Evaluation of Software Development Methodologies: A Final Report

Richard E. Nance, James D. Arthur, and Ashok V. Dandekar

TR 86-42

SRC-86-010

EVALUATION OF SOFTWARE DEVELOPMENT METHODOLOGIES

Final Report of the
Immediate Software Development Issues Project*

(August 1984 - December 1985)

Richard E. Nance
James D. Arthur
Ashok V. Dandekar

Systems Research Center
and
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061

* Work supported by the U.S. Navy through the Systems Research Center under Basic Ordering Agreement N60921-83-G-A165- 3003-1.

EXECUTIVE SUMMARY

The Immediate Software Development Issues project, initiated by SEA61Y in August 1984, sought to compare two current Navy software development methodologies, RNTDS and AEGIS Modular, and to assess the potential effect on each by the adoption of the DoD-standard language, Ada. The work was performed by the Systems Research Center (SRC) of Virginia Tech in conjunction with the Naval Surface Weapons Center and the Fleet Combat Direction Systems Support Activity, Dam Neck, Virginia. An evaluation procedure developed by SRC personnel was applied to software samples (programs and documentation) selected by NSWC personnel as representative of the development process.

The evaluation procedure is based on the recognition of linkages among software engineering objectives and principles and among principles and attributes of the software, i.e. programs and documentation. The identification of code and documentation properties and the definition of metrics for these properties enables an accumulative scaling of attributes, principles, and objectives.

The evaluation procedure is judged to be sound, and the results are judged to be indicative of the methodologies examined. The conclusions and recommendations of the study follow:

Conclusions:

1. AEGIS Modular supports the principle of structured programming and emphasizes the objective of software reliability.

2. RNTDS supports the principles of structured programming and good documentation and emphasizes the objectives of maintainability, adaptability, and reliability.

3. The principle of information hiding is not supported by either methodology.

4. Software produced by both methodologies fails to exhibit the attributes of traceability and well-defined interfaces.

5. RNTDS achieves software engineering objectives to a greater degree.

6. Implementation dependencies prohibit the derivation of benefits through insertion of Ada for selected functions.

Recommendations:

1. Common software support should be achieved as a consequence of the evolution to Ada.

2. Converged CDS objectives should be prioritized to drive the software development objectives.

3. The PGE Working Group should identify an underlying software development methodology.

4. Visibility and emphasis should be given to software engineering objectives in order to cope with long-term life cycle costs.

5. A research task to investigate the indicator concept should be chartered by the PGE Working Group or others concerned with software quality assurance.

Commentary, which expands and explains both the conclusions and the recommendations, is included in the body of the report.

ABSTRACT

A procedure for evaluating software development methodologies is developed based on the linkage among objectives, principles, and attributes. The procedure utilizes metrics defined to recognize surface properties of the software, leading to evidence of the presence or absence of particular attributes. Application of the procedure to samples from the RNTDS and AEGIS Modular Development programs reveals weaknesses and strengths that should be addressed in efforts to produce a common CDS. The extension of the work is proposed to address two major shortcomings of the evaluation procedure: (1) intensive demands on humans in the analysis of code and documentation and (2) a high degree of subjectivity in some property assessments.

PREFACE

Following the Executive Summary this report is organized and formatted to reflect the final briefing, given to RADM Lowell J. Holloway and the CDS Steering Group on 15 November 1985. Briefing transparencies appear as pages on the right, and the explanatory narrative accompanying each transparency, on the left. The report is concluded with a bibliography and acknowledgments. The interim report and a paper describing the early efforts to formulate and test the evaluation procedure are included as appendices. The examination of the effect of Ada on the current methodologies is reported in these earlier documents.

The sample data used to derive the accumulative scoring for attributes, principles, and objectives are too extensive to be included here. Questions concerning the data should be addressed to Dr. Harry E. Crisp, Naval Surface Weapons Center, Dahlgren, Virginia.

ACKNOWLEDGMENTS

The work leading to this report has included contributions from many persons. The authors extend their appreciation for the efforts of those listed below:

Dr. Harry E. Crisp, NSWC

Roy F. Bickerstaff, FCDSSA Dam Neck

Robert E. Braudes, CSC

Dr. Richard P. Cullen, NSWC

Clyde S. Evans, FCDSSA Dam Neck

George R. Gay, FCDSSA Dam Neck

Thomas J. Goodall, FCDSSA Dam Neck

Daniel T. Green, NSWC

Donald J. Lee, FCDSSA Dam Neck

James C. Martinette, FCDSSA Dam Neck

David E. McConnell, NSWC

Robert A. McClain, FCDSSA Dam Neck

James F. Reagan, NSWC

Terry Shull, FCDSSA Dam Neck

Judy B. Smith, FCDSSA Dam Neck

Charles H. Sperry, CSC

Cary D. Upshur, FCDSSA Dam Neck

OUTLINE

I. OBJECTIVE

II. APPROACH

III. DEVELOPMENT OF AN EVALUATION PROCEDURE

- A. Software Engineering and Software Development
- B. A Structured Evaluation Procedure
- C. Data Sources

IV. APPLICATION OF THE EVALUATION PROCEDURE

- A. Summary of Sample Data
- B. Illustration of the Elements, Metrics, and Properties

V. SUMMARY OF RESULTS

VI. CONCLUSIONS

VII. RECOMMENDATIONS

VIII. FUTURE PLANS

IX. BIBLIOGRAPHY

X. ACKNOWLEDGEMENTS

XI. APPENDICES

Initially, the task statement divided into three parts

C.1: Comparison of the AEGIS Modular and RNTDS methodologies in concert with an assessment of the cost and benefits of using multiple methodologies,

C.2: Investigation of Ada language issues and the difficulties and benefits in effecting the rapid transition of existing projects to Ada,

C.3: Assessing issues affecting the accommodation of Ada by either the AEGIS Modular or RNTDS software development methodologies.

With the growing appreciation of the difficulty of evaluating software development methodologies, the research team was directed to focus on the comparative evaluation of the AEGIS and RNTDS approaches. However, the Ada-related issues, emphasized in the interim report (see Appendix 1), were not ignored in the conclusions and recommendations provided in this final report.

OBJECTIVE

PRESENT RESULTS OF AN ASSESSMENT

OF THE AEGIS AND RNTDS SOFTWARE

DEVELOPMENT METHODOLOGIES

The evaluation of software development methodologies begins with the definition of "methodology." The literature in software engineering considers the software development methodology to:

(1) Include methods and techniques for accomplishing individual tasks (within the framework of global objectives),

(2) Prescribe an order in which tasks are done and decisions are made,

(3) Encompass the entire life cycle of software development.

A software development methodology defines the required utilities (tools) provided by the environment supporting that methodology, and further, defines the mutually supportive roles necessary to achieve the integration of utilities.

The contribution of Navy scientists and engineers is present throughout the report. Individual and group support is documented in the Acknowledgements.

APPROACH

1. DEVELOP AN EVALUATION PROCEDURE

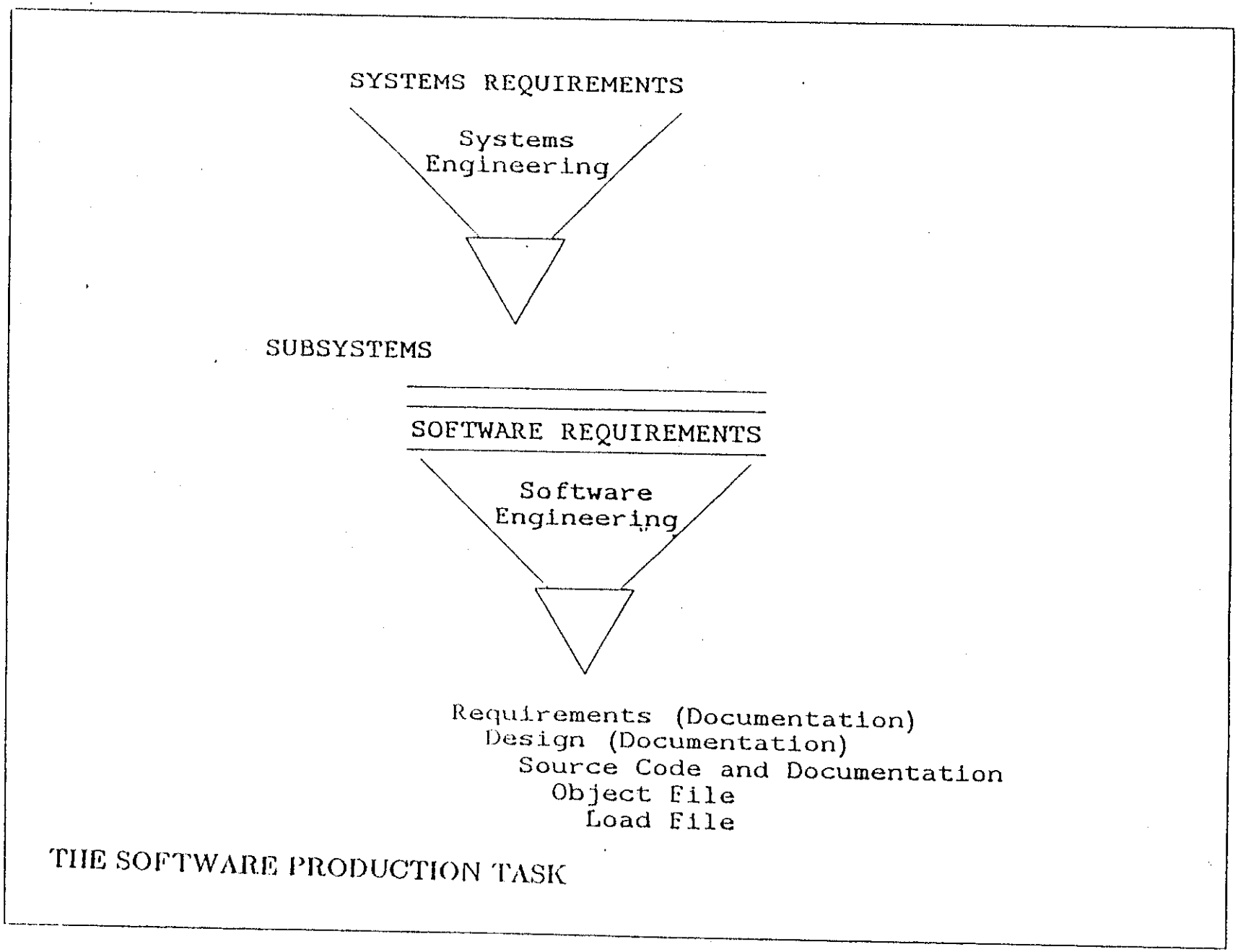
   - What is a methodology?
   - How can they be compared?

2. APPLY THE PROCEDURE

   - In consonance with PGE Working Group
   - Contributions from FCDSSA Dam Neck and NSWC

A major difficulty encountered in the development of software for combat systems support is in recognizing the distinction between systems and software requirements. The system requirements are developed as part of the systems engineering task, and system performance is the major goal of this engineering activity. Software performance, as it is required to attain performance of the encapsulating system, is not a software engineering objective. Software engineering must recognize the required performance as a constraint or requirement on the software engineering task.

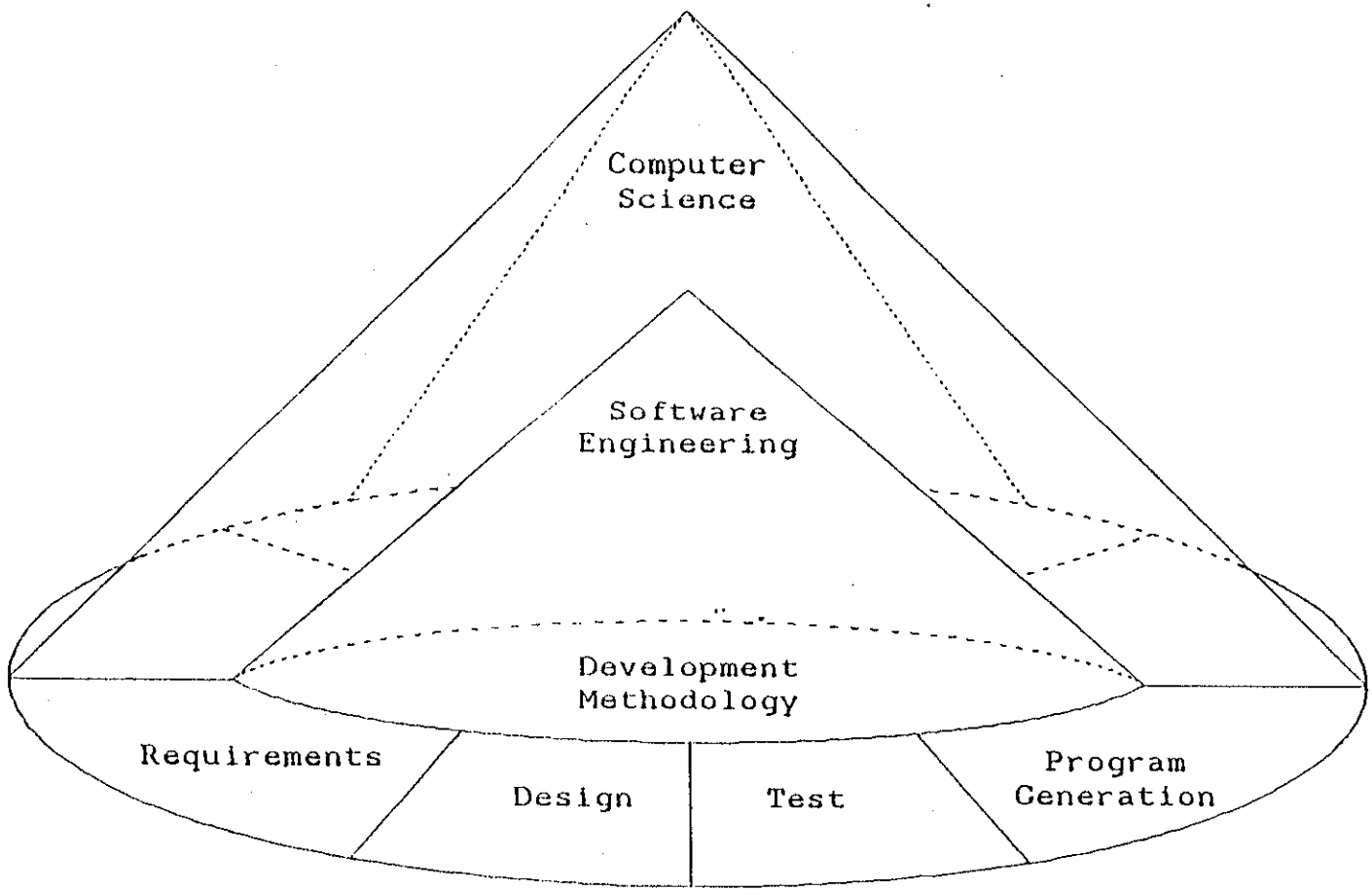
The software engineering activity must extend over the entire life cycle, beginning with the requirements and concluding with maintenance over an extended time period. Products of these activities include both documentation and code.



THE SOFTWARE PRODUCTION TASK

The recency of software engineering and computer science contributes to inconsistencies in terminology and uncertainties regarding relationships. Nevertheless, computing technology has already had a significant impact on terminology common to the technical work place. The pyramidal illustration is an attempt to depict software engineering as a subdiscipline of computer science. This important subdiscipline focuses on the identification of principles and techniques governing the software development process, in the attempt to evolve a science from the current art of software production. While a particular software development methodology might not embrace all the objectives of software engineering, nor give equal weight to those objectives included, the methodology should be clear and unequivocal in citing its objectives and the principles supported to enable achievement of those objectives. The tools and utilities embodying software engineering principles should also reflect desired principles from other areas of computer science, such as human machine interaction, concurrent programming, etc.

The importance of the methodology can not be overstressed, for in the methodology is found the basis for functional partitioning among tools and the basis for subsequent tool integration to form an environment.



Diciplines, Methodologies, and Tools

* Methodology is central
  + Defines needed tools and functionality
  + Basis for integration

Very little is found in the literature regarding comparative evaluation of software development methodologies. Forced to construct an evaluation procedure, the research team retreated to "first principles". The rationale used by the team: to identify software engineering objectives from a review of the open literature, to define principles required or necessary to achieve these objectives, and then to identify attributes of the software representing the product of a process utilizing the designated principles to achieve the identified objectives. The difficulty in recognizing code and documentation attributes in any software examination is all too apparent. Consequently, the need is to identify surface features and to define measures by which these features indicate the presence or absence of attributes.

PROCEDURE DEVELOPMENT

1. IDENTIFY OBJECTIVES

   - What qualities are desirable?

2. DEFINE PRINCIPLES

   - How are desirable characteristics obtained?

3. LINK PRINCIPLES TO OBJECTIVES

   - Which principles contribute to each objective?

4. DEFINE RESULTING ATTRIBUTES

   - Use of a principle induces what desirable characteristics?

5. DEFINE PROPERTIES ASSOCIATED WITH ATTRIBUTES

   - What surface features give evidence of attribute presence or absence?

   - How to measure surface features?

The seven objectives listed on the right represent the strong consensus to be found in the software engineering literature. The terminology may vary, for example "flexibility" as opposed to "adaptability". Nevertheless, one or more of these seven objectives can be found in any discussion of the software engineering process. As noted previously, a specific methodology may emphasize these objectives to different degrees, and that is certainly the case with AEGIS Modular and RNTDS. Any methodology should begin with a clear explication of the objectives to be realized in utilizing that methodology.

PRIMARY SOFTWARE ENGINEERING

OBJECTIVES

1. **ADAPTABILITY** - The ease with which software can accomodate to changing requirements

2. **CORRECTNESS** - Strict adherence to specifications

3. **MAINTAINABILITY** - The ease with which corrections can be made to respond to recognized inadequacies

4. **PORTABILITY** - The ease in transferring software to another host environment

5. **RELIABILITY** - The error-free behavior of software over time

6. **REUSABILITY** - The use of developed software in other applications

7. **TESTABILITY** - The ability to evaluate conformance with specifications

The seven principles of software engineering also represent a strong consensus. Two principles, top-down design and modularity, are representative of a much earlier period (the late 1960s). Subsequently, both have been subsumed by functional and hierarchical decomposition. Note that the abbreviated definitions for principles and objectives are described more fully in the interim report provided as an appendix.

PRINCIPLES FOR ACHIEVING OBJECTIVES

1. **DOCUMENTATION** - Management of supporting documents (system specifications, user manual, etc) throughout the life cycle

2. **FUNCTIONAL DECOMPOSITION** - Components are partitioned along functional boundaries

   TOP-DOWN DESIGN - Iteratively refining the definition of software components

   MODULARITY - The partitioning of software into components

3. **HIERARCHICAL DECOMPOSITION** - Components defined in a top-down manner

4. **INFORMATION HIDING** - Insulating the internal details of component behavior

5. **LIFE CYCLE VERIFICATION** - Verification of requirements throughout the design, development, and maintenance phases of the life cycle

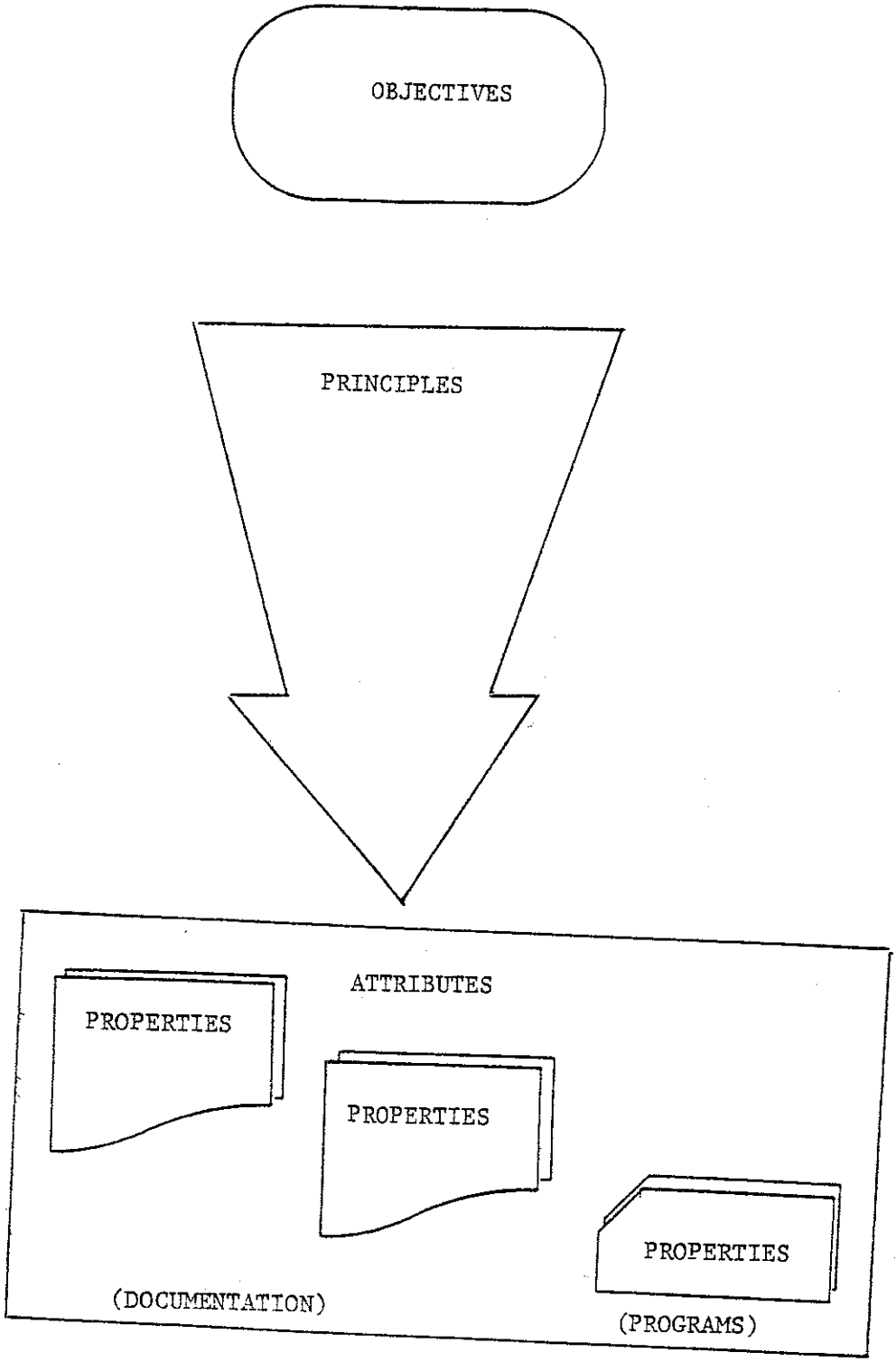
6. **STEPWISE REFINEMENT** - Utilizing a convergent design

7. **STRUCTURED PROGRAMMING** - Using a restricted set of control constructs

Having established the principles to be supported, encouraged, or required in the software development process, we then sought to identify the attributes resulting from their use. The nine attributes represent the desirable characteristics of the products of the software development process. Some attributes tend to be intangible and abstract; however, surface features manifesting these attributes are observable.

The evaluation procedure depends on the ability to link principles to objectives and attributes. That is, a methodology which emphasizes certain objectives should provide through its tools the utilities and methods embodying those principles so that the product (source code and documentation) reflects the desirable attributes. The attempt to define linkages among principles and attributes constitutes a major contribution to the software engineering literature.

A further contribution is the identification of properties, the surface characteristics of software, that provide evidence of the presence or absence of corresponding attributes. The formulation of metrics reflecting the degree to which a property is evident continues as an area of investigation.



THE EVALUATION PROCEDURE

Selecting a single objective as an example, we show the four principles contributing to the realization of that objective: reusable software. Expanding one principle, information hiding, we note the five attributes that should be evident in software developed using a process governed by the principle of information hiding. Narrowing our attention to one of these attributes, well defined interfaces, we note the five properties contributing evidence to the claim that a piece of software exhibits well defined interfaces.

The accumulative nature of the evaluation procedure should be clear from this illustration.

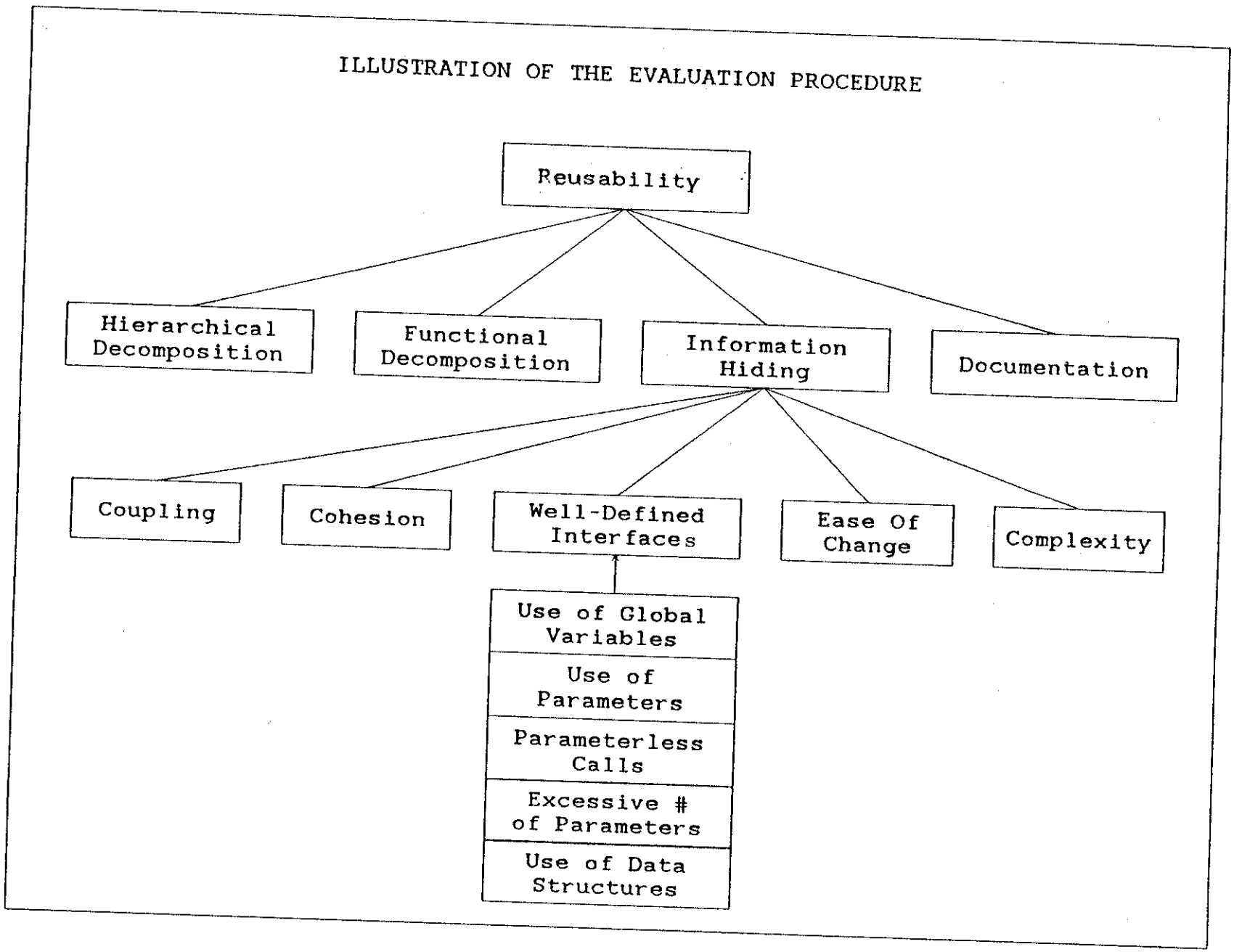


ILLUSTRATION OF THE EVALUATION PROCEDURE
Reusability
Hierarchical Decomposition
Functional Decomposition
Information Hiding
Documentation
Coupling
Cohesion
Well-Defined Interfaces
Ease Of Change
Complexity
Use of Global Variables
Use of Parameters
Parameterless Calls
Excessive # of Parameters
Use of Data Structures

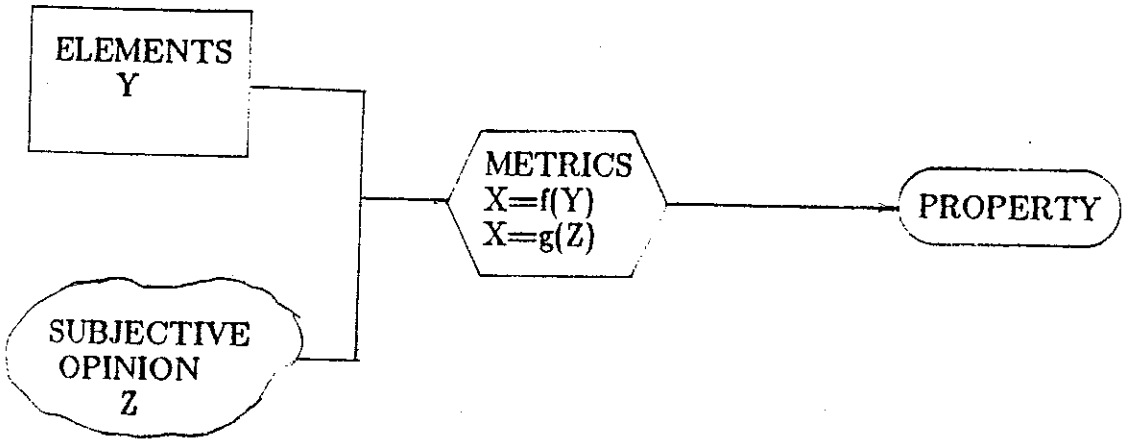
The product of the software development process is a collection of programs and documents. Thus, while one software component might possess a given property, another might not.

In the attempt to recognize the sampling nature of the evaluation procedure and to delineate more carefully between subjective and objective characteristics, a set of metrics are defined for each property. Contributing to each metric computation are objective observations achieved through software examination (elements), or subjective opinion based on the examiner's reasoned conclusions. A code example serves to illustrate the former and a documentation example the latter.

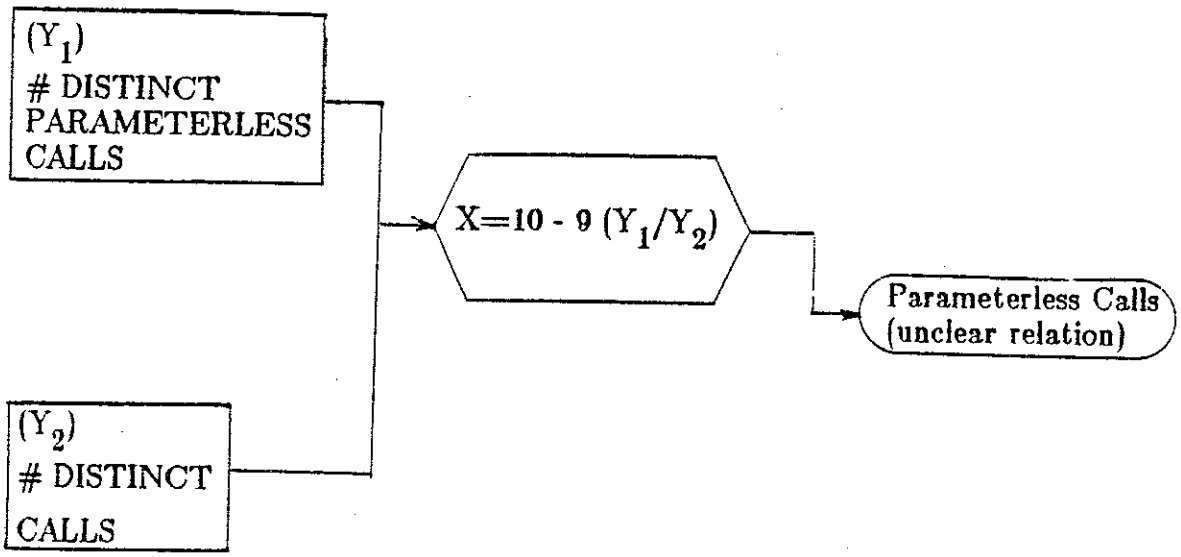
The determination of all metric values and their consequent property value is confined to an arbitrary range of one to ten. Such a range avoids a "black and white" pronouncement, for in reality properties can be exhibited to varying degrees.

ELEMENTS, METRICS, AND PROPERTIES

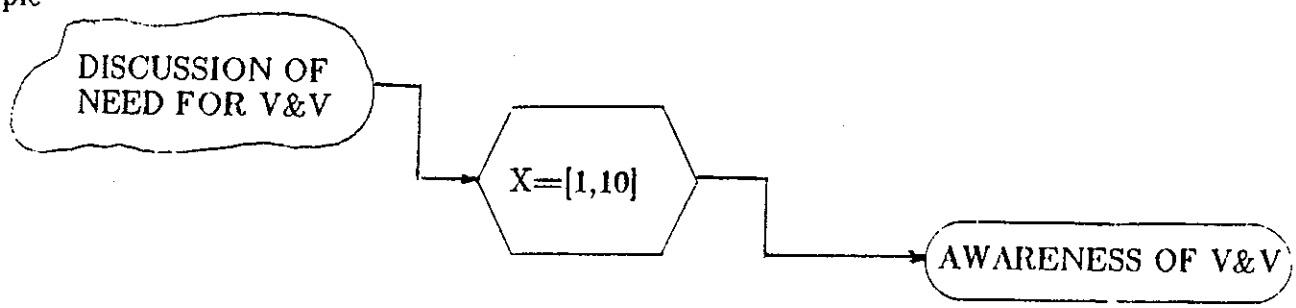
- Relationship



- Code
  Example



- Documentation
  Example



The code and documentation sampling process requires the computation of metric values, either through objective elements or subjective opinion, and the accumulation of these metric values to determine a property value. Property values are accumulated in order to give evidence of the presence or absence of an attribute. The scale of one to ten could then be divided into a continuum in which a region around five might indicate no evidence of either the presence or absence of an attribute or conflicting and offseting evidence. From six to ten the value indicates the strength of evidence for the presence of an attribute, and from one to four, the absence.

This evaluation procedure reflects a conclusion regarding a software attribute as based on an accumulation of evidence, much like a civil litigation process in which no right nor wrong can be clearly discernible. Rather, the attempt is to establish the preponderance of evidence in one direction or the other.

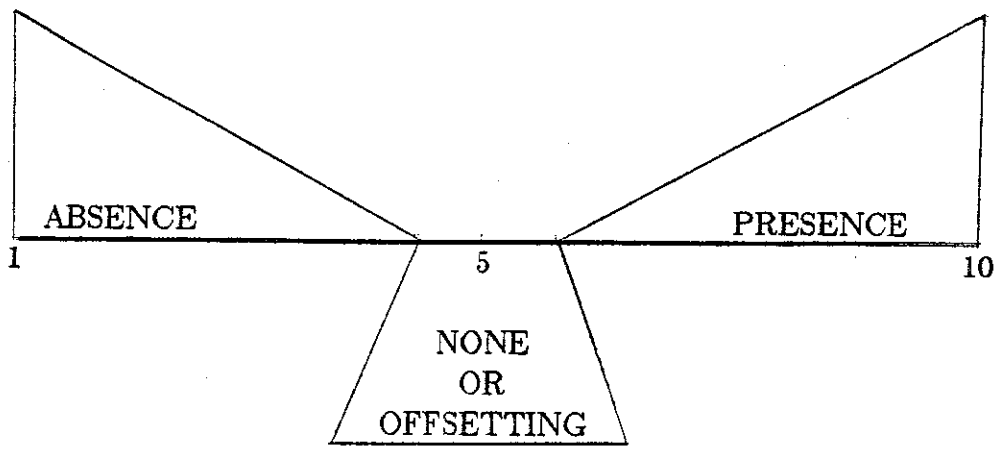
EVALUATION BY ACCUMULATION OF EVIDENCE

- METRIC GUIDELINES

- PROPERTY DETERMINATION:
  ACCUMULATION OF METRIC VALUES

- ATTRIBUTE ASSESSMENT:
  ACCUMULATION OF PROPERTIES

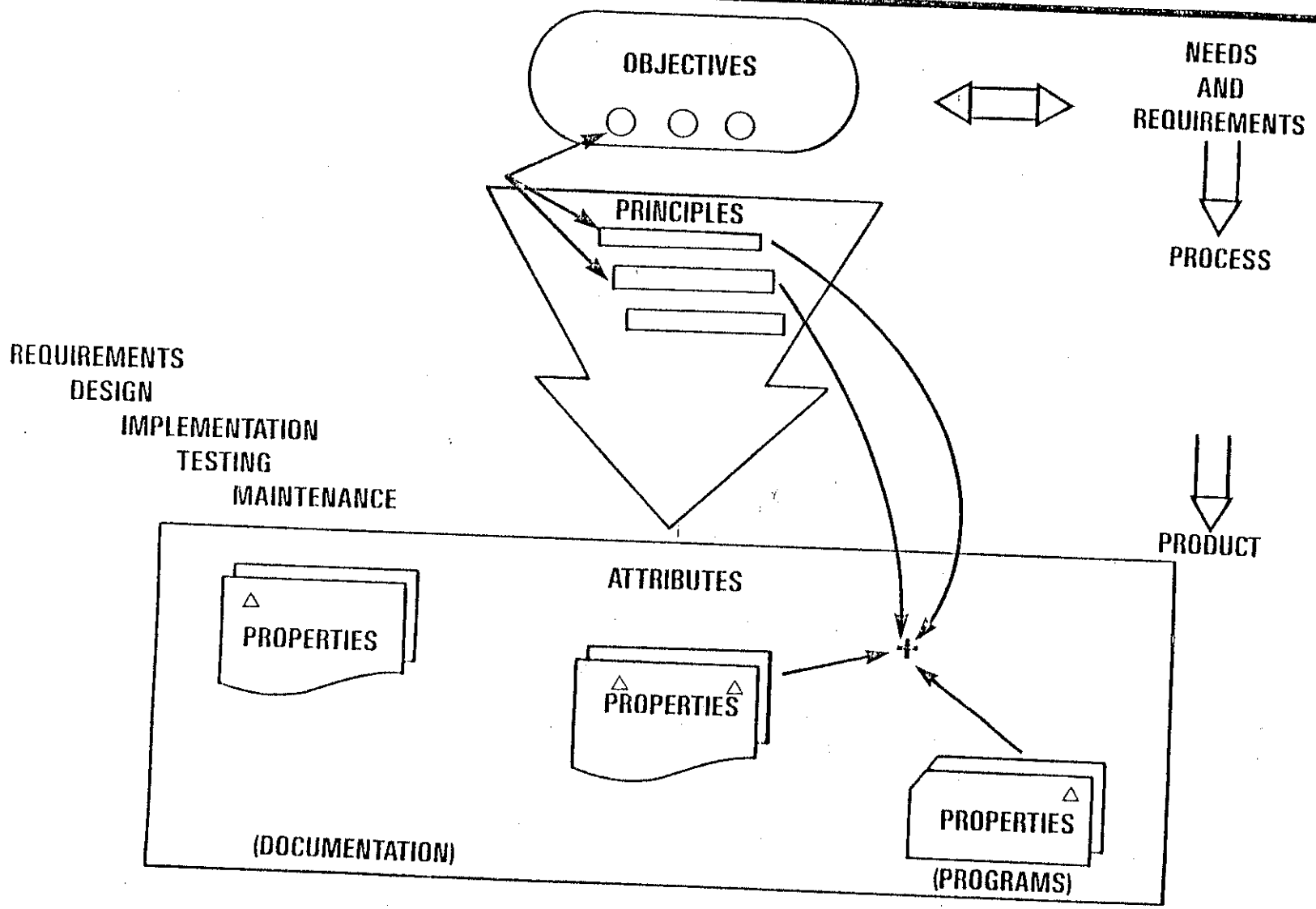
- SCALE



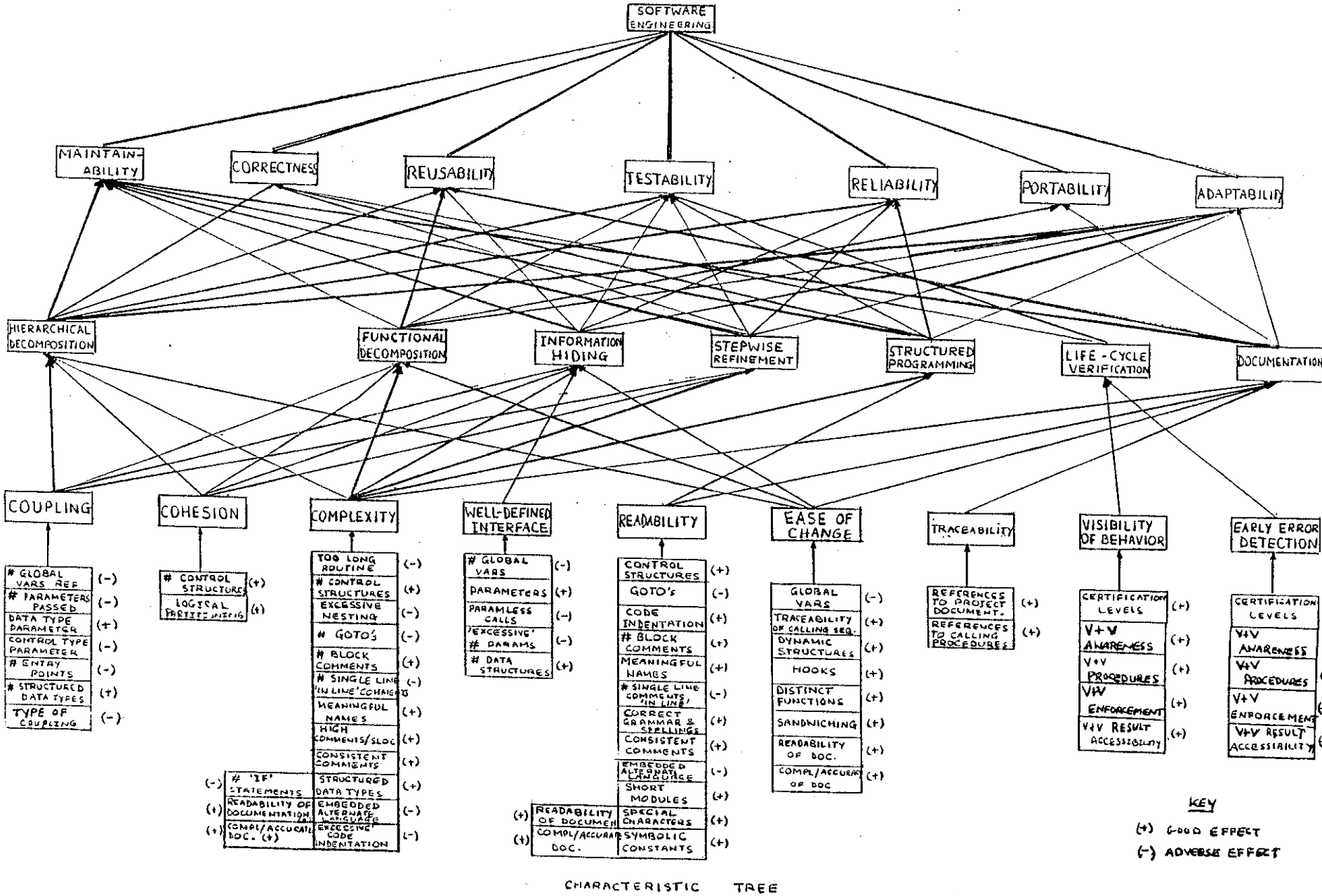
The embellished illustration of the evaluation procedure denotes the linkages among objectives and principles and among principles and attributes. Further it recognizes that properties of both documentation and source code can contribute to the evidence of the presence or absence of an attribute. This evidence should be implanted throughout the development process, from requirements to maintenance.

In summary, the logical basis for the evaluation procedure is actually a common characterization of all design problems: the accurate expression of needs and requirements and the generation of a product to meet them. The capability of the design and development process should be measured in those terms.

THE EVALUATION PROCEDURE



The complexity of the property definition underlying the attribute/principle/objective accumulation of evidence is shown in the relational diagram. A plus to the right of a property indicates a contribution to the presence of an attribute; a minus, to the absence.

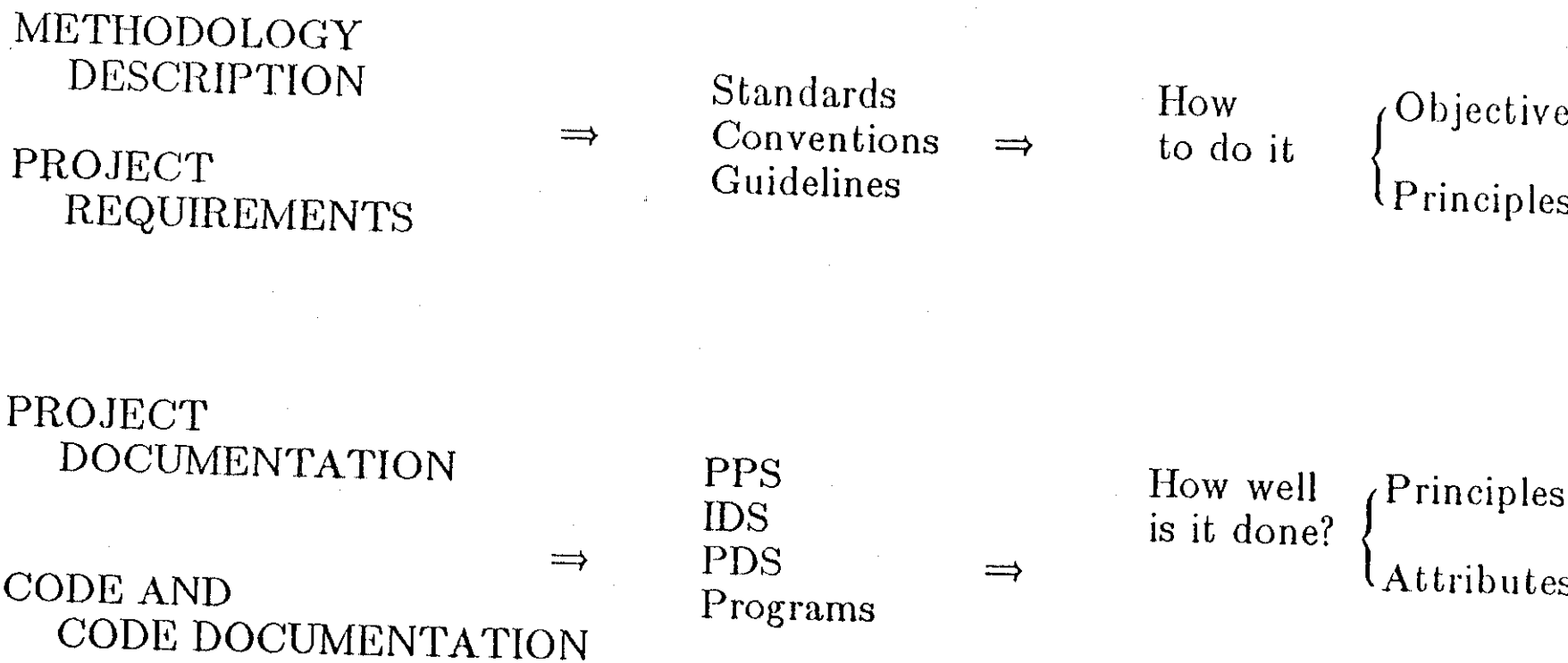


CHARACTERISTIC TREE

Evaluation of a methodology requires information describing the methodology itself as well as information derived from the application of the methodology in a particular project. Methodology description and project requirements generally provide information regarding the guidelines, conventions, or standards to be applied in producing the software. Often, objectives and principles are identified in these instructions of how to accomplish the process utilizing the methodology.

Project documentation, source code, and program documentation are sources for information related to principles and attributes. Within the program performance specifications and interface design specifications, we find information regarding the degree to which the methodology is understood and applied in a particular project. The surface properties of program design specifications and the source code provide the input to metrics for code properties, either in terms of objective elements or subjective opinion.

DATA SOURCES



APPLICATION OF THE EVALUATION PROCEDURE

- Restructured NTDS (RNTDS) for ACDS Block 0

- AEGIS Modular (AEGIS) for AEGIS Baseline 1

Altogether, twelve documents serve as primary sources for the AEGIS Modular and RNTDS samples. From these documents is extracted much of the subjective opinion contributing to those metrics related to documentation.

From the specifications of the program design, interface design, and program performance comes the data related to the source code. A total of 19 components from AEGIS and 99 from RNTDS are included.

SUMMARY OF SAMPLE DATA

DOCUMENTS (PRIMARY)

AEGIS: The AEGIS Combat System Development Plan

The AEGIS Computer Programming Manual

The CSC Program Development Manual for the AEGIS Program

Six Numbered Documents (PDS, IDS)

RNTDS: Functional Description for the Restructured NTDS Program

Two Numbered Documents (PDS, IDS)

SOURCE CODE:

AEGIS: Routines = 17 SLOC = 1170

SysProcs = 2 SLOC = 1370

PDS/IDS/PPS

RNTDS: Routines = 99 SLOC = 5729

PDS/IDS/PPS

The subjective nature of documentation metrics is apparent. Accuracy/completeness is one property and readability, another. Metrics such as connectivity, completeness of the document set, the presence of missing information, the use of appropriate presentation tools, and others are assessed with regard to one or both of these properties for each of the routines making up the sample set. An overall average is computed for the routines.

EXAMPLE OF DOCUMENTATION

METRICS AND PROPERTIES

	PROPERTIES:	ACCURACY/ COMPLETENESS	READABILITY
METRICS			
Connectivity		9.4	9.4
Completeness (Document set)		8.7	8.7
Missing Information ("TBS", "TBD")		8.7	
Use of appropriate presentation tools			9.2
Effective use of change bars			8.2
.		.	.
.		.	.
.		.	.
	Overall Average	7.9	7.9

An example of the objective nature of source code elements is shown to the right. Such items as "symbolic constants defined", "references to externally declared variables", and "number of go tos" are tallied to provide input data to the metrics defined for properties such as "use of global variables,""parameterless calls", and "use of structured data types." In total, some 62 metrics are used for source code analysis and 74 for code documentation.

MEASURABLE SOURCE CODE ELEMENTS

(Results for a single routine)

Total lines of source code (excluding comments)....	25
Data tags decl. in Proc. that are def'ed in SYS-DD.	0
Symbolic constants defined only in routine.....	0
Non-symbolic constants used in routine.....	3
Is the calling routine known?.....N	
Distinct ref's to externally declared variables....	5
Variables accessible by other routines (globals)...	195
Variables accessible only by routine (locals).....	0
Number of goto's.....	0
Number of distinct exit points.....	1
Number of distinct entry points.....	1
Number of comment lines.....	36
Number of commented references to documentation....	0
Total number of call statements.....	3
Total number of control structures used.....	2
Total number of structured data types ref'ed.....	3
Average number of parms per call.....	0.67
Maximum level of control structure nesting.....	1
Percent code enclosed in control structures.....	52
Number of comments (p27) / source lines of code....	1.44
Number of goto's / source lines of code.....	0.00
Number of exits / Number of entries.....	1.00
No. of local variables / No. of global variables...	0.00
No. single line comments / block comments.....	0.00

The comparison of AEGIS and RNTDS with regard to the nine attributes reveals the average values to be rather close. Only in the case of traceability does a marked difference appear. A noticeable difference also occurs for readability. In both cases the scores favor RNTDS. AEGIS scores slightly higher with respect to cohesion.

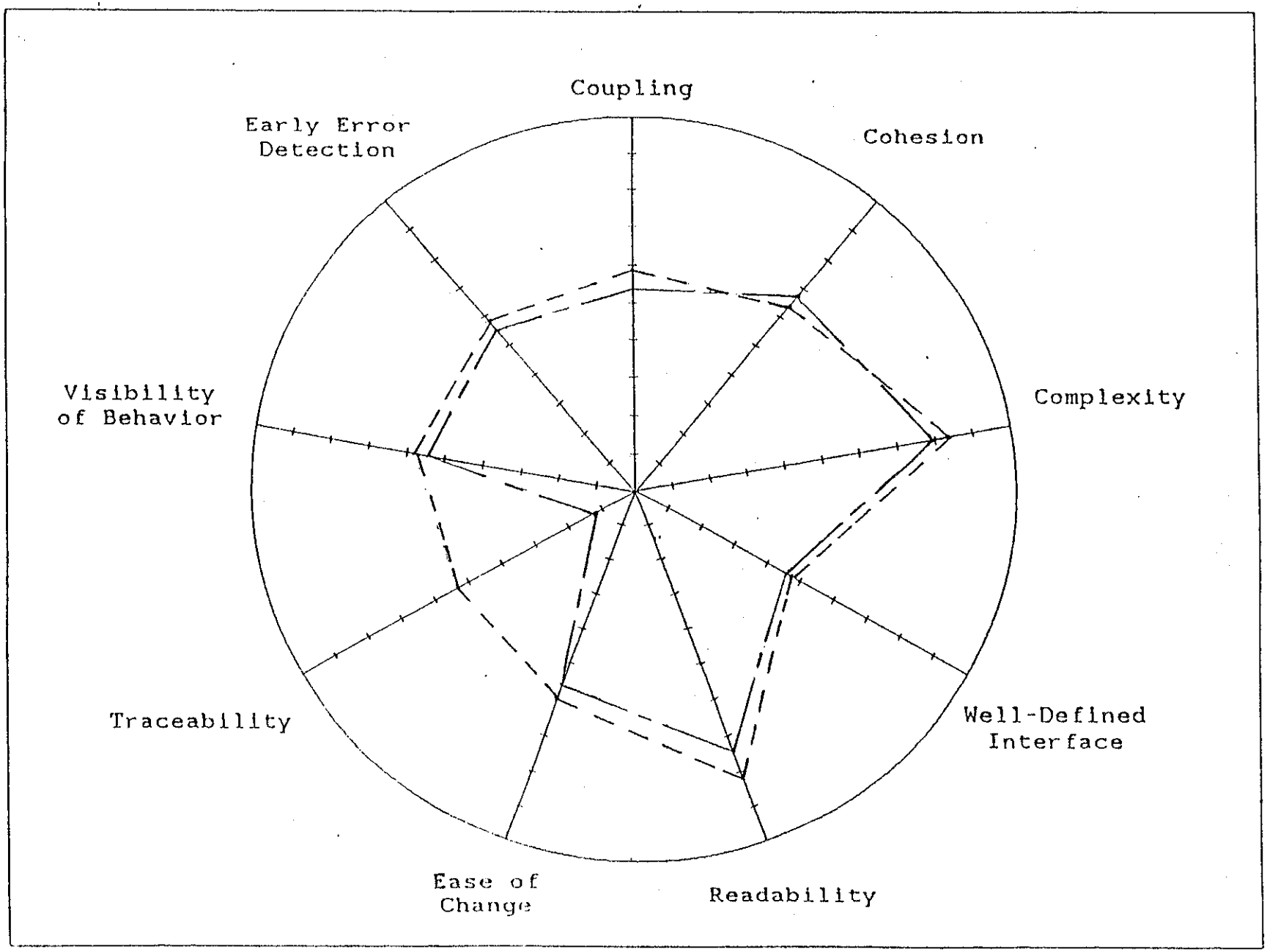
The excessive variability of RNTDS with regard to traceability gives cause for concern. Note also that both AEGIS and RNTDS exhibit wide variability for well defined interfaces.

SUMMARY OF RESULTS

ATTRIBUTES

	AEGIS				RNTDS			
	Avg	Min	Max	Range	Avg	Min	Max	Range
Coupling	5.4	4.4	9.1	4.7	5.9	4.4	8.9	4.5
Cohesion	6.8	5.0	9.3	4.3	6.4	5.0	9.2	4.2
Complexity	8.0	5.7	8.6	2.9	8.4	7.2	9.0	1.8
Well-defined Interfaces	4.7	2.2	7.4	5.2	4.8	1.8	8.1	6.3
Readability	7.4	6.4	8.4	2.0	8.2	7.2	9.2	2.0
Ease of Change	5.6	4.7	7.5	2.8	6.0	5.1	7.3	2.2
Visibility	5.6	5.6	5.6	0	5.8	5.8	5.8	0
Early Error Detection	5.6	5.6	5.6	0	5.8	5.8	5.8	0
Traceability	1.2	1.0	5.0	4.0	5.3	1.0	10.0	9.0

The Kiviat graph for attributes enables an overall comparison of the relative performance for the two methodologies. In general, the scores for attributes are very close, with a notable exception for traceability. RNTDS tends to score slightly higher than AEGIS except for cohesion.



KIVIAT GRAPH FOR ATTRIBUTES

AEGIS
RNTDS

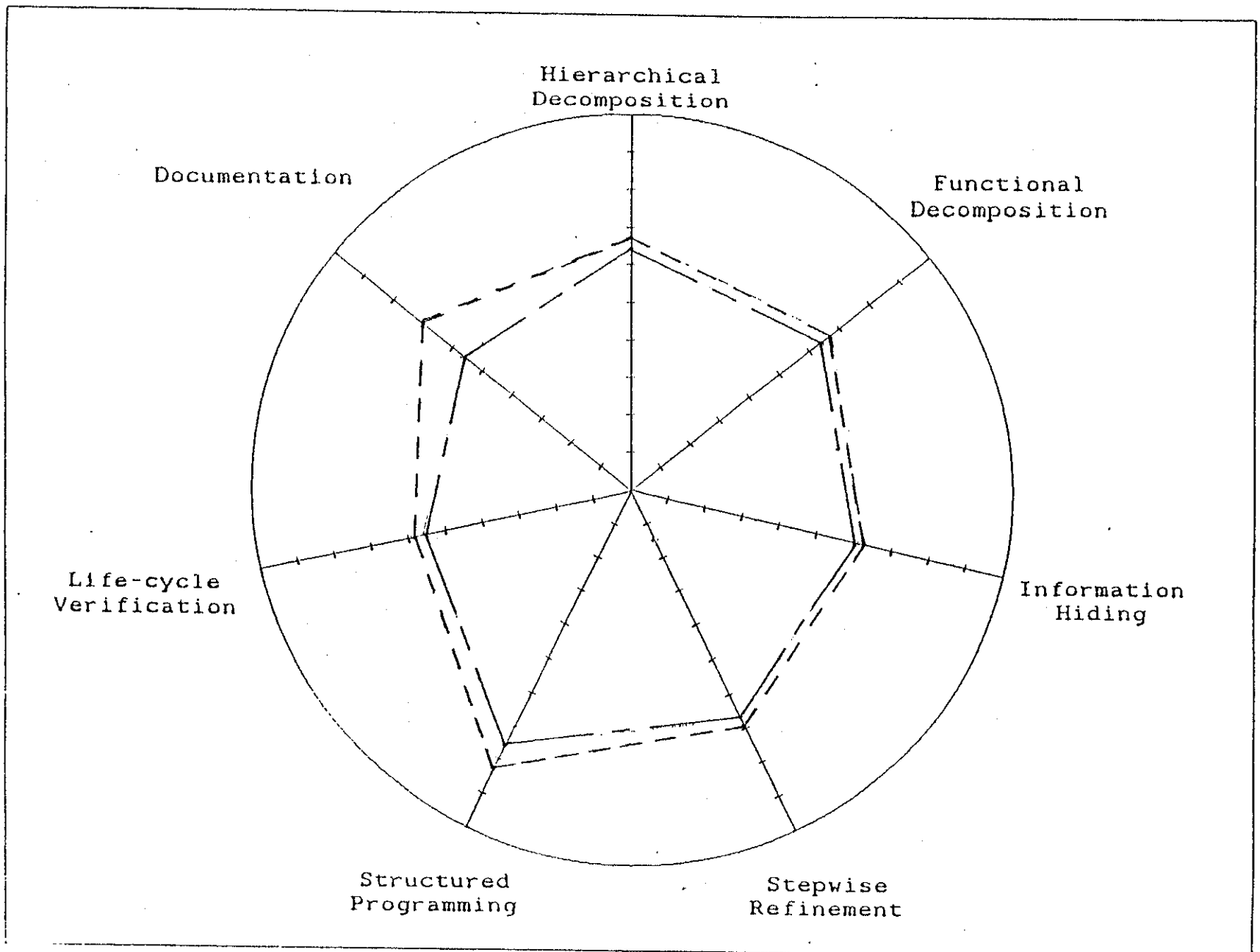
The accumulation of attribute values to reflect supporting principles reveals only small absolute differences in most cases. Notable exceptions are structured programming and documentation. With regard to the former, both methodologies score well; while RNTDS tends to score much higher for documentation.

SUMMARY OF RESULTS

PRINCIPLES

	AEGIS	RNTDS
Hierarchical Decomposition	6.4	6.7
Functional Decomposition	6.4	6.7
Information Hiding	6.1	6.3
Stepwise Refinement	6.7	6.9
Structured Programming	7.7	8.3
Life-cycle Verification	5.6	5.8
Documentation	5.6	7.0

The Kiviat graph for principles shows the generally higher scores for RNTDS. Differences are small except for documentation and structured programming, but the uniformly better performance of RNTDS in the support of principles is judged significant.



KIVIAT GRAPH FOR PRINCIPLES

AEGIS ———————
RNTDS - - - - - - -

The scoring for objectives in some sense forms the bottom line. In all cases higher scores are realized by RNTDS. It is difficult to judge whether the differences are "significant." Certainly, the nature of the sampling and the evaluation procedure at this point does not permit a determination of "statistical significance." However, the uniformly higher scores of RNTDS as a result is judged significant by the research team.

SUMMARY OF RESULTS

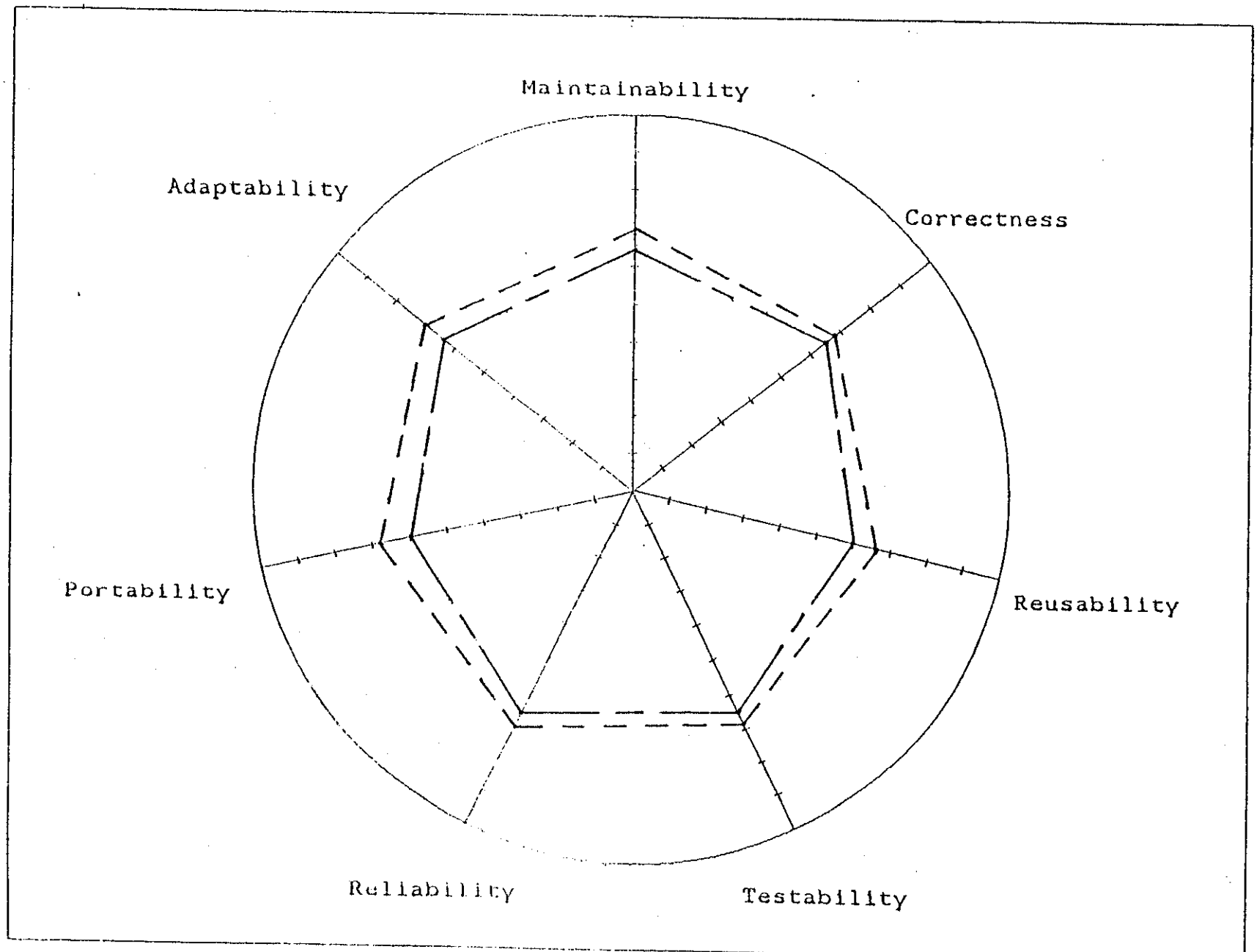
OBJECTIVES

	AEGIS	RNTDS
Maintainability	6.4	7.0
Adaptability	6.4	7.0
Reusability	6.0	6.7
Portability	6.0	6.8
Testability	6.5	6.8
Reliability	6.7	7.0
Correctness	6.5	6.8

As in the case for Principles, the Kiviat graph for Objectives shows the uniformly higher scores for RNTDS. The magnitude of the differences cannot be easily judged.

- The accumulative nature of the evaluation procedure tends to dampen the differences (a low score on one property is offset by a higher score on another)

- More experience with the evaluation procedure would improve our understanding of its discriminatory capabilities.



KIVIAT GRAPH FOR OBJECTIVES

AEGIS ——— — ———
RNTDS — — — — — —

With respect to conclusions and recommendations, the research team has briefly summarized both in the following pages. A more indepth statement together with commentary is included on the following pages. Commentary is bracketed because in some cases the opinion reflected in these comments is not derived from the study alone. Nevertheless, the team believes that it is useful to share these reactions.

CONCLUSIONS

1. AEGIS Modular:

   Principle of Structured Programming
   Reliability Objective

2. RNTDS:

   Principles of Structured Programming, Documentation, Maintainability, Adaptability, Reliability Objectives

3. Both fail to support information hiding

4. Both have difficulty with traceability and well-defined interfaces

5. RNTDS achieves software engineering objectives to greater degree

6. Implementation dependencies prohibit ADA insertion benefits

CONCLUSIONS AND COMMENTARY

1. AEGIS EMPHASIZES THE PRINCIPLE OF STRUCTURED PROGRAMMING AND SOFTWARE RELIABILITY IS MOST PROMINENT AMONG THE REALIZED OBJECTIVES.

2. RNTDS EMPHASIZES THE PRINCIPLES OF STRUCTURED PROGRAMMING AND DOCUMENTATION; MAINTAINABILITY, ADAPTABILITY AND RELIABILITY ARE PROMINENT AMONG THE REALIZED OBJECTIVES.

3. RELATIVELY LOW MARKS FOR THE PRINCIPLE OF INFORMATION HIDING INDICATE THAT BOTH METHODOLOGIES HAVE DIFFICULTY IN SUPPORTING SECURE, EXPLICIT COMMUNICATIONS AMONG COMPONENTS.

DIFFICULTIES WITH COMMUNICATIONS CONSTRUCTS ARE A MAJOR SOURCE OF INCOMPATIBILITY WITH ADA-DERIVED SOFTWARE DESIGN OR ADA-BASED METHODOLOGIES.

4. WITH REGARD TO ATTRIBUTES AND BASED ON THE PROPERTIES DEFINED, RNTDS EXHIBITS HIGH VARIABILITY AMONG ROUTINES IN TRACEABILITY AND WELL-DEFINED INTERFACES.
BASED ON THE PROPERTIES DEFINED, AEGIS EXHIBITS LITTLE EVIDENCE OF TRACEABILITY.

HIGH VARIABILITY AMONG ROUTINES FOR A GIVEN ATTRIBUTE REPRESENTS A POTENTIAL LACK OF UNDERSTANDING OF A METHODOLOGY OR THE INADEQUATE SUPPORT OF PRINCIPLES CONTRIBUTING TO THE PRESENCE OF THE ATTRIBUTE.

5. WHILE BOTH AEGIS AND RNTDS METHODOLOGIES SCORE WELL, RNTDS REALIZES THE SOFTWARE ENGINEERING OBJECTIVES TO A GREATER DEGREE THAN AEGIS.

PERFORMANCE (CAPABILITY) IS NOT A SOFTWARE ENGINEERING OBJECTIVE; IT IS AN ACCEPTED GOAL OF THE SYSTEMS ENGINEERING TASK. ONE CAN INTERPRET THE AEGIS RESULTS AS THE SACRIFICE IN SOFTWARE ENGINEERING OBJECTIVES TO ACHIEVE THE SYSTEMS ENGINEERING OBJECTIVE OF HIGH PERFORMANCE.

EMBEDDED SOFTWARE DEVELOPMENT METHODOLOGIES MUST BE RESPONSIVE TO THE OBJECTIVES OF THE ENCAPSULATING SYSTEM YET MUST EXERT AN INFLUENCE ON THE SYSTEMS ENGINEERING DECISIONS.

- DOES RNTDS HAVE A CLEAR, ACCURATE PERCEPTION OF THE ENCAPSULATING SYSTEM(S)?
- DOES AEGIS PROVIDE A BASIS FOR TRADEOFF ASSESSMENT?

6. BOTH METHODOLOGIES ARE TIED TO IMPLEMENTATION DEPENDENCIES (CMS-2, UYK 7, SYSTEM ARCHITECTURE) SO THAT INSERTION OF AN ADA-DERIVED "METHODOLOGY" WOULD (A) REQUIRE REDESIGN OF THE ENCAPSULATING SYSTEM (AEGIS OR RNTDS) OR (B) A SIGNIFICANT DEGRADATION OF THE ADA CONCEPTS AND BENEFITS.

CONSIDERING THE HISTORICAL TRENDS IN HARDWARE (SPEED, SIZE, AND COST) AND THE UNMISTAKABLE TREND TOWARD DISTRIBUTED PROCESSING, FOR THE LONG TERM IT WOULD SEEM MORE PRUDENT TO EMPHASIZE THE FACTORS AFFECTING LIFE-CYCLE SOFTWARE COST RATHER THAN PERFORMANCE.

THE DIFFICULTY IN AN ADA-DERIVED METHODOLOGY TRANSITION IS GREATER FOR AEGIS THAN FOR RNTDS, BUT THE DIFFERENCE IS NOT CONVINCING FOR THE POTENTIAL STRATEGY OF CONVERTING AEGIS TO RNTDS FOLLOWED BY A TRANSITION TO ADA.

PRIOR EFFORTS TO USE ADA AS A SPECIFICATIN OF DESIGN LANGUAGE FOLLOWED BY IMPLEMENTATION IN CMS-2 HAVE SUPPORTED THIS CONCLUSION.

RECOMMENDATIONS

1. Achieve commonality through ADA evolution

2. Converged CDS system objectives be prioritized to drive software objectives

3. PGE working group should identify software development methodology

4. Give visibility and emphasis to software engineering objectives

5. Initiate task to investigate indicator concept

RECOMMENDATIONS AND COMMENTARY

1. PROCEED WITH DEFINITION OF A CONVERGED CDS BUT RECOGNIZE IT AS AN INTERMEDIATE-TERM SOLUTION NECESSARY BECAUSE AEGIS AND RNTDS UPGRADES WILL EXTEND TO THE YEAR 2000 AND BEYOND.

2. THE AEGIS/ACDS COMMONALITY WORKING GROUP SHOULD BE GIVEN A CLEAR STATEMENT OF THE SYSTEM OBJECTIVES SO THAT THE IMPLICATIONS WITH RESPECT TO PRIORITIZATION OF OBJECTIVES FOR SOFTWARE DEVELOPMENT CAN BE ASSESSED. THE SYSTEMS' ENGINEERING AND SOFTWARE ENGINEERING REQUIREMENTS MUST BE CLEARLY DELINEATED.

3. THE PROGRAM GENERATION ENVIRONMENT SUBGROUP SHOULD GIVE IMMEDIATE ATTENTION TO THE IDENTIFICATION OF A SOFTWARE DEVELOPMENT METHODOLOGY.

A REVIEW OF TOOLS AND TOOL AVAILABILITY IS HELPFUL, BUT CONFIGURING AN ENVIRONMENT BASED ON AN ACCUMULATION OF TOOLS IS AKIN TO COMPOSING A SYMPHONY BASED ON THE AVAILABILITY OF INSTRUMENTS AND MUSICIANS.

4. SOFTWARE ENGINEERING OBJECTIVES BE EMPHASIZED DURING THE SYSTEMS ENGINEERING TASK SO THAT EARLY VISIBILITY IS GIVEN TO THE CONSEQUENCES OF DECISIONS AND THE IMPLICATIONS FOR THE SOFTWARE SPECIFICATION AND DESIGN.

AS ONE MOVES FROM REQUIREMENTS TO SPECIFICATION TO DESIGN, THE ABILITY TO REALIZE SOFTWARE ENGINEERING OBJECTIVES IS INCREASINGLY CONSTRAINED. SOFTWARE ENGINEERING OBJECTIVES MUST BE ELEVATED TO A PRIMARY LEVEL.

5. A TASK BE INITIATED TO INVESTIGATE THE INDICATOR CONCEPT AS A MEANS OF AUTOMATING THE ANALYSIS OF SOFTWARE (DOCUMENTATION AND CODE) AND TO EXTEND THE EVALUATION PROCEDURE AS A BASIS FOR SOFTWARE QUALITY ASSURANCE.

EXTENSION OF THE EVALUATION PROCEDURE TO THE AUTOMATIC ASSESSMENT OF CONFORMANCE WITH SPECIFICATIONS OF QUALITY IS EXCITING AND POTENTIALLY EXTREMELY BENEFICIAL.

The additional data with regard to DDG51 software development has been collected. The analysis is in progress, and the summary of the results will be provided by 1 February 1986.

Support for the PGE working group continues in the efforts of Dr. Arthur to assist Dave McConnell in the top level requirements document for the Converged CDS methodology.

Support is sought to extend the evaluation procedure by investigating the use of statistical "indicators" as a basis for incorporating a more objective analysis. Indicators are measurable characteristics which indirectly reflect non-measurable characteristics. They have been used extensively in certain areas of the social sciences. Further, the extended investigation would enable the development of more automated procedures for code and documentation analysis in order to reduce the high demands on human effort in the sampling process.

FUTURE PLANS

• OBTAIN ADDITIONAL AEGIS (DDG-51) DATA	30 NOV
• CONTINUE TO SUPPORT PGE WORKING GROUP (E.G., TOP LEVEL REQUIREMENTS DOCUMENT FOR CONVERGED CDS SOFTWARE DEVELOPMENT METHODOLOGY)	31 DEC
• FINAL REPORT DOCUMENT (CURRENT TASK)	15 JAN
• EXPLORE DEVELOPMENT OF "INDICATORS" TO PROVIDE A MORE QUANTITATIVE ANALYSIS	FUTURE
• INVESTIGATE A MORE AUTOMATED EVALUATION PROCEDURE	FUTURE

APPENDICES

APPENDIX A

James D. Arthur, Sallie M. Henry, and Richard E. Nance, "Immediate Software Development Issues for Embedded Systems Applications in Surface Combatants," Department of Computer Science, TR-85-19, 15 March 1985.

APPENDIX B

Sallie M. Henry, James D. Arthur, and Richard E. Nance, "A Procedural Approach to Evaluating Software Development Methodologies," Department of Computer Science, TR-85-20, 30 March 1985.

Note: Differences in content from the final report reflect the continuing refinement of the evaluation procedure.

IMMEDIATE SOFTWARE DEVELOPMENT ISSUES

for

EMBEDDED SYSTEMS APPLICATIONS IN SURFACE COMBATANTS

List of Source Documents

Aids Description and Applications Manual, AEGIS Ship Combat System, Naval Surface Weapons Center, Dahlgren, Virginia, March, 1985.

AEGIS ADA Investigation CDR-ADA First Cut', No. FS-85-058, Fleet Systems Department, Applied Physics Laboratory, Johns Hopkins University, March, 1985.

"AEGIS/ACDS Commonality/Convergence: Working Groups; Charters and Procedures," CDS, NAVSEA, 28 March 1985.

Appendix A, Existing Executives. (2 copies)

Beck, Leland L. and Perkins, Thomas E., "A Survey of Software Engineering Practice: Tools, Methods and Results," Transactions on Software Engineering, Vol. SE-9, No. 5, September 1983, pp. 541-561.

Boehm, Barry W., "Seven Basic Principles of Software Engineering," The Journal of Systems and Software, Vol. 3, 1983, pp. 3-24.

Bowen, John B., "Module Size: A Standard or Heuristic?," The Journal of Systems and Software, Vol. 4, 1984, pp. 327-332.

Bowen, Thomas P., Wigle, Gary B. and Tsai, Jay T., "Program Generation Environment Working Group," Report RADC-TR-85-37, Rome Air Dev. Center, Boeing, February 1985. (also TAD 501393(S) 4/24/85)

Britton, K. H. and D. L. Parnas, "A-7E Software Module Guide," NRL Memorandum Report 4702, Naval Research Laboratory, Washington, D. C., December, 1981.

Britton, K. H., R. A. Parker, and D. L. Parnas, "A Procedure for Designing Abstract Interfaces for Device Interface Modules," Code 7590, Naval Research Laboratory, Washington, D. C., 198?.

Chief of Naval Operations, Memo Subject: Implementation of Tadil J/JTIDS and Related Tactical, Department of the Navy, Office of the Chief of Naval Operations, Washington, D. C., 1984.

Clements, P. C., "Function Specifications for the A-7E Function Driver Module," NRL Memorandum Report 4658, Naval Research Laboratory, Washington, D. C., November, 27, 1984.

Clements, P. C., R. A. Parker, D. L. Parnas, J. Shore, and K. H. Britton, "A Standard Organization for Specifying Abstract Interfaces," NRL Report 8815, Naval Research Laboratory, Washington, D. C., June, 1984.

Cohen, Jacques and Kolodner, Stuart, "Estimating the Speedup in Parallel Parsing," IEEE Transactions on Software Engineering, Vol. SE-11, No. 1, January 1985, pp. 114-124.

Cohen, N. H., "Guidelines for the Selection of Identifiers in ADA Programs," SofTech, Inc., Huntingdon Valley, PA, 1983. (Includes other short papers: Guidelines for Selection of ADA Identifiers; Managing the Name Space; Separate Compilation Issues; ADA Style Guide; Comments on Recent ADA Experience with N30.)

Common Combat Direction System, Program Generation Environment (PGE) Working Group, "Plan of Action and Milestones," 5 April 1985.

Compteck Research, Inc., "ATES/RNTDS Comparative Analysis/Assessment Report," Arlington, VA, 1982. (2 copies)

Computer Program Development Philosophy, 1985.

Computer Sciences Corporation, "CG-47 Architectural Description Document," Vol. 1, 3, and 4 (three separate documents), New Jersey, June 1980.

Converse, R. A. and K. Paige, "Rationale for an ADA Software Engineering Environment for Navy Mission Critical Applications," ASNE Day Comments, Technical Session Papers, Naval Engineers Journal, July 1984.

Cook, D., et al., The CSC Computer Program Development Manual for the AEGIS Program, Computer Sciences Corporation, August 1979.

Dapra, A., et al., "Using ADA and APSE to Support Distributed Multimicroprocessor Targets," Ada Letters, Vol. 3, No. 6, May-June 1984, pp. 6.57-6.65.

Department of Defense, "Military Standard Software Development," DOD-STD-1679A (Navy), October 1983.

Department of the Navy, "Results of RNTDS Review Committee," (RNTDS Review Committee Report Dated 6 November 1984 and NAVSEA 61Y Memorandum Ser 61Y/071 of 16 August 1984), Naval Sea Systems Command, Washington, D.C., 1984.

Dijkstra, E. W. "Cooperating Sequential Processes," in F. Genuys (ed.), Programming Languages, Academic Press, 1968, pp. 43-112.

Downes, V. A. and S. J. Goldsack, "The Use of the ADA Language for Programming a Distributed System," Proceedings IFAC/IFIP Workshop on Real-Time Programming, 1980, pp. 39-44.

DSD Education, "AN/UYK-7 Study Guide CMS-2Y," Vol. II, PX-5852-2-7, UNIVAC Defense Systems Division, St. Paul, Minnesota, November 1977.

DSD Education, "AN/UYK-7 Study Guide Functional Description," UNIVAC Defense Systems Divison, St. Paul, Minnesota, March 1973.

Dunsmore, H. E. and J. D. Gannon, "Programming Factors -- Language Features that Help Explain Programming Complexity," ACM 0-89791- 000,January 1978, pp. 554-560.

Fischer, H., "MIL-STD-SDS Review Issues: Ada and Design Methodologies," A Report on the AdaTEC Review, Litton Data Systems, Van Nuys, CA, 1984.

Fisher, D. A. and J. N. Buxton, "Stoneman: Requirements for ADA Programming Support Environments," Department of Defense Report, February 1980.

Fischer, H., "MIL-STD-SDS Review Issues: Ada and Design Methodologies: A Report on the AdaTEC Review," ADA Letters, Vol. IV, No. 1, July/August 1984, pp. IV-1.7-IV-1.16.

Fleet Combat Direction Systems Support Activity, "CMS-2Y Programmers Reference Manual," M-5049, San Diego, CA, December 1976.

Fleet Combat Direction Systems Support Activity, "CMS-2Y Supporting Subsystems - Preliminary Copy," M-5050, San Diego, CA, December 1976.

Fleet Combat Direction Systems Support Activity, "User's Reference Manual (U) for Compiler, Monitor System-2 (CMS-2) for Use with AN/UYK-7 Computer," Vol. 1, M-5035, San Diego, CA, August 1975.

Fleet Combat Direction Systems Support Activity, "Program Design Specification for Introduction and System Data," NR-PDS-000-U-L, Dam Neck, Virginia Beach, VA, March 1980, revision September 1982.

Fleet Combat Direction Systems Support Activity, "Restructured Naval Tactical Data Systems Production Control Handbook," NR-H-000-U-L, Dam Neck, Virginia Beach, VA, February 1984.

Freeman, P., A. I. Wasserman, and R. C. Houghton, Jr., "Comparing Software Development Methodologies for ADA: A Study Plan," ACM Software Engineering Notes, Vol. 9, No. 4, July 1984,

pp. 22-55.

Gerson, G.M., and B.J. Knapp, "Report on Recommendations for AEGIS Lifetime Support Tools and Methodologies," No. T-0178, Database and Information Services Division, CACI, Inc., Washington, D.C., June, 1985.

Gordon, Michael, "The Byron* Program Development Language," Journal of Pascal and Ada May/June 1985, pp. 24-28.

Gordon, Michael, "Byron (TM) and The Byron/Ada PDL Toolset," Intermetrics, Cambridge, Massachusetts, ca. 1985.

Green, D.T., Memorandum containing information on ADA, RNTDS, and AEGIS, 1984.

Hamilton, M. and S. Zeldin, "The Functional Life Cycle Model and Its Automation: USE.IT," Journal of Systems and Software, 1983, Vol. 3, pp. 25-62.

Harrison, R., "Thoughts on Precisely - Timed Interrupt," February 8, 1984.

Harrison, R.D. and D.C. Murphy, "Transporting an Existing Real-Time Control System Design into the ADA Language (An Interim Report)," NSWC TR 83-xxx, Naval Surface Weapons Center, Dahlgren, VA, 1983.

Heninger, K. L., J. W. Kallander, J. E. Shore, and D. L. Parnas, "Software Requirements for the A-7E Aircraft," NRL Memorandum Report 3876, Naval Research Laboratory, Washington, D. C., November, 1978.

Heninger, K. L., "Specifying Software Requirements for Complex Systems: New Techniques and Their Application," IEEE Transactions on Software Engineering, Vol. SE-6, No. 1, January 1980.

Hoare, C. A. R. "Monitors: an Operating System Structuring Concept," Communications ACM, 17 (10): October 1974, pp. 549-557.

Hosseini, S. H., Kulh, Jon G., and Reddy, Sudhakar M., "A Diagnosis Algorithm for Distributed Computing Systems with Dynamic Failure and Repair," IEEE Transactions on Computers, Vol. C-33, No. 3, March 1984, pp. 223-233.

Ichbiah, J., "ADA: Past, Present, Future: An Interview with Jean Ichbiah, the Principal Designer of ADA," Communications of the ACM, Vol. 27, No. 10, October 1984, pp. 99-997.

IEEE Guide to Software Requirements Specifications, IEEE Computer Society, Report No. Std 830-1984, New York, NY, 1984.

IEEE Standard for Software Quality Assurance Plans, IEEE Computer Society, Report No. Std 730-1984, New York, NY, 1984.

IEEE Standard for Software Test Documentation, IEEE Computer Society, Report No. Std 829-1983, New York, NY, 1983.

IEEE Standard Glossary of Software Engineering Terminology, IEEE Computer Society, Report No. Std 729-1983, New York, NY, 1983.

Information Technology Review, NRL Brief 2, 4 Oct. 84.

Jensen, Howard A. and K. Vairavan, "An Experimental Study of Software Metrics for Real-Time Software," IEEE Transactions on Software Engineering, Vol. SE-11, No. 2, February 1985, pp. 231-234. Kamrad, J. M., II, "Runtime Organization for the ADA Language System Programs," ADA Letters, Vol. III, No. 3, 1983.

Langston Commission -- Purpose and Approach

LeGrasso, J. M., "RCA/Navy (PMS-400) Joint Interpretation of DOD-STD-1679A for DDG Computer Program Development," RCA, 19 June 1984.

Lindquist, T. L. and D. G. Kafura, "The Support and the use of ADA in Distributed Environments," A Draft Proposal submitted to the Department of Defense, 1983.

Liskov, B. and ? Berzins, "An Appraisal of Program Specifications," in P. Wegner, Research Directions in Software Technology, MIT Press, 1978.

Liskov, B. and S. Zilles, "Specification Techniques for Data Abstractions," IEEE Tramsactions on Software Engineering, SE-1 (1): March 1975, pp. ?.

Lohse, John B. and Zweben, Stuart H., "Experimental Evaluation of Software Design Principles: An Investigation into the Effect of Module Coupling on System Modifiability," The Journal of Systems and Software, Vol 4., 1984, pp. 301-308.

Masters, M. W. and M. J. Kuchinski, "Software Design Prototyping Using ADA," Naval Surface Weapons Center, Dahlgren, VA, September 1983.

Mathis, R. F., "Memorandum for the Record: Results of Final Review of STARS FY1985 Program Plan," OSD $E, 4 October 1984.

McConnell, D.E., "Automated Integrated Database System (AIDS)," Naval Surface Weapons Center, Dahgren, VA, 18 May 1984.

McConnell, D.E., "DNL Panel on Software Engineering

Environments," NSWC: N23, Naval Surface Weapons Center, Dahlgren, Virginia, December 21, 1984.

McConnell, D.E., "Evaluation/Validation Candidates for VPI Methodology," June, 1985.

Nance, R. E., NRL Brief 1, Information Technology Review, October 1984.

Nance, R. E., NRL Brief 2, Information Technology Review, October 1984.

Naval Ocean Systems Command, "System Specification for ADA Language System/Navy," NAVSEA 0967-LP-598-9710, Computer Architecture Branch, San Diego, CA, September 1983.

Naval Systems Department, "Computer Program Development Plan," RCA/Government Systems Division, Moorestown, NJ, June 1984.

Naval Systems Department, "Program Performance Specification for Aegis Tactical Executive Systems (ATES/43)," RCA/Government Systems Division, Moorestown, NJ, December 1983.

NAVSEA Subproject Program Plan, Several small papers together, below a listing of numbers and titles (1983):
SF 57-525 - Human Factors Engineering Technology for Ships
SF 12-133 - Multisensor Surveillance and Tracking (U)
SF 21-211 - Command and Control System Assessment (U)
SF 21-244 - Shipboard Internal Communications
SF 21-232 - Ship/Submarine Navigation Technology (U)
SF 21-242 - Shipboard Information Management, Assessment, and Display (U)
SF 21-243 - Computer Software Technology (U)
41-411 - Concept Assessment of Platforms and Systems

NSWC Report entitled "AEGIS Combat System Computer System Architecture."

NSWC Report entitled "DDG-51 Control System Computer System Architecture," February 1982.

NSWC Report entitled "Characteristics of Computer Hardware/Software and Interconnection Components of the Combat System," September 1982.

Nyari, Erika and Sneed, Harry, "SOFSPEC: A Pragmatic Approach to Automated Specification Verification," The Journal of Systems and Software, Vol. 3, 1983, pp. 193-200.

Paige, K. K. and R. A. Converse, "Rationale for an ADA Software Engineering Environment for Navy Mission Critical Applications," Naval Engineers Journal, July 1984.

Parker, R. A., K. L. Heninger, D. L. Parnas, and J. E. Shore,

"Abstract Interface Specifications for the A-7E Device Interface Module," NRL Memorandum Report 4385, Naval Research Laboratory, Washington, D. C., November, 1980.

Parnas, D. L. "Information Distribution Aspects of Design Methodology," Proceedings of IFIP Congress 71, North Holland Publishing Company, TA-3, 1972, pp. 26-30.

Parnas, D. L. "On the Criteria To Be Used in Decomposing Systems into Modules," Communications ACM, 15 (12):December 1972, pp. 1053-1058.

Parnas, D. L. "Use of Abstract Interfaces in the Development of Software for Embedded Computer Systems," NRL Report 8047, June 1977.

Parnas, D. L., P. C. Clements, and D. M. Weiss, "Enhancing Reusability with Information Hiding," Computer Science and System Branch, U. S. Naval Research Laboratory, September 1983.

Parnas, D. L., P. C. Clements, and D. M. Weiss, "The Modular Structure of Complex Systems," 7th International Conference on Software Engineering, Orlando, Florida, March, 1984.

Prather, Ronald E., "An Axiomatic Theory of Software Complexity Measure," The Computer Journal, Vol. 27, No. 4, 1984, pp. 340-347.

Prindle, Franklin "Beyond Efficient CMS-2 Programming," Naval Air Development Center Report (Code 5032), undated.

Reynolds, P. F., Jr., "Automated Dynamic Partitioning of Networks for Distributed Simulation," Department of Computer Science, University of Virginia, 1984.

RNTDS Programmers Reference Manual and CDS Model 5 Systems Engineering Handbook Series, ACDS (Advanced Combat Direction Systems) Report CD700376(S)-4/24/85-1.

SEATECS: Software Engineering Automation for Tactical Embedded Computer Systems
Top Level Requirements, Naval Ocean Systems Center, San Diego, CA, Aug. 31, 1983.

Schill, J., R. Smeaton and R. Jackman, "The Conversion of Command & Control Software to ADA: Experiences and Lessons Learned,"
ADA Letters, Vol. IV, Issue 4, January-February 1985.

Schmid, M.E., "CDR-ADA Presentation to PGE Working Group," Report 85-1477, Applied Physics Laboratory, John Hopkins University, 1985.

SE Systems Exploration Group, Minutes of May 1&2 1985 meeting; DNL SEE Development Plan (23 January 1985); letter, NAVSEA to CNO, re: "Implementation of TADIL J/JTIDS and Related Tactical Software (10 April 1985).

"Software Engineering Environment Development Plan," Director of Naval Laboratories, Naval Material Command, 30 Jan 1985.

Stadick, E.M., "A Real-Time Control System Implementation Study Using the ADA Programming Language," NSWC TR 83-213, Naval Surface Weapons Center, Dahlgren, VA, 1983.

Stars Joint Service Team for Software Engineering Environments, "Operational Concept Document (OCD)," 400 Army Navy Drive, Arlington, VA, 1984.

Stoegerer, J.K., "A Comprehensive Approach to Specification Languages," Austrailian Computer Journal Vol. 16 Issue 1, February 1984, pp. 1-13.

Stone, W.H., "Implications of the Use of ADA in Naval Tactical Computer Systems: Interim Report," NSWC TR 84-81, Combat Systems Department, Naval Surface Weapons Center, Dahlgren, Virginia, April 1984.

Stuebing, H. G., "A Software Engineering Environment (SEE) for Weapon System Software," IEEE Transactions on Software Engineering, Vol. SE-10, No. 4, July 1984, pp. 384-397.

"Use of PSL/PSA on the ADA/M(44) Project," No. T-366, Falls Church VA, February 5, 1985.

Wakeen, G., Memorandom including short papers on: Report on Testing of Interim Ballistics Module 4 Package; ADA Program Listing of Module 4 Package, BFUTARPO; Program Listing for Driver/Test Procedure, DRIVMOD4; Data Printout Yield for Test Case 1, 2, and 3; Flowchart Original, Module 4; Flowchart Modified/Condensed, Module 4. Naval Surface Weapons Center, Dahlgren, VA, June 1984.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: SRC-86-010

2. GOVT ACCESSION NO.:

3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): EVALUATION OF SOFTWARE DEVELOPMENT METHODOLOGIES Final Report of the Immediate Software Development Project

5. TYPE OF REPORT & PERIOD COVERED: August 1984 - December 1985

6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): Richard E. Nance, James D. Arthur, Ashok V. Dandekar

8. CONTRACT OR GRANT NUMBER(s): N60921-83-G-A165 B003-4

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Systems Research Center and Department of Computer Science, Virginia Tech, Blacksburg, VA 24061

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:

11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Sea Systems Command, SEA61E, Washington, D.C. 20362

12. REPORT DATE: 10 February 1986

13. NUMBER OF PAGES: 162

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Naval Surface Weapons Center, Dahlgren, VA 22448

15. SECURITY CLASS. (of this report): Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Final Distribution to sponsoring office, NSWC and FCDSSA.DamNeck personnel, and other interested Navy agencies.

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

18. SUPPLEMENTARY NOTES

Data related to the evaluation of Navy Methodologies for Embedded Systems Software Development may be obtained through Dr. Harry E. Crisp Mail Code F06, NSWC.

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

Software development, software engineering, methodologies, evaluation procedure, AEGIS, RNTDS, Ada, objectives, principles, attributes, software metrics.

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

A procedure for evaluating software development methodologies is developed based on the linkage among objectives, principles, and attributes. The procedure utilizes metrics defined to recognize surface properties of the software, leading to evidence of the presence or absence of particular attributes. Application of the procedure to samples from the RNTDS and AEGIS Modular Development programs reveals weaknesses and strengths that should be addressed in efforts to produce a common CDS. The extension of the work is proposed to address two major shortcomings of the evaluation

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

S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

procedure: (1) intensive demands on humans in the analysis of code and documentation and (2) a high degree of subjectivity in some property assessments.