST TO BE SAFE"
  - •• USE OF EXISTING OR GOVERNMENT FURNISHED SOFTWARE MAY RESULT IN POORLY DESIGNED SYSTEMS

3. GENERAL COMMENTS

This section contains comments which are pertinent to the entire specification. These comments are grouped into three categories: topics of agreement, topics of disagreement, and topics of concern. The topics of agreement and disagreement categories include remarks that point out the overall strengths and deficiencies of the specification. The topics of concern category includes subject matter that is not addressed in the specification, possible implications of the specification requirements, or lack of emphasis on important items. The specific paragraphs associated with these general comments are identified in section 4 of this report.

3.1 Topics of Agreement

The Rome Air Development Center Computer Software Development Specification assembles within one cover many aspects of modern software engineering that are useful in computer software development. The techniques of top-down design, structured programming, higher order languages, program support libraries, and software tools will aid project development, configuration control management, and software maintainability. The Central Data Systems Center has used all of these techniques in various deliverable software development projects, and the experience indicates that programmer productivity is increased and that the software system is easier to maintain. This specification should promote more widespread use and acceptance of these modern software engineering procedures. The expected result is more reliable and more maintainable software systems. In summary, the overall intent of this specification is endorsed.

An item of particular interest is the program support library. Its basic concepts as described in the specification are well founded. A very important task in software development is the organization of all the data relevant to a project. This data can be personnel assigned to various developmental tasks, all types of documentation (human-readable and machine-readable), catalogued procedures for automatically updating, retrieving, and storing modules, configuration of support software, and the current status of the project development effort. The program support library will provide a convenient and uniform vehicle to satisfy these requirements.

The format of the specification's paragraphs is very well organized. Within each section the paragraphs proceed from the least rigorous requirements to the most rigorous requirements. This hierarchical structure clearly shows the difference among the available options. The examples in the users guide illustrate well how to select and apply the appropriate paragraphs of the specification to typical software procurement situations.

3.2 Topics of Disagreement

The most encompassing criticism of the Rome Air Development Center Computer Software Development Specification is that it is often too restrictive. Therefore, it may tend to inhibit use of state of the art technologies; it may require procedures which are not cost effective, or it may specify personnel organizations that are incompatible with the scope of the project. In summary, the detailed requirements of the specification may prevent viable options from being exercised during a software development project.

A brief discussion of the specific areas where it is felt that the specification is overly restrictive follows.

One of the contributing factors to the very large increase in the number of software development projects is the advent of minicomputers and microcomputers. These machines often cannot be purchased from a vendor with one of the approved higher order languages. In addition, the technologies of the minicomputer and microcomputer are very dynamic. It is difficult to predict what advancements may be forthcoming, and if they will be adaptable to the stated requirements of maintainability, reliability, and documentation.

It is felt that the program design language is immature at this time. The program design language is by its nature oriented toward a single procedure. It does not adequately show the critical time dependence and interfaces among procedures. Other advanced software engineering techniques such as directed flowgraphs and structured analysis and design fulfill this need. For a single procedure the program design language can only display the logical structure by the statement indentations. However, a structured intermediate level flowchart provides a visual two-dimensional representation of the procedure's logical structure at a glance. The program design language does not

adequately replace the existing techniques of directed flowgraphs, structured analysis and design, and intermediate level flowcharts. It is felt that to mandate its use would result in a poorer technical description and increased costs.

The specification of the software security procedures did not mention the possible use of the National Bureau of Standards forthcoming encryption algorithm. The option to use the encryption algorithm whenever it becomes available should be included in the specification.

The use of MIL-STD-483 as the minimum documentation standard for all software development is a stringent requirement. The MIL-STD-483 documentation is very complete, and, where appropriate (e.g., operational software), constitutes excellent documentation. However, associated with a software development project are several tasks where the MIL-STD-483 level of documentation is not needed. These tasks include support software modules, vendor or government furnished software, hardware maintenance and diagnostic modules, software tools, and test driver modules for the operational software. In these instances a users manual or contractor format documentation is adequate. The experience of the Central Data Systems Center is that MIL-STD-483 documentation is a manpower intensive activity, and hence, it is very costly. A blanket requirement to deliver MIL-STD-483 documentation for all software modules will surely increase developmental costs and lengthen schedules. In addition, MIL-STD-483 requirements are normally implemented by a Data Item Descriptions (DID) which are subject to change from one contract to another contract as the DID's are updated to new requirements. Therefore, MIL-STD-483 documentation for one contract may not satisfy the MIL-STD-483 requirements for another contract due to the updated DID's.

Software tools, the program support library, and preprocessors and precompilers developed under a government software procurement contract are required to be delivered with their related documentation to the government with unlimited rights. Typically, in these areas a contractor will modify or enhance an existing software system to achieve the particular requirements of the contract. This technique enables the contractor to utilize existing software and decrease developmental costs. However, the baseline system may be leased or proprietary; consequently, delivery of the entire system with unlimited rights may not be possible. It is suggested that the unlimited data rights clause be relaxed for non-contractor developed software to

permit using previously developed software systems. If the contractor is not permitted to use leased or proprietary software, lengthened schedules and increased costs are likely to occur.

The specification explicitly states that a programmer team shall consist of six programmers, one of which is designated a leader. If more than one team is needed, a chief programmer and a librarian shall be designated. This personnel organization requirement is much too restrictive. The specified programmer team organization may not be the optimal organization for the whole spectrum of software development environments that can exist. It is urged that this requirement be relaxed.

The preceding paragraphs cited those areas of the specification that severely limit the contractor's options to develop a software system. The results of these restrictions may be a less technologically advanced system and increased developmental costs.

In order to alleviate this problem, it is suggested that the Rome Air Development Center document be officially classified as a military standard rather than a military specification. A military standard establishes a set of engineering, technical, and procedural guidelines which shall be followed by the contractor. The use of the military standard could still be mandated; however, flexibility is permitted within the constraints of these guidelines. This change would enable the contractor to accomplish the intent of the specification while retaining the use of modern software engineering techniques without undue restrictions. This alternative would be subject to contract approvals.

3.3 Topics of Concern

The specified software development procedures require tasks that traditionally were not bid in a contract. These tasks include MIL-STD-483 documentation, the program support library, and the collection of error data. These additional tasks will increase the costs of the developmental phase of the total software effort. Ever though the developmental phase is often funding limited, it must be realized that the extra funds allocated to the developmental phase will be recovered during the software maintenance phase. Contracting officers, program management, and negotiators should be made aware of this fact.

The task of the software development contractor is to deliver the completed operational software, support software, and documentation to the customer. A large portion of the life-cycle costs of a software system is its maintenance. Although the modern techniques of software engineering described in the Computer Software Development Specification can reduce the life-cycle costs, the customer's personnel organization and software maintenance techniques are the most important factors in life-cycle costs. The development contractor does not have control over the customer's internal procedures. If the customer does not adapt his maintenance procedures to advantagecusly use the contractor's delivered software, the expected life-cycle cost reductions may not be realized. This fact is particularly important during the transitional period from the old maintenance procedures to the new maintenance procedures.

This specification is intended to be used by persons who may be "...well versed in the technical aspects of an effort but have little or no experience in buying software". It is felt that a potential problem exists if such people use this specification. If the contract writer has no experience in buying software, he may require the most rigorous level options of the specification "just to be safe". If the scope of the software development project does not require the most rigorous levels of the specification, the inexperienced contract writer would unnecessarily increase the project's costs.

The importance of using structured design principles in software development is not emphasized ercugh. Structured programming, by itself, will not produce an easily maintainable software system. For example, the classic New York Times project was one of the first software developments to use structured programming. The initial statistics indicated increased programmer productivity and completion on schedule. However, when changes were attempted to certain functions of the system, many unrelated modules had to be changed. Even though each module was well structured, maintenance has been a problem because the system was not properly designed. In particular, this specification encourages the use of existing contractor or government furnished equipment software in a new software procurement. Although this procedure may be expedient and reduce costs in the developmental phase, what are the life-cycle costs consequences of a poorly designed system? The

interfaces may not be optimally designed, certain existing modules may require unusually high maintenance, and the documentation may be inadequate or inaccessible. It is strongly urged that these facts be considered during the initial design decisions of whether or not to use existing software modules.

4. SPECIFIC COMMENTS

This section identifies those paragraphs of the specification that need modification and support the general comments of section 3. Whenever changes are made to the specification, the appropriate paragraphs of the users guide and appendicies must be modified to ensure consistency with the specification.

- Page 8, paragraph 3.5, item g - section 3 of a structured source code listing contains the executing sequence among structured segments where it is straightforward. This section should be deleted. The material of this section should be in the documentation of the software system. It is not part of the source code listings.

- Page 9, paragraph 5. - append the following paragraph:
  In this specification sub-sections withir a section are arranged so that each successive sub-section within a section increases the requirements. Moreover, the successive sub-sections include the requirements of all the sub-sections preceding it. Therefore, whenever the appropriate degree of rigor has been determined and the sub-section selected, the preceding sub-sections do not need tc be explicitly included.

- Page 10 - add the following paragraph:
  5.1.4 Assembly Languages. Programs which must be written in assembly language shall use the techniques of structured programming to the extent possible.

- Page 11, paragraph 5.2.1, last sentence - the clause "...shall be delivered with unlimited data rights" should be relaxed if necessary.

- Page 11, paragraph 5.2.1.1, item c - the code auditor should not limit the size of a segment to an arbitrary number of lines. The lergth of a segment should be determined by its function. The phrase "limiting segment size" should be deleted.

- Pages 11 and 12, paragraph 5.2.2 - the progran design language is immature at this time. It does not adequately replace the existing techniques of structured analysis and design and directed flowgraphs. This paragraph should be deleted.

- Page 15, paragraph 5.3.2.2, last sentence - the clause "...shall be delivered with unlimited data rights" should be relaxed if necessary.

- Page 16, paragraph 5.3.3.1, item a - the requirement that all segments exclusive of annotations be no longer than one page ir length should be deleted. The length of a module should be determined by its functicn. To arbitrarily specify modules should be no longer than one page long creates unnecessary interfaces and a fragmented system.

- Page 18, paragraph 5.4.2, last sentence - the clause "...shall be delivered to the government with unlimited data rights" shculd be relaxed if necessary.

- Page 19, paragraph 5.4.4.1 - the requirement that a programming team shall consist of six programmers is much too restrictive. The size, composition, and organization of the programming teams must be flexible to suit the scope of the software project and the contractor personnel resources.

- Page 20, paragraph 5.4.6, item a - the requirement that microprogramming be subject to all the documentation and maintainability standards of operational software is excessive and shculd be relaxed. The function performed by the microcode is different from operational software, the customer typically does not need to change the microcode, and the requirements of complete documentation and update capabilities would be very costly.

- Page 21, paragraph 5.4.6, item b - computer programs intended for execution on microprocessor systems typically are not written in an approved higher order language. Consequently, all the requirements for a higher order language program on a large scale processor may not be applicable for the microprocessor program. The stated requirement should be relaxed.

- Page 21, paragraph 5.5 - security requirements should be changed to encompass possible use of the National Bureau of Standards encryption algorithm.

- Page 21, paragraph 5.6 - the requirement of MIL-STD-483 as the minimum documentation standard is much too restrictive. In many instances (support software, diagnostics, tools, and test drivers) MIL-STD-483 is not required or even desirable. Contractor format documentation should be included as an acceptable format in these instances.

RADC COMPUTER SOFTWARE DEVELOPMENT SPECIFICATION NO. CP0787796100D

The subject document has been reviewed in varying degrees of detail by 10 people in 5 different geographical plant locations. This response is a compilation of the evaluations from those reviews, with the more complete responses receiving heavier weights. All responders regarded the specification concept and approach favorably, qualifying that position with general suggestions as well as specific line-number-referenced comments.

General Comments

There appears to be general agreement that modern programming practices such as walk throughs, code reading, and structured programming techniques, all contribute in various significant degrees to improving software reliability and portability, as well as contributing strongly to cost containment. For example, at two plant locations, the techniques addressed in this specification have been employed for 3 years or more on a number of medium-to-large-scale software projects. At these locations these techniques have already become common practice. The use of standard practice and conventions helps reduce confusion during project definition stages and by improving communication, leads to better understanding by individuals of their responsibilities. Modern practices lead to less complex software, reduce debugging activity, and facilitate testing and documentation. Top-down design eliminates the need for writing throw away software test drivers.

Design walk throughs have proven valuable in determining, before coding begins, how well programmers understand the requirements, as well as finding problems in proposed solution logic. For less experienced programmers the walk throughs have also brought to light their unfamiliarity with particular algorithms effective for the problems at hand. Programmer teams have been very successful when the lead programmer/analyst understands the requirements thoroughly and is conscientious. However early performance must be monitored closely, since a lead programmer who is not performing his share of the work or does not understand completely the task to be implemented can immobilize a team and create a morale problem.

There is universal agreement that on medium and large scale jobs the implementation of a programs support library, under the control of a configuration control analyst or group is essential. This group may serve as librarian for several or all of the programmer teams. The library function is seen by some as including inspections for conformance to coding and documentation standards. However the function which is universally regarded as essential is in the area of configuration control. This control insures the workable versions of all modules are available when testing a new or modified module of the system. The current configuration of the operational system is known at all times, and the status of program progress and documentation may be tracked by means of file transactions.

In reporting the effect of modern programming techniques such as are discussed in this specification, one reviewer provided the following data: a recently completed project was made up of 383 routines containing a total of

29,703 lines of code plus 50,999 lines of comments. The productivity yield in this project was 13.6 code lines per man day (exclusive of comments), or 37.5 lines total per man day (inclusive of comments).

The specification does not address the problem of requirements documentation, verification, and traceability to design. The specification should call for requirements to use a common notation and to be measurable (e.g. reliability should be a defined and measurable concept per current metric concepts, if reliability is to be spoken of at all).

Verification and validation techniques should likewise be clarified. There is in the literature a good deal of discussion about whether or not statistical sampling is a good procedure for verification and validation as compared to attempts at exhaustive testing or the haphazard testing techniques of the past. In any case, the specification should define different levels of validation and verification to be imposed and must specify corresponding specific measurements to be used.

Specific Comments:

It would be helpful to the contractor-user of this standard if a Table of Contents were included, and if the User's Guide would also be published in conjunction with the specification, possibly preceding it, so that the general approach is understood before the specification is read. In our copy of this Guide, pages 69-70 were missing.

The wording in paragraph 1.1 and throughout Section 5.2 was seen by some as too tight, allowing the possibility of strict enforcement in spite of guidelines; stress on encouraging flexible alternate functional equivalence. The problem here from the contractor's point of view is that while the policy promulgated in the guidelines is quite appropriate and suitable, the guidelines are not a contractual document, and therefore the contractor has no assurance that the policy found therein will actually define the working requirement. Within the specification proper, the phrase "the following functions as a minimum" might be better replaced with "the following or equivalent functions."

The first three figures referenced in pages 1 and 14 of the specification and in the User's Guide could not be found.

Page 1, Section 1.2. The "software acquisition management guide books" referenced at this point are difficult to find in the Users Guide, unless it is noted that they are ESD technical reports.

The term "software segment" has been widely used in the specification to mean a sequence of program instructions usually known to the trade as a subroutine or as a sub-program. It seems that if the term "software segment" is required to be used, that it be defined in Section 3 (definitions). There are a number of points at which the term "software segment" is used where unless the term subroutine is understood, the meaning is either unclear of inconsistent.

On page 8, Section 3.5.1: the phrase "skipping levels is not permitted" does not carry any meaning for some of our reviewers, and represents a desirable philosophy for those who understand it. In any event, this notion, though it is generally desirable in terms of a top-down-design flow chart, is inconsistent with the proper reality which is reflected in the text of the specification on page 13, paragraph 5.3.1. This paragraph contains the statement "it should be noted that some segments may appear on more than one level in order to retain logical completeness at each level." This statement is of course correct, and this phenomenon is widely noticed in the form of service subroutines, such as input/output or other elementary functions which are appropriately called by functions at several levels.

Page 11, Paragraph 5.2.1.1C, Design Standards: Limiting segment size and placing restrictions on use of instructions rarely optimizes code relative to execution time as is stated here. It is more consistent with optimizing readability and maintainability of the resulting code.

On page 13, Section 5.3.1.1: the implied definition of "top down hierarchal design" given as "that each level of detail of the program is "logically complete" indicates. Our definitions state that each level of detail of a program shall represent a functionally allocated identity defined by the clearly described and meaningful elemental function that it performs after implementation.

Page 16, paragraph 5.3.3.1A: Limiting code to a standard printout page or less is desirable, but strict enforcement of this rule will often break up code for which readability should be left in a single routine. Using a maximum of 3 or 4 pages, with an average of 2 pages, seems more reasonable since data definitions and comments for routine will often take a major portion of a printout page.

Page 16, paragraph 5.3.3.1.2: Care should be taken in detailed coding conventions such as requiring that data declarations shall be grouped and arranged in a columnar string of one simple definition per line. The intention of this is to avoid packing data so densely as to make understanding difficult. A reasonable application of from 3 to 5 values in a line may very well increase the clarity over the long, cumbersome and excessively thin string of data statements which this specification would require in a significant number of cases. The point to be made is that in coding conventions moderation is the key word: that all extremes should be avoided in the pursuit of clarity.

Page 17, paragraph 5.3.3.2-.4: There are circumstances in which each of the three forms of code examination (reading, review, inspection) will work well and other circumstances in which each will work poorly. Most of our reviewers see the question as determining which one form will work best for a particular case, and agree that they should not be stacked up into a multiple requirement except in specific well-considered cases.

Page 19, paragraph 5.4.4.1: A programmer team is defined as exactly 6 programmers. In practice we are sure that some tasks will be more efficiently accomplished with a greater or fewer number of people. Because large teams do not work well, programmer teams should be defined as having a maximum of 6 people, with smaller numbers permitted where efficiency will be so served.

Comments on Appendices

General: Many of the appendices contain helpful material, but we feel strongly that appendices 30 and 50 are not appropriate, and leave the casual observer with entirely the wrong impression.

We presume that appendix 50 is intended to provide a restrictive definition of the allowable syntax and conventions for program structures. This should be done tersely (3-4 pages maximum), and the pre-processor design and coding techniques left to the technical literature. Appendix 20 performs the definition function (but not tersely enough) for PSL, and Appendix 30 should be omitted altogether.

Appendix 10:

Page A-10-3: paragraph 10.2 is aligned as an item under 10.1 so that there was confusion about the referenced document within which the specified pages to be updated were contained.

Page A-10-4, section 10.2: it is stated that "if structured programming technology is used, the design phase will concentrate on the development of requirements." The word "requirements" in this context is misleading. The word "requirement" has been used frequently in conjunction with the words such as system, design, etc., to mean overall system design requirements.

The design phase shall actually concentrate on the development of program design which will become coding requirements, at a later time for the coding phase.

Appendix 20: PSL Functions

How can anyone write such a document with only passing mention of version numbers and no software logic to handle version numbers? Configuration control MUST be a major function of PSL. There is also no mention of base-lining, a major fundamental of software development.

Clearly, computer vendors do not give us enough support for software development (several prominent computers do not even provide a source update processor). Attempts at an operating-system independent PSL are not practical at this time. There is too much development required.

There have been enough problems with software development; such a complex PSL as is specified here may very well burden the implementers with another large software package, more control cards, and potentially another set of unreadable documentation.

Appendix 40: A Program Design Language description.

This is a good paper on PDL. It gives a clear description of how PDL fits into the design and documentation process and the relative advantages of PDL over narratives/flow-chart program design methods. The descriptive material and the tutorial examples should be sufficient to introduce the uninitiated to PDL.

The major feature lacking in this PDL is a method for defining data connections: a way to show arguments being passed to lower level routines or to arguments used for an in line function. While defining the functions being performed in the PDL, it is also necessary to show the data being operated on for each function and what the data connections are between functions, in order to have a complete and verifiable design.

This specification calls for a two-column indentation convention. Several users having extensive experience with the clarity provided by structural indentation report that using four or five column indentation will markedly improve the readability of PDL.

Appendix 70: Design, Code, and Test Inspection

This material is seen as helpful, particularly in that the checklists are useful in assisting participants to determine what to look for in each type of review. We question the value of the forms used to track numbers of errors found during reviews. Formally collecting this type of data frequently fosters a defensive attitude towards alternative methods which otherwise frequently come out of design reviews.

Another type of review which could be considered, if not addressed elsewhere, is the requirements review. It is generally very helpful for the prospective designer to document his understanding of the functional requirement of the program and to have this concept reviewed prior to starting the design.

COMMENTS

1. The Spec does not differ much from its CP0787796100C version. I am enclosing a copy of our comments to USAF/ESD on that version. Basically, those comments conclude that the use of the Spec as it stands would not be in the best interests of either the Air Force or the software industry--but that an appropriate rework would be.

2. Our comments on sections 5.4.2.2 and 5.4.2.3 in version C now refer to the counterpart sections in Appendix 20 of version D.

3. The Users Guide is a good idea, and a good draft. It presents a few problems, however:

   a. In places, it is inconsistent with the Spec. For example, the Spec mandates the use of "programmer teams consisting of six programmers" (5.4.4.1). The Users Guide encourages the contractor to deviate from the Spec and use a team consisting of a "lead programmer, one to five backup programmers, and a secretary/librarian." Further, the use of five backup programmers would be inconsistent with the use of the term "backup programmer" in the referenced document, RADC-TR-74-300, Vol. X. (Users Guide, p. 32-33).

   b. In places, its advice is inconsistent with software project experience. An example again is the use of Chief Programmer Teams on large projects. See the comments on Section 5.4.4.1 of version C in the enclosed letter to USAF/ESD.

   c. It is incomplete. For example, page 1 of the Spec indicates that the Users Guide contains a complete list of Software Acquisition Management (SAM) Guidebooks. But it contains references to only two of the Mitre SAM Guidebooks, and none of the SAM Guidebooks produced by SDC, TRW, and Boeing.

We continue to feel that an appropriate rework of this spec would be a significant benefit to the Air Force and the software industry. I hope that your efforts will serve to expedite the achievement of this goal.

1. Potentially qualified vendors would generally be reluctant to respond to IFB's and RFP's which contained the detailed requirements defined herein.

2. Unless and until the parameters and design requirements specified herein are standardized, not only within the Federal Government, but also within the industry, viable commercial suppliers cannot economically and competitively respond. It is this reviewers opinion that specialty houses and those vendors without established in-house standards would be most apt to accept the constraints specified. Consequently, the major vendors in the Commercial sector would not be willing to modify their development and documentation standards except for major programs. The net result being that the government either pays an excessive premium or else is not serviced by large, established, competent vendors.

3. The emphasis on JOVIAL is extraneous to the coverage normally accorded the language. The use of JOVIAL in the Commerical/ Industrial sector is minimal.

4. Federal Government ADPE procurement procedures are, at best, confusing, burdensome, and expensive for the vendors. They frequently discourage vendor response, particularily for small quantity, small ticket items. The complex technique of referencing a myriad of regulations, publications, specifications, and documents, many of which are additive to, duplicative of, or excessive to the Commercial and/or industry standards for commercially available products, are an unreasonable and unnecessary burden on the vendors. Follow examples apply:

   - Armed Forces Procurement Regulation (ASPR)
   - Military Specifications (MIL SPECS)
   - Military Standards (MIL STDS)
   - General Services Administration Procurement (GSA)
   - Federal Information Processing Standards Publications (FIPS PUBS)

5. Vendors who desire to market ADPE and related services to the Federal Government must maintain a virtually dedicated entity separate and apart from the Commercial/Industrial entity. Frequently the requirements and constraints imposed by Federal Government solicitations work to the detriment of the best interests of the government because a majority of the vendors resources are not familiar with governmental procedures and cannot, therefore, make a meaningful contribution. In summary, procurement procedures for "Commercially available" ADPE are too complex, too unique and are wrought with excessive duplication and technical references.

All of these remarks are to be understood in the context that our experience is entirely in small-medium (1-40 man year) systems - not applications - programming projects.

1) High order languages are almost a necessity for any substantial programming project. Unfortunately, the high order languages mentioned in the specification - with the possible exception of Jovial - are suitable for only some applications programming, but not for systems programming. As an example, the data objects to be manipulated in most systems programming - tables, lists, etc. - are not Fortran or Cobol data types. Further, certain kinds of applications programs - e.g., programming Automatic Test Equipment - cannot be done in the three approved languages. I believe that the Specification should explicitly allow that a high order language be used on a project without requiring that the language be one of the approved ones.

2) Almost all of our work in the past ten years has been done using interactive operating systems with their usual features - text editors, debuggers, etc. The availability of interaction seems to be cost effective, but we have no exact figures for the actual cost savings. An unreliable interactive computing resource seems to have much more of a negative impact that an unreliable batch resource. Thus, a marginal interactive resource may well be less cost effective than a reliable batch resource.

3) The importance of careful top-down design cannot be over-emphasized.

4) Coding standards and conventions which focus on using only certain control structures obscure the issue. Properly structured code will naturally tend to use only good control structures, but code shouldn't be deformed to fit an unsuitable control structure. The literature is full of examples of "well-structured" code which cannot reasonably be rewritten in a form using some more-or-less arbitrary set of control structures. I believe that the Specification should be more flexible, at least in those cases where formal code reviews are done.

5) I am favorable to the intent of the Specification, but I am nervous about its practical application. I am afraid that it will be all too common for inexperienced project engineers to overload a project with too many requirements from the Specification. It is asking a great deal of an inexperienced person's judgement to decide at what phase in a project it becomes proper to require, e.g., quality assurance.

COMMENTS ON CONTENT OF SPECIFICATION CP 0787796100 D

1. Section 2.2. Non-Government Documents.

In addition to ANSI X 3.12 - Vocabulary for Information Processing - you may want to consider "Software Engineering Terminology" which is an authorized IEEE Standards project currently available in draft form from

Mr. Robert Poston
19 Onondago Trail
Medford Lakes NJ 08057

2. Section 5.3.2.1 Structured Coding

The requirement for a single exit is quite constraining (and may at times be harmful) for "rugged" programs which have frequent calls to exception or fault handlers. Permitting a separate exit to an exception handler can actually improve the readability of the code (and that is of course a primary objective of this paragraph of the specification) while at the same time reducingmemory requirements and increasing code execution.

An example of software design for exception handling by use of a special exit is shown in fig. 1 on the attached sheet while fig. 2 represents the same case with strict adherence to the single exit requirement. In many instances the complexity of multiple level flag setting and reading (which is required in order to vector the fault handler into the code segment that caused the fault) can be much worse than shown here.

Provisions for "rugged" software are becoming increasingly common. In the simplest case data can be examined for compliance with type and range declarations, or parity checks made on memory or bus access. But more sophisticated tests for data reasonableness (or consistency with prior values) will be utilized for critical real-time software. Permitting a well-defined exit when such a test is failed will simplify coding while at the same time promoting readability and maintainability of the software.

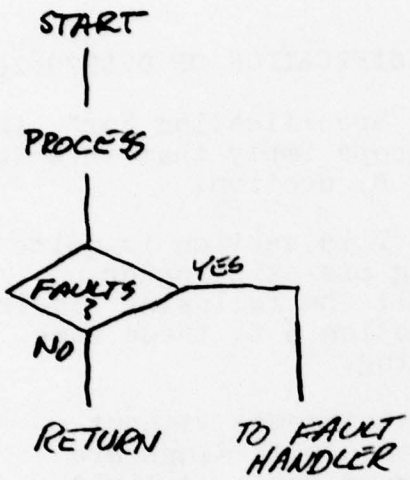


FIG 1
SEPARATE EXIT TO FAULT HANDLER

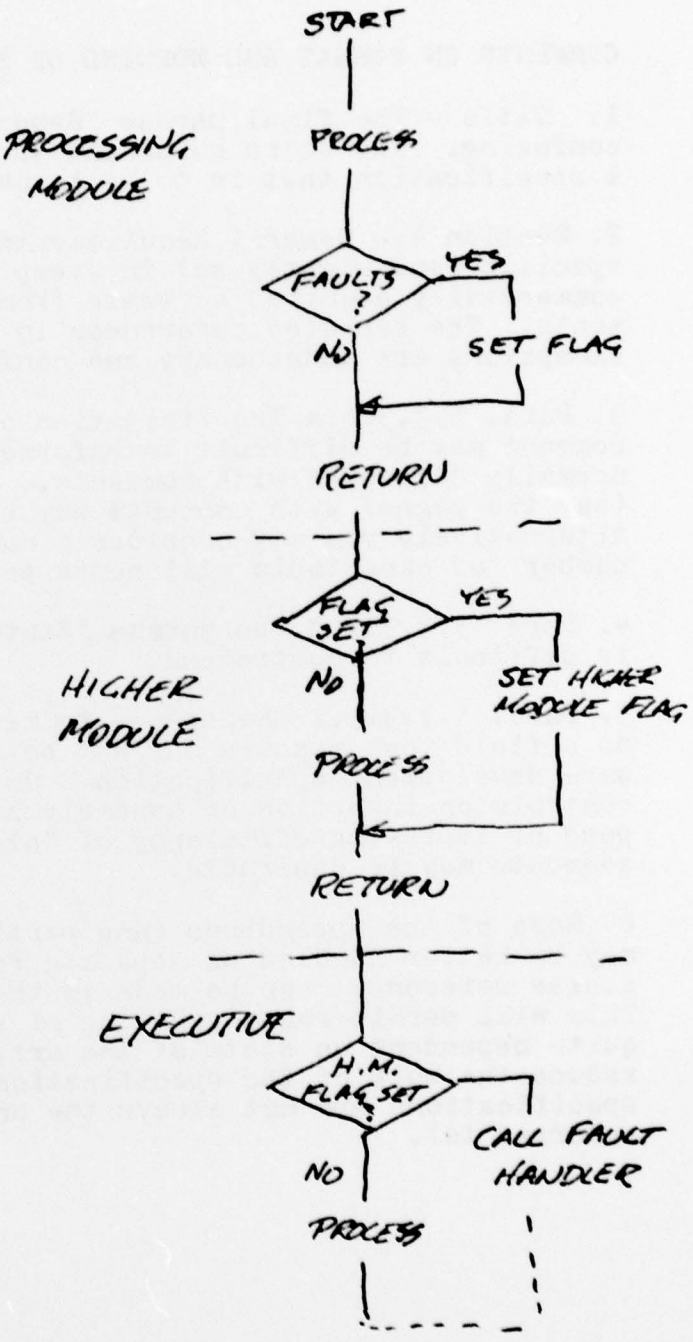


FIG 2
STRICT SINGLE EXIT

COMMENTS ON FORMAT AND WORDING OF SPECIFICATION CP 0787796100D

1. Title - The final phrase "General Specification For" is confusing. The users guide and the scope imply that this is a specification that is to be invoked by section.

2. Section 4 - General Requirements. This section is quite specific (and rightly so) in exempting pre-existing or commercially acquired software from all the following requirements. The repeated references in section 5 to these same exemptions are unnecessary and confusing.

3. Para. 5.3.3.1.a The limitation on statements without comment may be difficult to enforce because listings are normally delivered with comments. A more generous limit (say two pages) with comments may be easier to enforce. Alternatively you may consider a numerical limit on the number of executable statements per module.

4. Para. 5.3.3.1.d The phrase "Statements (i. e. Format)" is difficult to comprehend.

5. Para. 5.3.3.1.i Reference to machine language patches as a field test measure may not be appropriate in a software development specification. On the other hand, some controls on insertion of assembly sections for the purpose of improving efficiency of "high traffic" code segments may be desirable.

6. Some of the appendices (and particularly appendix 3) may be better handled as separate reports (to which of course reference can be made in the specification). This will permit ready updating of subjects that are quite dependent on state of the art. And it will also reduce the bulk of the specification document (bulky specifications are not always the ones most frequently referred to).

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.