NASA
Automatic System for Computer Program Documentation

VOLUME 1
FINAL REPORT

TEXAS A&M UNIVERSITY
DECEMBER 1972
AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Dick B. Simmons, Roger W. Elliott, Susan Arseven, and Daniel Colunga

Final Report
Contract NAS5-11911

December 1972

Submitted to
National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, Maryland 20771

Submitted by
Texas Engineering Experiment Station
Texas A&M University
College Station, Texas 77843
FOREWORD

This report presents the results of the project to design an automatic system for computer program documentation. This work was performed by the Texas Engineering Experiment Station at Texas A&M University, College Station, Texas. This work was performed under Contract NAS5-11911 for the National Aeronautics and Space Administration, Goddard Space Flight Center, Greenbelt, Maryland. The project monitor was Mr. E. P. Damon.

The authors wish to express their appreciation for the assistance provided by Randy Birge, Hank Goggan, Melvin McKinney, George Nichols, Ralph F. Planthold, Charles Schroeder, Andrew Sobey, Jr., Ernest R. Story, James R. Thames, and Darrell Ward.

This report was submitted on December 15, 1972.
ABSTRACT

This report describes the work done on a project to design an automatic system for computer program documentation. An extensive survey of documentation aids was made to determine what existing programs could be used effectively to document computer programs. Results of the study are included in the form of an extensive bibliography and working papers on appropriate operating systems, text editors, program editors, data structures, standards, decision tables, flowchart systems, and proprietary documentation aids. The preliminary design for an automated documentation system is included. An actual program has been documented in detail to demonstrate the types of output that can be produced by the proposed system.
# TABLE OF CONTENTS

**VOLUME I**

**FINAL REPORT**

- Foreword .................................................. i
- Abstract .................................................... ii
- 1.0 Introduction ......................................... 1
- 2.0 Documentation Aids - Study and Evaluation .......... 3
- 3.0 Automatic System for Computer Program Documentation . 7
  - 3.1 System Features ...................................... 8
  - 3.2 Initial Phase ........................................ 9
  - 3.3 System Structure ..................................... 10
  - 3.4 Advantages ........................................... 10
- 4.0 Sample Program Documentation ........................ 11
- 5.0 Deliverable Items ..................................... 14

- Appendix A. Bibliography ................................. A-1
- Appendix B. Existing Documentation Aids ................ B-1
- Appendix C. Comparison of Operating Systems, IBM S/360 OS,
  Univac Exec 8, CDC Scope3 ............................... C-1
- Appendix D. Audio Documentation Experiment ............ D-1
- Appendix E. Text Editors .................................. E-1
- Appendix F. Program Editors ............................... F-1
- Appendix G. Data Description in FORTRAN, COBOL, and PL/1 . G-1
- Appendix H. Documentation Standards ..................... H-1
Appendix I. Automatic System for Computer Program Documentation,  
A Proposal for Initial Implementation. I-1

VOLUME II

Appendix J. Sample Documented Program

Maintenance Manual. J-3.1
1.0 INTRODUCTION

Program documentation consists of all those items produced to help someone understand a program but which are not actually needed to produce the program. A program listing, when properly prepared, contains much descriptive information in addition to what is necessary to create the program. Documents such as written program descriptions, two dimensional data layouts, flowcharts, and cross references are all part of program documentation.

There is ample evidence that indicates the programmers, when properly motivated, are more than willing to provide needed program documentation. The problem of documentation is less one of ability to document than it is to define realistic documentation requirements. And, it is a general misconception that most programmers prefer to do things in their own way instead of in a standard way. Programmers are quite willing to follow a prescription for documentation as long as the policy is sound. A realistic management policy makes the difference between good and poor documentation.

To be generally useful, an automatic system for computer program documentation should document programs written in compiler or assembler languages. It should be independent of any manufacturer's computer or peripheral equipment. It should be applicable to a wide variety of types and sizes of programs. And, it should be capable of providing complete documentation for large, complex, high-use programs and more elementary documentation for smaller low-usage programs.

Such a system will be described in the following section. In general, documentation will be derived from information programmers normally supply when
designing, programming, and debugging a program. The system is both economical to operate and easy to use. The automated system described in this report allows a manager to specify the type and degree of documentation he would like produced on a project. The system aids him in controlling programming projects during development. Documentation produced by the system has been designed with flexibility to meet the needs of various types of programming projects.

The development of the automated documentation system design was accomplished over a period of nine months. During the first part of the study existing documentation aids were surveyed and their effectiveness was evaluated. An extensive bibliography of documentation aids was developed which is included in Appendix A of this report. To aid in evaluating documentation aids, working reports in areas such as program editors, text editors, operating systems, proprietary systems, flowcharting systems, and decision tables were produced. Existing documentation standards were examined. Audio-documentation techniques were explored as a possible alternative to written documentation.

Upon completion of the study and evaluation phase, an automatic documentation system was designed. One major objective of the system was to include existing documentation aids whenever possible. The system has been designed to make maximum use of existing software and to minimize the amount of additional software that needs to be developed in order to have an efficient, easy-to-operate, user-oriented automated documentation system. The system produces documentation during the development of a project as well as final documentation needed to understand and maintain the programming system.

In the following section the documentation aids study and evaluation phase will be described in detail. Then an overview of the system design will be
given after which a sample output from the proposed system will be described. A section describing the deliverable items described in the original contract is included after the sample program.

2.0 DOCUMENTATION AIDS - STUDY AND EVALUATION

An extensive study of existing documentation aids was made. Computer abstracts, relevant computer literature, and government information services were used to locate pertinent documents. A key word in context (KWIC) program was used to produce the bibliography contained in Appendix A. The bibliography cites 149 documents. In the first section of the bibliography, documents are listed by author. Each entry contains the author's name, the title of the document and the document number. In the next section, the document titles are listed according to key word with the document number to reference the full entry in the next section. In the third section of the bibliography, each document entered in the bibliography is listed by document number. The complete entry contains the name of the author, title, source of the reference and a brief statement about the contents of the reference. The last section of the bibliography is the KWIC index of document titles.

In addition to surveying existing literature and government data bases, a number of documentation aids which are supplied commercially were examined. A working paper was written describing AUTOCHART, AUTODIAGRAMMER, AUTODOC, AUTOFLOW, COMCHART, DYNACHART, EASYFLOW, FACTS, FLOWGEN/F, FORFLO, QUICKDRAW, SUPEREF, and FORDOC. All of the existing documentation aids had obvious failings. Many were restricted to either a single language or a single machine. None produced all of the documentation required to document a computer program. Most were able to document only at compilation level in contrast to a load module level
or a total system level. Many were limited to a single output device. Existing documentation aids are as difficult to use as a major compiler. If the programmer were to make use of more than one proprietary documentation aid he would have to make a major effort to learn the detailed rules of each package. Even though proprietary systems are quite expensive, they do not meet the total documentation needs of a major computer user in that most do not even address themselves to the problems of dealing with the textual descriptions, data layouts, etc., that must be included in every set of documentation.

While NASA has computers from almost every computer manufacturer, a major part of program development is done on IBM, CDC, and Univac systems. A working paper contained in Appendix C describes options available to the user through compilers, assemblers, and linkage editor/loaders. The operating systems evaluated were IBM's OS/360, Univac's EXEC 8, and CDC's SCOPE3. The languages chosen for study to determine options available were PL/1, FORTRAN, COBOL, and the Assembly language appropriate for the specific machine.

In the design of the automated documentation system, a decision was made not to modify any compiler or operating system. All information required for documentation would be either obtained from output normally produced by the appropriate software or from a special program produced to recreate the information. While the operating systems developed for the different machines are not the same, many options are very similar. An automatic documentation system that is designed to be used on any of the machines should take advantage of the common options of the operating systems. Tables within the working paper on operating systems describe the options of the systems and show which options are similar for the various operating systems.
Audio techniques were examined to determine their effectiveness as a tool for documentation. An experiment described in Appendix D was conducted in which programs were documented using both written and audio techniques. No significant advantage to either mode of documentation was noted. Probably the most useful place for audio recording techniques would be in the capturing of information which is later transcribed to written text. Many programmers who do not like to write descriptions of programming systems are perfectly willing to dictate them into a recorder. The recorder description can then be transcribed and edited into high quality written descriptions of programs.

Since much of the final documentation of programming systems is written text, text editors must be considered a major documentation aid. Many text editors such as the interactive ATS text editor on the IBM 360/370 or the batch TEXT360 available on the same systems required many man years to develop. Systems such as text editors should be used as subsystems of any automated documentation system developed so that it is not necessary to reproduce the features already built into existing text editors. Appendix E is a working paper describing text editors. Included are APL Text Editor, MTST, ASTROCOMP, DATATEXT, EDIT, ED PROCESSOR, ATS, TEXT/360, HES, FRESS, REDIT/RUNOFF, NLS, (TNLS, DNLS) and TEXT. Included in the Appendix are descriptions of the various systems and a table comparing them. Some of these text editors would be excellent subsystems of an automated documentation system.

Program editors were also examined during the study and evaluation phase. In Appendix F the CMS, WYLBUR, QED, TECO, TVEDIT, EDIT, and ED on-line program editors were compared. Also the LIBRARIAN, SIMPLE, CLOT, PROGRAM/MANAGE, CFMS, PANVALET, PLUS D-A, SPLIS-II, IEBUPDAT, IEBUPDTE, and ED off-line program
editors were described and compared. Program editors are not as complex as text editors, and most of the features of the program editors are included as subsets of the features of text editors. The program editor function would be developed as a part of the software necessary to control an automated documentation system that encompasses a wide spectrum of documentation aids.

A report was written by George C. Nichols describing a number of existing flowcharting systems.* A number of articles have been published on the various flowcharting systems. The most comprehensive comparison of flowcharting systems was written by Ned Chapin. ** N. Chapin mentions in his book that of the flowcharting systems he has evaluated, none could draw the standard flowchart symbols recommended by the American Standard Association. A system developed by D. B. Simmons at Bell Telephone Laboratories is operational and can draw any flowchart symbol on any type of output device to document any language. A number of flowcharting systems exist which could be driven by an automatic documentation system.

Decision tables are sometimes used as a type of documentation. Sometimes the actual source program is automatically produced from a type of decision table. An evaluation of existing decision table systems was done by J. R. Thames.*** In most cases, decision tables are used to produce a program instead of vice versa. It is felt that for most programming languages, it is not appropriate to automatically produce decision tables from source codes.

* Automatic documentation: A State of The Art Report by George C. Nichols written in partial fulfillment of a Master of Science degree in Computer Science

** Chapin, N., Flowcharts, Auerbach Publishers, Princeton, 1971

*** J. R. Thames, Documentation: Justification and Decision Table Exemplification, Texas A&M University, Spring, 1972 prepared in partial fulfillment of requirements for the degree of Master of Computing Science.
In most programming languages, the data description is either implicitly or explicitly described in a linear manner. When someone works with a complex data structure, special documentation techniques such as a two dimensional description of a data structure is useful. At the present time, very little has been done to automatically document data structures of programs. A short report was written and is included in Appendix G describing characteristics of the data structures of the PL/1, FORTRAN, and COBOL languages. Additional work needs to be done in the area of automatically describing data structures.

A very important facet of program documentation involves the standards that are used. There is no set of standards for all documentation that is produced describing programs. Standards exist for flowcharts but in other areas there are no generally accepted standards, and local practices usually take their place. A number of local documentation standards from both government and industry were examined and a short report on documentation standards is included in Appendix H.

As a result of the study evaluation phase, it was determined that many of the existing software systems such as text editors, flowcharters, language processors, operating systems, and data description software can be used as modules of a total documentation system. A flexible user-oriented documentation system is described in the following section.

3.0 AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

An automatic documentation system has been designed to produce timely up-to-date documentation at a relatively low cost. The system will document any computer language and run on any hardware taking advantage of existing
documentation aids. The system will be easy to use and will place no restrictions on the programmer. A detailed proposal for implementing the automatic documentation system is included in Appendix I. A brief summary of the system features is given in the next section.

3.1 System Features

The automatic documentation system will have the following features:

1. **Minimal programmer restrictions** - The system will be able to produce detailed, detail-suppressed and global flowcharts, data layouts, overlay descriptions, special cross reference glossaries, etc., from any computer program. Programmers who write comments in a standard way and who write their program descriptions using an interactive interrogator will be able to produce all documentation automatically. The system will be designed so that if a program is developed outside the system it will be fairly easy to retrofit the program into the system for documentation maintenance.

2. **Eliminate all redundant efforts** - Documentation produced in an early phase of the design or implementation process can be reused later on in the development process.

3. **No operating system modification** - No modifications will be made to any operating system. Items of documentation interest will be obtained by scanning output produced by operating systems or by reproducing information found in the internal tables of the operating system.
4. **Use of existing documentation aids** - Existing documentation aids will be used as modules in the comprehensive automated system.

5. **Interactive/batch system** - An interactive mode with special features for interrogating the user of the system will be used to obtain necessary documentation aids. An alternative batch mode will be available.

6. **Documentation during development** - The documentation data base will be constructed from information gathered during all design phases. As the design progresses, the user supplies only the information not available from previous steps.

7. **Accept any language** - The automated documentation system will be language independent. Initially the system will be designed to accept FORTRAN, COBOL, PL/1, and Assembly Language.

8. **Operate on any hardware** - Initially the system will be designed to operate on the IBM 360 with planned expansion to the Univac 1108 and the CDC 6600 computer systems.

9. **Monitor and control projects** - Features will be designed into the system to allow the project manager to monitor the exact status of program development and program documentation. In addition, system access and security will be under his control.

3.2 **Initial Phase**

It is proposed that in January, 1973, initial development of the automatic system for computer program documentation will begin. A twelve month period beginning at that time will be termed the initial phase. During the initial phase either FORTRAN or COBOL will be used to write programs that make up the automatic
documentation system. Initial development will be for a system operational on an IBM 360 computer, but software will be easily transportable to the Univac or CDC system. The automatic documentation system will document COBOL, FORTRAN, PL/1, or assembler language programs.

A detailed description of how the automated documentation system can be used for documentation during development as well as for documentation when development is complete is described in the proposal contained in Appendix I.

3.3 System Structure

Three types of programs will be used in the system. Types 1 and 2 are new programs that must be developed. Type 3 programs are existing programs that can be used without change. Type 3 programs make up a major part of the software necessary to implement the automatic documentation system. New programs will not be developed when operational documentation aids are available. By doing this, a sophisticated system will be developed at a relatively low cost. A detailed description of the Type 1, Type 2, and Type 3 programs is contained in Appendix I. The detailed descriptions of the programs to be developed and the types of documentation produced by the system are described in Appendix I.

3.4 Advantages

Use of the proposed automatic documentation system offers many advantages over other techniques for developing programs and producing documentation. The system will be user-oriented and will be as easy to operate as existing on-line or batch program editors. Programmers who use good programming practices in developing software can use all of the system features without
extra effort. Programmer productivity will be enhanced by improved communications during the development process. A modular system will allow new types of documentation to be easily added to the system. This is the first system to bring together all types of documentation aids into a single system. It emphasizes documentation on a load module and system basis as well as for a single compilation. Documents can be produced containing heterogeneous outputs such as text and flowcharts. Managers can use the system to control and monitor projects. Program and documentation standards can be enforced and taught through the use of automatic documentation systems. People who do not use the documentation system during program development will be able to use it for post-development documentation. Programs developed independently of the automatic documentation system can easily be retrofitted into the documentation data base.

4.0 SAMPLE PROGRAM DOCUMENTATION

A sample of the types of documentation to be produced by the automatic system is included in Appendix J. Since the types and formats of the documentation prepared is completely determined by the user when he specifies templates for data to be collected and the recipes for documentation to be produced, the sample is just one possible format that a manager could select. The templates and recipes can be varied from one project to the other.

The subject of the sample documentation was a set of fifteen Fortran programs called "DYNASOR-II; A Finite Element Program for the Dynamic Nonlinear Analysis of Shells of Revolution" developed for NASA by Joe R. Tillerson and Walter E. Haisler of the Aerospace Engineering Department, Texas A&M University. The DYNASOR II programs were developed to compute the nonlinear dynamic response of shells of revolution in relatively short periods of computer time for a large number of important shell problems.
The sample documentation for the DYNASOR II system is in three parts, a User's Manual, a Program Maintenance Manual and an Operations Manual. These manuals bring together information generated in various stages of the development process by such people as the specifier, designer, programmer and the validator. The sample documentation shows how the same information can be used to produce documentation for the system user, maintainer or operator.

The sample User's Manual is designed to explain to a scientist or engineer the problem solution and to show him how to prepare input to use the programs. It is prepared assuming that the user does not know computer programming.

The User's Manual begins with a title page, abstract, system overview, system flowcharts, and environment and configuration descriptions. These components can also be used without change in the maintenance and operator guides. The User's Manual then describes the method of analysis of the problem. Included next are guidelines for the user and program limitations. The next section covers the meaning of the input parameters and how to prepare the input. The solution of example problems using DYNASOR II are included to further clarify the system for the user. A list of references to the methods of analysis is given in case the user would like to study the techniques in detail. A special restart feature of the DYNASOR II program is explained in addition to a discussion on how to properly choose certain input parameters (in this case loads and temperatures) for best results. Functional flowcharts are included for the user interested in the structure of the programs, their functions and how they work together.

The Maintenance Manual is designed to familiarize the maintenance programmer with the set of DYNASOR II programs and to serve as his reference when modifying or correcting the programs. The maintenance manual is introduced by the title page, abstract, system overview, system flowchart and environment
The next sections contain detailed descriptions of the computer aspects of software being documented on system, global and local levels. On the system level there is a subroutine connectivity diagram, a system flowchart, and Job Control Language and deck setup descriptions. On the global level, overlay maps, global flowcharts, global data descriptions, control card descriptions and subroutine calls are included. On the local level, the Maintenance Manual contains detailed, detail suppressed, and functional flowcharts; local data descriptions; label cross references and individual program listings. Also included are system test samples and evaluation criteria. Other sections could be included here concerning such items as rules for programming practices, naming conventions, mathematical symbols and the like.

The Operator's Manual is designed to be used by the person who is responsible for running of the system on the computer. He is not required to have a knowledge of the scientific problem or of computer programming.

Again the manual begins with title page, abstract, system overview, system flowchart, and environment page duplicated automatically by the documentation system. The next sections describe components of the operation of the DYNASOR II runs. The purpose of each run is described along with its relationship to other runs, set up and run instructions, run frequency, run prerequisites, controls and schedules. Each data set is described along with the file characteristics. The job control language and control formats are detailed along with directions for setting up the deck.

Error messages are listed with explanations and actions to be taken. There is a section on checkpoint, restart, error procedures, backup and recovery procedures.

Any of the parts of any of the manuals in Appendix J can be produced separately at any stage of development where appropriate data is available in the data base.
5.0 DELIVERABLE ITEMS

Article II of NASA Contract NAS5-11911 requested the following deliverable items:

a) **Items of interest for documentation purposes that can be obtained automatically by the use of specifically designed computer programs and/or modifications to the operating system and associated software.** Output that can be obtained from commercial documentation aids is described in Appendix B. Text editors are described in Appendix C and program editors are described in Appendix F. No modifications to operating systems or associated software are recommended. Items of interest for documentation purposes that can be obtained automatically by the proposed automatic documentation system are described in Appendix I.

b) **Sample formats of useful and alternate printouts applicable to program documentation; derived from (a).** Sample outputs of the type of documentation that will be produced by the automatic documentation system are included in Appendix J.

c) **Methods by which the necessary data are obtained from source decks and/or operating system components (compilers, loaders, etc.).** A detailed description of the methods by which the necessary data are obtained from source decks and/or operating system components is described in the Appendix I.

d) **Programming disciplines, restrictions or coding requirements imposed upon programmers in order to enable them to produce documentation by using software recommended as a result of this effort.**
The report should indicate what behavior may be expected if required
disciplines are not observed and obeyed. In establishing these
constraints, consideration must be given as to what may be reasonably
expected in the way of programmer cooperation.

The automatic documentation system will be as easy to use as the typical
program editor. For programmers who use text editors in developing
their program descriptions, additional restrictions will not be
necessary to the textual part of the documentation. A large part
of the documentation necessary to describe the program can be
produced with no programmer restrictions as described in Appendix I.

For such things as high quality functional flowcharts, the programmer
will be required to stylize his program comments. A program developed
outside the automatic documentation system can be retrofitted into
the documentation system. In the design of the automatic documentation
system, a major effort will be made to minimize restrictions on
programmers.

e) Certain of the outputs desirable for program documentation purposes
may be obtained by modifying standard software components. Inasmuch
as most compilers, loaders, etc., extract relevant information and
retain it internally, much of the program development needed to
produce this data in readable form may be avoided; however, two points
must be borne in mind. First, this effort is not directed toward
a specific type or model of computer or software system; hence, the
techniques employed must be more or less generally applicable.
Second, it is most desirable that system modifications be kept to a
minimum; hence, it would be preferable to confine such modifications
to a few system modules as is practicable and to modularize any proposed extensions to the fullest possible extent. No standard software components will be modified.

f) If one or more specific operating systems are cited as illustrative examples, or for purposes of investigation, the particular modules or components of the system(s) requiring change, and the nature of such changes, should be fully described to the extent that they are penetrated by this study. No modules or components of operating systems need be changed to work with the proposed automatic documentation system.

g) While certain information may be extracted by altering system components, certain independent programs may be required to complete the package in order to supplement, broaden and/or organize meaningful documentation elements. Such programs must be fully described, including inputs required, functions performed, and outputs derived. Programs required to complete the package in order to supplement, broaden and/or organize meaningful documentation elements are Type 1 and Type 2 programs described in the proposal in Appendix I. Sample output provided by the system is described in Appendix J.

h) As a separate document or separate section of the final report, there will be a consolidation listing of all those techniques which could be implemented without changes in hardware or operating systems. As a subset of the above, there will be a designation of complexity level for the incorporation into a working program. Those which require little or no effort or change in style by the applications programmer should be specifically defined.
No changes will need to be made to program statements in a working program for documentation purposes. Documentation systems might make changes to the comment fields in a working program, but this will be done in a manner that will not, in any way, affect the other statements in operational programs. All noncomment program statements will be fully protected during any mode in which the program comments are being changed.

i) This section should be presented and recommended as an integrated, usable package which could be implemented immediately to provide a partial solution while further work is being done on the more elaborate improvements.

Commercially available documentation aids that can be used immediately are described in Appendix B. The initial phase of the automated documentation system to be implemented is described in Appendix I. Future elaborations and extensions to the system are also described in the proposal in Appendix I.

j) The outputs produced by the automated methods devised should contribute to final program documentation. Techniques should be recommended which best present a format for future computer program documentation.

A sample output of the type that would be produced by the automated system has been reproduced in Appendix J. This example demonstrates the type of output that would be produced by the system. The specific format of the manual desired by a project manager would be determined by the project manager.
<table>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIRES Corp.</td>
<td>AUTODIAGRAMMER II</td>
<td>720614</td>
</tr>
<tr>
<td>American National Standards Institute, Inc.</td>
<td>AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING</td>
<td>703905</td>
</tr>
<tr>
<td>Anderson, F. E.</td>
<td>AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER</td>
<td>650101</td>
</tr>
<tr>
<td>Applications Programming Co.</td>
<td>DYNACHART</td>
<td>720610</td>
</tr>
<tr>
<td>Applied Data Research</td>
<td>AUTOFLOW</td>
<td>720612</td>
</tr>
<tr>
<td>Applied Data Research, Inc.</td>
<td>LIBRARY</td>
<td>720601</td>
</tr>
<tr>
<td>ARMEDING, G.M.</td>
<td>THE JISS PRIMER</td>
<td>670804</td>
</tr>
<tr>
<td>ARPA Network Information Center</td>
<td>NETWORK INFORMATION CENTER USER GUIDE</td>
<td>720701</td>
</tr>
<tr>
<td>Auerrach Associates, Inc.</td>
<td>DATA MANIPULATION</td>
<td>720603</td>
</tr>
<tr>
<td>Azzi, A.I.</td>
<td>DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES</td>
<td>670901</td>
</tr>
<tr>
<td>Barnes, L.</td>
<td>RUNOFF: A PROGRAM FOR THE PREPARATION OF DOCUMENTS</td>
<td>700301</td>
</tr>
<tr>
<td>Beckwith, R.C.</td>
<td>A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION</td>
<td>680801</td>
</tr>
<tr>
<td>Bell Systems Practices</td>
<td>PROGRAM DOCUMENTATION STANDARDS</td>
<td>660403</td>
</tr>
<tr>
<td>Beloit Computer Center</td>
<td>AUTOCHART</td>
<td>729613</td>
</tr>
<tr>
<td>Berkeley, E.C.</td>
<td>RESEARCH IN COMPUTER-ASSISTED DOCUMENTATION OF COMPUTER PROGRAMS</td>
<td>701202</td>
</tr>
<tr>
<td>Blumberg, S.E.</td>
<td>AN INTERIM PROGRESS REPORT OF COMPUTER OUTPUT MICROFILM ACTIVITIES AND EXPERIENCES</td>
<td>709701</td>
</tr>
<tr>
<td>Brackett, J.W.</td>
<td>MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM</td>
<td>710602</td>
</tr>
<tr>
<td>Brooks, A.I.</td>
<td>DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES</td>
<td>670901</td>
</tr>
<tr>
<td>Business Automation</td>
<td>TALKING DOWN A PROGRAM</td>
<td>723501</td>
</tr>
<tr>
<td>Calcomp</td>
<td>FLOWGEN/F</td>
<td>727609</td>
</tr>
<tr>
<td>Callahan, M.D.</td>
<td>AUTODOC: COMPUTER-BASED ASSISTANCE FOR DOCUMENT PRODUCTION</td>
<td>672801</td>
</tr>
<tr>
<td>Cantrell, H.N.</td>
<td>LOGIC STRUCTURE TABLES</td>
<td>610601</td>
</tr>
<tr>
<td>Chapin, N.</td>
<td>FLOWCHARTS</td>
<td>610404</td>
</tr>
<tr>
<td></td>
<td>PARSING OF DECISION TABLES</td>
<td>673003</td>
</tr>
<tr>
<td></td>
<td>PROGRAM DOCUMENTATION: THE VALUABLE BURDEN</td>
<td>683901</td>
</tr>
<tr>
<td></td>
<td>PERSPECTIVE ON FLOWCHART PACKAGES</td>
<td>710301</td>
</tr>
</tbody>
</table>
CHUBA, C.P.
DEVELOPMENT OF A COMPUTER PROGRAM FOR
GENERATING TROUBLE-SHOOTING DECISION TREES 670901

COMMUNICATIONS COMPUTER PROGRAMMING CENTER
AUTOMATED WEATHER NETWORK 000101

COMP. SCI. DEPT. UNIVERSITY OF ILLINOIS
ILLIAC IV QUARTERLY PROGRESS REPORT 671201

COMPRESS, INC.
DOCUMENTATION SOFTWARE 670801
CONVISON 720605

CUHLi'b, C.F.
EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV. 000102

CONTI, E.
STUDY OF MACHINE-AIDED POST EDITING 680202

CONTROL DATA CORPORATION
CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 60100000 659701
CONTROL DATA 6400/6550/6600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60191200 690901
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 60195900 691001
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 60174900 691103
EDIT - KRONOS TEXT EDITOR (EDIT) REFERENCE MANUAL 710104
KRONOS TEXT EDITOR (EDIT) 710105

CULLINANE CORPORATION
PLUS D/A, A DIRECT ACCESS SOFTWARE SECURITY AND CONTROL SYSTEM 720201

DATA FOR MANAGEMENT DECISION
FORTRAN VARIABLE NAME DOCUMENTER 720607
DATA INSTRUMENTS CO.
AUTOCOC 720615

DEFENSE DOCUMENTATION CENTER
BIBLIOGRAPHY ON INFORMATION SCIENCES, VOLUME I 680201
COMPUTER IN INFORMATION SCIENCES, VOLUME III 681092
MICROFICHE, MICROFILM, AND RELATED EQUIPMENT 711001

DEMUTH, N.
STUDY OF MACHINE-AIDED POST EDITING 680202

DIAL, R.R.
DECISION TABLE TRANSLATION 700902

DNA SYSTEMS, INC.
FORLOG 720608

ELLIS, T.O.
ARPA NETWORK SERIES 1. INTRODUCTION TO THE ARPA NETWORK AT RAND AND TO THE RAND VIDEO GRAPHICS SYSTEM 710901

ENSLO, JR. P.M.
DIAGNOSTIC PROGRAMMING FOR DIGITAL COMPUTERS A BIBLIOGRAPHY 670472

FELDMANN, C.G.
MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM 700904

FELSMA, W.O.
COST ADVANTAGES OF AN INTEGRATED DOCUMENTATION APPROACH 701113

FERGUS, R.M.
AN INTRODUCTION TO DECISION TABLES 680701
MAIBT, L.M.

A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS

HANNEY, R.

AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION

HANNON, T.J.

DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES

HOLZER, J.E.

A SYSTEM OF PROGRAM DOCUMENTATION

HOWDEN, W. E.

A PROGRAM FOR THE CONSTRUCTION OF PORT FLOWCHARTS

HUNTER, J.R.

DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES

IBM CORPORATION

PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS

USER'S MANUAL NMCS INFORMATION PROCESSING SYSTEM (NIPS) (IBM 1410/7074)

NMCS INFORMATION PROCESSING SYSTEM IBM 1410/7074 GENERAL PURPOSE COMPUTER MULTIPLE OUTPUT AND FILE GENERATION

SYSTEM/360 ADMINISTRATIVE TERMINAL SYSTEM--OS

INDEX PREPARATION FOR PUBLICATIONS INDUSTRY

PROBLEM LANGUAGE ANALYZER (PLA) USERS' INTRODUCTION

SYSTEM/360 FLOWCHART USER'S MANUAL OPERATOR'S MANUAL

A CONVERSATIONAL CONTEXT-DIRECTED EDITOR

IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LOADER PROGRAM NUMBERS
MANAGEMENT SYSTEMS CORPORATION

MANTech Corporation

Marks, S.L.

Martin, W. W.

FLOWCHARTING: SHORTHAND, ANALYSIS AND MODEL.

McClure, C.W.

AUTOMATED ENGINEERING DESIGN AN APPROACH TO AUTOMATED DOCUMENTATION

Meadow, C.T.

EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV.

Mel'chuk, I.A.

AUTOMATION IN LINGUISTICS

Menkus, R.

DEFINING ADEQUATE SYSTEMS DOCUMENTATION

Mercep, V.

AUTOCHART NEW METHOD CUTS MANUFACTURING.

Mills, M.D.

SYNTAX-DIRECTED DOCUMENTATION FOR PL360

Montalano, M.

TABLES, FLOW CHARTS, AND PROGRAM LOGIC

Mujumdar, M.J.

DOCUMENTATION: MOTIVATION AND TRAINING OR AUTOMATION

Muthukrishnan, C.R.

ON THE CONVERSION OF DECISION TABLES TO

Muthukrishnan, C.R.

CONTINUATION

COMPUTER PROGRAMS

NATIONAL COMPUTER ANALYSIS, INC.

CORAL FLOWCHARTING

QUICK-DRAW

NATIONAL MILITARY COMMAND SYSTEM SUPPORT CENTER COMPUTER

USER'S MANUAL NMC'S INFORMATION PROCESSING SYSTEM (NIPS) IBM 1410/7010

Neely, M.D.

AUTOMATIC PROGRAM ANNOTATION (AUTOMOTO)

Neperlo, R.

MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDY

Ninke, W. M.

THE GROWTH OF COMPUTER GRAPHICS AT BELL LABORATORIES

O'Brien, F.

A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION

OPERATIONS DIVISION STRATEGIC SYSTEMS DEPARTMENT

AUTOMATIC DATA PROCESSING SUPPORT FOR FLAG PLANT

Pansophic Systems, Inc.

Panvalet - the direct access library system

Pardee, S.

BellFlow automatic flowcharting system

Parkfr, D.M.

ON THE FEASIBILITY OF FORMAL CERTIFICATION

PenSiero, W.S.

SLIP - Source Library Inquiry Program, No.13
<table>
<thead>
<tr>
<th>Author</th>
<th>Title</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tatum, J.C.</td>
<td>Achieving Proper Program Documentation</td>
<td>711101</td>
</tr>
<tr>
<td>Teplitz, A.</td>
<td>Microfiche Storage and Retrieval Systems Study:</td>
<td>700801</td>
</tr>
<tr>
<td>Thomas, V.L.</td>
<td>Automated Documentation of an Assembly Program</td>
<td>701110</td>
</tr>
<tr>
<td>Tupac, J.W.</td>
<td>An Approach to Software Evaluation</td>
<td>670401</td>
</tr>
<tr>
<td>Tyson, J.W.</td>
<td>Automatic Program Annotation (AUTOMOTE)</td>
<td>701104</td>
</tr>
<tr>
<td>USAF</td>
<td>Military Standards</td>
<td>700003</td>
</tr>
<tr>
<td></td>
<td>Documentation Standards</td>
<td>720102</td>
</tr>
<tr>
<td>Van-Dam, A.</td>
<td>C4-Line Text Editing: A Survey</td>
<td>723604</td>
</tr>
<tr>
<td>Vartabedian, A.G.</td>
<td>The Design of Visual Displays</td>
<td>700901</td>
</tr>
<tr>
<td>Veinott, C.G.</td>
<td>Programming Decision Tables in Fortran, COGUL, or ALGOL</td>
<td>660102</td>
</tr>
<tr>
<td>Walsh, D.</td>
<td>A Guide for Software Documentation</td>
<td>700001</td>
</tr>
<tr>
<td>Wells, E.E.</td>
<td>A Library of Data Collection and Manipulation Subroutines</td>
<td>710902</td>
</tr>
</tbody>
</table>
ALGORITHMS

AUTOMATIC PREPARATION OF FLOWCHART LISTING 580101
TABLES, FLOW CHARTS, AND PROGRAM LOGIC 620901
A FLOW ANALYSIS ALGORITHM 641001
USE OF DECISION TABLES IN COMPUTER PROGRAMMING 650102
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO COMPUTER PROGRAMS 661101
DATA, DOCUMENTATION, AND DECISION TABLES 660101
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS I: MINIMUM AVERAGE PROCESSING TIME 660801
CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS BY RULE MASK TECHNIQUES 661101
DECISION TABLES 670602
PARSING OF DECISION TABLES 670803
DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES 670901
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II: MINIMUM STORAGE REQUIREMENT 671102
SIMULATION WITH DECISION TABLES 680101
THE USE OF DECISION TABLES WITHIN SYSTEMATICS 680802
AMBIGUITY IN LIMITED ENTRY DECISION TABLES 681001
THE INTERPRETATION OF LIMITED ENTRY DECISION TABLE FORMAT AND RELATIONSHIPS AMONG CONDITIONS 691101
ON THE CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS 700601
DECISION TABLE TRANSLATION 700902
RESEARCH IN COMPUTER-ASSISTED DOCUMENTATION OF COMPUTER PROGRAMS 701202
UNIVAC 1100 SERIES OPERATING SYSTEM 710102

CONTINUATION

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO COMPUTER PROGRAMS: A PROPOSED MODIFICATION TO POLLACK'S ALGORITHM 710201
A LIBRARY OF DATA COLLECTION AND MANIPULATION SUBROUTINES 719902
ASSEMBLER-Routines
ILLIAC IV QUARTERLY PROGRESS REPORT 671201
CONTROL SECTION CROSS-REFERENCE AND LINK EDIT EDITOR 681003
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 6019260 691001
IBM SYSTEM/360 OPERATING SYSTEM ASSEMBLER PROGRAMMERS GUIDE PROGRAM NUMBER 360L-AS-037 701203

COMPILERS
LOGIC STRUCTURE TABLES 612001
UNIVAC 1100 COROL UP-4048 679101
THE GROWTH OF COMPUTER GRAPHICS AT BELL LABORATORIES 680601
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COROL REFERENCE MANUAL PUBLICATION NUMBER 60191260 690901
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 60174900 691103
IBM SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARD COROL PROGRAMMERS GUIDE NUMBERS 3605-CP-545 700602
IBM SYSTEM/360 OPERATING SYSTEM FORTRAN IV (G AND II) PROGRAMMERS GUIDE PROGRAM NUMBERS 3605-FU-500 3605-FU-520 700603
IBM SYSTEM/360 OPERATING SYSTEM PL/I (P) PROGRAMMERS GUIDE PROGRAM NUMBER 3605-LP-511 710101
UNIVAC 1100 SERIES OPERATING SYSTEM 719102
COMPUTER-PERSONNEL

COMPUTERIZING A GOVERNMENT DATA SYSTEM: A MANAGEMENT OVERVIEW OF THE STEPS REQUIRED AND THE TIME NEEDED. 671202

VIEWS ON COMPUTER PROGRAM DOCUMENTATION AND AUTOMATION 701107

ACHIEVING PROPER PROGRAM DOCUMENTATION 711101

COPYING

INDEX PREPARATION FOR PUBLICATIONS INDUSTRY 680902

COST-ANALYSIS

ON THE FEASIBILITY OF FORMAL CERTIFICATION 000103

USE OF DECISION TABLES IN COMPUTER PROGRAMMING 650102

CONVERSION OF LIMITED ENTRY DECISION TABLES TO COMPUTER PROGRAMS 651101

DECISION TABLES FOR RISKER MANAGEMENT SYSTEMS 660301

A SYSTEM OF PROGRAM DOCUMENTATION 660901

CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS BY RULE MASK TECHNIQUES 661101

AN APPROACH TO SOFTWARE EVALUATION 670401

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II: MINIMUM STORAGE REQUIREMENT 671102

COBOL FLOWCHARTING 680301

DOCUMENTATION STANDARDS 700002

IN THE CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS 700601

AN INTERIM PROGRESS REPORT OF COMPUTER OUTPUT MICROFILM ACTIVITIES AND EXPERIENCES 700701

DEFINING ADEQUATE SYSTEMS DOCUMENTATION PERSPECTIVE ON FLOWCHART PACKAGES 701201

DATA COMPRESSION IN LARGE

COST-ANALYSIS

CONTINUATION

BUSINESS-ORIENTED FILES 711002

ACHIEVING PROPER PROGRAM DOCUMENTATION 711101

COST-EFFECTIVENESS

TALKING DOWN A PROGRAM 720501

CROSS-REFERENCES

COMCHART 720605

SUPERFORTRAN VARIABLE NAME DOCUMENTER 720606

DYNACHART 720610

QUICK-DRAW 720611

AUTOFLOW 720612

AUTOCART 720613

AUTOGRAMMER II 720614

AUTOCUC 720615

EZFLOW 720616

DATA-DESCRIPTION

THE INTEGRATION OF SYSTEM SPECIFICATIONS AND PROGRAM CISING. A REPORT ON EXPERIENCE OF THE MEDLARS II PROJECT. 701101

DATA-PROCESSING

EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV. 001102

DATA, DOCUMENTATION, AND DECISION TABLES 660101

DECISION TABLES FOR RISKER MANAGEMENT SYSTEMS 660301

AUTOMATIC DATA PROCESSING SUPPORT FOR FLAG PLINT 660402

SLIP - SOURCE LIBRARY INQUIRY PROGRAM, NO. 3600-GC.11.009 660601

A SYSTEM OF PROGRAM DOCUMENTATION 660901
DATA-PROCESSING

DATA-PROCESSING CONTINUATION

CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS BY RULE MASK TECHNIQUES 661101
PARSING OF DECISION TABLES 670803
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II: MINIMUM STORAGE REQUIREMENT 671102
COMPUTERIZING A GOVERNMENT DATA SYSTEM: A MANAGEMENT OVERVIEW OF THE STEPS REQUIRED AND THE TIME NEEDED 671202
SIMULATION WITH DECISION TABLES 680101
USER'S MANUAL NICS INFORMATION PROCESSING SYSTEM (NIPS) (IBM 1410/7010) 680102
BIBLIOGRAPHY ON INFORMATION SCIENCES, VOLUME 1 680201
STUDY OF MACHINE-AIDED POST EDITING 680202
NICS INFORMATION PROCESSING SYSTEM IBM 141C/77110 GENERAL PURPOSE COMPONENTS MULTIFILE OUTPUT AND FILE GENERATION 680302
S/360 ASSEMBLIER/COMPILER LISTINGS LIBRARY MAINTENANCE, NO. 3600-00-C.015 681101
DOCUMENTATION STANDARDS 690103
VIEW'S ON COMPUTER PROGRAM DOCUMENTATION AND AUTOMATION 701107
MICROFICHE, MICROFILM, AND RELATED EQUIPMENT 711001
DATA COMPRESSION IN LARGE BUSINESS-ORIENTED FILES 711002
ON-LINE TEXT EDITING: A SURVEY 720604

DATA-PROCESSING TERMINALS

CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 60100000 650701
WYLUR 700903
APPA NETWORK SERIES I: INTRODUCTION TO THE ARPA NETWORK AT RAND AND TO THE RAND VIDEO GRAPHICS SYSTEM 710901

DATA-STORAGE

EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-V 609102
DATA, DOCUMENTATION, AND DECISION TABLES 660101
INFORMATION SCIENCES: SOME RESEARCH DIRECTIVES 663401
SLIP - SOURCE LIBRARY INQUIRY PROGRAM, NO. 3600-00-3.079 660601
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS III: MINIMUM STORAGE REQUIREMENT 671102
AUTOMATION IN LINGUISTICS 680103
S/360 ASSEMBLIER/COMPILER LISTINGS LIBRARY MAINTENANCE, NO. 3600-00-C.015 681101
IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LINKAGE PROGRAM NUMBERS 3605-ED-510 3605-ED-521 3605-ED-547 691102
ON THE CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS 702501
MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDY 702901
DATA COMPRESSION IN LARGE BUSINESS-ORIENTED FILES 711002
LIBRARIAN 723601

DATA-TRANSMISSION

DATA, DOCUMENTATION, AND DECISION TABLES 667101
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS III: MINIMUM STORAGE REQUIREMENT 671102
IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LINKAGE PROGRAM NUMBERS 3605-ED-510 3605-ED-521 3605-ED-547 691102
AUTOMATIC SELECTIVE DOCUMENTATION SERVICES 711002
MICROFICHE, MICROFILM, AND RELATED EQUIPMENT 711001
## DECISION-TABLES

### LOGIC STRUCTURE TABLES
- TABLES, FLOW CHARTS, AND PROGRAM LOGIC
- USING DECISION STRUCTURE TABLES
- USE OF DECISION TABLES IN COMPUTER PROGRAMMING
- CONVERSION OF LIMITED ENTRY DECISION TABLES TO COMPUTER PROGRAMS
- DATA, DOCUMENTATION, AND DECISION TABLES
- PROGRAMMING DECISION TABLES IN FORTRAN, COBOL OR ALGOL
- DECISION TABLES FOR RISK MANAGEMENT SYSTEMS
- PROGRAM DOCUMENTATION STANDARDS
- CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS
- MINIMUM AVERAGE PROCESSING TIME
- CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS BY RULE MASK TECHNIQUES
- DECISION TABLES
- PARSING OF DECISION TABLES
- CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS
- MINIMUM STORAGE REQUIREMENT
- SIMULATION WITH DECISION TABLES
- AN INTRODUCTION TO DECISION TABLES
- THE USE OF DECISION TABLES WITHIN SYSTEMATICS
- GOOD DECISION TABLES AND THEIR USES
- AMBIGUITY IN LIMITED ENTRY DECISION TABLES
- DOCUMENTATION STANDARDS
- THE INTERPRETATION OF LIMITED ENTRY DECISION TABLES FOR PROGRAM AND RELATIONSHIPS AMONG CONDITIONS

### DECISION-TABLES (CONTINUATION)

- A GUIDE FOR SOFTWARE DOCUMENTATION
- DOCUMENTATION STANDARDS
- MILITARY STANDARDS
- ON THE CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS
- DECISION TABLE TRANSLATION
- AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING
- DEFINING ADEQUATE SYSTEMS DOCUMENTATION
- CONVERSION OF LIMITED-ENTRY DECISION TABLES TO COMPUTER PROGRAMS: A PROPOSED MODIFICATION TO POLLACK'S ALGORITHM

### DESIGN

- EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV
- FLOWCHARTS
- AUTOMATIC PREPARATION OF FLOWCHART LISTING
- COMPUTER-DRAWN FLOWCHARTS
- AUTOCHART NEW METHOD CUTS MANUFACTURING COSTS
- A FLOW ANALYSIS ALGORITHM
- AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER
- CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 6C10000
- AUTOMATIC DATA PROCESSING SUPPORT FOR ILLIAC IV
- FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS
- UNIVAC 1109 CMPL UP-4040
- ILLIAC IV QUARTERLY PROGRESS REPORT
PROGRAM DOCUMENTATION: THE VALUABLE BURDEN
680501
THE GROWTH OF COMPUTER GRAPHICS AT BELL LABORATORIES
680601
AN INTRODUCTION TO DECISION TABLES
680701
A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION
680801
THE USE OF DECISION TABLES WITHIN SYSTEMATICS
680802
DOCUMENTATION STANDARDS
680103
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60194200
690901
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60194900
691001
IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LOADER PROGRAM NUMBERS 3405-ED-510 1655-ED-521 3605-LO-547
691101
CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 60174900
691103
DOCUMENTATION STANDARDS
700002
SYNTAX-DIRECTED DOCUMENTATION FOR PL/360
700401
IBM SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARD COROL PROGRAMMERS GUIDE NUMBERS 3605-CD-545
700602
IBM SYSTEM/360 OPERATING SYSTEM FORTRAN IV (C AND H) PROGRAMMERS GUIDE PROGRAM NUMBERS 3605-FH-500 3605-FO-529
700603
MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDY
700801
THE DESIGN OF VISUAL DISPLAYS
700901
DOCUMENTATION: MOTIVATION AND TRAINING OR AUTOMATION
701102
MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION A FEATURE OF TOTAL SYSTEM
EDITING

SLIP - SOURCE LIBRARY INQUIRY PROGRAM, NO. 3600-C-73-79

STUDY IF MACHINE-AIDED POST EDITING

SYSTEM 360 ADMINISTRATIVE TERMINAL SYSTEM-00

INDEX PREPARATION FOR PUBLICATIONS INDUSTRY

A CONVERSATIONAL CONTEXT-DIRECTED EDITOR

RUNOFF: A PROGRAM FOR THE PREPARATION OF DOCUMENTS

CLUT - CARD LIBRARY ON TAPE

WYLBUK

AUTOFLOW ENHANCEMENTS OF DOCUMENTATION AND MAINTENANCE OF SCIENTIFIC APPLICATIONS

AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION

PROGRAM AUTOMATED DOCUMENTATION METHODS

AUTOMATIC EDITING OF MANUALS

ED PRODUC Transport - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE

EDIT - KRMIS TEXT EDITOR (EDIT) REFERENCE MANUAL

KRMIS TEXT EDITOR (EDIT)

TECN - TEXT EDITOR AND CORRECTOR - REFERENCE MANUAL

CFMS - CARD FILE MAINTENANCE SYSTEM

PLUS D/A, A DIRECT ACCESS SOFTWARE SECURITY AND CONTROL SYSTEM

PANVALET - THE DIRECT ACCESS LIBRARY SYSTEM

XEROX TEXT LANGUAGE AND OPERATIONS REFERENCE MANUAL

LIBRARIAN

EDITING CONTINUATION

PROGRAM/MANAGE, A PROGRAM TO MAINTAIN SOURCE PROGRAM

DATA MANIPULATION

ON-LINE TEXT EDITING: A SURVEY

PRESS USER'S GUIDE

NETWORK INFORMATION CENTER USER GUIDE

ASTECOMP

EXECUTIVE-ROUTINES

UNIVAC 1100 SERIES OPERATING SYSTEM

FLOWCHARTS

FLOWCHARTS

AUTOMATIC PREPARATION OF FLOWCHART LISTING

A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS

LOGIC STRUCTURE TABLES

TABLES, FLOW CHARTS, AND PROGRAM LOGIC

COMPUTER-DRAWN FLOWCHARTS

PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS

AUTOCHART NEW METHOD CUTS MANUFACTURING

A FLOW ANALYSIS ALGORITHM

AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER

USE OF DECISION TABLES IN COMPUTER PROGRAMMING

PROGRAMMING DOCUMENTATION

DATA, DOCUMENTATION, AND DECISION TABLES

DECISION TABLES FOR BERRY MANAGEMENT SYSTEMS

PROGRAM DOCUMENTATION STANDARDS
<table>
<thead>
<tr>
<th>FLOWCHARTS CONTINUATION</th>
<th>FLOWCHARTS CONTINUATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLOWTRACE: A PROGRAM FOR FLOWCHARTING PROGRAMS</td>
<td>DESIGN</td>
</tr>
<tr>
<td>AUTOCOC: COMPUTER-BASED ASSISTANCE FOR DOCUMENT PRODUCTION</td>
<td>BELLFLOW AUTOMATIC FLOWCHARTING SYSTEM</td>
</tr>
<tr>
<td>DOCUMENTATION SOFTWARE</td>
<td>AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION</td>
</tr>
<tr>
<td>A PROGRAM FOR THE CONSTRUCTION OF PERT FLOWCHARTS</td>
<td>PROGRAM AUTOMATED DOCUMENTATION METHODS</td>
</tr>
<tr>
<td>SIMULATION WITH DECISION TABLES</td>
<td>AUTOMATIC EDITING OF MANUALS</td>
</tr>
<tr>
<td>COHOL FLOWCHARTING</td>
<td>DEFINING ADEQUATE SYSTEMS DOCUMENTATION</td>
</tr>
<tr>
<td>KMCS INFORMATION PROCESSING SYSTEM IBM 1410/7210 GENERAL PURPOSE COMPONENTS MULTIFILE OUTPUT AND FILE GENERATION</td>
<td>PERSPECTIVE ON FLOWCHART PACKAGES</td>
</tr>
<tr>
<td>A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION.</td>
<td>A LIBRARY OF DATA COLLECTION AND MANIPULATION SUBROUTINES</td>
</tr>
<tr>
<td>AMBIGUITY IN LIMITED ENTRY DECISION TABLES</td>
<td>DOCUMENTATION STANDARDS</td>
</tr>
<tr>
<td>SYSTEM/160 FLOWCHART USER'S MANUAL OPERATOR'S MANUAL</td>
<td>STANDARDS GUIDE FOR SPACE AND EARTH SCIENCES COMPUTER SOFTWARE</td>
</tr>
<tr>
<td>DOCUMENTATION STANDARDS</td>
<td>CONMCHART</td>
</tr>
<tr>
<td>THE INTERPRETATION OF LIMITED ENTRY DECISION TABLE FORMAT AND RELATIONSHIPS AMONG CONDITIONS</td>
<td>FORFLO</td>
</tr>
<tr>
<td>A GUIDE FOR SOFTWARE DOCUMENTATION</td>
<td>FLINGEN/F</td>
</tr>
<tr>
<td>DOCUMENTATION STANDARDS</td>
<td>DYNACHART</td>
</tr>
<tr>
<td>MILITARY STANDARDS</td>
<td>QUICK-DRAW</td>
</tr>
<tr>
<td>DECISION TABLE TRANSLATION</td>
<td>AUTOFLOW</td>
</tr>
<tr>
<td>AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING</td>
<td>AUTOCHART</td>
</tr>
<tr>
<td>AUTOFLCH ENHANCEMENTS OF DOCUMENTATION AND MAINTENANCE OF SCIENTIFIC APPLICATIONS</td>
<td>AUTOGRAMMER II</td>
</tr>
<tr>
<td>DOCUMENTATION: MOTIVATION AND TRAINING OR AUTOMATION</td>
<td>AUTOCOC</td>
</tr>
<tr>
<td>MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION A FEATURE OF TOTAL SYSTEM</td>
<td>EZFLOW</td>
</tr>
<tr>
<td></td>
<td>FORMS</td>
</tr>
<tr>
<td></td>
<td>AUTOMATED WEATHER NETWORK</td>
</tr>
<tr>
<td></td>
<td>PROGRAM DOCUMENTATION STANDARDS</td>
</tr>
<tr>
<td></td>
<td>PROGRAM DOCUMENTATION: THE VALUABLE GURDEN</td>
</tr>
<tr>
<td></td>
<td>DOCUMENTATION STANDARDS</td>
</tr>
<tr>
<td>FORMS</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>ON-LINE MANAGEMENT SYSTEMS INFORMATION</td>
<td>690401</td>
</tr>
<tr>
<td>DOCUMENTATION STANDARDS</td>
<td>700002</td>
</tr>
<tr>
<td>MILITARY STANDARDS</td>
<td>700003</td>
</tr>
<tr>
<td>DOCUMENTATION STANDARDS</td>
<td>720102</td>
</tr>
<tr>
<td>STANDARDS GUIDE FOR SPACE AND EARTH SCIENCES COMPUTER SOFTWARE</td>
<td>720103</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FUNCTIONAL-SUBROUTINES</th>
</tr>
</thead>
<tbody>
<tr>
<td>COST ADVANTAGES OF AN INTEGRATED DOCUMENTATION APPROACH</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GRAPHIC-METHODS</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTOMATIC PREPARATION OF FLOWCHART LISTING</td>
</tr>
<tr>
<td>A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS</td>
</tr>
<tr>
<td>COMPUTER-DRAWN FLOWCHARTS</td>
</tr>
<tr>
<td>PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS</td>
</tr>
<tr>
<td>AUTOCART NEW METHOD CUTS MANUFACTURING</td>
</tr>
<tr>
<td>A FLOW ANALYSIS ALGORITHM</td>
</tr>
<tr>
<td>AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER</td>
</tr>
<tr>
<td>INFORMATION SCIENCES: SOME RESEARCH DIRECTIVES</td>
</tr>
<tr>
<td>FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS</td>
</tr>
<tr>
<td>DOCUMENTATION SOFTWARE</td>
</tr>
<tr>
<td>A PROGRAM FOR THE CONSTRUCTION OF PERT FLOWCHARTS</td>
</tr>
<tr>
<td>ILLIAC IV QUARTERLY PROGRESS REPORT</td>
</tr>
<tr>
<td>THE GROWTH OF COMPUTER GRAPHICS AT BELL LABORATORIES</td>
</tr>
<tr>
<td>A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GRAPHIC-METHODS</th>
</tr>
</thead>
<tbody>
<tr>
<td>A GUIDE FOR SOFTWARE DOCUMENTATION</td>
</tr>
<tr>
<td>OVERLAY TREE PROCESSOR</td>
</tr>
<tr>
<td>THE DESIGN OF VISUAL DISPLAYS</td>
</tr>
<tr>
<td>PROGRAM AUTOMATED DOCUMENTATION METHODS</td>
</tr>
<tr>
<td>PERSPECTIVE ON FLOWCHART PACKAGES</td>
</tr>
<tr>
<td>A SURVEY OF DATA STRUCTURES FOR COMPUTER GRAPHICS SYSTEMS</td>
</tr>
<tr>
<td>HARD LOOK AT SOFTWARE DOCUMENTATION</td>
</tr>
<tr>
<td>ARPA NETWORK SERIES 1, INTRODUCTION TO THE ARPA NETWORK AT RAND AND TO THE RAND VIDEO GRAPHICS SYSTEM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HEURISTIC-METHODS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROGRAMMING DECISION TABLES IN FORTRAN, COBOL OR ALGOL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INPUT-OUTPUT-RUTINES</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTOMATED WEATHER NETWORK</td>
</tr>
<tr>
<td>COMPUTER-DRAWN FLOWCHARTS</td>
</tr>
<tr>
<td>PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS</td>
</tr>
<tr>
<td>AUTOCART NEW METHOD CUTS MANUFACTURING</td>
</tr>
<tr>
<td>AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER</td>
</tr>
<tr>
<td>CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 6010000</td>
</tr>
<tr>
<td>FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS</td>
</tr>
<tr>
<td>AUTODOC: COMPUTER-BASED ASSISTANCE FOR DOCUMENT PRODUCTION</td>
</tr>
<tr>
<td>UNIVAC 1109 COBOL UP-4048</td>
</tr>
<tr>
<td>A PROGRAM FOR THE CONSTRUCTION OF PERT FLOWCHARTS</td>
</tr>
<tr>
<td>STUDY OF MACHINE-AIDED POST EDITING</td>
</tr>
</tbody>
</table>
## MAINTENANCE CONTINUATION

| GOH DECISION TABLES AND THEIR USES                  | 680901 |
| S/36 Asembler/Compiler Listings Library Maintenance, No. 3600-00,0,015 | 681101 |
| A CONVERSATIONAL CONTEXT-DIRECTED EDITOR            | 690701 |
| DOCUMENTATION STANDARDS                             | 700002 |
| CLUT - CARD LIBRARY ON TAPE                          | 700604 |
| WYLBUK-AUTOMATED ENGINEERING DESIGN AN APPROACH TO AUTOMATED DOCUMENTATION | 701108 |
| ED PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE | 710103 |
| EDIT - KRUNJS TEXT EDITOR EDIT REFERENCE MANUAL      | 710104 |
| TEGO - TEXT EDITOR AND CORRECTOR - REFERENCE MANUAL | 710603 |
| DATA COMPRESSION IN LARGE BUSINESS-ORIENTED FILES   | 711100 |
| ACHIEVING PROPER PROGRAM DOCUMENTATION              | 711101 |
| CFMS - CARD FILE MAINTENANCE SYSTEM                  | 720101 |
| PLUS DPA, A DIRECT ACCESS SOFTWARE SECURITY AND CONTROL SYSTEM | 720201 |
| PANVALET - THE DIRECT ACCESS LIBRARY SYSTEM          | 720502 |
| LIBRARIAN                                            | 720601 |
| PROGRAM/MANAGE, A PROGRAM TO MAINTAIN SOURCE PROGRAM | 720602 |
| DATA MANIPULATION                                   | 720603 |
| ON-LINE TEXT EDITING: A SURVEY                       | 720604 |

## MANUALS CONTINUATION

| CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 60100000 | 650701 |
| PROGRAMMING DOCUMENTATION                                                   | 651001 |
| PROGRAM DOCUMENTATION STANDARDS                                             | 660403 |
| A SYSTEM OF PROGRAM DOCUMENTATION                                           | 660901 |
| UNIVAC 1100 COROL UP-404A                                                   | 670101 |
| THE JOSSE PRIMER                                                            | 670604 |
| USER'S MANUAL NYCS INFORMATION PROCESSING SYSTEM (NIPS) (IBM 9110/7901)     | 680102 |
| IBM NYCS INFORMATION PROCESSING SYSTEM IBM 9110/7910 GENERAL PURPOSE COMPONENTS MULTIPLE OUTPUT AND FILE GENERATION | 680302 |
| DOCUMENTATION STANDARDS                                                    | 690103 |
| CONTROL DATA 6460/6600/6600 COMPUTER SYSTEMS CONVIVL REFERENCE MANUAL PUBLICATION NUMBER 60191200 | 699901 |
| CONTROL DATA 6460/6500/6600 COMPUTER SYSTEMS CONVIVL REFERENCE MANUAL PUBLICATION NUMBER 63150300 | 691001 |
| IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LOADER PROGRAM NUMBERS 3605-ED-510 3605-ED-521 3605-ED-547 | 671102 |
| CONTROL DATA 6460/6500/6600 COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 60174900 | 691103 |
| MILITARY STANDARDS                                                          | 700003 |
| RENIF: A PROGRAM FOR THE PREPARATION OF DOCUMENTS                           | 700301 |
| IBM SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARD COROL PROGRAMMERS GUIDE NUMBERS 3645-CB-545 | 700602 |
| IBM SYSTEM/360 OPERATING SYSTEM FORTRAN IV (G AND H) PROGRAMMERS GUIDE PROGRAM NUMBERS 3645-FD-500 3645-FD-525 | 700603 |
| COSMIC: PROGRAM DOCUMENTATION EXPERIENCE                                   | 701101 |
| IBM SYSTEM/360 OPERATING SYSTEM ASSEMBLY PROGRAMMERS GUIDE PROGRAM NUMBER 360L-AS-037 | 701203 |
| IBM SYSTEM/360 OPERATING SYSTEM PL/I(P)                                     | 701303 |
MANUALS

PROGRAMMERS GUIDE PROGRAM NUMBER

360S-EN-511

INIVAC 1100 SERIES OPERATING SYSTEM

10101

HARD LOOK AT SOFTWARE DOCUMENTATION

10102

STANDARDS GUIDE FOR SPACE AND EARTH SCIENCES COMPUTER SOFTWARE

10103

MANUFACTURERS

ON THE FEASIBILITY OF FORMAL CERTIFICATION

000103

USING DECISION STRUCTURE TABLES

640103

CONTROL DATA 6600 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 661100002

650701

UNIVAC 1100 CONTROLLER UP-4040

670101

AN APPROACH TO SOFTWARE EVALUATION

670401

CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS CONTROLL REFERENCE MANUAL PUBLICATION NUMBER 60141200

690901

CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS CONTROLL REFERENCE MANUAL PUBLICATION NUMBER 60144900

691103

IBM SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARDS GUIDE NUMBERS 360S-CB-545

700602

IBM SYSTEM/360 OPERATING SYSTEM FORTRAN IV AND II PROGRAMMERS GUIDE PROGRAM NUMBERS 360S-FO-500 360S-FO-520

700603

COSMIC PROGRAM DOCUMENTATION EXPERIENCE

701111

UNIVAC 1100 SERIES OPERATING SYSTEM

710102

MASTER-DATA-DESCRIPTION

THE INTEGRATION OF SYSTEM SPECIFICATIONS AND PROGRAM Coding, A REPORT ON EXPERIENCE OF THE NELAKS II PROJECT

701114

META-LANGUAGE

EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV

600102

META-LANGUAGE

FLOWPACE, A PROGRAM FOR FLOWCHARTING PROGRAMS

601201

AUTOMATION IN LINGUISTICS

601301

AUTOMATED ENGINEERING DESIGN AN APPROACH TO AUTOMATED DOCUMENTATION

701108

METAPROGRAMS

COST ADVANTAGES OF AN INTEGRATED DOCUMENTATION APPROACH

701113

MICROFILM

THE GROWTH OF COMPUTER GRAPHICS AT BELL LABORATORIES

680101

A GUIDE FOR SOFTWARE DOCUMENTATION

700101

AN INTERIM PROGRESS REPORT OF COMPUTER OUTPUT MICROFILM ACTIVITIES AND EXPERIENCES

700701

MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDY

703801

AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION

700116

AUTOMATIC SELECTIVE DOCUMENTATION SERVICES

712402

MICROFICHE, MICROFILM, AND RELATED EQUIPMENT

711501

OBJECT-PROGRAMS

CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 601100002

650701

CONTROL SECTION CROSS-REFERENCE AND LINK EDIT EDITOR

681903

IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LOADER PROGRAM NUMBERS 360S-C0-510 360S-C0-521 360S-LU-547

691102

UNIVAC 1100 SERIES OPERATING SYSTEM

710102

OVERLAY-TREE-STRUCTURE

OVERLAY TREE PROCESSOR

703101
<table>
<thead>
<tr>
<th>PLOTTING</th>
<th>PRINTING</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTOMATIC PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER</td>
<td>AUTOMATIC PREPARATION OF FLOWCHART LISTING</td>
</tr>
<tr>
<td>THE GROWTH OF COMPUTER GRAPHICS AT RELL LABORATORIES</td>
<td>COMPUTER DRAWN FLOWCHARTS</td>
</tr>
<tr>
<td>A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION</td>
<td>A FLOW ANALYSIS ALGORITHM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROGRAMMING-LANGUAGES</th>
<th>CONTINUATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODL (OR ALGOL)</td>
<td>6601C2</td>
</tr>
<tr>
<td>INFORMATION SCIENCES: SOME RESEARCH DIRECTIVES</td>
<td>6601C1</td>
</tr>
<tr>
<td>UNIVAC 110A MANUAL UP-4448</td>
<td>6701C1</td>
</tr>
<tr>
<td>COMPUTER IN INFORMATION SCIENCES, VOLUME III</td>
<td>681002</td>
</tr>
<tr>
<td>CONTROL DATA 6400/6410/6420 COMPUTER SYSTEMS COREFERENCE MANUAL PUBLICATION NUMBER 60191200</td>
<td>690901</td>
</tr>
<tr>
<td>CONTROL DATA 6400/6420/6460 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60196900</td>
<td>6910C1</td>
</tr>
<tr>
<td>CONTROL DATA 6400/6420/6400 COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 62174900</td>
<td>6911C3</td>
</tr>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARD CONDOL PROGRAMMERS GUIDE NUMBERS 3605-C0-545</td>
<td>700602</td>
</tr>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM FORTRAN IV (C AND H) PROGRAMMERS GUIDE PROGRAM NUMBERS 3605-F0-57C 3605-F0-527</td>
<td>700603</td>
</tr>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM ASSEMBLER PROGRAMMERS GUIDE PROGRAM NUMBER 3605-AS-037</td>
<td>7110C3</td>
</tr>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM PL/1(P) PROGRAMMERS GUIDE PROGRAM NUMBER 3605-NL-911</td>
<td>7110C1</td>
</tr>
<tr>
<td>A SURVEY OF DATA STRUCTURES FOR COMPUTER GRAPHICS SYSTEMS</td>
<td>7110C1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROGRAMMING-TECHNIQUES</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLES, FLOW CHARTS, AND PROGRAM LOGIC</td>
</tr>
<tr>
<td>AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER</td>
</tr>
<tr>
<td>USE OF DECISION TABLES IN COMPUTER PROGRAMMING</td>
</tr>
<tr>
<td>CONVERSION OF LIMITED ENTRY DECISION TABLES TO COMPUTER PROGRAMS</td>
</tr>
</tbody>
</table>
DATA, DOCUMENTATION, AND DECISION TABLES 660101
PROGRAMMING DECISION TABLES IN FORTRAN, CUMIL OR ALGOL 660102
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS I: MINIMUM AVERAGE PROCESSING TIME 660901
CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS BY RULE MASK TECHNIQUES 661101
DECISION TABLES 670802
PARSING OF DECISION TABLES 670803
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II: MINIMUM STORAGE REQUIREMENT 671102
COMPUTERIZING A GOVERNMENT DATA SYSTEM: A MANAGEMENT OVERVIEW OF THE STEPS REQUIRED AND THE TIME NEEDED, 671202
SIMULATION WITH DECISION TABLES 680101
AUTOMATION IN LINGUISTICS 680103
PROGRAM DOCUMENTATION: THE VALUABLE BURDEN 680501
THE USE OF DECISION TABLES WITHIN SYSTEMATICS 680802
AMBIGUITY IN LIMITED ENTRY DECISION TABLES 681001
THE INTERPRETATION OF LIMITED ENTRY DECISION TABLE FORMAT AND RELATIONSHIPS AMONG CONDITIONS 691101
ON THE CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS 700301
DECISION TABLE TRANSLATION 700902
MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM 700904
AUTOFLOW ENHANCEMENTS OF DOCUMENTATION AND MAINTENANCE OF SCIENTIFIC APPLICATIONS 701101
AUTOMATIC PROGRAM ANNOTATION (AUTONOTE) 701104
AUTOMATED DOCUMENTATION OF AN ASSEMBLY 701105

PROGRAM DOCUMENTATION EXPERIENCE 701110
THE INTEGRATION OF SYSTEM SPECIFICATIONS AND PROGRAM CODING: A REPORT ON EXPERIENCE OF THE MEDIARS II PROJECT 701114
RESEARCH IN COMPUTER-ASSISTED DOCUMENTATION OF COMPUTER PROGRAMS 701202
MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM 710602

SLIP - SOURCE LIBRARY INQUIRY PROGRAM, NO. 3640-00-36-039 6A30C1
A CONVERSATIONAL CONTEXT-DIRECTED EDITOR 6933F1
CLOT - CARD LIBRARY ON TAPE 7036C4
WYLUR 703903
AUTOMATIC EDITING OF MANUALS 701112
ED PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE 710103
EDIT - KRONOS TXT EDITOR (EDIT) REFERENCE MANUAL 710104
TFCD - TXT EDITOR AND CORRECTOR - REFERENCE MANUAL 710603
CFMS - CARD FILE MAINTENANCE SYSTEM 7291C1
PLUS OFF, A DIRECT ACCESS SOFTWARE SECURITY AND CONTROL SYSTEM 7202C1
PANVALET - THE DIRECT ACCESS LIBRARY SYSTEM 720502
LIBRARIAN 723601
PROGRAM/MANAGE, A PROGRAM TO MAINTAIN SOURCE PROGRAM 7226C2
DATA MANIPULATION 72363
ON-LINE TEXT EDITING: A SURVEY 720604
PROJECT-MANAGEMENT
AUTOMATED WEATHER NETWORK 000101
EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV 000102
ON THE FEASIBILITY OF FORMAL CERTIFICATION 000103
DECISION TABLES FOR RISK MANAGEMENT SYSTEMS 660301
A SYSTEM OF PROGRAM DOCUMENTATION 660901
GOOD DECISION TABLES AND THEIR USES 680901
DOCUMENTATION STANDARDS 690103
DOCUMENTATION STANDARDS 700002
COSMIC PROGRAM DOCUMENTATION EXPERIENCE 701111
DEFINING ADEQUATE SYSTEMS DOCUMENTATION 701201
ACHIEVING PROPER PROGRAM DOCUMENTATION 711101

PROJECT-PLANNING
EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV 000102
A SYSTEM OF PROGRAM DOCUMENTATION 660901
GOOD DECISION TABLES AND THEIR USES 680901
DOCUMENTATION STANDARDS 690103
DOCUMENTATION: MOTIVATION AND TRAINING OR AUTOCALATION 701102
MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION A FEATURE OF TOTAL SYSTEM DESIGN 701103
DEFINING ADEQUATE SYSTEMS DOCUMENTATION 701201
ACHIEVING PROPER PROGRAM DOCUMENTATION 711101

REPRODUCTION
INDEX PREPARATION FOR PUBLICATIONS INDUSTRY 680902

SOFTWARE-COMPANIES
ON THE FEASIBILITY OF FORMAL CERTIFICATION 000103

SOFTWARE-COMPANIES
CONTINUATION
PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS 631001
AUTOCHART NEW METHOD CUTS MANUFACTURING 640501
AUTODESIGN COMPUTER-BASED ASSISTANCE FOR DOCUMENT PRODUCTION 670001
AN APPROACH TO SOFTWARE EVALUATION 670401
DOCUMENTATION SOFTWARE 670801
DECISION TABLES 672802
CONCIL FLOWCHARTING 680401
THE GROWTH OF COMPUTER GRAPHICS AT BELL LABORATORIES 680601
A CONVERSATIONAL CONTEXT-DIRECTED EDITOR 690201
A GUIDE FOR SOFTWARE DOCUMENTATION 700301
CLUT - CARD LIBRARY ON TAPE 700404
MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM 700504
AUTOMATED ENGINEERING DESIGN AN APPROACH TO AUTOMATED DOCUMENTATION 701108
PROGRAM AUTOMATED DOCUMENTATION METHODS 701109
COSMIC PROGRAM DOCUMENTATION EXPERIENCE 711111
UNIVAC 1100 SERIES OPERATING SYSTEM 712102
ED PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE 712103
EDIT - KRONOS TEXT EDITOR (EDIT) REFERENCE MANUAL 712104
PERSPECTIVE ON FLOWCHART PACKAGES 712301
AUTOMATIC SELECTIVE DOCUMENTATION SERVICES 712302
MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM 712602
TECO - TEXT EDITOR AND CORRECTOR - REFERENCE MANUAL 712603
SOFTWARE-COMPANIES
CONTINUATION
A LIBRARY OF DATA COLLECTION AND MANIPULATION SUBROUTINES 710902
CFMS - CARD FILE MAINTENANCE SYSTEM 720101
PLUS D/A, A DIRECT ACCESS SOFTWARE SECURITY AND CONTROL SYSTEM 720201
PANVALET - THE DIRECT ACCESS LIBRARY SYSTEM 720502
LIBRARIAN 720601
PROGRAM/ MANAGE, A PROGRAM TO MAINTAIN SOURCE PROGRAM 720602
DATA MANIPULATION 720603
SOURCE-PROGRAMS
CONTINUATION
TALKING DOWN A PROGRAM 720501
SOURCE-PROGRAMS
PROGRAMMING DECISION TABLES IN FORTRAN, COBOL OR ALGOL 660102
SLIP - SOURCE LIBRARY INQUIRY PROGRAM, NO. 3600-9C13-079 660601
S/360 ASSEMBLER/COMPILER LISTINGS LIBRARY MAINTENANCE, NO. 3609-00.0.015 681101
A CONVERSATIONAL CONTEXT-DIRECTED EDITOR 690301
CLOT - CARD LIBRARY ON TAPE 700604
WYRUR 700903
AUTOMATIC PROGRAM ANNOTATION (AUTO NOTE) 701104
RELLFLOW AUTOMATIC FLOWCHARTING SYSTEM 701105
AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION 701106
AUTOMATED DOCUMENTATION OF AN ASSEMBLY PROGRAM 701110
LN PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE 710103
FOIT - KRAMS TEXT EDITOR (EDIT) REFERENCE

SOURCE-PROGRAMS
CONTINUATION
MANUAL 710104
TECU - TEXT EDITOR AND CORRECTOR - REFERENCE MANUAL 710603
CFMS - CARD FILE MAINTENANCE SYSTEM 720101
PLUS D/A, A DIRECT ACCESS SOFTWARE SECURITY AND CONTROL SYSTEM 720201
PANVALET - THE DIRECT ACCESS LIBRARY SYSTEM 720502
LIBRARIAN 720601
PROGRAM/ MANAGE, A PROGRAM TO MAINTAIN SOURCE PROGRAM 720602
DATA MANIPULATION 720603
ON-LINE TEXT EDITING: A SURVEY 720604
STANDARDS
ON THE FEASIBILITY OF FORMAL CERTIFICATION FLOWCHARTS 600104
A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS 590701
PROGRAMMING DECISION TABLES IN FORTRAN, COBOL OR ALGOL 660102
PROGRAM DOCUMENTATION STANDARDS 660703
A SYSTEM OF PROGRAM DOCUMENTATION 663901
A PROGRAM FOR THE CONSTRUCTION OF PERT FLOWCHARTS 671101
USFRI'S MANUAL NMCS INFORMATION PROCESSING SYSTEM (NIPS) (IBM 1410/7801) 680102
NMCS INFORMATION PROCESSING SYSTEM IBM 1410/7801 GENERAL PURPOSE COMPONENTS MULTIFILE OUTPUT AND FILE GENERATION 685702
AN INTRODUCTION TO DECISION TABLES 690701
DOCUMENTATION STANDARDS 690103
ON-LINE MANAGEMENT SYSTEMS INFORMATION 690401
STANDARDS

CONTINUATION

DOCUMENTATION STANDARDS 700002
MILITARY STANDARDS 700003
THE DESIGN OF VISUAL DISPLAYS 700901
AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING 700905
AUTOFLOW ENHANCEMENTS OF DOCUMENTATION AND MAINTENANCE OF SCIENTIFIC APPLICATIONS 701101
DOCUMENTATION: MOTIVATION AND TRAINING OR AUTOMATION 701102
MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION A FEATURE OF TOTAL SYSTEM DESIGN 701103
BELLFLOW AUTOMATIC FLOWCHARTING SYSTEM 701105
PROGRAM AUTOMATED DOCUMENTATION METHODS 701109
DEFINING ADEQUATE SYSTEMS DOCUMENTATION 701201
HARD LOOK AT SOFTWARE DOCUMENTATION 710601
ACHIEVING PROPER PROGRAM DOCUMENTATION 711101
DOCUMENTATION STANDARDS 720102
STANDARDS GUIDE FOR SPACE AND EARTH SCIENCES COMPUTER SOFTWARE 720103
SUBPROGRAMS
COST ADVANTAGES OF AN INTEGRATED DOCUMENTATION APPROACH 701113
SUBROUTINE-LIBRARIES
IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LOADER PROGRAM NUMBERS 3605-ED-510 3605-ED-521 3605-LO-547 691102
SYMBOLS
PROGRAM DOCUMENTATION STANDARDS 669403
AN INTRODUCTION TO DECISION TABLES 680701
DOCUMENTATION STANDARDS 690103

SYMBOLS

CONTINUATION

DOCUMENTATION STANDARDS 700002
MILITARY STANDARDS 700003
SYNTAX-DIRECTED DOCUMENTATION FOR PL/360 700401
THE DESIGN OF VISUAL DISPLAYS 700901
AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING 700905
BELLFLOW AUTOMATIC FLOWCHARTING SYSTEM 701105
PROGRAM AUTOMATED DOCUMENTATION METHODS 701109
RESEARCH IN COMPUTER-ASSISTED DOCUMENTATION OF COMPUTER PROGRAMS 701202
A SURVEY OF DATA STRUCTURES FOR COMPUTER GRAPHICS SYSTEMS 710303
DOCUMENTATION STANDARDS 720102
STANDARDS GUIDE FOR SPACE AND EARTH SCIENCES COMPUTER SOFTWARE 720103
SYSTEM-DESIGN
PROBLEM LANGUAGE ANALYZER (PLAN) USERS' INTRODUCTION 690101
SYSTEMS-ANALYSIS
AUTOMATED WEATHER NETWORK 690101
CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 6010006 653701
A SYSTEM OF PROGRAM DOCUMENTATION 662901
BIBLIOGRAPHY ON INFORMATION SCIENCES VOLUME 1 680201
AMBIGUITY IN LIMITED ENTRY DECISION TABLES 681001
DOCUMENTATION STANDARDS 690103
DOCUMENTATION STANDARDS 700002
MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDIES 700801
TECHNICAL-WRITING

INDEX PREPARATION FOR PUBLICATIONS INDUSTRY 680202

TEXT

STUDY OF MACHINE-AIDED POST EDITING 680202
SYSTC/V360 ADMINISTRATIVE TERMINAL SYSTEM-OS 680303
INDEX PREPARATION FOR PUBLICATIONS INDUSTRY 680902
RUNOFF: A PROGRAM FOR THE PREPARATION OF DOCUMENTS 700301
WYLDUR 700903

AUTOFLOW ENHANCEMENTS OF DOCUMENTATION AND MAINTENANCE OF SCIENTIFIC APPLICATIONS 701101
AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION 701106
PROGRAM AUTOMATED DOCUMENTATION METHODS 701109
AUTOMATIC EDITING OF MANUALS 701112
ED PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE 710103
EDIT - KRONOS TEXT EDITOR (EDIT) REFERENCE MANUAL 710104
KRONOS TEXT EDITOR (EDIT) 710105
XEROX TEXTR LANGUAGE AND OPERATIONS REFERENCE MANUAL 720503
ON-LINE TEXT EDITING: A SURVEY 720604
FRESS USER'S GUIDE 720617
NETWORK INFORMATION CENTER USER GUIDE 720701
ASTROCOMP 720702

TIME-SHARING

INFORMATION SCIENCES: SOME RESEARCH DIRECTIVES 660401
THE JOSS PRIMER 670804

TIME-SHARING CONTINUATION

COMPUTER IN INFORMATION SCIENCES, VOLUME III 681002
RUNOFF: A PROGRAM FOR THE PREPARATION OF DOCUMENTS 700301
PROGRAM AUTOMATED DOCUMENTATION METHODS 701109
EDIT - KRONOS TEXT EDITOR (EDIT) REFERENCE MANUAL 710104
COMMUNICATIONS COMPUTER PROGRAMMING CENTER
AUTOMATED WEATHER NETWORK
AUTOMATED WEATHER DIVISION, CPC, TINKER AIR FORCE BASE,
This paper describes the automated weather network and
mentions some subroutines and input-output handling
methods used in the project. This is a very specialized
paper of little general interest.

CONKLIN, C.F.; MEADOW, C.T.
EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV.
IBM CENTER FOR EXPLORATORY STUDIES, ROCKVILLE, MD, 1968
This paper is a proposed system whose purpose is to acquire, store,
retrieve, publish, and disseminate all documentation, exclusive
of graphics for a large computer programming activity.
VOLUME I: SYSTEM DESCRIPTION.
VOLUME II: CONTROL AND USE OF THE SYSTEM.
VOLUME III: THE CANT EXECUTIVE LANGUAGE & INSTR GENERATOR.
VOLUME IV: PROGRAMMING SPECIFICATIONS.

KEITHFORD, R.F.; PARKER, D.R.
ON THE FEASIBILITY OF FORMAL CERTIFICATION
STANFORD RESEARCH INSTITUTE, MOUNTAIN VIEW, CALIFORNIA
This paper considers the feasibility of establishing
certification and validation services to ascertain the
reliability of software products.

CHAPIN, N.
FLOWCHARTS
A TWO-PART PUBLISHERS, PRINCETON, 1971
A comprehensive study of how graphic techniques can be used
to explain and interpret the complicated workings of a
computer. The book explains how to apply flowcharts to
all phases of data preparation and processing.

SCOTT, A.E.
AUTOMATIC PREPARATION OF FLOWCHART LISTING.
J. ACM 5, 1(JAN, 1958), 57-66
COVERS INITIAL MEANS OF AUTOMATING FLOWCHART PRODUCTION, A
METHOD FOR FLOWCHART PRODUCTION IS TO MAKE IT NECESSARY
to publish a detailed block diagrams. As a debugging tool,
it has great value.

HAIJO, L.H.
A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS
AFIPS WCPC PROC. (1959), 131-137.
GENERAL COVERS THE VARIOUS TYPES OF FLOWCHARTING.

CANTOR, H.N.; KING, F.E.
LOGIC STRUCTURE TABLES
COMM. ACM 8(11) (1965), 272-275
DESCRIPTS SET OF RULES FOR WRITING AND USING LOGIC TABLES.
LOGIC TABLES ARE TWO-DIMENSIONAL ENABLING ONE TO FULLY
EXPRESS BOTH SEQUENTIAL AND PARALLEL ASPECTS. THEY CAN BE
COMPILED DIRECTLY INTO A PROGRAM ELIMINATING NEED FOR
FLOWCHARTING AND HAND CODING.

MINTALBAND, M.
TABLES, FLOW CHARTS, AND PROGRAM LOGIC
IBM SYSTEMS J. 1,4(SEPT 1962), 51-63.
DECISION TABLES ARE INTRODUCED WITH REFERENCE TO BUSINESS
DATA PROCESSING. COMPLETENESS AND CONSISTENCY CHECKS ARE MADE. CONVERSION OF TABLES TO PROGRAMS IS
CONSIDERED AND A TECHNIQUE OF OBTAINING A PROGRAM WHICH
MINIMIZES BRANCHING REQUIREMENTS WITH RESPECT TO MEMORY
AND EXECUTION TIME. DEBUGGING AND OPTIMIZATION ARE ALSO
DISCUSSED.

KNUTH, M.E.
COMPUTER-DRAWN FLOWCHARTS.
COMM. ACM 6, 9(SEP, 1963), 555-563
SIMPLE SYSTEM FOR EFFECTIVE COMMUNICATION IS PRESENTED.
SUPPOSEDLY HELPS HET WITH JOVER DOCUMENTATION OF PROGRAMS
IN SIMPLE FORMAT, COMPUTER PREPARES FLOWCHARTS AND OTHER
CROSS-REFERENCE LISTING FROM THIS INPUT.

IBM CORPORATION
PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS.
COMPUTERS AND AUTOMATION 12, 10(OCT, 1963), 60.
A GENERAL SYSTEM OF FLOWCHARTING, THE COMPUTER PRINTS A
FLOWCHART.

SHMITT, D.T.; KAVANAGH, T.E.
USING DECISION STRUCTURE TABLES
DATA COMPREHENSION 10, 2(MARCH 1964), 6-8.
THIS ARTICLE EMPHASIZES MANUFACTURING APPLICATIONS OF
DECISION-TABLES. IT ALSO LISTS REASONS WHY DECISION
TABLES PAY OFF.

MECER, V.
AUTOCHART NEW METHOD CUTS MANUFACTURING.
COMMERCIAL PROGRAM PACKAGE CAN BE USED TO DECREASE THE COST
OF PRODUCING PROGRAMS.

KREIDER, L.
A PATH ANALYSIS ALGORITHM
J. ACM 11, 10(OCT, 1964), 420-436.
ALGEBRAIC REPRESENTATION OF THE FLOW OF A COMPUTER PROGRAM
IS DESIGNED. ALGORITHM PRESENTED CAN MANIPULATING THIS
REPRESENTATION INTO A FORM FROM WHICH A FLOWCHART CAN BE
DRAWN. ALSO, A PROCEDURE FOR FORMING THE ALGEBRAIC
REPRESENTATION FROM A COMPILER SOURCE CODE IS GIVEN.

ANDERSON, M.E.
AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER
COMM. ACM. 8, 1(JAN 1965), 36-39.
PROGRAM IS DESIGNED WHICH ACCEPTS FORTRAN II SOURCE PROGRAM
AS INPUT, PRODUCES ANNOTATED FLOWCHARTS AS OUTPUT. UNIONS
IN THAT A DIGITAL INCREMENTAL PLOTTING IS USED AS OUTPUT
MEDIUM, AND THAT ITS DESIGNED FOR A SMALL (40X) COMPUTER.

KIRK, M.M.
USE OF DECISION TABLES IN COMPUTER PROGRAMMING
COMM. ACM. 8, 1(JAN 1965), 41-43.
650102 ARTICLE DESCRIBES DECISION TABLE ADVANTAGES. A TECHNIQUE FOR CONVERSION OF TABLES TO PROGRAMS IS PRESENTED; IT IS THE RULE MASK TECHNIQUE. ADVANTAGES GAINED THROUGH UTILIZATION OF THE TECHNIQUE ARE ALSO DESCRIBED.

650701 CONTROL DATA CORPORATION
CONTROL DATA AEGO SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 60100000
CONTROL DATA CORPORATION
THIS MANUAL DESCRIBES THE SYSTEM CHARACTERISTICS OF THE AEGO SERIES. ALSO DISCUSSED ARE THE CENTRAL MEMORY, CENTRAL PROCESSOR, PERIPHERAL AND CONTROL PROCESSORS SYSTEM INTERRUPTS AND MANUAL CONTROL OF THE SYSTEM.

651001 SNYDER, A.
PROGRAMMING DOCUMENTATION
17TH ANNUAL TENNIS TOURNAMENT (OCTOBER 1965), 42-45
DOCUMENTATION IS A KEY PRODUCT OF A GOOD SYSTEMS JOB, BUT IT IS EVEN MORE IMPORTANT DURING COMPUTER SYSTEM CONVERSION. THIS ARTICLE LISTS WHAT GOES INTO AN EFFECTIVE JOB.

651101 POLLACK, S.L.
CONVERSION OF LIMITED ENTRY DECISION TABLES TO COMPUTER PROGRAMS
COMM. ACM 9, 11 (NOV 1966), 677-682
TWO ALGORITHMS ARE PRESENTED FOR CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS. BOTH ALGORITHMS PUNTO REDUNDANCIES AND CONTRADICTIONS IN THE TABLES. THE FIRST ALGORITHM MINIMIZES STORAGE SPACE; THE SECOND MINIMIZES RUN TIME.

660101 FISHER, D.L.
DATA, DOCUMENTATION, AND DECISION TABLES
COMM. ACM 9, 11 (JAN 1967), 26-31
IN THE FIELD OF DATA PROCESSING, IT IS NECESSARY TO BE ABLE TO DEFINE AND DOCUMENT DATA FILES, PROGRAMS AND DECISION RULES IN A WAY THAT REPRESENTS THEIR CHANGING INFORMATION FORMAT AND THEIR CONTINUOUS INTERACTION. TABLES MAKE THIS POSSIBLE. THIS ARTICLE IS BASICALLY A BUILDUP OF THE VIRTUES OF DECISION TABLES.

660102 VEINOTT, E.G.
PROGRAMMING DECISION TABLES IN FORTRAN, COBOL OR ALGOL
COMM. ACM 9, 11 (JAN 1967), 31-35
A BASED APPROACH FOR PROGRAMMING DECISION TABLES IN FORTRAN OR COBOL IS DEVELOPED. ONLY ONE OR TWO STATEMENTS ARE KEPT NECESSARY WITH INPUT IN A STANDARDIZED FORM. ACTUAL PROGRAMMING TECHNIQUES ARE OFFERED. THE TECHNIQUES ARE ALSO EXTENDED TO ALGOL.

660201 MARTIN, W.W.
FLOWCHARTING: SHORTHAND, ANALYSIS AND MODEL
JOURNAL OF SYSTEM MANAGEMENT 17, 2 (MARCH-APRIL 1966), 14-22

660301 SIDERY, J.A.
DECISION TABLES FOR BUSINESS MANAGEMENT SYSTEMS AND PROCEDURES J. 17, 2 (MARCH-APRIL 1966), 28-32

660401 THIS ARTICLE BEGINS WITH A LIST OF BENEFITS GAINED BY DECISION TABLE UTILIZATION; SOME EXAMPLES OF USAGE ARE PRESENTED. DECISION TABLES ARE DIRECTLY COMPARED TO FLOWCHARTS.

660403 HELL SYSTEMS PRACTICES
PROGRAM DOCUMENTATION STANDARDS
HELL SYSTEMS PRACTICES, AMERICAN TELEPHONE AND TELEGRAPH CO.
THIS IS A MANUAL WHICH HAS THE STANDARDS FOR THE HELL SYSTEM. IT INCLUDES STANDARDS FOR PROGRAM DOCUMENTATION, FLOWCHART SYMBOLS AND NOTATIONS, DECISION TABLES ARE INCLUDED AS WELL AS NUMEROUS EXAMPLES.

660404 PENFIELD, W.E.
SLIP-SURF SOURCE LIBRARY INQUIRY PROGRAM, NO. 3070-033600
IM CORPORATION, NEW YORK, NY, 1966
THIS PROGRAM STORES SYMBOLIC PROGRAM MODULES ON MAG TAPE AND RETRIEVES SELECTED MODULES TO BE LISTED, PUNCHED, OR REFERENCED TO ANOTHER MAG TAPE IN A FORMAT ACCEPTABLE TO THE ASSEMBLER. IT RUNS ON A 5/30/17 OR LAWDER, UNDER 9k BPS.

660501 REINWALD, L.T.
SOLAND, R.N.
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II: MINIMUM AVERAGE PROCESSING TIME.
ACM 13, 7 (JULY 1966), 337-35A
THIS ARTICLE PRESENTS A HIERARCHY DESCRIPTION OF DECISION TABLES AND THEN PRESENTS A DISCUSSION OF ALTERNATE EXPRESSIONS FOR THEM IN THE FORM OF SEQUENTIAL TESTING PROCEDURES. AN ALGORITHM IS DEVELOPED FOR CONVERSION OF TABLES TO A PROGRAM WITH MINIMUM AVERAGE PROCESSING TIME.

660901 HULP S.
ARTICLE RELATES WHY DOCUMENTATION SHOULD BE USED TO ACHIEVE EFFECTIVE COMPUTER UTILIZATION IN AN ORGANIZATION. A DESCRIPTION OF THE FACETS INVOLVED IN PROGRAM DOCUMENTATION IS ALSO PRESENTED, A MANUAL CONTAINING ALL DOCUMENTATION IS SUGGESTED.

661101 KING, P.J.H.
CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS BY RULE MASK TECHNIQUES
COMM. ACM 9,11 (NOV 1966), 796-801.

**ARTICLE PRESENTS A DESCRIPTION OF THE RULE MASK TECHNIQUE FOR CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS. THE TECHNIQUE, AS PRESENTED, MAY GIVE UNDESIRABLY LONG RUN TIMES. A MODIFICATION IS PROPOSED TO HELP ALLEVIATE THIS PROBLEM.**

SHERMAN, PHILIP M.

**FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS.**

COMM. ACM 9, 12 (DEC. 1966), 845-846.

FLOWTRACE PRODUCES FLOWCHARTS OF PROGRAMS WRITTEN IN ALMOST ANY PROGRAMMING LANGUAGE. META LANGUAGE AVAILABLE FOR DESCRIPTION OF SYNTAX OF CONTROL STATEMENTS.

CALLAHAM, M.D., GRACE, G.L.

**AUTORDOC: COMPUTER-BASED ASSISTANCE FOR DOCUMENT PRODUCTION**

PROC. ACM CONF 22 (1967), 177-185

AUTORDOC IS A MAN-MACHINE DESIGNED TO PROVIDE AUTOMATED DOCUMENTATION PLUS THIS SYSTEM PERMITS THE ADDED FEATURES OF DOCUMENT PRODUCTION, MAINTENANCE, AND BOOKKEEPING.

SPERRY RAND CORPORATION

UNIVAC 1101, COBOL UP-4040

UNIVAC CORPORATION

**THIS MANUAL EXPLAINS THE COBOL-LANGUAGE AS IT APPLIES TO THE UNIVAC 1101 COMPUTER.**

TUPAC, J.D.

**AN APPROACH TO SOFTWARE EVALUATION**

NTIS-AD651812

A PROCEDURE FOR SOFTWARE EVALUATION IS INTRODUCED. MAJOR IDEAS ARE WHEN AND HOW A MANAGER SHOULD EVALUATE TWO MAJOR AREAS OF SOFTWARE EVALUATION ARE FUNCTIONAL CAPABILITIES AND PERFORMANCE.

SALISBURY, A.B., ENSLOW, R.P.

**DIAGNOSTIC PROGRAMMING FOR DIGITAL COMPUTERS A BIBLIOGRAPHY**

NTIS-AD613831

THIS BIBLIOGRAPHY CONSISTS OF 88 REFERENCES ON DIAGNOSTIC PROGRAMMING FOR DIGITAL COMPUTERS. EACH ENTRY INCLUDES AUTHOR(S), TITLE, SOURCE, AN ABSTRACT (FORMALLY THE AUTHOR'S) AND USUALLY SOME COMMENTS PERTAINING TO ITEMS OF SPECIAL INTEREST OR ADDITIONAL INFORMATION TO HELP THE READER DECIDE IF THE DOCUMENT WOULD BE OF INTEREST TO HIM.

COMPRESS, INC.

**DOCUMENTATION SOFTWARE CONTESTATION 13, AUGUST 1967), 113-125**

UPIC (DOCUMENTATION OF PROGRAMS IN CORE) GENERATES MACHINE PRODUCED FLOWCHARTS, CROSS-REFERENCE LIST AND CHART INDEX FOR OBJECT PROGRAMS.

KING, P.J.H.

**DECISION TABLES**


THE ARTICLE DESCRIBES THE BASIC FEATURES OF DECISION TABLE DEVELOPMENT, AND IT ILLUSTRATES APPLICATIONS IN VARIOUS FIELDS. SOME PROGRAMMING TECHNIQUES ARE PRESENTED AN

EXTENSIVE DECISION TABLE RADIOLARIAN IS ALSO PRESENTED WITH EXPLANATION OF THE RELATIVE MERITS OF EACH.

**CAPPIN, N.**

**PARSING OF DECISION TABLES**


DEALS WITH REDUCTION OF DECISION TABLES BY SEVERAL TECHNIQUES. TECHNIQUES CONSIDERED ARE BY PARSING WITH REGARD TO VERTICAL AND HORIZONTAL DATE STRUCTURES, JOB IDENTIFICATION, HARDWARE AND JOB PRIORITY, AND CONFLICT RELATIONSHIPS. THE PARSING RELIES ON CONVENTIONS FOR DECISION TABLE LINKAGE.

MARKS, S.L.

**THE JSS PRIMER**

NTIS-AD659734

AN INTRODUCTION TO THE JSS TIME-SHARING SYSTEM. IT IS AN ON-LINE, TIME-SHARED COMPUTING SERVICE DEVELOPED AT THE RAND CORPORATION TO PROVIDE THE SOLUTION OF NUMERICAL PROBLEMS.

HANNOH, T.Y., AZZARI, A.L., BROOMES, A.L.

**STINERFORD, L., CHUJ, G.D.**

**DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES**

NTIS-AD646003

ESTABLISHES ALGORITHM FOR GENERATING A DECISION LOGIC-STRUCTURE OR TREE DIAGRAM WHICH PROCEDURAL TROUBLE SHOOTING INSTRUCTIONS COULD BE PREPARED.

HINNEN, W. E.

**A PROGRAM FOR THE CONSTRUCTION OF PERT FLOWCHARTS**

COMPUTER JOURNAL 10, 11 (MAY, 1967), 278-281

APPLICATIONS OF PERT AND PERT OFTEN INVOLVE FLOWCHARTS AND NETWORKS. PERT DIAGRAMS MUST BE EASY TO READ AND BE USEFUL. IN THIS ARTICLE, EVENTS AND ACTIVITIES THAT FORM A SUBNETWORK ARE GROUPED TOGETHER AND THE RELATIVITY OF ANY TWO NETWORKS REFLECTS THEIR INTERDEPENDENCE.

REINWALD, L.T.

**CONVERSION OF LIMITED-ENTRY DECISION TABLES II: OPTIMAL COMPUTER PROGRAMS II: MINIMUM STORAGEM REQUIREMENT**

J. ACM 14, 1 (JULY 1967), 742-756.

AN ALGORITHM IS DEVELOPED WHICH WILL CONVFRM ANY LIMITED ENTRY DECISION TABLE TO A COMPUTER PROGRAM WITH MINIMUM STORAGEM REQUIREMENT. IT IS INITIALLY DEVELOPED FOR PROGRAMS WITH A TREE-LIKE FORM AND THEN EXTENDED. THIS ALGORITHM CAN BE COMBINED WITH ONE FOR MINIMUM AVERAGE PROCESSING TIME, RESULTING IN A MINIMUM AVERAGE COST PROGRAM.

CIMPA, CTI. DEPT. UNIVERSITY OF ILLINOIS

**ILLIAK IV QUARTERLY PROGRESS REPORT**

NTIS-AD646720

A REPORT ON THE PROGRESS OF THE ILLIAK IV SYSTEM IS LISTED. THIS INCLUDES THE GRAPHICAL DISPLAY SYSTEM.

JONES, M.V.
671202

COMPUTERIZING A GOVERNMENT DATA SYSTEM: A MANAGEMENT OVERVIEW OF THE STEPS REQUIRED AND THE TIME NEEDED.
NTIS-AD62974

This paper discusses what must be done and how long it will take to computerize a manual data system in a government agency to a computerized system.

680101

LUDWIG, H.R.
SIMULATION WITH DECISION TABLES
J. DATA MGMT. 6,1(JAN 1968),20-27.
THE ARTICLE DEALS PRIMARILY WITH A SIMULATION PROCEDURE, USING THE DECISION TABLE STRUCTURE TO GOVERN THE BEHAVIOR OF THE OUTLINING ACTION. A SIMPLIFIED ALGORITHM IS PRESENTED TO SHOW SOMETHOW HOW THE DECISION TABLE STRUCTURE WAS USED.

680132

NATIONAL MILITARY COMMAND SYSTEM SUPPORT CENTER COMPUTER
IBM CORPORATION
USER'S MANUAL NMCS INFORMATION PROCESSING SYSTEM (NIPS) (IBM 1410/7010)
NTIS-AD81017
NMCS INFORMATION PROCESSING SYSTEM (IBM 1410/7010) IS DESCRIBED IN SIX DOCUMENTS: (1) SYSTEM DESCRIPTION, (2) OPERATOR'S MANUAL, (3) USER'S MANUAL, (4) ANALYTICAL MANUAL, (5) PROGRAM DESCRIPTION, AND (6) DATA BASE DOCUMENTATION.

680103

MEL'CHUK, I.A.
AUTOMATION IN LINGUISTICS
NTIS-AD342125
This article relates the problems of automation in linguistics, reviews and excerpts of translations by experts are discussed.

680201

DEFENSE DOCUMENTATION CENTER
BIBLIOGRAPHY ON INFORMATION SCIENCES. VOLUME I
DOC-AD829301
This bibliography is a compilation of references in the area of information science and technology. The references represent research on the basic activities common to most information-oriented missions.

680202

DEMUTH, N.
CONTU, E.
STUDY OF MACHINE-AIDED POST EDITING
DOC-AD829584
The primary objective of this report was to determine the feasibility of machine-aided post editing at FTD. From detection analysis was used to develop an experimental post-editing system (XPD).

680391

NATIONAL COMPUTER ANALYSIS, INC.
COROL FLOWCHARTING
DATAWATER 14, 3(MAR 1968), 135.
NATIONAL COMPUTER ANALYSIS, INC. PRINTS AND SALE'S PACK FLOWCHARTS ARE PRODUCED FOR AND COROL PROGRAM DECKS.

680392

IBM CORPORATION
NMCS INFORMATION PROCESSING SYSTEM IBM 1410/7010 GENERAL PURPOSE COMPONENTS MULTIFILE OUTPUT AND FILE GENERATION

680302

NMCS INFORMATION PROCESSING SYSTEM (IBM 1410/7010) IS DESCRIBED IN SIX DOCUMENTS: (1) SYSTEM DESCRIPTION, (2) OPERATOR'S MANUAL, (3) USER'S MANUAL, (4) ANALYTICAL MANUAL, (5) PROGRAM DESCRIPTION, AND (6) DATA BASE DOCUMENTATION.

680303

IBM CORPORATION
SYSTEM/360 ADMINISTRATIVE TERMINAL SYSTEM—OS
IBM CORPORATION
This system consists of control and functional programs that permit many different text-processing and data-handling activities to be carried on simultaneously through different terminals attached to an IBM SYSTEM/360.

680501

CHAMPIN, N.
PROGRAM DOCUMENTATION: THE VALUABLE BURDEN
SOFTWARE AGE, 21(FEB 1968), 24-26
ADVANTAGES AND OBJECTIVES OF DOCUMENTATION ARE DISCUSSED. TWELVE MAJOR DOCUMENTATION ELEMENTS ARE LISTED.

680601

NINKE, W.H.
THE GROWTH OF COMPUTER GRAPHICS AT RELL LABORATORIES
DELL LABORATORY RECORD 46, 6(JUNE 1968), 130
COMPUTER GRAPHICS IS A USEFUL TECHNIQUE FOR COMMUNICATING WITH COMPUTERS. THIS TECHNIQUE IS HELPING MAN MAKE MORE EFFECTIVE USE OF THE MACHINE TO SOLVE A wide VARIETY OF PROBLEMS WHILE PROVIDING A RECORD OF PROBLEM SOLUTION.

680701

FERGUS, R.M.
AN INTRODUCTION TO DECISION TABLES
SYSTEMS AND PROCEDURES J. 19(JULY-AUG 1968), 24-27
This article describes the characteristics of decision tables, also giving guidelines and examples of table construction as initial steps toward the use of tables to perform, communicate, and document decision making.

680801

O'BRIEN, F.
BECKETT, P.C.
A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION
COMPUTER JOURNAL 11, 9(AUG 1968), 130-143
AN APPROACH TO COMPUTER FLOWCHART GENERATION CALLED ICCOW PROGRAM IS SIMPLE AND EASY AND WHICH PRODUCE AN UNANNOTATED LINEAR FLOWCHART, THE FLOWCHARTS ANNOTATION HAS BEEN LEFT TO THE PROGRAMMER CONCERNED.

680802

GRINLEY, C.R.
THE USE OF DECISION TABLES WITHIN SYSTEMATICS
COMPUTER J. 11, 9(AUG 1968), 129-133
SYSTEMATICS IS A SET OF TECHNIQUES FOR DESIGNING AND DEVELOPING INFORMATION SYSTEMS. A GENERALIZED DESCRIPTION OF DECISION TABLES IS PRESENTED. THE FORM OF DECISION TABLE USED IN A SYSTEMATICS TECHNIQUE IS DESCRIBED ALONG WITH ADVANTAGES GAINED FROM IT.

680901

FERGUS, R.M.
GUIDED DECISION TABLES AND THEIR USES
SYSTEMS AND PROCEDURES J. 19, 5(SEPT-OCT 1968), 18-21
SEVEN STEPS ARE GIVEN FOR IMPLEMENTATION OF DECISION TABLE USE WITHIN AN ORGANIZATION. TABLES CAN BE USED AT MANY
LEVELS OF AN ORGANIZATION. TABLES CAN BE ANALYZED TO PROMOTE COMPLETENESS AND ACCURACY AND TO DETECT EXCESSIVE RULES, CONTRADICTIONS.

IBM CORPORATION
INDEX PREPARATION FOR PUBLICATIONS INDUSTRY
IBM CORPORATION J360D-29.4.0B
THIS PROGRAM PRODUCES AN INDEX IN DOUBLE-COLUMN FORMAT WHICH IS SUITABLE FOR REPRODUCTION BY PHOTO OFFSET PRINTING. THE PROGRAM WOULD BE OF VALUE TO ANY PUBLICATION DEPARTMENT THAT HAS TO PREPARE INDEXES AS A PART OF ITS PUBLICATIONS.

KING, P.J.H.
AMBIGUITY IN LIMITED ENTRY DECISION TABLES
COMM. ACM 11(1007) 68B, 68F.
AUTHOR CLAIMS RULES CONCERNING REDUNDANCY, CONTRADICTION AND COMPLETENESS AS ESTABLISHED IN 1963 ARE UNSATISFACTORY. AMBIGUITY IS CLAIMED TO BE AN IMPORTANT ASPECT IN CHECKING TABLES. PROCEDURE FOR PRODUCING CHECKED OUT TABLES IS PROPOSED. IMPORTANCE OF WELL-DESIGNED DIAGNOSTIC FACILITIES IS EMPHASIZED.

DEFENSE DOCUMENTATION CENTER
COMPUTER IN INFORMATION SCIENCES: VOLUME III
DC-ADAR42620
THIS BIBLIOGRAPHY COMPIL ES REFERENCES DEALING SPECIFICALLY WITH THE ROLE OF COMPUTERS IN INFORMATION SCIENCES.

MALTBY, Q. J.
CONTROL SECTION CROSS-REFERENCE AND LINK EDIT EDITOR
IBM CONTRIBUTED PROGRAM LIBRARY
THESE TWO ASSEMBLED PROGRAMS MAY PROVE USEFUL IN ANALYSIS AND DEBUGGING WORK WITH RESPECT TO LARGE AND COMPLEX PLANNED OVERLAY MODULES.

RISER, M. H.
LOBACK, L. R.
5/36 ASSEMBLER/COMPILER LISTINGS LIBRARY MAINTENANCE
NO. 3540-00-0.015
IBM CORPORATION, HAWTHORN, NY, 1968
THIS PROGRAM CREATES AND MAINTAINS A LIBRARY OF CURRENT DOS ASSEMBLER AND/C OR COMPILER (FORTAN, PAG, COOL, PL/1) OUTPUT LISTINGS ON IBM 2311 OR 2314 CDS. USING A SPACE COMPRESSION TECHNIQUE, IT RUNS ON ANY 5/360 UNDER DCS, SUPPORTING DM (DIRECT ACCESS METHOD), REQUIRING AT LEAST 16K OF STORAGE.

IBM CORPORATION
PROBLEM LANGUAGE ANALYZER (PLAN) USERS' INTRODUCTION
IBM CORPORATION, WHITE PLAINS, NY, 1969
THE IBM PROBLEM LANGUAGE ANALYZER (PLAN) WAS DESIGNED TO LOWER THE COST OF DEFINING, IMPLEMENTING, AND USING PROGRAM-ORIENTED LANGUAGES. PLAN IS A SET OF PROGRAMS THAT OPERATES ON THREE LEVELS TO SUPPORT LEVEL TO HELP THE PLAN PROGRAMMER PRODUCE NEW MODULES TO GIVE APPLICATION OF SIGNERS STANDARDS COMMANDS FOR CATALOGING THE SEMANTICS OF A NEW DL TO ACCEPT INPUT STATEMENTS, EXECUTE AND PRODUCE RESULTS.

IBM CORPORATION
SYSTEM 360 FLOWCHART USER'S MANUAL
IBM CORPORATION, WHITE PLAINS, NY, 1969
THE IBM SYSTEM 360 FLOWCHART MANUFACTURER'S MANUAL FLOWCHARTS UNDER THE IBM DOS. THIS DOCUMENTATION IS INTENDED TO MINIMIZE THE PLANNING AND EFFORT REQUIRED TO PRODUCE AND MAINTAIN PROGRAM DOCUMENTATION.

GRAY, M.
LONDON, K.P.
DOCUMENTATION STANDARDS
IBM SYSTEMS PRESS, PRINCETON, 1969
THIS BOOK GIVES AN UPDATED OVERVIEW OF WHAT DOCUMENTATION STANDARDS SHOULD BE. INCLUDED ARE SYSTEMS DEVELOPMENT DOCUMENTATION AND PROGRAM DOCUMENTATION FROM A DATA PROCESSING VIEWPOINT. NUMEROS FORMS AND EXAMPLES ARE SHOWN. FLOWCHARTS, DECISION TABLE AND SYMBOL STANDARDS (NATIONAL) AND RELATED ARE INCLUDED IN THE APPENDICES.

IBM CORPORATION
A CONVERSATIONAL CONTEXT-DIRECTED EDITOR
IBM CAMBRIDGE SCIENTIFIC CENTER, CAMBRIDGE, MA (1969)
THIS ON-LINE EDITOR OPERATES UNDER THE CP-67/C SYSTEMS. IT ALLOWS EDITING OF FILES SMALLER THAN AVAILABLE CORE SIZE ONLY BY BRINGING IN A BLOCK AT A TIME FOR EDITING, WITH NO BACKWARD BLOCK ACCESS ALLOWED.

READ, N.J.
ON-LINE MANAGEMENT SYSTEMS INFORMATION
DATAMATION 10(10) (APRIL 1969), 39
DISCUSSES THE NEED FOR PROGRAMMING STANDARDS AND A FRAMEWORK OF DOCUMENTATION WHICH HELPS DETERMINE THE SUCCESS OF THE EFFORT IN COMPLETING THESE STANDARDS.

CONTROL DATA CORPORATION
CONTROL DATA 6400/6500/7600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 601312CC

CONTROL DATA CORPORATION
CONTROL DATA 6400/6500/7600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60129000

CONTROL DATA CORPORATION
CONTROL DATA 6400/6500/7600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60197000

CONTROL DATA CORPORATION
CONTROL DATA 6400/6500/7600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60197000

CONTROL DATA CORPORATION
THE INTERPRETATION OF LIMITED ENTRY DECISION TABLES AND RELATIONSHIPS AMONG CONDITIONS
MICROPRO. J. 17, 111-121 (1969), 326-326
THIS ARTICLE EMPHASIZES THE INTERPRETATION OF BASIC FORMAT, RELATIONSHIPS AMONG CONDITIONS ARE SHOWN. A SEPARATE SET OF BASIC FORMAT, NRAL DEFINITIONS ARE PROPOSED.

IBM CORPORATION
THE INTERPRETATION OF LIMITED ENTRY DECISION TABLES AND RELATIONSHIPS AMONG CONDITIONS
MICROCOMPUTER J. 17, 111-121 (1969), 326-326
THIS ARTICLE EMPHASIZES THE INTERPRETATION OF BASIC FORMAT, RELATIONSHIPS AMONG CONDITIONS ARE SHOWN. A SEPARATE SET OF BASIC FORMAT, NRAL DEFINITIONS ARE PROPOSED.
70021
LCH-COST MICROFICHE STORAGE AND RETRIEVAL SYSTEM, DESIGN
OBJECTIVES AND SPECIFICATIONS FOR TWO OPTIMUM STORAGE AND
RETRIEVAL DEVICES ARE PRESENTED.

70041
VIRTUE, A.C.G.
THE DESIGN OF VISUAL DISPLAYS
BELL LABORATORIES REPORT 48, 81SEP70, 226.
VISUAL DISPLAYS APPEAL TO THE ECONOMY OF CRT DISPLAYS FOR
MAN-MACHINE INTERPLAY, COMMUNICATION CAPABILITY IS INCREDIBLY
INCREASED THROUGH CAREFULLY CHOOSING THE CHARACTERISTICS OF THE
DISPLAY SYMBOLS.

70052
DIAK, R.B.
DECISION TABLE TRANSLATION
COMM. ACM 13, 9SEP70, 571-573.
THE AUTHOR PRESENTS AN ALGORITHM FOR CONVERSION OF A LIMITED
ENTRY DECISION TABLE INTO A MACHINE PROCESSABLE
TEXT-AND-BRANCH MATRIX. HE POINTS OUT THE RELATION OF THIS
ALGORITHM TO OTHERS IN THE LITERATURE OF SIMILAR DESIGN

70070
STANFORD UNIVERSITY COMPUTATION CENTER
WYLUR
STANFORD UNIVERSITY COMPUTATION CENTER, STANFORD, CA (1970)
THIS PACKAGE CONSISTS OF AN MV'T MILLEN/WYLUR OPERATOR'S
GUIDE EMISNIM--REALLY A USER REFERENCE MANUAL. WYLUR
IS AN ON-LINE EDITOR AND PROMOTE JOB ENTRY AND RETRIEVAL
SYSTEM USING SELECTIC TYPEWRITER TERMINALS IN EITHER AN
MAT OR MV'T ENVIRONMENT. WYLUR'S PROGRAM EDITING CAPABILITIES
ARE TREATED AS A SUBSET OF ITS FREE-FORM TEXT EDITING
CAPABILITIES. THE COVER LETTER WAS PRODUCED BY WYLUR

70094
FELDMANN, C.G.
MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM
AEC-ADM-70-8211
THE EDE SYSTEM CAN BE INCREASED (SEE EDE-75-495 FOR DESCRIPTION)

70115
AMERICAN NATIONAL STANDARDS INSTITUTE, INC.
AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE
IN INFORMATION PROCESSING
AMERICAN NATIONAL STANDARDS INSTITUTE INC, X395-1970, NY, NY,
1970
THIS DOCUMENT GIVES A DETAILED DESCRIPTION OF FLOWCHART
SYMBOL AND DECISION TABLE STANDARDS ACCEPTED BY ANSI.

71110
GUTZ, M.A.
AUTOMATIC ENHANCEMENTS TO DOCUMENTATION AND MAINTENANCE OF
SCIENTIFIC APPLICATIONS
AUTOMATIC METHODS OF COMPUTER PROGRAM DOCUMENTATION,
(1970)
THE AUTHOR PREPARES AND DISCUSSES THE FOLLWING THREE
ELEMENTS WHICH HE FEELS ARE ESSENTIAL TO PROVIDING A
INTEGRATED DOCUMENTATION SYSTEM: 1)LOGICAL ANALYSIS OR
GRAPHIC DISSECTION OF A PROGRAM(AUTFLOW), 2)HISTORY AND
CONTROL OF PROGRAM(LIBRARIAN), 3)UNDERSTANDING THE PROGRAM
(TEXT EDIT PROGRAM).

71116
HEDKIN, M.A.
DOCUMENTATION: MOTIVATION AND TRAINING OR AUTOMATION

71119
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION,
(1970)
THE AUTHOR DESCRIBES WHAT MANUALS NEED TO BE DEVELOPED WHILE
DESIGNING A SYSTEM, BY WHOSE AND FOR WHOM THEY ARE DESIGNED.
AND EMPHASIZES THE NEED FOR DEVELOPING SYSTEM DOCUMENTS
BEFORE SYSTEM IS FINISHED.

71110
WOLF, A.W.
MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION A FEATURE OF
TITAN SYSTEM DESIGN
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION,
(1970)
The author notes that documents are often the afterthought of
designers. The author describes the design of a new software
system by AFSC for the air force, and the problems of
documentation. Of special interest is the discussion of the
data base description called 'COMPOOL' for common pool
of information.

71114
NEELY, M.D.
TYSON, J.W.
AUTOMATIC PROGRAM ANNOTATION (AUNET)
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION,
(1970)
THIS PAPER DESCRIBES A HYPOTHETICAL AUTOMATIC DOCUMENTATION
SYSTEM BASED ON SUPPLEMENTING RATHER THAN REPLACING
THE PROGRAMMER'S COMMENTS IN THE SOURCE DECK. IT IS VERY
DETAILED.

71115
PAPF, S.
BELLFLOW AUTOMATIC FLOWCHARTING SYSTEM
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION,
(1970)
THE BELLFLOW SYSTEM IS DISCUSSED WITH RESPECT TO THE THREE
METHODS OF OPERATION. THEY ARE CALLED THE SOURCE CODE,
COMMENT MODE, AND MIXED MODE. THE BELLFLOW SYSTEM ALLOWS
FOR USEFUL AND MEANINGFUL COMMENTS TO DERIVE A AUTOMATIC
FLOWCHART. BELLFLOW IS A TABLE-DRIVEN SYSTEM.

71116
HANNFY, M.
AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION,
(1970)
THE AUTHOR DEVELOPED A DOCUMENTATION PROGRAM WITH EMPHASIS
PLACED ON TEXT CONTENT RATHER THAN FLOWCHARTING. THE
PROGRAM GENERATES THE ENTIRE DOCUMENT AND IT IS KEYWORD
ORIENTED.

71117
GRUENEME, M.H.
VIEWS ON COMPUTER PROGRAM DOCUMENTATION AND AUTOMATION
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION,
(1970)
DR. GRUENEME EXPRESSES SOME BROAD OPINIONS AND PHILOSOPHY
ABOUT THE PROBLEMS OF DOCUMENTATION AND DESCRIPTION. HE
CAUTIONS ABOUT THE SEMANTICS ON THE SUBJECT OF
DOCUMENTATION. SOME ESTIMATED COSTS OF COMPUTING AND DATA
PROCESSING IN THE FEDERAL ESTABLISHMENT ARE ALSO GIVEN.

71118
MCCLURE, C.W.

AUTOMATED ENGINEERING DESIGN AN APPROACH TO AUTOMATED DOCUMENTATION
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION.
(NOV 1970)
THE AUTOMATED ENGINEERING DESIGN SYSTEM IS A SYSTEM OF COMPUTER PROGRAMS DESIGNED FOR USE IN BUILDING SOFTWARE SYSTEMS. DOCUMENTATION (ANYTHING THAT HELPS A PERSON UNDERSTAND A PROGRAM) IS A NECESSARY PART OF SOFTWARE DEVELOPMENT BECAUSE THE SOFTWARE WILL INEVITABLY UNDERGO CHANGES OVER THE YEARS.

LANZANO, D.C.
PROGRAM AUTOMATED DOCUMENTATION METHODS
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION.
(NOV 1970)
THIS PAPER PRESENTS A DESCRIPTION OF A PROGRAM WHICH ASSISTS IN AUTOMATING THE DOCUMENTATION OF SUBROUTINES (I.E., AN EXPOSE OF TWO FLOWCHARTING PROGRAMS AUTOFLOW, FLOWGEN). SOME NOTES ON USEFUL PROGRAM INTERNAL CROSS-REFERENCE INFORMATION, A TEXT-EDITING PROGRAM FOR A TIME-SHAPED ENVIRONMENT, A SYSTEM TO AID PROGRAM DOCUMENTATION UTILIZING A GRAPHICS DISPLAY CONSOLE.

THOMAS, V.L.
AUTOMATED DOCUMENTATION OF AN ASSEMBLY PROGRAM
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION.
(NOV 1970)
THIS PAPER IS AN AUTOMATIC DOCUMENTATION PROPOSAL WHICH EXTRACTS COMMENTS FROM CODE AND OUTPUTS IT IN AN ORDERLY LIST SEPARATE FROM THE LISTING.

KILAF, W.C.
JUICE: PROGRAM DOCUMENTATION EXPERIENCE
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION.
(NOV 1970)
THE PAPER DESCRIBES THE MINIMUM DOCUMENTATION EFFORT REQUIRED IN ORDER TO DISSEMINATE PROGRAMS TO USERS IN A USEFUL FORM.

RICH, R.P.
AUTOMATIC EDITING OF MANUALS
REFLECTIVELY DISCUSSIONS EDITING SYSTEM (INFIO 360) AND PROVIDES AN OVERVIEW OF IT. ALSO PROPOSES A METHOD OF DOCUMENTATION CALLED "MOD DOCUMENTATION" WHICH IS A "SUPER-LISTING" OF A PROGRAM. THIS METHOD WOULD HAVE ALL RELEVANT DOCUMENTATION NEEDED FOR DESCRIBING THE PROGRAM MAINTAINED WITH IT. WHEN THE PROGRAM IS UPDATED OR CHANGED THE APPROPRIATE DOCUMENTATION IS UPDATED OR CHANGED ALSO.

FELSMA, R.C.
COST ADVANTAGES OF AN INTEGRATED DOCUMENTATION APPROACH
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION.
(NOV 1970)
THE USE OF METAPROGRAMS IS ADVOCATED TO REDUCE PROGRESS TIME AND ASSOCIATED DOCUMENTATION PREPARATION, GENERIC PROGRAMS SHOULD BE WRITTEN WHICH CAN BE TAILORIED BY MEANS OF A METAPROGRAM WHICH PROCESSES THE PARAPERS TO PERFORM SPECIFIC FUNCTION. DOCUMENTATION COST CAN BE FURTHER REDUCED THROUGH THE USE OF SMALL PREDOCUMENTED SUPROGRAMS.

LUEFKE, W.R.
THE INTEGRATION OF SYSTEM SPECIFICATIONS AND PROGRAM COSING
A REPORT ON EXPERIENCE OF THE MEDLARS IT PROJECT
AUTOMATED METHODS OF COMPUTER PROGRAM DOCUMENTATION
(NOV 1970)
DESIGNING CSS EXPERIENCE IN MAINTAINING UP-TO-DATE DOCUMENTATION FOR ONE MODULE OF A VERY LARGE SCALE PROJECT. SEDLAPS, SEVERAL INNOVATIVE TECHNIQUES HAVE BEEN EXPLORED IN THE CONTENT OF MEDLARS' DATA MANAGEMENT ENVIRONMENT USING PL/1 AS AN AUTOMATIC DOCUMENTATION. PL/1'S DATA DESCRIPTION PROVIDES AUTOMATIC DOCUMENTATION USING MASTER DESCRIPTION OF DATA ELEMENTS, WITH MEANINGFUL NAMES.

MENKUS, B.
DEFINING ADEQUATE SYSTEMS DOCUMENTATION
AUTHOR DEFINES SYSTEMS DOCUMENTATION AND ENUMERATES FIVE FUNCTIONS PERFORMED BY ADEQUATE SYSTEMS DOCUMENTATION. HE ALSO LIST SEVERAL IMPORTANT GENERAL CONSIDERATIONS ABOUT DOCUMENTATION, CATEGORIZES TWO AREAS, SYSTEMS DEVELOPMENT DOCUMENTATION, AND SYSTEMS OPERATIONAL DOCUMENTATION UNDER THE GENERAL headings OF SYSTEMS DOCUMENTATION, ALSO CONSIDERS STANDARDS.

BERKLEY, E.C.
RESEARCH IN COMPUTER-ASSISTED DOCUMENTATION OF COMPUTER PROGRAMS
NTIS-AN711451
DISCUSSES THE PROBLEM OF DOCUMENTATION OF COMPUTER PROGRAMS, PRESENTS A FURTHER MODEL OF A "SIMULATING-ANALYZER" COMPUTER PROGRAM. A COMPLETE COMPUTER PROGRAM FOR AN SA IS ALSO PRESENTED. HE DISCUSSES COMMENTS AND SYMBOIC SYMBOIS IN COMPUTER PROGRAMS.

IBM CORPORATION
IBM SYSTEMS/360 OPERATING SYSTEM ASSEMBLER PROGRAMS
GUIDE PROGRAM NUMBER 360L-45-37
IBM CORPORATION
THIS MANUAL DISCUSSES PROGRAM ASSEMBLING, LINKAGE EDITING, EXECUTING, INTERPRETING LISTINGS, ASSEMBLER PROGRAMMING CONSIDERATIONS, DIAGNOSTIC MESSAGES, AND OBJECT OUTPUT CARDS.

IBM CORPORATION
IBM SYSTEMS/360 OPERATING SYSTEM PL/1(E) PROGRAMS
GUIDE PROGRAM NUMBER 360S-5L-411
IBM CORPORATION
THIS MANUAL EXPLAINS HOW TO USE 05/390 TO COMPLETE LINK-EDIT AND EXECUTE PROGRAMS WRITTEN IN IBM'S VERSION OF PL/1.

SPERRY RAND CORPORATION
UNITIAC 110 SERIES OPERATING SYSTEM
UNITIAC 110 SERIES OPERATING SYSTEM NUMBER 4144 REV 2
THIS MANUAL DISCUSSES THE BASE PORTION OF THE OPERATING SYSTEM EXEC #1 AND THE ASSOCIATED SOFTWARE NECESSARY TO CONSTRUCT, EXECUTE, AND MAINTAIN USER PROGRAMS.
This article discusses the advantages of on-line editing of both computer programs and free-form text, the distinguishing characteristics of two types of on-line editors, and the capabilities of several working systems of both types. A rather comprehensive bibliography follows the article.

COMPRESS, INC.

COMPRESS

automatically documents COBOL and assembly source programs. It can also be used with a special design language to help in program development. Comchart produces a detailed flowchart and a number of cross-references.

MANTECH CORPORATION

MANTECH CORPORATION

Superfex automatically produces a cross-reference listing of Fortran source programs. No flowcharts are given.

DATA FIP MANAGEMENT DECISION

FJTRAN variable name documenter

DATA FIP MANAGEMENT DECISION

produces a source listing and alphabetical list of referenced variable names.

FMA SYSTEMS, INC.

FIRE FLO

FJTRAN automatically documents Fortran source programs. It produces detailed flowchart and a resequenced source listing.

CALCIMP

CALCIMP

FLWCHEN/F

FLWCHEN/F automatically produces a detailed flowchart from a Fortran source program. It produces the flowchart on a Calcimp plotter.

APPLICATIONS PROGRAMMING CO.

DYNACHART

APPLICATIONS PROGRAMMING CO.

DYNACHART automatically documents COBOL source programs. It produces a detailed flowchart and a cross-reference listing.

NATIONAL COMPUTER ANALYSIS, INC.

QUICK-DRAW

NATIONAL COMPUTER ANALYSTS

QUICK-DRAW automatically documents COBOL, FORTRAN, and assembly source programs. It produces a detailed flowchart, cross-references, and some special listings.

APPLIED DATA RESEARCH

AUTOFLOW

APPLIED DATA RESEARCH, INC.

AUTOMATICALLY DOCUMENTS COBOL, PL/I, ASSEMBLY, AND FORTRAN source programs. AUTOFLOW produces flowcharts of different levels of detail, cross-references, and many special listings.

BELOIT COMPUTER CENTER

AUTOCHEF

BELOIT COMPUTER CENTER

DOCUMENTS COBOL source programs. Produces a detailed flowchart and a cross-reference listing of all labels used in the program.

AIRRES, CORP.

AUTOMAP

AIRRES, CORP.

AUTOMATICALLY DOCUMENTS COBOL source programs. Produces a detailed flowchart, a diagnostic listing, an altered statement cross-reference, a record layout, and an unmatched label table.

DATA INSTRUMENTS CO.

DATA INSTRUMENTS CO.

AUTOMATIC DOCUMENTATION OF COBOL AND ASSEMBLY SOURCE PROGRAMS produces a detailed flowchart and cross-references. It can also optionally produce a logic chart.

SYSTONETICS CORP.

EFLFLOW

SYSTONETICS CORP.

AUTOMATIC DOCUMENTATION OF FORTRAN source programs produces a detailed flowchart and cross-references. EFLFLOW also can resequence source code for label numbers.

TEXT SYSTEMS, INC.

PRESS USER'S GUIDE

TEXT SYSTEMS, INC.

PRESS FILE RETRIEVAL AND EDITING SYSTEM IS A SOPHISTICATED AND COST EFFECTIVE TEXT MANIPULATION SYSTEM. IT IS A VASTLY ENHANCED, MULTICONSOLE PRODUCTION VERSION OF THE PROTOTYPE TYPE SYSTEM. IT IS AN "ON-LINE" SYSTEM USING IBM 2741 + 226 TERMINALS OR EQUIVALENT.

ARA NETWORK INFORMATION CENTER

NATIONAL INFORMATION CENTER USER GUIDE

ARA NETWORK INFORMATION CENTER, STANFORD RESEARCH INSTITUTE, MENLO PARK, CA, JULY, 1972

CONTENT: 1. SAMPLE MESSAGE SENDING SESSION. 2. NIC TNC USER GUIDE. 3. NIC JOURNAL SYSTEM USER GUIDE. 4. LOCATOR. 5. PHONE SERVICE. 6. FOLKLORE DESCRIBES HOW TO USE THE FACILITIES OF THE ARA NETWORK INFORMATION CENTER.

INFORMATION CONTROL SYSTEMS, INC.

ASTROCOMP

INFORMATION CONTROL SYSTEMS, INC.

THIS SYSTEM PROVIDES TEXT EDITING CAPABILITIES ON MAGNETIC TAPE. WIDE VARIETY OF OPTIONS ARE ALSO AVAILABLE.
BUSINESS-ORIENTED CONTINUATION

BUSINESS-ORIENTED FILES 711002

CAPD

CLCT - CARD LIBRARY ON TAPE 700604

CFMS - CAPD FILE MAINTENANCE SYSTEM 729101

CENTER

NETWORK INFORMATION CENTER USER GUIDE 720701

CERTIFICATION

UN THE FEASIBILITY OF FORMAL CERTIFICATION 000103

CFMS

CFMS - CARD FILE MAINTENANCE SYSTEM 720101

CHARTS

TABLES, FLOW CHARTS, AND PROGRAM LOGIC 620901

CLCT

CLCT - CARD LIBRARY ON TAPE 700604

COBOL

PROGRAMMING DECISION TABLES IN FORTRAN, COBOL AND ALGOL 660102

UNIVAC 1108 COBOL UP-4048 670101

COBOL FLOWCHARTING 680301

CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COBOL REFERENCE MANUAL PUBLICATION NUMBER 60191220 690901

ION SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARD COBOL PROGRAMMERS GUIDE NUMBERS 3600-C8-545 700602

COLLECTION

A LIBRARY OF DATA COLLECTION AND MANIPULATION SUBROUTINES 710902

CONCERT

CONCERT 720605

COMPASS

CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60190000 691001

COMPILER

S/360 ASSEMBLER/COMPILER LISTINGS LIBRARY MAINTENANCE, NO: 3600-006-0015 681101

COMPUTE

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO COMPUTER PROGRAMS: A PROPOSED MODIFICATION TO POLLACKS ALGORITHM 710201

COMPUTER

AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER 650101

USE OF DECISION TABLES IN COMPUTER PROGRAMMING 650102

CONTROL DATA 6000 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 601030X 653701

CONVERSION OF LIMITED ENTRY DECISION TABLES TO COMPUTER PROGRAMS 651101

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II: MINIMUM AVERAGE PROCESSING TIME 660301

CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS BY RULE MASK TECHNIQUES 661101

DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES 670901

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS III: MINIMUM STORAGE REQUIREMENT 671102

THE GROWTH OF COMPUTER GRAPHICS AT AELL LABORATORIES 680601

A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION 680401

COMPUTER IN INFORMATION SCIENCES, VOLUME III 681302

CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COBOL REFERENCE MANUAL PUBLICATION
COMPUTER
CONTINUATION

NUMBER 66191200

CONTROL DATA 6600/6500/6600 COMPUTER
SYSTEMS COMPASS REFERENCE MANUAL
PUBLICATION NUMBER 6190400

CONTROL DATA 6600/6500/6600 COMPUTER
SYSTEMS FORTRAN REFERENCE MANUAL
PUBLICATION NUMBER 60174900

ON THE CONVERSION OF DECISION TABLES TO
COMPUTER PROGRAMS

AN INTERIM PROGRESS REPORT OF COMPUTER
OUTPUT MICROFILM ACTIVITIES AND
EXPERIENCES

MAKING AUTOMATED COMPUTER PROGRAM
DOCUMENTATION A FEATURE OF TOTAL SYSTEM
DESIGN

VIEWS ON COMPUTER PROGRAM DOCUMENTATION
AND AUTOMATION

RESEARCH IN COMPUTER-ASSISTED
DOCUMENTATION OF COMPUTER PROGRAMS

A SURVEY OF DATA STRUCTURES FOR COMPUTER
GRAPHICS SYSTEMS

STANDARDS GUIDE FOR SPACE AND EARTH
SCIENCES COMPUTER SOFTWARE

COMPUTER-ASSISTED
RESEARCH IN COMPUTER-ASSISTED
DOCUMENTATION OF COMPUTER PROGRAMS

COMPUTER-RASFC

AUTO001: COMPUTER-BASED ASSISTANCE FOR
DOCUMENT PRODUCTION

COMPUTER-DRAWN

COMPUTER-DRAWN FLOWCHARTS

COMPUTERIZING

COMPUTERIZING A GOVERNMENT DATA SYSTEM: A
MANAGEMENT OVERVIEW OF THE STEPS REQUIRED
AND THE TIME NEEDED.

COMPUTERS

DIAGNOSTIC PROGRAMMING FOR DIGITAL
COMPUTERS: A BIBLIOGRAPHY

CONSTRUCTION

A PROGRAM FOR THE CONSTRUCTION OF PERT
FLOWCHARTS

CONTEXT-DIRECTED

A CONVERSATIONAL CONTEXT-DIRECTED EDITOR

CONTROL

CONTROL DATA 6000 SERIES COMPUTER SYSTEMS
REFERENCE MANUAL NUMBER 6000000

CONTROL SECTION CROSS-REFERENCE AND LINK
EDIT EDITOR

CONTROL DATA 6400/6500/6600 COMPUTER
SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION
NUMBER 60191200

CONTROL DATA 6400/6500/6600 COMPUTER
SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION
NUMBER 60174900

CONTROL DATA 6400/6500/6600 COMPUTER
SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION
NUMBER 60174900

PLUS DATA, A DIRECT ACCESS SOFTWARE
SECURITY AND CONTROL SYSTEM

CONVERSATIONAL

A CONVERSATIONAL CONTEXT-DIRECTED EDITOR

CONVERSION

CONVERSION OF LIMITED ENTRY DECISION
TABLES TO COMPUTER PROGRAMS

CONVERSION OF LIMITED-ENTRY DECISION
TABLES TO OPTIMAL COMPUTER PROGRAMS I:
MINIMUM AVERAGE PROCESSING TIME

CONVERSION OF DECISION TABLES TO COMPUTER
PROGRAMS BY RULE MASK TECHNIQUES

CONVERSION OF LIMITED-ENTRY DECISION
TABLES TO OPTIMAL COMPUTER PROGRAMS II:
MINIMUM STORAGE REQUIREMENT
DRAW
A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS 590001
PROGRAM ENABLES IBM TC70/7074 TO DRAW
FLOWCHARTS. 631001
DYNACHART
DYNACHART 720610
EARTH
STANDARDS GUIDE FOR SPACE AND EARTH
SCIENCES COMPUTER SOFTWARE 720103
EDIT
CONTROL SECTION CROSS-REFERENCE AND LINK
EDIT EDITOR 681003
EDIT - KRONOS TEXT EDITOR (EDIT) REFERENCE
MANUAL 710104
KRONOS TEXT EDITOR (EDIT) 710105
EDITING
STUDY OF MACHINE-AIDED POST EDITING 680202
AUTOMATIC EDITING OF MANUALS 701112
ON-LINE TEXT EDITING: A SURVEY 720604
EDITOR
CONTROL SECTION CROSS-REFERENCE AND LINK
EDIT EDITOR 681003
A CONVERSATIONAL CONTEXT-DIRECTED EDITOR 690301
IBM SYSTEM/360 OPERATING SYSTEM LINKAGE
EDITOR AND LOADER PROGRAM NUMBERS 691102
3605-ED-515 3605-ED-521 3605-ED-547
EDIT - KRONOS TEXT EDITOR (EDIT) REFERENCE
MANUAL 710104
KRONOS TEXT EDITOR (EDIT) 710105
TECH - TEXT EDITOR AND CORRECTOR -
REFERENCE MANUAL 710603
ENGINEERING
AUTOMATED ENGINEERING DESIGN AN APPROACH 720616
ENGINEERING
CONTINUATION
TO AUTOMATED DOCUMENTATION 701108
ENHANCEMENT
MAINTENANCE AND ENHANCEMENT OF THE AED
SYSTEM 700904
MAINTENANCE AND ENHANCEMENT OF THE AED
SYSTEM 710602
ENHANCEMENTS
AUTOFLOW ENHANCEMENTS OF DOCUMENTATION AND
MAINTENANCE OF SCIENTIFIC APPLICATIONS 701101
ENTRY
CONVERSION OF LIMITED ENTRY DECISION
TABLES TO COMPUTER PROGRAMS 651101
AMBIGUITY IN LIMITED ENTRY DECISION TABLES 681201
THE INTERPRETATION OF LIMITED ENTRY
DECISION TABLE FORMAT AND RELATIONSHIPS
AMONG CONDITIONS 691101
EQUIPMENT
MICROFICHE, MICROFILM, AND RELATED
EQUIPMENT 711001
EVALUATION
AN APPROACH TO SOFTWARE EVALUATION 670401
EVOLUTIONARY
EVALUATION SYSTEM FOR DATA PROCESSING. VOLUMES I-IV. 600102
EXPERIENCE
COSMIC: PROGRAM DOCUMENTATION EXPERIENCE 701111
THE INTEGRATION OF SYSTEM SPECIFICATIONS
AND PROGRAM COSING. A REPORT ON
EXPERIENCE OF THE MEDLARS II PROJECT. 701114
EZFLOW
EZFLOW 720616
FEASIBILITY
ON THE FEASIBILITY OF FORMAL CERTIFICATION 000103

FEATURE
MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION A FEATURE OF TOTAL SYSTEM DESIGN 701103

FILE
ICS INFORMATION PROCESSING SYSTEM IBM 1410/7010 GENERAL PURPOSE COMPONENTS MULTIFILE OUTPUT AND FILE GENERATION 680302
CFMS - CARD FILE MAINTENANCE SYSTEM 720101

FILES
DATA COMPRESSION IN LARGE BUSINESS-ORIENTED FILES 711902

FLAG
AUTOMATIC DATA PROCESSING SUPPORT FOR FLAG PLOT 660402

FLOW
TABLES, FLOW CHARTS, AND PROGRAM LOGIC 620901
A FLOW ANALYSIS ALGORITHM 641001

FLOWCHART
AUTOMATIC PREPARATION OF FLOWCHART LISTING 580101
A TECHNIQUE FOR COMPUTER FLOWCHART GENERATION 680101
SYSEX/15 FLOWCHART USER'S MANUAL OPERATOR'S MANUAL 690102
AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING 700905
PERSPECTIVE ON FLOWCHART PACKAGES 710301

FLOWCHARTING
FLOWCHARTING: SHORTHAND, ANALYSIS AND MODEL 660201

FLOWCHARTING CONTINUATION
FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS 661201
COBOL FLOWCHARTING 650301
BELLFLOW AUTOMATIC FLOWCHARTING SYSTEM 721105

FLOWCHARTS
FLOWCHARTS
A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS 590301
COMPUTER-DRAWN FLOWCHARTS 630901
PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS 631001
AUTOMATED PLOTTING OF FLOWCHARTS ON A SMALL COMPUTER 650101
A PROGRAM FOR THE CONSTRUCTION OF PERT FLOWCHARTS 671101

FLOWGEN
FLOWGEN/F 725308

FLOWTRACE
FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS 661201

FORFLO
FORFLO 726608

FORMAL
ON THE FEASIBILITY OF FORMAL CERTIFICATION 000103

FORMAT
THE INTERPRETATION OF LIMITED ENTRY DECISION TABLE FORMAT AND RELATIONSHIPS AMONG CONDITIONS 681101

FURTRAN
PROGRAMMING DECISION TABLES IN FURTRAN, COBOL OR ALGOL 660102
CONTROL DATA 6400/6500/6600 COMPUTER

PAGE 15
<table>
<thead>
<tr>
<th>OPERATIONS</th>
<th>PAGE 27</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xerox Text Language and Operations Reference Manual</td>
<td>720503</td>
</tr>
<tr>
<td>OPERATORS</td>
<td></td>
</tr>
<tr>
<td>SYSTEM/360 Flowchart User's Manual</td>
<td>690102</td>
</tr>
<tr>
<td>OPTIMAL</td>
<td></td>
</tr>
<tr>
<td>Conversion of Limited-Entry Decision Tables to Optimal Computer Programs II: Minimum Average Processing Time</td>
<td>660001</td>
</tr>
<tr>
<td>Conversion of Limited-Entry Decision Tables to Optimal Computer Programs III: Minimum Storage Requirement</td>
<td>671102</td>
</tr>
<tr>
<td>OUTPUT</td>
<td></td>
</tr>
<tr>
<td>NMCIS Information Processing System IBM 1405/7C10 General Purpose Components Multifile Output and File Generation</td>
<td>680302</td>
</tr>
<tr>
<td>An Interim Progress Report of Computer Output Microfilm Activities and Experiences</td>
<td>700701</td>
</tr>
<tr>
<td>OVERLAY</td>
<td></td>
</tr>
<tr>
<td>OVRFLAY TREE PROCESSOR</td>
<td>700101</td>
</tr>
<tr>
<td>OVERVIEW</td>
<td></td>
</tr>
<tr>
<td>Computerizing a Government Data System: An Management Overview of the Steps Required and the Time Needed.</td>
<td>671202</td>
</tr>
<tr>
<td>PACKAGES</td>
<td></td>
</tr>
<tr>
<td>Perspective on Flowchart Packages</td>
<td>710301</td>
</tr>
<tr>
<td>PANVALET</td>
<td></td>
</tr>
<tr>
<td>PANVALET - The Direct Access Library System</td>
<td>720502</td>
</tr>
<tr>
<td>PARSE</td>
<td></td>
</tr>
<tr>
<td>Parsing of Decision Tables</td>
<td>670003</td>
</tr>
<tr>
<td>PERSPECTIVE</td>
<td></td>
</tr>
<tr>
<td>Perspective on Flowchart Packages</td>
<td>710301</td>
</tr>
<tr>
<td>PEKT</td>
<td></td>
</tr>
<tr>
<td>A Program for the Construction of PEKT Flowcharts</td>
<td>671101</td>
</tr>
<tr>
<td>PLOT</td>
<td></td>
</tr>
<tr>
<td>Automatic Data Processing Support for Flag Plot</td>
<td>660402</td>
</tr>
<tr>
<td>PLOTTING</td>
<td></td>
</tr>
<tr>
<td>Automated Plotting of Flowcharts on a Small Computer</td>
<td>650101</td>
</tr>
<tr>
<td>PLUS</td>
<td></td>
</tr>
<tr>
<td>PLUS D/A, A Direct Access Software Security and Control System</td>
<td>720201</td>
</tr>
<tr>
<td>PL360</td>
<td></td>
</tr>
<tr>
<td>Syntax-Directed Documentation for PL360</td>
<td>700401</td>
</tr>
<tr>
<td>PULLACKS</td>
<td></td>
</tr>
<tr>
<td>Conversion of Limited-Entry Decision Tables to Computer Programs A Proposed Modification to Pullacks Algorithm</td>
<td>712001</td>
</tr>
<tr>
<td>POST</td>
<td></td>
</tr>
<tr>
<td>Study of Machine-Aided Post Editing</td>
<td>680202</td>
</tr>
<tr>
<td>PREPARATION</td>
<td></td>
</tr>
<tr>
<td>Automatic Preparation of Flowchart Listing</td>
<td>600101</td>
</tr>
<tr>
<td>Index Preparation for Publications Industray</td>
<td>680902</td>
</tr>
<tr>
<td>RUNNIFE: A Program for the Preparation of Documents</td>
<td>700301</td>
</tr>
<tr>
<td>PRIMER</td>
<td></td>
</tr>
<tr>
<td>The Joss Primer</td>
<td>670804</td>
</tr>
<tr>
<td>PROCESSING</td>
<td></td>
</tr>
<tr>
<td>Evolutionary System for Data Processing, Volumes I-IV.</td>
<td>600102</td>
</tr>
<tr>
<td>Automatic Data Processing Support for Flag Plot</td>
<td>660402</td>
</tr>
<tr>
<td>PROCESSING CONTINUATION</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td></td>
</tr>
<tr>
<td>CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS I: MINIMUM AVERAGE PROCESSING TIME</td>
<td>640801</td>
</tr>
<tr>
<td>USER'S MANUAL CMS INFORMATION PROCESSING SYSTEM (HIPS) (IBM 1410/7010)</td>
<td>640102</td>
</tr>
<tr>
<td>CMS INFORMATION PROCESSING SYSTEM IBM 1410/7710 GENERAL PURPOSE COMPONENTS: MULTIFILIP OUTPUT AND FILE GENERATION</td>
<td>640302</td>
</tr>
<tr>
<td>AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING</td>
<td>700905</td>
</tr>
<tr>
<td>PROCESSOR</td>
<td></td>
</tr>
<tr>
<td>OVERLAY TREE PROCESSOR</td>
<td>700101</td>
</tr>
<tr>
<td>ED PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE</td>
<td>710103</td>
</tr>
<tr>
<td>PRODUCTION</td>
<td></td>
</tr>
<tr>
<td>AUTOGCO: COMPUTER-BASED ASSISTANCE FOR DOCUMENT PRODUCTION</td>
<td>670001</td>
</tr>
<tr>
<td>PROGRAM</td>
<td></td>
</tr>
<tr>
<td>A PROGRAM TO DRAW MULTILEVEL FLOWCHARTS</td>
<td>590901</td>
</tr>
<tr>
<td>TABLES, FLOW CHARTS, AND PROGRAM LOGIC</td>
<td>620901</td>
</tr>
<tr>
<td>PROGRAM ENABLES IBM 7070/7074 TO DRAW FLOWCHARTS</td>
<td>631001</td>
</tr>
<tr>
<td>PROGRAM DOCUMENTATION STANDARDS</td>
<td>660403</td>
</tr>
<tr>
<td>SLIP - SOURCE LIBRARY INQUIRY PROGRAM, NO. 3605-3603.039</td>
<td>660601</td>
</tr>
<tr>
<td>A SYSTEM OF PROGRAM DOCUMENTATION</td>
<td>660901</td>
</tr>
<tr>
<td>FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS</td>
<td>661201</td>
</tr>
<tr>
<td>DEVELOPMENT OF A COMPUTER PROGRAM FOR GENERATING TROUBLE-SHOOTING DECISION TREES</td>
<td>670901</td>
</tr>
<tr>
<td>A PROGRAM FOR THE CONSTRUCTION OF PERT FLOWCHARTS</td>
<td>671101</td>
</tr>
<tr>
<td>PROGRAM DOCUMENTATION: THE VALUABLE BURDEN</td>
<td>680501</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROGRAM CONTINUATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LOADER PROGRAM NUMBERS</td>
</tr>
<tr>
<td>RUNOFF: A PROGRAM FOR THE PREPARATION OF DOCUMENTS</td>
</tr>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM FORTRAN IV (G AND H) PROGRAMMERS GUIDE PROGRAM NUMBERS 3605-EO-500 3605-EO-520</td>
</tr>
<tr>
<td>MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION A FEATURE OF TOTAL SYSTEM DESIGN</td>
</tr>
<tr>
<td>AUTOMATIC PROGRAM ANNOTATION (AUTOCODE)</td>
</tr>
<tr>
<td>AN AUTOMATED SYSTEM FOR GENERATING PROGRAM DOCUMENTATION</td>
</tr>
<tr>
<td>VIEWS ON COMPUTER PROGRAM DOCUMENTATION AND AUTOMATION</td>
</tr>
<tr>
<td>PROGRAM AUTOMATED DOCUMENTATION METHODS</td>
</tr>
<tr>
<td>AUTOMATED DOCUMENTATION OF AN ASSEMBLY PROGRAM</td>
</tr>
<tr>
<td>COSMIC: PROGRAM DOCUMENTATION EXPERIENCE</td>
</tr>
<tr>
<td>THE INTEGRATION OF SYSTEM SPECIFICATIONS AND PROGRAM COATING, A REPORT ON EXPERIENCE OF THE HEIDELBERG 14 PROJECT</td>
</tr>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM ASSEMBLER PROGRAMMERS GUIDE PROGRAM NUMBER 360L-AS-737</td>
</tr>
<tr>
<td>IBM SYSTEM/360 OPERATING SYSTEM PL/I (P) PROGRAMMERS GUIDE PROGRAM NUMBER 3605-PL-511</td>
</tr>
<tr>
<td>ACHIEVING PROPER PROGRAM DOCUMENTATION</td>
</tr>
<tr>
<td>TALKING DOWN A PROGRAM</td>
</tr>
<tr>
<td>PROGRAM/MAKE, A PROGRAM TO MAINTAIN SOURCE PROGRAM</td>
</tr>
<tr>
<td>PROGRAMMER</td>
</tr>
<tr>
<td>ED PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE</td>
</tr>
</tbody>
</table>
PROGRAMMERS

IBM SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARD CODING PROGRAMMERS GUIDE NUMBERS 360S-CB-545 7C0602
IBM SYSTEM/360 OPERATING SYSTEM FORTRAN IV (G AND H) PROGRAMMERS GUIDE PROGRAM NUMBERS 360S-FO-500 360S-FO-520 7C0603
IBM SYSTEM/360 OPERATING SYSTEM ASSEMBLER F PROGRAMMERS GUIDE PROGRAM NUMBER 360L-AS-337 7C1203
IBM SYSTEM/360 OPERATING SYSTEM PL/I (P) PROGRAMMERS GUIDE PROGRAM NUMBER 360S-NL-511 7C1011

PROGRAMMING

USE OF DECISION TABLES IN COMPUTER PROGRAMMING 650102
PROGRAMMING DOCUMENTATION 651001
PROGRAMMING DECISION TABLES IN FORTRAN. COBL OR ALGOL 660102
DIAGNOSTIC PROGRAMMING FOR DIGITAL COMPUTERS A BIBLIOGRAPHY 670402

PROGRAMS

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO COMPUTER PROGRAMS 651101
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II: MINIMUM AVERAGE PROCESSING TIME 660801
CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS IV: RULE MASK TECHNIQUES 661101
FLOWTRACE, A PROGRAM FOR FLOWCHARTING PROGRAMS 661201
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS III: MINIMUM STORAGE REQUIREMENT 671102
IN THE CONVERSION OF DECISION TABLES TO COMPUTER PROGRAMS 7C0601
RESEARCH IN COMPUTER-ASSISTED DOCUMENTATION OF COMPUTER PROGRAMS 7C1202

PROGRAMS CONTINUATION

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO COMPUTER PROGRAMS A PROPOSED MODIFICATION TO POLLACKS ALGORITHM 7C1001

PROGRESS

ILLIAC IV QUARTERLY PROGRESS REPORT 671201
AN INTERIM PROGRESS REPORT OF COMPUTER OUTPUT MICROFILM ACTIVITIES AND EXPERIENCES 7C0701

PROJECT

THE INTEGRATION OF SYSTEM SPECIFICATIONS AND PROGRAM COSING. A REPORT ON EXPERIENCE ON THE MEDLARS II PROJECT 7C1114

PROPER

ACHIEVING PROPER PROGRAM DOCUMENTATION 7C11101

PROPOSED

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO COMPUTER PROGRAMS A PROPOSED MODIFICATION TO POLLACKS ALGORITHM 7C1201

PUBLICATION

CONTROL DATA 660C/650C/660C COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 6519200 690001
CONTROL DATA 660C/650C/660C COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 6519290 691001
CONTROL DATA 660C/650C/660C COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 6517490 691109

PUBLICATIONS

INDEX PREPARATION FOR PUBLICATIONS INDUSTRY 680902

QUARTERLY

ILLIAC IV QUARTERLY PROGRESS REPORT 671201
STEPS
COMPUTERIZING A GOVERNMENT DATA SYSTEM: A MANAGEMENT OVERVIEW OF THE STEPS REQUIRED AND THE TIME NEEDED 671202

STORAGE
CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS III: MINIMUM STORAGE REQUIREMENT 671102

MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDY 700801

STRUCTURE
LOGIC STRUCTURE TABLES 610601
USING DECISION STRUCTURE TABLES 640301

STRUCTURES
A SURVEY OF DATA STRUCTURES FOR COMPUTER GRAPHICS SYSTEMS 710303

STUDY
STUDY OF MACHINE-AIDED POST EDITING 680202
MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDY 700901

SUBROUTINES
A LIBRARY OF DATA COLLECTION AND MANIPULATION SUBROUTINES 710902

SUPERREF
SUPERREF 720606

SUPPORT
AUTOMATIC DATA PROCESSING SUPPORT FOR FLAG PLUT 660402

SURVEY
A SURVEY OF DATA STRUCTURES FOR COMPUTER GRAPHICS SYSTEMS 710303
ON-LINE TEXT EDITING: A SURVEY 720604

SYMBOLS
AMERICAN NATIONAL STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING 700905

SYNTAX-DIRECTED
SYNTAX-DIRECTED DOCUMENTATION FOR PL/360 700401

SYSTEM
EVOLUTIONARY SYSTEM FOR DATA PROCESSING, VOLUMES I-IV 602102
A SYSTEM OF PROGRAM DOCUMENTATION 660901
COMPUTERIZING A GOVERNMENT DATA SYSTEM: A MANAGEMENT OVERVIEW OF THE STEPS REQUIRED AND THE TIME NEEDED 671202
USER'S MANUAL NMC'S INFORMATION PROCESSING SYSTEM (NIPS) IBM 1417/7C191 680102
NMC'S INFORMATION PROCESSING SYSTEM 1417/7C19 GENERAL PURPOSE COMPONENTS MULTILIFT OUTPUT AND FILE GENERATION 680302
SYSTEM/360 ADMINISTRATIVE TERMINAL SYSTEM -- I 690903
SYSTEM/360 FLOWCHART USER'S MANUAL OPERATOR'S MANUAL 690102
IBM SYSTEM/360 OPERATING SYSTEM LINKAGE EDITOR AND LOADER PROGRAM NUMBERS 3605-ED-51C 3605-ED-521 3605-ED-547
IBM SYSTEM/360 OPERATING SYSTEM FULL AMERICAN NATIONAL STANDARD COBOL PROGRAMMERS GUIDE NUMBERS 3605-CB-546
IBM SYSTEM/360 OPERATING SYSTEM EXTRAN IV (G AND H) PROGRAMMERS GUIDE PROGRAM NUMBERS 3605-ED-503 3605-ED-520
MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM 700603
MAKING AUTOMATED COMPUTER PROGRAM DOCUMENTATION MAINTENANCE OF A FLATULAR SYSTEM DESIGN 710103
BELLFLOW AUTOMATIC FLOWCHARTING SYSTEM 771103
AN AUTOMATED SYSTEM FOR GENERATING PROGRAM
SYSTEM CONTINUATION

DOCUMENTATION

THE INTEGRATION OF SYSTEM SPECIFICATIONS AND PROGRAM CODING: A REPORT ON EXPERIENCE OF THE MEDLARS II PROJECT.

IBM SYSTEM/360 OPERATING SYSTEM ASSEMBLER PROGRAMMERS GUIDE PROGRAM NUMBER 360L-AS-037

IBM SYSTEM/360 OPERATING SYSTEM PL/I (P) PROGRAMMERS GUIDE PROGRAM NUMBER 360S-PL-511

UNIVAC 1100 SERIES OPERATING SYSTEM

ED PROCESSOR - UNIVAC 1100 SERIES OPERATING SYSTEM PROGRAMMER REFERENCE

MAINTENANCE AND ENHANCEMENT OF THE AED SYSTEM

ARPA NETWORK SERIES I. INTRODUCTION TO THE ARPA NETWORK AT RAND AND TO THE RAND VIDEO GRAPhICS SYSTEM

CFMS - CARD FILE MAINTENANCE SYSTEM

PLUS D/5, A DIRECT ACCESS SOFTWARE SECURITY AND CONTROL SYSTEM

PANDALET - THE DIRECT ACCESS LIBRARY SYSTEM

SYSTEM-OS

SYSTEM/360 ADMINISTRATIVE TERMINAL SYSTEM-OS

SYSTEMATICS

THE USE OF DECISION TABLES WITHIN SYSTEMATICS

SYSTEMS

CONTROL DATA 6600 SERIES COMPUTER SYSTEMS REFERENCE MANUAL NUMBER 6C1006C

DECISION TABLES FOR AED MANAGEMENT SYSTEMS

ON-LINE MANAGEMENT SYSTEMS INFORMATION

SYSTEM CONTINUATION

CONTROL DATA 6400/6510/6600 COMPUTER SYSTEMS CONDL REFERENCE MANUAL PUBLICATION NUMBER 60191200

CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS COMPASS REFERENCE MANUAL PUBLICATION NUMBER 60190900

CONTROL DATA 6400/6500/6600 COMPUTER SYSTEMS FORTRAN REFERENCE MANUAL PUBLICATION NUMBER 60174900

MICROFICHE STORAGE AND RETRIEVAL SYSTEMS STUDY:

DEFINING ADEQUATE SYSTEMS DOCUMENTATION

A SURVEY OF DATA STRUCTURES FOR COMPUTER GRAPHICS SYSTEMS

TABLE

THE INTERPRETATION OF LIMITED ENTRY DECISION TABLE FORMAT AND RELATIONSHIPS AMONG CONDITIONS

DECISION TABLE TRANSLATION

TABLES

LOGIC STRUCTURE TABLES

TABLES, FLOW CHARTS AND PROGRAM LOGIC

USING DECISION STRUCTURE TABLES

USE OF DECISION TABLES IN COMPUTER PROGRAMMING

CONVERSION OF LIMITED ENTRY DECISION TABLES TO COMPUTER PROGRAMS

DATA, DOCUMENTATION, AND DECISION TABLES

PROGRAMMING DECISION TABLES IN FORTRAN, COBOL OR ALGOL

DECISION TABLES FOR AED MANAGEMENT SYSTEMS

CONVERSION OF LIMITED-ENTRY DECISION TABLES TO OPTIMAL COMPUTER PROGRAMS II MINIMUM AVERAGE PROCESSING TIME
TABLES
CONTINUATION

CONVERSION OF DECISION TABLES TO COMPUTER
PROGRAMS BY RULE MASK TECHNIQUES
DECISION TABLES
PARSING OF DECISION TABLES
CONVERSION OF LIMITED-ENTRY DECISION
TABLES TO OPTIMAL COMPUTER PROGRAMS II:
MINIMUM STORAGE REQUIREMENT
SIMULATION WITH DECISION TABLES
AN INTRODUCTION TO DECISION TABLES
THE USE OF DECISION TABLES WITHIN
SYSTEMS
GOOD DECISION TABLES AND THEIR USES
AMBIGUITY IN LIMITED ENTRY DECISION TABLES
ON THE CONVERSION OF DECISION TABLES TO
COMPUTER PROGRAMS
CONVERSION OF LIMITED-ENTRY DECISION
TABLES TO COMPUTER PROGRAMS. A PROPOSED
MODIFICATION TO POLLACK'S ALGORITHM
TALKING
TALKING DOWN A PROGRAM
TAPE
CLOT - CARD LIBRARY ON TAPE
TECO
TECO - TEXT EDITOR AND CORRECTOR -
REFERENCE MANUAL
TERMINAL
SYSTEM/360 ADMINISTRATIVE TERMINAL
SYSTEM--OS
TEXT
EDIT - ANONYMOUS TEXT EDITOR (EDIT) REFERENCE
MANUAL
KRONOS TEXT EDITOR (EDIT)
APPENDIX B

EXISTING DOCUMENTATION AIDS
AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Working Paper No. 1
July 25, 1972
Proprietary Documentation Systems

by

Randy Birge

Texas A&M University
Texas Engineering Experiment Station
ABSTRACT

This report is intended to acquaint the reader with features supplied by proprietary automatic documentation software packages currently on the market. A number of systems are listed with their corresponding purchase and rental costs, description of outputs, source languages processed, system environment, and in some cases sample outputs of the system.
Table of Contents

I. Introduction E-3
II. Comparison Charts E-6
III. Package Reports E-13
   AUTOCHART E-15
   AUTODIAGRAMMER II E-16
   AUTODOC E-28
   AUTOFLOW E-42
   COMCHART E-86
   DYNACHART E-97
   EZFLOW E-98
   FACTS E-105
   FLOWGEN/F E-112
   FORFLO E-115
   FORTRAN VARIABLE NAME DOCUMENTER E-116
   QUICK-DRAW E-117
   SUPEREF E-133
   FORDOC E-136

IV. Summary
INTRODUCTION

Every data processing manager and programmer is familiar with the problem posed by the requirements for accurate program documentation. The preparation of this documentation has traditionally been time consuming and, consequently, it has often been set aside in order to allow the programmer to work on higher priority projects.

The major goal inherent in any good documentation procedure is to construct a system for communicating a document's content to a human being. A well documented computer program enables user personnel to understand the program's objectives, its relationship to other programs in the overall installation, and its position in the system workflow.

To relieve the programmer of much of the responsibility of program documentation many proprietary automatic documentation system have developed over the past few years. Many of these are generalized documentation packages which can accept source code as input and generate routines for computer analysis of the program requiring documentation as well as routines for producing reports on graphical representation that describe the program.

To achieve a permanent record of the details of an existing program, the following program feature require documenting:

1) Functional relation
2) Cross-reference
3) Furnish suitable cross-referencing facilities so that references to and from a given data element can be adequately presented.
Each cross-reference can be indicated on the flowchart at the place where the related element is mentioned, or a separate listing can be printed. A separate list has the advantage of furnishing all cross-references in a convenient format.

4) Allow flexibility in the level of detail. A chart can be developed on a symbol-per-statement basis, or it can be presented as an overall picture of the program's functions.

5) Convert comments in the source statements as well as other explanatory remarks into annotative flowchart statements.

6) In the absence of explanatory remarks, generate the necessary annotations.

7) Generate symbols that conform to accepted conventions or adhere to documented standards such as the ANSI standards.

Some of the proprietary automatic documentation packages listed in this report contain all of the above features and more, while some simply give simple cross-references. The purchase and rental costs of these documentation packages are a function of the number of output features they list. Purchase price range from $175 for a simple cross-referencing documentation package, FXREF, to $7,000 for the AUTOFLOW system which contains possibly the greatest number of output features.

The information for this report was obtained from the various software companies which produce proprietary automatic documentation software packages and from the Auerbach Computer Technology Reports. Letters requesting information on particular software packages were sent out to thirty five software companies. Of these original thirty five requests, about fifty percent of the companies responded.
A second letter was sent out to those companies who did not respond to the first letter. Less than fifty percent of these companies responded to the second letter.

Some of the information which could not be obtained from the software companies which produced these automatic documentation packages was supplemented by the Auerbach Computer Technology Reports.
COMPARISON CHARTS

Introduction

The following charts present many characteristics of existing documentation packages. These charts are intended as a guidance for comparing available documentation packages. Quick comparisons may be made through the use of the charts, and further information on a particular documentation package can then be found in the Package Reports section of this report.

- The charts are divided into two major sections:
  - System Requirements
  - Characteristics
<table>
<thead>
<tr>
<th>PACKAGE</th>
<th>AUTOCHART</th>
<th>AUTODIAGRAMMER II</th>
<th>AUTODOC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Beloit Computer Center</td>
<td>Aires Corp.</td>
<td>Data Instruments Co.</td>
</tr>
<tr>
<td>System</td>
<td>IBM 360/25</td>
<td>IBM 360/30 , UNIVAC 9300</td>
<td>any capable of COBOL support</td>
</tr>
<tr>
<td>Main Storage</td>
<td>32K</td>
<td>52K</td>
<td>28-64K</td>
</tr>
<tr>
<td>Auxiliary Storage</td>
<td>Disk</td>
<td>Magnetic tape or disc</td>
<td></td>
</tr>
<tr>
<td>Input/Output</td>
<td>card reader/punch, printer</td>
<td>Disc, Cardreader, Printer</td>
<td>line printer, card reader disc, tape</td>
</tr>
<tr>
<td>Operating System</td>
<td>DOS, OS-MVT</td>
<td>DOS, OS</td>
<td>any capable of COBOL support</td>
</tr>
<tr>
<td>Source Language</td>
<td>(not available)</td>
<td>Assembly Language</td>
<td>COBOL and IBM assembly</td>
</tr>
<tr>
<td>Package Type</td>
<td>Program Documentation; detailed flowchart, cross-reference listing of labels</td>
<td>Program Documentation; detail flowchart, or high level. cross-reference lists, logic charts, label cross-reference, unmatched label tables, record and report layout source listing</td>
<td>Documentation Aid; cover page, source list, error list, report and record layout, cross-reference list, detail flowchart, brief logic chart</td>
</tr>
<tr>
<td>Logic Flow Presentation</td>
<td>Detail flowchart at same level as source code</td>
<td>Detail flowchart at source code level; variable symbols contain source codes, flow is vertical and page formatting is available. high-level logic chart ill.par. relationships</td>
<td>narrative description from source comments, detail flowchart with ANSI or IBM symbols same level of detail as source code, depicts page, connector, no. of referenced statements and referencing statements. logic chart depicts statements that affect only logic flow</td>
</tr>
<tr>
<td>Cross References</td>
<td>shows location of each label in the flowchart, location of each label that branches to that statement</td>
<td>Paragraph cross-references in high-level logic chart; Label cross-reference table, Altered Statements Unmatched unreferenced labels</td>
<td>listing of data items, procedure names, ext. names, literals, figurative constants, system names, alphanumeric ordering gives label type, defining and referencing source code number also on detail flowchart</td>
</tr>
<tr>
<td>Man-Machine Interface</td>
<td>Input-card reader, Output-printer</td>
<td>Input-card reader, Output-printer</td>
<td>Input-cards, disc, tape; can input up to 99 programs in batched mode from card reader or from system library</td>
</tr>
<tr>
<td>Cost</td>
<td>Purchase - $950.00 Rental - (none)</td>
<td>Purchase - $3,200 Rental - (none)</td>
<td>Purchase - $4,800 (3-yr lease) Rental - (none)</td>
</tr>
<tr>
<td>Source Programs Processor</td>
<td>COBOL</td>
<td>COBOL FORTRAN } optimal</td>
<td>COBOL IBM assembly</td>
</tr>
<tr>
<td>Comments</td>
<td>Two phase system; flowchart, cross-referencing</td>
<td>about 20-30 installations</td>
<td>about 50 installations</td>
</tr>
<tr>
<td>PACKAGE</td>
<td>AUTOFLOW (1964)</td>
<td>COMCHART</td>
<td>DYNACHART</td>
</tr>
<tr>
<td>---------</td>
<td>----------------</td>
<td>----------</td>
<td>-----------</td>
</tr>
<tr>
<td>FEATURE</td>
<td>Applied Data Research</td>
<td>Compress</td>
<td>Applications Programming Co.</td>
</tr>
<tr>
<td>System</td>
<td>IBM 360, 1400; RCA Spectra 70; H200</td>
<td>IBM 360/370; RCA Spectra 70</td>
<td>any with COBOL Compiler</td>
</tr>
<tr>
<td>Main Storage</td>
<td>Support for OS, DOS, TOS, TDOS</td>
<td>65K bytes</td>
<td>30K character</td>
</tr>
<tr>
<td>Auxiliary Storage</td>
<td>System residence device</td>
<td>4 sequential files</td>
<td>1 disc or 4 tapes</td>
</tr>
<tr>
<td>Input/Output</td>
<td>card reader, tape, disc, line printer</td>
<td>card reader, tape, disc, line printer</td>
<td>1/0 - card reader, printer, disc, tape</td>
</tr>
<tr>
<td>Operating System</td>
<td>OS, DOS, TOS (IBM 360), TDOS (RCA)</td>
<td>OS, TOS, TDOS</td>
<td>any with COBOL compiler</td>
</tr>
<tr>
<td>Source Language</td>
<td>Basic Assembly Language</td>
<td>CODOL and Assembly</td>
<td>COBOL</td>
</tr>
<tr>
<td>Package Type</td>
<td>Documentation aid; generates a flowchart used for debugging summarizing an existing system constructing a preimplementation flowchart</td>
<td>Documentation aid: produces a flowchart, deck listing, a cross-reference index of element names; also a design language for developing programs</td>
<td>Documentation aid which produces; flowchart, diagnostics, cross-references</td>
</tr>
<tr>
<td>Logic Flow Presentation</td>
<td>In form of text and flowcharts; flowchart at same level as source code, distinct units for processing blocks, page, source card, symbol (box) nos. on and off-page connections</td>
<td>Standard symbols used in flowchart; built in cross-references; detailed analysis of each statement</td>
<td>Flowcharts at same level of detail as source code; separate page per subroutine; paragraph not split between logical pages</td>
</tr>
<tr>
<td>Cross References</td>
<td>Table of contents and references index precedes flowchart and contains cross-referencing information: Listing of COBOL data names in alphanumeric order with page, flowchart box#, and source sequence #.</td>
<td>Alphabetical index of all elements: outline of procedures, listing all incoming and outgoing references</td>
<td>Labels alphabetically listed as are defining source code sequence number and all statements referencing label</td>
</tr>
<tr>
<td>Characteristics</td>
<td>Input - card reader, Output - printer</td>
<td>Input - card reader, tape, disc, Output - line printer</td>
<td>Input - card, tape, disc, Output - printer</td>
</tr>
<tr>
<td>Interface</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost</td>
<td>Purchase - $3,000-7,000</td>
<td>Purchase - (none)</td>
<td>Purchase - $4,400 (1st yr)</td>
</tr>
<tr>
<td></td>
<td>Rental - per installation basis</td>
<td>Rental - $2,425 (1 yr); $4,225 (10 yrs)</td>
<td>Rental - $700/yr after 1st yr</td>
</tr>
<tr>
<td>Source Programs Processor</td>
<td>COBOL, PL/I Assembly, FORTRAN</td>
<td>COBOL, Assembly, Design Language</td>
<td>COBOL</td>
</tr>
<tr>
<td>Comments</td>
<td>About 1400 installations</td>
<td>Over 50 installations</td>
<td>About 45 installations</td>
</tr>
<tr>
<td>FEATURE</td>
<td>EZFLOW Systonetics Corp.</td>
<td>FACTS Bonner &amp; Moore Asso.,Inc.</td>
<td>FLOWGEN/F Calcomp</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------</td>
<td>----------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>System</td>
<td>IBM 360, CDC 6000 Series</td>
<td>Sigma 7</td>
<td>Not Available</td>
</tr>
<tr>
<td>Main Storage</td>
<td>110K bytes (IBM), 32K words (CDC)</td>
<td>Not Available</td>
<td>Not Available</td>
</tr>
<tr>
<td>Auxiliary Storage</td>
<td>Card reader, tape, or disc, line printer</td>
<td>Not Available</td>
<td>Not Available</td>
</tr>
<tr>
<td>Input/Output</td>
<td>Card reader, tape, printer</td>
<td>Card reader, CalComp plotter</td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td>OS(IBM), SCOPE(CDC)</td>
<td>Not Available</td>
<td>Not Available</td>
</tr>
<tr>
<td>Source Language</td>
<td>FORTRAN IV</td>
<td>Not Available</td>
<td>Not Available</td>
</tr>
<tr>
<td>Package Type</td>
<td>Documentation Aid that generates flowchart input/output source deck a cross-reference list statement reference table</td>
<td>Documentation aid; cross referencing</td>
<td>Program Documentation Detailed Flowchart</td>
</tr>
<tr>
<td>Logic Flow Presentation</td>
<td>Flowchart at same level as source code; up and down single-col flow</td>
<td>None</td>
<td>Detail Flowchart</td>
</tr>
<tr>
<td>Cross References</td>
<td>Listing of statement labels and references to show old and resequenced source code label numbers and source list line number</td>
<td>Six reports are given; common report, local report, format statement label report, recap, global report</td>
<td>(none)</td>
</tr>
<tr>
<td>Man-Machine Interface</td>
<td>Input-card reader, tape, or disc Output-printer</td>
<td>Input-card reader, tape Output-printer, tape</td>
<td>Input-card reader, drum, disc, or tape Output-CalComp Plotter</td>
</tr>
<tr>
<td>Cost</td>
<td>Purchase - $1,750</td>
<td>Purchase - Not Available</td>
<td>Rental - Not Available</td>
</tr>
<tr>
<td>Source Processor</td>
<td>FORTRAN</td>
<td>FORTRAN</td>
<td>FORTRAN</td>
</tr>
<tr>
<td>Comments</td>
<td>under 10 installations</td>
<td>Not Available</td>
<td>Not Available</td>
</tr>
<tr>
<td>Feature</td>
<td>FORFLO DNA Systems, Inc.</td>
<td>FORTRAN VARIABLE NAME DOCUMENTER Data for Management Decision</td>
<td>QUICK-DRAW National Computer Analysts</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------</td>
<td>---------------------------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>System</td>
<td>IBM 1130/1800</td>
<td>Designed for user's configuration</td>
<td>Any with COBOL compiler</td>
</tr>
<tr>
<td>Main Storage</td>
<td>1130(8K core); 1800 (10K variable core)</td>
<td>Capable of supporting FORTRAN compiler</td>
<td>32K bytes (DOS, TOS); 65K bytes (OS)</td>
</tr>
<tr>
<td>Auxiliary Storage</td>
<td>1 disc</td>
<td>System residence device</td>
<td>System residence device</td>
</tr>
<tr>
<td>Input/Output</td>
<td>disc, card reader, line printer</td>
<td>Card reader, line printer</td>
<td>Card reader, tape, disc, line printer</td>
</tr>
<tr>
<td>Operating System</td>
<td>Monitor, TSX version, MTX</td>
<td>Specific system not required</td>
<td>Specific system not required</td>
</tr>
<tr>
<td>Source Language</td>
<td>FORTRAN and Subroutines</td>
<td>FORTRAN</td>
<td>COBOL, Assembler</td>
</tr>
<tr>
<td>Package Type</td>
<td>Documentation package; gives flowcharts and resequenced source listing</td>
<td>Documentation tool; source list and alphabetical list of referenced variable names</td>
<td>Documentation aid</td>
</tr>
<tr>
<td>Logic Flow</td>
<td>Flowchart at same level as source code, source statements within blocks</td>
<td>Flowchart at same level as source code uses symbols, text, flow-connectors, reference information each element of chart identified by separate block, QUICK-DRAW-assignment of source card sequence, &amp; statement nos. or tag double-page format</td>
<td>Flowchart at same level as source code uses symbols, text, flow-connectors, reference information each element of chart identified by separate block, QUICK-DRAW-assignment of source card sequence, &amp; statement nos. or tag double-page format</td>
</tr>
<tr>
<td>Cross References</td>
<td>(none)</td>
<td>Sequence number of all statements that reference each variable name</td>
<td>Referencing described by flowchart coordinates &amp; source code no. Cross-reference tables include data-defined fields, Alphabetic listing of tag names, procedural statements and subroutines, data and procedure name, procedural logic</td>
</tr>
<tr>
<td>Non-Machine Interface</td>
<td>Input-cards or disc</td>
<td>Input-card reader</td>
<td>Input-card reader, tape, disc</td>
</tr>
<tr>
<td></td>
<td>Output-printer or disc</td>
<td>Output-line printer</td>
<td>Output-printer</td>
</tr>
<tr>
<td>Cost</td>
<td>Purchase - $480(cards); $600(disc)</td>
<td>Purchase - $350(object); $500(source)</td>
<td>Purchase - none</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Rental - $1,400-3,100 (1yr); $1,900-6,300 (3yrs)</td>
</tr>
<tr>
<td>Source Programs Processor</td>
<td>FORTRAN</td>
<td>FORTRAN</td>
<td>FORTRAN, FORTRAN Assembly</td>
</tr>
<tr>
<td>Comments</td>
<td>Rosequence listing arranges source statement label nos. by fives, about 63 installations</td>
<td>About 6 installations</td>
<td>QUICK-DRAW is being used at over 400 installations</td>
</tr>
<tr>
<td>PACKAGE FEATURE</td>
<td>SUPEREF Mantech Corporation</td>
<td>FORDOC J. Toellner &amp; Asso.</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------</td>
<td>-----------------------------</td>
<td></td>
</tr>
<tr>
<td>System</td>
<td>CDC 6000 series</td>
<td>not Available</td>
<td></td>
</tr>
<tr>
<td>Main Storage</td>
<td>Not Available</td>
<td>Not Available</td>
<td></td>
</tr>
<tr>
<td>Auxiliary Storage</td>
<td>Not Available</td>
<td>Not Available</td>
<td></td>
</tr>
<tr>
<td>Input/Output</td>
<td>Card reader, printer</td>
<td>Card reader/punch, printer</td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td>Not Available</td>
<td>Not Available</td>
<td></td>
</tr>
<tr>
<td>Source Language</td>
<td>Not Available</td>
<td>Not Available</td>
<td></td>
</tr>
<tr>
<td>Package Type</td>
<td>Documentation Aid</td>
<td>Program restructuring &amp; documentation Cross-reference listing and restructured source deck.</td>
<td></td>
</tr>
<tr>
<td>Logic Flow</td>
<td>(none)</td>
<td>(none)</td>
<td></td>
</tr>
<tr>
<td>Cross References</td>
<td>Produces a comprehensive symbolic name Cross-reference dictionary</td>
<td>Variable name cross-reference.</td>
<td></td>
</tr>
<tr>
<td>Human-Machine Interface</td>
<td>Not Available</td>
<td>Input - cardreader Output - printer, card punch</td>
<td></td>
</tr>
<tr>
<td>Cost</td>
<td>Not Available</td>
<td>Not Available - Purchase Not Available - Retail</td>
<td></td>
</tr>
<tr>
<td>Source Programs Processor</td>
<td>FORTRAN</td>
<td>FORTRAN</td>
<td></td>
</tr>
<tr>
<td>Comments</td>
<td>Not Available</td>
<td>Not Available</td>
<td></td>
</tr>
</tbody>
</table>
PACKAGE REPORTS
PACKAGE REPORTS

Introduction

This section contains information on documentation aids packages currently available to the data processing market. Sample outputs are given for a number of the packages.

Each report has two major sections:

GENERAL - In this section features such as package purpose, system requirements, pricing, and source languages processed are given.

PACKAGE OUTPUT - This section lists the various outputs that the particular package gives. A description of each output is also given.
AUTOCHART
Beloit Computer Center, Inc.

GENERAL

AUTOCHART accepts COBOL source programs as input and produces a detailed flowchart and a cross-reference listing of all labels used in the program. These two phases can be run in one jobstream or they can be executed as separate programs.

The minimum configuration for using AUTOCHART is an IBM System 360 Model 25 on up with a 32K central processor, one disk storage file (2311 or 2314), one card reader/punch, and a line printer. A magnetic tape drive is optional if you wish to execute the two phases in one jobstream. The system operates either under DOS or OS-MVT.

AUTOCHART is a low cost system with limited capabilities. The purchase price for AUTOCHART is $950.00 complete.

PACKAGE OUTPUT

Flowchart - A detailed flowchart is given.

Cross-Reference - This listing shows the location of each label in the flowchart as well as the location of each label that branches to that statement.

(No output examples are available)
AUTODIAGRAMMER II
AIRES CORPORATION

GENERAL

AUTODIAGRAMMER II is used as a documentation aid and as a debug tool for COBOL source programs. Standard outputs are a detailed flowchart, a diagnostic listing, an altered statement cross-reference table, a record layout, a source listing, and an unmatched label table.

The package can be implemented on any IBM System/360 configuration capable of operating under DOS (Model 30 and up) with 64K bytes or OS (Model 40 and up) with 128K bytes. DOS requires a 2111 or 2314 disc; OS requires any direct access device. Additional requirements are one card reader and one printer. AUTODIAGRAMMER II is written in 360 assembly language.

Purchase price of the package is $3,200.

PACKAGE OUTPUT

The detailed flowchart is given in which each statement is displayed with a separate logic block. User can select one or two logical pages per physical page, margin size, and six or eight line-per-inch printing density. All symbols are of variable size. Processing comments associated with each source code are included within its corresponding flowchart symbol. Narrative source code comments are printed in-line and are not enclosed within symbols.

The diagnostic table lists all error the diagnostic error codes, the statements in error, the location of each statement in the detailed
flowchart and the error code associated with each mistake.

The altered Statement Cross Reference listing displays the paragraph names and flowchart locations of the statements being altered, the location of each altering statement, and the name and flowchart location of each destination.

The high level logic chart displays the relationships between paragraphs in the program.

The input/output chart shows each file name, unit assigned, record name, and input or output relationship to the program.

The label cross-reference table displays each label by name, location, and sequence number and gives the flowchart location of each reference to the label by statements in the program.

The record layout graphically displays each record defined in the program with its field and sub-field name, length, and type specifications.

The sample report page displays a sample printed page that would result from the Report Definition Section of the user's program.

The source listing shows all the source statements and remarks as they appear in the source deck.

The unmatched label table displays each label which is defined but never referenced, and each label which is referenced but never defined.
CCBC DIAGNOSTIC ERROR CODES

A: MISSING PROGRAM I.O. LABEL
B: MISSING I.O. DIVISION LABEL
C: MISSING ENVIRONMENT LABEL
D: MISSING FILE CONTROL SECTION
E: MISSING END STATEMENT
F: MISSING SELECT CLAUSE
G: MISSING ASSIGN CLAUSE
H: MISSING C-Level STATEMENT
I: LABEL IN EXCESS OF HEIRY CHARACTERS
J: INVALID PICTURE CLAUSE
K: INVALID REDISAE
L: STATEMENT LUT OF SEQUENCE
M: INVALID CONTINUATION OR MISSING PERIOD
N: INVALID VALUE CLAUSE
O: IMPROPER MARGIN USE
P: INVALID SUBRNAME
Q: PROCEDURE NAME IS A CCBC RESERVED NAME
R: NON-TERMINATING SUBSCRIPT OR A-NUERNIC LITERAL

Figure 4. Diagnostic Table and Error Codes
**High Level Logic Chart.**

The High Level Logic Chart illustrates the relationships between the paragraphs in a COBOL program. By portraying the linkage of process blocks, this chart assists the programmer in visualizing control transfer patterns and reviewing them for accuracy.

Each paragraph in the source program is shown with its name and flowchart location in a processing block.

The processing blocks are listed in the order of appearance in the source program. They are connected by vertical lines whenever the preceding paragraph is not an EXIT or GO TO paragraph. To the left of each block are listed any paragraphs which transfer control into the block. To the right of each block are listed any paragraphs to which the block may transfer control. Figure 5 illustrates the format of the High Level Logic Chart.

---

**Figure 5.** The High Level Logic Chart for the program in Appendix "A".
Continuation of Figure 5.

TRANSMITTED
0051 CUST-TR-ON

0052 CUST-TR-MI

0062 EXP-M1-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR

0062 EXP-M2-OR
Figure 6. The Input-Output Logic Chart.
Figure 7 shows the Label Cross-Reference Table produced by AUTODIAGRAMMER II using the source program in Appendix "A".

Figure 7. Label Cross-Reference Table

The use of the Label Cross-Reference Table in check transfer instructions is shown in Figure 7a. This kind of step by step analysis is applicable to labels which are referenced in the source program.
### REGULAR LAYOUT/S

**CONTOL**

<table>
<thead>
<tr>
<th>LABEL</th>
<th>POSITION</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUS#1</td>
<td>1</td>
<td>AN</td>
</tr>
<tr>
<td>CUS#2</td>
<td>2</td>
<td>AM</td>
</tr>
<tr>
<td>CUS#3</td>
<td>3</td>
<td>AM</td>
</tr>
<tr>
<td>CUS#4</td>
<td>4</td>
<td>AM</td>
</tr>
<tr>
<td>FILLER</td>
<td>5-7</td>
<td>AM</td>
</tr>
<tr>
<td>FILLER</td>
<td>8-10</td>
<td>AM</td>
</tr>
<tr>
<td>FILLER</td>
<td>11-12</td>
<td>AM</td>
</tr>
<tr>
<td>FILLER</td>
<td>13-15</td>
<td>AM</td>
</tr>
<tr>
<td>FILLER</td>
<td>16-18</td>
<td>AM</td>
</tr>
<tr>
<td>FILLER</td>
<td>19-21</td>
<td>AM</td>
</tr>
<tr>
<td>FILLER</td>
<td>22-24</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>25-28</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>29-32</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>33-36</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>37-40</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>41-44</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>45-48</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>49-52</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>53-56</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>57-60</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>61-64</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>65-68</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>69-72</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>73-76</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>77-80</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>81-84</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>85-88</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>89-92</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>93-96</td>
<td>AM</td>
</tr>
<tr>
<td>STEEL</td>
<td>97-100</td>
<td>AM</td>
</tr>
</tbody>
</table>

### SECURE LAYOUT/S

**CONTOL**

<table>
<thead>
<tr>
<th>LABEL</th>
<th>POSITION</th>
<th>WORKFILE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILLER</td>
<td>1-2</td>
<td>55</td>
</tr>
<tr>
<td>FILLER</td>
<td>3-4</td>
<td>55</td>
</tr>
<tr>
<td>FILLER</td>
<td>5-6</td>
<td>55</td>
</tr>
<tr>
<td>FILLER</td>
<td>7-8</td>
<td>55</td>
</tr>
<tr>
<td>FILLER</td>
<td>9-10</td>
<td>55</td>
</tr>
<tr>
<td>FILLER</td>
<td>11-12</td>
<td>55</td>
</tr>
<tr>
<td>FILLER</td>
<td>13-14</td>
<td>55</td>
</tr>
<tr>
<td>FILLER</td>
<td>15-16</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>17-18</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>19-20</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>21-22</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>23-24</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>25-26</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>27-28</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>29-30</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>31-32</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>33-34</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>35-36</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>37-38</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>39-40</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>41-42</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>43-44</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>45-46</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>47-48</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>49-50</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>51-52</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>53-54</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>55-56</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>57-58</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>59-60</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>61-62</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>63-64</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>65-66</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>67-68</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>69-70</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>71-72</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>73-74</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>75-76</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>77-78</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>79-80</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>81-82</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>83-84</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>85-86</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>87-88</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>89-90</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>91-92</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>93-94</td>
<td>55</td>
</tr>
<tr>
<td>STEEL</td>
<td>95-96</td>
<td>55</td>
</tr>
</tbody>
</table>

---

06/05/69

******* ARIES CORP. 00 SAMPLE AUTODIAGRAMMER II ********

---

06/05/69

******* ARIES CORP. 00 SAMPLE AUTODIAGRAMMER II ********

---

06/05/69

******* ARIES CORP. 00 SAMPLE AUTODIAGRAMMER II ********
g. **Sample Report Page.**

When a program has been written using the COBOL Report Writer Feature, the programmer may request that AUTODIAGRAMMER II produce a sample printed page of the report. This output would show the report exactly as it is established in the Report Definition Section of the User's Program.

A sample report is given below.

---

A Sample Report produced by AUTODIAGRAMMER II.
Figure 9. Unmatched Label Table.
<table>
<thead>
<tr>
<th>CARD ID</th>
<th>PAGE/EXEC</th>
<th>DIAGNOSTIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>012500</td>
<td>2.01</td>
<td>IMPROPER USE OF RESERVED WORD - START</td>
</tr>
<tr>
<td>014600</td>
<td>3.18</td>
<td>UNDEFINED - ERROR - EXTERNAL REFERENCE</td>
</tr>
<tr>
<td>027231</td>
<td>5.23</td>
<td>IMPROPER USE OF RESERVED WORD - NOTE</td>
</tr>
<tr>
<td>027232</td>
<td>5.23</td>
<td>UNDEFINED PROGRAM REFERENCE - ERROR-1</td>
</tr>
<tr>
<td>027233</td>
<td>6.01</td>
<td>ALLOCATED LINE NOT GO TO - END-OF-JOB</td>
</tr>
<tr>
<td>027233</td>
<td>6.01</td>
<td>UNDEFINED PROGRAM REFERENCE - ERROR-2</td>
</tr>
<tr>
<td>027233</td>
<td>6.01</td>
<td>NO ENTRANCE TO THIS STATEMENT</td>
</tr>
<tr>
<td>027234</td>
<td>6.02</td>
<td>INVALID SYNTAX - 2</td>
</tr>
</tbody>
</table>
AUTODOC
DATA INSTRUMENTS COMPANY

GENERAL

AUTODOC accepts COBOL source code and, when implemented on an IBM 360, will also accept assembly language source code. AUTODOC generates a cover page, a source program listing, and an error list. At the user's option it can also generate a document for describing report (COBOL only) and record layouts, for listing Data and Procedure cross references, and for creating both a detailed flowchart and a logic chart. AUTODOC can process programs individually, or it can process up to 99 source programs in the batch mode.

AUTODOC can operate on the following systems: Honeywell 200 central processor with 28K characters of core, IBM System 360 (model 25 and up) configured with a 48K problem program partition and capable of operating in a DOS or OS environment, CDC Series 3304 or 3504 central processor with 32K words of core, and NCR century 100 or 200 with 32K core, and a Burroughs B 5500 with 64 K core. AUTODOC is written in COBOL.

Cost for AUTODOC is $4,800 for a 3-year license agreement for the first customer installation. Each additional installation of the same language costs $2,500. Separate AUTODOC packages for processing COBOL and assembly language can be purchased and the package for processing the second language costs $1,800.
PACKAGE OUTPUT

The **cover page** gives the name and author of the program. It also gives information describing the program's hardware environment, security status, and origination date. The remarks section can give a complete narrative description of the program.

The **source program list** lists the statements processed along with their sequence number. Thus number can be used as a reference number later by autodoc.

The **error list** gives certain syntax errors found. It lists the statement in error and the sequence number of this statement.

The **report layout** (COBOL only) is provided for all reports defined in the Report Section.

The **data reference list** (COBOL only) lists all data items used in the program along with information pertaining to each of them.

The **procedure reference list** (COBOL only) lists alphabetically all procedure names used in the program along with its associated source sequence number. Also given are the page connector number of the flow-chart connector symbol generated by the procedure name and the source sequence numbers of statements which reference it.

The **special reference list** (COBOL only) lists all source sequence numbers referencing an external name, a literal, figurative constants, and system names.

The **label reference list** (assembly only) gives an alphabetical listing of all labels defined within the CSECT or DSECT being processed. Their associated sequence number is also given. Additional entries include the page and connector symbol generated by the label, and the
source sequence numbers of all statements that reference that label name.

The detail flowchart provides a two-dimensional representation of the logic flow. It constructs a symbol for each source statement and its related text. The user can select ANSI Standard or IBM flowchart symbols. Each flowchart logical page is divided vertically into four position segments, with flow direction from top to bottom. The flowchart can be printed as one or two physical pages per logical page. Both offpage and onpage connectors are generated.

The logic chart (COBOL only) charts only statements which affect the logical flow of the program, as well as statements of the notes and input/output type.
AUTODOC

DOCUMENTATION OF

"AUTODOC SAMPLE"

AUTHOR
CTC COMPUTER CORPORATION, PROPRIETARY SYSTEMS DIV.

INSTALLATION
CTC - PSD.

SECURITY
THIS IS A PROPRIETARY PROGRAM, THE USE OF WHICH IS GOVERNED BY CONTRACTUAL AGREEMENT.

DATE WRITTEN
1969.

REMARKS
AUTODOC IS A TOTAL PROGRAM DOCUMENTATION SYSTEM IT PROVIDES AUTOMATICALLY THE FOLLOWING PRODUCT OUTPUTS -

1) COVER PAGE
2) RECORD LAYOUT
3) SOURCE LISTING
4) ERROR LIST
5) DATA REFERENCE LIST
6) PROCEDURE REFERENCE LIST
7) SPECIAL REFERENCE LIST
8) DETAIL FLOWCHART
9) LOGIC CHART

AUTODOC IS AN EXTREMELY EASY SYSTEM TO USE - NO SPECIAL CODING OR INPUT PREPARATION IS REQUIRED. ALL PRODUCTS ARE AUTOMATICALLY GENERATED UNLESS SPECIFICALLY SUPPRESSED THROUGH THE USE OF SIMPE PARAMETER CARDS.

PROPRIETARY SYSTEMS DIVISION
<table>
<thead>
<tr>
<th>REF</th>
<th>ERROR DESCRIPTION</th>
<th>SOURCE IMAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>84:</td>
<td>SYNTAX ERROR...SISPLAY</td>
<td>'OH HAPPY DAY' UPON CONSOLE.</td>
</tr>
</tbody>
</table>
**PROGRAM: AUTODIC SAMPLE**
**RECORD LAYOUT**

<table>
<thead>
<tr>
<th>FILE NAME</th>
<th>REF NBR</th>
<th>MODE</th>
<th>RECORD SIZE</th>
<th>BLOCKED</th>
<th>FILE SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAST-IN</td>
<td>36</td>
<td>F</td>
<td>80</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BYTE</th>
<th>RECORD/DATA NAME</th>
<th>PFN NBR</th>
<th>TYPE</th>
<th>FORM</th>
<th>LOG SIZE</th>
<th>PHYS SIZE</th>
<th>OCCURS</th>
<th>JST</th>
<th>Sync</th>
<th>SIGN</th>
<th>POINT</th>
<th>FDT</th>
<th>CHAP</th>
<th>FLT</th>
<th>LV</th>
<th>ALNK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COPY</td>
<td>41</td>
<td>RCD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>C</td>
<td>42</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>D</td>
<td>43</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>E</td>
<td>44</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>F</td>
<td>45</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAST-IN</td>
<td>46</td>
<td>RCD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>A</td>
<td>47</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>48</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAST-OUT</td>
<td>49</td>
<td>RCD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>A</td>
<td>50</td>
<td>ELFH</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>51</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>52</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>D</td>
<td>53</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Working Storage

<table>
<thead>
<tr>
<th>BYTES</th>
<th>RECORD/DATA NAME</th>
<th>RFF NBR</th>
<th>TYPE</th>
<th>FORM</th>
<th>LIG SIZE</th>
<th>PHYS SIZE</th>
<th>OCCURS</th>
<th>JUST</th>
<th>SYNC</th>
<th>SIGN</th>
<th>POINT</th>
<th>FR</th>
<th>CHM</th>
<th>FLT</th>
<th>LV</th>
<th>BLNK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TEMP-REC</td>
<td>65</td>
<td>RCD</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>A</td>
<td>66</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>67</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>K</td>
<td>68</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>69</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>THE-WORLD</td>
<td>70</td>
<td>RCD</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>THIS-IS-THE-LONGEST-NAME</td>
<td>71</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>H-STORAGE</td>
<td>72</td>
<td>RCD</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>H1</td>
<td>73</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>H2</td>
<td>74</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>H3</td>
<td>75</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AN-PRICE</td>
<td>76</td>
<td>ELEM</td>
<td>A/N DISP</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PRICE</td>
<td>77</td>
<td>ELEM</td>
<td>NUM DISP</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PRICES</td>
<td>78</td>
<td>ELEM</td>
<td>NUM DISP</td>
<td>14</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>DUMMY-DATA-NAME</td>
<td>80</td>
<td>RCD</td>
<td>A/N DISP</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>REF NBR</td>
<td>TYPE</td>
<td>REFERENCES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
<td>------</td>
<td>------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNDEFINED</td>
<td></td>
<td></td>
<td>212</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>47</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>50</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>66</td>
<td>02</td>
<td>127 155 156 162 199 200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AN-PRICE</td>
<td>76</td>
<td>02</td>
<td>77</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>48</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>51</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>67</td>
<td>02</td>
<td>127 155 156 162 199</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>42</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>52</td>
<td>02</td>
<td>133 155 162 201 202</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COPY</td>
<td>41</td>
<td>01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRAW-ADR</td>
<td>20</td>
<td>SPL</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>43</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>53</td>
<td>02</td>
<td>133 155</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DUMMY-NAME</td>
<td>80</td>
<td>01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>44</td>
<td>02</td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>45</td>
<td>02</td>
<td>129 130</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H-STORE</td>
<td>72</td>
<td>01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H1</td>
<td>73</td>
<td>02</td>
<td>159 199 200 203 213</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H2</td>
<td>74</td>
<td>02</td>
<td>135 199 201 202 204 206 208</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H3</td>
<td>75</td>
<td>02</td>
<td>135 162 201 202 204 206 208</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K</td>
<td>68</td>
<td>02</td>
<td>133 162 203 204</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>69</td>
<td>02</td>
<td>162 204</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASTER-IN</td>
<td>36</td>
<td>FD</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASTER-OUT</td>
<td>46</td>
<td>01</td>
<td>33 40 103 112 116 118 121</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASTER-OUT</td>
<td>49</td>
<td>01</td>
<td>40 103 110 111 112 116 118 121 126 128 129 130 132 134 136 138</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASTER-OUT</td>
<td></td>
<td>UNDEFINED</td>
<td>105 107 109 110 114 116 121 123 125 127 129 131 133 135 137 139 141 143</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASTER-OUT</td>
<td></td>
<td>UNDEFINED</td>
<td>104 108 117 158</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ND</td>
<td>55</td>
<td>08</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRICE</td>
<td>77</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRICES</td>
<td>79</td>
<td>02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QTY-1</td>
<td>63</td>
<td>77</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QTY-2</td>
<td>64</td>
<td>77</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RCS-S-IN</td>
<td>41</td>
<td>77</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>REF NBR</td>
<td>CMT LOC</td>
<td>REFERENCES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
<td>---------</td>
<td>------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE</td>
<td>85</td>
<td>2-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE-0</td>
<td>86</td>
<td>2-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE-1</td>
<td>89</td>
<td>2-02</td>
<td>132 136 137 138 139 141</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE-2</td>
<td>91</td>
<td>2-03</td>
<td>90 135 150</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE-3</td>
<td>94</td>
<td>2-04</td>
<td>92 150</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE-4</td>
<td>97</td>
<td>2-05</td>
<td>90 95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE-5</td>
<td>98</td>
<td>2-05</td>
<td>95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABLE-6</td>
<td>115</td>
<td>2-06</td>
<td>95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAKER</td>
<td>145</td>
<td>4-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAKER-0</td>
<td>146</td>
<td>4-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAKER-1</td>
<td>149</td>
<td>4-02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAKER-2</td>
<td>152</td>
<td>4-03</td>
<td>150</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAKER-3</td>
<td>154</td>
<td>4-04</td>
<td>151</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAKER-4</td>
<td>164</td>
<td>5-01</td>
<td>162</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHARLEY</td>
<td>171</td>
<td>6-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHARLEY-0</td>
<td>172</td>
<td>6-01</td>
<td>V3 REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHARLEY-1</td>
<td>175</td>
<td>6-02</td>
<td>163 167 194</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHARLEY-2</td>
<td>178</td>
<td>6-03</td>
<td>166</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHARLEY-3</td>
<td>180</td>
<td>6-04</td>
<td>167</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHARLEY-4</td>
<td>182</td>
<td>6-05</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOG</td>
<td>195</td>
<td>7-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOG-0</td>
<td>196</td>
<td>7-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOG-1</td>
<td>198</td>
<td>7-02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EASY</td>
<td>203</td>
<td>8-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EASY-0</td>
<td>206</td>
<td>8-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EASY-1</td>
<td>209</td>
<td>8-02</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EASY-2</td>
<td>219</td>
<td>8-03</td>
<td>V3 REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MIRACLES</td>
<td>UNDEFINED</td>
<td>83</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>START</td>
<td>82</td>
<td>1-01</td>
<td>NO REF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| IN ABLE        | 94      | 2-04    | 215         |
| IN CHARLEY     | 178     | 6-03    | 176 186     |
| IN CHARLEY     | 180     | 6-04    | 176 179     |</p>
<table>
<thead>
<tr>
<th>LITERALS, FIG CONS, SYTEM NAMES</th>
<th>REFERENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>.0079</td>
<td>166</td>
</tr>
<tr>
<td>.15</td>
<td>125</td>
</tr>
<tr>
<td>.3</td>
<td>165</td>
</tr>
<tr>
<td>.17</td>
<td>161</td>
</tr>
<tr>
<td>-1</td>
<td>160</td>
</tr>
<tr>
<td>-3</td>
<td>161</td>
</tr>
<tr>
<td>&quot; TYPE = &quot;</td>
<td>156</td>
</tr>
<tr>
<td>&quot;.&quot;</td>
<td>184</td>
</tr>
<tr>
<td>&quot;*&quot;</td>
<td>130</td>
</tr>
<tr>
<td>&quot;CODE = &quot;</td>
<td>156</td>
</tr>
<tr>
<td>&quot;MERRY CHRISTMAS&quot;</td>
<td>104</td>
</tr>
<tr>
<td>&quot;THIS IS TO TEST FOR THE FLOAT&quot;</td>
<td>217</td>
</tr>
<tr>
<td>&quot;X&quot;</td>
<td>129</td>
</tr>
<tr>
<td>&quot;Y&quot;</td>
<td>129</td>
</tr>
<tr>
<td>&quot;I&quot;</td>
<td>155</td>
</tr>
<tr>
<td>QUOTE</td>
<td>130</td>
</tr>
<tr>
<td>SPACES</td>
<td>128</td>
</tr>
<tr>
<td>TALLY</td>
<td>144</td>
</tr>
<tr>
<td>ZERO</td>
<td>130 160 202</td>
</tr>
<tr>
<td>ZEROS</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>112</td>
</tr>
<tr>
<td>1</td>
<td>117 120 136</td>
</tr>
<tr>
<td>10</td>
<td>141 167</td>
</tr>
<tr>
<td>10.92</td>
<td>160</td>
</tr>
<tr>
<td>100</td>
<td>123</td>
</tr>
</tbody>
</table>
AUTOFLOW
APPLIED DATA RESEARCH

GENERAL

AUTOFLOW translates source language programs, written in COBOL, FORTRAN, PL/1, assembly language or numerous autocode languages, into flowchart documents in various levels of detail. This includes statement analysis, page allocation, line drawing, and rearrangement of source input as necessary.

The package runs on IBM 360 Series (under OS, DOS, TOS), IBM 1400 Series, IBM 7090 Series, RCA Spectra 70 Series (under TDOS), Honeywell 200 Series; requires one tape or disc and printer, microfilm, or plotter. AUTOFLOW is a single, multiphase program written in BAL.

A permanent license costs $3,000 to $7,000 depending on the language features required.

An unlimited monthly usage license is priced at a flat monthly rate depending on the language features required.

PACKAGE OUTPUT

The AUTOFLOW chart set is produced which includes:

Title Sheet - This listing contains the program name, date, and other pertinent information.

Input Listing - This printout contains a complete 80/80 listing of the input program.

Procedural Statement Label Index - This listing specifies appropriate section and paragraph names, labels, or statement numbers in alphabetic order and provides a quick reference between the source program and the flowchart.
Table of Contents and References - This cross-reference table provides indexing information for locating transfers of control, both within the flowchart and the source program, whether the references are explicit or implicit.

Table of Diagnostics - This listing contains a record of logical-flow errors, incomplete paths, missing references, and other programming errors.

Flowchart - Each flowchart covers two consecutive printer sheets and can contain up to four columns of flow paths. The symbols on each page are numbered consecutively.

SPECIAL LISTINGS FOR COBOL:

COBOL Diagnostic Analysis - Analysis of the COBOL program is performed identifying logical flow errors, as well as syntax errors.

Procedure Division Analysis - Provides a summary of the various vital activities which take place in the Procedure Division.

Data Division Analysis - (Data Cross Reference) shows the flowchart locations and source sequence numbers for each data name reference in the program.

Data Record Map - Presents a descriptive layout of all group and elementary items in the records within all sections of the Data Division.
Data Division Index - Contains all data and mnemonic name items sequenced alphanumerically, as well as pertinent information for each item.

High Level Flowchart - The COMPRESS facility of Autoflow system enables a user to control the level of flowchart detail.

SPECIAL LISTINGS FOR ASSEMBLY:

EQU Statements - A chronological collection of special symbols to represent all EQU statements used in the source program.

Constants and Storage Areas Listings - A listing of all constants and storage areas.

Modified Tag Summary - A listing of all modified tag references, as well as their location.

Literal Summary - A listing of all literals used in the source program as well as the sequence number in which each literal appears.

Macro Usage Summary - A listing of all macros used in the program, of where they were invoked, and of where the applicable definition is located.

SPECIAL LISTINGS FOR PL/I:

On-Unit Action Blocks - Statements which comprises interrupt condition specifications are flowcharted as separate units.

Called Procedures Cross Reference - This chart provides a summary display of all CALLED entry points in the source input.
Signalled On-Unit Action Blocks - This chart graphically represents all signalled interrupt conditions and their points of reference.

Label Assignment Cross Reference - This is a chart to illustrate the effect of label variables used in GO TO statements on the logical flow of a program.

Duplicate Declaration Map - Multiple declaration of an identifier are listed.

Condition Prefix Map - This illustrates the physical placement of condition prefixes within the nested procedures and BEGIN blocks in the source input.

Declaration Statements - This listing displays non-procedural declaration statements in the order of their appearance in the source input.

GET/PUT, FORMAT Statements - These type statements are printed in this listing.

Note: Each language has the CHART option which enables the user to control the level of detail in the flowchart.

Comment: Autoflow seems to be the most complete and successful proprietary automotive documentation package on the market today. It by far has the most installations of any of the documentation packages, and has more features than any other package.
<table>
<thead>
<tr>
<th>PC. RX</th>
<th>NAME</th>
<th>NAME</th>
<th>PC. RX</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.19</td>
<td>ADD-TRANS</td>
<td>3.03 END-OF-CODE-Routine</td>
<td>7.14</td>
<td>MOVE-KEY</td>
</tr>
<tr>
<td>3.20</td>
<td>CHANGE-ADDR</td>
<td>2.20 END-TH-JMP</td>
<td>5.23</td>
<td>NOTE</td>
</tr>
<tr>
<td>1.26</td>
<td>CHANGE-CITY</td>
<td>5.02 END-TH-END</td>
<td>4.01</td>
<td>PUT-MASTER</td>
</tr>
<tr>
<td>3.04</td>
<td>CHANGE-ERROR</td>
<td>5.12 END-TO-EOF-EXIT</td>
<td>4.07</td>
<td>PUT-MASTER-EXIT</td>
</tr>
<tr>
<td>5.20</td>
<td>CHANGE-NAME</td>
<td>2.27 EOJ</td>
<td>7.05</td>
<td>READ-NEXT-TRANS</td>
</tr>
<tr>
<td>3.27</td>
<td>CHANGE-RA</td>
<td>2.28 EOJ</td>
<td>2.01</td>
<td>STATE</td>
</tr>
<tr>
<td>5.27</td>
<td>CHANGE-REP</td>
<td>3.19 ERROR-PROC</td>
<td>7.19</td>
<td>TEST-PUTAREA</td>
</tr>
<tr>
<td>2.29</td>
<td>COMPARE-INPUTS</td>
<td>4.17 GET-MASTER-EXIT</td>
<td>3.11</td>
<td>TEST-TRANS-CODE</td>
</tr>
<tr>
<td>5.22</td>
<td>DELETE-TRANS</td>
<td>5.01 GET-MASTER-EXIT</td>
<td>2.31</td>
<td>TRANS-LOW</td>
</tr>
<tr>
<td>5.10</td>
<td>END-OF-ACCT-EXIT</td>
<td>4.08 GET-MASTER-Routine</td>
<td>2.11</td>
<td>TRANS-EOF</td>
</tr>
<tr>
<td>5.07</td>
<td>FMT-OF-ACCT-Routine</td>
<td>4.09 GET-MASTER-Routine-1</td>
<td>2.30</td>
<td>TRANS-LOW</td>
</tr>
<tr>
<td>3.09</td>
<td>END-OF-CODE-EXIT</td>
<td>4.15 MASTER-EOF</td>
<td>3.16</td>
<td>TRANSACTION-CODE-ERROR</td>
</tr>
</tbody>
</table>
REMARKS. AN AUTOFLOW CHART PRODUCED FROM COBOL SOURCE INPUT
MAY CONTAIN UP TO FIVE PARTS, THE LISTING OF THE SOURCE
INPUT, THE TABLE OF CONTENTS AND CROSS REFERENCE LISTING,
THE TABLE OF DIAGNOSTICS, A 'REMARKS' PORTION AND THE FLOW
CHART PRODUCED FROM THE PROCEDURE DIVISION OF THE SOURCE
PROGRAM.

THE LISTING OF THE SOURCE INPUT IS OPTIONAL, SELECTED BY
THE USER IN HIS PARAMETER CARD. THE INPUT LIST OPTION
HAS NOT BEEN INVOKED FOR THE AUTOFLOW CHARTING OF THIS
PROGRAM.

THE TABLE OF CONTENTS INDICATES THE CHART LOCATION OF
EACH SECTION OR PARAGRAPH WITHIN THE PROGRAM AS WELL AS
ALL REFERENCES TO THE SECTION OR PARAGRAPH PRODUCED FROM
THE GO TO, PERFORM, ALTER, AND PROCESS VERBS. IT ALSO
INDICATES THE SOURCE CARD NUMBER [IN THE NAME FIELD] OF
ALTER STATEMENTS ALONG WITH THE CHART LOCATION OF THE
STATEMENT BEING ALTERED. CROSS REFERENCES PRODUCED BY
LOCAL DECISIONS, THOSE DECISIONS WHOSE PATHS COME
TOGETHER AT "NEXT SENTENCE", ARE INDICATED WITH A BLANK
NAME FIELD.

WHEN PRESENT, THE 'REMARKS' STATEMENT IN THE
IDENTIFICATION DIVISION WILL APPEAR AS INTRODUCTORY
NARRATIVE IN THE TEXT FORMAT OF AN AUTOFLOW FLOW CHART.
IF NO 'REMARKS' STATEMENT IS PRESENT, THIS NARRATIVE
CHART IS NOT PRESENT. THE MATERIAL CURRENTLY BEING
PRINTED IS AN EXAMPLE OF THE 'REMARKS' STATEMENT NARRATIVE
CHART.

EACH PROCEDURE DIVISION 'SECTION' PRODUCES A CHART, WHOSE
CHART TITLE IS THE SECTION NAME. THIS CHART TITLE APPEARS
IN THE HEADING OF EACH FLOW CHART PAGE PRODUCED FOR THIS
SECTION'S PROCEDURES. AS EACH SECTION BECOMES A CHART,
THE FLOW CHARTS FOR ANY SECTION MAY BE LOCATED FROM THE
TABLE OF CONTENTS.

WHEN A PROGRAM IMPLIES A BRANCH FROM ONE SECTION TO THE
NEXT, A BRANCH SYMBOL IS GENERATED IN THE FLOW CHART.

THIS PROGRAM IS INTENDED ONLY AS A SAMPLE OF A PROGRAM
WHICH MIGHT BE RUN UNDER AUTOFLOW. IT PROBABLY IS NOT
SUBJECT TO CLOSE SCRUTINY FOR LOGICAL PROGRAM CAPABILITIES.
PERFORMED PROCEDURES

FROM-TO-JJ2
PERFORMED AT 0027400 2.27 IN EO31

GET-MASTER-ROUTINE
PERFORMED AT 0021200 2.04 IN START
0018600 3.13 IN TEST-TRANS-CODE

PUT-MASTER
PERFORMED AT 0019400 2.16 IN TEST-OUTAREA
0015900 2.24 IN COMPARE-INPUTS
0027100 2.72 IN END-OF-JOB

FILE ACTIVITY

MASTER-IN
0012000 2.04 IN START
0015900 2.04 IN GET-MASTER-ROUTINE-1

GET INPUT MASTER-IN

MASTER-OUT
0012000 2.04 IN START
0019400 2.04 IN PUT-MASTER-ROUTINE-1

OPEN OUTPUT MASTER-OUT

TRANSACTION-OUT
0012000 2.04 IN START
0015900 2.04 IN GET-TRANSACTION-OUT

TRANSACTION-OUT

STOP SUMMARY
0012700 2.10 IN READ-TRANSACTION-OUT
0013500 2.11 IN END-OF-JOB

DISPLAY TRANSACTION-FILE OUT-OF-SEQ UPON CONSOLE

DISPLAY TRANSACTION-FILE OUT-OF-SEQ UPON CONSOLE

CALLED PROCEDURES
ENTRY
CALLED FROM 0019400 3.18 IN ERROR-PROC
-54

SFt - SAMPif

AtIilltn CHATI

crATh
064RF&4F

C04Aj27O
C.'0

trfFkINCFs

OTAz *A,,R

IT

C04'00

CP A

016TO

7..31

0009o0
0ODROO

C MA K
C#At

02000

4.05

01 000

1.01

CMK

016OlbO00

2.20

cURg-#Io~T IN

O8oA'0n

046o00

.21
4.15_

(0002741
018500

4.11-A
3.12

OOq000

OUR 00
008000
onOnoO

cORt

PAGE

07o200

5
404
4:16

021T00

4.11-A

024000

022300

4.04-A

025300

S.05-A

023 00

4.12

CUR- A TIN-ACCT
CUQR-A"SI

N-COOt

CUQD-MASIIN-KFY

OOROO

STANDAPDO

Nno. AND PAGFRI&aI

ISfOUtNoC

CUR-VA"&SIIN-PEFeR
0(1500
Oos0o
009100

CU00-VASTI-tVPFF
cuqq-#AsTl%-TvPv
CURR-"ASTOIIIT

014900

2.14

009700
ooRIoO
009100
OtqTO0

CUR. "ASTrTU7-ACCI

027220

5.22 -

OL8500

3.12-A

CTJq-.4ASTfnlT-COTE
CURR-MASIOUT-KEY

000000
OOPO
0OOIno0

M

10002641

4.00-N

CURQ- MAST ~JT-RFFES
CUQ F-M#A$TOiT-I VPE
014000

InF-Sw

011100
020600

EARO'F-TYPf
003100

INACCT

003700

tNAD00SF$'

003 OD

IN£1TY

003400

It4CnDE

2.12

. 0A
o 00'9-"

6.06

024600

4.1s-A

025500

S.01-#

014200

2.1 3-

022400

014000
20 00o

o 0.16-A4

019600

3.ol-U

018400

3.12

022200

4.0'

023500

O

404-00

~:S,

81:88

82t988

4:0

01150S * 8S
823

026900

5.20 A

0

026500

5.18-9

001600
003300
CO04QOO

INOEF

003~00

INSTATE

003200

INTYDE

0040001
002700

INZIP
MASTER-1N

004.100

#ASTEa-nlJT7

00200

MAST IN

001000

MASTIN-KFT

00300

MASTIN-NAAf-AOOPESS

004600

MASTOUT

012600

2.01

01000

4.0

021700

4.11
5.13-N
9.05-A
3.00-9
2.23
3.28-A

005200

MASTOUT-NAAE-A0nqls%

004 AOO

'3JTACC I

026300
025200
011SOO
01S400

005500

nUTADLE'SS

021000

009700

nUTARFA-S¥

0141008

Oo 700

A&STnUT-KST

S025
IY

005600

OUTC

005100

nRTCODA.

00300n

1UTNAME

005000

MUTRFF

O5 100

OUTSTATF

0

2.5ol
3

020800
076400
027000

3.26-A
5.15

020600

0.25-A

021600
023800

3.21-A
4.12

10003001

5.21-A

011400
01000

2.08
3.01

200

;:tt'

s.0-OS

025200

58

8001800
0252
0700

4.09-A
N

00

026500

5.16-N

021100
024000

3.28-N
4.16-A

014500

2.14-A

220

5.22-

5.21-A

OUTYFE

UOSPO0

007 P00

OUT/IP
PREV-MASTIN

007100

FPRV-MA$TT

01

o000

UT

PREV-MASTnlIT

01 05osoo00

PIZV-MASTOUT-ACCT
"^TAN-NIAf-VEIES
*RF V-4ASTUtJT-COOE

010100

PRFV-MA&STOIIT-KEY

On400oo

PCEV-VASTOUT-RFFER

010300

PREV-.ASTnUT-TYPE

OO0OO0

PRFV-TRANSIN

011600

TRC

011500

TACCT-CnOE

011100
011 300

TACCT-NO
TACCT-TIypF

0l1 350

TAK

022000

4.03

o10o00

012200
FAN

01600

2.031

01100

TAt

01 ?bOO

0.06

018000

3.14

02S400

9.06

012000

026400

5.14

T ADIORss

0.21
2.23
1.26

3.22

00?200

020100
015500
021300

020000

00t.000

TC2
TRAECT

014900

2.16
2.07-A
3.16-A
2.02

019800
013400

2. 23
2.04

014300

2.13--

013100

2.05-"

013900

2.11

011200

01

b0600

01 OqOO
Cl 17o00

TRA'A-ACCT
IVAN-ACCT-KFI
TRAN-SQ

-KTEY

01000
011 !00

TIANS-CTOFE

009000

TRANS-IN

001000

TAANS-NAME-AOORESS

C06400

TRANSIN

006.00

TANS IN-KEY

013000
10002351
012700
026300

*.13

014S00

2.14


<table>
<thead>
<tr>
<th>ID</th>
<th>FIELD NAME</th>
<th>OFFSET</th>
<th>LENGTH</th>
<th>DEC</th>
<th>OCCURS</th>
<th>CLASS</th>
<th>USAGE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>002400</td>
<td>MASTER-IN</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>002400</td>
<td>MASTER-OUT</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>004400</td>
<td>MASTER-IN</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>004400</td>
<td>MASTER-OUT</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>009400</td>
<td>TRANS-IN</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>009400</td>
<td>TRANS-OUT</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
</tbody>
</table>

**FILE SECTION**

<table>
<thead>
<tr>
<th>ID</th>
<th>LEVEL</th>
<th>FIELD NAME</th>
<th>OFFSET</th>
<th>LENGTH</th>
<th>DEC</th>
<th>OCCURS</th>
<th>CLASS</th>
<th>USAGE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>002400</td>
<td>0</td>
<td>MASTER-IN</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>002400</td>
<td>0</td>
<td>MASTER-OUT</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>004400</td>
<td>0</td>
<td>MASTER-IN</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>004400</td>
<td>0</td>
<td>MASTER-OUT</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>009400</td>
<td>0</td>
<td>TRANS-IN</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td>009400</td>
<td>0</td>
<td>TRANS-OUT</td>
<td>0000</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
</tbody>
</table>

**DATA RECORD MAP**

**AUTOFLOW CHART SET**

**SAMPLE STANDARD COBOL**

**PAGE 1**

**LABEL RECORD IS STANDARD, BLOCK CONTAINS 2000 CHARACTERS, RECORD CONTAINS 100 CHARACTERS**

<table>
<thead>
<tr>
<th>OFFSET</th>
<th>LENGTH</th>
<th>DEC</th>
<th>OCCURS</th>
<th>CLASS</th>
<th>USAGE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-4</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
</tbody>
</table>

**LABEL RECORD IS STANDARD, BLOCK CONTAINS 2000 CHARACTERS, RECORD CONTAINS 100 CHARACTERS**

<table>
<thead>
<tr>
<th>OFFSET</th>
<th>LENGTH</th>
<th>DEC</th>
<th>OCCURS</th>
<th>CLASS</th>
<th>USAGE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-4</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
</tbody>
</table>

**LABEL RECORD IS STANDARD, BLOCK CONTAINS 2000 CHARACTERS, RECORD CONTAINS 100 CHARACTERS**

<table>
<thead>
<tr>
<th>OFFSET</th>
<th>LENGTH</th>
<th>DEC</th>
<th>OCCURS</th>
<th>CLASS</th>
<th>USAGE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-4</td>
<td>10</td>
<td>0</td>
<td></td>
<td>A/N</td>
<td>DISPLAY</td>
<td></td>
</tr>
</tbody>
</table>
PROCEDURE DIVISION.

NOTE
A. THE MATCHING OF BOYS & GIRLS FOR HIGH SCHOOL DANCE...

START.

NOTE
6. I READ FIRST BOY AND FIRST GIRL.
7. T GIRL-COUNTER IS A COUNTER USED TO CONTROL THE LOOPING
   THRU THE GIRLS INPUT FILE.

PREMAY.

NOTE
11. BY (SECONDARY) PRIMARY MATCH EQUAL??.
12. I, PUNCH PUNCH DATE-NAME CARD.

READ-A-BOY.

NOTE
15. I READ ANOTHER BOY.
16. BY (PRIMARY) EOF?
17. I CLOSE ALL FILES.
18. H END OF JOB.

SECONDARY.

NOTE
21. P ADD I TO GIRL-COUNTER.
22. ON (PREPARE) ENTIRE GIRLS FILE SEARCHED?

READ-GIRL.

NOTE
25. I READ ANOTHER GIRL.
26. ON (EOF-GIRL) EOF?
27. 8 (PRIMARY) GO TO PRIMARY MATCHING PROCEDURE.

PREPARE.

NOTE
30. T PREPARE FOR SECONDARY MATCHING PROCEDURE.
31. P CLEAR GIRL-COUNTER.

READ-GIRL2.

NOTE
34. I READ ANOTHER GIRL.
35. ON (EOF-GIRL2) EOF?
36. 8 (PUNCH) SECONDARY MATCH EQUAL??.
37. P ADD I TO GIRL-COUNTER.
38. BY (READ-GIRL2) ENTIRE GIRLS FILE SEARCHED?
39. I PRINT NO-MATCH MESSAGE "SORRY FELLA".
40. 8 (READ-A-BOY).
41. EOF GIRLS.

NOTE
43. I CLOSE GIRLS TAPE FILE.
44. I OPEN GIRLS TAPE FILE.
45. 8 (PRIMARY-READ-GIRLS) WHICH MATCHING PROCEDURE?
46. 8 (READ-GIRLS) PRIMARY.
CHART TITLE - THE MATCHING OF BOYS & GIRLS FOR HIGH SCHOOL DANCE.

START

READ FIRST BOY

READ FIRST GIRL

GIRL-COUNT IS A

READ THE GIRLS INPUT

COUNT

COUNT

PRIMARY

SECONDARY

ADD 1 TO GIRL-COUNTER

ENTIRE

GIRED FILE

SEARCHED

REACHED-GIRL

YES

NO

EDRF

EDRF

EDRF

EDRF

EDRF

REACHED-GIRL

PRINT NO-MATCH MESSAGE GIVE FELLA

CLOSE GIRLS TAPE FILE

OPEN GIRLS TAPE FILE

PRINT NO-MATCH MESSAGE GIVE FELLA

PRIMARY

SECONDARY MATCH PROCEDURE

READ FIRST BOY

READ FIRST GIRL

GIRL-COUNT IS A

READ THE GIRLS INPUT

COUNT

COUNT

PRIMARY

SECONDARY

ADD 1 TO GIRL-COUNTER

ENTIRE

GIRED FILE

SEARCHED

REACHED-GIRL

YES

NO

EDRF

EDRF

EDRF

EDRF

EDRF

REACHED-GIRL

PRINT NO-MATCH MESSAGE GIVE FELLA

CLOSE GIRLS TAPE FILE

OPEN GIRLS TAPE FILE

PRINT NO-MATCH MESSAGE GIVE FELLA

PRIMARY
<table>
<thead>
<tr>
<th>PG,IN</th>
<th>NAME</th>
<th>PG,IN</th>
<th>NAME</th>
<th>PG,IN</th>
<th>NAME</th>
<th>PG,IN</th>
<th>NAME</th>
<th>PG,IN</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.16</td>
<td>ACE</td>
<td>2.14</td>
<td>DELIC</td>
<td>2.01</td>
<td>INLYC</td>
<td>4.01</td>
<td>INDEEP</td>
<td>3.13</td>
<td>WRITE</td>
</tr>
<tr>
<td>1.19</td>
<td>ACE</td>
<td>2.23</td>
<td>DEL15</td>
<td>2.04</td>
<td>INRST</td>
<td>1.08</td>
<td>INDEEP</td>
<td>3.22</td>
<td>WRITE01</td>
</tr>
<tr>
<td>1.26</td>
<td>ACE3</td>
<td>2.20</td>
<td>DEL20</td>
<td>2.11</td>
<td>INLIC</td>
<td>1.09</td>
<td>MAIN</td>
<td>3.25</td>
<td>WRITE02</td>
</tr>
<tr>
<td>1.29</td>
<td>ACE10</td>
<td>2.22</td>
<td>DEL30</td>
<td>1.18</td>
<td>INITCH</td>
<td>1.01</td>
<td>NAMCHG</td>
<td>3.17</td>
<td>WRITE10</td>
</tr>
<tr>
<td>1.21</td>
<td>ADC</td>
<td>4.17</td>
<td>EOJ</td>
<td>4.09</td>
<td>INIT01</td>
<td>3.05</td>
<td>NAMP5</td>
<td>8.03</td>
<td>WRITE11</td>
</tr>
<tr>
<td>1.29</td>
<td>ADC10</td>
<td>1.04</td>
<td>INICES</td>
<td>4.13</td>
<td>INIT02</td>
<td>3.05</td>
<td>NAMP10</td>
<td>3.33</td>
<td>WRITE20</td>
</tr>
<tr>
<td>5.01</td>
<td>BLANK</td>
<td>1.06</td>
<td>INCO</td>
<td>5.01</td>
<td>INDOC</td>
<td>3.10</td>
<td>NAMC0</td>
<td>3.30</td>
<td>WRITE30</td>
</tr>
<tr>
<td>2.13</td>
<td>DEL</td>
<td>1.07</td>
<td>INNC10</td>
<td>1.01</td>
<td>INVINC</td>
<td>5.04</td>
<td>SPACE</td>
<td>5.02</td>
<td>WRITE44</td>
</tr>
<tr>
<td>2.15</td>
<td>DEL05</td>
<td>2.09</td>
<td>INAC20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reproduced from best available copy.
<table>
<thead>
<tr>
<th>CARD/ LOCATION</th>
<th>DIAGNOSTIC</th>
<th>PAGE/ PAGE</th>
<th>VALUE</th>
<th>UNDEFINED</th>
<th>REG. VALUE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>K00003/1269</td>
<td>1.07</td>
<td>1.07</td>
<td></td>
<td>UNDEFINED</td>
<td>REG. VALUE</td>
<td></td>
</tr>
<tr>
<td>K00012/1880</td>
<td>5.02</td>
<td>5.02</td>
<td></td>
<td>UNDEFINED</td>
<td>REG. VALUE</td>
<td></td>
</tr>
</tbody>
</table>
09/10/70  INPUT LISTING
AUTOFLOW CHART SET - SAMPLE  ASSEMBLY WITH CHART CODES

CHART/ASSEMBLY - (LIST1)

<table>
<thead>
<tr>
<th>CARD NO</th>
<th>****</th>
<th>CONTENTS</th>
<th>****</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CVT</td>
<td>TITLE 'CCONVERT ALPHA FORMAT NUMBER TO DECIMAL'</td>
<td>BY 0010</td>
</tr>
<tr>
<td>2</td>
<td>CECCVT</td>
<td>CSECT (DECCVT)</td>
<td>EN 0020</td>
</tr>
<tr>
<td>3</td>
<td>CSECT</td>
<td>TITLE 'PRINT CHART SET'</td>
<td>EN 0030</td>
</tr>
<tr>
<td>4</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0040</td>
</tr>
<tr>
<td>5</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0050</td>
</tr>
<tr>
<td>6</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0060</td>
</tr>
<tr>
<td>7</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0070</td>
</tr>
<tr>
<td>8</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0080</td>
</tr>
<tr>
<td>9</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0090</td>
</tr>
<tr>
<td>10</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0100</td>
</tr>
<tr>
<td>11</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0110</td>
</tr>
<tr>
<td>12</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0120</td>
</tr>
<tr>
<td>13</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0130</td>
</tr>
<tr>
<td>14</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0140</td>
</tr>
<tr>
<td>15</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0150</td>
</tr>
<tr>
<td>16</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0160</td>
</tr>
<tr>
<td>17</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0170</td>
</tr>
<tr>
<td>18</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0180</td>
</tr>
<tr>
<td>19</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0190</td>
</tr>
<tr>
<td>20</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0200</td>
</tr>
<tr>
<td>21</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0210</td>
</tr>
<tr>
<td>22</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0220</td>
</tr>
<tr>
<td>23</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0230</td>
</tr>
<tr>
<td>24</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0240</td>
</tr>
<tr>
<td>25</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0250</td>
</tr>
<tr>
<td>26</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0260</td>
</tr>
<tr>
<td>27</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0270</td>
</tr>
<tr>
<td>28</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0280</td>
</tr>
<tr>
<td>29</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0290</td>
</tr>
<tr>
<td>30</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0300</td>
</tr>
<tr>
<td>31</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0310</td>
</tr>
<tr>
<td>32</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0320</td>
</tr>
<tr>
<td>33</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0330</td>
</tr>
<tr>
<td>34</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0340</td>
</tr>
<tr>
<td>35</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0350</td>
</tr>
<tr>
<td>36</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0360</td>
</tr>
<tr>
<td>37</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0370</td>
</tr>
<tr>
<td>38</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0380</td>
</tr>
<tr>
<td>39</td>
<td>MVC</td>
<td>TITLE 'BEGIN INVALUE'</td>
<td>EN 0390</td>
</tr>
</tbody>
</table>

DATA HAS BEEN VALIDITY CHECKED
PACK OUTVALD,INVALC
UNTIL OUTVALD=10,INVALC
CONVERT VALUE TO DECIMAL

EXIT RETURN (14,12,RC=15)

SPACE 6

INVALUE DC CL8
OUTVALD DC P'00000000000'

END
AUTOFLOW CHART SET - SAMPLE ASSEMBLY WITH CHART CONCE

DECVT

ROUTINE ENTRANCE

POINT

01

VALIDITY CHECK
ALL DIGITS AND
LEADING ZERO FILL
FIELD

02

ALL DIGITS: YES
VALID DECIMAL: NO

03

SET RC = 8

04

EXIT

RETURN 14,12,RC=15

05

NOTE
DATA HAS NOW BEEN
VALIDITY CHECKED

06

CONVERT VALUE TO
DECIMAL

07

SET RC = 0

08

YES
VALUE ZERO

09

SET RC = 4

10

EXIT

RETURN 14,12,RC=15
<table>
<thead>
<tr>
<th>PG.NE</th>
<th>NAME</th>
<th>PG.NE</th>
<th>NAME</th>
<th>PG.NE</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.01</td>
<td>UN-UNIT01</td>
<td>1.30</td>
<td>MERG1</td>
<td>2.00</td>
<td>MAIN2</td>
</tr>
<tr>
<td>5.11</td>
<td>UN-UNIT02</td>
<td>1.38</td>
<td>MERG1A</td>
<td>2.10</td>
<td>MAIN22</td>
</tr>
<tr>
<td>5.18</td>
<td>UN-UNIT03</td>
<td>1.25</td>
<td>MERG2</td>
<td>2.20</td>
<td>MAIN3</td>
</tr>
<tr>
<td>1.26</td>
<td>EXIT</td>
<td>1.32</td>
<td>MERG3</td>
<td>2.30</td>
<td>MAIN4</td>
</tr>
<tr>
<td>1.17</td>
<td>GETINC</td>
<td>5.01</td>
<td>PRIME1</td>
<td>4.01</td>
<td>PUTINS</td>
</tr>
<tr>
<td>1.16</td>
<td>GETINS</td>
<td>5.06</td>
<td>PRIME2</td>
<td>4.01</td>
<td>PUTINS2</td>
</tr>
<tr>
<td>1.13</td>
<td>GETMAIN</td>
<td>5.07</td>
<td>PRIME3</td>
<td>4.01</td>
<td>PUTINS3</td>
</tr>
<tr>
<td>1.01</td>
<td>MERGE</td>
<td>5.08</td>
<td>PRIME4</td>
<td>4.12</td>
<td>PUTINS4</td>
</tr>
<tr>
<td>1.20</td>
<td>MFCGO</td>
<td>2.01</td>
<td>PRIME5</td>
<td>7.01</td>
<td>PUTINS5</td>
</tr>
<tr>
<td>CARD ID</td>
<td>PAGE/DMX</td>
<td>DIAGNOSTIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000048</td>
<td>5.08</td>
<td>UNDEFINED - SMRT EXTERNAL REFERENCE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000049</td>
<td>5.09</td>
<td>UNDEFINED - SMRT EXTERNAL REFERENCE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000070</td>
<td>9.10</td>
<td>UNDEFINED - RELEASE EXTERNAL REFERENCE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
PROCEDURE MERGE

00000021 SI2 OF MAIN
00000073 HMX OF MAIN
00000073 SHT OF MAIN
00000073 CNT OF MAIN
00000021 NAM OF MAIN
00000023 SFC OF MAIN
00000023 DST OF MAIN
00000023 TET OF MAIN
00000095 SIE OF INS
00000095 HMX OF INS
00000095 SHT OF INS
00000095 CNT OF INS
00000095 NAM OF INS
00000095 SFC OF INS
00000095 DST OF INS
00000095 TET OF INS
00000045 SIE OF CHG
00000045 NAM OF CHG
00000045 DST OF CHG

PROCEDURE PITMAIN

END PROCEDURE

PROCEDURE PITINS

END PROCEDURE

END PROCEDURE
DECLARE MAINFILE FILE INPUT BUFFERED ENVIRONMENT MEDIUM (SYSFILE,255) # (1) #
DECLARE INFILE FILE INPUT BUFFERED ENVIRONMENT MEDIUM (SYSFILE,255) # (2) #
DECLARE OUTFILE FILE OUTPUT BUFFERED ENVIRONMENT MEDIUM (SYSFILE,255) # (3) #
DECLARE PRINTED FILE OUTPUT PRINT STREAM ENVIRONMENT MEDIUM (SYSFILE,255) # (4) #
DECLARE DEFSRCUGR15FILE RT (S) # (5) # NUM FIXED DECIMAL (2,01) # (6) #
DECLARE CLS FIRST FIXED DECIMAL (7,01,2) NUM FIXED DECIMAL (2,01,2) # (7) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (8) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (9) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (10) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (11) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (12) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (13) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (14) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (15) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (16) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (17) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (18) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (19) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (20) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (21) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (22) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (23) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (24) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (25) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (26) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (27) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (28) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (29) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (30) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (31) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (32) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (33) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (34) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (35) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (36) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (37) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (38) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (39) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (40) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (41) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (42) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (43) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (44) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (45) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (46) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (47) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (48) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (49) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (50) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (51) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (52) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (53) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (54) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (55) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (56) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (57) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (58) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (59) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (60) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (61) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (62) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (63) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (64) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (65) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (66) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (67) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (68) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (69) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (70) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (71) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (72) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (73) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (74) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (75) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (76) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (77) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (78) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (79) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (80) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (81) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (82) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (83) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (84) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (85) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (86) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (87) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (88) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (89) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (90) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (91) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (92) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (93) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (94) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (95) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (96) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (97) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (98) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (99) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (100) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (101) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (102) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (103) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (104) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (105) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (106) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (107) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (108) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (109) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (110) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (111) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (112) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (113) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (114) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (115) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (116) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (117) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (118) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (119) #
DECLARE 1 CRT FIXED DECIMAL (7,01,2) # (120) #
DECLARE 2 CRT FIXED DECIMAL (7,01,2) # (121) #
CHART TITLE - GET/PUT, FORMAT STATEMENTS

00000009 LIST
00000010 TOTAL
00000178
00000180
00000225
00000227
00000249
00000251

FORMAT (COLUMN 78),A,X (8),P 'ZZZ',X (5),P 'YYY',X
FORMAT (SKIP (2),COLUMN (23),A,COLUMN (41),P COLUMN (61)
\( \text{GET/PUT 001 PUT FILE (PRINTER) EDIT ('MAIN.TXT'=MAIN.TXT) (COLUMN (10),A (9),X (5),A (10))} \)
\( \text{GET/PUT 002 PUT FILE (PRINTER) SKIP (2)} \)
\( \text{GET/PUT 003 PUT FILE (PRINTER) EDIT ('INS.TXT'=INS.TXT) (COLUMN (10),A (9),X (5),A (10))} \)
\( \text{GET/PUT 004 PUT FILE (PRINTER) SKIP (2)} \)
\( \text{GET/PUT 005 PUT FILE (PRINTER) EDIT ('INS.TXT'=INS.TXT) (COLUMN (10),A (9),X (5),A (10))} \)
\( \text{GET/PUT 006 PUT FILE (PRINTER) SKIP (2)} \)
CHART/PL/I MODULE (LIST21)

CARO NO. **** CONTENTS  ****
1 THI_MATCHING_OF_BoYS_AND_GIRLSFOR_A_MATCH_SCHOOL_DANCE; PROCEDURE; 2 START;
3 /*1 READ FIRST BOY AND FIRST GIRL */
4 /*2 GIRL-COUNTER IS A COUNTER USED TO CONTROL THE LOOPING */
5 /*3 THRU THE GIRLS INPUT FILE */
6 PRIMARY:
7 /*4 SECONDARY'S PRIMARY MATCH EQUAL? */
8 /*5 PRIMARY Equals */
9 READ_A_Boy;
10 /*6 READ ANOTHER BOY */
11 /*7 PRIMARY ENDF */
12 /*8 CLOSE ALL FILES */
13 /*9 END OF JOB */
14 SECONDARY:
15 /*10 ADD 1 TO GIRL-COUNTER */
16 /*11 PREPARE ENTIRE GIRLS FILE SEARCHED */
17 READ_Girls;
18 /*12 READ ANOTHER GIRL */
19 /*13 EOF_GIRLS ENDF */
20 /*14 PRIMARY GO TO PRIMARY MATCHING PROCEDURE */
21 PREPARE;
22 /*15 PREPARE FOR SECONDARY MATCHING PROCEDURE */
23 /*16 CLEAR GIRL-COUNTER */
24 READ_Girls;
25 /*17 READ ANOTHER GIRL */
26 /*18 EOF_GIRLS ENDF */
27 /*19 SECONDARY MATCH EQUAL? */
28 /*20 ADD 1 TO GIRL-COUNTER */
29 /*21 PRINT MATCH MESSAGE 'SORRY SELLA' */
30 /*22 READ_A_Boy */
31 /*23 EOF_GIRLS */
32 /*24 CLOSE GIRLS TAPE FILE */
33 /*25 OPEN GIRLS TAPE FILE */
34 /*26 PRINTSECREAD_GirlsWHICH MATCHING PROCEDURE? */
35 /*27 READ_Girls PRIMARY */
36 /*28 EOF_GIRLS */
37 /*29 END THE MATCHING_OF_BoYS_AND_GIRLS FOR A HIGH SCHOOL DANCE */
<table>
<thead>
<tr>
<th>CHART/PL/I MODULE</th>
<th>SAMPLE CHART PL/I FACILITIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHART TITLE - PROCEDURE THE_MATCHING_OF_BOYS_AND_GIRLS_FOR_A_HIGH_SCHOOL_DANCE</td>
<td></td>
</tr>
<tr>
<td>00000002</td>
<td>1.01</td>
</tr>
<tr>
<td>00000004</td>
<td>1.02</td>
</tr>
<tr>
<td>00000004</td>
<td>1.03</td>
</tr>
<tr>
<td>00000009</td>
<td>1.04</td>
</tr>
<tr>
<td>00000014</td>
<td>1.06</td>
</tr>
<tr>
<td>00000017</td>
<td>1.10</td>
</tr>
<tr>
<td>00000024</td>
<td>1.13</td>
</tr>
<tr>
<td>00000032</td>
<td>1.19</td>
</tr>
</tbody>
</table>
COMCHART

GENERAL

COMCHART accepts source programs written in assembly language and COBOL. The package can also take a designer's specifications and produce flowcharts, using a special Design Language that involves a coding technique similar to that of most assembly languages. A listing of the source deck plus a cross-reference listing of all element and procedure names are also generated, but can be suppressed at the user's option.

The package runs on an IBM 360 (OS) with a minimum of 65K bytes of main core storage and 4 sequential files on any devices, IBM 360 (DOS) with a minimum of 65K bytes of core storage and 4 sequential work files (a 2311, a 2314, or 4 tapes), RCA Spectra 70 (TDOS) with a minimum of 65K bytes of main storage and 4 tapes or 1 disc.

The package consists of 3 flowcharter subsystems. The source language is a combination of assembly language and Cobol.

PACKAGE OUTPUT

Flowcharts - Detailer flowchart where each line of source code is represented by a symbol. Each flowchart bears a page number, a user-furnished identification of the program, and the name of the user's organization.

Element Index - (COBOL only) an alphabetical cross-reference index of element names is given. Each index contains each name in the program, its type (e.g. data level, paragraph name, file name), and the name of the procedure which references it.
Procedure Skeleton - This lists each procedure name in the procedure division, together with all procedures referenced by and referring to that procedure.

Source Deck Listing - An 80/80 list of the source deck is printed.

Diagnostics - This report follows the deck listing for a Design Language program, and given a narrative description of errors in the Design Language definition.
SAMPLE TO DEMONSTRATE COMCHART FOR ASSEMBLY

START 0
00010
EQU 0
00015
R0
EQU 1
00020
R1
EQU 2
00025
R2
EQU 3
00030
R3
EQU 4
00035
R4
EQU 5
00040
R5
EQU 6
00045
R6
EQU 7
00050
R7
EQU 8
00055
R8
EQU 9
00060
R9
EQU 10
00065
R10
EQU 11
00070
R11
EQU 12
00075
R12
EQU 13
00080
R13
EQU 14
00085
R14
EQU 15
00090
R15

REGISTER USAGE
00105
R.0;1,2,13,14,15--AS DEFINED BY IBM.
00110
R.10,11--OPEN ADDRESSING.
00115
R.12--SELECTIVE ADDRESSING.
00120
R.3,4,5,6,7,8,9--SPAN-OF-ATTENTION.
00125

FIRST
00130
STM R14,R12,12(R13)
00135
BALR R10,0
00140
SET UP OPEN REGS.
00143
USING *R10,R11
00145
LA R3,4095
00150
LA R11,1(R10,R3)
00155
CNOP 2,4
00160
MVC B(R4,R13),A(*R14)
00165
ST R13,**12
00170
BAL R13,FGO
00175
SET UP SAVE AREA.
00180
DS 18F
00185
FGO
00205
EQU *
00210

READ IN THE SELECTOR TABLE.
00215

OPEN FILEB
00230
GET FILEB,RECB
00235
PICK UP ADR. OF PREVIOUS ENTRY.
00240
L R3,TABTOP
00245
ADD FOR THIS ADR.
00250
LA R3,7(R3)
00255
TABLE EXHAUSTED?
00260
C
00265
TABEND FB
00270
ST R3,TABTOP
00280
IF NOT, STORE CURRENT ENTRY ADR.
00285
MVC D(7,R3),RECB
00290
MOVE IN THE ENTRY.
00295
B FA

TOO MANY SELECTORS.
SAMPLE TO DEMONSTRATE CONCMART FOR ASSEMBLY

START

001 (21)
REGISTER USAGE

005 (21)
4.10--OPEN ADDRESSING.

006 (22)
4.12--SELECTIVE ADDRESSING.

007 (23)
4.13--ADDITION OF 7.8.9.

008 (25)
FIRST

009 (25)

011 (35)
BALR

012 (33)
BAL

013 (35)
CG 73

014
PG AB D0 014
IDENTIFICATION DIVISION.
PROGRAM-ID, 'SAMPLE'.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
SELECT FILE-A ASSIGN 'FILEA' UTILITY.
SELECT FILE-B ASSIGN 'FILEB' UTILITY.
SELECT FILE-C ASSIGN 'FILEC' UTILITY.
DATA DIVISION.
FILE SECTION.
FD FILE-A BLOCK 5 RECORDS RECORDING F LABEL RECORDS OMITTED.
DATA RECORD REC-A.
02 CODE-TEST PICTURE X.
02 NAME-FIELD PICTURE A(20).
02 DATE-FIELD.
03 MONTH-FIELD PICTURE 99.
03 DAY-FIELD PICTURE 99.
03 YEAR-FIELD PICTURE 99.
FD FILE-B BLOCK 1 RECORDS RECORDING F LABEL RECORDS OMITTED.
DATA RECORD REC-B.
01 REC-B PICTURE X(7).
FD FILE-C BLOCK 5 RECORDS RECORDING F LABEL RECORDS OMITTED.
DATA RECORD REC-C.
01 REC-C.
02 CODE-TEST PICTURE X.
02 NAME-FIELD PICTURE A(20).
02 DATE-FIELD.
03 MONTH-FIELD PICTURE 99.
03 DAY-FIELD PICTURE 99.
03 YEAR-FIELD PICTURE 99.
WORKING-STORAGE SECTION.
QUALIFY-SWITCH PICTURE S9 COMPUTATIONAL.
TABLE-LIMIT PICTURE 59(5) COMPUTATIONAL VALUE 0.
TABLE-ENTRY OCCURS 100.
TABLE-CODE PICTURE X.
TABLE-DATE PICTURE 9(6).
PROCEDURE DIVISION.
REMARKS SECTION.
NOTE.
THIS PROGRAM USES A DECK OF SELECTOR CARDS IN THE PARTIAL.
COPYING OF AN INPUT TO AN OUTPUT REEL OF TAPE.
THE PROCESSING PROCEEDS BY CHECKING EACH INPUT RECORD
AGAINST EACH SELECTOR CARD, AND WRITING THE QUALIFYING
RECORDS--REFORMATTED--TO THE OUTPUT FILE.
The-FIRST SECTION.
OPEN INPUT FILE-B.
MOVE 0 TO TABLE-USED.
READ-IN.
READ FILE-B AT END GO TO FINISH OF THE-FIRST.
THIS PROGRAM USES A DECK OF SELECTOR CARDS IN THE PARTIAL COPYING OF AN INPUT TO AN OUTPUT REEL OF TAPE.

THE PROCESSING PROCEEDS BY CHECKING EACH INPUT RECORD AGAINST EACH SELECTOR CARD AND WRITING THE QUALIFYING RECORDS—REFORMATTED—TO THE OUTPUT FILE.

THE PROGRAM USES A DECK OF SELECTOR CARDS IN THE PARTIAL COPYING OF AN INPUT TO AN OUTPUT REEL OF TAPE.

THE PROCESSING PROCEEDS BY CHECKING EACH INPUT RECORD AGAINST EACH SELECTOR CARD AND WRITING THE QUALIFYING RECORDS—REFORMATTED—TO THE OUTPUT FILE.

THE PROCESSING PROCEEDS BY CHECKING EACH INPUT RECORD AGAINST EACH SELECTOR CARD AND WRITING THE QUALIFYING RECORDS—REFORMATTED—TO THE OUTPUT FILE.
<table>
<thead>
<tr>
<th>TYPE</th>
<th>STMT</th>
<th>NAME</th>
<th>REFERENCES IN DECK SEQUENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>O2</td>
<td>14</td>
<td>CODE-TEST</td>
<td></td>
</tr>
<tr>
<td>O2</td>
<td>16</td>
<td>DATE-FIELD</td>
<td></td>
</tr>
<tr>
<td>O2</td>
<td>26</td>
<td>DATE-TEST</td>
<td></td>
</tr>
<tr>
<td>O3</td>
<td>18</td>
<td>DAY-FIELD</td>
<td></td>
</tr>
<tr>
<td>O3</td>
<td>29</td>
<td>DAY-FIELD</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>96</td>
<td>E-Q</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>73</td>
<td>EACH-INPUT</td>
<td></td>
</tr>
<tr>
<td>SECT</td>
<td>95</td>
<td>EACH-Q</td>
<td></td>
</tr>
<tr>
<td>FD</td>
<td>11</td>
<td>FILE-A</td>
<td></td>
</tr>
<tr>
<td>FD</td>
<td>20</td>
<td>FILE-B</td>
<td></td>
</tr>
<tr>
<td>FD</td>
<td>23</td>
<td>FILE-C</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>65</td>
<td>FINISH</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>83</td>
<td>FINISH</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>50</td>
<td>INITIALIZE</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>70</td>
<td>INITIALIZE</td>
<td></td>
</tr>
<tr>
<td>O3</td>
<td>17</td>
<td>MONTH-FIELD</td>
<td></td>
</tr>
<tr>
<td>O3</td>
<td>28</td>
<td>MONTH-FIELD</td>
<td></td>
</tr>
<tr>
<td>O2</td>
<td>15</td>
<td>NAME-FIELD</td>
<td></td>
</tr>
<tr>
<td>O2</td>
<td>30</td>
<td>NAME-FIELD</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>87</td>
<td>Q-A</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>89</td>
<td>Q-B</td>
<td></td>
</tr>
<tr>
<td>SECT</td>
<td>86</td>
<td>QUALIFIER</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>32</td>
<td>QUALIFY-SWITCH</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>42</td>
<td>R-M</td>
<td></td>
</tr>
<tr>
<td>SECT</td>
<td>41</td>
<td>RE-MARKS</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>53</td>
<td>READ-IN</td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>13</td>
<td>REC-A</td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>22</td>
<td>REC-B</td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>25</td>
<td>REC-C</td>
<td></td>
</tr>
<tr>
<td>PARA</td>
<td>55</td>
<td>STOP-READ</td>
<td></td>
</tr>
<tr>
<td>O3</td>
<td>38</td>
<td>TABLE-CODE</td>
<td></td>
</tr>
<tr>
<td>O3</td>
<td>39</td>
<td>TABLE-DATE</td>
<td></td>
</tr>
<tr>
<td>O2</td>
<td>37</td>
<td>TABLE-ENTRY</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>33</td>
<td>TABLE-LIMIT</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>34</td>
<td>TABLE-USED</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>35</td>
<td>TABLE-X</td>
<td></td>
</tr>
<tr>
<td>SECT</td>
<td>49</td>
<td>THE-FIRST</td>
<td></td>
</tr>
<tr>
<td>SECT</td>
<td>69</td>
<td>THE-SECOND</td>
<td></td>
</tr>
</tbody>
</table>
**SAMPLE TO DEMONSTRATE CONCHART FOR COBOL PROCEDURE SKELETON**

<table>
<thead>
<tr>
<th>PROCEDURES IN DECK SEQUENCE</th>
<th>REFERENCES IN DECK SEQUENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>41 RE-MARKS</td>
<td>FROM 57 USE-RECB TO 63 FINISH</td>
</tr>
<tr>
<td>42 R-M</td>
<td>FROM 57 USE-RECB TO 57 USE-RECB</td>
</tr>
<tr>
<td>49 THE-FIRST</td>
<td>FROM 55 STOP-READ TO 53 READ-IN</td>
</tr>
<tr>
<td>50 INITIALIZE</td>
<td>FROM 55 STOP-READ TO 55 STOP-READ</td>
</tr>
<tr>
<td>53 READ-IN</td>
<td>FROM 53 READ-IN</td>
</tr>
<tr>
<td>53 STOP-READ</td>
<td></td>
</tr>
<tr>
<td>57 USE-RECB</td>
<td></td>
</tr>
<tr>
<td>65 FINISH</td>
<td></td>
</tr>
<tr>
<td>69 THE-SECOND</td>
<td></td>
</tr>
<tr>
<td>70 INITIALIZE</td>
<td>FROM 73 EACH-INPUT TO 73 EACH-INPUT</td>
</tr>
<tr>
<td>73 EACH-INPUT</td>
<td>TO 73 EACH-INPUT TO 83 FINISH</td>
</tr>
<tr>
<td>83 FINISH</td>
<td>FROM 86 QUALIFIER TO 73 EACH-INPUT</td>
</tr>
<tr>
<td>86 QUALIFIER</td>
<td>FROM 73 EACH-INPUT</td>
</tr>
<tr>
<td>87 Q-A</td>
<td></td>
</tr>
<tr>
<td>89 Q-B</td>
<td></td>
</tr>
<tr>
<td>95 EACH-Q</td>
<td></td>
</tr>
<tr>
<td>96 E-Q</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FROM 89 Q-A</td>
</tr>
</tbody>
</table>

**PROCEDURES IN DECK SEQUENCE**

- 41 RE-MARKS
- 42 R-M
- 49 THE-FIRST
- 50 INITIALIZE
- 53 READ-IN
- 53 STOP-READ
- 57 USE-RECB
- 65 FINISH
- 69 THE-SECOND
- 70 INITIALIZE
- 73 EACH-INPUT
- 83 FINISH
- 86 QUALIFIER
- 87 Q-A
- 89 Q-B
- 95 EACH-Q
- 96 E-Q
DIAGNOSTIC MESSAGE COUNT 0
This program uses a set of selector cards to choose specified records from a reel of magnetic tape and copy them to a second tape. Reformatting them for a sort in the process.

File purpose:
A. Input tape
B. Selectors
C. Output tape

Read the selectors into a table.

Were there too many for core? Yes

Display 'too many' to operator.

By-pass the remaining selectors.

Come from pg aa box 006

Does the record qualify? Yes

Format the output.

Go to pg aa box 015

Each input

End of file?

Read an input record.

More

Use subroutine to set qualify switch.

Close all files.

Open the input and output tapes.

Write the output record.

Go to pg aa box 011

Open the input.

The second

Close the input.

Return to supervisor.

Open the input.

The second

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.

Return to supervisor.

Close the input.
DYNACHART
Application Programming Company

GENERAL

DYNACHART accepts COBOL programs as input and produces a flowchart and diagnostic messages as well as optional individual listings of the source program, and cross references within the source program's Data Division only on both the Data and Procedure Divisions.

DYNACHART can be implemented on any computer system capable of supporting a COBOL compiler and configured with at least 30K characters of core, one disc or four tape units, one card reader, and one line printer. DYNACHART is written in a minimum subset of COBOL.

It leases for $4,400 the first year. Maintenance use charge for subsequent years is $700 annually.

Package Output

Source Program Listing - Each line consists of an 80-80 image of the source card preceded by a generated line number.

Flowchart - Detailed flowchart in same logical order as the program source code. Many features under user's control.

Label Cross Reference - Labels are alphabetically listed as are defining source code sequence number and all statements referencing label.

Note: DYNACHART is one of the most expensive documentation packages. Its output features are not so unique as to justify the cost of the package.
EZFLOW
Systonetics Corporation

GENERAL

EZFLOW accepts FORTRAN source programs, reformats and renumbers statements, derives a cross-reference list between statement labels and references of original and new program versions, produces a statement number reference table that shows use and location of numbers on restored flowcharts, and produces a new source deck listing. The package uses a default option control card when none of the option control cards are needed.

The package runs on IBM 360/30 and up (OS), with 110K bytes of core storage; CDC 6000 Series (SCOPE), with 32K bytes of core storage. Peripherals include a card reader and line printer, and optionally a disc or tape. The source language of this package is FORTRAN IV.

EZFLOW is available in two versions off-the-shelf: for the CDC 6000 series and for the IBM 360 or 370 series. The price is $3,500 for a three-year use license as a one-time charge.

PACKAGE OUTPUT

Flowchart - This is a logic flowchart of the restored deck. This is a single column chart of rectangular blocks and diamond-shaped boxes, one source code statement is equal to one block on the chart.
Input and Output Source Deck - The package punches and prints a copy of the output source deck and prints a listing of the input deck.

Cross-Reference List - This listing of statement labels and references shows the old and resequenced source code label numbers and the source list line number.

Statement Reference Table - This table summarizes the program flow by giving the use and location of numbers that are in parentheses in the flowchart.

Note: EZFLOW has automatic conversion to and from the BCD and EBCDIC character sets.
**EZFLOW**

**PROCESSING ROUTINE -- MAIN**

1. **SUBROUTINE SAMPLE (A, B, C, D)**
2. **This is an example to demonstrate some of the features**
3. **DIMENSION A (10), Z(91), X, Y, Z (1000), WXYZ (3)**
4. **DIMENSION QISOI, LABEL (10), ENDI (20)**
5. **DOUBLE PRECISION B(20)**
6. **REAL A, B**
7. **COMPLEX C, D**
8. **REAL 80, DATALABEL**
9. **THIS IS A SAMPLE OF A DDATA STATEMENT**
10. **DIMENSION I, A(1)**
11. **EQV I, A(1)**
12. **RETURN**

**NEW - OLD STATEMENT NUMBER**

<table>
<thead>
<tr>
<th>NEW</th>
<th>OLD</th>
<th>STATEMENT</th>
<th>NUMBER</th>
<th>CROSS</th>
<th>REFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>95</td>
<td>15</td>
<td>2</td>
<td>70</td>
<td>31</td>
</tr>
<tr>
<td>20</td>
<td>51</td>
<td>20</td>
<td>4</td>
<td>60</td>
<td>29</td>
</tr>
<tr>
<td>30</td>
<td>152</td>
<td>25</td>
<td>1</td>
<td>130</td>
<td>43</td>
</tr>
<tr>
<td>40</td>
<td>84</td>
<td>25</td>
<td>6</td>
<td>120</td>
<td>42</td>
</tr>
<tr>
<td>50</td>
<td>20</td>
<td>27</td>
<td>20</td>
<td>50</td>
<td>27</td>
</tr>
<tr>
<td>60</td>
<td>4</td>
<td>29</td>
<td>50</td>
<td>150</td>
<td>45</td>
</tr>
<tr>
<td>70</td>
<td>2</td>
<td>31</td>
<td>52</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>80</td>
<td>63</td>
<td>34</td>
<td>63</td>
<td>80</td>
<td>34</td>
</tr>
<tr>
<td>90</td>
<td>77</td>
<td>36</td>
<td>77</td>
<td>90</td>
<td>36</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>34</td>
<td>78</td>
<td>180</td>
<td>46</td>
</tr>
<tr>
<td>110</td>
<td>400</td>
<td>44</td>
<td>84</td>
<td>40</td>
<td>25</td>
</tr>
<tr>
<td>120</td>
<td>8</td>
<td>42</td>
<td>95</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>130</td>
<td>7</td>
<td>43</td>
<td>152</td>
<td>30</td>
<td>23</td>
</tr>
<tr>
<td>140</td>
<td>1000</td>
<td>44</td>
<td>1000</td>
<td>140</td>
<td>44</td>
</tr>
<tr>
<td>150</td>
<td>50</td>
<td>45</td>
<td>1500</td>
<td>140</td>
<td>44</td>
</tr>
<tr>
<td>160</td>
<td>5000</td>
<td>28</td>
<td>4000</td>
<td>100</td>
<td>39</td>
</tr>
<tr>
<td>170</td>
<td>8041</td>
<td>30</td>
<td>5000</td>
<td>160</td>
<td>28</td>
</tr>
<tr>
<td>180</td>
<td>78</td>
<td>46</td>
<td>8041</td>
<td>170</td>
<td>30</td>
</tr>
</tbody>
</table>
**- - - - E Z F L O W * - - - - ** PROCESSING ROUTINE **- - - - SAMPLE**

SUBROUTINE SAMPLE (A,B,CC,C*,*,D) SAMP 10

C THIS IS AN EXAMPLE TO DEMONSTRATE SOME OF THE FEATURES SAMP 20
C OF E Z F L O W SAMP 30
C
C NOTE THAT STATEMENTS OF THE SAME TYPE ARE ALL GROUPED TOGETHER SAMP 50
DIMENSION A(10),Z(5),XXYZ(1000),WXYZ(3) SAMP 60
DIMENSION O(50),LABEL(10),ENDDD(20) SAMP 70
DOUBLE PRECISION R(20) SAMP 80
COMPLEX*16 C120) SAMP 90
REAL*8 O SAMP 100

DATA LABEL/* THIS IS A SAMPLE OF A DATA STATEMENT */ SAMP 120
EQUIVALENCE (A,81.1C,DI SAMP 130
C THIS IS AN EXAMPLE OF A MULTIPLE RETURN ROUTINE SAMP 140
10 CALL SUB2 (A,BC70.C50,SINOEXI SAMP 150
CALL SUB1 (A,B,6HLABEL / LABEL 2 /67O) SAMP 160
C EXAMPLE OF A COMPUTED GO TO STATEMENT SAMP 170
CGO TO 120,80,70,60,10,40,10,40), INDEX SAMP 180
C EXAMPLE OF DD LOOP PROCESSING SAMP 190
20 DO 30 I=MNOFDO SAMP 200
AI1=-1(1ZI SAMP 210
WXYZIII=I SAMP 220
30 ENDDOIII=(III)*A(I) SAMP 230
C EXAMPLE OF ARITHMETIC IF STATEMENT SAMP 240
40 IF (ENDDD(IO)-123456*SIN(FLOAT(INDEXI)) 140,110,100 SAMP 250
C EXAMPLE OF READ STATEMENT SAMP 260
50 READ (5,160,ENDT70,ERRI140) A SAMP 270
60 WRITE (*170) A SAMP 280
70 RETURN 1 SAMP 290
C/ THIS COMMENT CARD WILL START A NEW PAGE (SLASH IN COLUMN 2) SAMP 300
C EXAMPLE OF A DD LOOP TERMINATING IN A LOGICAL IF STATEMENT SAMP 310
60 DO 90 K=1,SO SAMP 320
AIK=K*K*A(K) SAMP 330
90 IF (MOD(K,10).EQ.0) READ (5,160,END=70,ERR=140) C,D SAMP 340
GO TO 120 SAMP 350
C EXAMPLE OF ANOTHER TYPE OF MULTIPLE RETURN STATEMENT SAMP 360
100 CALL SUB3 (A,B),RETURNS(70,50) SAMP 370
GO TO 150 SAMP 380
110 CONTINUE SAMP 390
120 DO 130 J=M,N SAMP 400
130 A(J)=J SAMP 410
140 RETURN 2 SAMP 420
150 WRITE (6,180) A SAMP 430
RETURN SAMP 440
SAMP 450
C
160 FORMAT (10E8.2) SAMP 460
170 FORMAT (*1 0 INPUT DATA A */(10E8.2)) SAMP 470
180 FORMAT (*A*+E18.5) SAMP 480
END SAMP 490
PROCESSING ROUTINE - Sample

SUBROUTINE SAMPLE (A, B, C, *, D)

This is an example to demonstrate some of the features of EZ FLOW.

Note that statements of the same type are all grouped together.

DIMENSION A(10), B(15), C(20), D(30)

DOUBLE PRECISION B(20)

COMPLEX*16 C(20)

REAL*8 D

DATA LABEL/

This is an example of a multiple return routine.

10 CALL SUB2 (A, B, C, D, E)

IF SUBROUTINE RETURNS
RETURN +1
GO TO 70, 50

CALL SUB1 (A, B, C, D, E)

IF SUBROUTINE RETURNS
RETURN -1
GO TO 70

EXAMPLE OF A COMPUTED GO TO STATEMENT

INDEX = 1, 2, 3, 4, 5, 6, 7, 8
GO TO 20, 30, 40, 50, 60, 70, 80, 90

EXAMPLE OF DO LOOP PROCESSING

20 DO 30 I = 1, N

30 ENDDO

EXAMPLE OF ARITHMETIC IF STATEMENT

40 IF (ENDDO(I) - 123456*$INIFLOAT(INDEX))
EXAMPLE OF READ STATEMENT

READ (5,160,END=70,ERR=100) C,D

EXAMPLE OF A DO LOOP TERMINATING IN A LOGICAL IF STATEMENT

READ (5,160,END=70,ERR=100) C,D

EXAMPLE OF A 00 LOOP TERMINATING IN A LOGICAL IF STATEMENT

READ (5,160,END=70,ERR=100) C,D
EXAMPLE OF ANOTHER TYPE OF MULTIPLE RETURN STATEMENT

100 --- >1 CALL SUB3 (A,R),RETURNS(70,50) [111]

110 --- >1 CONTINUE

120 --- >1 DO 00 J (13)

130 --- >1 RETURN = J= J=N

140 --- >1 RETURN 2

150 --- >1 WRITE (6+100) A

160 --- >1 FORMAT (10E8.2)

170 --- >1 FORMAT (*A=1E18.5)

180 --- >1 END

---

* | E Z F L O W | ** | PROCESSING ROUTINE | ** | SAMPLE |
---|---|---|---|---|---|
10 | - | 3 GO | - | - | - | - | - | - |
20 | - | 3 GO | - | - | - | - | - | - |
30 | - | 4 GO | - | - | - | - | - | - |
40 | - | 3 GO | - | 3 GO | - | - | - | - |
50 | - | 1 CALL | - | 11 CALL | - | - | - | - |
60 | - | 3 GO | - | - | - | - | - | - |
70 | - | 1 CALL | - | 2 CALL | - | 3 GO | - | 6 I/O |
80 | - | 3 GO | - | - | - | - | - | - |
90 | - | 8 GO | - | - | - | - | - | - |
100 | - | 5 IF | - | 9 IF I/O | - | - | - | - |
110 | - | 5 IF | - | - | - | - | - | - |
120 | - | 10 GO | - | - | - | - | - | - |
130 | - | 13 GO | - | - | - | - | - | - |
140 | - | 5 IF | - | 6 I/O | - | - | - | - |
150 | - | 12 GO | - | - | - | - | - | - |
160 | - | 6 I/O | - | 9 IF I/O | - | - | - | - |
170 | - | 7 I/O | - | - | - | - | - | - |
180 | - | 14 I/O | - | - | - | - | - | - |
FACTS
Bonner and Moore Associates, Incorporated

GENERAL

FACTS accepts FORTRAN source programs as input, analyzes programs and subroutines, and produces cross-referenced information according to the option selection by the programmer.

The package runs on the Sigma 7 computer.

PACKAGE OUTPUT

Source Listing - Source statements of the program are automatically listed.

Program Reports - Six program reports are generated:
(1) Common Report
(2) Local Report
(3) Format Statement
(4) Statement Label Report
(5) Recap
(6) Global Report
<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>DECLARATIVE</th>
<th>DEFINITIONAL</th>
<th>REFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NV1</td>
<td>I</td>
<td>00C5</td>
<td>0017 0021</td>
<td>0037</td>
</tr>
<tr>
<td>NV2</td>
<td>I</td>
<td>00C5</td>
<td>0019</td>
<td>0017 0019 0023 0037</td>
</tr>
</tbody>
</table>

All variables *Com-FACTS generated* FACTS generated statement *Four digit FACTS generated statement.

named in blank* plex statement number* numbers in which variable *ment numbers in which the blank common are *Dou-*bers in which *name is defined (as in *common name is referenced (as printed* ble* the blank com-*left hand side of arith-*right hand side of arithmetic alphabetical-* pre-*mon variable *metic statement). *statement).

cally.* ci-*name appears in*
*sion *a declarative *

Each variables*Inte-*manner; (dimen-*
type is indi-*ger *sion, equiva-*
cated by one*Logi-*lence state-*
of the follow-* cal *ments, etc.) by *
ing one-*Real *four digit *
character *sequence num-*
codes. *bers. *

Figure 6-1. Common Report
### VARIABLE APPEARANCES

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>DECLARATIVE</th>
<th>DEFINITIONAL</th>
<th>REFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIT</td>
<td></td>
<td>0027 0029 0032 0034 0040</td>
<td>0027 0029 0032 0034 0040</td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td></td>
<td>0039</td>
<td>0001 0015 0048</td>
<td>004</td>
</tr>
</tbody>
</table>

All variables *Com-plex* statement numbers in which variable name is defined (as in left hand side of arithmetic statement).

Each variable type is indicated by one *Logic* of the following: *Real* four digit character *sequence numbers* *bers.*

Figure 6-2. Local Report
**Format Statement Report**

<table>
<thead>
<tr>
<th>FORMAT LABEL</th>
<th>APPEARANCES</th>
<th>DEFINITION</th>
<th>REFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>901</td>
<td></td>
<td>FACTS generated statement number</td>
<td>0037</td>
</tr>
<tr>
<td>902</td>
<td></td>
<td>in which format statement label is defined.</td>
<td>0040</td>
</tr>
<tr>
<td>903</td>
<td></td>
<td>in the sub-routine or program.</td>
<td>0043</td>
</tr>
</tbody>
</table>

Figure 6-3. Format Statement Report
**HTAPE (04)** STATEMENT LABEL REPORT

<table>
<thead>
<tr>
<th>STATEMENT LABEL</th>
<th>APPEARANCES</th>
<th>DEFINITION</th>
<th>REFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 100</td>
<td>0014 0026</td>
<td>FACTS generated statement number in which the format statement label is defined.</td>
<td>FACTS generated statement number of the instructions which reference the statement listed in the first column.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6-4. Statement Label Report
<table>
<thead>
<tr>
<th>Routine Entries</th>
<th>Internal Functions</th>
<th>External Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>AT</td>
<td>Name</td>
</tr>
<tr>
<td>XXXX (03) 0011</td>
<td>F (03) 0004 0008</td>
<td>A (01) 0005</td>
</tr>
<tr>
<td>YYYY (03) 0006</td>
<td></td>
<td>IABS (01) 0013</td>
</tr>
<tr>
<td></td>
<td></td>
<td>V (01) 0007</td>
</tr>
</tbody>
</table>

Name and location of all subroutines or function entry points, or if a program, the name specified in the PC2 name field, with the location blend.

*Self explanatory*

*All references in this program or function to external subroutines or functions.*

**Figure 6-5. Recap Report**
## Global Report

### Program Entries

<table>
<thead>
<tr>
<th>PRIMARY</th>
<th>SECONDARY</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>MAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTAPE (C4)</td>
</tr>
<tr>
<td>XXX (03)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RTAPE (C4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXX (03)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IABS (01)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTEPE (04)</td>
</tr>
<tr>
<td>XX (02)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IABS (01)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR (01)</td>
</tr>
</tbody>
</table>

List of every program entry.

- All external references for the point defined in the entire run.

Entry points:
- Defined via the PC2 name
- Defined via field, subroutine, routine or function statement

- All programs or subroutines of functions listed under primary.

- Programs or subroutines listed in run which call subroutines.

*Donner & Moore Associates, Inc.*

---

**Figure 6-6. Global Report**
FLOWGEN/F
CalComp

GENERAL

FLOWGEN/F accepts FORTRAN source cards and produces ink-on-page flowcharts. It generates plot commands to drive a CalComp Plotter. It gives no other listings about the program than the flowchart.

PACKAGE OUTPUT

Flowchart - This is a detailed flowchart that is given on a ink-on-paper CalComp Plotter. Comments are enclosed in boxes.
* * * * * * *

** DIMENSION DATA (1024)
** EXECUTABLE STATEMENTS ARE SEPARATED FROM COMMENTS
** UNNUMBERED NON-BRANCHING STATEMENTS ARE STACKED UP TO
** 8 IN A BOX AS FOLLOWS

AN = 0.1
RF = AN * 0.01745
XC = COS (RF)
Y = AY + XC
AN = AN + AD
X = X + 0.0625
XB = 1.0
AY = 3.0
ANZ = 0.0

LOGICAL -IF- STATEMENTS ARE AS FOLLOWS
1 IF (A.NE.B) RETURN
   GO TO 5
   THE PREVIOUS STATEMENT REPRESENTS A -GO TO- STATEMENT
   CALLS TO SUBROUTINES ARE AS FOLLOWS - CONTINUATION CARD SHOWN ALSO
   CALL SUB (P1,P2,P3
1 P4,P5,P6)
   A -DO LOOP- IS SHOWN IN THE FOLLOWING MANNER
5 DO 6 I=1,3
   X = X+1.0
   IF (X-2.0)6,6,7
   CONTINUE
   AN ARITHMETIC -IF- WAS SHOWN IN THE ABOVE -DO LOOP-
   COMPUTED AND ASSIGNED -GO TO - ARE REFLECTED AS FOLLOWS
7 GO TO (10,5,1),K
   NUMBERED STATEMENTS ARE PLACED IN SEPARATE BOXES
10 K = 1
   UNNUMBERED STATEMENTS ..RE STACKED IN THE SAME BOX
   K = 1
   A = 1.0
   READ 12, J
12 FORMAT (1A1)
   THIS IS AN OFF-PAGE DO-LOOP EXAMPLE
   DO 13 I =0, 1
   X = X+1.0
13 CONTINUE
   RETURN AND STOP AND END ARE AS FOLLOWS
   RETURN
   STOP
   AFTER THE FOLLOWING END STATEMENT IS DRAWN AN -UNFULFILLED
   REFERENCE- STATEMENT IS LISTED ON THE PRINTER. ANY
   STATEMENT NUMBERS OR THE WORD NONE WILL THEN BE LISTED.

** END
FORFLOW
DNA System, Inc.

GENERAL

FORFLOW accepts FORTRAN source decks as input. It is a two-program system which consists of two distinct programs: FLOWA produces a flowchart of the source program and SEQ generates a source listing containing re-sequenced FORTRAN statement numbers in ascending order by fives, and changes all branch and formatted input/output statement numbers to agree with the new statement numbers.

The package can operate on an IBM 1130 computer under control of the monitor system or on an IBM 1800 under TSX version 3 or MTX. Minimum machine requirements for the 1130 include 8K bytes of core, 1 card reader, 1 line printer, and 1 disc. For 1800 a minimum of 10K bytes of variable core is required. The package itself is written in FORTRAN and includes some assembler subroutines.

The package costs $480.00 for the card system and $600.00 for disc system.

PACKAGE OUTPUT

Flowchart - at some level of detail as the source program. All flowchart blocks are rectangular.

Resequenced listing - this can be a printed listing of the resequenced source deck, or it can be of punched cards.
FORTRAN VARIABLE NAME DOCUMENTER

Data for Management Decision

GENERAL

The FORTRAN Variable Name Documenter accepts FORTRAN source code as input. The package generates a sequenced listing of a FORTRAN program, a numerical list of the statement numbers used in that program, and an alphabetical list of variable names in the program.

The package is designed for any user's configuration that is capable of supporting a FORTRAN compiler. Auxiliary storage is provided by one disc. The package consists of two programs and a sort. Source language is FORTRAN.

Purchase price is $350 for the object deck or $500 for the source deck.

PACKAGE OUTPUT

Source Listing - a listing of the FORTRAN source code.

Statement Number Listing - a numeric listing of the program statement numbers.

Variable Name Listing - an alphabetic listing of program variable names.
QUICK-DRAW

National Computer Analysts, Incorporated

GENERAL

QUICK-DRAW translates source programs written in Assembler, AUTOCODER, COBOL, PL/I, and FORTRAN into flowcharts and related cross-references. It also provides a diagnostic check list and modified instruction list to aid with debugging.

The system runs on IBM 360; Burroughs 25/35/55; HIS 200, 400, 600, 800; ICL 1900; RCA Spectra 70 or 3301; and Univac 9400/1100. System supervision may be OS, TOS, DOS, MCP, DAPS, MODZ, TDOS, or EXEC 8. Minimum core storage is 32K bytes for DOS and TOS and 65K bytes for OS.

Leases are $1,900 to $6,300 depending on version (3-year lease) and $1,400 to $2,100 a year depending on version.

PACKAGE OUTPUT

Listings for Assembly:

Flowchart - This is a detailed flowchart of the program. It contains all the ASSEMBLY statements in logical groups placed in boxes whose shape is determined by the type of statement.

Cross-Reference by Term - This listing contains entries for all terms used within the program.
Cross-Reference to Equator - This listing repeats the definition and references to equated tags in same format as in the cross-reference by term.

Source Program Listing - Shows the contents of each card read.

Supplementary Listing - This list consists of the following:

1. Diagnostic Check List
2. Unrecognized Op-Codes
3. Assembler Directing Op-Codes
4. Input-Output Type of-Codes
5. Perform Type Instruction
6. Path Terminations
7. Privileged Instructions
8. Program Linkage Op-Code - Entered here are ENTRY, EXTRN, etc.
9. User Macros and Macro Calls
10. Program Segmentation Op-Codes

Listings for FORTRAN:

Source Program Listing - Shows contents of each card.

Flowchart - This is a two-dimensional flowchart. Flowcharts may be in a double-page format.

Statement Label Cross-Reference - This listing contains all statement numbers belonging to procedural statements or referenced by procedural statements.
Data Name Cross Reference - This contains all data field names, subroutine names, and labels used in the program and lists, by card number, references made to those names.

Diagnostic Check List - This provides a number of diagnostic printouts for those statements with error.

Listing for COBOL:

Source Statement Listing - Shown five digit card number used for card identification, plus the card contents.

Flowchart - A detailed flowchart of the program is given.

Cross-Reference by Term - This listing contains entries for data-names, procedure-names, literals, figurative constants. Each entry contains the term and all references to that term.

Diagnostic Flow Summary - This is a graphic summary of the structure of the program in flowchart form. It contains all paragraph and section names together with the range of card numbers comprising each paragraph, their locations on the flowchart and notes and diagnostic pertaining to each paragraph.

Note: All versions of the flowcharts are at the same level of detail as the source code. The COBOL flowchart only, gives the option of compressing the level of detail to achieve a higher level flowchart.
QUICK-DRAW is designed to select a convenient branch point whenever possible to end a page.

QUICK-DRAW is second only to AUTOFLOW in its number of installations.
ROSS REFERENCE TO EQUATES

This listing repeats the definition and references to equated tags in the same format as in the CROSS REFERENCE BY TERM.

6/72 CROSS REFERENCE TO EQUATES II

<table>
<thead>
<tr>
<th>TERAL OR TAG AND INCREMENT</th>
<th>LINE REFERENCING LINE OPERAND NUMBER AND OP-CODE</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Tag</th>
<th>Line</th>
<th>Operand Number and Op-Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKSTAF</td>
<td></td>
<td>2655</td>
</tr>
<tr>
<td>AMVTD</td>
<td></td>
<td>2659</td>
</tr>
<tr>
<td>LSHT</td>
<td></td>
<td>2650</td>
</tr>
<tr>
<td>EQU 14</td>
<td>0043</td>
<td>0116 1 BR 0114 1 LM 0064 1 STM</td>
</tr>
<tr>
<td>EQU 15</td>
<td>0044</td>
<td>0065 2 LR</td>
</tr>
<tr>
<td>EQU 2</td>
<td>0031</td>
<td>0102 1 BAL 0109 1 BAL 0112 1 BAL 0144 1 BR</td>
</tr>
<tr>
<td>EQU 3</td>
<td>0032</td>
<td>UNREFERENCED</td>
</tr>
<tr>
<td>EQU 4</td>
<td>0033</td>
<td>0108 1 BASE 0111 1 LA 0094 1 LM 0107 2 BASE 016</td>
</tr>
<tr>
<td>EQU 5</td>
<td>0034</td>
<td>0096 1 BASE 0146 1 BASE 0094 2 LM</td>
</tr>
<tr>
<td>EQU 6</td>
<td>0035</td>
<td>UNREFERENCED</td>
</tr>
<tr>
<td>EQU 7</td>
<td>0036</td>
<td>UNREFERENCED</td>
</tr>
<tr>
<td>EQU 8</td>
<td>0037</td>
<td>0065 1 LR 0063 2 USING</td>
</tr>
<tr>
<td>EQU 9</td>
<td>0038</td>
<td>UNREFERENCED</td>
</tr>
</tbody>
</table>

SPECIAL CROSS REFERENCES

This listing repeats entries from the CROSS REFERENCE BY TERM if they are one of the following:

1. Non-Branch References to Instruction Tags. This will call attention to modified instructions, such as switches and also serve as a check for the "UNENTERED STATEMENT" diagnostic, where in fact, the statement was entered by means of a load address and a branch register.

2. Branches to Data.

3. Anything unusual that we feel should be highlighted, such as ENTRY and EXTRN references.

6/72 SPECIAL CROSS REFERENCES FOR QUICK-DRAW SAMPLE PROGRAM A-12 FOR ASSEMBLY

<table>
<thead>
<tr>
<th>TERAL OR TAG AND INCREMENT</th>
<th>LINE REFERENCING LINE OPERAND NUMBER AND OP-CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSTIME</td>
<td>0072</td>
</tr>
<tr>
<td></td>
<td>0073 1 MVI</td>
</tr>
<tr>
<td>LOW</td>
<td>0146</td>
</tr>
<tr>
<td></td>
<td>0138 3 PRTDV</td>
</tr>
<tr>
<td>INTER</td>
<td>0153</td>
</tr>
<tr>
<td></td>
<td>0136 1 CTRL 0138 1 PRTDV 0142 1 PUT 0074 1 MVI</td>
</tr>
<tr>
<td>EQU 14</td>
<td>0043</td>
</tr>
<tr>
<td></td>
<td>0116 1 BR</td>
</tr>
<tr>
<td>EQU 2</td>
<td>0031</td>
</tr>
<tr>
<td></td>
<td>0144 1 BR</td>
</tr>
<tr>
<td>LOWSw</td>
<td>0098</td>
</tr>
<tr>
<td></td>
<td>0087 1 MVI 0090 1 MVI</td>
</tr>
</tbody>
</table>
PROGRAM SEGMENTATION OP-CODES

CSECT and START are in this section.

SAMPLE SUPPLEMENTARY LISTING

SUPPLEMENTARY LISTINGS FOR QUICK-DRAW SAMPLE PROGRAM A-12 FOR ASSEMBLY

DIAGNOSTIC CHECK LIST

*00146 UNENTERED STATEMENT
*00162 UNENTERED STATEMENT

PROGRAM LINKAGE OP-CODES

*00061 ENTRY B1B095

FOR ACCESS BY COBOL.

ASSEMBLER DIRECING OP-CODES

*00003 SPACE 10
*00023 EJECT 2
*00043 SPACE 2
*00053 EJECT 12
*00060 SPACE 2
*00063 USING *ERB

ESTABLISH ADDRESSABILITY

*00149 SPACE 1
*00152 SPACE 2
*00161 EJECT 4
*00162 SPACE 2
*00169 LTORG 3

INPUT-OUTPUT TYPE OP-CODES

*00136 CNTRL PRINTER,SP,1

ADVANCE ONE LINE IMMEDIATELY,

*00138 PRTDV PRINTER,12,OVERFLOW

TEST FOR OVERFLOW

*00142 PUT PRINTER,(R4)

ADDRESS OF COBOL-LINE IN GPR

*00153 DTFPR DEVADDR,SYSLST

PRINTER ON SYSTEM LOGICAL

UNIT

IDAREA1=ALINE1, IDAREA2=ALINE2,

WORKA=YES,

SPECIFIED IN GPR 4

BLKSIZE=13, PRTDV=YES,

TEST FOR CHANNEL=12,

CONTROL=YES

CNTL MACRO USED FOR

FORMS-CONTROLS

*00162 PRNODE IDAREA2,YES, WORKA, YES

PRTDV=YES, CONTROL=YES

PERFORM TYPE INSTRUCTIONS

*00102 BAL R2,PRINT

PRINT AS SET UP BY COBOL-

ADVANCE 1-

*00109 BAL R2,PRINT

AND PRINT CAPPED COBOL LINE.

*00112 BAL R2,PRINT

PRINT UNDERLINE.
Reference Format Listing (Source Program Listing)

The entire source program is listed as QUICK-DRAW reads it for processing.

The complete contents of each card is printed, including the card sequence number found in columns 73-80. In addition, a four-digit statement sequence number, assigned by QUICK-DRAW, is printed to the left of the card image. This sequence number will appear on the flowchart above the top left corner of the corresponding symbol.

The heading of each page is labeled "Reference Format Listing." The QUICK-DRAW Release Number is printed following the last source statement.

```plaintext
*0044*  IBIG = 0
*0045*  MOST = 0
*0046*  K = 0
*0047*  INDEX8 = 1
*0048*  J = (JHIGH - 1)/8 + 1
*0049*  DO 54 I = 1, J
*0050*  DO 53 L = 1, 8
*0051*  NTII (L) = 0
*0052*  54 WRITE (8'INDEX8,20) (NTII(M),NTJI(M),M = 1,8)
*0053*  WRITE (3,501) JHIGH
*0054*  501 FORMAT (1HO,14,19H NODES SET TO ZERO.)
*0055*  INDEX7 = 1
*0056*  J = 0
*0057*  6 READ (7'INDEX7,5) NODI, NODJ, ITIME, COST
*0058*  5 FORMAT (315,F6.0)
*0059*  IF (NODI - 9999) 7, 11, 7
*0060*  7 INDEX8 = (NODI - 1)/8 + 1
*0061*  READ(8'INDEX8,20) (NTII(M),NTJI(M),M = 1,8)
*0062*  INDEX8 = INDEX8 - 1
*0063*  M = 8 - (INDEX8*8 - NODI)
*0064*  NTII(M) = 1
*0065*  WRITE(8'INDEX8,20) (NTII(M),NTJI(M),M = 1,8)
*0066*  INDEX8 = (NODJ - 1)/8 + 1
*0067*  READ(8'INDEX8,20) (NTIJ(N),NTJJ(N),N = 1,8)
*0068*  INDEX8 = INDEX8 - 1
*0069*  N = 8 - (INDEX8*8 - NODJ)
*0070*  NTJJ(N) = 1
*0071*  WRITE(8'INDEX8,20) (NTIJ(N),NTJJ(N),N = 1,8)
*0072*  TCOST = TCOST + COST
*0073*  MOST = MOST + ITIME
*0074*  K = K + 1
*0075*  IF (NODI - IBIG) 9, 9, 8
*0076*  8 IBIG = NODI
*0077*  9 IF (NODI - IBIG) 6, 10, 9
*0078*  10 IBIG = NODI
*0079*  GO TO 6
*0080*  11 KLAST = K
*0081*  WRITE (3,502)
*0082*  502 FORMAT (1HO,36H FLAGS ALL SET FOR NODE NUMBER CHECK.)
*0083*  IRST = 0
*0084*  LAST = 0
*0085*  K = IBIG
*0086*  INDEX8 = (K - 1)/8 + 1
*0087*  READ(8'INDEX8,20) (NTII(M),NTJI(M),M = 1,8)
*0088*  M = 8 - ((INDEX8 - 1)*8 - K)
*0089*  IF (NTII(M)) 12, 13, 12
*0090*  12 IF (NTJI(M)) 14, 15, 14
```
Figure 1 - Sample Portion of Flowchart
Statement Label Cross Reference Listing

The first column of this listing contains all statement numbers belonging to procedural statements or referenced by procedural statements. These statement numbers are listed in numeric order. In addition, Column 1 contains the names of any subroutines referenced by the program.

The second column lists the page and symbol box numbers where each statement number or subroutine appears on the flowchart. These location references are in the form "pp.bb", where pp is the flowchart page number and bb is the symbol box number on that page. If a statement number is undefined, Column 2 contains the entry "UNDEF". If a referenced subroutine is not included in the source program, Column 2 contains the entry "EXTRN". Format statement labels will have the entry "FORMAT" in this column.

The third column lists all references to the statement number or subroutine name. These references are also in the form "pp.bb". If a statement number is not referenced, it is so indicated.

Each subprogram included in the source deck has its own cross-reference listing, beginning on a separate page. If any statement number appears more than once in the main program or in the same subprogram, it is flagged as MULTIPLY-DEFINED.

<table>
<thead>
<tr>
<th>Statement Number</th>
<th>Page</th>
<th>Symbol Box</th>
</tr>
</thead>
<tbody>
<tr>
<td>52</td>
<td>01.01</td>
<td>04.02</td>
</tr>
<tr>
<td>53</td>
<td>01.17</td>
<td>01.15</td>
</tr>
<tr>
<td>54</td>
<td>01.20</td>
<td>01.14</td>
</tr>
<tr>
<td>55</td>
<td>01.44</td>
<td>02.09</td>
</tr>
<tr>
<td>60</td>
<td>04.03</td>
<td>03.32</td>
</tr>
<tr>
<td>63</td>
<td>04.33</td>
<td>04.30</td>
</tr>
<tr>
<td>64</td>
<td>05.02</td>
<td>05.01</td>
</tr>
<tr>
<td>65</td>
<td>05.27</td>
<td>05.26</td>
</tr>
<tr>
<td>66</td>
<td>06.05</td>
<td>04.45</td>
</tr>
<tr>
<td>67</td>
<td>06.16</td>
<td>NOT REFERENCED</td>
</tr>
<tr>
<td>71</td>
<td>03.18</td>
<td>03.17</td>
</tr>
<tr>
<td>80</td>
<td>06.12</td>
<td>05.01</td>
</tr>
</tbody>
</table>
Data Name Cross Reference Listing

This table contains all data field names, subroutine names, and labels used in the program and lists the card numbers in which references are made to those names or labels.

This listing is not available from the FORTRAN compiler and offers a distinct advantage to programmers for debugging, maintenance and documentation purposes. Very often a quick glance through this listing is sufficient to detect keypunching errors or incorrect references.

<table>
<thead>
<tr>
<th>NCA FORTRAN QUICK DRAW SAMPLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROSS REFERENCE BY CARD NUMBER</td>
</tr>
<tr>
<td>* 0076.</td>
</tr>
<tr>
<td>* 0017</td>
</tr>
<tr>
<td>* 0010</td>
</tr>
<tr>
<td>* 0137 0150 0159 0168 0174 0200</td>
</tr>
<tr>
<td>* 0108 0117 0119 0121 0130 0132 0198 0205 0213</td>
</tr>
<tr>
<td>* 0231 0233 0241 0242</td>
</tr>
<tr>
<td>* 0076 0077</td>
</tr>
<tr>
<td>* 0133 0169 0201 0205 0226 0231</td>
</tr>
<tr>
<td>* 0135 0170 0203 0205 0220 0228 0231</td>
</tr>
<tr>
<td>* 0133 0172 0204 0205 0222 0223 0225 0231</td>
</tr>
<tr>
<td>* 0134 0170 0202 0205 0220 0221 0227 0231</td>
</tr>
<tr>
<td>* 0077</td>
</tr>
<tr>
<td>* 0058 0085 0127 0132</td>
</tr>
<tr>
<td>* 0056 0087 0121 0129 0132 0152</td>
</tr>
<tr>
<td>* 0057 0083 0121 0122 0128 0132 0151</td>
</tr>
<tr>
<td>* 0092</td>
</tr>
<tr>
<td>* 0003 0019 0046 0049 0052 0084 0085 0101 0161</td>
</tr>
<tr>
<td>* 0194 0201</td>
</tr>
<tr>
<td>* 0059 0090 0123 0124 0126 0132 0153</td>
</tr>
<tr>
<td>X</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
<tr>
<td>LITERAL</td>
</tr>
</tbody>
</table>
SAMPLE DIAGNOSTIC FLOW SUMMARY

0361 0369 01.29
- (ACCEPT-DATE=300) 
- 
- > E 0368

0370 0387 01.33
- (TEST-GO-NUM)
- (ACCEPT-NUM-TABLES=300)
- ALTER 0409 TO 0420
- 
- > E 0361

0378 RESERVED WORD MARGIN A

0388 0393 01.47
- (INVALID-DATE=305)
- 
- > E UNDF E 0453 E 0361

0390 NO LITERAL CLOSING QUOTE

0392 UNDEFINED REFERENCE

0394 BAD-NUM-TABLES=306

0395 NON-ANSI CHARACTER SEQ.

0396 NESTED CONDITIONAL

0402 0408 02.12
- (GET-NEXT-DISK-REC-310)
- 
- > E 0403 E 0408 E 0399

0403 0408 E 0382
- 
- > E 0403 E 0408 E 0399

0409 0419 02.17
- (MATCHED-BILL=320)
- ALTER 0492 TO GO TO 0436
- 
- > A 0378 - 0420 E 0409

0410 STATEMENT UNALTERABLE

0412 NULLABLE LOOP

0413 NESTED CONDITIONAL

0420 PARAGRAPH ENDS BADLY

0426 EXIT AFTER GO

0426 EXIT AFTER GO

0427 EXIT AFTER GO

0436 0436 02.31
- (TOTAL-OUT)
- NULL PARAGRAPH
- 
- > E 0436 E 0436

0446 0452 02.36
- (IT-IS-GREAT)-323
- 
- > E 0438 E 0446 P 0420 E 0453

0447 0449 02.32
- (CHECK-OUT)
- 
- > E 0437 E 0449

0449 0451 02.36
- (IT-IS-GREAT)
SUPEREF
Mantech Corporation

GENERAL

SUPEREF accepts FORTRAN coded source programs and produces a comprehensive symbolic name cross reference dictionary.

SUPEREF is operational on any CDC 6000 series computer whose software capability has been updated to include the Random access (mass storage) File Routines OPENMS, STINDX, READMS, and WRITMS.

PACKAGE OUTPUT

Variable Name Dictionary - This listing includes the variable name being used, the routine that it is located in, and various other information.

Note: A note of interest is that Mantech Corporation was awarded two contracts from the U. S. Naval Ordinance Laboratory in Silver Spring, Maryland to have two large-scale FORTRAN simulation programs processed by SUPEREF.
PROGRAM INTEST? (INPUT, OUTPUT, SAVEF, TAPES=INPUT, TAPES=OUTPUT, TAPES=OUTPUT, TAPES=SAVEF)
COMMON NOLN4, NOLN5, NOLN6, NOLN7, NOLN8, NOLN9, NPAGE7
COMMON /AHCD/ A(5), R(5,2), C(10), D(10,2), F(10,4)
DIMENSION (E(O,4), E(O,5))
COMMON NV(NPAG5, NPTC)
COMMON /AHCD/ A(5), R(5,2), C(10), D(10,2), F(10,4)
DIMENSION (E(O,4), E(O,5))
COMMON NV(NPAG5, NPTC)

100 FORMAT (3(I4,F10.3))
DO 25 K=1,3
IF (LOAD(K,1).EQ.9999) GO TO 20
IF (LOAD(K,1).EQ.0) GO TO 20
IF (LOAD(K,1).LE.IDMTBL(I+1),AND...
1 (LOAD(K,2),LE.IDMTBL(I+2),AND...
2 (LOAD(K,3),LE.IDMTBL(I+3),)) GO TO 23
WRITE (6,350) NAME, LOAD(K,1), LOAD(K,2), LOAD(K,3), VALUE(K), NAME,
120H FOR ARRAY/VARIABLE /A6+1H(*I4+1M+I4+1H+I4+1H/)
CALL LINE6(1)
ISW=1
GO TO 25
23 I=IDMLOC(IO,MTBL(I+1),MTBL(I+2),MTBL(I+3),
1 LOAD(K,1), LOAD(K,2), LOAD(K,3)) + INCRM
E[I]=VALUE(K)
WRITE (6,750) NAME, LOAD(K,1), LOAD(K,2), LOAD(K,3), VALUE(K)
350 FORMAT (1H7X,A6+X1H(*I4+1M+I4+1H+I4+1H+I4+1H),3X,F10.3)
CALL LINE7(1)
CONTINUE
GO TO 20
30 IF (ISW.EQ.0) GO TO 35
WRITE (6,305)
CALL LIN6(1)
305 FORMAT (1H30NO DIAGNOSTICS FOR THIS RUN OF ITEST2.
35 WRITE (6,500) ITRTL1(1):A
CALL LINE6(2)
WRITE (6,500) ITRTL2(2):R
CALL LIN6(3)
WRITE (6,500) ITRTL3(3):C
CALL LINE6(3)
WRITE (6,500) ITRTL4(4):D
CALL LINE6(4)
WRITE (6,500) ITRTL5(5):F
CALL LINE6(5)
WRITE (6,500) ITRTL6(6):G
CALL LINE6(6)
500 FORMAT (1H 1A6+/S(5(F10,3))
ENDFILE 7
CALL EXIT
END
**ILLUSTRATIVE PROBLEM**

**FORTRAN STATEMENT**

<table>
<thead>
<tr>
<th>ROUTINE NAME</th>
<th>FORTRAN STATEMENT</th>
<th>VARIABLE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DATA A /5=1.0/</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>EQUIVALENCE (E(1)*A(1))</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>WRITE (6,500) IRTBL(1)*A</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DATA B /10=2.0/</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>WRITE (6,500) IRTBL(2)*B</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DATA C /10=3.0/</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>WRITE (6,500) IRTBL(3)*C</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DATA D /40=4.0/</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>WRITE (6,500) IRTBL(4)*D</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DIMENSION E(106)</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>EQUIVALENCE (E(1)*A(1))</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>E(I)=VALUE(K)</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>CALL EXIT</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DATA F /5=6.0/</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>WRITE (6,500) IRTBL(5)*F</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DATA G /40=6.0/</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>WRITE (6,500) IRTBL(6)*G</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>DATA H /40=6.0/</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>COMMON / ABCD</td>
<td></td>
</tr>
<tr>
<td>INITL</td>
<td>WRITE (6,500) IRTBL(7)*H</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>FUNCTION IDMLOC (II<em>JJ</em>K<em>K</em>II<em>K</em>II)</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>IDMLOC=II<em>JJ</em>(II-K-1)</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>10 IDMLOC=II<em>JJ</em>(K-1)</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>20 IDMLOC=II<em>JJ</em>K</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>25 IDMLOC=II<em>JJ</em>K<em>K</em>II*K</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>IDMLOC=II<em>JJ</em>K<em>K</em>II<em>K</em>K</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>LOADR(K+1)+LOADR(K+2)+LOADR(K+3)</td>
<td>INCRM</td>
</tr>
<tr>
<td>IDMLOC</td>
<td>IDMLOC=II<em>JJ</em>K<em>K</em>II<em>K</em>K</td>
<td></td>
</tr>
<tr>
<td>IDMLOC</td>
<td>DIMENSION IRTBL(7,2)*IDMTRI (6,3)+LOADR(3,3)+VALUF(3)</td>
<td></td>
</tr>
</tbody>
</table>

**PAGE 1**
FORDOC

J. Toellner and Associates

GENERAL

FORDOC accepts Fortran source program as input and produces a restructured source deck along with a variable name cross-reference. The system is composed of six modules that can be run separately or as an entire series.

PACKAGE OUTPUT

Restructured source deck - the original source deck is "cleaned up" by FORDOC giving an easier to read source deck.

Cross-Reference - this listing is a variable name cross-reference listing references made to a given variable name.
SUMMARY

The need for adequately documented computer programs is an essential element common to all data processing centers. This need for documentation can be efficiently accomplished through the use of automatic documentation software packages. The use of these packages saves both time and money as well as establishing a standardized documentation presentation for each program written.

This report has shown most of the better proprietary automatic documentation systems along with their output features. RPG documentors have been excluded intentionally.

It is evident that there is much room for improvement in this area of program documentation. The door is open for an automatic documentation system which does a better job of presenting a program structure. Better automatically produced flowcharts and descriptions of data structures are two areas in which work needs to be done to develop new modes of presentation.

There also is a total lack of any aids to produce global or system-wide documentation in the software packages reviewed here.

Perhaps through the combinations of certain features found in automatic documentation packages existing today, and by utilizing new ideas to supplement these features, an ideal system could be developed.
COMPARISON OF OPERATING SYSTEMS, IBM S/360 OS, UNIVAC EXEC 8, CDC SCOPE 3
AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Working Paper No. 5
July 14, 1972

Documentation and Operating Systems

by
Andrew Sobey, Jr.

Texas A&M University
Texas Engineering Experiment Station
ABSTRACT

The various options available to the user through compilers (PL/I, FORTRAN, and COBOL), assemblers and linkage editors/loaders are listed. Conclusions are then drawn as to which of these options would be useful in the development of an automatic documentation system.
TABLE OF CONTENTS

I. ABSTRACT ........................................ C-1
II. INTRODUCTION ................................. C-3
III. COMPILER/ASSEMBLER OPTIONS ............. C-4
    A. Options ...................................... C-4
    B. Results and Remarks ....................... C-4
IV. LINKAGE EDITOR/LOADER OPTIONS ........... C-30
    A. Options ...................................... C-30
    B. Results and Remarks ....................... C-33
V. CONCLUSION ..................................... C-34

TABLES

I  Assembler Options ......................... C-5
II FORTRAN Options ............................. C-12
III COBOL Options ............................... C-17
IV PL/I Options ................................. C-24
INTRODUCTION

The purpose of this paper is to describe the interaction between a user program and the operating system of a computer. The motivation for this study (also the main objective of the study) was 1.) to determine those options available to the user without modification of the operating system which would be of significant importance to the development of an automatic program documentation system and 2.) to determine what difficulties might be encountered at the operating system level in developing an automatic documentation system to be used on many different manufacturers' computers. As the basis for the discussion of this topic, the following three operating systems will be considered:

1. IBM's OS/360
2. UNIVAC's EXEC 8
3. CDC's SCOPE 3

To explain the medium by which this interaction takes place, and also the extent to which it can take place, the compiler and linkage editor/loader options available to the user through the control language for these systems will be discussed. As it would be impossible in the allotted time to examine all existing higher level languages and their compiler options (just as it would also be impossible to study all operating systems in existence), the following three languages have been chosen for study as being representative of all higher level languages: PL/I, FORTRAN, and COBOL. Also, the assembler languages of the above mentioned three machines and their options will be considered.
A. Options

The first step in the study was to compile a list of all the compiler/assembler options available to the user by the operating systems for the four languages under consideration. These options were then studied in light of their possible usefulness as part of an automatic system of program documentation. A list of these options is indicated on the next several pages. It will contain an explanation of the option, and is broken down by manufacturer and language.

It is urged that the reader study this list of options before proceeding to the next section of the paper (See list on following pages).

B. Results and Remarks

Because one of the criteria of the study was to determine those items which could be obtained without modification to the operating system, let us consider those options which are common to FORTRAN, COBOL, and Assembler languages under all operating systems (PL/I is not offered on the UNIVAC 1108 or CDC 6600 machines.)

1. A source listing may be obtained.
2. A cross-reference table may be obtained.
3. The user may specify from where to read the input file.
4. The user may specify where to write the object module.

Thus, it can be seen that although a myriad of options are offered by the three computer manufacturers, only four options are common to all of them. These options, however, the author feels would be of significant value to the development of an automatic documentation system.
<table>
<thead>
<tr>
<th>OPTION</th>
<th>CDC 6600</th>
<th>IBM 360</th>
<th>UNIVAC 1108</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify that an assembler language listing is to be produced</td>
<td>L=fn; place a full listing on the file named fn; if L=0, put a brief listing on the file named OUTPUT; if absent OUTPUT is assumed for fn</td>
<td>LIST (NO)</td>
<td></td>
</tr>
<tr>
<td>Specify from what file the input is to be read.</td>
<td>I = fn; the input is on the file named fn; if absent INPUT is assumed for fn</td>
<td>Done through JCL</td>
<td>The file name to be read is coded on the control statement.</td>
</tr>
<tr>
<td>Specify the file the object module is to be written upon.</td>
<td>B=fn; a binary file is written on the file named fn; if absent, LGO is assumed for fn; if B=0 is coded, the binary file is suppressed.</td>
<td>LOAD, pace on the drive specified by the SYSGO DD statement</td>
<td>The file name the object module is to be placed on is specified on the control statement.</td>
</tr>
<tr>
<td>Specify where to search to find the systems text.</td>
<td>if absent, or S, the systems text is on SYSTEXT(SCOPE Central Processor Macros) S=rname; the systems text is on the overlay named rname. S=SCPTEXT; the systems text if from the library overlay named SCPTX which contains the system symbol definitions S=SMTEXT; systems text for SORT/MERGE macros plus SCPTEX macros</td>
<td>Done through JCL</td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>----------------------</td>
<td>-----------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Specify that a deck is to be produced of the object module</td>
<td>S=PPTEXT; systems text for PP macros (SCOPE Peripheral Processor Macros)</td>
<td>DECK(NO)</td>
<td></td>
</tr>
<tr>
<td>Specify that the object module is to be tested.</td>
<td>TEST(NO); the object module contains the special source symbol table required by the test translator (TESTRAN) routine and the TSO command processor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a cross-reference table is to be produced.</td>
<td>L</td>
<td>XREF (NO)</td>
<td></td>
</tr>
<tr>
<td>Specify that the assembler check for possible coding violations of program reenterability</td>
<td>RENT (NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the number of lines to be printed between headings in the listing.</td>
<td>LINECNT=xx; where xx is the number of lines desired.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that boundary alignment errors are to be printed</td>
<td>ALGN (NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Specify what capability the assembler has.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify an optional file upon which to write diagnostic messages</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify numbers are to be written in the beginning of each line for which diagnostic information is given</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that statement numbers are to be supplied for statements for which diagnostic information is given</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the run is to be continued even though errors have been detected</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the run is to take the error exit if errors are detected</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Indicate the relocatable is output code is quarter-word sensitive.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify compressed card input in columns 1 through 80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that input cards have sequence numbers in columns 73-80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify compressed card input in columns 1-72 and sequence numbers in column 73-80.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify a sequence check is to be performed when reading the input cards (columns 73-80)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the assembler's internal symbol table and procedure sample table areas are to be expanded</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify all listings are to be suppressed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify a machine language listing is to be produced</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Specify the source language output should be in Field data code</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the ASCII character set is to be used</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the amount of core storage is to be lessened for the internal tables</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the relocable output code is third-word sensitive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify an update is to be made of an existing source language input element to the next higher element cycle</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the correction lines are to be listed at the head of the printer listing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the name and version of the source element</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Specify the name and version of an updated source language module, ie, where to place the updated module</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the name and version of the updated object module, ie, where to store it</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that corrections are to be noted on the listing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify alternate Assembler is to be used</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify a set of system symbol definitions are to be provided to the assembler before the assembler source code</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify comments are to be inserted in the source or object module</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify no listing is desired</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the output punched is in multiple word octal format</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify an updated output source language element is to be punched on cards</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the correction deck is to be listed prior to the assembler listing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Suppress formation of information normally given to the diagnostic routine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----------</td>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>Specify that a machine language listing of the compiled program is to be produced</td>
<td>0; in octal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a source listing of the program to be compiled is to be produced</td>
<td>L</td>
<td></td>
<td>SOURCE(NO)</td>
</tr>
<tr>
<td>Cross Reference Table-Showing where the variables were defined and where they were referenced</td>
<td>R</td>
<td></td>
<td>XREF(NO)</td>
</tr>
<tr>
<td>List All diagnostics indicating all non-ASA usage</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify an assembler-language listing is to be produced</td>
<td></td>
<td></td>
<td>LIST(NO)</td>
</tr>
<tr>
<td>Tells on which file to write the output</td>
<td>l=fn; l is the type of listing desired (one of the above four); fn is the file name which is to be written upon</td>
<td>Done through JCL</td>
<td>Done through control statement</td>
</tr>
<tr>
<td>Indicate from what file the source program is to be read</td>
<td>I=fn; fn tells which file is to be read from</td>
<td>Done through JCL</td>
<td>Done through the Control Statement</td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC 6600</td>
<td>IBM 360</td>
<td>UNIVAC 1108</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Indicate where to write the machine language file of the compiled program</td>
<td>B=fn; a binary relocatable file is to be written on the file-name fn</td>
<td>LOAD; write the object module on the data set specified by the SYSLIN DD statement</td>
<td>Done through the control statement</td>
</tr>
<tr>
<td>Provide and error trace back and calling sequence</td>
<td>T</td>
<td>ID; also, internal statement numbers are generated for statements that call a subroutine or contain an external function references (four extra bytes are needed for linkage)</td>
<td>Done by 'SNOOPY'</td>
</tr>
<tr>
<td>permits the programmer to name his main program routine</td>
<td></td>
<td>NAME = xxxxxx; where xxxxxx is a six character name conforming to FORTRAN rules of variable names</td>
<td></td>
</tr>
<tr>
<td>Indicate the maximum number of lines to be written on a page of the source listing</td>
<td></td>
<td>LINECNT=XX; where xx is the number of lines desired</td>
<td></td>
</tr>
<tr>
<td>Indicate an object module is to be punched</td>
<td></td>
<td>DECK(NO)</td>
<td>p</td>
</tr>
<tr>
<td>Include a table of named variables, their type and location, and a table of labels</td>
<td></td>
<td>MAP(NO)</td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM 360</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td>Indicate which character set the compiler should accept</td>
<td></td>
<td></td>
<td>BCD or EBCDIC</td>
</tr>
<tr>
<td>Indicate to what degree the compiler is to optimize the coding produced</td>
<td></td>
<td></td>
<td>OPT=0; no optimization</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OPT=1; each source module is treated as a single program loop each loop is optimized with regard to register allocation and branching</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OPT=2; the compiler treats each source module as a collection of program loops and optimize each loop with regard to register allocation, branching, common expression, elimination, and replacement of redundant computations</td>
</tr>
<tr>
<td>Indicate the amount of main storage available to the compiler</td>
<td></td>
<td></td>
<td>Size=nnnnk; where nnnn is between 115 and 9999</td>
</tr>
<tr>
<td>Specify that a listing indicating the loop structure and the logical continuity of the program is to be produced</td>
<td></td>
<td></td>
<td>EDIT; A SYSPRINT DD statement must be included, and OPT=2 must also be specified</td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Specify whether or not the output from the compiler is to be accepted if an error has been detected</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify inline double precision coding is to be generated</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the labeled common is to be attached to only those segments which use it</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the alternate FORTRAN compiler is to be used</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify an updated source module is to be punched</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the time of each phase and the total compilation time is to be printed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the correction deck is to be listed prior to the compilation listing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify patch cards are to be used to alter the compiler operation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the formation of information normally given to the diagnostic-routine is to be suppressed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>------</td>
</tr>
<tr>
<td>Specify what source element is to be updated</td>
<td>Done through JCL</td>
<td>N1/V1; when N1 is the source element name and V1 is the version number</td>
<td></td>
</tr>
<tr>
<td>Specify an updated source element is to be stored</td>
<td>Done through JCL</td>
<td>N2/V2; where N2 is the source element name and V2 is the version number</td>
<td></td>
</tr>
<tr>
<td>Specify where to store a relocatable object module</td>
<td>Done through JCL</td>
<td>N3/V3; where N3 is the source element name and V3 is the version number</td>
<td></td>
</tr>
</tbody>
</table>
TABLE III. COBOL OPTIONS

<table>
<thead>
<tr>
<th>OPTION</th>
<th>CDC</th>
<th>6600</th>
<th>IBM</th>
<th>360</th>
<th>UNIVAC</th>
<th>1108</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicate that the source module is to be listed</td>
<td>L</td>
<td>-</td>
<td>SOURCE(NO)</td>
<td>-</td>
<td>I; single</td>
<td>listing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>includes items</td>
<td>specified by</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C,D and O</td>
<td>options</td>
</tr>
<tr>
<td>List all diagnostics indicating non-ASA usage</td>
<td>X</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Indicate that items copies from the library are to be</td>
<td>C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>listed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indicate that a cross reference table is to be produced</td>
<td>R</td>
<td>-</td>
<td>XREF; lists</td>
<td>-</td>
<td>R; lists</td>
<td>data names and</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>data names and</td>
<td></td>
<td>file names</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>procedure</td>
<td></td>
<td></td>
<td>names</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>names</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indicate that the object module is to be listed</td>
<td>O</td>
<td>-</td>
<td>PMAP; listed</td>
<td>-</td>
<td>O; listed in</td>
<td>octal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in hexadecimal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>also lists</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>register</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>assignments,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>global tables,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>and literal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>pools</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indicate where to write the file containing the last</td>
<td>l=fn; where l is L or</td>
<td>-</td>
<td>Done through</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>output</td>
<td>any combination of L</td>
<td></td>
<td>JCL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>with X,C,R,O,M (see</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>above) and fn is the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>file name on which</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>the output is to be</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>written</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC</td>
<td>1108</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----</td>
<td>------------</td>
<td>-----</td>
<td>-----------</td>
<td>--------</td>
<td>------------</td>
</tr>
<tr>
<td>Indicate from what file the source program is to be read</td>
<td>I=fn; where fn is the filename to be read</td>
<td>Done through JCL</td>
<td></td>
<td></td>
<td></td>
<td>Done through the control statement</td>
</tr>
<tr>
<td>Indicate on what file to write the object module</td>
<td>B=fn; write the relocatable binary file named fn; B=0 suppresses the writing of this file</td>
<td>LOAD; place the object module on mass storage or a tape volume</td>
<td></td>
<td></td>
<td></td>
<td>Done through Control Statement</td>
</tr>
<tr>
<td>Indicate on what file the COBOL library is located</td>
<td>S=fn; fn is the file on which the COBOL library is located; needed only if fn is not COLIB</td>
<td>Done through JCL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Suppress all DATA DIVISION binary output except from the WORKING STORAGE SECTION and CONSTANT SECTION for a subcompiled program which would duplicate output from a separately combined main program - this enables them to be properly loaded together</td>
<td>SUB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Separate the overlay segments from the main programs so that separately compiled programs can be loaded properly</td>
<td>OB=fn; the overlay segments are written on the file named fn</td>
<td>SIZE=yyyyyyyy; is measured in bytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the amount of main storage available for compilation</td>
<td></td>
<td></td>
<td></td>
<td>SIZE=yyyyyyyy; is measured in bytes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the amount of storage allocated for buffers</td>
<td></td>
<td></td>
<td></td>
<td>BUF=yyyyyy. measured in bytes; if BUF and SIZE are both specified,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC</td>
<td>1108</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Specify that a condensed listing is to be produced</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List a glossary of symbols used in the program</td>
<td></td>
<td></td>
<td>DMAP(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify an object module deck is to be produced</td>
<td></td>
<td></td>
<td>DECK(NO)</td>
<td></td>
<td></td>
<td>S; a deck of an updated file is produced</td>
</tr>
<tr>
<td>Provide a sequence check of the source module statements</td>
<td></td>
<td></td>
<td>SEQ(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indicate the number of lines to be printed on each page of the source listing</td>
<td></td>
<td></td>
<td>LINECNT=xx; where xx is the number of lines desired</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indicate the severity of error messages to be printed</td>
<td></td>
<td></td>
<td>FLAGN; list all warning and diagnostic messages</td>
<td></td>
<td>FLAGE; list all but the warning messages</td>
<td></td>
</tr>
<tr>
<td>the amount of storage in BUF is included in SIZE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC 1108</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>Indicate the object code listing, object module and link edit decks are to be suppressed if an E level error message is generated by the compiler</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SUPMAP(NO)</td>
<td></td>
</tr>
<tr>
<td>Indicate the type of spacing desired on the source listing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SPACE1; single space</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SPACE2; double space</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SPACE3; triple space</td>
<td></td>
</tr>
<tr>
<td>Indicate what kind of movement of computational fields is desired</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TRUNC; if the number of digits in the sending field is greater than the number of digits in the receiving field, the arithmetic item is truncated to the number of digits specified in the PICTURE clause of the receiving field</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NOTRUNC; the movement of the item depends on the size of the receiving field (fullword, halfword)</td>
<td></td>
</tr>
<tr>
<td>Indicate which character should be accepted to delineate literals and to be used in the generation of figurative constants</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>QUOTE; (') is the acceptable character</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>APOST; ('') is the acceptable character</td>
<td></td>
</tr>
<tr>
<td>Specify the output from the compiler is to be accepted even if errors are detected</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done through JCL</td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC</td>
<td>1108</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Specify that the checking of punched card sequence numbers (columns 1-6) is to be ignored</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
<tr>
<td>Specify the matched names of CORRESPONDING data names are to be listed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>Specify an alphabetic list of all data names and file names is to be printed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D</td>
</tr>
<tr>
<td>Specify a detailed list of diagnostics is to be printed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>E</td>
</tr>
<tr>
<td>Specify the alternate CUBOL compiler is to be called</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>J</td>
</tr>
<tr>
<td>Specify that a list of items requested through the COPY and INCLUDE verbs is to be produced</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>K</td>
</tr>
<tr>
<td>Specify a list of procedure names with the same first five characters is to be produced</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>M</td>
</tr>
<tr>
<td>Specify that the source listing is to contain only those items requested by options</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N; if absent E, I, and K are assumed</td>
</tr>
<tr>
<td>Specify the updated module is to be punched on cards</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>S</td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC</td>
<td>1108</td>
</tr>
<tr>
<td>--------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Specify that the label will consist of the first five characters preceded by one element - specified unique character</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>Specify that the contents of columns 72-80 are to be ignored</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>Specify that the compiler has encountered a subprogram rather than a main program</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>V; the generation of a starting address is suppressed</td>
<td></td>
</tr>
<tr>
<td>Specify that the correction deck is to be listed before the compilation listing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>Specify the run is to be aborted if an error is found</td>
<td></td>
<td></td>
<td>Done through JCL</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Specify that the formation of information normally given to the diagnostic routine is to be suppressed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Z</td>
<td></td>
</tr>
<tr>
<td>Specify the name and version of the source element</td>
<td></td>
<td></td>
<td>Done through JCL</td>
<td></td>
<td>N1/V1; where N1 is the name of the source element and V1 is the version number</td>
<td></td>
</tr>
<tr>
<td>Specify the name and version of an updated source language module, i.e. where to store it</td>
<td></td>
<td></td>
<td>Done through JCL</td>
<td></td>
<td>N2/V2; where N2 is the name of the module and V2 is the version number</td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC 1108</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Specify the name and version of the updated object module, i.e., where to store it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N3/V3 where N3 is the object module name and V3 is the version number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC 6600</td>
<td>IBM 360</td>
<td>UNIVAC 1108</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------</td>
<td>---------</td>
<td>-------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the amount of main storage available for compilation</td>
<td>SIZE=yyyyy; indicates yyyyyy bytes of storage are available SIZE=yyyK; indicates yyyK bytes of storage are available SIZE=999999; instructs the compiler to obtain as much main storage as it can</td>
<td>OPT=0; keep object program storage requirements to a minimum at the expense of execution time OPT=1; causes object program execution time to be reduced at the expense of storage OPT=2; includes OPT=1, but also requests the compiler to optimize the machine instructions generated for certain DO-loops and expressions in subscript lines</td>
<td>STMT(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the type of compiler optimization desired</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC</td>
<td>1108</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Permit the programmer to name the load module that will be created by the linkage editor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OBJNM=xxxxxxxxx; must begin with an alphabetic character</td>
<td></td>
</tr>
<tr>
<td>Permit the operating system to properly handle interrupts on the IBM models 91 and 195</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OBJIN; if execution takes place on the IBM 91 or 195 computers OBJOUT; if execution does not take place on the IBM 91 or 195 computers</td>
<td></td>
</tr>
<tr>
<td>Cause the compiler to construct larger dictionaries</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EXTDIC; of the dictionary block size is 1K bytes a dictionary 1.5 times that of normal size is used; the dictionary used is 3.5 times normal size if the block size was greater than 1K bytes</td>
<td></td>
</tr>
<tr>
<td>Specify the condition for termination after syntax checking if errors are detected</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SYCHKE; if errors of severity &quot;ERROR&quot; or above are found SYCHKS; if errors of severity 'SEVERE' or above are found SYCHKT; if errors of severity 'TERMINATION' are found</td>
<td></td>
</tr>
<tr>
<td>Specify that the preprocessor is to be used</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MACRO(NO)</td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM</td>
<td>360</td>
<td>UNIVAC</td>
<td>1108</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Specify the compiler is to immediately compile the source module produced by the preprocessor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the output from the preprocessor is wanted in the form of a card deck</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify whether the source statements are written in the 48 or 60 character set</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify whether the EBCDIC or BCD character sets is to be accepted</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the extent of the part of each input record that contains PL/I source statements. It can also specify the position of the ANSI carriage control character desired to control the format of the listing produced</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- COMP(NO)
- MACDCK(NO)
- CHAR60; written in the 60 character set
- CHAR48; written in the 48 character set (the compiler will accept both if CHAR48 is coded, however)
- EBCDIC; BCD

- SORMGIN=(mmm,nnn[,ccc])
  where:
  - mmm is the first byte of the field that contains the source statements
  - nnn is the last byte of the source statement field
  - ccc is the byte containing the carriage control character and
    - mmm < nnn < 100 and
    - ccc < mmm or ccc > nnn
<table>
<thead>
<tr>
<th>OPTION</th>
<th>CDC</th>
<th>6600</th>
<th>IBM</th>
<th>360</th>
<th>UNIVAC</th>
<th>1108</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify where to place the object module produced</td>
<td>LOAD; place the object module on the SYSLIN data set</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a card deck is to be produced of the object module</td>
<td>DECK; columns 73-76 contain a code to identify the object module and columns 77-80 contain a 4 digit serial number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the number of lines to be printed on a page of the source listing</td>
<td>LINECNT=XX; where xx is the number of lines desired</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the compiler options are to be listed at the start of compilation</td>
<td>OPLIST(NO)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a listing of the source statements input to the preprocessor is to be produced</td>
<td>SOURCE2(NO)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a listing of the source statements processed by the compiler is to be produced</td>
<td>SOURCE(NO); the statements are either the original source statement or the output from the preprocessor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that for each statement of a DO-group, the block and nesting level should be indicated on the source listing</td>
<td>NEST(NO)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>CDC</td>
<td>6600</td>
<td>IBM 360</td>
<td>UNIVAC 1108</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------</td>
<td>-------------------------------------</td>
<td>---------------------------------------</td>
<td>------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a table of source program identifiers and their attributes should be produced</td>
<td>ATR(NO); also an aggregate length table, giving the length in bytes of all major structured and non-structured arrays will be produced</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a cross-reference table is to be produced</td>
<td>XREF(NO); if ATR(above) and XREF are both specified, the table is combined</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that the source listing should contain the ESD</td>
<td>EXTREF(NO)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a list of the machine instructions generated by the compiler is to be produced</td>
<td>LIST(NO); the list is similar to SYSTEM/360 assembler language instructions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify the minimum level of severity that requires a diagnostic message to be printed</td>
<td>FLAGW; list all diagnostic messages FLAGE; list all but warning messages FLAGS; list only 'severe' and 'termination' messages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify that a formatted listing of the compiler modules, compiler storage, and compiler control blocks is to be produced if an unrecoverable error is encountered</td>
<td>DUMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The author would now like to list those options provided by any/all of the systems which he feels would be of use to the development of an automatic documentation system.

1. A source listing
2. An object listing
3. A cross-reference table
4. Specify the input source/object file
5. Specify the output source/object file
6. Permit the programmer to name his main program module
7. A table of named variables including their type and location
8. A table of label names
9. An indication of what character set is to be accepted (BCD, EBCDIC, ASCII, FIELDATA, etc.)
10. A listing indicating the loop structure and the logical continuity of the program
11. Specify if an alternate version of a compiler is to be used
12. An indication of the time of compilation, execution, etc.
13. An updated source/object was produced
14. An indication of all non-ASA standard usage
15. An indication of those modules which were subcompiled
16. An indication of the main storage and buffer size
17. An indication of which segments are overlays, which segments they can overlay, and the "path" of the overlay
18. Specify what kind of computational field movement was used
19. Indicate whether the quote ("), or the apostrophe (') is to be accepted
20. Indicate what level/capability of compiler/assembler is needed
21. Specify that internal table space have been increased/decreased
22. Specify if the preprocessor was used and a listing of the generated code
23. Indicate what portion of the card contained the source statements

LINKAGE EDITOR/LOADER OPTIONS

A. Options

The second step of the study was to compile a list of the linkage editor/loader options available to the user. (The author regrets that at this time insufficient information could be obtained for the CDC 6600 and UNIVAC 1108 loaders.) These options were then studied in light of their possible usefulness as part of an automatic system of program documentation. A list of the IBM linkage editor/loader options follows.

Linkage Editor Options

1. **Downward Compatible** - makes the load module processable by either the level E or level F linkage editor - it lets the level E reprocess load modules produced by the level F linkage editor (some differences between the levels E and F linkage editors are: the size of main storage used, the number of entries permitted in the ESD or the RLD, the number of segments allowed, the maximum blocking factor allowed).

2. **Hierarchy Format** - the programmer can specify his program to be loaded into either processor storage (hierarchy 0) or IBM 2361 core storage (hierarchy 1) - the program can be block or scatter loaded in either area.
3. **Not Editable** - a module with this attribute has no ESD and can't be reprocessed by the linkage editor. Because no ESD is produced, less space is needed. It is used primarily by the control program.

4. **Only Loadable** - modules with this attribute can only be brought into core using the LOAD macro - this is done because some subsets of the control program use a smaller control table when a load module is invoked with the LOAD macro, thereby requiring a smaller storage area. It must be entered by a CALL or branch instruction - this attribute is used mainly by the control program and it is suggested that the programmer not use this option because it can impair the usability of a module.

5. **Overlay Attribute** - is structured according to OVERLAY control statements - overlay modules can only be blocked-loaded, are not refreshable, are not reenterable, are not serially reusable, and cannot be assigned a storage hierarchy.

6. **Reusability** - this means that the same copy of a load module can be used by more than one task, either concurrently or one at a time. 
   a. **re-enterable** - can be executed by more than one task at a time - this module can't be modified by itself or any other coding during execution.
   b. **serially reusable** - can be executed by only one task at a time - this type of module must initialize itself or restore any instructions modified during previous execution.

7. **Refreshable** - can be replaced by a new copy during execution by a recovery rearrangement routine without changing either the sequence or results of processing.
8. **Scatter Format** - does not need to be loaded in a contiguous block of storage - the programmer specifies the dynamic loading of control sections into noncontiguous or scattered areas within his assigned main storage. The control program can determine a scatter format but for best results the programmer should specify his own control sections which are to be scatter loaded.

9. **Test** - a module is to be tested and has the symbol tables needed by the test translator (TESTTRAN) or the TSO TEST command processor.

10. **Exclusive Call** - the link editor marks the output module as executable when valid exclusive references have been made between segments (However, a warning message is also issued) - the OVERLAY option must also be specified.

11. **LET** - when specified, the linkage editor marks a module as executable even though it has found a severity 2 error condition - (example: unresolved external references, valid or invalid exclusive calls requested by an overlay program, an error on a link-edit control statement, a library module can't be found, or there is no more space in the directory of the output module library).

12. **No Automatic Library Call** - the linkage editor does not call library members to resolve external references, and the output module is still marked as executable (provided no other errors exist)

13. **SIZE** - Specifies the amount of storage to be used by the level F linkage editor - this option also can be used to specify how much storage is to be used as the load module/text buffer(used to house input/output data).

14. **DCBS** - lets the user specify the block size for the SYSLMOD data set
in the DCB of a DD statement. SYSLMOD is a DD statement which
describes an output module library. It is a partitioned data set
on a DASD and has a member name.
15. **LIST** - lists the control statements processed by the linkage editor
on the diagnostic output data set.
16. **Module Map** - produces a module map on the diagnostic output data set.
17. **Cross-reference table** - produces a cross-reference table on the diagnostic
output data set.
18. **Alternate Output or SYSTERM Option** - produces the linkage editor error/
warning messages on the SYSTERM data set:

**Loader Options**

1. **MAP** - same as 16 above.
2. **RES** - an automatic search of the link pack area queue is to be made.
3. **CALL** - an automatic search of the SYSLIB data set will be made.
4. **LET** - same as 11 above.
5. **SIZE** - same as 13 above.
6. **EP = name** - specifies the external name assigned to the entry point
of the loaded program.
7. **NAME = name** - specifies the name to be used to identify the loaded
program to the system.
8. **PRINT** - write informational and diagnostic messages on the SYSLOUT data set.

**B. Results and Remarks**

All of the above mentioned options would provide useful information
to the user. Thus, the author feels that they should all be included in
an automatic documentation system.
CONCLUSION

In conclusion, we can see that even though the three general operating systems (IBM, CDC, Univac) reviewed in this report offered a myriad of options, only a few of them are common. However, this does not seem to be a hindrance because the individual compilers, assemblers and linkage-editors/loaders build tables and produce outputs which provide much valuable information for automatic documentation.

At the present time, the operating systems have not been studied to a sufficient depth to determine if any additional information for documentation can be derived from other operating system components. It is hoped that further investigation will also reveal if modification to the operating systems will be necessary to obtain this data.
APPENDIX D

AUDIO DOCUMENTATION EXPERIMENT
A TEST OF AN AUDIO DOCUMENTATION TECHNIQUE

E. R. Story
Roger W. Elliott

Texas A&M University
College Station, Texas
77843

December 1972
ABSTRACT

A technique for documenting programs by means of an audio recording is proposed.

An experiment designed to test the effectiveness of the technique was conducted. The results of this experiment indicate that in some circumstances programmers can use audio documentation as effectively as more conventional forms of documentation.

In general, audio documentation is far easier and cheaper to prepare.
INTRODUCTION

Even though the importance and value of proper documentation have been extensively discussed, most computer programs are not adequately documented. Since the value of documentation is virtually unquestioned and its utilization is so inconsistent, then perhaps an alternative to the conventional documentation package needs to be developed. The purpose of this paper is to present an alternative and to evaluate its effectiveness.

The technique proposed makes use of audio tapes for storing the majority of the documentation for a specific program. An experiment was designed to test the utility of this for program maintenance as compared to more conventional forms of documentation.
A STANDARD FOR AUDIO DOCUMENTATION

The audio documentation package developed for this experiment contains the following four items:

(1) Cover Sheet
(2) Audio Tape
(3) File Information Sheet (if required)
(4) Program Listing.

The cover sheet (fig. 1, p. 3) is a table of contents to the audio tape. The first section provides general information such as program name, programmer, source language, date, tape speed and type of tape. The second section provides the table of contents for the tape. One of the major disadvantages of the audio technique is the difficulty in finding the desired information. The table of contents is designed to alleviate this problem. For the major components of the program, the table of contents provides: line number of the program listing, paragraph or routine name, and a footage meter reading indicating where on the tape the detailed comments for the particular section may be found. If the programmer desires to listen to a particular portion of the tape, he can then use either the fast forward or fast rewind to position the tape to the desired location.

As can be seen from the table of contents, the tape is composed of two main sections or parts. The first should always contain the following sections: (1) Identification, (2) Overview and I/O, (3) General Flow, and (4) Symbols and Variables. The second major section contains
the detailed descriptions of small segments of the program.

The Identification section always begins by stating the program name. This requirement is imposed for two reasons: (1) To insure the tape has not been mislabeled, and (2) To allow meter synchronization.

The Overview and I/O section includes such items as source language, system or application, entry and exit points, external references, tables, description of input and output, special register usage, etc.

The General Flow section could be considered a verbal flowchart depicting the gross logic of the program. This section describes the major segments of the program and provides the general purpose or function of each major program paragraph or subroutine and indicates the order in which segments are performed or called.

The Symbols and Variables section is self explanatory. Sequentially following the program listing each symbol or variable is named, its line number given and its purpose or usage described.

The most important advantage of audio documentation is provided by the remainder of the tape wherein a detailed description is provided for each paragraph or subsection of the program. The advantage is that much more detail can be presented on the tape than would normally be provided by any of the more conventional forms of documentation.
Table of Contents

1. Program Identification
2. Program Overview and Input/Output
3. Segmentation and General Flow
4. Glossary
5. Description of Program Segment Logic
   a. 
   b. 
   c. 
   d. 
   e. 
   f. 
   g. 
   h. 
   i. 
   j. 
   k. 

Figure 1 - Audio Documentation Cover Sheet
AN EXPERIMENT

As previously stated, the purpose of this experiment was to test the utility of this form of documentation program maintenance compared to those of the more conventional forms of documentation. One of the first decisions that had to be made was to determine which programming language should be used for the experiment. Since the purpose of the experiment was to evaluate two different types of program documentation, most of the higher level languages were eliminated because, to a certain extent, they are all self-documenting. Assembly language was therefore chosen.

The next decision was to limit the programs to between 250 to 350 lines of code. This was made for several reasons, the most important being that a more difficult program would have placed an unreasonable burden on the volunteer test subjects.

Another decision was to determine how well the two programs should be "commented." Any well-developed set of standards for a program documentation package would require that the program listing be well commented, especially if it were written in assembly language. However, a decision was made to include no comments within the source code. This decision was made for two major reasons. The first was that the experiment was being designed to evaluate program maintenance time of audio documentation against the more conventional forms of documentation (mainly flowcharts), and any comments contained in the source listing itself could bias the results. The second reason was
more of a qualitative one. If the two programs were to be partially "commented," then the questions arose as to which parts should be "commented" and the level of detail.

A summary of how the experiment was conducted is presented below.

(1) Two programs were selected. The programs were S/360 assembler programs about 300 statements in length.

(2) For each program an audio documentation package was prepared and a more conventional documentation package consisting of a textual description, flow charts, etc., was also prepared.

(3) 12 Programmers were selected and randomly divided into two groups, 1 and 2.

(4) Programming group 1 was given the audio documentation package for program A, and the conventional documentation package for program B, Group 2 was given the conventional package for program A, and the audio documentation package for program B.

(5) Each programmer was asked to work independently and, using only the documentation provided, make a specified change to each program. The programming time required to make the change using audio documentation and the time required to make the change using conventional documentation were then determined.

(6) An Aspin-Welch t-test analysis was then performed to determine if there were significant differences between the
times.

The experiment's original scope was quite narrow in that it was limited to comparing only the average time required to make changes to a program using audio documentation and the average time required to make the same changes using standard documentation.

No attempt was made to compare the total amount of time required to prepare the respective types of documentation, although the audio documentation is easier to prepare and takes much less time to produce. No attempt was made to perform any type of qualitative evaluation of whether the documentation did meet the established guidelines.
ANALYSIS

When the programmers had completed their changes and the totals calculated as shown on the following page, it was readily apparent that the results were either contradictory or inconclusive. For program "A", the average time of the programmers who had used conventional documentation was less, but for program "B", the average time of the programmers who had used audio documentation was the shorter. When the two programs were compared, "B" had the lower average time, and when totals were run by tapes of documentation, the audio packages had the lower average time.

Thus, it was decided to make four tests. The first two tested the average time of audio documentation versus conventional documentation for the two programs separately. For these, a two-tail test at the 0.05 level of significance, was conducted. The third test compared the two programs themselves, i.e. the average time required to change program "A" vs. the average time to change program "B". This was again a two-tail test; however, it was at the 0.1 level of significance. A larger than usual risk of making a Type I error could be accepted for this test in order to reduce the risk of making a Type II error. This risk needed to be reduced because test four should not have been conducted unless the hypothesis of test three was accepted. Test four compared the overall average time to make the changes to both programs using audio documentation vs. the overall average time.
<table>
<thead>
<tr>
<th>Programmer</th>
<th>Program &quot;A&quot;</th>
<th></th>
<th>Program &quot;B&quot;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Audio</td>
<td>Conventional</td>
<td>Audio</td>
<td>Conventional</td>
</tr>
<tr>
<td>1</td>
<td>59</td>
<td></td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>43</td>
<td>34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>35</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>39</td>
<td></td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>67</td>
<td></td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>42</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>39</td>
<td>38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>43</td>
<td></td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>54</td>
<td></td>
<td>68</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>42</td>
<td>29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>35</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>33</td>
<td></td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>TOTALS:</td>
<td>295</td>
<td>236</td>
<td>197</td>
<td>288</td>
</tr>
</tbody>
</table>

Total "Audio" Time:  295 + 197 = 492
Total "Conventional" Time:  236 + 288 = 524

Table 1. Times (in minutes) required to make changes to two programs by 12 programmers using different types of documentation.
using conventional documentation. Obviously, the validity of this test could be subjected to severe criticism if test three indicated a significant amount of difference between the two programs. The fourth test was two-tail test at the 0.05 level of significance. None of the tests turned out to be statistically significant.
OBSERVATIONS & CONCLUSIONS

There are probably several reasons why none of the tests were statistically significant. Some of these are as follows:

1) The programs were too short.
2) The required changes were too minor.
3) An insufficient number of samples was taken.
4) Variances among individual programmers' abilities are so large that only very large differences between sample means will be statistically significant.
5) There are too many variables to isolate whether or not the differences can actually be attributed to the type of documentation used.
6) There actually is no real significance between the sample populations.

The author believed that (1) and (2) above may have played very important roles in this experiment. These limitations were accepted so as not to impose a hardship upon the volunteers who were making the changes. However, these limitations may have precluded the observations of any significant differences in the experiment.

Number (3) above does not appear to be near as critical as (4). The standard deviation among the programmers was so large, that had the difference between the means remained constant, the sample size could have been increased to 120 and still not have been statistically significant.
The large number of variables, number (5) above, is a problem that is extremely difficult to resolve in an experiment of this nature. The ideal situation would be to have only one program that is being changed, by two groups of programmers. One group using audio documentation, the other using conventional. However, this presents the problem of the difference in abilities of the two groups of programmers. Thus, the question could arise as to whether or not the difference should be attributed to the type of documentation used or to programmer abilities. To resolve this question, one can use two programs, and thus both groups get to use both types of documentation. This immediately leads to the comparison of the difficulty of the two programs. All of which brings back the original premise that there may possibly be too many variables involved to obtain a test that is statistically significant. The authors do not believe this is the case, but it is being presented as an alternative possibility.

The last possibility (#6 above) is that there is no significant difference between program maintenance using audio documentation and program maintenance using conventional documentation. Looking back at the data, and after evaluating the four tests, this has to be considered a very strong possibility.
SUGGESTIONS FOR FURTHER RESEARCH

Although the results of this experiment failed to prove that one type of documentation is better than the other for program maintenance purposes, the authors are convinced that additional research should be conducted into the use of audio techniques. The experiment demonstrated that much more detail can be easily provided by audio documentation than by conventional documentation; however, the experiment as conducted was somewhat limited and may have partially nullified this advantage.

To test fully the effectiveness of the two techniques, longer programs should be used and more complicated changes attempted. The changes should affect more than one section of the programs being tested, and thus, a better evaluation of the differences of the techniques should be provided.

Indications are that audio documentation is easier and faster to prepare; however, a detailed cost analysis study (which was beyond the scope of this project) would be required to prove or disprove this point.

Several of the programmers admitted that they had no real idea as to what to expect from the audio documentation, and that they could probably utilize it better if they were to use it again. In future experiments, programmers should be provided better training in the use of audio documentation prior to the actual experiment itself.
SUMMARY

The experiment conducted in this paper demonstrated that for assembly language programs of approximately 300 lines of code, program maintenance time is approximately the same when using either audio documentation techniques or conventional documentation techniques, thus, audio techniques should be considered as an alternative to the conventional forms of program documentation. Further research should be conducted to determine the effectiveness of the technique for longer programs requiring more complicated changes.

The most important advantage of audio documentation is the volume and degree of detail it can provide about the program. There is absolutely no realistic way that conventional documentation can compete in this area. This should prove to be especially advantageous when the maintenance programmer has not written the original code.

A second advantage of audio documentation may be that it is easier to prepare. Drawing a neat, detailed flowchart using a template is a time consuming job. Drafting and processing the textual material to support a conventional documentation package is even more time consuming.

The detail provided by audio documentation becomes even more important whenever a program listing is poorly commented. Programmers are often hesitant to fully comment a program before it is completely debugged because of the numerous changes that often have to be made. If comments have already been included, then a double change is usually
necessitated. They usually have good intentions of fully commenting the program after it is completely debugged, but for one reason or another, they seldom do. The use of audio documentation greatly alleviates this problem in that the comments are merely placed on the tape.

Perhaps a fourth advantage of the audio documentation technique is that it virtually forces an organized approach to program maintenance, whereas a programmer using conventional documentation may or may not have an organized plan of attack.

Two of the disadvantages of audio documentation are inherent to any sequentially accessed file: (1) It is more difficult to change, and (2) Advance/rewind time to the desired position on the tape may be significant.

Thus, audio documentation is presented as an alternative to the conventional documentation process. Whether or not it should be implemented at any particular data processing center depends upon many factors. These include the characteristics of the projects being documented, the technical environment, and management's commitment to the concept.
APPENDIX E

TEXT EDITORS
AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Working Paper No. 2
July 14, 1972
Text Editors

by
Hank Goggan

Texas A&M University
Texas Engineering Experiment Station

E/1-a
ABSTRACT

This paper is a summary of current methods available for the editing of text on a computer. A brief discussion of the advantages of using computer-aided text editing for documentation of computer software systems is followed by short reports on present systems, and a conclusion. The principal topic of each report is a contrast of features characteristic of each system.
# TABLE OF CONTENTS

I. INTRODUCTION .............................................. E-3

II. DISCUSSION .................................................. E-4

III. TYPICAL STRUCTURE OF A TEXT EDITOR ................. E-5

IV. SYSTEM REPORTS ............................................. E-7
   a.) APL TEXT EDITOR ........................................ E-8
   b.) MTST ..................................................... E-10
   c.) ASTROCOMP ............................................... E-12
   d.) DATATEXT ............................................... E-13
   e.) EDIT ..................................................... E-14
   f.) ED PROCESSOR ............................................ E-15
   g.) ATS ....................................................... E-16
   h.) TEXT/360 ................................................ E-17
   i.) HES ....................................................... E-19
   j.) FRESS ..................................................... E-21
   k.) REDIT/RUNOFF ......................................... E-23
   l.) TEXT ...................................................... E-24
   m.) NLS ....................................................... E-25

V. CONCLUSIONS .................................................. E-27

VI. TABLES AND CHARTS .......................................... E-29
INTRODUCTION

"With the advent of inexpensive terminals that communicate directly with a general-purpose computer, there has been a noticeable movement in the computing industry towards utilizing the resources of the computer in many new, non-numerical applications. For example, on-line creation and modification of programs and their documentation have become widely accepted as productive and cost-effective uses of the computer. In fact, it has been realized that the facilities provided by a time-sharing system's central editing program and its command language are among the most important determinants of the system's convenience, power, and consequent utility. Along the same lines, special-purpose computer-assisted text editing packages have become accepted in industry and government for the preparation or printing of technical manuals, proposals, and other documents in which many updates are necessary and revision time is at a premium."¹

It, therefore, seems evident that some form of text editing package would be extremely beneficial in the preparation of documentation for computer programs. The ability to produce informative manuals (user guides, operators manuals, maintenance procedures, etc.), descriptions of the methods employed, formal proposals, at minimal cost and effort, with an increase in efficiency, is no longer a nicety, but has now become a necessity.

Discussion

Typically or traditionally, the method of composing some form of text has been merely a typing/proofreading/retyping task - and it is still the same today - but in a slightly modified and extremely more efficient form. Today the editing responsibilities of providing a clean-looking, errorless manuscript have been placed on the computer. This computer text editor still requires much of the same information as the old-fashioned human editor; the text itself, the form or format that it is to be in, plus the knowledge of what is correct and what is not. So in short, the computer text editor is a software package which takes text as input, stores it, modifies it according to the authors' wishes (accomplished by edit codes), and outputs it whenever and however the author specifies.

The sum product of this is the elimination of the communication between author and publisher. The author (programmer or system designer) is now capable of producing and publishing his own works with a tremendous reduction of time and effort. This would enable the author of a computer program to fully document or explain his program with much less effort and frustration than before. Furthermore, if his program were altered in any way, additions or deletions of documentation would only be required in the appropriate places, the additions proofread, and the entire document reprinted on high-speed line printers or the like. This capability of using high-speed line printers also provides an efficient means of producing many hard-copies of the authors' work for distribution.
Structure of a Text Editor

The typical structure of a text editor is briefly outlined in Figure 1. Several different forms can be used to input the text (keypunched, typewritten code, etc.). This input is generally decoded and fed into the computer text editor. If work is needed on the text, it can now be accomplished or it can be accomplished at a previous or prior time, depending on the type of text editor. Then the desired output is specified and printed or stored for later work or printing.
Systems Reports
NAME OF SYSTEM       APL TEXT EDITOR
PRICE                NOT APPLICABLE
ON-LINE/OFF-LINE     ON-LINE
MACHINE USED ON      IBM/360, CDC 6600
LANGUAGE WRITTEN IN  NOT AVAILABLE
INPUT FORM(S)        TELETYPED
INPUT DEVICE(S)      IBM 2741 COMMUNICATIONS TERMINAL
OUTPUT FORM(S)       TELETEYPEWRITTEN
OUTPUT DEVICE(S)     IBM 2741 COMMUNICATIONS TERMINAL

DISCUSSION:

The APL/TEXT EDITOR is used primarily for form letters or short reports. A limiting factor in the size of the data base is determined by the amount of space allocated to the APL workspace.

All input and output is through the IBM 2741 Communications Terminal. This means that the input rate is determined by the speed of the typist; while the output is at the rate of 140 words per minute.

The system is capable of producing well formatted text (one column only) that can be right justified if specified. Typical usage has shown that less than five pages of text is the maximum amount easily handled. The ability to change typing elements provides for flexibility in fonts and styles of type. This would be valuable for short scientific papers. Text can also be added at any point of the text, thus enabling its usage for form letters.

The edit commands and rules for this system are relatively simple and easy to learn. Experience has shown that only an hour or two is
required to learn how to use the system efficiently. (See chart for list of capabilities.)

COMMENTS:

The best use of this system would be to provide small reports on the status of computer programs or brief overall characteristics (general documentation) for a global view.

It is one of the better small text editing systems.
<table>
<thead>
<tr>
<th>NAME OF SYSTEM</th>
<th>MAGNETIC TAPE SELECTRIC TYPEWRITER (MT/ST)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRICE</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>OFF-LINE</td>
</tr>
<tr>
<td>MACHINE USED ON</td>
<td>SMALL CONTROL/MEMORY UNIT</td>
</tr>
<tr>
<td>LANGUAGE WRITTEN IN</td>
<td>NOT APPLICABLE</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TELETYPED, (TAPE)</td>
</tr>
<tr>
<td>INPUT DEVICE(S)</td>
<td>IBM 2741 Communications Terminal</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TELETYPED</td>
</tr>
<tr>
<td>OUTPUT DEVICE(S)</td>
<td>IBM 2741 Communication Terminal</td>
</tr>
</tbody>
</table>

DISCUSSION:

The MT/ST provides neat-looking (but unjustified) form letters and manuscripts. It is primarily used for documents of high-use and low-change rate.

The only editing allowed in MT/ST is to substitute equal length character strings; to effect a substitution of a larger for a smaller string, the entire manuscript with the correction must be copied onto a second tape. The correct place for an edit is located by printing out the contents of the tape until the area of the edit is approached; the operator must then manually stop the processor and retype the line.

Printing is done on the typewriter at the rate of 150 words per minute, and various fonts and type sizes may be used by changing the type ball.

COMMENTS:

The cost and storage of tapes plus the intricacies of revision
(but done by the original typist) are two points which must be considered.

For technical documents which are single column format, use of the MT/ST makes correcting easy; once the document is completed, the tapes can be erased and reused. The MT/ST would be valuable if it is used primarily as a production tool and not as a storage device.
<table>
<thead>
<tr>
<th>NAME OF SYSTEM</th>
<th>ASTROCOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRICE</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>OFF-LINE</td>
</tr>
<tr>
<td>MACHINE USED ON</td>
<td>DEC PDB8L 8K MINICOMPUTER</td>
</tr>
<tr>
<td>LANGUAGE WRITTEN IN</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TELETYPEDWRITTEN, (TAPE)</td>
</tr>
<tr>
<td>INPUT DEVICE(S)</td>
<td>TELETYPEDWRITER</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TELETYPEDWRITTEN, PHOTO-COMPOSITION</td>
</tr>
<tr>
<td>OUTPUT DEVICE(S)</td>
<td>TELETYPEDWRITER, PHOTO-COMPOSITER</td>
</tr>
</tbody>
</table>

**DISCUSSION:**

The ASTROCOMP system is similar to MT/ST in its intended use (small amounts of text, form letters, etc.). This system does have an added advantage of having up to four typewriters connected to a single control unit (DEC PDP-8).

The basic editing command is SUBSTITUTE. As in the APL and MT/ST systems, the "old" (text to be replaced) is typed in to locate it (thus uniquely identifying the text) and the "new" text is then added - replacing the old text.

This system is slightly more powerful than either APL or MT/ST and is a bit more reasonable than MT/ST in its editing features.

It is capable of producing output in photo-composition form, which is a feature neither APL nor MT/ST have.
NAME OF SYSTEM: CALL/360: DATA TEXT

PRICE: NOT AVAILABLE

ON-LINE/OFF-LINE: ON-LINE

MACHINE USED ON: IBM 360

LANGUAGE WRITTEN IN: NOT AVAILABLE

INPUT FORM(S): TELETYPETRITTEN

INPUT DEVICE(S): IBM 2741

OUTPUT FORM(S): TELETYPETRITTEN

OUTPUT DEVICE(S): IBM 2741

DISCUSSION:

DATATEXT is a terminal-oriented, on-line system for data entry, change, and retrieval. It is comprised of an IBM 2741 Communications Terminal in the user's office, phone linking connections, and an IBM 360.

The system is comparable to the TEXT 360 system as far as most formatting capabilities are concerned, but the DATATEXT system is limited to the typewriter terminal for output. (This means 140 words/minute vs. line-printer speed for TEXT 360.)

However, there is an advantage to having typewriter terminals for output; that being the ability to have the document (or sections of) printed out whenever there is a terminal. This saves-time because distribution of the document is not necessary.
<table>
<thead>
<tr>
<th>NAME OF SYSTEM</th>
<th>EDIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRICE</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
</tr>
<tr>
<td>MACHINE USED ON</td>
<td>CDC 6000</td>
</tr>
<tr>
<td>LANGUAGE WRITTEN IN</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TYPEWRITTEN, CRT KEYED</td>
</tr>
<tr>
<td>INPUT DEVICE(S)</td>
<td>TYPEWRITTEN, CRT</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TYPEWRITTEN, CRT DISPLAY</td>
</tr>
<tr>
<td>OUTPUT DEVICE(S)</td>
<td>TYPEWRITTEN, CRT</td>
</tr>
</tbody>
</table>

**DISCUSSION:**

The Text Editor (EDIT) subsystem executes data file manipulations specified by the time-sharing terminal user. These manipulations are performed on a new file or a data file which has been saved in the KRONOS permanent file system.

EDIT allows the user to edit a data file. The data file being edited is known as the search file. During editing, the search-pointer identifies the line of the search file that is currently accessible. The search-pointer can be moved forward and backward during editing to specify a new line. The search-pointer is always positioned at the beginning of a line.
<table>
<thead>
<tr>
<th>NAME OF SYSTEM</th>
<th>ED PROCESSOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRICE</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
</tr>
<tr>
<td>MACHINE USED ON</td>
<td>UNIVAC 1100</td>
</tr>
<tr>
<td>LANGUAGE WRITTEN IN</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TELETYPewriter</td>
</tr>
<tr>
<td>INPUT DEVICE(S)</td>
<td>TELETYPewriter</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TELETYPewriter, PRINTED, CARDS</td>
</tr>
<tr>
<td>OUTPUT DEVICE(S)</td>
<td>TTY, PRINTER, CARD-PUNCH</td>
</tr>
</tbody>
</table>

**DISCUSSION:**

The ED processor proceeds sequentially through the text. It is therefore more efficient to perform editing operation in a more or less sequential manner starting at the beginning of the text. There are certain processes within the editor which if indiscriminately interrupted can cause the processor to fail. To protect against this, the processor is designed to break at certain points when it is safe to do so.

**Comments:**

Not much information on this text editor was available at the present time. However, it appears to be below average as far as flexibility and ease of learning is concerned when compared with other text editor systems.
<table>
<thead>
<tr>
<th>NAME OF SYSTEM</th>
<th>ADMINISTRATIVE TERMINAL SYSTEM (ATS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRICE</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
</tr>
<tr>
<td>MACHINE USED ON</td>
<td>IBM/360</td>
</tr>
<tr>
<td>LANGUAGE WRITTEN IN</td>
<td>IBM 360 ASSEMBLY</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TYPED, PUNCHED</td>
</tr>
<tr>
<td>INPUT DEVICE(S)</td>
<td>TYPEWRITER KEYBOARD, CARD READER, MAGNETIC TAPE</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TYPED, PUNCHED</td>
</tr>
<tr>
<td>OUTPUT DEVICE(S)</td>
<td>IBM 2741, CARD PUNCH, LINE-PRINTER</td>
</tr>
</tbody>
</table>

DISCUSSION:

This system consists of control and functional programs that permit many different text-processing and data-handling activities to be carried on simultaneously through different typewriter terminals attached to an IBM System/360. Written to operate under OS/360, the Administrative Terminal System runs in a multi-programming environment. It will run concurrently with and independently of other tasks in other partitions/regions.

COMMENTS:

The ATS system is one of the most versatile text processing system that is available. It is powerful and flexible. The output can be in any conventional form and written out at any time.

This system provides the best approach to solving documentation procedures and requirements in regards to cost, ease, and efficiency, as of present.
NAME OF SYSTEM  TEXT/360  
PRICE  FREE FROM IBM  
ON-LINE/OFF-LINE  OFF-LINE  
MACHINE USED ON  IBM/360  
LANGUAGE WRITTEN IN  PL/I IBM 360 ASSEMBLY  
INPUT FORM(S)  CARDS  
INPUT DEVICE(S)  CARD READER  
OUTPUT FORM(S)  CARDS, PRINTED  
OUTPUT DEVICE(S)  CARD PUNCH, LINE-PRINTER  

DISCUSSION:

The TEXT/360 system is strictly an off-line text editor. All text is initially read into the computer via keypunched cards. Edit codes are embedded in this text prior to the input.

Once the cards have been read in, the text is stored on magnetic tape. Any updating or editing after this is also done by cards, using slightly different codes.

This is rather cumbersome and time-consuming for the initial setup, but the power of this system comes from the unlimited amount of text that can be read in, the relative ease of updating, and the elaborate printed output formats for the text.

The only feature not readily available on this system is the ability to change fonts and type style due to the problems involved in changing print chains.

COMMENTS:

This is a well-established and widely used system. Its principle
value comes from the wide range of features available. Also, the only expense to a computer installation would be the running of the software package itself. It requires no special equipment and on-line connection expense. It is also very easy to learn and use.

The system provides excellent text editing capabilities for typical text production needs.
The Hypertext Editing System is a flexible, CRT-based system allowing full editing and formatting capabilities. It is oriented towards "typeset" output (using a computer line printer) as well as flexible input and on-line editing. A lightpen and a set of "function keys", under program control, are used to indicate to the system the nature of the edit to be performed. The portion(s) to text to which the function applies are then indicated by pointing at the text with the lightpen. No command codes for the functions need be remembered, and no extra typing is required to indicate a context string.

Editing commands are INSERT, DELETE, SUBSTITUTE, REARRANGE, and copy.

Many formatting options are available so that text may be formatted both for on-line display and hard-copy printouts. Usually the TEXT/360 program is used for final hard-copy printing.

HES has a unique, however, complicated data structure. A practical
example of a HYPERTEXT system might be an on-line encyclopedia or a set of programming and systems reference manuals, with each cross-reference lightpen sensitive.

COMMENTS:

Even though many nice features are available through this system - easy editing by lightpen and random accessing to any point within the text, providing efficient time-saving updates, it is more costly due to the expensive CRT with lightpen facilities and large amounts of computer time required to operate the system.
<table>
<thead>
<tr>
<th>NAME OF SYSTEM</th>
<th>FILE RETRIEVAL AND EDITING SYSTEM (FRESS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRICE</td>
<td>$35,000 or $500/month</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
</tr>
<tr>
<td>MACHINE USED ON</td>
<td>IBM/360</td>
</tr>
<tr>
<td>LANGUAGE WRITTEN IN</td>
<td>NOT AVAILABLE</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TELETYPEWRITTEN, KEYED</td>
</tr>
<tr>
<td>INPUT DEVICE(S)</td>
<td>IBM 2741 COMMUNICATIONS TERMINAL</td>
</tr>
<tr>
<td></td>
<td>IBM 2260 CRT</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TYPEWRITTEN, PHOTO-COMPOSITION</td>
</tr>
<tr>
<td>OUTPUT DEVICE(S)</td>
<td>IBM 2741 COMMUNICATIONS TERMINAL</td>
</tr>
<tr>
<td></td>
<td>LINE PRINTER</td>
</tr>
<tr>
<td></td>
<td>PHOTO-COMPOSER</td>
</tr>
</tbody>
</table>

**DISCUSSION:**

FRESS is a sophisticated and cost effective text manipulation system, commercially available, and capable of supporting a spectrum of terminals in such a way that all functions are available even on the lowest power terminals.

FRESS is the production version of the predecessor prototype HES. Beyond normal editing and formatting commands of most text editors, FRESS includes completely arbitrary size string edits, pattern scanning, keyword retrieval, photo-composition output, interfile linking and editing, and protection of files and blocks of text by passwords.

**COMMENTS:**

Eventhough the price is somewhat steep, the capabilities and usage of this system far exceed any other system reviewed to date.

The commands are simple, but extremely powerful, and easily learned.
FRESS was designed more in the light of being dynamic than other systems, therefore indicating that the system can be "tailor-made" for the customer.
NAME OF SYSTEM  REDIT/RUNOFF
PRICE  NOT AVAILABLE
ON-LINE/OFF-LINE  ON-LINE
MACHINE USED ON  TSS/360
LANGUAGE WRITTEN IN  NOT AVAILABLE
INPUT FORM(S)  TELETYPewriter
INPUT DEVICE(S)  TELETYPewriter
OUTPUT FORM(S)  TELETYPewriter, PRINTED
OUTPUT DEVICE(S)  TELETYPewriter, LINE-PRINTER

DISCUSSION:

A terminal is connected to TSS/360 by telephone lines of text are typed on the terminal, transmitted to TSS/360 and permanently stored. The text is entered without concern for staying within certain margins or running of the bottom of the page. Special format control words are included in the document text which informs the document printer as to margins, size of paper, spacing, paragraphs, etc.

Delete character and line symbols allow typing errors to be corrected as they are made. Typing errors which have been stored are easily corrected with the text editor. In addition a document can be revised by adding, deleting or re-arranging sections of text.

The document can be printed at the terminal at various stages of revision.

COMMENTS:

As far as time-sharing editing systems are concerned, this is one of the better thus far reviewed. It provides powerful commands and is relatively easy to learn.
<table>
<thead>
<tr>
<th>NAME OF SYSTEM</th>
<th>TEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRICE</td>
<td>Not Available</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
</tr>
<tr>
<td>MACHINE USED ON</td>
<td>Sigma 6/7/9</td>
</tr>
<tr>
<td>LANGUAGE WRITTEN IN</td>
<td>Not Available</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>Teletypwritten</td>
</tr>
<tr>
<td>INPUT DEVICE(S)</td>
<td>Teletypwritten</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>Teletypwritten</td>
</tr>
<tr>
<td>OUTPUT DEVICE(S)</td>
<td>Teletypwritter, Printer</td>
</tr>
</tbody>
</table>

**DISCUSSION:**

TEXT is Xerox's text editing system. It is an on-line system that is quite similar to ATS and other line text editors.

The only problem that is immediately evident is the fact that it is designed to operate on the Sigma computers, which may hinder using TEXT as a documentation aid for producing automatic program documentation for any programs on any machine.

**Comments:**

Disregarding other factors and considering only text editing capabilities, TEXT is a very powerful and capable text editing system. The commands are simple and it is easy to learn how to use. Definitely it should be high on the list of available text editing systems.
NAME OF SYSTEM: NLS; TNLS (TYPEWRITER NLS); DNLS (DISPLAY NLS)

PRICE: NOT CURRENTLY AVAILABLE OUTSIDE THE ARPA NETWORK

ON-LINE/OFF-LINE: ON-LINE

MACHINE USED ON: DEC PDP 10

LANGUAGE WRITTEN IN: UNAVAILABLE

INPUT FORM(S): TELETYPE, CRT

INPUT DEVICE(S): ANY TERMINAL ACCEPTABLE TO ARPA NETWORK

OUTPUT FORM(S): TELETYPE, CRT, PRINTER

OUTPUT DEVICE(S): ANY DEVICE ACCEPTABLE TO ARPA NETWORK

DISCUSSION:

The NLS text manipulation and editing system were developed by the Augmentation Research Center of Stanford Research Institute. NLS is used as part of ARPA Network Information Center (NIC). NIC performs information retrieval functions.

When working with NLS through a terminal one is at all times constructing, studying, or modifying a file. NLS files have a hierarchical, tree, or outline structure.

NLS has commands which manipulate data on a file level and on the text level. On the file level data is loaded, updated, verified, output and locked or unlocked all through terminal commands. On the text level it is possible to access data in the hierarchy directly by giving a tree address, or relatively according to the current portion being viewed. Commands allow the user to proceed to the next item at the same level,
forward or back, to go up levels or down levels. Text is stored and viewed in upper and lower case founts.

Data is easily updated, move, copied or deleted. It can be viewed on any level with easy transition between levels. Output to any device can be controlled from the terminal.

The NLS system can be used with any terminal accepted by the ARPA Network or with the Special terminal developed by the Augmentation Research Center which uses a 5-key keyboard and a "mouse" to control the cursor in addition to the normal terminal keyboard.

COMMENTS:

This is perhaps the most powerful and versatile text editor reviewed. It's use is at present limited to members of the ARPA Network and personnel at SRI.
Conclusion

Computer aided editing of text has become an established cost-effective use of computers. There exists as wide a range of text editors as there are varieties of text. For small amounts of text, such as memoranda, form letters, brief descriptions, or update reports, systems like the MT/ST, APL text editor, or ASTROCOMP are effective and money-saving. However, for large amounts of text that require more storage, more intricate formats, etc., to produce program descriptions, user manuals, etc., systems like the ATS, TEXT/360, NLS, FRESS would produce the best results.

Any of the systems mentioned previously can be extremely valuable to anyone who uses them, provided that their requirements are in accordance to the system's capabilities and limitations. It is sometimes advisable to employ a combination of these systems to accomplish any desired task.

The principle advantage is not a direct result of these capabilities. It is the indirect effect it has on the documentor. For example, the typical computer programmer enjoys the programming itself, but lacks the desire to sit down and tediously write good documentation for it. The inability to use the computer to help with this task frustrates and annoys him. But with a computer text editor, he is able to type in any information or explanation he likes, changes it at a later date if it is not what he wants, and provides himself and his co-workers with a good, well-formatted and well-documented explanation of what he has done, all with minimal effort.
It can therefore be concluded, that a computer aided text editing system can cut the cost of printed material, stimulate the desire to document, and provide a means to keep the documentation up-to-date.
Foreword to Table I

The following table lists the most of capabilities of text editors. Each text editor feature is briefly described.

In most cases these capabilities are given mnemonic codes which are instructions to the text editor. They are most commonly embedded within the input text itself.

These commands represent the power of the system to update text and to produce formatted text documents.
TABLE I

Capabilities of the Systems

1) TEXT COLUMNS - the ability to have text printed in one or two columns.
2) PAGE/TEXT WIDTH - the ability to specify the number of characters per line; either one or two columns.
3) PAGE/TEXT DEPTH - the ability to specify the number of lines on a page (normal range: approximately 25 to 75 lines).
4) NEW PAGE/NEW COLUMN - for shipping to a new page/column or several pages/columns.
5) RUNNING HEADS AND FEET - allows for running heads (title, subtitle, date, etc.) and running feet, to be printed on each page, only even-numbered, or only odd numbered.
6) RIGHT-HAND PAGES - allows for text to be only printed on one side (right-handed).
7) "AS-IS" TEXT - allows for text to be printed in any format that is inputed (blanks are not squeezed out, or added to aid in right justification).
8) CENTERED TEXT - provides for specified text to be centered, equally between right and left hand margins.
9) LINE-SPACING - specifies single or double-spaced printing.
10) BLANK LINES - provides for skipping lines.
11) PARAGRAPHS - specifies that a line should be skipped and indented a specified or default number of spaces.
12) INDENTIONS - allows for a specified indention for any line of text.
13) HANGING INDENTIONS - allows for the text to be indented for as many lines as specified.
14) COLUMN JUSTIFICATION - causes all the text columns on a page to be equal in length by spacing the columns out to the page depth.
15) LINE JUSTIFICATION - causes all text lines to be spaced out to the line length by the insertion of blank characters between words.
16) TABULAR TEXT - specifies tab settings and indicates the text to be printed at the settings.
17) HYPHENATION - allows the program to hyphenate the last word on a line if needed for justification purposes.
18) FOOTNOTES - allows specified text that is to be printed as a footnote.

19) HEADINGS - provides for line(s) of text that are set apart from the text body and provides a description of the section that follows it.

20) KEEPS - specifies a portion of text, such as a table, heading, or the like, so that it is not split between two columns or pages.

21) TABLES - allows the preparation of tables and charts within the text.

22) HORIZONTAL LINES - provides the ability to print horizontal lines.

23) VERTICAL LINES - provides the ability to print vertical lines.

24) CAPITALIZATION - allows capital letters.

25) UNDERSCORING - allows the underscoring of any text.

26) USER-DEFINED CODES - permits the user to define his own code which can be comprise of combinations of regular edit codes.(This capability is generally used for repetitive edit functions)

27) SUPPLEMENTAL LISTINGS - provides specified text to be printed at specific times. (generally used to create Table of Contents, list of figures, etc.)

28) MULTIPLE PRINTING OF TEXT - allows specified pages or documents to be printed more than once at any given time.

29) REVISION BARS - allows updated text to be indicated by a vertical bar in the margin, signifying that portion of text has been revised.

30) RENUMBERING - permits the automatic renumbering of pages in a document or report after updating (addition or deletion of text).

31) REPAGING - permits the automatic repaging of text.

32) VARIABLE STYLES - allows for the use of special characters within a text. Normally this is limited to the type of print chain available or the kinds of typing elements available.
Foreword to Table II

This table lists what are commonly referred to as the edit commands. These commands differ from the format commands in the fact that they are used to manipulate portions of text to make changes, deletions or insertions within a document instead of formatting the text itself.

These commands generally vary the most in format from one system to the next, but all systems can accomplish these basic functions.
<table>
<thead>
<tr>
<th></th>
<th>Editing Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ADDITION/DELETION OF CHARACTERS</td>
</tr>
<tr>
<td>2</td>
<td>ADDITION/DELETION OF LINES</td>
</tr>
<tr>
<td>3</td>
<td>ADDITION/DELETION OF PARAGRAPHS</td>
</tr>
<tr>
<td>4</td>
<td>ADDITION/DELETION OF PAGES</td>
</tr>
<tr>
<td>5</td>
<td>TRANSFERING CHARACTERS</td>
</tr>
<tr>
<td>6</td>
<td>TRANSFERING LINES</td>
</tr>
<tr>
<td>7</td>
<td>TRANSFERING PARAGRAPHS</td>
</tr>
<tr>
<td>8</td>
<td>TRANSFERING PAGES</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>APL</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>On-Line</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>Typed</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>Typed</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>On-Line</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>Typed</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>Typed</td>
</tr>
</tbody>
</table>

**EDITING FEATURES**

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>APL</th>
<th>MIST</th>
<th>ASTROCOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD/DELETE CHARACTERS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ADD/DELETE WORDS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ADD/DELETE LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER WORDS</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TRANSFER PARAGRAPHS</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER PAGES</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**FORMAT FEATURES**

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>APL</th>
<th>MIST</th>
<th>ASTROCOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEXT COLUMNS</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>PAGE/TEXT WIDTH</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PAGE/TEXT DEPTH</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NEW PAGE/NEW COLUMN</td>
<td>Yes/No</td>
<td>Yes/No</td>
<td>Yes/No</td>
</tr>
<tr>
<td>RUNNING HEADS AND FEET</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RIGHT-HAND PAGES</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>&quot;AS-IS TEXT&quot;</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>CENTERED TEXT</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LINE-SPACING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>BLANK LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>APL</td>
<td>MIST</td>
<td>ASTROCMP</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>FORMAT FEATURES (cont.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HANGING INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>COLUMN JUSTIFICATION</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LINE JUSTIFICATION</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TABULAR TEXT</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HYPHENATION</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FOOTNOTES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HEADINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>KEEPS</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TABLES</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HORIZONTAL LINES</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>VERTICAL LINES</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CAPITALIZATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>UNDERSCORING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>USER-DEFINED CODES</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SUPPLEMENTAL LISTINGS</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MULTIPLE PRINTINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>REVISION BARS</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RENUMBERING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>REPAGING</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>VARIABLE STYLES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>DATA TEXT</td>
<td>EDIT</td>
<td>ED PROCESSOR</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------</td>
<td>-------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>On-Line</td>
<td>On-Line</td>
<td>On-Line</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TTY</td>
<td>TTY</td>
<td>CRT</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TYPED, PRINTED</td>
<td>TYPED</td>
<td>CRT DISPLAY, PRINTED</td>
</tr>
<tr>
<td>EDITING FEATURES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE CHARACTERS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE WORDS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER WORDS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER PAGES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORMAT FEATURES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEXT COLUMNS</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PAGE/TEXT WIDTH</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PAGE/TEXT DEPTH</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>NEW PAGE/NEW COLUMN</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes/No</td>
</tr>
<tr>
<td>RUNNING HEADS AND FEET</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RIGHT-HAND PAGES</td>
<td>Not Available</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>&quot;AS-IS TEXT&quot;</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CENTERED TEXT</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LINE-SPACING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>BLANK LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>DATATEXT</td>
<td>EDIT</td>
<td>ED PROCESSOR</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------</td>
<td>------</td>
<td>--------------</td>
</tr>
<tr>
<td>FORMAT FEATURES (cont.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HANGING INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>COLUMN JUSTIFICATION</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LINE JUSTIFICATION</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TABULAR TEXT</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HYPHENATION</td>
<td>Not Available</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>FOOTNOTES</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HEADINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>KEEPS</td>
<td>Not Available</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TABLES</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HORIZONTAL LINES</td>
<td>Not Available</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>VERTICAL LINES</td>
<td>Not Available</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CAPITALIZATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>UNDERSCORING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>USER-DEFINED CODES</td>
<td>Not Available</td>
<td>Not Available</td>
<td>No</td>
</tr>
<tr>
<td>SUPPLEMENTAL LISTINGS</td>
<td>Not Available</td>
<td>Not Available</td>
<td>No</td>
</tr>
<tr>
<td>MULTIPLE PRINTINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>REVISION BARS</td>
<td>Not Available</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RENUMBERING</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>REPAGING</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>VARIABLE STYLES</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>ATS</td>
<td>TEXT/360</td>
<td>HMS</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
<td>OFF-LINE</td>
<td>ON-LINE</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TTY</td>
<td>CARDS</td>
<td>KEYED INPUT</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>TYPED, PRINTED, CARDS, TAPE</td>
<td>PRINTED</td>
<td>CRT DISPLAY, TYPED</td>
</tr>
<tr>
<td>EDITING FEATURES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE CHARACTERS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE WORDS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER WORDS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>FORMAT FEATURES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEXT COLUMNS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PAGE/TEXT WIDTH</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PAGE/TEXT DEPTH</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NEW PAGE/NEW COLUMN</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes/No</td>
</tr>
<tr>
<td>RUNNING HEADS AND FEET</td>
<td>Yes</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>RIGHT-HAND PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>&quot;AS-IS TEXT&quot;</td>
<td>Yes</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>CENTERED TEXT</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LINE-SPACING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>BLANK LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>ATS</td>
<td>TEXT/360</td>
<td>RES</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----</td>
<td>----------</td>
<td>-----</td>
</tr>
<tr>
<td>FORMAT FEATURES (cont.)</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>HANGING INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>COLUMN JUSTIFICATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LINE JUSTIFICATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>TABULAR TEXT</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HYPHENATION</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>FOOTNOTES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HEADINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>KEEPS</td>
<td>Not Available</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>TABLES</td>
<td>Yes</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>HORIZONTAL LINES</td>
<td>Not Available</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>VERTICAL LINES</td>
<td>Not Available</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>CAPITALIZATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>UNDERSCOrING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>USER-DEFINED CODES</td>
<td>Not Available</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>SUPPLEMENTAL LISTINGS</td>
<td>Not Available</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>MULTIPLE PRINTINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>REVISION BARS</td>
<td>Not Available</td>
<td>Yes</td>
<td>Not Available</td>
</tr>
<tr>
<td>RENUMBERING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>REPAGING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>VARIABLE STYLES</td>
<td>Yes</td>
<td>Limited to Typo-Chains</td>
<td>No</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>FRESS</td>
<td>REDIT/RUNOFF</td>
<td>TEXT</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------</td>
<td>--------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
<td>ON-LINE</td>
<td>ON-LINE</td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>KEYED INPUT</td>
<td>TTY</td>
<td>TTY</td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>CRT DISPLAY</td>
<td>PRINTED</td>
<td>PRINTED</td>
</tr>
<tr>
<td></td>
<td>PHOTO-COMPOSITION</td>
<td>TELETYPED</td>
<td>TELETYPED</td>
</tr>
<tr>
<td>EDITING FEATURES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD/DELETE CHARACTERS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE WORDS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ADD/DELETE PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER WORDS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TRANSFER PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>FORMAT FEATURES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEXT COLUMNS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PAGE/TEXT WIDTH</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PAGE/TEXT DEPTH</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NEW PAGE/NEW COLUMN</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes/No</td>
</tr>
<tr>
<td>RUNNING HEADS AND FEET</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>RIGHT-HAND PAGES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>&quot;AS-IS TEXT&quot;</td>
<td>Yes</td>
<td>Not Available</td>
<td>Yes</td>
</tr>
<tr>
<td>CENTERED TEXT</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LINE-SPACING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>BLANK LINES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PARAGRAPHS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>PRESS</td>
<td>REDIT/RUNOFF</td>
<td>TEXT</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------</td>
<td>--------------</td>
<td>------</td>
</tr>
<tr>
<td>FORMAT FEATURES (cont.)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HANGING INDENTIONS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>COLUMN JUSTIFICATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LINE JUSTIFICATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TABULAR TEXT</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HYPHENATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>FOOTNOTES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HEADINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>KEEPS</td>
<td>Yes</td>
<td>Not Available</td>
<td>Yes</td>
</tr>
<tr>
<td>TABLES</td>
<td>Yes</td>
<td>Not Available</td>
<td>Yes</td>
</tr>
<tr>
<td>HORIZONTAL LINES</td>
<td>Yes</td>
<td>Not Available</td>
<td>Yes</td>
</tr>
<tr>
<td>VERTICAL LINES</td>
<td>Yes</td>
<td>Not Available</td>
<td>Yes</td>
</tr>
<tr>
<td>CAPITALIZATION</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>UNDERSCORING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>USER-DEFINED CODES</td>
<td>Yes</td>
<td>Not Available</td>
<td>No</td>
</tr>
<tr>
<td>SUPPLEMENTAL LISTINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>MULTIPLE PRINTINGS</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>REVISION BARS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>RENUMBERING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>REPAGING</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>VARIABLE STYLES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NAME OF SYSTEM</td>
<td>NLS, TNLS, DNLS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ON-LINE/OFF-LINE</td>
<td>ON-LINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPUT FORM(S)</td>
<td>TTY, CRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTPUT FORM(S)</td>
<td>Typed, CRT, printed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDITING FEATURES</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD/DELETE CHARACTERS</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD/DELETE WORDS</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD/DELETE LINES</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD/DELETE PARAGRAPHS</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD/DELETE PAGES</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANSFER WORDS</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANSFER PARAGRAPHS</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANSFER PAGES</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORMATTING FEATURES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEXT COLUMNS</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAGE/TEXT WIDTH</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAGE/TEXT DEPTH</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NEW PAGE/NEW COLUMN</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RUNNING HEADS AND FEET</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIGHT-HAND PAGES</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;AS-IS TEXT&quot;</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CENTERED TEXT</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINE-SPACING</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLANK LINES</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PARAGRAPHS</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INDENTIONS</td>
<td>YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name of System</td>
<td>NLS, TNLS, DNLS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Format Features (cont.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hanging Indentions</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Column Justification</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line Justification</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tabular Text</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hyphenation</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Footnotes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Headings</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Keeps</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tables</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horizontal Lines</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vertical Lines</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Capitalization</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Underlining</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User-Defined Codes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supplemental Listings</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multiple Printings</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Revision Bars</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Renumbering</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repaging</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variable Styles</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX F

PROGRAM EDITORS
AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Working Paper No. 3
August 25, 1972
Program Editors

By
Ralph F. Planthold

Texas A&M University
Texas Engineering Experiment Station
ABSTRACT

This report defines a program editor, differentiates between a program editor and free-form text editor, and delineates the two major categories of program editors: on-line and off-line. It shows that the characteristics of on-line program editors may differ because of the particular type of terminal involved.

The report lists the information sources tapped and responses received. Next is a comparison of six on-line editors and a comparison of eleven off-line editors, both in tabular form.

Finally, there is a list of criteria for an ideal program editor and a nomination of the on-line editors and the off-line editors which, in the author's opinion, come closest to satisfying the listed criteria.
Table of Contents

Introduction ................................................. F-3
Background .................................................. F-5
State of the Art ............................................. F-6
  On-Line Program Editors ................................. F-8
  Off-Line Program Editors ............................... F-10
Conclusions: The "Ideal" Program Editor ............... F-13
Bibliography ................................................ F-14
INTRODUCTION

A program editor is basically a software package which allows one to modify and document one or more source program modules without the danger of hand-carrying (and possibly dropping or losing) sizable decks of cards and without the attendant need for reading in and compiling the entire source module(s) every time a change is incorporated. As such, a program editor is a rather specialized version of a text editor (a software package for the manipulation of free-form textual material).

A program editor differs from a text editor in the way its material is stored and displayed. Storage in card image or printer line image format is generally quite adequate for performing the "in place" modification - substitution of one opcode, operand, address, label, etc., for another - of computer source programs. Free-form text editors, because of their more powerful capabilities of arbitrary placement and replacement of arbitrarily sized character strings, require a flexible storage structure, one capable of a noticeable amount of dynamic expansion and contraction. For this the unit of storage must be at least a "super-line" of several hundred characters, and may even be a paragraph or a whole page.

For the output of a program editor, only an upper case character set is needed; and the text is printed as it was stored, line by line. A text editor is limited if it does not have at least upper and lower case in its character set, because the fairly complex typesetting codes it employs would lose a great deal of their impact in such an environment.
Program editors fall into one of two categories: on-line or off-line, depending on their mode of operation. Off-line editors are automatically self-documenting, providing a hard-copy listing of all changes made during the current run and, optionally, a wide variety of reports covering such areas as a program evolution history and a current file status summary. On-line editors provide no such automated documentation, relying on the individual programmer or another documentation system to perform this function.

Off-line editors permit the creation of entire source modules only, whereas on-line editors allow the user to create new routines as he thinks them up, one statement at a time. Since off-line editors are governed by control cards supplied by the user, an error-handling technique is provided. In most cases, if an error is encountered on a control card, all modifications of the source module to which the control card applies are ignored. One noteworthy exception to this is "Simple", an off-line program editor from Computer Services which terminates the program upon finding any errors in its control cards. On-line editors can immediately alert the user to any errors in control information. Upon correction, the user may then proceed with the editing of the program.

Various on-line editors exhibit differences due to the type of terminal they employ. Those communicating with the user via a teletype/typewriter (TTY) terminal show marked similarities to off-line editors in the method of specifying operations and in the manner in which the text is presented. For this reason, any reference to on-line editors
in the comparison of these methods and manners in the two following paragraphs will apply only to those using cathode-ray tube (CRT) units as terminals.

For an off-line editor, specifying what is to be done and where it is to be done requires a command name or abbreviation together with a location denoted by a line number and/or context string. An on-line (CRT) editor typically requires "pointing" to the target text area via a light-pen, a keyboard-controlled cursor, etc., then modifying the text by (perhaps depressing a function key and) typing the new text right over the old.

With an off-line editor, the user is forced to work from some printed copy, making changes thereon and transcribing those changes, resulting in duplication of effort. Because an on-line (CRT) editor simultaneously displays many lines of text with virtually no time lag, the user can think out and implement desired changes in one operation instead of two.

BACKGROUND

Over the past three months, the following sources have been searched for information on the subject of program editors: Auerbach Computer Technology Software Reports, Journal of the Association for Computing Machinery (ACM), ACM Computing Surveys, ACM Computing Review, Communications of the ACM, U.S. Government Research and Development Reports, Proceedings of the AFIPS Fall and Spring Joint Computer Conferences, the Computer Journal, the IBM S/360 OS Utilities manual, the Univac 1100 Series Operating System Programmer Reference, and the Control Data 6000 Series Computer Systems KRONOS Text Editor Reference Manual.
The Auerbach Reports carried information about a large number of data manipulation packages, four of which were found to be worthwhile as (off-line) program editors. Two of the four originating firms have sent further literature in response to our request for same; one firm remains untraceable. The ACM Computing Surveys carried information on seven (on-line) program editors. Two operated on certain source languages only (one on JOVIAL, the other on PL/I and Gedanken) and have therefore been omitted*; requests for additional information on the rest remain as yet unanswered.

In addition, letters have been sent to thirteen manufacturers of source program maintenance packages listed in the 1971 edition of the Datamation Industry Directory. Four positive replies have been received as of this date; one other firm has gone out of business.

STATE OF THE ART

Summarized in the following tables are the characteristics of six on-line and eleven off-line program editors. The on-line editors treated are: Conversational Context-Directed Editor (CMS), developed at the IBM Cambridge Scientific Center; WYLBUR, developed at the Stanford Computation Center; Quick Editor (QED), developed by the University of California at Berkeley and revised extensively for commercial use by Com-Share, Inc. - the only editor of either category to maintain additional files of editing changes for testing slightly different versions of a program without modifying or duplicating the original program; Text Editor and Corrector (TECO), developed by the Massachusetts Institute of Technology and Project

* All program editors treated in this report will operate on any computer source program, written in any source language.
MAC; TVEDIT, developed by the Stanford Computation Center; and EDIT, developed by Control Data Corporation for its KRONOS Time-sharing System.

The off-line editors treated are: LIBRARIAN, by Applied Data Research, Inc.; SIMPLE, by Computer Services; Card Library On Tape (CLOT), by International Telecontrol Corporation; PROGRAM/MANAGE, by Management Systems Corporation; Card File Maintenance System (CFMS), by Sigma Software Company; PANVALET, by Pansophic Systems, Incorporated; PLUS D/A, by the Cullinane Corporation; the Source Program Library System (SPLIS-II), by the Webster Computer Corporation; IEBUPDAT and IEBUPDTE, two IBM System/360 Operating System Utilities programs; and the ED Processor, a text editor which is part of the Univac EXEC-8 Monitor.

Within the tables, an entry of "Not Given" means: "Has or could have this attribute, but no conclusive information given." An entry of "Not Applicable" means: "Could in no way have this attribute." For insertion of a character string or replacement of a shorter one by a longer one, if the record length is exceeded, an entry of "Truncation" means: "Any excess is ignored and lost," while an entry of "Overflow" means: "Any excess causes no insertion/replacement to be made, and a warning to be issued."
### ON-LINE PROGRAM EDITORS

<table>
<thead>
<tr>
<th></th>
<th>CMS</th>
<th>WYLBUR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Computer Configuration</strong></td>
<td>IBM S/360/67 under CP/CMS Operating System</td>
<td>IBM S/360/67 under OS/NET, 1 disk drive</td>
</tr>
<tr>
<td><strong>Man-Machine Interface</strong></td>
<td>IBM 2741 TTY</td>
<td>IBM 2741 TTY</td>
</tr>
<tr>
<td><strong>Storage Requirements (bytes)</strong></td>
<td>Not Given</td>
<td>Not Given</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td>Not Given</td>
<td>Not Given</td>
</tr>
<tr>
<td><strong>Language Written In</strong></td>
<td>Not Given</td>
<td>IBM S/360 BAL</td>
</tr>
<tr>
<td><strong>Target Record Type</strong></td>
<td>Fixed Length Line</td>
<td>Variable Length Line</td>
</tr>
<tr>
<td><strong>Target Record Size (characters)</strong></td>
<td>80</td>
<td>1-133 (Default is 72)</td>
</tr>
<tr>
<td><strong>Target Record Processing Fashion</strong></td>
<td>Pages Sequential, Random within Page</td>
<td>Random</td>
</tr>
<tr>
<td><strong>Target Text Pinpoint Location Requirements</strong></td>
<td>Relative Line No.</td>
<td>Absolute Line No.</td>
</tr>
<tr>
<td><strong>Fixed Position (Label) Scan Capability</strong></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Arbitrary Position (Context String) Scan Capability</strong></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Record-Oriented Commands</strong></td>
<td>Single Record, Multiple Records</td>
<td>Insert, Delete, Replace, Insert, Delete, Replace</td>
</tr>
<tr>
<td><strong>String-Oriented Commands</strong></td>
<td>Single Record, Limited Record Range</td>
<td>Insert, Delete, Replace (Truncation), Replace (Truncation)</td>
</tr>
<tr>
<td></td>
<td>Entire Record Range (Universal)</td>
<td>Replace (Truncation)</td>
</tr>
<tr>
<td><strong>Temporary Edit Capability for Testing</strong></td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Machine-Readable Output</strong></td>
<td>Updated Master File Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Job Stream JCL Retrievable from System or Library</td>
<td>Not Applicable No</td>
</tr>
<tr>
<td></td>
<td>Job Stream Execution without Operator Intervention</td>
<td>Not Applicable Yes</td>
</tr>
<tr>
<td></td>
<td>Selected Modules in Punched Deck Form</td>
<td>No Yes</td>
</tr>
<tr>
<td><strong>Hard-Copy Documentation Output</strong></td>
<td>Current Run Permanent Edit Listing Not Applicable</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Current Run Temporary Edit Listing</td>
<td>Not Applicable Not Applicable</td>
</tr>
<tr>
<td></td>
<td>Updated Module Listing</td>
<td>Not Applicable Yes</td>
</tr>
<tr>
<td></td>
<td>Program Evolution History</td>
<td>Not Applicable No</td>
</tr>
<tr>
<td></td>
<td>Current File Status Summary</td>
<td>Not Applicable Yes</td>
</tr>
<tr>
<td><strong>Security Controls</strong></td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td><strong>Additional Capabilities</strong></td>
<td>Tabbing Capability</td>
<td>Tabbing Capability</td>
</tr>
<tr>
<td>OED</td>
<td>TECO</td>
<td>TVEDIT</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Not Given, 1 disk drive</td>
<td>DEC PDP-10, 1 disk drive</td>
<td>IBM S/360/67, 1 disk drive</td>
</tr>
<tr>
<td>TTY</td>
<td>TTY</td>
<td>CRT</td>
</tr>
<tr>
<td>Not Given</td>
<td>4K</td>
<td>Not Given</td>
</tr>
<tr>
<td>Not Given</td>
<td>Not Given</td>
<td>Not Given</td>
</tr>
<tr>
<td>Not Given</td>
<td>Not Given</td>
<td>Not Given</td>
</tr>
<tr>
<td>Variable Length</td>
<td>Variable Length Page</td>
<td>Variable Length Page</td>
</tr>
<tr>
<td>Super-Line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-500</td>
<td>Unlimited</td>
<td>1-128 per line</td>
</tr>
<tr>
<td>Random</td>
<td>Pages Sequential, Random within Page</td>
<td>Random</td>
</tr>
<tr>
<td>Relative Super-Line</td>
<td>Relative Character or Line Displacement</td>
<td>Absolute Page No., followed by Visual Scan</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Insert, Delete, Replace</td>
<td>Not Applicable</td>
<td>Delete</td>
</tr>
<tr>
<td>Insert, Delete, Replace</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Replace (Truncation)</td>
<td>Insert, Delete (1 character at a time)</td>
<td>Insert, Delete, Replace (Truncation)</td>
</tr>
<tr>
<td>Replace (Truncation)</td>
<td>Insert, Delete (1 character at a time)</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Replace (Truncation)</td>
<td>Insert, Delete (1 character at a time)</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Yes</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>None</td>
<td>Arithmetic, Tabbing Capability</td>
<td>Command Repetition Factor</td>
</tr>
</tbody>
</table>
# OFF-LINE PROGRAM EDITORS

<table>
<thead>
<tr>
<th><strong>LIBRARIAN</strong></th>
<th><strong>SIMPLE</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Computer Configuration</strong></td>
<td>IBM S/360 under OS or DOS, 1 disk drive</td>
</tr>
<tr>
<td><strong>Man-Machine Interface</strong></td>
<td>Card Reader/Punch, Printer</td>
</tr>
<tr>
<td><strong>Storage Requirements (bytes)</strong></td>
<td>50K</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td>$3600 for 1st 3 years, then $500 per year</td>
</tr>
<tr>
<td><strong>Language Written In</strong></td>
<td>IBM S/360 BAL</td>
</tr>
<tr>
<td><strong>Target Record Type</strong></td>
<td>Fixed Length Card Image</td>
</tr>
<tr>
<td><strong>Target Record Size (characters)</strong></td>
<td>80</td>
</tr>
<tr>
<td><strong>Target Record Processing Fashion</strong></td>
<td>Sequential</td>
</tr>
<tr>
<td><strong>Target Text Pinpoint Location Requirements</strong></td>
<td>Absolute Card Sequence No.</td>
</tr>
<tr>
<td><strong>Fixed Position (Label) Scan Capability</strong></td>
<td>No.</td>
</tr>
<tr>
<td><strong>Arbitrary Position (Context String) Scan Capability</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Record-Oriented Commands (handle at a time)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Single Record</strong></td>
<td>Insert, Delete, Replace</td>
</tr>
<tr>
<td><strong>Multiple Records</strong></td>
<td>Insert, Delete, Replace</td>
</tr>
<tr>
<td><strong>String-Oriented Commands (operate within)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Single Record</strong></td>
<td>Insert, Replace (Overflow)</td>
</tr>
<tr>
<td><strong>Limited Record Range</strong></td>
<td>Insert, Replace (Overflow)</td>
</tr>
<tr>
<td><strong>Entire Record Range (Universal)</strong></td>
<td>Insert, Replace (Overflow)</td>
</tr>
<tr>
<td><strong>Temporary Edit Capability for Testing</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Machine-Readable Output</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Updated Master File</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Job Stream JCL Retrievable from System or Library</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Job Stream Execution without Operator Intervention</strong></td>
<td>Yes (MFT/MVT only)</td>
</tr>
<tr>
<td><strong>Selected Modules in Punched Deck Form</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Hard-Copy Documentation Output</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Current Run Permanent Edit Listing</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Current Run Temporary Edit Listing</strong></td>
<td>No</td>
</tr>
<tr>
<td><strong>Updated Module Listing</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Program Evolution History</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Current File Status Summary</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Security Controls</strong></td>
<td>Unchanging, b: passable, pnm-generated password</td>
</tr>
<tr>
<td><strong>Additional Capabilities</strong></td>
<td>COHOL (any level), Syntax checker (12K bytes); Resequencing; Module Manipulation</td>
</tr>
</tbody>
</table>
## OFF-LINE PROGRAM EDITORS

<table>
<thead>
<tr>
<th>CLOT</th>
<th>PROGRAM/MANAGE</th>
<th>CFMS</th>
<th>PANVALET</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM S/360/25 under OS or DOS, 2 tape drives</td>
<td>IBM S/360/30 under OS or DOS, 2 tape drives</td>
<td>IBM S/360, 370 under DOS, 3 tape drives</td>
<td>IBM S/360/22, 370/135 OS or DOS, 1 disk drive</td>
</tr>
<tr>
<td>Card Reader, Printer</td>
<td>Card Reader/Punch, Printer</td>
<td>Card Reader/Punch, Printer</td>
<td>Card Reader/Punch, Printer</td>
</tr>
<tr>
<td>Not Given</td>
<td>32K</td>
<td>32K</td>
<td>19K</td>
</tr>
<tr>
<td>$950 plus optional $100 per year maintenance</td>
<td>$24.55 each authorization</td>
<td>$900 plus optional $423 installation charge</td>
<td>$3700 plus optional $600 per year maint. after 3</td>
</tr>
<tr>
<td>COBOL</td>
<td>COBOL</td>
<td>COBOL</td>
<td>Not Given</td>
</tr>
<tr>
<td>Fixed Length Card Image</td>
<td>Fixed Length Card Image</td>
<td>Fixed Length Card Image</td>
<td>Fixed Length Card Image</td>
</tr>
<tr>
<td>80</td>
<td>80</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>Random</td>
<td>Not Given</td>
<td>Not Given</td>
<td>Sequential</td>
</tr>
<tr>
<td>Absolute Card Sequence No.</td>
<td>Absolute Card Sequence No.</td>
<td>Not Given</td>
<td>Absolute Card Sequence No.</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Insert, Delete</td>
<td>Insert, Delete, Replace</td>
<td>Insert, Delete</td>
<td>Insert, Delete, Replace</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Scan and Print only</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Scan and Print only</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Scan and Print only</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
<td>Up to 3 levels of security control codes</td>
</tr>
<tr>
<td>Resequencing</td>
<td>Resequencing</td>
<td>None</td>
<td>Module Manipulation; Formatting; Resequencing; Cross Referencing;</td>
</tr>
<tr>
<td>PLUS D/A</td>
<td>SPLIS-II</td>
<td>IERUFDAT and IERUFDTE</td>
<td>ED</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>IBM S/360, 370 under OS or DOS, 1 disk drive</td>
<td>IBM S/360/30, 370 under OS, 1 disk drive</td>
<td>IBM S/360, 370 under OS, 1 disk drive</td>
<td>UNIVAC 1100 series under EXEC-3, 1 disk drive</td>
</tr>
<tr>
<td>Card Reader/Punch, Printer</td>
<td>Card Reader/Punch, Printer</td>
<td>Card Reader/Punch, Printer</td>
<td>Card Reader/Punch, Printer</td>
</tr>
<tr>
<td>$3000 for last 3 years, then $500 per year</td>
<td>$2500 1 install, $2000 2nd, $5000 each additional</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>IBM S/360 BAL</td>
<td>COROL</td>
<td>Not Given</td>
<td>Not Given</td>
</tr>
<tr>
<td>Fixed Length Card Image</td>
<td>Fixed Length Card Image</td>
<td>Fixed Length Card Image</td>
<td>Fixed Length Line</td>
</tr>
<tr>
<td>80</td>
<td>80</td>
<td>80</td>
<td>Not Given (Default 80)</td>
</tr>
<tr>
<td>Sequential</td>
<td>Random</td>
<td>Random</td>
<td>Random</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Insert, Delete, Replace</td>
<td>Insert, Delete</td>
<td>Insert, Delete, Replace</td>
<td>Insert, Delete Replace</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Replace</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Not Given</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>No</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>No</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Installation-changeable scrambler character set</td>
<td>Password</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Resequencing</td>
<td>Module Manipulation; Resequencing</td>
<td>Resequencing; Change Data Set Organization (IERUFDTE only)</td>
<td>Tabbing Capability; Module Manipulation</td>
</tr>
</tbody>
</table>
CONCLUSIONS: THE "IDEAL" PROGRAM EDITOR

While user requirements differ widely, and no one program editor could ever satisfy all potential users, certain desirable characteristics do stand out above the rest. The "ideal" program editor should be capable of:

1. Easy mastery by programmer personnel (a minimal amount of memorization and time required for entry of edit command codes and operands)
2. High-speed response
3. Processing records out of sequence
4. Universal replacement of character strings
5. Inserting a character string or replacing a shorter with a longer without fear of truncation
6. Maintaining files for edit operations, applicable at will to a master file
7. Creating a job stream for compilation/execution of changed modules, with JCL capable of being accepted from the user and/or operating system and/or source program library
8. Producing hard-copy documentation of all permanent and temporary changes made during the current run and a history of all changes ever made to a given module
9. Displaying many lines of text simultaneously, with the ability to display any portion of the entire text very rapidly (requires an on-line editor with a CRT terminal)

Of the program editors treated in this report, the two most nearly satisfying these ideal requirements for both categories are:

1. On-Line - TVEDIT and WYLBUR
2. Off-Line - LIBRARIAN and PANVALET
BIBLIOGRAPHY


APPENDIX G

DATA DESCRIPTION IN FORTRAN, COBOL AND PL/1
AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Working Paper No. 6
July 14, 1972
Data Description in Programming Languages
(FORTRAN, COBOL, PL/I)

by
Darrell Ward

Texas A&M University
Texas Engineering Experiment Station
6-1-72
ABSTRACT

The various forms of data encountered in FORTRAN, COBOL, and PL/I, are described at the local and global levels, along with ambiguities that arise within each language.

Suggestion for a syntax directed general purpose algorithm is also discussed.
# TABLE OF CONTENT

I. Abstract ................................. G-1

II. FORTRAN
   A. Local Level - Detailed .................. G-3
   B. Local Level - Detailed Suppressed ..... G-5
   C. Global Level ........................... G-7
   D. Ambiguities ............................ G-8

III. COBOL
   A. Local Level - Detailed .................. G-9
   B. Local Level - Detailed Suppressed ..... G-11
   C. Global Level ........................... G-12
   D. Ambiguities ............................ G-13

IV. PL/I
   A. Local Level - Detailed .................. G-14
   B. Local Level - Detailed Suppressed ..... G-16
   C. Global Level ........................... G-17
   D. Ambiguities ............................ G-18

V. Conclusions .............................. G-19
FORTRAN

Local Level - Detailed

The IBM FORTRAN H compiler presents a local map of names encountered during a compilation. All names used as variables, statement functions, subprograms and internal functions.

All names are accompanied by descriptive tags. A brief description of each tag follows:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The variable was used as an argument.</td>
</tr>
<tr>
<td>F</td>
<td>The variable has appeared on the right of an equal sign.</td>
</tr>
<tr>
<td>S</td>
<td>The variable has appeared to the left of an equal sign.</td>
</tr>
<tr>
<td>C</td>
<td>The variable is in COMMON.</td>
</tr>
<tr>
<td>E</td>
<td>The variable has appeared in an EQUIVALENCE statement.</td>
</tr>
<tr>
<td>IF</td>
<td>Indicates an internal function.</td>
</tr>
<tr>
<td>NR</td>
<td>The variable has not been referred to.</td>
</tr>
<tr>
<td>SF</td>
<td>Arithmetic statement functions.</td>
</tr>
<tr>
<td>XF</td>
<td>A subprogram.</td>
</tr>
<tr>
<td>XR</td>
<td>Variables, arrays or subprograms that are referenced by name.</td>
</tr>
</tbody>
</table>

The FORTRAN (H) compiler also produces a relative address for those names that are used internally. All functions and subprograms are assigned a relative address of 000000.

The type and length of each variable is also displayed as part
of the MAP option. For example, a double precision real variable would have a TYPE of R*8.

A description of COMMON blocks is also given. The name of each block is presented along with the length of the block. All variables in the block are described in the same manner as presented under the MAP option, with the exception of an identifying tag.

Following this data layout is a map of any equivalence made for the block. This is in the form of the variable name and its offset from the beginning of the block.
Local Level - Detail Suppressed

There is currently no method for suppressing detail. The FORTRAN (G) compiler does separate the scaler variables from the array variables. It seems a combination of this feature along with the COMMON way presented by the FORTRAN (H) compiler would be a good approach to a local level description of the data.

A restriction of the above to arrays only would also appear to be an approach to suppression of detail at the local level. The data layout for all arrays and any equivalences on an array would be quite functional at the local level. This could be accomplished by utilizing the source code and generating the data layout from all specification statements which allow dimensioning. Equivalences could be resolved before the final output of the layout.

Some type of abbreviated notation would be applicable in the presentation of the array structures. For example, the presentation of a 2-dimensional array with subscript limits of 5 and 10 respectively might be presented as follows

\[
\begin{align*}
A(1, 1) & \ldots A(1, 10) \\
& \ldots \\
& \ldots \\
& \ldots \\
A(5, 1) & \ldots A(5, 10)
\end{align*}
\]
A notation for describing the storage method used might also be considered. Consider the following:

\[
\begin{array}{c}
A(1, 1) \ldots A(1, 10) \\
\vdots \\
\vdots \\
\vdots \\
A(5, 1) & A(5, 10) \\
* & * \\
* & * \\
* & *
\end{array}
\]

1st 5 locations in memory for this array
last 5 locations in memory for this array
Global Level

There is currently no procedure for illustrating global data structures. A reasonable approach to this problem would require an analysis and collection of information in several categories.

The COMMON statement and any equivalences affecting COMMON must be considered for a global layout. The actual parameters in any FUNCTION or SUBROUTINE call must be analyzed since they are essentially global variables. An analysis of formal parameters in a SUBROUTINE or FUNCTION statement require analysis because they may certainly be representing a data structure at the global level. Actual parameters may also have an interaction with other variables via an EQUIVALENCE statement, thus the EQUIVALENCE statement must be inspected closely. Formal parameters may not be included in an EQUIVALENCE statement, hence we can disregard it in this instance.

The collection of the above information for subsequent processing is necessary for any global layout of data. Of course, it should be clear that relationships among the main program and subprograms will control the layout of any data relationships that exists across program boundaries.

It seems that a sophisticated system would be required to handle any data layout on the global level. Generally, the above problems can be applied to other high level programming languages such as COBOL and PL/I. Thus, a system should be designed so that it could be table driven and applicable to any of the mentioned languages. Table driven would essentially mean to supply to syntax and semantics of the few statements in each language which affects global layout.
Ambiguities

The EQUIVALENCE statement causes the sharing of locations in FORTRAN. This allows the partitioning and extension of one data structure by possibly several other data structures.

This presents problems at both the local and global level since COMMON variables and actual parameters may appear in the EQUIVALENCE list.
COBOL
LOCAL LEVEL-DETAILED

The use of the option DMAP will provide information about names in the COBOL source program. The following will be a description of the information that is output as a result of specifying the DMAP option.

The internal name generated by the compiler is output in order to facilitate the reading of the object code. The level number of the particular name is also generated when applicable. The description FD is used when the name is a file definition name. The source name used in the program is also output hence we have available three descriptions of importance: 1) the internal name, 2) the level number, and 3) the source name.

The remaining descriptors can be classed in the area of usage and location. All data names will have a base and displacement associated for the purpose of describing relative locations within the data structure. DECB and DCB information is presented in the case of file names. The storage assigned to the data name is described in terms of bytes used and also in terms of the type of data associated with that name. For example, the storage definition could be described in terms of characters or fullwords depending on the usage clause for that variable. This information is presented in assembler language terminology.

The usage of names is described in the following manner. For FD entries the access method utilized is identified. All group items are identified as such and all elementary items are described in terms of their USAGE clause.
All data names that redefine other data names are described with the tag R. Any data names for which an OCCURS has been specified is described with the tag O. A Q indicates that the data name is the object or contains the object of the DEPENDING ON option of the OCCURS clause. An M indicates that the format of the records of the file is:

- F = fixed
- V = variable
- U = undefined
- S = spanned
Currently there is no procedure for suppressing the detail at the local level. The approach to pursue would parallel the FORTRAN and PL/I approach.

Two types of information would be informative at the local level in a functional sense. The data names which have OCCURS clauses specified for them are candidates for a data layout. Again, as in FORTRAN and PL/I one must consider the possibility of memory sharing through the use of the REDEFINES clause. The second type of data layout would be created from the FD sections. Hence the record layouts could be described. This is essentially done by the DMAP option yet many other variables are interspersed and the record layouts are not readily discerned in this environment.

The above approach would require an analysis of the DATA division quite rigorously. Again, as in FORTRAN and PL/I, a syntax driver algorithm could possibly be utilized for this purpose.
GLOBAL LEVEL

The conventions used in COBOL for subroutine linkages are quite similar to that used in FORTRAN. The key to a global analysis of COBOL would be the detection of the USING clause in the CALL statement. The isolation of all identifiers that follow and are non-scalar is the next step to be taken in the construction of a data layout.

The utilization of the source code as input and the output of the variables that are global type variables is the required task in such an environment. The consideration of the DEFINED clause is quite necessary to resolve or analyze any sharing of locations by variables.
AMBIGUITIES

The DEFINED statement is the COBOL statement that causes the sharing of locations in COBOL. The problem requires analysis in order to adequately describe both local and global layouts.
The IBM PL/I compiler (F) produces a detailed description of all identifiers that are used in a PL/I program. The IBM compiler will be used as the focal point in the description of the information which can be obtained about identifiers. To activate a detailed printout of the attributes of identifiers, the user must utilize the compiler option ATR. The following description demonstrates the attributes that are presented at the detailed level.

All arithmetic variables have a set of attributes that may include the following:

a. BINARY|DECIMAL
b. FIXED|FLOAT (SINGLE|DOUBLE)
c. PICTURE
d. INITIAL
e. DEFINED
f. ALIGNED|UNALIGN

g. dimension information
h. precision
i. STATIC|AUTOMATIC
j. INTERNAL|EXTERNAL

String variables are described with many of the same attributes as arithmetic variables. For example i) represents the storage class and j) represents the scope of the variable or identifier. These attributes are present for all types of identifiers. The following is a list of
attributes for string variables:

a. BIT|CHARACTER
b. PICTURE
c. INITIAL
d. DEFINED
e. ALIGNED|UNALIGNED
f. precision
g. dimension information
h. storage class
i. scope

One can easily see that many attributes are available via the ATR option. These are presented in alphabetical order with the declaration statement # associated with the identifier.

In addition to the above attributes, all elementary items of a structure are described. All major and minor structure names are correctly identified in the listing.

Other fringe type attributes are also available. For example, ENTRY, RETURNS, GENERIC, BUILTIN, etc. are descriptions of variables. Another area which may be considered is that of file descriptions. The ATR option permits the description of files. For example, such attributes as INPUT, OUTPUT, UPDATE, SEQUENTIAL, DIRECT are available.

The above demonstrates that a very detailed description of identifiers is available as output of the PL/I compiler. This is restricted to a single compilation and hence it applies essentially on the local level.
Local Level - Detail Suppressed

In addition to the attributes of the identifiers, the compiler produces an aggregate length table which is the length in bytes of all major structures and non structured arrays. This table is produced by the IBM compiler when the ATR option is used. This presents a very functional presentation of the overall structure of the major components of the identifiers.
Global Level

There are no present tools available to present a data layout over separate compilations. There are possibly two approaches to such a global description of data structures.

The first approach would necessitate a program to analyze the declaration statements of the source language. All identifiers which have the attributes of EXTERNAL and STATIC are candidates for a global data layout.

The second approach would require an analyzation and interpretation of the ESD which the compiler produces. All external symbols which are placed in a COMMON control section should be considered for a global presentation. In addition to these identifiers, under certain conditions control sections of the type SD will be created for global identifiers. This control section is not a COMMON control section but is accessible by other compilations thus, in effect, it is common to other procedures.
Ambiguities

The possibility of misinterpreting or misrepresenting the data layout exists because of the DEFINED attribute. The DEFINED attribute essentially causes the sharing of memory by several identifiers. This presents a problem in the presentation of the data layout since we may now have several arrays and structures utilizing the same memory locations.

This problem is also complicated by allowing character arrays to have other names specified for part of the character strings. This is implemented by the use of the POSITION attribute in conjunction with the DEFINED attribute.
CONCLUSIONS

Clearly, if one considers the source languages of the three languages at a global level, it becomes apparent that all have similar methods for transmission of variables, redefinition of variables and declaration of arrays and structures. PL/I is the encompassing language in the sense that it includes the features of both COBOL and FORTRAN.

Since the features, which must be considered, are quite similar the approach of a syntax driven collection algorithm might be quite feasible. The fact that considerably enhances such an approach is that fact that such a limited analysis of source statements is necessary. The main considerations are to be given to specification statements and calling - answering sequences. The types of specification statements will include array specification, structure specification and redefinition statements.

The above approach could be applied for a global layout and a subset could be utilized at the local level for a functional picture of the data structures. The local level detailed layouts are handled adequately by the compilers and the emphasis should be placed on the detail suppressed mode and global description.
APPENDIX H

DOCUMENTATION STANDARDS
AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Working Paper No.4
July 14, 1972
Documentation Standards

By
Charles Schroeder

Texas A&M University
Texas Engineering Experiment Station
ABSTRACT

This paper investigates the state of the art of documentation standards. The few standards that do exist are reviewed and standards at various levels are suggested where possible. It was found that existing automatic documentation packages fail to meet any sort of standards.

Systems Documentation and Program Documentation are then discussed with an overview of what is generally acceptable and what might be incorporated into a set of standards.

Existing flowchart and decision table standards are presented.
TABLE OF CONTENTS

INTRODUCTION ............................................. H-3

SYSTEMS DOCUMENTATION STANDARDS

   State of the Art ........................................ H-5
   Recommended (suggested) Standards ..................... H-7

PROGRAM DOCUMENTATION STANDARDS

   State of the Art ........................................ H-12
   Recommended (suggested) Standards ..................... H-13

NATIONAL STANDARDS

   Flowcharts ............................................. H-27
   Decision Tables ....................................... H-27

DOCUMENTATION PROCEDURES CURRENTLY IN USE ............. H-37

AN EXAMPLE OF DOCUMENTATION .............................. H-39

SUMMARY ................................................ H-41

BIBLIOGRAPHY ............................................ H-42
INTRODUCTION

Basically, Systems Documentation may be categorized into developmental documentation and control documentation. Since one of the major problems constantly discussed in almost any technical or administrative setting is the problem of poor communication, the above documentation types are intended to lessen this problem. Developmental documentation is descriptive of the system itself, i.e., performance characteristics, tools and materials. It is therefore the means of communication about the system. Control documentation is concerned with communicating information about resources used to develop the system and involves project development organization, personnel, time, materials, and money.

Systems designed are never static. They must change to meet new customer requirements or in order to be compatible with hardware and software modifications. More often than not the designer of the original system is no longer available to make the necessary changes, therefore a clear and complete record of what each system does and how it was developed is essential. Hence, at least some standard must be developed in order to assist the programmer/analyst in preparing good documentation to establish a common language of communication.

Program documentation deals more specifically with producing what might be called a Job Documentation Package or a Program Manual. In general, the aim in writing programs is to use the computer to solve problems which are too tedious, repetitive or uneconomical for manual methods. From some
problem definition and solution specification, detailed program logic is designed, and the program coded and tested. The product is thus a proved and documented (hopefully) program ready for operation, probably initially operated within a test environment.

There is a wide range of application differences, however within such a definition of a program, from a scientific problem-solving application which can perhaps be defined and solved by a single person to a detailed systems specification requiring many designers and programmers.

The former may serve a one-time purpose and specific records of input and file records may not be necessary. In this case rather little documentation may be required.

The case of the detailed systems specification made up of a number of program specifications describing some complex data processing system is much more complex. Here, many "in-house" programmers will be involved and possibly outside contractors will be employed. In this case, it is generally agreed that accurate program documentation is a must.

The real problem in setting standards is to specify how much documentation is required and from whom. No national standards for program documentation exist, but some software packages are commercially available to produce automatic documentation at the program level. Although rather poor in general and far from meeting even the most loosely defined set of standards, they are an attempt at standardization.
In any discussion concerning standards, it is first necessary to lay some ground rules. First of all, what are standards and what has been standardized?

Currently the only set standards for computer software and/or computer software documentation deal specifically with standard flowchart symbols and abbreviations list, and decision table standards. Beyond these items, the word "standards" generally refer to what a particular installation feels is necessary. Very little information has been found concerning standards of any kind, but most reports on any type of standards do agree in some broad areas.

At present, there are no universal documentation standards which are directly applicable to all installations. This is true because of the type and level of complexity of documentation for one location may be inadequate or inappropriate in another environment. Each data processing department must therefore implement a documentation system which suits its own environment. This could be done by adapting a general documentation system to local conditions.

The development of a documentation system for a particular data processing organization must take account of the major influencing factors which include:
(1) management commitment
(2) project characteristics
(3) corporate environment and organizational characteristics, and
(4) technical environment
In any event, with respect to systems documentation, the following documents might be recommended.
(1) A User Request - which is the initial approach for the user for data processing assistance. It contains a brief problem description.
(2) The Systems Proposal - essentially a major report resulting from a systems survey. It may be considered as a feasibility report, including a detailed specification of the recommended approach and project plan.
(3) The Analytical Report - essentially a project plan, supplemented as necessary with evaluation information.
(4) The Design Requirement Statement - which is essentially a specification of requirements in systems terms.
(5) The Systems Summary - is a general description of the complete system or system change.
(7) Transaction (Input) Specifications - This document describes all inputs to the system.
(8) Output Specifications - This document details the systems outputs—why and when produced, contents, formats and recipients.

(9) Segment (Processing) Specification - essentially a statement of the design requirements and general logic for a program.

(10) Systems Test Plan - A permanent record of the testing procedure to prove the system prepared by the systems design function.

(11) Programming Specifications - (which are discussed separately later)

The above list of systems documents are basically those of Gray & London in *Documentation Standards*.²

The present state of the art on the above set of documents follows the trend of filling out appropriate printed forms for each step in the documentation process. These forms are generally suited to the individual needs of the specific installation and seem to follow no real fixed pattern. No mention is made of automatically producing any of these forms or their equivalents. It seems very feasible that a great deal of this documentation could be automatically produced, as is discussed in other reports.

**Suggested Standards for Systems Documentation**

The problem of how much documentation is needed for a particular system is the obvious problem here. In addition to the standards mentioned in discussing the state of the art, minimal standards should include most of the following.
I. Systems Flowcharts - Flowcharts provide a means of identifying programming functions and graphically visualizing the logic and the path of its flow in the solution of a problem. Generally, two levels of flowcharts are recommended. A third level, a MICRO or logic detailed flowchart may be desirable but can be considered as optional.

A. First Level System Flow. This level should be a general system flow, giving an overview of the major processing areas and programming requirements (programs) that will be required for system development. This flow should be in block form to identify the core resident programs and the programming path necessary to accept, process and output data. The general flow should also show the support routines, e.g., overlay, and common routines. The first level flow may require more than one page to identify core resident and support programs or routine. Core resident programs should be grouped within a block to show the processing area to which the program belongs. Systems should be designed so core resident and support routines/programs are developed in modular form.

1. Core resident and support routines developed to aid processing and message switching functions should be identified by name or subject, followed by some distinguishable alphanumeric program name.
2. A flow depicting the hardware configuration should be drawn showing all hardware and peripheral components, and the path of the data flow through the system.

B. Second level MACRO flow. The programming functions identified in the first level system flowchart should be developed into MACRO flow (semi-detailed) diagrams.

1. Some method for identifying each symbol or block of the MACRO flow should be used.

2. The MACRO flowchart expands the program or major processing function of the first level into semi-detailed component blocks or programming symbols. Flowchart pages should be numbered.

3. As the second level flowcharts are reviewed by the lead programmer/analyst, processing errors and omissions should be detected prior to coding.

4. The MACRO flow should depict, in graphic and symbolic form, input/output functions, subroutine, major processing functions and the processing sequence for the coding of a program. It should be in sufficient detail to permit another programmer to develop the source coding.

5. Standard flowcharting symbols should be used.

6. The general flow should be from top to bottom, left to right.

7. Off-page connectors should contain unique tags or
coordinates to which they point, and the flowchart page number. If the off-page connector does not relate to the preceding page, show the page number of the page where the flow was broken, e.g. from page 4 of 12.

8. On-page connectors should contain unique tags or coordinates to which they point.

II. Narrative Documentation


While the system is under development, a general narrative should be written giving a brief history and purpose of the system. This will serve as the composite document for the system and will be developed concurrently with the system. The introductory chapter will give the development history and identify each program/routine and contain a brief statement giving the purpose and function of each program/routine. This first chapter should contain the first level block diagram flows. Subsequent chapters should contain the second level MACRO flowchart and a general narrative for each program.

B. Operator (User) and Test and Acceptance Manuals

Information for the Operator's and the Test and Acceptance Manual, as applicable, can be lifted from the program abstract and other narratives as the system is being developed. This information, along with the Systems Reference Manual, will be developed concurrently with the system.
C. When a system is ready for implementation, system documentation should be submitted to a Documentation Library.

III. "Stand-Alone" Programs or Modifications to existing programs.

A. When a "stand-alone" program is developed or a modification to existing programs takes place, it should be narratively documented.

B. When the program is ready for release, the documentation, as required, should go to some Documentation Library.
PROGRAM DOCUMENTATION STANDARDS

State of the Art

Just as there are many different ways to write a program as there are programmers, there are as many ways to document a program. To determine what program documentation is necessary or sufficient is the purpose of program documentation standards.

The United States Air Force feels that acquiring and maintaining accurate and up-to-date documentation is an essential part of the software production and maintenance process. Well-documented programs are necessary in effective communication of software system ideas and techniques between organizations with both operational and economic benefits. In addition, good documentation

1. allows the lead programmer/analyst to review the efforts of an individual programmer to ensure conformance with system design criteria.
2. assists programmers in testing and debugging.
3. enables programmers to update old programs with minimal difficulties.
4. permits new programmers to learn their systems without having to recode programs to discover what they do.
5. makes available information that can be used in the design of new systems within a given unit and to eliminate duplication of effort wherever possible.
Recommended Standards for Program Documentation

The following is a set of general recommendations which can be adopted, or at least partially so, as the required standards for program documentation. These standards were collected from several sources, each different installation making suggestions with slight variations.

A consensus of the standards reviewed recommends that a program manual, a complete final document of a program, should be prepared. It should contain:

1. a general description of the function, use, and methodology of the program.
2. a description of input, files and output used or produced by the program.
3. flow diagrams showing the logic of the program. (Flow diagrams are discussed later in this report.)
4. a description of instructive output messages, e.g. output on console or printer, etc.
5. coding information, e.g. an assembly listing, memory print, descriptions of matrices or tables used.
6. a test plan.
7. program test and operating instructions.

Operations Documentation should be prepared partially by the system designer and partially by the programmer. This documentation, prepared for both the user and the data processing operations staff, should contain:
1. Program Test Instructions, a document comprised of those instructions which are necessary to guide the computer operator in running a test program, prepared by the programmer.

2. Systems Operating Instructions, a list of processing steps, in execution sequence, defining all operating requirements. It should include:
   a) Summary workflow schedule
   b) Data collection and preparation instructions
   c) Input control instructions
   d) Job assembly instructions
   e) Output review and control instructions

A basic collection of forms for System Operating Instructions would include:
   a) workflow summary
   b) general clerical
   c) data preparation (i.e. keypunching)
   d) auxiliary machine (by machine category)
   e) computer operating (by computer type)

Each operating instruction form should bear the basic identifying information comprised of at least:
   a) systems identification
   b) operation identification (brief title)
c) step number for this operation

d) previous operation step number/next operation step number

e) date

f) originator/authority

Any number of other entries may appear on each form. These may include:

a) responsibility for performing the instructions

b) input: description and source

c) output: description and destination

d) process: summary of functions performed

Each specific type of form will have further detail pertinent to that particular form. These forms can be quite lengthy and, rather than list the necessary information for each, a few examples follow on the next pages.

The forms featured in the above suggested documentation can vary greatly. Several different forms (i.e. printed forms) may be required for any one of them. These forms can be detailed or general, depending upon the particular user's needs. Military Standards\textsuperscript{6} generally require detailed information while smaller systems, with little outside communication necessary, could be documented adequately with less detail.

At present, most program documentation information is recorded by hand. It would be desirable if this documentation could be generated automatically from the program and from previously prepared systems documentation. No present documentation package offers any means for
doing this. However, it would be relatively simple to require the programmer to input any additional undervisible information using a CRT. These forms could be produced on the screen and the programmer would enter the necessary information. This information could then be printed with the program output.

Flowcharts can be produced by some packages now available. Flowcharts are in a later section of this report.
Sample Program Documentation Forms
MACHINE SET UP FORM

PROG NAME ______________________  PROG # ______________________  USER I.D. CODE ______________________

AREA ______________________  PROJECT ______________________  REGION SIZE ______________________

JOB PREQ ______________________  PROGRAMMER ______________________

DIRECT ACCESS REQUIREMENTS:

PERMANENT

<table>
<thead>
<tr>
<th>USER ASSIGNED PACK</th>
<th>DATA SET NAME</th>
<th>DDNAME</th>
<th>SERIAL # / CELL #</th>
<th>BIN #</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TEMPORARY

<table>
<thead>
<tr>
<th>USER ASSIGNED PACK</th>
<th>DATA SET NAME</th>
<th>DDNAME</th>
<th>SERIAL # / CELL #</th>
<th>SPACE =</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TAPE REQUIREMENTS:

#7 TRK, UNITS ______________________  #9 TRK, UNITS ______________________

<table>
<thead>
<tr>
<th>DATA SET NAME</th>
<th>DDNAME</th>
<th>LABEL TYPE</th>
<th>7 OR 9</th>
<th>DEN/ TRK</th>
<th>WRITE IN</th>
<th>RET PD IN DAYS</th>
<th>OUTPUT FROM RUN #</th>
<th>INPUT TO RUN #</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ADDITION □  REPLACEMENT □

(FOR ADDITIONAL INFORMATION ATTACH ANOTHER SHEET)

DATE _____  SECTION _____  PAGE _____
CARD READER REQUIREMENTS:

<table>
<thead>
<tr>
<th>DATA SET NAME</th>
<th>DDNAME</th>
<th>SOURCE</th>
<th>DISPOSITION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CARD PUNCH REQUIREMENTS:

<table>
<thead>
<tr>
<th>DATA SET NAME</th>
<th>DDNAME</th>
<th>POCKET #</th>
<th>DISPOSITION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

PRINTER REQUIREMENTS:

<table>
<thead>
<tr>
<th>DATA SET NAME</th>
<th>DDNAME</th>
<th>PRINT TRAIN</th>
<th>FORM #</th>
<th>SETUP #</th>
<th>LINES PER INCH</th>
<th>BURST</th>
<th>DECOL-LATE</th>
<th>DISPOSITION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ADDITION ☐ REPLACEMENT ☐

DATE _____ SECTION _____ PAGE _____
### DOCUMENTATION APPROVAL FORM

**Check Type of Release:**
- [ ] NEW PROGRAM
- [ ] MAINTENANCE
- [ ] SYSTEM

<table>
<thead>
<tr>
<th>SYSTEM OR JOB NAME</th>
<th>PROGRAM #</th>
<th>PROGRAMMING AREA</th>
<th>DIRECT INQUIRIES TO:</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME OF PROJECT</td>
<td>NAME OF PROGRAMMER</td>
<td>DATE</td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION OF SYSTEM, NEW PROGRAM OR MAINTENANCE:**

<table>
<thead>
<tr>
<th>Program Configuration:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIRECT ACCESS STORAGE</td>
</tr>
<tr>
<td>□ Permanent □ Temporary</td>
</tr>
<tr>
<td>DIRECT ACCESS DATA SETS</td>
</tr>
<tr>
<td>□ Tables □ Core Res. Modules</td>
</tr>
<tr>
<td>□ Disk Resident Modules</td>
</tr>
<tr>
<td>□ Data Sets</td>
</tr>
</tbody>
</table>

**Items Affected By This Release:**
- [ ] Program
- [ ] Documentation
- [ ] Tables or Data Sets
- [ ] External Procedures

**Maintenance Priority:**
- [ ] EMERGENCY - Contact Name: __________ Phone: ________
- [ ] SPECIFIC DATE ________ CYCLE ________
- [ ] WHEN POSSIBLE

**SUBMITTED BY**

---

**Complete This Section For New Program and When Maintenance Affects One of the Following:**

**PROGRAM INFORMATION - CHECK ONE FOR EACH ITEM:**
- [ ] Batch
- [ ] Teleprocessing

**RUN FREQUENCY:**
- [ ] Daily
- [ ] Monthly
- [ ] Annually
- [ ] One Time
- [ ] On Request

**ESTIMATED TIME:**
- [ ] Minutes

**LANGUAGE:**
- [ ] COBOL
- [ ] FORTRAN
- [ ] RPG
- [ ] PL/1
- [ ] S/360 AL

**CORE STORAGE:** Enter Region Size ________ K
(Do Not Include OS)

**EQUIPMENT INFORMATION - ENTER WHERE APPLICABLE:**
- [ ] 360/65
- [ ] 1401
- [ ] Card Reader
- [ ] Card Punch
- [ ] Printer
- [ ] Drum
- [ ] Optical Scanner
- [ ] Data Communications
- [ ] Number of Tape Units
- [ ] Number of Disk Storage Drives
- [ ] Number of Data Cell Drives

---
# APPROVALS (Manager Level or Higher)

<table>
<thead>
<tr>
<th>(Compl. this line for New Program Only)</th>
<th>NAME</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBMITTING OFFICE:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROGRAMMING AREA:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| (Compl. this line for Maintenance Only) | |
| MAINTAINING OFFICE:                     | |
| PROGRAMMING AREA:                       | |

| MANAGER REVIEW | |

<table>
<thead>
<tr>
<th>REASON FOR REJECTION OR COMMENTS:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
Sample Micro-flowchart
<table>
<thead>
<tr>
<th>Process Step No.</th>
<th>Operation (Process) Flowchart</th>
<th>Specification No.</th>
<th>Volumes</th>
<th>Latest Time To This Operation</th>
<th>Latest Time Out From This Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Check receipt of batches from all stores</td>
<td>1.1</td>
<td>17 batches of approx. 90 to 1200 documents</td>
<td>10.00 Tuesday</td>
<td>12.00 Tuesday</td>
</tr>
<tr>
<td>2</td>
<td>Punch/Verify Issue/Receipts</td>
<td>1.2</td>
<td>12000 cards</td>
<td>12.00 Tuesday</td>
<td>17.00 Thursday</td>
</tr>
<tr>
<td>3</td>
<td>Balance Batches</td>
<td>1.3</td>
<td>N/A</td>
<td>17.15 Thursday</td>
<td>19.00 Thursday</td>
</tr>
<tr>
<td>4</td>
<td>Error Check to Source</td>
<td>2.0</td>
<td>Variable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Prepare Monthly Recount Cards</td>
<td>1.4</td>
<td>N/A</td>
<td>08.45 Friday</td>
<td>09.00 Friday</td>
</tr>
<tr>
<td>6</td>
<td>Take-on Program SYM 37</td>
<td>1.5</td>
<td>12000 cards 120 forms</td>
<td>09.00 Friday</td>
<td>11.45 Friday</td>
</tr>
</tbody>
</table>
### Decision Table Form

<table>
<thead>
<tr>
<th>Condition</th>
<th>Table</th>
<th>Operating/Closed</th>
<th>Action</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Sample decision table form**
## Decision Table Form

### System Name: Airline Reservation Example

<table>
<thead>
<tr>
<th>Condition</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 First class requested</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2 Tourist class requested</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>3 First class available</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4 Tourist class available</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>5 Alternate class acceptable</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

### Actions

<table>
<thead>
<tr>
<th>Action</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 Write first class ticket</td>
<td>X</td>
<td>X</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>7 Write tourist class ticket</td>
<td>X</td>
<td>X</td>
<td>I</td>
<td>I</td>
<td>X</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>8 Adjust 1st cl. seat invent.</td>
<td>X</td>
<td>X</td>
<td>I</td>
<td>I</td>
<td>X</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>9 Adjust tourist seat invent.</td>
<td>X</td>
<td>X</td>
<td>I</td>
<td>X</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>10 Suggest another flight</td>
<td>I</td>
<td>I</td>
<td>X</td>
<td>I</td>
<td>I</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>11 Go to next request</td>
<td>X</td>
<td>X</td>
<td>I</td>
<td>I</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

### Frequency

| Frequency | 20 | 8 | 6 | 10 | 40 | 9 | 7 |

Sample limited entry decision table
### Decision Table Form

**System Name:** Airline Seat Reservation  
**System Number:** CX  
**Date:** September 1967  
**Drawn by:** R. Thomas

<table>
<thead>
<tr>
<th>Condition</th>
<th>Rule 1</th>
<th>Rule 2</th>
<th>Rule 3</th>
<th>Rule 4</th>
<th>Rule 5</th>
<th>Rule 6</th>
<th>Rule 7</th>
<th>Rule 8</th>
<th>Rule 9</th>
<th>Rule 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request for</td>
<td>First</td>
<td>First</td>
<td>First</td>
<td>First</td>
<td>Tourist</td>
<td>Tourist</td>
<td>Tourist</td>
<td>Tourist</td>
<td>Tourist</td>
<td>Tourist</td>
</tr>
<tr>
<td>Space available</td>
<td>First</td>
<td>Both</td>
<td>Tourist</td>
<td>Tourist</td>
<td>None</td>
<td>Tourist</td>
<td>Both</td>
<td>First</td>
<td>First</td>
<td>None</td>
</tr>
<tr>
<td>Alternate class</td>
<td>-</td>
<td>-</td>
<td>OK</td>
<td>NG</td>
<td>-</td>
<td>-</td>
<td>OK</td>
<td>NG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reduce seat inventory</td>
<td>First</td>
<td>First</td>
<td>Tourist</td>
<td>-</td>
<td>-</td>
<td>Tourist</td>
<td>Tourist</td>
<td>First</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Write ticket</td>
<td>First</td>
<td>First</td>
<td>Tourist</td>
<td>-</td>
<td>-</td>
<td>Tourist</td>
<td>Tourist</td>
<td>First</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Go to next</td>
<td>Req</td>
<td>Req</td>
<td>Req</td>
<td>Flight</td>
<td>Flight</td>
<td>Req</td>
<td>Req</td>
<td>Req</td>
<td>Flight</td>
<td>Flight</td>
</tr>
</tbody>
</table>

**FREQUENCY**

---

Sample extended entry decision table
Flow Chart Standards

The ANSI Standards define three major groups of flowchart outlines.

1) Basic - this includes specifications of symbols representing four functions considered to be the minimum symbols required for adequately representing data processing action. These functions are input/output, processing, flow direction, and annotation.

2) Specialized - consists of outlines for specifying three distinct groups:
   a. data - carrying media (document, magnetic tape, etc)
   b. peripheral equipment type (on-line storage, manual input, etc.)
   c. selected types of processing action (decision, sort, collate, etc.)

3) Additional - these outlines include symbols for representing origins, terminations and functions.

It appears that flowcharting symbols are one of the few things that are generally accepted as standard. These standards are included on the following pages.

Decision Tables

Some standards have also been set up for use with respect to decision tables. These standards generally adhere to the rules which follow. Examples are also provided.
1. **Process Symbol** used to represent any kind of processing function, or any operation for which no particular symbol is provided.

2. **Decision Symbol** used to represent a decision that determines which of a number of alternative paths is to be followed.

3. **Manual Operation Symbol**

4. **Auxiliary Operation Symbol**

5. **Merge**

6. **Extract**

7. **Collate**

8. **Sort**

Standard symbols for system flowcharts
SYSTEMS FLOWCHART SYMBOLS

9. Manual Input

10. Generalized Input/Output Symbol

11. On-Line Storage Symbol represents the use of any kind of on-line backing, store, i.e., disc, drum or magnetic tape

12. Off-Line Storage Symbol represents the function of storing information off-line, regardless of the medium on which the data is recorded.

13. Document

14. Punched Card

15. Deck of Cards

16. File of Cards: this symbol represents a collection of related punched card records.

17. Punched Tape

18. Magnetic Tape
SYSTERS FLOWCHART SYMBOLS

19. Magnetic Drum

20. Magnetic Disc

21. Core Store

22. Display

23. Communication Link: this symbol represents transfer of information by a telecommunication link.

24. Graph Plotter

25. Connector

26. Comment

27. Flow Indicators
1. **General Operation Symbol**: used for any operation which creates, alters, transfers or erases data, or any other operation for which no specific symbol has been defined in the Standard.

2. **Subroutine (Predefined Process) Symbol**: used when a section of program is considered as a single operation for the purpose of this flowchart.

3. **Generalized Input/Output Symbol**: used where it is desired to stress I/O operations. The symbol is used as an alternative to the specific device symbols when:
   - at the time of flowcharting the actual device to be used has not been decided,
   - the flowchart is drawn as an example, and is not related to any specific I/O function,
   - local standards specify its use.

4. **Magnetic Tape I/O**

5. **Disc I/O**

6. **Drum I/O**

7. **Document I/O**

8. **Punched Card I/O**

9. **Punched Paper Tape I/O**
10. Preparation Symbol: used where it is desired to accentuate an operation that partially or completely determines the selection of a particular exit at given Branch Symbols.

11. Branch Symbol: has one entry line and more than one exit. The symbol contains a description of the test on which the selection of an exit is based. The various possible results of this test are shown against the corresponding exits.

12. Offpage Connector Symbol: used as a linkage between two blocks of logic that are to be found on separate pages of the flowchart. The symbol is only used on the 'exit' page, on the 'entry' page an on-page symbol is used.

13. Onpage Connector Symbol: used as a linkage between two blocks on the same page, when it is not desirable to connect them using a linkage line. The label of the block to which the connection is being made is written inside the symbols.

14. Terminal Symbol: used as the beginning or end of a flowline (e.g., start or end of a program).

15. Annotation Symbol: used to add additional information to a symbol or block of program.

16. Flowlines (Linkage Lines): used to show the flow between blocks of a flowchart. The normal flow is from top to bottom and left to right of the page. The programmer may dispense with the use of the direction arrows when the chart follows the normal flow. They must be used, however, for any portion of the diagram which does not follow the normal flow.
List of Standard Symbols and Abbreviations

+       Plus or positive
−       Minus or negative
±       Plus or minus, positive or negative
×       Multiplied by
÷ or /  Divided by
=       Equals
≠       Does not equal
>       Greater than
<       Less than
≥       Greater than or equal to
≤       Less than or equal to
c(x)    Contents of location X
cf or :  Compare or compared with
         Used within an operational symbol to denote transfer of data
EØF    End of file
EØR    End of reel
EØJ    End of job
#      Reserved for local use
No.     Number
Sample Macro-flowchart
These rules are:

1) Decision tables should be drawn on the form shown at the end of this section. The layout of the elements of the decision table should be:

<table>
<thead>
<tr>
<th>CONDITION STUB</th>
<th>CONDITION ENTRY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION STUB</td>
<td>ACTION ENTRY</td>
</tr>
</tbody>
</table>

The double line separating the stubs from the entries is predrawn. The designer should draw his own double line to separate condition and action areas of the form.

2) Only one table should be drawn on each sheet.

3) Tables must be named at the head of the table. A name of the form "TABLE XX" is preferred, but other naming standards may be specified locally. Following the name, the words OPEN or CLOSED should be written to indicate they type of table.

4) No decision table should be drawn that has:
   a. more than 4 condition variables, if neither dashes nor an ELSE rule are used, or
   b. more than 6 condition variables if dashes and/or an ELSE rule are used;
   c. more than 12 decision rules, and
   d. more than 15 action variables.
5) Blanks must not be left in the condition entries. Dashes should be used to indicate that the value of a condition does not affect a particular action.

6) On the condition entries, Y should be used to indicate the truth and N the falsity of a condition.

7) On the action entries, X should be used to indicate that an action is to be followed and I to indicate that it is to be ignored.

8) Actions must be written in the order in which they are to be executed.

9) Every effort should be made to combine rules within a table which give rise to the same action. It will often be found that the value of one condition is immaterial.

10) Tables must be drawn up in such a way that all rules are true alternatives; rules may be examined in any order but only one rule can satisfy a given set of conditions.

11) The final action entry for each rule must specify where to go next.

12) Where the information is available, it can be of considerable assistance to the programmer if the expected frequency of satisfaction of each rule is indicated on the table.
DOCUMENTATION PROCEDURES CURRENTLY IN USE

The information in this section was selected from material requested from various companies about their respective documentation procedures. The replies received ranged from very brief and indefinite requirements to highly specialized and detailed specifications. In general, the size of the company and the amount of computer use seemed to affect the volume of documentation required. It should be noted that the standards mentioned below are a subset of the documentation standards recommended earlier in this report.

In general, the following items seem to make up some sort of accepted "standard."

1. Abstract - This was required by most of the companies involved. The detail of the abstract did vary a little, but the basic ideas seemed constant. This generally included items such as Program Title, Program Number, Language, Machine Configuration, I/O Description, etc.

2. Source Program Decks and Listings - There were some variations here. In general, source programs were kept on file and one company even suggested maintaining actual card decks if the source program was not on tape. In addition to this, general setup of Input, JCL, and special information about control of the program was also used.

3. Program Flowcharts in general were required as some part of a documented program. The level of these flowcharts varied. One
company specified the use of AUTO-FLOW for program logic flow.

4. Systems Flowcharts were used by some of the companies involved while others made no mention of them.

5. Almost all information received made mention of procedures for updating documentation for revised jobs. This process varied from a detailed set of steps of completely redocumenting the program to merely adding update information to the previous documentation.

6. Details of Data Layout was generally required. This included information about data cards, tapes and other information necessary for processing. Keypunch instruction was also requested at some levels.

7. An Operator's Guide was required by a few of the companies involved, however, this was not required by all.

The above list makes up the bulk of the required documentation. The detail required obviously varied greatly. Some major companies required considerably more than the above mentioned items, while a very small company included required only comments in the program listing.

It was interesting to note that in this latter case, 3 levels of comments were specified, depending upon the number of asterisks appended to the comment.
AN EXAMPLE OF DOCUMENTATION

One of the better examples of automatically produced program documentation which was found was from the USAF. This message switching program included in its computer printed output the following:

(1) A detailed table of contents, which included a list of major headings along with subheadings of everything included in this listing. Also included was notation used for separating sections, sub-sections and further subdivisions via lozenges, periods and asterisks.

(2) Section I of the output was called "1108 STANDARDS". This included a list of 23 requirements for running on the particular system used. This included information concerning external labels, entrance and exit requirements, tags, external drum equates, information about standard date, tape codes, etc.

(3) Section II was called "A CHRONOLOGICAL HISTORY OF COLUMNS 79 AND 80". This reference to the columns mentioned gave in detail information about successive assembly dates and implementation timetable. The listing included all changes made in these assemblies and their dates. These columns are used to record program modification levels. This section also gives a detailed explanation of what these modifications were and why they were made.
(4) Section III, "PROGRAM INTERFACE AND RELATED INFORMATION". This included instructions to follow in interfaces, overlays, flags and related information. Also included in this section is information on interrupt analysis.

(5) Section IV, called "BUFFERS, MAPS, AND TABLES", gave drum maps, circuit lists, and several pages of tables. Included was polling information, table assignment tables, and requirements for polling. All of this information was dated.

(6) Section V, called "PROGRAM DESCRIPTIONS", contained the names of and information about all program segments used in this run. Each program was listed separately and its contents and relation to other programs were given. Information concerning on-line or not, where control is received from, specific function of program, what all message codes mean, register usage, and meanings of switches and table indexes were included.

Following the above information, which, incidentally, covered 85 printed pages, came a complete program listing for each program used. Comments were used freely throughout.

It should be emphasized that all of this information (196 pages of it!) was printed output. Assembly dates and codes were included on each entry.
CONCLUDING COMMENTS

The preceding pages give a summary of what might be desired in some standards for systems and program documentation. The automatic production of such information is a more challenging problem. Not only do standards not exist, but to aggravate matters, documentation software now available fails to meet any of the existing standards. Flowcharting packages do not follow the "rules" already established and standard symbols are not consistently used.

There is no reason to believe that reasonably good automatic documentation cannot be developed. Two possible approaches to achieve this goal might be: first, if some reasonable documentation standards can be established, users should then force software vendors to meet these standards. Secondly, if a good automatic documentation system could be developed which would satisfy a large portion of the users, its general acceptance could make it the standard. The establishment of good standards for program documentation will enhance communication at the local project level, between different user groups and/or with outside contractors.
BIBLIOGRAPHY


3. *IBM Study Organization Plan*, IBM Corporation, 1963
   - The Approach F20-8135
   - The Method Phase I F20-8136
   - The Method Phase II F20-8137
   - The Method Phase III F20-8138
   - Documentation Techniques C20-8175


5. Documentation Standards, USAF - Tinker AFB, 1971

6. Military Standards #480, #483, #490, USAF

7. MRI Systems Corporation, Samples of Documentation, (Not a publication)


AUTOMATIC SYSTEM FOR COMPUTER PROGRAM DOCUMENTATION

Technical Proposal

Submitted to

National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, Maryland 20771

Submitted by

Data Processing Center
Texas Engineering Experiment Station
College of Engineering
Texas A&M University
College Station, Texas 77843

October 1972
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>I-4</td>
</tr>
<tr>
<td><strong>Part I - Technical Proposal</strong></td>
<td></td>
</tr>
<tr>
<td>1.0 Introduction</td>
<td>I-6</td>
</tr>
<tr>
<td>2.0 System Design</td>
<td>I-8</td>
</tr>
<tr>
<td>3.0 Automatic Documentation System Design</td>
<td>I-9</td>
</tr>
<tr>
<td>3.1 System Features</td>
<td>I-9</td>
</tr>
<tr>
<td>3.2 Initial Phase</td>
<td>I-12</td>
</tr>
<tr>
<td>3.3 System Structure</td>
<td>I-15</td>
</tr>
<tr>
<td>4.0 Advantages</td>
<td>I-26</td>
</tr>
<tr>
<td>5.0 Future Expansion</td>
<td>I-26</td>
</tr>
<tr>
<td>6.0 Plan of Activity</td>
<td>I-28</td>
</tr>
<tr>
<td>6.1 Task 1 - Detail Design.</td>
<td>I-28</td>
</tr>
<tr>
<td>6.2 Task 2 - Program Implementation</td>
<td>I-29</td>
</tr>
<tr>
<td>6.3 Task 3 - Program Checkout and Test</td>
<td>I-29</td>
</tr>
<tr>
<td>6.4 Task 4 - System Installation at Goddard</td>
<td>I-30</td>
</tr>
<tr>
<td>6.5 Task 5 - Study and Evaluation</td>
<td>I-30</td>
</tr>
<tr>
<td>7.0 Data Documentation and Reports</td>
<td>I-30</td>
</tr>
<tr>
<td>7.1 Progress Report</td>
<td>I-30</td>
</tr>
<tr>
<td>7.2 Briefing</td>
<td>I-30</td>
</tr>
<tr>
<td>7.3 Detailed Design</td>
<td>I-30</td>
</tr>
<tr>
<td>7.4 Deliverable Items</td>
<td>I-30</td>
</tr>
</tbody>
</table>
7.5 Final Project Report. . . . . . . . . . . . . . . . . . . I-31

8.0 Program Organization . . . . . . . . . . . . . . . . . . I-32

9.0 Personnel Qualifications . . . . . . . . . . . . . . . . . I-33

10.0 Facilities . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . I-36

11.0 Program Schedule . . . . . . . . . . . . . . . . . . . . . . . I-37
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Program Development Process</td>
<td>I-13</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Automatic Documentation System Structure</td>
<td>I-16</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Type 1 Programs</td>
<td>I-17</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Type 2 Programs</td>
<td>I-18</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Production of Documentation</td>
<td>I-21</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Levels of Documentation</td>
<td>I-23</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Type 3 Programs</td>
<td>I-24</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Programs Tasks</td>
<td>I-27</td>
</tr>
</tbody>
</table>
SUMMARY

Part I of this proposal discusses the technical approach to the implementation of an automatic system for computer program documentation. Section 1.0 contains a statement of the philosophy. Section 2.0 contains general background information and a concise summary of studies that led to the system design. Section 3.0 contains detailed design specifications for the automatic documentation system. Section 4.0 lists the advantages of the initial phase of the implementation of the automatic documentation system and Section 5.0 covers future expansion. Plan of activity, reports and documents, program organization, personnel qualifications, facilities, and program schedule are contained in Sections 6 through 11.
Part I - Technical Proposal
1.0 INTRODUCTION

The Data Processing Center at Texas A&M University is pleased to submit this proposal to the National Aeronautics and Space Administration, Goddard Space Flight Center in Greenbelt, Maryland. The purpose of the proposed project is to implement an automatic system for computer program documentation. The system will produce timely, up-to-date documentation at relatively low cost. The system will be designed to document any computer language and run on any hardware while taking advantage of existing documentation aids. The system will be easy to use and will place minimum restrictions on the programmer.

In computer program development, one of the major problems contributing to low programmer productivity is poor communications. The automatic documentation system will produce documentation at all phases of system design and implementation thereby reducing the communication problem.

The proposed project will span twelve months. The twelfth month will be reserved for installing the automated documentation system at Goddard Space Flight Center in Greenbelt, Maryland. The program is divided into five tasks. The goal of Task 1 will be to describe in detail all of the programs that will be written. Since many of the program requirements are already known, Task 2, writing the programs, can begin concurrently with Task 1 and will take about eight months. System integration will begin at the end of the third month and will be Task 3. Task 3 will include the program checkout phase and should cover about seven months. Once the system is in operation at Texas A&M University, Task 4, installation of the system at Goddard Space Flight Center in Greenbelt, Maryland, will begin. Task 5 commences at the end of the second month.
when the detailed design is being phased out. This task will be the continuation of the evaluation and planning phase of the automatic documentation project. At the end of twelve months, personnel working on Task 5 will have evaluated the existing documentation system and proposed a follow-up system for extending and adding capability to the automatic documentation system. Documentation of the automatic documentation system will be delivered along with the final report at the end of the twelfth month.

The Data Processing Center along with the Computer and Information Sciences Division of the Industrial Engineering Department possesses the resources, know-how, and interest to successfully conduct this program. Dr. D. B. Simmons, Director of the Data Processing Center at Texas A&M University, will act as principal investigator. He is uniquely qualified to oversee the implementation of an automatic system for computer program documentation. The Data Processing Center has experience with implementing and operating computer software systems of all sizes and complexity. All necessary hardware and manpower resources will be made available to successfully complete the proposed project. Dr. Simmons has experience in all levels of computer hardware and software design. While an officer in the Signal Corps he evaluated both hardware and software for the U.S. Army. As a member of the staff at Bell Telephone Laboratories he designed and implemented a design automation system for electronic switching systems. One of the main by-products of the system was automated documentation. Also while at Bell Telephone Laboratories, Dr. Simmons designed and implemented the FLARE automatic flowcharting system. This system reduced the cost of producing a flowchart page at Bell Telephone Laboratories from $75 to 50¢. At present this system is the primary flowcharting system used to document programs for electronic switching systems.
The Data Processing Center and the Computer and Information Sciences Division believe that the tasks and systematic approach presented in this proposal are the steps necessary to develop a cost-effective user-oriented automatic documentation system. The proposed system will offer many capabilities that do not exist in any similar system today.

2.0 SYSTEM DESIGN

Under sponsorship of NASA contract NAS5-11911, a detailed survey of all existing documentation aids was conducted. Initially all computer-oriented literature was searched for anything related to documentation or documentation aids. An automated bibliography was used to record all references and key words relating to documentation areas. While documentation is one of the critical areas in software development, literature describing documentation is sparse.

A fruitful source of information about documentation aids was literature supplied by software houses that sell proprietary systems. Every organization that could be found advertising proprietary software aids was sent a letter requesting information. The surprising thing from the information received about the proprietary systems was the lack of features offered by them. Fairly unsophisticated systems are marketed at relatively high prices. No organization has implemented a comprehensive system that covers the whole spectrum of documentation aids. To use existing systems, users are usually required to become expert in numerous special procedures and conventions.

The proposed system will document programs written in FORTRAN, COBOL, PL/1, and Assembly language for the IBM 360 series. It can be easily expanded for use with the Univac 1108 and the CDC 6600 series computers. The various
operating systems were examined and evaluated for similarity. Working papers summarizing the conclusions found in the area of flowcharts, decision tables, operating systems, text editors, program editors, documentation standards, and computer listing formats were produced.

Audio techniques are valuable for capturing documentation information. An experiment was conducted to determine if audio techniques would be useful as an alternative to written text. While it was found that audio techniques have special areas of usefulness, no conclusive evidence was found to justify replacing other techniques with audio documentation.

Following the evaluation of existing documentation aids, a comprehensive automatic documentation system was designed. Details of the design will be presented in the following section. A preliminary design was presented to the NASA representatives at Ames, California during August, 1972. Suggestions made by NASA were incorporated into the design. As a result of the August meeting, a typical program was chosen to demonstrate the type of documentation that would be produced from the automatic documentation system.

3.0 AUTOMATIC DOCUMENTATION SYSTEM DESCRIPTION

3.1 SYSTEM FEATURES

The automatic documentation system will have the following features:

1. **Minimal programmer restrictions** - For the programmers who write programs in their own unique way, the automated system will be able to produce documentation such as detailed, detail-suppressed, and global flowcharts, data layouts, overlay descriptions, etc. Special cross-reference glossaries can also be produced. Those who use the automatic documentation system during initial project
phases will automatically obtain extensive documentation. But the system will also be designed so that if a program is developed outside the system it will be fairly easy to retrofit the program into the system for documentation maintenance.

2. **Eliminate all redundant effort** - Documentation produced in an early phase of the design or implementation process can be reused later on in the development process. For example, if the designer of a programming system describes in detail the function of each program subroutine that is produced, the programmer need not redefine the function of the routine when he writes his program. The comments describing the purpose of the subroutine will be automatically inserted in the programmer's subroutine.

3. **No operating system modification** - No justification can be found for modifying an operating system to obtain documentation. Most items of documentation interest can be obtained by scanning output produced by operating systems. Information available in internal tables of an operating system can be reproduced with less effort than it would take to make and maintain modifications to the operating system.

4. **Use existing documentation aids** - Many man-years of programming effort have gone into developing existing documentation aids. A number of flowcharters already exist. Text editors containing sophisticated algorithms for hyphenation and text layout have been developed. Existing documentation aids will be used as modules in the comprehensive automated system.

5. **Interactive/batch system** - Users will be able to make use of the system by using batch or interactive systems. The data base
management system that will be used can accept data using either an interactive or a batch mode. The interactive mode will have special features for interrogating the user of the system to obtain necessary documentation data. In the batch mode, the user will supply necessary information using key word or positional parameters. The most user-oriented technique would be the interactive version.

6. **Documentation during development** - The documentation data base will be constructed from information gathered during all design phases. For example, background or design philosophy sections generated for system specifications can also be used in final documentation. System flowcharts and block diagrams entered during system design can be retrieved from the data base to produce final documentation. Such things as title of program, person responsible for the program, function of subroutine, etc. prepared during the design phase would not have to be re-created during the program implementation phase. The users would supply only that information not available from a previous stage.

7. **Accept any language** - The automatic documentation system will be language independent. Initially the system will be designed to accept FORTRAN, COBOL, PL/1, and Assembly languages. There will be no restrictions inherent to the documentation system design that will prevent it from being used to document other languages.

8. **Operate on any hardware** - Initially the system will be designed to operate on the IBM 360 with planned expansion to the Univac
1108 and the CDC 6600 computer systems. These are the three major systems used by NASA. All programs written for the automatic documentation system will be written in a machine independent language so that the automatic documentation system can be easily moved from computer to computer.

9. **Monitor and control project** - Features will be designed into the system to allow the project manager to monitor the exact status of program development and documentation. In addition, system access and security will be under his control.

### 3.2 INITIAL PHASE

Work covered by this proposal is termed the initial phase of development of the automatic documentation system. The system will be designed to allow new features to be added at a later date. During the initial phase either FORTRAN or COBOL will be used to write programs that make up the automatic documentation system. Initial development will be for a system operational on an IBM 360 computer, but software will be easily transportable to the Univac or CDC systems. The automatic documentation system will document COBOL, FORTRAN, PL/1, or Assembler language programs.

The development process can be segmented into many phases. One way of subdividing the process is shown in Figure 1 where monitor, specifiers, designers, programmers, and validators participate in the development of computer software. From the very beginning of the program development process, the monitor can determine exactly what can be inserted into the data base. The specifier writes the Request for Proposal (RFP) or outlines a need for a program to be developed. He can enter such items as title, abstract, system specifications, testing criteria, block diagrams, and other elements of the RFP. All of
Figure 1 Program Development Process
this information will be stored in the documentation database. In addition to documentation information, the data base will contain the program source, object and job control modules. In other words, the data base will contain all information necessary to describe and develop the program.

From global specifications, such as the RFP, the designer would draw a detailed system flowchart, produce system descriptions, data descriptions, program standards, subprogram design, test procedures, and produce a development schedule. All design information will be stored into the data base. Documentation aids such as text editors, will minimize the designer's effort related to writing and rewriting his design documents. Information such as background material and subroutine descriptions will be carried through to final documentation. Important design philosophies would be captured at early stages and retained.

Once the design is complete, the programmer can write source code for the program and store it in the data base. From source code and information in the data base such things as program flowcharts, data layouts, glossaries, overlay structures, and extensive cross-reference information can be produced. The monitor program will assist the project manager in determining whether a programmer is conforming to documentation standards and using good programming practices. The monitor can determine the status of the project. Access to different modules will be controlled by the program monitor. Documentation produced during the development process will aid programmers in understanding the operation of other parts of a programming system. This is a major step forward in improving communications between programmers and greatly improves programmer productivity.
Once a set of programs has been completed, a validator can verify that they work. All information necessary for the validation process will have been established by either the specifier, designer, or programmer.

Once the development and implementation of a program has been completed, documentation can be produced for the operation and maintenance phases. The automatic documentation system will produce such things as a user, operator, and maintenance manuals and abstract information for user libraries.

3.3 SYSTEM STRUCTURE

The structure of the proposed automatic documentation system is shown in Figure 2. Three types of programs will be used in the system. Types 1 and 2 are new programs to be developed. Type 3 are existing programs that can be used without change. Type 3 programs make up a major part of the software necessary to implement the automatic documentation system. New programs will not be developed where operational documentation aids are available. By doing this, a sophisticated system will be developed at a relatively low cost.

Type 1 programs make up the executive program routines which will constitute a small part of the overall system. The executive will control access to the system, do syntax analysis on the basic commands (both batch and interactive), gather usage statistics which are not a function of the database or a single application program and initiate Type 2 programs. Type 1 functions are shown in Figure 3. The major development effort will be concentrated on Type 2 programs.

Type 2 programs that will be developed are listed in Figure 4. A generalized editor will take the place of the normal program editor and do preprocessing
Figure 2  Automatic Documentation System Structure
EXECUTIVE

1. System Access Security
2. Command Processor (Batch/Interactive)
3. Use Statistics
4. Type 2 Initiators

Figure 3 Type 1 Programs
DEVELOPED APPLICATION PROGRAMS

1. Monitor (Control & Statistics)
2. Data Collection
3. Interrogators
4. Batch Command Processors
5. Template Builders
6. Recipe Builders
7. Recipe Scanners
8. Drivers

Figure 4  Type 2 Programs
for a text editor. The reason for using a preprocessor is to facilitate the use of many different text editors. The text entered by the user will be put into a canonical form which can be transformed for use with any proprietary text editor that the user may choose. The editor will also have the ability to insert text information into a program listing without altering the program or data structure.

A monitor program will be developed which will contain control features and produce necessary statistics. The monitor program will check the program listing to insure that proper standards and programming practices have been used. The documentation data base will be checked and statistics produced identifying the types and amounts of data supplied and items that are missing. Exact project status can thus be determined. Documentation error analysis information and access to all data base and program modules will be controlled by the monitor.

Data collection routines construct the data base. Information supplied by the programmer will be inserted into the data base by data collection programs in a manner transparent to both language and hardware. A standard data base management system will be used, greatly reducing the amount of programming effort necessary to implement the automatic documentation system.

The interactive version of the automatic documentation system will have interrogators for prompting users. In the batch system, a command processor will be necessary to perform the same function as the interactive interrogator system.

During the specification, design, programming, and validation process appropriate information will be stored in the data base. Documentation required during development can then be produced automatically. Data templates
will define information to be stored in the data base. To produce a manual, a recipe must be defined. This recipe consists of items describing exactly what type of information is required in the manual. Recipe building programs will specify the structure for final documentation reports such as user, operator, or maintenance manuals. Recipe scanner programs will then scan these recipes and drive the necessary programs to produce a complete manual. Templates for the data base and recipes for final documentation will be under the complete control of the manager of a programming project. He can decide exactly how much documentation and what type of documentation is to be produced. When the manager does not want to worry himself with format details of final documentation, standard templates of the data base and recipes for the manuals will be supplied by the system. Recipe processing programs will scan the recipe and drive the programs necessary to produce a complete manual. Drivers are called by the recipe scanner that drive application programs.

Relationships among program modules are shown in Figure 5. For example to produce a manual, a recipe for the manual must be defined. Text information entered during the specification or design phases may be needed in the user's manual to describe philosophy behind program development. These items could be specified as part of the recipe. At the point in the manual that the system flowchart appears, the flowchart will be produced automatically from the job control language. To produce the manual, the recipe will be scanned and the appropriate drivers called. To produce the text part of a manual, a driver will take text from the data base and drive a text editor. Using this technique, a different text editor can be used for each different computer. For example, the text editor used for the IBM system might be the TEXT360 or ATS system while for the CDC system, the text editor EDIT, under the KRONOS operating system may be used. Drivers will be tailored to drive available
Figure 5  Production of Documentation
flowchart systems to produce flowcharts of programs in various languages.

The system will be able to produce documentation which is not currently available from any existing system. Figure 6 shows levels of documentation that can be produced by the system. Most proprietary systems produce documentation at the local level. They typically produce listings, flowcharts, and some type of cross reference. Some, but not all, systems produce detail suppressed flowcharts. A few are able to produce functional flowcharts. Data layout descriptions usually must be prepared by hand. The proposed system will be able to give a two-dimensional detailed data description and a detail suppressed description. In the future functional data layouts will be automatically produced.

The local level of documentation is associated with a single compilation or assembly. For a program composed of a number of separate modules, global flowcharts, and overlay diagrams are useful. The proposed system will be able to produce these automatically. Global glossaries, cross-reference tables, and data layouts will also be produced. Text will be produced at all levels of documentation. Global flowcharts will be produced from load modules and system flowcharts will be produced from the job control language. System block diagrams can be stored in graphic format and reproduced by the documentation system. All levels of documentation are necessary to understand a complex program. Existing commercial systems give only a small fraction of the necessary information.

The Type 3 programs which will be utilized by the system are shown in Figure 7. These programs will be utilized without modification. For the IBM 360 version, flowcharting systems such as OS/360 Flowchart, QUICKDRAW, or AUTOFLOW will be used. A number of text editors are also available. For the initial phase, the TEXT360 text editor will be used. Output from operating
Figure 6 Levels of Documentation
EXISTING APPLICATION PROGRAMS

1. Flowcharters
2. Text Editors
3. Program Editors
4. Index Generators
5. Compilers
6. Tidy Programs
7. Utilities
8. Linkage Editors/Loader
9. Data Base Management
10. Descriptive Statistics

Figure 7 Type 3 Programs
systems and compilers will be used as input to programs that draw cross-reference tables and flowcharts. Therefore, compilers, linkage editors, and loaders are Type 3 programs. Another useful Type 3 program is the TIDY program which does such things as renumber FORTRAN statements in ascending order.

The key to development of the automated documentation system will be use of a generalized data base management system. The System 2000 data base management system developed by MRI Systems Corporation of Austin, Texas will be used. System 2000 is a general purpose data management system with features that include a report writer, a user-oriented language providing on-line access to non-programmers, a procedural language interface for programmer use, sequential file processing, two teleprocessing monitors, and a multiple thread feature. The system provides archival copies of data bases and records an audit trail of changes made to the data base. It is capable of reconstructing a data base by applying an audit trail, completely or in part, to an archival copy of the data base. The procedural language features enable users to manipulate data from COBOL, PL/1, FORTRAN, or Assembler language programs. The data base can be accessed from model 33/35 teletypewriters, IBM 2741 hard copy terminals, and IBM 2260 CRT terminals.

System 2000 provides a wide range of features to insure data base integrity. Five levels of security are provided. Security can be provided at remote terminals through the use of terminal identification or passwords. Password security is available at the system, data base, command, and component level. Component level security enables the data base administrator to offer four types of access for each component in the data base definition. Thus, for each component in the data base there exist sixteen possible access combinations.
4.0 ADVANTAGES

Use of the proposed automatic documentation system offers many advantages over other techniques for developing programs and producing documentation. The system will be user-oriented and will be as easy to operate as existing on-line or batch program editors. Programmers who use good programming practices when developing software can use all of the system features without extra effort. Programmer productivity will be enhanced by improved communication during the development process. The modular system will allow new types of documentation to be easily added to the system. This is the first system that brings together all types of documentation aids into a single user-oriented system. It emphasizes documentation on a load module and system basis as well as for a single compilation. Documents can be produced that are made up of heterogenous output such as text and flowcharts. Managers can use the system to control and monitor projects. Program and documentation standards can be enforced and taught through the use of the automatic documentation system. People who do not use the documentation system during program development will be able to use it for post-development documentation. Programs developed independently of the automatic documentation system can easily be retrofitted into the documentation data base.

5.0 FUTURE EXPANSION

As previously mentioned, the system has been designed in a machine independent and language independent manner so that it can be easily moved from machine to machine and used to document various languages. During the initial phase the only graphics will be flowcharts. Graphic capabilities can be added to the system at a future time. Any type of two-dimensional drawing
Figure 8  Program Tasks
could then be placed in a data base. Provision for drawing sophisticated data layout descriptions can be added to the system. Documentation of data bases and complex data structures can also be automatically drawn. During the initial phase the system will be designed to run on the IBM 360, but it can easily be expanded to systems such as the Univac 1108 and CDC 6600 that have FORTRAN or COBOL compilers. A syntax-directed language extension program can be developed to facilitate addition of new languages and output devices.

Long-term future developments could be to look into the possibility of producing all types of documentation from programs that contain unstructured comments. A number of program ambiguities can be resolved by simulating program execution. Management tools such as PERT diagrams of project status can be automatically produced from information found in the documentation data base.

6.0 PLAN OF ACTIVITY

The implementation of the automatic system for computer program documentation will be divided into five tasks as shown in Figure 8. The first task will be to design each program in detail. The second task will be to write the programs. The third task will be to test the programs. The fourth task will be to install the working system at the Goddard Space Flight Center. The final task will be to continue the study and evaluation of documentation aids to extend the initial phase of the automatic documentation system development.

6.1 TASK 1 - DETAIL DESIGN

While the design of the overall system will be completed before the projects starts, the detailed design of each routine of the system will take place during the initial development process. A design document will be
produced as a result of Task 1. Final documentation describing the automatic documentation system will be produced by the automatic documentation system.

6.2 TASK 2 - PROGRAM IMPLEMENTATION

Task 2 will be the major program implementation phase. Programming will fall into four major areas: (1) monitor program, (2) template, recipe, and data base builders and recipe scanner, (3) program editor, text editor driver, routines for storing FORTRAN, Assembler, COBOL, and PL/1 source programs and routines for using a language processor and operating system as input (4) drivers to produce local, global, and system documentation. A programmer will be assigned to each of these major areas. In addition, a number of modules for producing different types of documentation will be produced by graduate students as part of their normal course work at no cost to NASA.

6.3 TASK 3 - PROGRAM CHECKOUT AND TEST

The automatic documentation system will document the following programs: (1) All Type 1 and Type 2 programs of the automatic documentation system, (2) A scientific program written in FORTRAN, (3) Administrative programs developed by the Texas A&M University Data Processing Center. Checking and debugging of the system will be done by the programmers assigned to software development. Exercising of the system will be done by graduate assistants assigned to the project. Members of the Data Processing Center staff who use the automatic documentation system to document their programs will not charge their time to the project. Computer costs resulting from documentation produced for the Texas A&M University Data Processing Center will not be charged to the project.
6.4 TASK 4 - SYSTEM INSTALLATION AT GODDARD

The system will be installed at Goddard Space Flight Center after it has been extensively checked out at Texas A&M University. Therefore, installation and on-site evaluation should take less than a month. Project evaluation will be conducted during this period.

6.5 TASK 5 - STUDY AND EVALUATION

Study and evaluation of all types of documentation aids will continue. The bibliography on automatic documentation will be updated and maintained. Personnel assigned to this task will do an on-going evaluation of the automatic documentation system. Future extensions will be designed by this group.

7.0 DATA DOCUMENTATION AND REPORTS

Texas A&M University will furnish the National Aeronautics and Space Administration the following items.

7.1 PROGRESS REPORT

A monthly progress report covering progress made during the previous month shall be delivered to the Contracting Officer by the tenth of the month following the month reported. This report shall state, in concise terms, items such as accomplishments, estimates of funds, commitments during the reporting period, plans for the next period, problems and anticipated delays, and specific recommendations to facilitate execution of the contract.

7.2 BRIEFING

During the first month after contract award, a meeting with the Contract Officer and other interested NASA officials in Maryland can be held to discuss
the detailed design. At this meeting, Texas A&M will discuss planning for project activities. At the end of the fourth month, the detailed design will be completed and can be presented to the Technical Officer and other members at NASA. If deemed appropriate, another briefing can be given at the end of the eighth month. At the end of the project, a briefing can be given to describe what has been accomplished and suggest system extensions.

7.3 DETAILED DESIGN

At the end of the fourth month the detailed design will be completed and a written report will be produced. This design document will be fed into the data base as an integral part of the documentation to be produced for the project.

7.4 DELIVERABLE ITEMS

Source and object programs will be delivered to NASA in Greenbelt, Maryland and installed on an IBM 360 computer. Complete documentation of the programs will be supplied. All of the documentation will be produced using the automatic documentation system. Final documentation produced will be an operator's manual, user's manual, and maintenance manual.

7.5 FINAL PROJECT REPORT

The contractor shall furnish a draft of the final project report within 338 days after the effective date of the contract. The government shall be allowed seven days to review and return the report with comments and recommendations to the contractor. The comments and recommendations of the Contract Manager shall be taken into consideration in preparing the final report which will be submitted to NASA within 21 days after the Contract Officer approval of the date. The final project report will completely document all research,
recommendations, and results of the efforts during the performance of the contract.

8.0  PROGRAM ORGANIZATION

The proposed program will be conducted by members of the professional staff of the Data Processing Center and of the Computer and Information Sciences Division within the College of Engineering at Texas A&M University. Dr. D. B. Simmons, who is Director of the Data Processing Center and a member of the Computer and Information Sciences Division faculty will act as principal investigator. The Data Processing Center facilities are designed to accomodate the teaching, research, and administrative needs of the university. Approximately 50% of the machine time is used by research projects, 20% by academic efforts, and 30% for administrative processing. The Data Processing Center employs approximately 80 people, including 33 professionals. Personnel are organized into seven groups: System Software, Computer Operations, Computer Systems, Administrative Applications, Agricultural/Statistics, Office Operations, and Fiscal. The Data Processing Center operates as a separate entity with a 72-73 Fiscal Year budget in excess of $1.5 million.

The Computer and Information Sciences Division is part of the Industrial Engineering Department within the College of Engineering at Texas A&M University. The Industrial Engineering Department has one of the best programs in the country maintaining an effective balance between teaching and research. The Computer and Information Sciences Division was established in 1963 and its growth is best indicated by its size:

(1) 16 professional staff members - 9 with Ph.D.'s and 7 with Ph.D.'s in progress.

(2) 148 graduate students, 112 at the Master level and 36 working toward a Ph.D.
(3) a new undergraduate program in Computing Science will start in the late fall of 1972.

9.0 PERSONNEL QUALIFICATIONS

The Data Processing Center along with the Computer and Information Sciences Division possess the resources, know-how, and interest to successfully conduct this project. The following people will work directly on the project: Dr. D. B. Simmons - one quarter time, Dr. R. W. Elliott - one quarter time, Dr. D. Colunga - one quarter time, Ms. S. Arseven - full time, Mr. G. H. Kemper - full time, Mr. M. H. Lyle - full time, and Mr. P. Crews - full time.

Dr. D. B. Simmons, who will serve as principal investigator, has been associated with computer field for over 13 years. He started in circuit design in 1959 on the early RCA semiconductor computers. He served in the Army as Systems Evaluator for computer systems used by the Army. In 1963 he joined Bell Telephone Laboratories where he worked in the areas of logic design, design automation, automatic program documentation, and automatic flowcharting. He designed and developed the FLARE automatic flowcharting system. He served as supervisor of the Advanced Programming and Processor Technology Group which worked on designs for the 1980's, programs for improving design automation systems, high-level electronic switching systems (ESS) languages, and automatic flowcharting systems. Since joining Texas A&M University in 1970, Dr. Simmons has been involved in teaching, consulting, and development of computer operating systems. He has served as principal investigator on a project for the development of an operating system for mini-computers and as principal investigator on the project to design the automatic documentation system for NASA. He is currently Director of the Data Processing Center and Associate Professor of Computing Science.
Dr. Elliott's main area of interest is Computing Science with specialization in computer graphics and information retrieval. Dr. Elliott was Associate Director of the Texas Regional Academic Computing Experiment Project sponsored by the National Science Foundation. This project had regional computing facilities to Tarleton, Prairie View, and Texas Southern Universities from Texas A&M University's computer center. Dr. Elliott has experience in computer programming, consulting in various computer areas, and in teaching computer oriented subjects.

Dr. Colunga has worked with computers since 1958. From that time, he has acquired experience on the following computers: IBM: 650, 1620, 7090, 7094, 360/65; CDC: 1604, 3600, 6600; Univac 1108. Until last year Dr. Colunga was associated with the Theory and Analysis office of the Computation and Analysis Division, under the direction of Eugene Brock at NASA/MSC at Houston, Texas. While there, Dr. Colunga initiated efforts toward the problem of documentation of control-optimization programs available to NASA/MSC users through the program share library facilities coordinated by Mr. John Leonard. Dr. Colunga is therefore aware of the problems associated with NASA's needs for scientific computer program documentation.

Ms. Susan Arseven has been active in the field of Computer and Information Science for more than 8 years. She began her participation in 1964 designing and marketing automated systems for libraries and information centers in New York City for IBM Corporation. In 1967 she went to the University of Pennsylvania as the Head of the System Planning Office at the University Library where she designed and developed an automated circulation control system, a serials catalog, and a book acquisition system and served as advisor to the
Director of Libraries on computers and automated techniques. She has made major contributions to the project to design the automated documentation system for NASA. She is currently a Systems Analyst at the Data Processing Center and an Assistant Professor of Computing Science.

Mr. Kemper and Mr. Lyle have both had extensive experience in systems programming and applications programming at Texas A&M University and in industry. They are currently both senior members of the Systems Programming Group of the Data Processing Center. Mr. Crews, a new addition to the programming staff at the Data Processing Center, has a background in scientific programming and has developed a mini-computer independent macro processor.

Besides the above mentioned personnel, this project will use other members of the Data Processing Center and Computer and Information Sciences Division faculty on a consulting basis. In addition, there are a number of Air Force officers who are working on Master's and Ph.D. degrees in Computing Science. They have extensive experience in application areas of computers within the government and have a first-hand knowledge of documentation problems. Any Air Force officer who works on this project in conjunction with a Master's or Ph.D. dissertation will charge no time to the contract since he is already being fully funded by the government.

In addition to the professional staff, three research assistants will work in a supporting capacity. Also a half-time secretary will support the project. The Data Processing Center and the Computer and Information Sciences Division have the expertise in designing, developing, and coordinating software systems to successfully accomplish the goals stated in this proposal.
10.0 FACILITIES

Texas A&M Data Processing Center is a centralized facility serving the computing needs of the entire Texas A&M University campus. It is responsible for all of the administrative data processing, educational support, and research support on campus.

The principal computer is an IBM 360/65 with 512 thousand bytes of main core storage and 2 million bytes of extended core storage (large core storage). This computer is supplemented by an IBM 7090 system used to control experiments located in the Cyclotron building. In addition, there is a tape-oriented 1401 computer that supports the 7090 computer. There is also a PDP mini-computer supporting the IBM 360/65 computer.

The initial system 360 was installed in December, 1967. The present configuration includes three card reader punches, four printers, seven magnetic tape drives, 24 2314 disk drives, two IBM 1050 communication terminals, four 2740 terminals, five 2741 terminals, and 19 720 Sander Associates terminals. The computer also has remote job entry stations at other locations on campus such as the library, the Engineering Building, and it has entry stations off campus in Houston, San Antonio, Stephenville, and Austin. A CalComp Plotter and a Gerber automatic drafting machine are available for producing two-dimensional graphs.

Besides the main computer center, there are numerous small computers around campus. The Computer and Information Sciences Division has a Data General Nova computer with a magnetic tape unit, high speed paper tape reader, and an interactive CRT. Also the division has an intelligent terminal connected to the main computer.
11.0 PROGRAM SCHEDULE

The program will be divided into five tasks that will cover a period of 365 days. The tasks will be broken down as follows:

**TASK ONE - DETAILED DESIGN**

Starting Time - Beginning First Month

Duration - 4 Months

Personnel Involved:  
- Dr. D. B. Simmons - 1 man month  
- Dr. R. W. Elliott - 3/4 man month  
- Dr. D. Colunga - 3/4 man month  
- Ms. S. Arseven - 1 man month  
- Graduate Research Assistant - 4 man months  
- Secretary - 2 man months

**TASK TWO - PROGRAM IMPLEMENTATION**

Starting Time - Beginning First Month

Duration - 8 Months

Personnel Involved:  
- Ms. S. Arseven - 6 man months  
- Mr. G. H. Kemper - 8 man months  
- Mr. M. H. Lyle - 8 man months  
- Mr. P. Crews - 8 man months  
- Graduate Assistant - 8 man months

**TASK THREE - PROGRAM CHECKOUT AND TEST**

Starting Time - Beginning of Fourth Month

Duration - 8 Months