(12)

TECHNICAL REPORT TR 78-2-72

BENEFIT MODEL FOR HIGH ORDER LANGUAGE

DECISIONS AND DESIGNS INCORPORATED

Joseph M. Fox

March 1978

ADVANCED DECISION TECHNOLOGY PROGRAM

CYBERNETICS TECHNOLOGY OFFICE
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY
Office of Naval Research • Engineering Psychology Programs

The objective of the Advanced Decision Technology Program is to develop and transfer to users in the Department of Defense advanced management technologies for decision making. These technologies are based upon research in the areas of decision analysis, the behavioral sciences and interactive computer graphics. The program is sponsored by the Cybernetics Technology Office of the Defense Advanced Research Projects Agency and technical progress is monitored by the Office of Naval Research — Engineering Psychology Programs. Participants in the program are:

Decisions and Designs, Incorporated
Harvard University
Perceptronics, Incorporated
Stanford Research Institute
Stanford University
The University of Southern California

Inquiries and comments with regard to this report should be addressed to:

Dr. Martin A. Tolcott
Director, Engineering Psychology Programs
Office of Naval Research
800 North Quincy Street
Arlington, Virginia 22217

or

LT COL Roy M. Gulick, USMC
Cybernetics Technology Office
Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Virginia 22209

TECHNICAL REPORT TR-78-2-72

BENEFIT MODEL FOR HIGH ORDER LANGUAGE

by

Joseph M. Fox

Prepared for

Defense Advanced Research Projects Agency

N00014-76-C-0074, ARPA Order 3052

March 1978

DECISIONS and DESIGNS, INC.

Suite 600, 8400 Westpark Drive
P.O. Box 907
McLean, Virginia 22101
(703) 821-2828

CONTENTS

FIGURES

TABLES

ACKNOWLEDGMENT

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Office of Naval Research under Contract No. N00014-76-C-0074.

BENEFIT MODEL FOR HIGH ORDER LANGUAGE

1.0 INTRODUCTION

In mid-July 1977, DARPA tasked Decisions and Designs, Incorporated (DDI) to perform a two-part effort. The first part was to design a computer program to manipulate decision-analytic models to assist in making the economic benefit decision made for the proposed DoD Common High Order Language. The second task was to implement and run the models.

DDI assigned Joseph M. Fox to manage the project. His background in software management includes 21 years at IBM, with 7 1/2 years as Vice President, Federal Systems Division. In this position, Mr. Fox managed some of the largest software efforts: the Safeguard system, Apollo and Skylab ground control system, the Defense Support Project, the En-Route Air Traffic Control System - both U.S. and U.K., the ground and space control of the Space Shuttle, and dozens of other large projects. Mr. Fox assigned his values to the models at the conclusion of the effort and presented both the models and the economic judgments to the High Order Language (HOL) Management Steering Committee on 20 October 1977.

DDI was able to utilize existing models almost exclusively. A series of three models enables an evaluator to compare languages, that is, to compare scenarios that hypothesize the use of different mixes of languages varied by date of introduction, acceptance rate, and other factors. A DECISION model allows probabilities to be assigned and assessed for various possible future events and decisions in the HOL program.

2.0 STATEMENT OF PROBLEM

Within DoD today over 450 computer programming languages are currently in use for the development or support of application software for "embedded" computers. (An embedded computer is one which is included as an integral part of a larger, special-purpose system. For example, a computer used for general-purpose payroll or inventory applications or even generic scientific applications is not considered embedded; a computer included within a radar system or a gun-control system is considered embedded.) This proliferation of languages (HOLs) has resulted in a host of obvious difficulties in software development projects and in the maintenance of extant software. These difficulties, in turn, have contributed to the steadily increasing costs of DoD computer applications--estimated currently to be about $3.2 billion per year--or at least have hindered attempts to control such costs.

In light of the apparent costs generated by this redundancy of HOLs, DoD has been engaged in an effort to determine the degree to which reducing the number of HOLs used for embedded computer applications would result in cost savings. DoD is considering that a significant reduction in software development costs could be achieved by adopting a single HOL for all future DoD embedded-computer applications. Intuitively, there seems to be overwhelming reason to support this conclusion. However, attempts to support the conclusion with quantitative arguments have foundered, primarily because there is an extreme paucity of actual data. Folklore abounds, but documented cost figures are elusive.

Although the lack of hard data confounds the problem of developing a solidly defendable case, it does not present a hopeless situation. An evaluation scheme which decomposes cost and benefit assessments to detailed defendables at a subjective level can help corroborate the conclusions derived. Under DARPA sponsorship (among others), DDI has developed over the past several years sophisticated decision-analytic methodologies to cope with exactly this type of problem.

3.0 ACCOMPLISHMENTS: OVERVIEW

DDI created three separate models to assess the economic value of a common DoD high order language and/or other high order languages. All models were implemented in APL for the IBM 5100 computer.

3.1 Models

An existing model system, EVAL, was structured to manipulate the values assigned to attributes of languages. One language versus another was scored on fourteen different criteria; the relative importance of each criteria could be set by assigning weights. The model was structured to evaluate six high order languages and to compare them with each other and against a baseline, which is Assembler language. The six languages are generically designated the DoD common language (DoD), Air Force 1 (AF1), Air Force 2 (AF2), Navy 1 and 2 (N1 and N2), and Army 1 (A1).

Some programming effort was expended to restructure SPREAD, a computer routine that takes concise inputs describing situations and generates precise descriptions of scenarios. SPREAD details by year, for twenty years past and future, such items as number of projects per year, number of projects using each of several languages, phase of project (Requirements, Design, Maintenance) by year, and so on.

DECISION, a software modeling system, was configured to reflect the next several decisions and events in the DoD commonality program by using values generated by SPREAD and probabilities estimated and input by the investigator.

3.2 Economic Analysis

Economic analysis was performed by running ten different scerarios through the models. All these scenarios were based on the language evaluations described in section 4.0. Then scenarios were chosen based on different years of DoD1 being added to DoD 5000.31, and different years of cutover (all new starts beyond this year were to be in DoD) were chosen in order to run economic analysis.

Runs were primarily directed toward examining the sensitivity of the introduction date and the rate of acceptance. Detailed results are given in Table 3-1. An introduction date of 1980 and total acceptance in 1985, Column I, may be compared with an introduction in 1985 and total acceptance in 1990, Column II, or with an introduction in 1980 and a slower acceptance, being complete only in 1990, etc. Both the saving rate per year and the total savings to 1996 are shown. It must be noted that even the total acceptance of a language means only that all new programs are being initiated in the language while older programs in other languages continue for their life cycles; therefore, savings take some time to build up. On the other hand, the savings, when established, are enormous; and recapture of language development costs can easily occur in much less than a year, once use is established.

All savings are compared against the baseline which would result from the exclusive use of Assembler Language; therefore, any model considering even the present use of high order languages will exhibit savings. The impact of the common language program is therefore the difference between any proposed scenario with a common language, both savings and costs, less the savings calculated from present trends. Column X in Table 3-1 represents the "no-change" in DoD Directive 5000.31 estimate.

	I	II	III	IV	V	VI	VII	VIII	IX 4 Lang. DoD	X
	1980-85	1985-90	1980-90	1981-86	1983-88	1987-92	1982-83	1982-85	1980-85	No DoD
1977	487	487	487	487	487	487	487	487	487	490
1978	476	476	476	476	476	476	476	476	476	479
1979	483	482	482	482	482	482	482	482	482	486
1980	638	637	637	637	637	637	637	637	637	642
1981	676	654	654	654	654	654	654	654	654	657
1982	707	652	674	674	652	652	652	652	645	653
1983	761	493	612	636	493	493	828	588	557	511
1984	917	516	678	740	533	516	987	690	598	537
1985	1,064	553	776	861	601	553	1,113	811	641	576
1986	1,399	629	976	1,113	656	620	1,558	1,037	786	651
1987	1,561	679	1,097	1,294	787	637	1,783	1,218	839	675
1988	1,775	809	1,242	1,511	1,017	700	2,065	1,500	934	701
1989	1,691	987	1,301	1,570	1,288	749	1,749	1,610	872	620
1990	1,774	1,208	1,461	1,606	1,500	867	1,781	1,726	911	628
1991	2,016	1,554	1,739	1,900	1,827	1,087	2,024	1,969	1,021	656
1992	2,363	1,934	2,100	2,326	2,214	1,386	2,373	2,316	1,180	738
1993	2,511	2,162	2,295	2,472	2,409	1,633	2,520	2,461	1,261	762
1994	2,554	2,313	2,427	[illegible]	2,450	1,854	2,563	2,502	1,309	778
1995	2,385	2,237	2,293	2,341	2,275	1,968	2,396	2,330	1,244	758
1996	2,333	2,278	2,235	2,285	2,217	2,020	2,345	2,273	1,256	763
TOTAL	28,571	21,740	24,642	26,737	23,677	18,471	29,473	26,419	16,790	12,761

YEARLY SAVINGS: 1st Year is Year of Introduction
2nd Year is Year in which all new starts are in DoD; except for CASE IX, where the second year is year in which all new starts are divided among four languages, one of which is DoD.

Table 3-1

HOL MODEL RESULTS

Technical savings expected even from the use of existing languages are quite significant. They are based upon some detailed consideration of the individual languages. (Indeed, if significant savings were not expected, then the present policy [DoDD 5000.29] towards high order languages would be ill-conceived.) The proposed new common high order language is examined in detail and is expected to give significantly higher technical savings. All savings are estimated against a software expenditure of $3.2 billion per year, and all savings are derived proportional to this number.

3.3 Results

An examination of the results based upon the total savings to 1996 gives the following trend:

- For a five-year introduction period, delay of the year of introduction through the period 1980 to 1987, with a corresponding delay of the complete adoption from 1985 to 1992, gives an average reduction in savings of about $1.5 billion per year delayed. This is simply the average magnitude of savings once full use is established.

- Keeping a constant total acceptance date of 1985, delay of the introduction from 1980 to 1982 costs about $1 billion per year delayed.

- Having established an introduction date, slow acceptance or delay of the total usage date costs about $1 billion per year delayed.

- All these figures are proportional to the total software costs envisioned. For example, $1.5 billion corresponds to about 10% of the calculated additional savings due to the use of the new language. Thus, a delay of one year in the introduction could only be justified if it resulted in an offsetting integrated savings of more than 10%, indicating an improved maximum savings rate increase of up to 20%.

It is recommended that the DoD Single Common High Order Language be introduced as rapidly as possible without penalizing technical quality or acceptability much more than 10%. Costs are small in all cases, being less than 1% of savings.

4.0 USER INSTRUCTIONS

All the models are interactive, elicitive, and easy to use. Presented here is a brief overview of the needed inputs, and a printed output of a sample run of each of the models.

4.1 HOL EVAL

EVAL is a program that calculates multi-attribute utilities by using scores and weights. The savings factors for the high order languages are broken into two separate groupings, Technology and Commonality. Two sets of numbers are required for both Commonality and Technology: weight values, which show relative importance, and score values, which show comparison between languages.

Commonality benefits are calculated not from the specific languages adopted, but from the number of HOLs adopted. One language, and only one, would produce more Commonality benefits than a two-language selection.

The scenarios run for this effort were for one language; for three - one each for Army, Navy, and Air Force; for four - Army, Navy, Air Force, and DoD; for five - one for Army, two each for Navy and Air Force (which is the current DoDl 5000.31 menu); and for six - these five plus DoD.

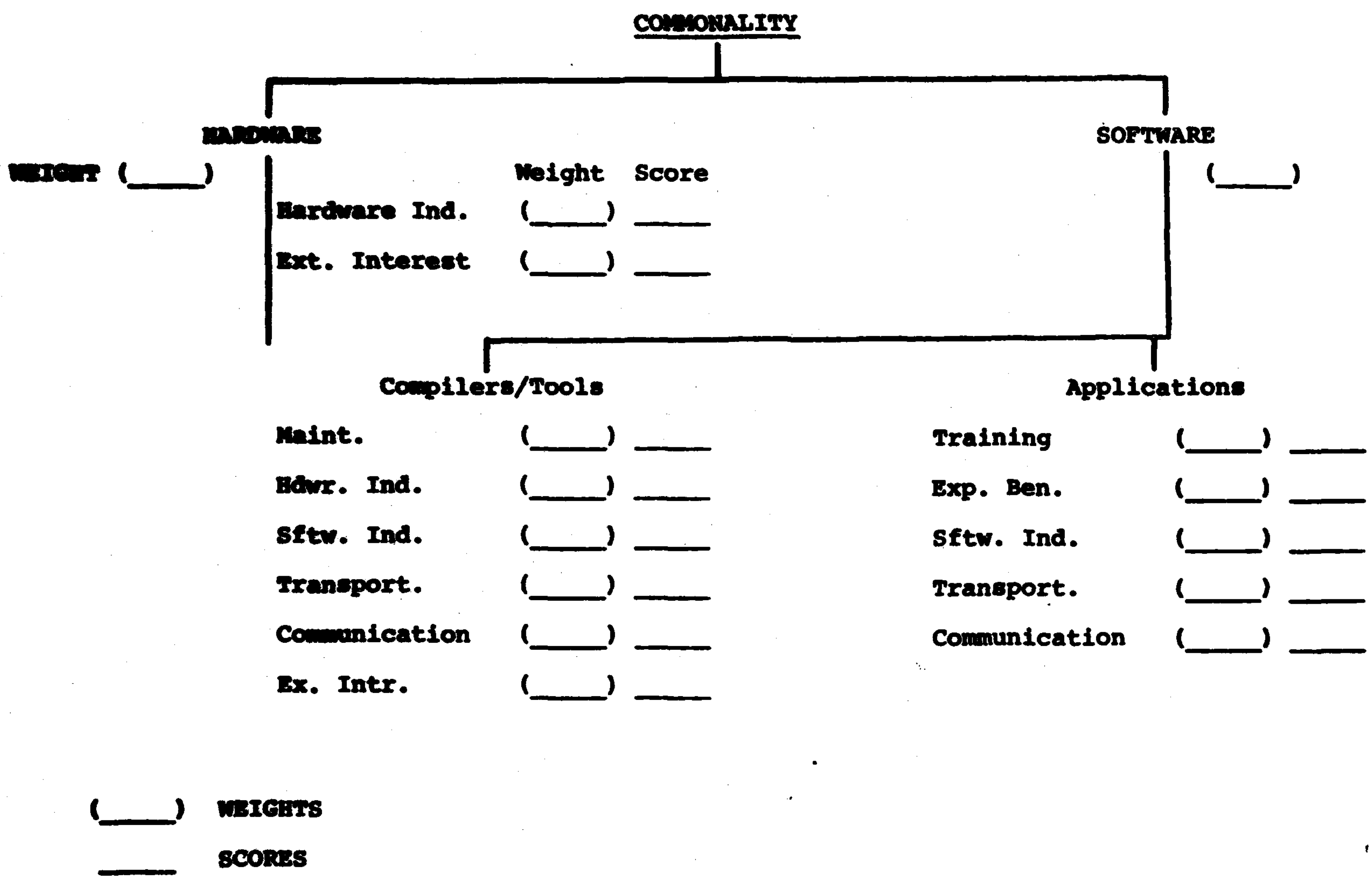
4.1.1 Commonality elements - Commonality elements represent savings that would occur because a language is chosen to be the standard. These elements do not depend on technology improvement in any chosen standard language. For estimating, the easiest way to think of these elements is to assume that an existing high order language is chosen and no improvements are made to it. This does not address the technical question of whether an existing language has sufficient machine independence to be used in the required wide range of applications. Figure 4-1 shows the structure of Commonality.

- Training. If the number of programming languages permitted for use is reduced, the expense of retraining programmers from one language to another will be reduced.

- Experience Benefit (Learning Curve). Each programmer will become more and more familiar with the language, learning the powerful aspects of it, and the pitfalls. By using this knowledge, he should be able to write programs faster and to create more reliable programs. This depth of knowledge is lessened when the programmer changes from language to language.

- Increased Choice: Hardware. With 100% of the code written in a single-source language (presumably supported by all or most vendors), dependence on a single hardware supplier is significantly diminished. The "sunk-cost" trap (i.e., so much money is in programming for this, and only this, machine) will be avoided in most cases. Also, freedom to move from one model of a computer to another will be enhanced.

- Increased Choice: Software. There will be more flexibility in avoiding "sunk-cost" traps (i.e., "our people wrote this and only they can fix it." The resulting freedom will not be as great as with hardware, but it will be significantly more than at present.



One Set of Values Per Language

Figure 4-1
COMMONALITY

- Maintenance. Maintenance should be easier to accomplish as increased use of a language should "shake it down" and "level it out" sooner. More emphasis can be put on maintenance per language or compiler set if there are fewer languages.

- Transportability. There are several items that contribute to this element. Basically, transportability means the use by user 1 of programs written by user 2, 3...n. A smaller number of languages accelerates this cross-use in the following ways:

  - Utilities, Production, Support Systems, Interactive Debugging Systems, Application Subroutines, Data Bases. Since fewer languages are to be used, the problem of unfamiliarity with the language in which routines are coded is avoided.

  - Documentation. Fewer languages will naturally bring increased standardization not only to the language used but to the manner of its documentation.

- Communication between Professional Groups and Individuals. Fewer languages enhances the flow of complex information, and reduces the problem of understanding new innovations and techniques.

- External Interest. Fewer languages being used by tens of thousands of programmers for DoD will attract users outside of DoD; therefore, money, effort and research funded by non-DoD entities will be applicable, improving the usability of the language.

4.1.2 Technology - The savings for Technology depend entirely on the language chosen and on the features of the language. This is shown in Figure 4-2. We model for eight technical criteria:

- Ability to Handle Non-Standard I/O Interfaces. A great deal of programming involves redoing simple I/O routines for devices that are needed for the system being developed but which do not meet any of the established standard interfaces. With a new common language, macro-like I/O routines can be produced which will greatly simplify the production of this I/O code.

- Real-Time Features of the New Language. A great percentage of the embedded computer systems are used in real-time applications. Peculiar stresses are placed on the computer system and its software by these applications. Levels of interrupts and real-time clocks must be handled quickly, whereas in non-real-time systems they are either ignored or minimumly used. The new language will handle the real-time features of the application at the high order language level and will not require the programmer to revert to assembly code.

- Parallel Processing. The growth of use of multiple cpu's in systems is accelerating. Few, if any, of today's HOL languages facilitate the programming of these complex systems. The new HOL will provide tools at the high order level to handle and optimize this parallelism.

- File Processing. The language will provide file processing abilities which will enhance the creation of programs using data files.

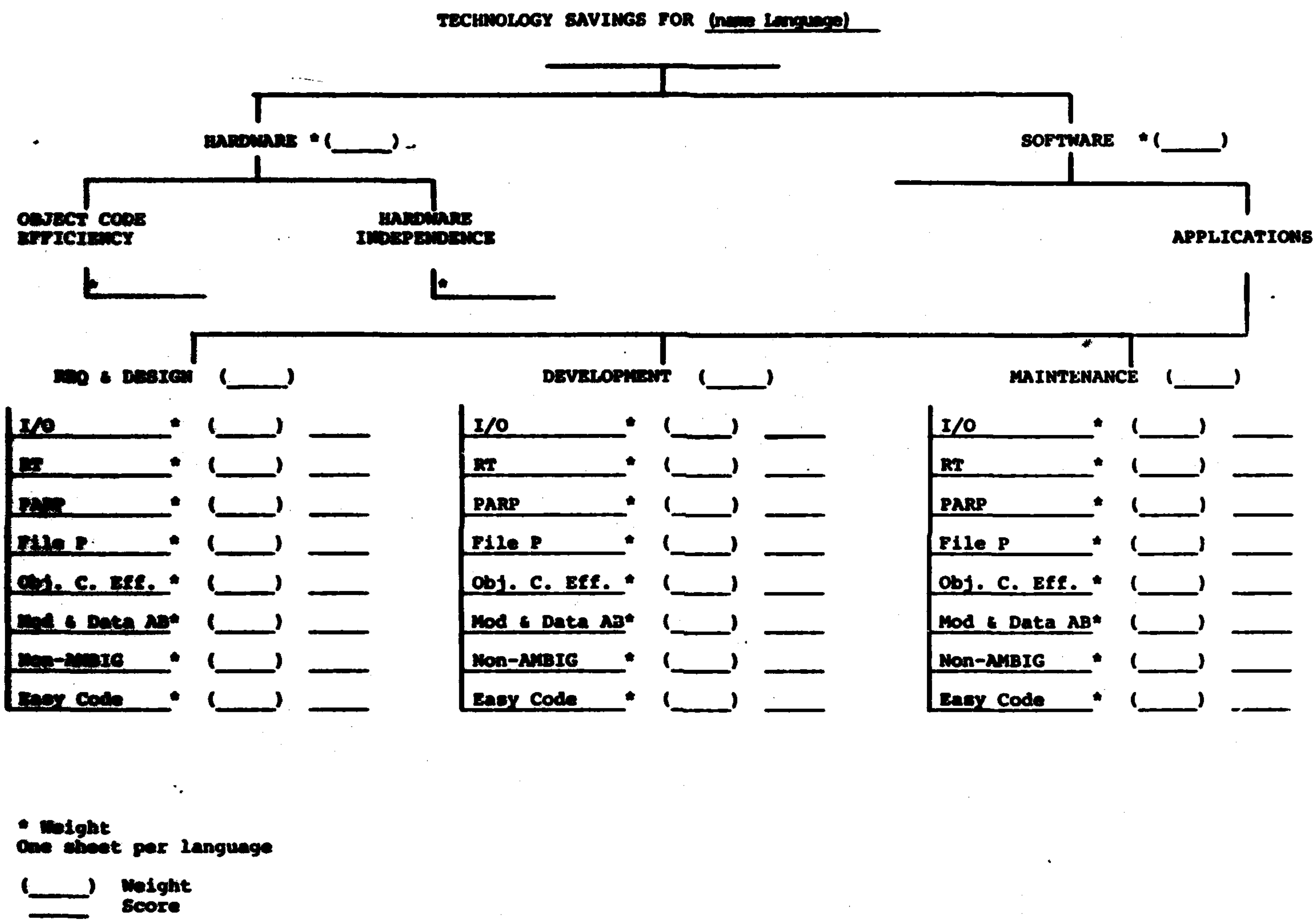


Figure 4-2
TECHNOLOGY SAVINGS

- **Object Code Efficiency**

  - Object code produced by a new technology language will be much more optimum than that produced by the average assembly language programmer.

  - Optimization specificity will be a part of the language. The programmer can tell the compiler to optimize parts of the program in a priority sequence.

- **Modularity and Data Abstraction**. Modularity and Data Abstraction is a facility for making data more easily usable by programmers and reducing ambiguity in data. This greatly reduces the possibility of subtle, long-lived programming bugs.

- **Non-Ambiguity**. Because of the new technical features of the language, reliability and maintainability of the resultant code will be enhanced.

  - **Tightness**. Few, if any, ambiguities will be allowed in the code.

  - **Error Recovery**. Error recovery at object run time will be enhanced.

  - **Readability**. Most programs will be self-documenting and easily readable, making cross-checking of programs among programmers much more feasible.

  - **Correctness**. Proof-of-correctness facilities will be provided.

- <u>Ease of Coding</u>. The new language may require greater discipline, but overall it will be easier to code than the older languages.

The first four attributes (I/O, real time, parallel processing and file processing) are considered to contribute to independent parts of the programming problem. Their advantages are combined additively, weighted with the fraction of the process impacted. This combined effect is melded with the persuasive values (efficiency, modularity, non-ambiguity, and ease of coding) in a residual value model.

The significance of this rule is that it restricts the total savings to a maximum of 100 percent no matter how many factors there are or how great the savings of each factor. An additive rule, on the other hand, would permit savings to exceed 100 percent and to grow without limit. With the residual value rule, the savings contributed by each factor simply reduces whatever remains after the savings contributed by the other factors have been realized.

The residual value node (or Z-node) provided by the EVAL software implements the following combinatorial rule:

$$v = 1.0 - \prod_{i=1}^{n} (1.0 - r_i)$$

where r_i represents the fraction by which the i^{th} factor reduces a value, and v represents the total fraction by which the original value is reduced. Each r_i can be viewed as the savings contributed by the i^{th} factor, and v can be viewed as the total savings.

For example, if there are two factors which contribute to savings, and if factor 1 results in savings of

20% and factor 2 results in further savings of 30%, then the total savings is:

$$1.0-(1.0-.2)(1.0-.3) = 1.0-(.8)(.7) = 1.0-.56$$

or 44 percent (not 50 percent).

4.2 HOL SPREAD

SPREAD calculates the economic benefits of the high order languages.

The HOL SPREAD program takes user inputs and explodes concise inputs into a year-by-year numerical representation of projects using different high order languages. HOL SPREAD generates project numbers by language and by phase (Requirements and Design, Development, Maintenance) from twenty years in the past through twenty years in the future. The numbers for each language depend on the drop-out rates, years of each project's life, and the total number of projects allowed each year (which can be set by the analyst).

HOL SPREAD allows the user to specify a savings rate for Commonality based on the percentage of total projects in a number of languages. Specifically, if 90% of all projects are using one language and the remaining 10% are divided among three other languages, the savings due to Commonality are much greater than if the four languages each represented 25% of the projects. A curve-fitting routine is used.

A distinction is made between phases of a project: requirements and design, development, and maintenance. Their duration of time is a variable; their percentage of continuity--whether any projects stop after each phase--is a variable.

Table 4-1 is an actual computer print-out of the programs. It is reproduced here in order to illustrate the input/output requirements and the interactive nature of the program.

BEST AVAILABLE COPY

```
SELECT ANY SINGLE OPTION BY TYPING A CHARACTER UNDER ITS POINTER

              WHICH CONSTANT DO YOU WISH TO CHANGE?
 ┌──MAX PROJECTS/YEAR
 |     ┌──PERIOD FOR RES/DES
 |     |     ┌──PERIOD FOR DEVELOP.
 |     |     |     ┌──PERIOD FOR MAINT.
 |     |     |     |     ┌──DROP BEFORE DEVELOP.
 |     |     |     |     |     ┌──DROP BEFORE MAINT.
 |     |     |     |     |     |     ┌──NUMBER OF YEARS
_|_   _|_   _|_   _|_   _|_   _|_   _|_
*
MAX PROJECTS/YEAR    100.00 *
```

```
SELECT ANY SINGLE OPTION BY TYPING A CHARACTER UNDER ITS POINTER

              WHICH CONSTANT DO YOU WISH TO CHANGE?
 ┌──MAX PROJECTS/YEAR
 |     ┌──PERIOD FOR RES/DES
 |     |     ┌──PERIOD FOR DEVELOP.
 |     |     |     ┌──PERIOD FOR MAINT.
 |     |     |     |     ┌──DROP BEFORE DEVELOP.
 |     |     |     |     |     ┌──DROP BEFORE MAINT.
 |     |     |     |     |     |     ┌──NUMBER OF YEARS
_|_   _|_   _|_   _|_   _|_   _|_   _|_
       M
PERIOD FOR RES/DES     3.00 *
```

```
SELECT ANY SINGLE OPTION BY TYPING A CHARACTER UNDER ITS POINTER

              WHICH CONSTANT DO YOU WISH TO CHANGE?
 ┌──MAX PROJECTS/YEAR
 |     ┌──PERIOD FOR RES/DES
 |     |     ┌──PERIOD FOR DEVELOP.
 |     |     |     ┌──PERIOD FOR MAINT.
 |     |     |     |     ┌──DROP BEFORE DEVELOP.
 |     |     |     |     |     ┌──DROP BEFORE MAINT.
 |     |     |     |     |     |     ┌──NUMBER OF YEARS
_|_   _|_   _|_   _|_   _|_   _|_   _|_
             M
PERIOD FOR DEVELOP.    3.00 *
```

***Input Variables**

Table 4-1

SPREAD;
EXAMPLES OF INPUT

```
THE LANGUAGE MIX FOR THE CURRENT DIRECTIVE IS:
     DOD    AF1    AF2    N1     N2     A      OTHER
     .00    .27    .20    .20    .20    .07    .07
```

```
THE LANGUAGE MIX FOR THE CURRENT DIRECTIVE IS:
     DOD    AF1    AF2    N1     N2     A      OTHER
     .00    .27    .20    .20    .20    .07    .07
DOD WILL BE INTRODUCED IN  1980
IMPLEMENTATION OF THE PROPOSED LANGUAGE MIX WILL
BE COMPLETED BY  1990
```

```
THE LANGUAGE MIX FOR THE CURRENT DIRECTIVE IS:
     DOD    AF1    AF2    N1     N2     A      OTHER
     .00    .27    .20    .20    .20    .07    .07
DOD WILL BE INTRODUCED IN  1980
IMPLEMENTATION OF THE PROPOSED LANGUAGE MIX WILL
BE COMPLETED BY  1990
THE PROPOSED MIX OF LANGUAGES TO BE IMPLEMENTED IS:
     DOD    AF1    AF2    N1     N2     A      OTHER
     1.00   .00    .00    .00    .00    .00    .00
```

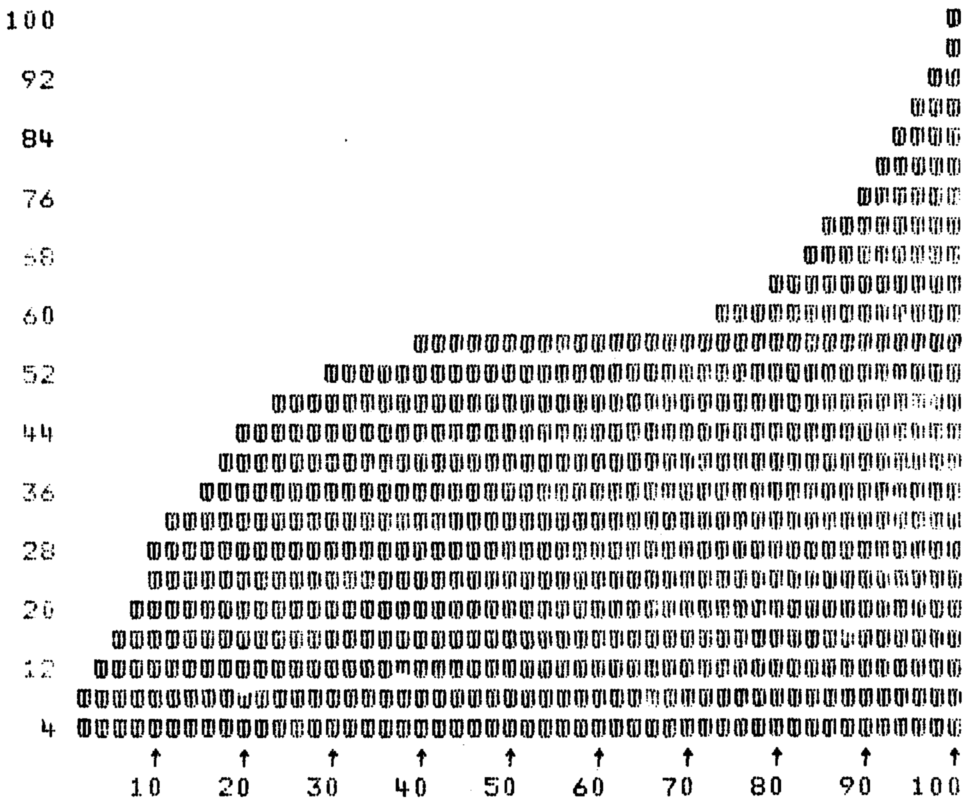
```
          _____SOFTWARE_____ _HDW_
           R/D    DEV    MNT
.20 **     .05    .05    .11    .02
```

```
THE MAXIMUM SOFTWARE SAVINGS FOR COMMONALITY IS     .05
THE MAXIMUM HARDWARE SAVINGS FOR COMMONALITY IS     .04
```

****DoD is one language; values would be called for all languages**

Table 4-1 (Con't)
SPREAD;
EXAMPLES OF INPUT

```
    GRAPH OF COMMONALITY SAVINGS
PERCENT OF PROJECTS IN A SINGLE LANGUAGE (X AXIS)
  VS PERCENT OF TOTAL POSSIBLE SAVINGS (Y AXIS)
```




```
THE MAXIMUM SOFTWARE SAVINGS FOR COMMONALITY IS   .05
THE MAXIMUM HARDWARE SAVINGS FOR COMMONALITY IS   .04


ENTER THE BOUNDARY PTS OF THE CENTER PORTION OF THE CURVE.
LEFT PT :  25 50
RIGHT PT.  80 60
```

Table 4-1 (Con't)

SPREAD:
EXAMPLES OF INPUT

	1977	1978	1979	1980	1981	1982	1983	1984
DIST OF ANN AVG COST	5	5	7	9	5	5	5	5
	1985	1986	1987	1988	1989	1990	1991	1992
DIST OF ANN AVG COST	5	5	5	5	5	5	5	5
	1993	1994	1995					
DIST OF ANN AVG COST	5	5	5					

COSTS:		
HARDWARD COSTS:		4000.00
SOFTWARE COSTS:		
AVERAGE ANNUAL COST:		3200.00
AVERAGE NUMBER OF PROJECTS:		100
RATIO OF COST DISTRIBUTION:		
	RES/DES:	2
	DEV:	4
	MAINT:	1
AVERAGE TOTAL COST PER PROJECT:		416.00
DISTRIBUTION OF COST BY PHASE:		
	RES/DES:	99.84
	DEV:	199.68
	MAINT:	116.48
AVG. ANNUAL COST BY PER PROJ. :		
	RES/DES:	33.28
	DEV:	66.56
	MAINT:	16.64

Table 4-1 (Con't)

SPREAD:
EXAMPLES OF INPUT

MAX PROJECTS/YEAR	100.00
PERIOD FOR RES/DES	3.00
PERIOD FOR DEVELOP.	3.00
PERIOD FOR MAINT.	14.00
DROP BEFORE DEVELOP.	.00
DROP BEFORE MAINT.	.50
NUMBER OF YEARS	20.00

	RES/DES	DEV	MAINT.	HDW.
DOD	.05	.05	.11	.02
AF1	.01	.01	.02	.01
AF2	.01	.01	.02	.01
N1	.01	.01	.02	.01
N2	.01	.01	.02	.01
A	.00	.01	.03	.01
OTHER	.00	.00	.00	.00

THE LANGUAGE MIX FOR THE CURRENT DIRECTIVE IS:

DOD	AF1	AF2	N1	N2	A	OTHER
.00	.27	.20	.20	.20	.07	.07

DOD WILL BE INTRODUCED IN 1980
IMPLEMENTATION OF THE PROPOSED LANGUAGE MIX WILL
BE COMPLETED BY 1990
THE PROPOSED MIX OF LANGUAGES TO BE IMPLEMENTED IS:

DOD	AF1	AF2	N1	N2	A	OTHER
1.00	.00	.00	.00	.00	.00	.00

Table 4-1 (Con't)

SPREAD;
EXAMPLES OF INPUTS

In addition, there is a ratio of expenditure rate required as input. The ratio of expenditure rate is usually equivalent to manning, but occasionally computer resources and/or test facilities may be of such magnitude that they should be considered. An example explains this: If we believe a typical project per year spends three times as much manpower and resources in development as in maintenance, and twice as much in requirements and design as in maintenance, then our effort ratio is:

R/D	D	M
2:	3:	1

Said another way, if we have fourteen men on Requirements and Design, then we will have twenty-one on Development and seven on Maintenance. This ratio is independent of years and is therefore a rate. It is used as follows:

1. Multiply Expenditure Rate Ratio by Phase
   x No. of Years by Phase
   = Units of Expenditure by Phase

2. Add units to get Total Units of Expenditure

3. Calculate % of Expenditure by Phase

4. Use % from 3 above x Total Average Project Cost to get $ Expended by Phase

5. ÷ $ (from line 4) by Years in Phase to get $ Expended Per Year Per Phase

Since the Technology savings (in EVAL) are estimated by language by phase (Requirements and Design, etc.), these savings percentages by phase must be multiplied by values calculated for the average project. The SPREAD program does all of this automatically.

This complicated logic allows for a variable number of projects per year, variable total cost, variable average project cost, variable duration of projects, variable duration of project phase and variable ratios of phase to phase expenditure rate.

SPREAD takes the EVAL (or directly estimated) values for software and hardware savings and the analyst's inputs described above and prints out:

- number of projects by language, by phase, by year;
- total software expenditures, total hardware expenditures;
- software savings from software used by phase;
- hardware savings from software used by phase.

The savings shown in Figure 3-1 resulted from running the SPREAD program for ten different time periods using the inputs in Figure 4-3 (from J. M. Fox from EVAL).

4.3 HOL DECISION

The HOL DECISION program is aimed at assisting the analyst to estimate the best future program plan for DoD/HOL options.

HOL DECISION uses the savings figures calculated by HOL SPREAD as input, and the probability estimates of the analyst for future events. The values and percents are then used in calculating the best program plan.

	RES/DES	DEV	MAINT	HDW
DOD	.75	.75	.61	.50
AF1	.10	.15	.12	.11
AF2	.10	.20	.10	.10
N1	.10	.11	.12	.11
N2	.11	.11	.12	.11
A	.10	.11	.13	.15
OTHER	.00	.00	.00	.00

Figure 4-3
% SAVINGS OVER ASSEMBLY LANGUAGE

The program allows for five uncertainties (probabilities are assigned to each) and four decisions. The decisions are in chronological sequence: number of HOLs to write; test and evaluate in series or in parallel; accelerate DoD1 usage; eliminate or allow other languages.

The outcomes or uncertainties are, in chronological order: program fail; difficulties in parallel test and evaluation; program fail; user acceptance of DoD1.

To run HOL DECISION, the user must enter: 1) probabilities, and 2) a value at each endpoint. The flow of the DECISION program can be seen from the node numbering sequence. This is shown in Table 4-2.

1.1. -HI CORREL -FOUR HOLS

1) N ERLY BMB ____
2) EARLY BOMB ____

1.1.1.1. -HI CORREL -FOUR HOLS -N ERLY BMB-PARALLEL

1) GLITCH ____
2) NO GLITCH ____

1.1.1.1.1. -FOUR HOLS -N ERLY BMB-PARALLEL -GLITCH

1) NO LT BMB ____
2) LATE BOMB ____

1.1.1.1.1.1.1. -PARALLEL -GLITCH -NO LT BMB -PUSH DOD1

1) SER RESIST ____
2) NO SER RES ____

1.1.1.1.2. -FOUR HOLS -N ERLY BMB-PARALLEL -NO GLITCH

1) NO LT BMB ____
2) LATE BOMB ____

1.1.1.1.2.1.1. -PARALLEL -NO GLITCH -NO LT BMB -PUSH DOD1

1) SER RESIST ____
2) NO SER RES ____

1.1.1.2. -HI CORREL -FOUR HOLS -N ERLY BMB-SERIES

1) NO LT BMB ____
2) LATE BOMB ____

1.1.1.2.1.1. -N ERLY BMB-SERIES -NO LT BMB -PUSH DOD1

1) SER RESIST ____
2) NO SER RES ____

Table 4-2

DECISION:
PROBABILITIES

BEST AVAILABLE COPY

1.2. -HI CORREL -TWO HOLS
1) N ERLY BMB ____
2) EARLY BOMB ____

1.2.1.1. -HI CORREL -TWO HOLS -N ERLY BMB-PARALLEL
1) GLITCH ____
2) NO GLITCH ____

1.2.1.1.1. -TWO HOLS -N ERLY BMB-PARALLEL -GLITCH
1) NO LT BMB ____
2) LATE BOMB ____

ENTER PROBABILITIES

1.2.1.1.1.1.1. -PARALLEL -GLITCH -NO LT BMB -PUSH DOD1
1) SER RESIST ____
2) NO SER RES ____

1.2.1.1.2. -TWO HOLS -N ERLY BMB-PARALLEL -NO GLITCH
1) NO LT BMB ____
2) LATE BOMB ____

1.2.1.1.2.1.1. -PARALLEL -NO GLITCH -NO LT BMB -PUSH DOD1
1) SER RESIST ____
2) NO SER RES ____

1.2.1.2. -HI CORREL -TWO HOLS -N ERLY BMB-SERIES
1) NO LT BMB ____
2) LATE BOMB ____

1.2.1.2.1.1. -N ERLY BMB-SERIES -NO LT BMB -PUSH DOD1
1) SER RESIST ____
2) NO SER RES ____

Table 4-2 (Con't)

DECISION:
PROBABILITIES

1.3. -HI CORREL -ONE HOL
1) N ERLY BMB ____
2) EARLY BOMB ____

1.3.1.1. -HI CORREL -ONE HOL -N ERLY BMB-PARALLEL
1) GLITCH ____
2) NO GLITCH ____

1.3.1.1.1. -ONE HOL -N ERLY BMB-PARALLEL -GLITCH
1) NO LT BMB ____
2) LATE BOMB ____

1.3.1.1.1.1.1. -PARALLEL -GLITCH -NO LT BMB -PUSH DOD1
1) SER RESIST ____
2) NO SER RES ____

1.3.1.1.2. -ONE HOL -N ERLY BMB-PARALLEL -NO GLITCH
1) NO LT BMB ____
2) LATE BOMB ____

1.3.1.1.2.1.1. -PARALLEL -NO GLITCH -NO LT BMB -PUSH DOD1
1) SER RESIST ____
2) NO SER RES ____

1.3.1.2. -HI CORREL -ONE HOL -N ERLY BMB-SERIES
1) NO LT BMB ____
2) LATE BOMB ____

1.3.1.2.1.1. -N ERLY BMB-SERIES -NO LT BMB -PUSH DOD1
1) SER RESIST ____
2) NO SER RES ____

Table 4-2 (Con't)

DECISION:
PROBABILITIES

In Table 4-3 dollar savings values are to be entered into NETVAL, and probability estimates are to be entered into the series of numbered nodes.

NETVAL


```
1.  HI CORREL
1.1.  FOUR HOLS
1.1.1.  N ERLY BMB
1.1.1.1.  PARALLEL
1.1.1.1.1.  GLITCH
1.1.1.1.1.1.  NO LT BMB
1.1.1.1.1.1.1.  PUSH DOD1
1.1.1.1.1.1.1.1.  SER RESIST      ----
1.1.1.1.1.1.1.2.  NO SER RES      ----
1.1.1.1.1.1.2.  DON'T PUSH        ----
1.1.1.1.1.2.  LATE BOMB           ----
1.1.1.1.2.  NO GLITCH
1.1.1.1.2.1.  NO LT BMB
1.1.1.1.2.1.1.  PUSH DOD1
1.1.1.1.2.1.1.1.  SER RESIST      ----
1.1.1.1.2.1.1.2.  NO SER RES      ----
1.1.1.1.2.1.2.  DON'T PUSH        ----
1.1.1.1.2.2.  LATE BOMB           ----
1.1.1.2.  SERIES
1.1.1.2.1.  NO LT BMB
1.1.1.2.1.1.  PUSH DOD1
1.1.1.2.1.1.1.  SER RESIST        ----
1.1.1.2.1.1.2.  NO SER RES        ----
1.1.1.2.1.2.  DON'T PUSH          ----
1.1.1.2.2.  LATE BOMB             ----
1.1.2.  EARLY BOMB                ----
1.2.  TWO HOLS
1.2.1.  N ERLY BMB
1.2.1.1.  PARALLEL
1.2.1.1.1.  GLITCH
1.2.1.1.1.1.  NO LT BMB
1.2.1.1.1.1.1.  PUSH DOD1
1.2.1.1.1.1.1.1.  SER RESIST      ----
1.2.1.1.1.1.1.2.  NO SER RES      ----
1.2.1.1.1.1.2.  DON'T PUSH        ----
1.2.1.1.1.2.  LATE BOMB           ----
1.2.1.1.2.  NO GLITCH
1.2.1.1.2.1.  NO LT BMB
1.2.1.1.2.1.1.  PUSH DOD1
1.2.1.1.2.1.1.1.  SER RESIST      ----
1.2.1.1.2.1.1.2.  NO SER RES      ----
1.2.1.1.2.1.2.  DON'T PUSH        ----
1.2.1.1.2.2.  LATE BOMB           ----
1.2.1.2.  SERIES
1.2.1.2.1.  NO LT BMB
1.2.1.2.1.1.  PUSH DOD1
1.2.1.2.1.1.1.  SER RESIST        ----
1.2.1.2.1.1.2.  NO SER RES        ----
1.2.1.2.1.2.  DON'T PUSH          ----
1.2.1.2.2.  LATE BOMB             ----
1.2.2.  EARLY BOMB                ----
```

Table 4-3

NETVAL

BEST AVAILABLE COPY

```
1.3.  ONE HOL
1.3.1.  N ERLY BMB
1.3.1.1.  PARALLEL
1.3.1.1.1.  GLITCH
1.3.1.1.1.1.  NO LT BMB
1.3.1.1.1.1.1.  PUSH DOD1
1.3.1.1.1.1.1.1.  SER RESIST      ____
1.3.1.1.1.1.1.2.  NO SER RES      ____
1.3.1.1.1.1.2.  DON'T PUSH        ____
1.3.1.1.1.2.  LATE BOMB           ____
1.3.1.1.2.  NO GLITCH
1.3.1.1.2.1.  NO LT BMB
1.3.1.1.2.1.1.  PUSH DOD1
1.3.1.1.2.1.1.1.  SER RESIST      ____
1.3.1.1.2.1.1.2.  NO SER RES      ____
1.3.1.1.2.1.2.  DON'T PUSH        ____
1.3.1.1.2.2.  LATE BOMB           ____
1.3.1.2.  SERIES
1.3.1.2.1.  NO LT BMB
1.3.1.2.1.1.  PUSH DOD1
1.3.1.2.1.1.1.  SER RESIST        ____
1.3.1.2.1.1.2.  NO SER RES        ____
1.3.1.2.1.2.  DON'T PUSH          ____
1.3.1.2.2.  LATE BOMB             ____
1.3.2.  EARLY BOMB                ____
1.4.  STOP PROJ                   ____
```

Table 4-3 (Con't)

NETVAL

CONTRACT DISTRIBUTION LIST
(Unclassified Technical Reports)

Director Advanced Research Projects Agency Attention: Program Management Office 1400 Wilson Boulevard Arlington, Virginia 22209	2 copies
Office of Naval Research Attention: Code 455 800 North Quincy Street Arlington, Virginia 22217	3 copies
Defense Documentation Center Attention: DDC-TC Cameron Station Alexandria, Virginia 22314	12 copies
DCASMA Baltimore Office Attention: Mr. K. Gerasim 300 East Joppa Road Towson, Maryland 21204	1 copy
Director Naval Research Laboratory Attention: Code 2627 Washington, D.C. 20375	6 copies
Office of Naval Research Attention: Code 102IP 800 North Quincy Street Arlington, Virginia 22217	6 copies

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: TR 78-2-72	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): BENEFIT MODEL FOR HIGH ORDER LANGUAGE	5. TYPE OF REPORT & PERIOD COVERED: Technical
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Joseph M. Fox	8. CONTRACT OR GRANT NUMBER(s): N00014-76-C-0074
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Decisions and Designs, Inc. Suite 600, 8400 Westpark Drive, P.O. Box 907 McLean, VA 22101	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22217	12. REPORT DATE: March 1978
	13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Office of Naval Research 800 North Quincy Street Arlington, VA 22217	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

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

18. SUPPLEMENTARY NOTES

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

High Order Languages
Savings from High Order Languages
Savings from Commonality
Embedded Computers
Software Costs
Models
Technical Savings

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

The purpose of this project was to assess the economic value of DoD standardizing on one software high order language (HOL). Three decision-analytic models (i.e. EVAL, SPREAD, and DECISION) were created to make the economic benefit decision for the proposed DoD Common HOL. All models were implemented in APL for the IBM 5100 computer.

These models allow a user to start from specific savings values for various language features and derive a year-by-year savings in software expenditures. Implementation of each model is demonstrated.

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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)