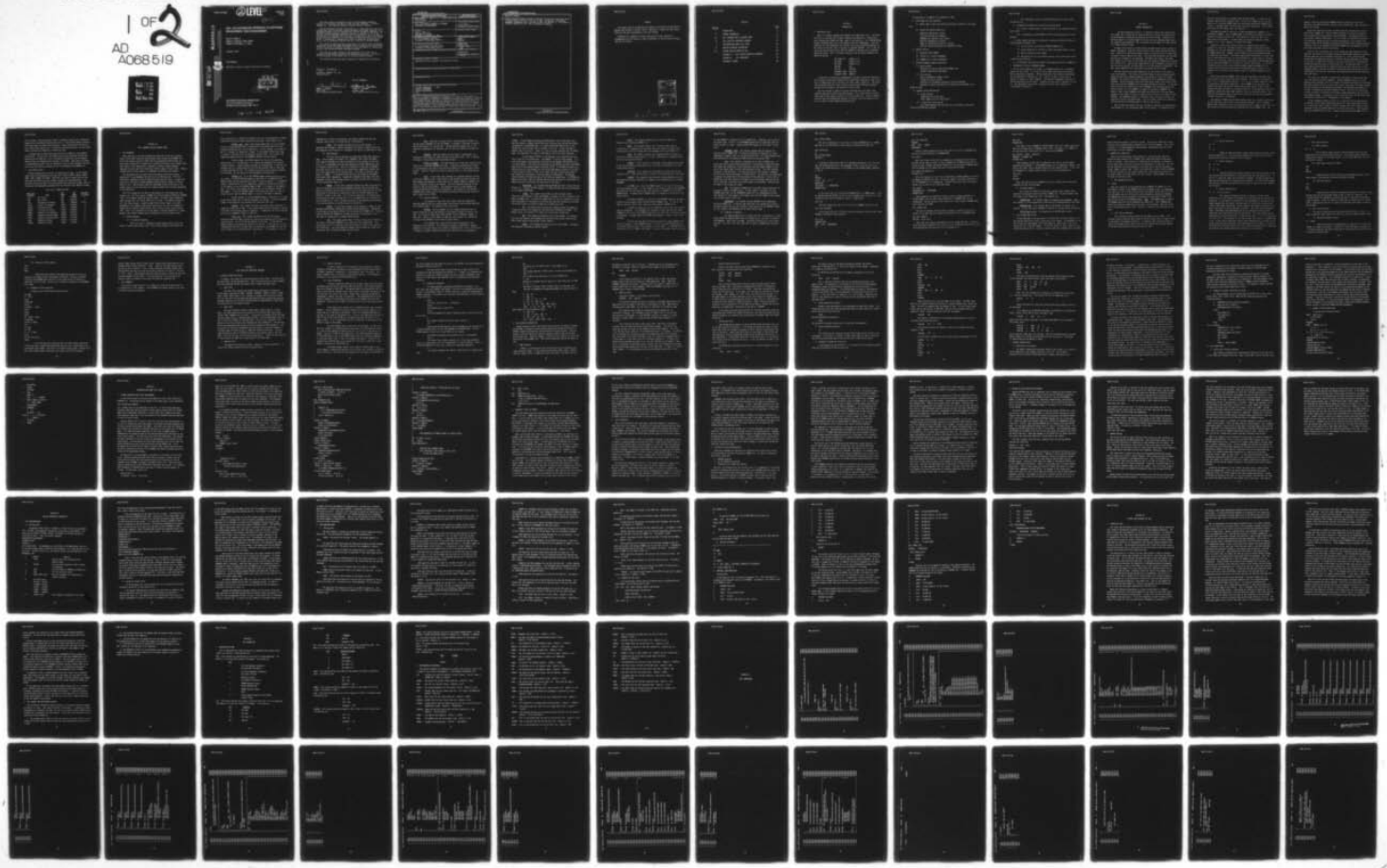
AD-A068 519 AIR FORCE WEAPONS LAB KIRTLAND AFB NM F/G 9/2
SAIL, AN AUTOMATED APPROACH TO SOFTWARE DEVELOPMENT AND MANAGEM--ETC(U)
JAN 79 L P GABY, D C GRAHAM, C E RHOADES
UNCLASSIFIED AFWL-TR-78-80 SBIE-AD-E200 258 NL



AFWL-TR-78-80

(2) LEVEL III

AFWL-TR-78-80

DDC

AD A068519

DDC FILE COPY

SAIL, AN AUTOMATED APPROACH TO SOFTWARE DEVELOPMENT AND MANAGEMENT

Lewis P. Gaby II
David C. Graham, Capt, USAF
Clifford E. Rhoades, Jr., PhD

January 1979

Final Report

Approved for public release; distribution unlimited.

AIR FORCE WEAPONS LABORATORY
Air Force Systems Command
Kirtland Air Force Base, NM 87117

This final report was prepared by the Air Force Weapons Laboratory, Kirtland Air Force Base, New Mexico, under Job Order 88091822. Clifford E. Rhoades, Jr., (DYP) was the Laboratory Project Officer-in-Charge.

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been authored by employees of the United States Government. Accordingly, the United States Government retains a nonexclusive, royalty-free license to publish or reproduce the material contained herein, or allow others to do so, for the United States Government purposes.

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

This technical report has been reviewed and is approved for publication.

Clifford E. Rhoades Jr.

CLIFFORD E. RHOADES, JR., PhD
Project Officer

FOR THE COMMANDER

Norman F. Roderick

NORMAN F. RODERICK
Major, USAF
Chief, Advanced Concepts Branch

Thomas W. Ciambrone

THOMAS W. CIAMBRONE
Colonel, USAF
Chief, Applied Physics Division

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFWL-TR-78-80	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SAIL, AN AUTOMATED APPROACH TO SOFTWARE DEVELOPMENT AND MANAGEMENT		5. TYPE OF REPORT & PERIOD COVERED Final Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Lewis P. Gaby, II Capt David C. Graham Clifford E. Rhoades, Jr., PhD		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Weapons Laboratory Kirtland Air Force Base, NM 87117		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62601F 88091822
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Weapons Laboratory (DYP) Kirtland Air Force Base, NM 87117		12. REPORT DATE January 1979 13. NUMBER OF PAGES 102
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

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

18. SUPPLEMENTARY NOTES

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

software management HULL
machine independent
software maintenance
SAIL

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

SAIL is a software development and management aid which provides a basis for disciplined design, programming, maintenance, and execution of computer programs. The system achieves a new level of centralized software development and maintenance control, while simultaneously decentralizing applications, providing controlled task specialization at compile time, and realizing portability between dissimilar machines. The system encourages economy by reducing duplication of effort often found when many versions of a basic source code are required. The

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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20. (Contd)

system achieves efficient hardware utilization by providing source code which is specialized at execution time for the task. Except for a few machine dependent statements, SAIL is written in ANSI (66) FORTRAN IV. The system is currently operating on CDC, IBM, and Honeywell computers.

PREFACE

The authors wish to acknowledge the numerous contributions by Maj Daniel A. Matuska, Maj Richard E. Durrett, and Dr. Reginald W. Clemens who were responsible for the early development of the SAIL program.

Reference to a company or product name does not imply approval or recommendation of the product by the US Government to the exclusion of others that may be suitable.

CONTENTS

SECTION I

INTRODUCTION

1. OVERVIEW OF SAIL

SAIL is a computer program that manages card-image data files. Like other utilities of its type, SAIL creates, updates, and operates on library files which contain data needed for other steps of job execution. As a FORTRAN-based program, SAIL provides a very high degree of machine independence. With the exception of a few statements and routines which must be machine-dependent for the sake of efficiency, SAIL is written in standard FORTRAN. Thus, it may be run on any machine which has an ANSI 66 FORTRAN IV compiler and a reasonable operating system. SAIL is currently running on the following computers and operating systems:

CDC CYBER 176	(NOS/BE 1.2)
CDC 6600	(SCOPE 3.4.4)
CDC 7600	(SCOPE 2.1.4)
IBM 360	(OS)
IBM 370	(OS/VS2)
Honeywell 6080	(MULTICS)
Honeywell 6080	(WWMCS)

In addition to high machine independence, the most significant features of SAIL include variable-value substitution, conditional selection of code, inclusion of common code, and a flexible, yet simple, macro capability. The selection and modification of lines of data are controlled by OPTION parameters according to the needs of the particular task at hand.

SAIL was written to address two major problems that arise when large computer programs are maintained and executed. First, duplication of effort often results when several versions of a source code exist. Second, computer time and space are wasted when problems of various sizes and complexity are run on a code of fixed dimensions and generality.

The following is a summary of the features of SAIL.

a. Single Mode for Each Execution

(1) Normal mode in which the executive processor produces a card-image file.

(2) Library maintenance modes are:

UPDATE for modifying a library.
LIST for printing its contents.
GENERATE for creating a new library.
COPY for reproducing a library.
PUNCH for punching portions of a library.
SCAN for finding occurrences of character strings.

b. Default Choices

(1) Operating mode (NORMAL)

(2) OPTION list (library dependent)

(3) PROGRAM list (library dependent)

c. OPTION Parameters (Name-Value Pairs)

(1) Which control:
Selection of cards written during NORMAL runs.
Dynamic substitution (see below)

(2) Which can be:
Set to nonnegative integer values.
Treated as logical values.
Defined by an arithmetic expression involving OPTIONS.
Employed in logical expressions for definition of OPTIONS or for selection tests.

d. Dynamic String Substitution

(1) Allows setting:
Dimensions to fit each task.
Other numbers such as loop indices.

(2) In which the string can be:
An integer encoded from the result of an arithmetic calculation involving OPTIONS and constants.

The alphanumeric name of an OPTION selected from a table within the OPTION list.

e. Intermachine Transport of Libraries and Change Cards

(1) Allows a library used on several machines to be supported from any one of them.

(2) Is enhanced by using OPTIONS to select the code peculiar to each machine.

f. PROCs (Blocks Defined for Conditional Insertion at Multiple Points on the Processed Card-Image File)

(1) Can be used for coordinating FORTRAN COMMON blocks.

(2) Allow reordering of routines or other contiguous blocks of code.

(3) May be inserted in other PROCs.

(4) Have macro ability so different strings can be substituted within a PROC at each insertion.

g. An Alternate Input Data to Control Data Generation During a NORMAL Run

2. OVERVIEW OF THE SAIL REFERENCE MANUAL

As its title suggests, this report is intended primarily as a reference manual for SAIL users. It is also of use to those who install and maintain SAIL on particular computers. Before using SAIL, users should be acquainted with the material in Sections II, III, IV and V, as well as that part of Section VI specific to their computer. The extensive examples in Section VI should prove particularly useful. Examples are also given in Sections III and IV. The appropriate job control language for typical SAIL executions is illustrated in Section VI.

SECTION II

GENERAL INFORMATION

The basic objective of SAIL is to maintain library files that contain card images for a related set of tasks so that the file for any specific task can be produced by selecting the appropriate control OPTION. Although this reference manual is oriented toward its original use of managing source cards for compilers and assemblers, SAIL can handle any card-image file that contains 72 or fewer columns of data.

A library file consists of SAIL records preceded by two logical records of header information. Each SAIL record consists of an identifier and 80 characters. The first 72 characters contain either a directive to SAIL or data that is useful to one or more of the tasks. The last eight characters give the date the card was inserted or last modified. Throughout this manual, the terms SAIL record or line refer to the identifier and character information as stored on a library file. Card or card image refers to the information as written to the file to be used for the task, that is, just the character information. The SAIL record identifier will occasionally be called the line number.

The SAIL records are stored sequentially on the library in the order of their line numbers. The library can be partitioned by the *B directive which identifies the cards between consecutive occurrences of itself as a named PROGRAM. The lines on the library preceding the first program are called the PROLOGUE. During executions other than those which create a new library, SAIL deals only with the PROGRAMs which have been selected by the user. If the user does not explicitly request one or more PROGRAMs, SAIL processes either the entire library or, for a NORMAL run, those on the default PROGRAM list. Default lists of both PROGRAMs and OPTIONs are maintained in the two header records of the library along with a table giving the first line number in each PROGRAM. The PROGRAM structure, as defined on this last table, can only be modified on runs in the UPDATE mode. The first header record also contains a SYSTEM name and the VERSION number to identify the library.

What actions SAIL performs during a particular execution are determined by the contents of a file designated INPUT in this manual. The general form of

this file has two parts, the request area and the changes. If there are any changes, there must be a request area. If the file is empty, all of the SAIL defaults will be taken (NORMAL mode with the default OPTION and PROGRAM selections). A nonempty INPUT file must begin with the request SAIL. For an exact description of the request area, see Section 3.

The operating mode for the run is either designated by a request or is defaulted to NORMAL. Only in NORMAL mode is the execution processor called to process the executive directives. (See Section III, para 1(a) for a full description of each mode.) In other modes the executive processor creates, updates, lists, scans, punches, or copies the library.

Changes to lines on the library consist of a general SAIL directive (Section III, para 2b), followed by cards to be added to the library either permanently or for the remainder of the run. Changes are recognized by SAIL for all modes except COPY. However, only in UPDATE runs are they made permanent by creating a new library. The library modification directives (1) allow insertion of cards after any line, (2) allow deletion of a single line or a consecutive set of lines with insertion of new cards in the place of the deleted cards, and (3) allow replacement of a string on a line by a new string. If a line or consecutive set of lines to be inserted already appears elsewhere on the library, the *M directive can be used to effectively place a copy of it in a change set at any place a card containing data or storable directives could occur.

Within each selected PROGRAM, SAIL lines are processed in the order they occur on the library as modified by the changes. Each record is examined to see if it is a recognizable directive. If it is, it is either ignored or acted on according to the mode of execution. During NORMAL runs, any data lines (those lines which are not SAIL directives) are written out to SAIL unless skipping has been initiated by a previous directive. An exception to the strict sequential processing occurs when groups of cards are defined as PROCs. The PROC with the matching name can be written to the file for the task wherever an *INCLUDE directive is encountered.

The subject of PROCs and selected skipping brings us to another distinction arising from the origin of SAIL in two separate programs. This is between general SAIL directives, which are recognized during all modes of execution, and SAIL executive directives which are processed only during NORMAL runs. The

general directives include the PROGRAM definition directive, the library modifiers, the INPUT modifiers, and the listing controllers. Most are used to manage the library.

Executive directives are used to select cards for the particular task at hand during a NORMAL run. Most of these directives can contain an operand field consisting of an expression involving OPTIONs. The operand field can have either a TRUE or FALSE value. If it is FALSE, it inhibits the action of the directive, thereby acting as a selection mechanism. The values of the operand fields can be changed by modifying the OPTION values, allowing numerous different files to be produced on different NORMAL runs corresponding to each particular task dealt with by the library.

Ideally, once a particular set of card images has been tested for proper operation and stored on the library along with appropriate executive directives, the user need only set the OPTIONs to activate it. Thus, an entirely new program may be created by selecting a hitherto untried combination of OPTIONs. If an error is found on the library, it can be corrected for all tasks whose OPTIONs activate the cards containing the error. On the other hand, if a new task requires modification of a group of cards, but the original set is still needed by others, the user can define a new OPTION and/or new values for existing OPTIONs to activate and inactivate the appropriate cards without affecting the other users.

This last use of OPTIONs is particularly applicable to source decks that are to be used on different computers. Source statements peculiar to each machine can be selected, while the library file remains the same on all machines. When SAIL produces the file for the task at hand, only those card images needed for that task and on that machine are present. Multimachine maintenance of a library is enhanced by two other features of SAIL. The first is the ability of SAIL to convert its library to and from a coded format which can be used on any computer system. The second is the fact that SAIL does its own free-format processing of INPUT which is independent of the computer.

A final noteworthy feature of SAIL is its character string substitutions. In addition to the library modifying directive *C, which makes substitutions in a single SAIL record, there is the EDIT request which will replace a given character string by another wherever it occurs on the library. EDIT operates only during UPDATE and LIST runs. Dynamic substitution during NORMAL runs is

also available. While especially useful in changing several array dimensions between which calculable relations must exist, it can do more general substitutions than those in dimension declarations. These substitutions include not only setting loop indices, etc., to match the chosen dimensions, but any alphanumeric substitution needed for a particular task.

A related type of substitution that can occur in PROCs is called MACRO substitution. In this case, the *PROC directive which defines the name and beginning of the PROC has a list of character strings which can be changed wherever they are flagged within the body of the PROC. What they will become is determined by similar lists of character strings on the *INCLUDE directives which cause the PROC to be placed on the file produced by the NORMAL run.

2. FILES USED BY SAIL

The SAIL program uses ten files in processing its data. In this manual, these files are designated as OLD, NEW, SAIL, INPUT, INPUT2, OUTPUT, ERROR, PROCF, SAVE, CHANGE, and TEMPF. What SAIL uses each of these files for and what they are named on various computer systems are tabulated below. (For external structure of these files and substitutions that can be made, refer to the section on programming considerations for the particular computer system.)

SAIL FILES - NAMES AND USES

Name Used in Manual	Use	CDC Name	IBM Name	Honeywell File Code
OLD	Old Library	OLD	FT02F001	2
NEW	New Library	NEW	FT01F001	1
SAIL	Processed card-images	SAIL	FT08F00n	15,16,...
INPUT	Control and changes	INPUT	FT05F001	I*
INPUT2	Alternate control	INPUT2	FT09F004	9
OUTPUT	Lists and run status	OUTPUT	FT06F001	P*
ERROR	Nonfatal error messages	SSSSER	FT04F001	4
PROCF	Temporary storage-random	SSSSPR	FT10F001	10
TEMPF	Temporary storage-random	SSSSTM	FT11F001	11
SAVE	Temporary storage	SSSSTM	FT12F001	12
CHANGE	Temporary storage	SSSSCH	FT03F001	3

SECTION III

SAIL LANGUAGE USED IN GENERAL MODES

1. SAIL REQUESTS

Most functions of the SAIL program are controlled by the SAIL requests found in the input stream and the alternate input stream during NORMAL runs. All of the records (cards) on INPUT that are encountered before either one containing an asterisk in column one or an end-of-file make up the request area. Each record in the request area is assumed to contain request elements separated by one or more SAIL delimiters (blank, comma, or equal sign). Request elements may be either requests or parameters associated with requests.

Two structures that can occur within the request area merit mention. A request pair is two elements that must occur on the same card. The first identifies a parameter to be set and the second gives the value. A specification area is the set of request elements starting after a request to form or modify a list. Specification areas are terminated either by the end of the request area or by the occurrence of a request element that has been designated as the terminator.

The first request must be SAIL or the program will not process the input stream at all, but will execute a NORMAL run with the default OPTION and PROGRAM lists. This request identifies the SAIL input. On Control Data Corporation (CDC) systems, SAIL will search INPUT for its input. The remaining requests fall into three classes. Except as specifically noted, there is no order dependence to the requests that either stand alone, are the first members of request pairs, require a specification area, or occur within a specification area. There is no limitation as to the number of request elements on a record. However, each request element must be completed on an input record (card).

a. Primary Requests

(1) File Control Requests

The first class is composed of those requests which affect the status of either OLD or NEW or both. Any of these requests found before the

first occurrence of a request from another class (or an unrecognizable request) affects OLD, while any later occurrence affects NEW. These requests are:

SYSTEM = name. Name is the system identifier for an old SYSTEM If OLD is being affected, name is compared with the SYSTEM name on OLD and processing continues only if the names match. If this request is not made for OLD or is ignored, no comparison is made. If the request affects NEW, name is placed in the name field of the header record of NEW. (This form of the request has meaning only for the GENERATE, UPDATE, and COPY modes.)

On CDC systems, the SYSTEM request for OLD has the additional feature that, if it is specified, it is used as the permanent file name for internal attaches. These internal attaches are performed only if OLD does not already exist as a recognizable OLD file and the requests TAPE or LOCAL have not been specified for OLD. For further discussion of the internal permanent file attaches, see the section on CDC programming considerations.

VERSION = nn. nn is the VERSION number existing on OLD or to be put on NEW. If the old VERSION number is specified it is compared to the VERSION number on OLD and processing stops if they are unequal. If no VERSION number is specified for OLD, no comparison is made and processing continues. If the request is for the VERSION number of NEW, it is stored in the header of NEW. (This form of the request has meaning only in the GENERATE, UPDATE, or COPY modes.) If the new VERSION number is not specified, then it is set to either one plus the old VERSION number or to one if the mode is GENERATE.

On CDC systems, if the old VERSION number is specified, it is used as the cycle number for an internal permanent file attach. If the old VERSION number is not specified, any attaches will fetch the last cycle.

CONVERT. This request indicates that coded format conversion is needed for either OLD or NEW. If the request specifies OLD, the mode is forced to be COPY (see the mode selection requests). A coded NEW file can be produced only during a GENERATE, UPDATE, or COPY run.

TAPE = vsn. This request has meaning only on CDC systems. indicates that OLD or NEW is found on a tape to be requested from the operating system by SAIL. vsn is the VSN and density of the tape. If the tape has default density the element may contain only the VSN. If the density is to be requested and the VSN is shorter than six characters, the element has the form xxx-dd, where xxx is the VSN and dd is the density. If the density is to be

requested for a VSN of six characters, the request element has the form xxxxxxdd, where xxxxxx is the VSN and dd is the density.

LOCAL. This request has meaning only on CDC systems. It suppresses the automatic internal permanent file attach is specified for OLD and suppresses the allocation to a permanent file device if specified for NEW.

(2) Mode Requests

The next class of requests are those that select SAIL operation modes. These requests are UPDATE, LIST, GENERATE, COPY, SCAN, and PUNCH. Since some of the modes can affect the meaning of the requests in the final class, it is recommended that any mode requests be placed immediately after all requests affecting OLD. There is an exception in the case of GENERATE which must be the last request on the record which immediately precedes the first card image record to be placed on NEW. If no mode is selected, the default is NORMAL. It should be noted that with the exception of GENERATE, which takes effect immediately regardless of whether other modes have been requested; if more than one mode request is found, only the first is honored. The SAIL operational modes are:

NORMAL. In this mode, requested changes are made to OLD and the result is passed to the SAIL data management executive which processes the executive directives and writes the final form of the records on the file SAIL. The actions of the executive directives on the SAIL records depend on the OPTIONs in effect at the time. This is the only mode that will produce more than one block on SAIL, and then only if it is requested to do so by the executive directive *E. (For a further discussion of the operation of the executive, see the section on the SAIL executive directives.)

UPDATE. In this mode, changes to OLD supplied in INPUT are made permanent and new SAIL records are assigned record identifiers. Changes may also be made to the default OPTION list and the default PROGRAM list. (See the requests OPTIONs and PROGRAM for further discussion of the default lists.) Then, NEW is created with an updated VERSION number.

LIST. This mode causes either all SAIL records on the library or just those in selected PROGRAMs to be listed with their identifiers. The listing may be full or directory (see the discussion of *DIR in the section on SAIL directives). Any cards added by changes found on INPUT are listed with identifiers that indicate that they are new records.

COPY. Here OLD is copied to NEW. Only the default OPTION list and the default PROGRAM list may be changed. Frequently, this mode is used to convert the format of a library file. The two possible formats are (1) the coded one recognized by all machines, and (2) the local version of packed format.

GENERATE. When the request for this mode is encountered, the remainder of the records on INPUT are read and placed on NEW. This is the way to create a library which can be processed by SAIL.

PUNCH (or SOURCE). This mode writes all of the SAIL records on the library or in selected PROGRAMs onto SAIL. Insertions and deletions on INPUT will be processed, but none of the resulting records will be processed further by SAIL. They will simply be written as found on OLD or in the change sets on INPUT.

SCAN. In this mode, SAIL searches for selected character strings in the SAIL records of OLD. Each record in the request area which follows the SCAN request and precedes a record whose first request element is ENDSCAN, contains a character string to be located. The first nonblank character on the record is the string delimiter. The character string is found between the first and second occurrences of the delimiter. Any records in the selected PROGRAMs which contain one of the character strings are written out along with their identifiers on OUTPUT.

(3) Function Requests

The requests in the final class control many SAIL operations. Some have meaning only for certain modes, while others have different meanings for different modes. The requests in this class are:

OPTIONs. During a NORMAL run this request allows the user to modify values on, or add new OPTIONs to, the default OPTION list which is on OLD. It is used to create new default OPTION lists on GENERATE, UPDATE, or COPY runs. During a LIST, PUNCH, or SCAN run, this request will change the listing of the OPTIONs but will have no other effect.

This request uses a specification area as described at the beginning of this section. The terminating request is ENDOPTIONS. Request elements in the specification area consist of OPTION names, each of which may be followed by a request element specifying the OPTION value as a nonnegative

integer. If the element following an OPTION name is not on the same card or does not contain a number, the OPTION value is set to zero and the new element is treated as a new OPTION name. If a specified OPTION name exists in the OPTION list, only the new value is placed in the list. If not, the name is added to the list as well. There is a limit of eight characters for an OPTION name. Any name which is longer is shortened to the first eight characters.

If the request "AFTER" is encountered in this specification area then a table will be constructed. (The symbol " is part of the request. It is an 8-4 multipunch, the alpha equal punch on the 026 keypunch which prints as a ≠ on CDC systems.) The table will be placed in the OPTION list after the OPTION name found in the request element following AFTER. All OPTIONs following this name are placed in the OPTION list with values determined as above in the order in which they are found in INPUT until either the specification area ends or two consecutive occurrences of the same name are encountered. The OPTIONs which originally followed the head of the table are pushed down. As OPTIONs are added to the table, they are removed from any other positions in the OPTION list where they occur, including prior positions in the table.

DELOPTIONS. This request deletes OPTIONs and their values from the OPTION list. Like the OPTIONs request, its specification area can be terminated by the request ENDOPTIONS.

LINENO. This request causes SAIL to write the SAIL record identifier (line number) of each card-image in columns 73 to 80 (in place of the date) on NORMAL runs. If the entire identifier will not fit, omission of a non-zero card number is indicated by a trailing plus sign and truncation of the most significant digits of the block number by a leading plus sign.

COL. The number which follows this request defines the number of columns which are to be scanned for any free-field information on INPUT. The default value is 72 unless it is reset for a particular installation.

BCD. This request has meaning only on IBM systems. It indicates that SAIL must convert the information on the following records from 026 key-punch format (BCD) to 029 keypunch format (EBCDIC).

EBCDIC. This request has meaning only on IBM systems. It negates the conversion initiated by the BCD request.

NOLIST. This request suppresses the listing of NEW which normally occurs during an UPDATE run.

NOAST. This request affects only listings of SAIL records. If specified, it suppresses the asterisks which are normally printed in the listing next to the card images which were changed by the last UPDATE run.

LINES. The number following this request defines the number of lines to be printed on each page of a listing. The default is normally 60 but may be set by the installation.

COLMARK. If this request is specified, column numbers are printed at the top and bottom of each page during a listing to allow easy use of the column change feature.

DIRECTORY. This request sets the global list type to directory instead of full. (See the discussion of *DIR in the section on SAIL directives.)

PROGRAM. This request allows the user to make PROGRAM selections. The specification area following this request can be terminated by the request ENPROGRAM.

On NORMAL runs, only the PROGRAMs named in the specification area will be processed by SAIL. If this request is not found or none of the PROGRAMs specified exist on the library, SAIL will process the PROGRAMs in the default PROGRAM table. An empty default PROGRAM table is treated as a selection of the entire library.

On UPDATE, GENERATE, or COPY runs, the PROGRAMs named in the specification area are used to form the new default PROGRAM table. If none of the names on the new library, SAIL creates an empty default PROGRAM table.

On PUNCH or SCAN runs, only the PROGRAMs found in the specification area are processed. If no PROGRAMs are requested (either there were no known PROGRAMs in the list or the PROGRAM request was not specified), the entire library is processed. In these modes the PROGRAM name PROLOGUE is used to specify the prologue.

On LIST runs the specification area identifies those PROGRAMs that are to be listed. If the PROGRAM request is in use, the name PROLOGUE must be in the specification area if the prologue is to be listed. If a PROGRAM name is followed by "DIR" or "NODIR" (the symbol " is part of the name), then the listing

of that PROGRAM is directory or full, respectively. Otherwise, the listing is the global type. Finally, if the PROGRAM request was not specified then the entire library is listed. (See the DIRECTORY request for the global listing type.)

PROSNAME = name. This selects program name for special processing during a NORMAL run. If special processing is selected only lines requested in program 'name' will be written onto SAIL. All other programs that were selected will be scanned for PROCs which can be used by name.

EDIT. This request is honored only in LIST or UPDATE runs. It allows the user to make string substitutions. Each record in INPUT following the request and preceding a record beginning with the request ENDEDIT or the end of the request area, contains two character strings. The first nonblank character is the delimiter. The character string to be replaced is between the first and second occurrences of the delimiter. The string to be substituted is between the second and third occurrences of the delimiter. During LIST or UPDATE runs, the SAIL records are scanned for the requested character strings after insertions, deletions, and column changes are processed. Substitutions are made if the strings are found.

SEQ. This request resequencing of the SAIL record identifiers of the entire library on UPDATE runds. If this request is omitted, only the PROGRAMs specified by the SEQPROGRAM request will be resequenced and other new records will be assigned a card number with the block number of the preceding existing record in the file.

SEQPROGRAM. This request selects PROGRAMs to be resequenced during an UPDATE run. If SEQ has been specified, this request has no effect. The specification area for this request is the same as for the PROGRAM request. For this request, the PROLOGUE must be identified by name.

(4) Examples of Requests

For these examples we will assume that SAIL is processing a library whose SYSTEM and VERSION identifiers are EXAMPLE and 12, and that it contains the PROGRAMs: TRIAL, TEST, and DONE. The default OPTION list of the library contains: A=2, B=3, C=12, D=0, E=36, and its default PROGRAM table contains TEST.

```
SAIL SYSTEM EXAMPLE
```

This set of requests will cause SAIL to process PROGRAM TEST in a NORMAL mode and to use the OPTIONs as they appear in the default OPTION list.

```
SAIL SYSTEM EXP
or
SAIL SYSTEM EXAMPLE
VERSION 12
```

In either of these cases, SAIL will terminate abnormally, if in the first set, the SYSTEM identifier of the requested library is not EXP and in the second, the SYSTEM identifier is not EXAMPLE or the VERSION number requested is not 12.

```
SAIL
OPTIONS
CC=10,   B=2,   E=11
ENDOPTIONS
DELOPTIONS   C   ENDOPTIONS
PROGRAM TRIAL
```

In this case, SAIL will process the PROGRAM TRIAL in a NORMAL mode. C will be removed from the OPTION list and CC will be added with a value of 10. The values of B and E will be reset to 2 and 11, respectively.

```
SAIL LIST
```

With this request set, SAIL will list all the PROGRAMs that are on OLD.

```
SAIL LIST DIRECTORY
```

In this case, SAIL will list only those card-images on OLD which are found between a *DIR and a *EDIR directive.

```
SAIL LIST
PROGRAM TRIAL
DONE   "DIR"   ENDPROGRAM
or
```

```
SAIL LIST DIRECTORY
PROGRAM
DONE   TRIAL   "NODIR"
ENDPROGRAM
```

Either of these request sets will cause SAIL to list all of PROGRAM TRIAL and the directory portion(s) of PROGRAM DONE.

```
SAIL UPDATE
```

Here, the library on OLD will be updated by the changes that are found in the rest of the input in order to form a VERSION 13 of EXAMPLE which will be written on NEW. Neither the default OPTION list nor the default PROGRAM table will be changed by this update and none of the existing record.

```
SAIL UPDATE SEQ VERSION 15
OPTIONS CX=14
```

With this request set, the library identified by SYSTEM EXAMPLE on OLD will be updated to VERSION 15 on NEW. During the update, the record identifiers in all the PROGRAMs will be resequenced. The new library will have a different OPTION list to which CX will have been added.

```
SAIL UPDATE
SEQPROGRAM TEST   ENDPROGRAM
PROGRAM DONE
```

These requests will update the library on OLD to produce VERSION 13 of SYSTEM EXAMPLE, but only the record identifiers in PROGRAM TEST will be resequenced. The new version of the library will have a default PROGRAM table which contains only DONE.

```
SAIL CONVERT COPY
```

This set of requests instructs SAIL to convert a coded file on OLD to a packed file on NEW, and not to modify the OPTION list or the PROGRAM table.

```
SAIL COPY CONVERT
OPTIONS   XX=12, A=10
```

Here, SAIL will convert a packed OLD file to a coded NEW file. There will be an additional OPTION XX with a value of 12 on the default OPTION list of the resulting NEW and the value of A will be changed to 10.

```
SAIL COPY
PROGRAM NONE
```

Since NONE is not a PROGRAM on SYSTEM EXAMPLE, SAIL will simply copy OLD to NEW, with an empty default PROGRAM table on NEW. The default on a NORMAL run for the resulting library would be all the existing PROGRAMs.

```
SAIL SYSTEM   SYSNEW   VERSION 2
GENERATE   PROGRAM TEST
OPTIONS XX=2
```

These requests instruct SAIL to generate a new library called SYSNEW beginning with VERSION 2. The PROGRAM request will be completely ignored. The record that looks like an OPTIONS request will be treated as data. It will not be processed as a request either on this run or on any subsequent run.

```
SAIL SCAN
-XXYY-
$ZQ VW$
```

Here, SAIL would scan the PROGRAMs on OLD for records which contain the character strings XXYY and ZQVW.

b. Alternate Requests

SAIL reads file INPUT2 for additional requests which effect option selection, and program selection during a NORMAL run. On the alternate input file there is one request per line. The requests are:

PROGRAM name. This selects name as a program to be processed. When this request is first encountered it overrules any program selection from INPUT.

PROSNAME name. This selects name as the program for special processing. It acts the same as PROGRAM in addition to setting special processing.

OPTION name = value. The request set as OPTION name to value.

2. SAIL RECORD IDENTIFIERS

As was mentioned in the discussion of current SAIL structure (Section 2.1), SAIL record identifiers contain a block and a card number. As parameters of the SAIL directives, they appear in the form bbbbb.ccc, where bbbb is the block number and ccc is the card number. If the card number is zero, neither ccc nor the period need be given. Actually, the user need only remember that the proper

record identifiers appear in the SAIL listing. This form of identifier allows libraries to be updated without resequencing all the line numbers. Thus, while a library is in a development and testing phase, change decks need only minor alterations in spite of updates.

3. GENERAL SAIL DIRECTIVES

There are thirteen general SAIL directives. Unlike the SAIL executive directives which operate only in NORMAL runs, these take effect during executions in any mode. Each directive is identified by an asterisk in column one of the input record, with the directive verb beginning in column two. The verb must be separated from any parameters by at least one of the SAIL delimiters (blank, comma, or equal sign). One of the directives is used to define PROGRAMs in the SAIL system, four are used to modify OLD, five affect the structure of the input stream, and three control the SAIL listing. During NORMAL runs, none of these are written to SAIL.

a. SAIL PROGRAM Definition

*B pname

When this directive is encountered during a GENERATE or UPDATE run, the name pname is placed in the PROGRAM table in the header of NEW. On subsequent runs when this file is reidentified as OLD, the records found after this directive and before the next *B directive are considered to compose one PROGRAM. The PROGRAM may then be selected for processing by referring to pname in the specification area of the PROGRAM request. Note: The PROGRAM name is added to the PROGRAM table only during GENERATE and update runs. Thus, PROGRAM name definition directives added during any other run are ignored. When this directive is encountered during a SAIL listing it is forced to be at the top of the page.

b. SAIL Library Modifiers

All of these directives modify OLD and appear only on INPUT. Only one modification directive should specify any SAIL record (this includes records in the range of the delete directive). If, however, more than one directive specifies the same record, then all but one are ignored and a warning message is printed indicating which directives were not used.

(1) Record Insertions

*A nl

or

*I nl

Either of these directives inserts records after the record in OLD with the identifier nl. All the records between this directive and the next modification directive or end-of-file are inserted.

(2) Record Deletions

*D nl n2

or

*C nl n2

These directives are used to delete all of the SAIL records on OLD from the one with the identifier nl through the one with the identifier n2. If only nl is specified, then only one record is deleted. If both are specified, they must be separated by at least one SAIL delimiter. All the records between this directive and the next modification directive or end-of-file are inserted at this point.

(3) Record Modification

*C nl (c1,c2,cn1,cn2)

This directive replaces columns c1 through c2 of the record identified by nl with columns cn1 through cn2 of the record which follows this directive. If the number of characters in the replacement is not equal to the number removed from the record, the characters in the original record to the right of the change are shifted. Should nonblank characters be shifted off the end of the record, a warning message is printed. The column change parameter field is defined by the parentheses and must be separated from the record identifier by at least one of the SAIL delimiters. The column numbers may be separated by an SAIL delimiter(s); c1 must be specified. If c2 is not specified it is set to 72. The defaults for cn1 and cn2 are 1 and cn1+c2-c1, respectively. If cn1 is zero then no characters replace those removed from the record and the record containing the characters to be substituted may be omitted. All records following this directive, except for the one holding the substitution characters, are ignored until another modification directive is encountered.

c. SAIL Input Modifiers

(1) Record Copying

*M n1 n2

This directive copies records from OLD beginning with the one identified by n1 to the one identified by n2. The records copied effectively replace this directive in the input stream. If n2 is not specified, only the one record is copied.

(2) Input Record Conversion (IBM)

*BCD
and
*EBCDIC

These directives affect conversion of subsequent records in the input stream in the same way as the BCD and EBCDIC requests.

(3) Text Data Control

*TXT
and
*ETXT

These directives identify the beginning and end of a block of input records which are not processed either by SAIL or the SAIL executive. The included records are treated as ordinary data even if they have the format of a SAIL directive. (This also applies to additional occurrences of *TXT before *ETXT.) Because they also act as executive directives, they appear as data on OLD and NEW.

d. SAIL Listing Control

This set of directives gives the user control of SAIL listings. In order to maintain their function on later runs, they are stored as data on NEW.

(1) Paging and Summary Control

*p sname

When this directive is encountered as a record to be listed, SAIL forces it to begin a new page. In addition, if the name sname is present, it is written along with the record identifier for the directive at the end of the listing on a summary table.

(2) Directory Listing Control

*DIR
and
*EDIR

These directives identify the beginning and end of a block of records to be listed if the listing type is directory. During a directory listing of the PROGRAM where they occur, only those blocks of records so identified will be listed. During a full listing, all records in the PROGRAM are listed.

e. Examples of SAIL Directives

A typical set of modification directives are:

```
*A 10000
*P   TEST
Card1
*M 942,966
Card 2
*D 12642.1   12713
Card3
Card4
Card5
*M   15462   15477
*D 942,966
*D 16111, 16452
*TXT
*A  TEXT
*D  TEXT
*C  10   (TEXT)
*ETXT
*C 1214 (12,14,2,3)
ABCDE
```

The first directive instructs SAIL to insert records containing *P TEST and Card1, records 942 through 966 from OLD, and a record containing Card2 after record 10000. Then SAIL is to delete records 12642.1 through 12713 and insert three records containing Card3, Card4, and Card5 followed by

records 15462 through 15477 in their place. Records 942 through 966 are to be deleted without making insertions. (Note: these are the same records copied during the first insertion.) SAIL is then instructed to delete records 16111 through 16452 and insert as text the records containing *A TEXT, *D TEXT and *C TEXT. (Note: these records would have been mistaken as directives had they not been between *TXT and *ETXT.) The last directive is an instruction to replace columns 12 through 14 on record 1214 with the characters BC.

4. SAIL COMMENTS

Any record, either on OLD or in a change set, which has an equal sign (=) in column one is a SAIL comment. During NORMAL runs these records are not sent to the SAIL executive processor.

SECTION IV

SAIL EXECUTIVE PROCESSOR LANGUAGE

1. EXECUTIVE DIRECTIVE FIELDS

In general, SAIL executive directives contain three fields: the verb field, the name field, and the operand field. Each field contains one or more elements which are separated by SAIL delimiters (blank, comma, and equal sign).

a. Verb Field

The verb field contains one element which must begin in column one of the record, and must be *PROC, *INCLUDE, *ENDPROC, *KEEPTO, *SKIPTO, *LABEL, *DEFL, *DEFN, *AUTO, *MAN, or *E. Because all general SAIL directives except *B, *TXT, and *ETXT are stripped from the modified library before the executive processor operates, any card not containing one of these three or a SAIL executive directive will be treated as data.

b. Name Field

The name field may contain a simple name, a macro name, or a table name. The simple name form can be used for all directives that need a name (*AUTO, *MAN, *ENDPROC and *E do not). It contains one element called the name. The macro name contains a parenthesized subfield. The left parenthesis must be part of the first element of the name field. In this case, the name is the portion of the first element which precedes the left parenthesis. The subfield within the parentheses may contain up to nine elements called parameters. (For a discussion of the macro parameters see the section in the macro processor.) The macro name field may be used in *PROC and *INCLUDE directives. The table name field may be used in *INCLUDE directives and contains three or four elements, the first and last of which are the character $. (Refer to the discussion of the inclusion of PROCs for a description of the table name.)

c. Operand Field

The operand field contains either a numeric or logical expression. It may be left blank to make the directive unconditional.

(1) Numeric Operands

A numeric expression is one element long and contains integers, nonnumeric OPTION names (replaced by the corresponding OPTION value) and the arithmetic operators + (addition), - (subtraction), and / (division). The numeric operand field is used only on the *DEFN directive. Its evaluation is described in the section on OPTION redefinition.

(2) Logical Operands

A logical expression contains one or more logical units which are connected using the logical operators AND, OR, and NOT. Each logical operator must be an element of the field. The operand field is processed from left to right so each logical operator acts on the values of the total expression to its left and the first unit to its right. Higher level units may be defined by parentheses to modify the order of processing. The parentheses may be either separate elements or concatenated onto logical unit elements (but not onto operators). Each unit has a TRUE or FALSE value for use in the logical operations.

Each lowest level logical unit is composed of one, two, or three elements. The one element units are of the form xxx or xxxnn, where nn is a number and xxx is the name of an OPTION. (SAIL will terminate abnormally if the name is not found in the OPTION list.) If nn is not specified, the unit is TRUE if the value of the OPTION xxx is greater than zero and FALSE if it is equal to zero. If nn is specified, the unit is TRUE only if the value of the OPTION is equal to the number.

The two element logical units occur in two forms. In the first, the first element is an OPTION name and the second is of the form yynn, where yy is one of the comparative operators EQ, NE, LT, LE, GT, or GE and nn is a number. The unit is TRUE if the comparison is satisfied and FALSE otherwise. If the OPTION name is not in the OPTION table, SAIL will terminate abnormally. In the second form, "DEF" is the first element (the symbol " is part of the element) and an OPTION name is the second one. The unit is TRUE only if the OPTION is currently in the OPTION list.

The three element logical units contain an OPTION name in the first element, a comparative operation in the second, and a number or an OPTION name in the third. The result of the comparison of the value of the OPTION in

the first element and the number or value of the OPTION in the third determines the logical value of the unit.

The user should refer to the discussion of each directive for information on how the operand field affects it. If there is no operand field present on an executive directive which can have a logical expression, SAIL acts as if a TRUE result were obtained (this is considered an unconditional directive.

(3) Examples of Operands

For these examples, we assume the OPTION list contains: A=1, B=2, C=42, D=0, Q=16, BVD=22, and XXX=45. The first five examples are assumed to occur in the operand fields of *DEFN directives, so they are interpreted as numberic operands in accordance with the rules explained below in the discussion of that directive.

A+B*C

This has a value of 126. (3 times 42)

A-B+Q

This operand has a value of 15.

A-C+2

Since the computation yields a negative result, the value will be set to zero.

B

As a numeric operand, this field has a value of 2.

C1O

Because the OPTION name-test value concatenation has no meaning in a numeric operand field, c1O will be treated as the OPTION name. No such OPTION appears in the list, so the *DEFN will be ignored.

The remaining examples are assumed to occur in the operand.

A-C+2

This looks like a numeric operand, but it has been assumed to occur in a place where it will be interpreted as a logical operand. There is no OPTION names A-C+ in the list, so SAIL will terminate abnormally.

B

As a logical operand, the result is TRUE since B is greater than zero.

```
C10
```

The result will be FALSE since C is not equal to 10.

```
"DEF"    X
```

This logical operand is FALSE since X is not in the OPTION list.

```
"DEF"    Q
```

This field will be TRUE since Q is in the OPTION list.

```
Q   GE5
```

Because Q is greater than or equal to 5, this field will be TRUE.

```
Q   EQ   C
```

The result from this field is FALSE since Q is not equal to C.

The next six examples of compound operands will have a result of TRUE.

```
A   OR   BO
A   AND   B
A   AND   D   OR   Q   GE12
A   OR   (D   AND   Q)
B3   OR   (D   OR   (C42   AND   BVD))
"DEF"   A   OR   (B   AND   "DEF"   X)
```

while these will be evaluated as FALSE:

```
A   AND   BO
A   OR   Q   GE12   AND   D
A   AND   (A2   OR   B1)
XXX   LE100   OR DO   AND   "DEF"   X
```

2. SAIL EXECUTIVE DIRECTIVES

These directives control the creation and inclusion of blocks of records called PROCs, skipping of records during processing, further definitions of OPTIONs, dynamic substitution, and termination of blocks on SAIL. They are stored in the card portion of the SAIL records on NEW so they can be used during later runs. They will appear in listings of the library, but will only take effect during NORMAL runs. After being processed, they do not appear on SAIL.

a. PROC Creation

a PROC is a collection of SAIL records that can be included at any later point(s) on SAIL. If the executive directives *DEFN, *DEFL, *KEEPTO, *SKIPTO, *LABEL, *AUTO, or *MAN occur within the body of the PROC, they are

processed at creation, not at inclusion. *INCLUDE and *E are implemented when the PROC is included. PROCs are identified and created by the directives:

*PROC name operand

and

*ENDPROC

If the logical expression in the operand field is TRUE, the records between these directives are saved as a PROC named name. Note: *PROC and *ENDPROC directives are paired. Therefore, SAIL will abort if *PROC occurs in a block identified as a proc (between a *PROC and *ENDPROC directive), or if *ENDPROC occurs without a paired *PROC.

b. PROC Inclusion

Inclusion of PROCs is initiated by the directive:

*INCLUDE name operand

This directive causes the block of records in the PROC identified by the name field to be included if the result of the operand is TRUE. PROCs may be included within other PROCs, but the level of inclusions is limited to eight.

SAIL requires that a PROC be created before the request for its inclusion is honored. For *INCLUDEs outside of *PROCs, this is at the time the directive is processed. If the *INCLUDE directive is located in a block defined as another *PROC, it is honored when the PROC that contains it is included.

This directive can have a table name field. The field is defined by a dollar sign ($) being the beginning and ending elements. The rest of the field may be made up of one or two elements called table elements. The first is of the form xxxnn, where xxx is an OPTION name and nn is a number. The OPTION list is searched for the name xxx and the corresponding value NV is obtained. Then the values of the next NV OPTIONs in the list (table) are compared with nn. If the name xxx is not found, SAIL will terminate abnormally. If none of the OPTION values searched are equal to nn, then the *INCLUDE is not honored. However, if an OPTION is found whose value is equal to nn, the characters that make up that OPTION name become the first part of the PROC name. If there is a second table element, it is concatenated on the end to complete construction of the name.

c. Record Processing Control

Control over which records within each PROGRAM are processed by the SAIL executive is established by the directives:

```
*SKIPTO    name    operand
*KEEPTO    name    operand
```

and

```
*LABEL     name
```

The first two conditionally initiate skipping of the records identified by the name field. SAIL starts skipping after *SKIPTO if the operand field is TRUE and after *KEEPTO if it is FALSE. Otherwise, processing continues with the next card. If the name is of the form *nn, the next nn records are skipped. (SAIL comment, *P, *ETXT, *DIR, and *EDIR are not counted.) If the name element is any other character string, the records are skipped until a *LABEL directive is encountered with either no name field or a name identical to the one on the *SKIPTO or *KEEPTO directive. *LABEL cards are ignored if skipping is not in progress or if the names do not match. Skipping by the SAIL executive is automatically terminated at the end of a PROGRAM.

Since these directives can cause SAIL to skip SAIL executive directives (including other *SKIPTO and *KEEPTO directives), care should be taken that the records skipped do not unbalance the required pairing between *PROC and *ENPROC directives.

d. OPTION Definition

New OPTIONs may be added or existing OPTION values modified by the use of two SAIL executive directives. If an OPTION is defined in either of these ways, SAIL writes an informative message on OUTPUT identifying the OPTION name and its new value. These values only apply to occurrences of the OPTION name that physically follow the defining directive and precede any redefinition of the same OPTION. If the OPTION identified in the name field is in the OPTION list, these directives redefine the value. If not, the OPTION is added to the end of the list.

An OPTION may be defined as the result of a logical expression by the directive:

```
*DEFL      name    operand
```

The logical value of the operand determines whether the OPTION, identified in the name field, is set to one (TRUE) or zero (FALSE). Otherwise, it is added to the OPTION list.

An OPTION may be defined by an arithmetic expression by using the directive:

*DEFN name operand

This is the only directive that has an arithmetic expression as an operand field. The expression is evaluated in strict left to right sequence. Parentheses cannot be used to define subexpressions and no hierarchy is recognized among the operators. If an OPTION name occurring in the expression is not on the OPTION list, a warning message is printed and the directive is ignored. If the final result is negative, it is set to zero. Otherwise, the result of the expression is used as the new value of the OPTION identified in the name field.

e. Value Substitution Control

Dynamic value substitution is not performed for every SAIL record. This processing occurs only for those records which have a dollar sign ($) in column one or which follow the directive:

*AUTO

and which precede the directive:

*MAN

The operation of dynamic substitution is described in paragraph 3.

f. Block Termination Control

*E

This directive will cause the current block being written on SAIL to be terminated. If it occurs in a PROC, it takes effect when the record containing the directive would have been written on SAIL had it been a data record.

g. Examples of Executive Directives

If we assume that the OPTION list is the same as for the operand examples, then typical executive directives would be:

```
*PROC    ABC
Card1
Card2
Card3
*ENDPROC
*PROC    XYZ    A    OR    B0
Card4
Card5
*INCLUDE    ABC
*ENDPROC
*PROC    QRS    A    AND    B0
Card6
Card7
*ENDPROC
```

These directives will cause two PROCs to be created. The PROC named ABC will be created unconditionally and will contain records Card1, Card2, and Card3. XYZ will be created since the operand is TRUE. It will contain Card4, Card5, and all the records in PROC ABC. QRS will not be created since the operand is FALSE. Records Card6 and Card7 will be ignored.

```
*INCLUDE    VVV
```

The records in PROC VVV will be included unconditionally at the position where this directive occurs.

```
*INCLUDE    VXZ    Q    GE10
```

This directive causes the records in VXZ to be included because the operand is TRUE.

```
*INCLUDE    VVX    D
```

In this case, no records will be included because the operand is FALSE.

```
*KEEPTO    *1    A
CardA
*SKIPTO    *2    B
CardB
CardC
*SKIPTO    END    D
CardD
```

```
CardE
*KEEPTO    END    BVD    LT5
CardF
*LABEL     END
```

This set of directives will cause the records CardA, CardD and CardE to be processed, and records CardB, CardC, and CardF to be skipped.

```
*DEFL     YYY    A     NE1    OR    B3
*DEFL     ZZZ    A     AND    Q     GT5
*DEFN     BBB    Q*C/BVD
*DEFN     XXX    12
```

Here, YYY, ZZZ, and BBB will be added to the OPTION list with values 0, 1, and 30, respectively. The value of XXX will be redefined as 12.

```
*KEEPTO     *1    Q
*E
```

These directives will cause the current block being written on SAIL to be terminated.

If in addition to the OPTIONs mentioned, the OPTION list contained the table: TAB=4, TIM=3, QVZ=4, ZZX=2, and VWW=1, then

```
*INCLUDE    $    TAB1    CCX    $
```

would cause the records contained in the PROC named VWWCCX to be included at this point, while

```
*INCLUDE    $    TAB6    XX    $
*INCLUDE    $    TAB4    XY    $    QO
*INCLUDE    $    TAB2    $    A    AND    B
```

would cause only PROC ZZX to be included. The first *INCLUDE would be ignored because there is no OPTION following TAB which has the value 6. The second is ignored because the operand is FALSE.

3. DYNAMIC SUBSTITUTION

a. Description of Processor

The dynamic substitution processor modifies the contents of a record to values depending on the OPTIONs currently in the list. An arithmetic expression which contains OPTION names and/or integers separated by the

arithmetic operators + (addition), - (subtraction), * (multiplication), and / (division) can be processed. Each expression must be enclosed by its own pair of the characters produced by a 0-8-5 multipunch (printed as ¬ on CDC systems and as _ on IBM systems and designated as the special delimiter in the following discussion). The expression is processed from left to right in the same manner as the numeric operand field (see the description of the *DEFN directive). If any of the OPTION names in the expression are not found in the OPTION list, the executive processor writes an error message and sets the fatal termination flag. Otherwise, the encoded result of the expression replaces that expression in line (delimiters are also removed). SAIL will terminate abnormally if the dynamic substitution processor finds an odd numer of occurrences of this delimiter on a card.

The user may request SAIL to make more general character substitutions. The substitution identifier is enclosed by the same delimiters and is of the form $xxxnn (the $ is required), where xxx is an OPTION name and nn is an integer. The OPTION list is assumed to contain the name xxx followed by at least as many OPTIONs as the value of OPTION xxx. This identifier and its delimiters are replaced by the first OPTION name from the group mentioned whose value is equal to nn. If either xxx or a value equal to nn is not found, the identifier is replaced in the record after the delimiters and $ are removed.

Two other delimiters are recognized by SAIL as defining the beginning of a substitution field. These are left parenthesis and slash. Right parenthesis and slash are the respective terminators. Subfields may be defined by commas. If the special delimiter appears, it terminates the current subfield and the processing described in the two paragraphs above occurs. A new subfield begins after the special substitution field unless the appropriate terminator (right parenthesis or slash) appears immediately. Since the occurrence of a left parenthesis or a slash begins a specific field definition for this form of processing, one of these types cannot be inside the field defined by the other. The user should be certain that the characters which begin and end these fields are properly paired. The substitution is limited to the replacement of a single OPTION name by its value. If the user wishes to use an expression to define the substitution field, he must use the special field form. During substitution in fields delimited by parentheses or slashes, none of the delimiters (except those defining special substitution subfields) are removed. This form allows values to be substitued in FORTRAN DIMENSION and DATA statements. It is

the user's responsibility to insure that records are not identified for substitution processing if they contain variables identical to OPTION names enclosed by parentheses or slashes.

The next blank field following any substitution area will be either expanded or contracted, if possible, to place the next nonblank character in the same column that it occupied prior to substituion. This realignment allows the SAIL user to maintain tabulated columns in COMMON and DATA statements, since the next blank may be inside an H field. This problem is easily overcome by inserting an extra blank before the H field.

b. Examples of Dynamic Substitution Fields

For these examples, we will assume that the OPTION list is VV=2, QT=3, Z2X=4, QVX=3, C1B=1, C2B=4, C3X=7. When the SAIL executive processes the following records:

```
$    DIMENSION A(VV,QT),C(C3X),BX(_C2B*C3X_)
$    DATA A(2,C1B) /_C1B*2+1_/
     DIMENSION Q(Z2X), AX(_VV*QT_)
*AUTO
     CALL_$QVX4_(X)
     CALL_$QVX10_(X)
     _Z2X+1*QT_
     _A*B+C_
*MAN
```

it will produce:

```
     DIMENSION A(2,3),C(7),BX(28)
     DATA A (2,1)    /3/
     DIMENSION Q(Z2X),AX(_VV*QT_)
     CALL C2B(X)
     CALL QV10(X)
     45
     A*B+C    *FATAL ERROR*
```

4. SAIL MACRO PROCS

a. Description of Macro Processor

Macro PROCs are identified by parenthesized subfields in the name field of the *PROC and *INCLUDE directives. When a macro PROC is being included, each

record of the PROC is scanned for a field enclosed by a unique pair of the characters ". (This character is the alpha shift of the equal sign on the 026 keypunch and is printed as ≠ on CDC systems.) If the field within the delimiters matches one of the subfield parameters on the *PROC directive, the corresponding parameter on the *INCLUDE directive replaces the field and its delimiters if it was specified as nonblank. If an *INCLUDE directive for a macro proc is found in another macro PROC, then each time that include is honored, its parameters are scanned in the same way as other records and substitutions are made. After any scan, if the field within the delimeters does not contain a parameter that appeared on the PROC directive or there is no corresponding parameter on the *INCLUDE directive, then the delimiters are removed and the field is returned to the record. Just as in the dynamic substitution, the first blank field following each macro substitution is used to realign the following nonblank character. (See the discussion at the end of the description of the dynamic substitution processor.) In the case of macro PROCs, the realignment allows correct placement of the character following a substituted FORTRAN statement number, as well as tabulated COMMON and DATA statements.

b. Example of Macro PROC Usage

If we have the following records:

```
*PROC    MAC1(A,B,C)
     "C"Y="A"+"B"X
C    "A"B""C"
*ENDPROC
*PROC    NORM(11,I,N,X,Y)
     "Y" = 0
     DO "11" "I" = 1, "N"
"11" "Y" = "Y" + "X"("I")*"X"("I")
     "Z" = SQRT("Y")
*INCLUDE MAC1("Y",D"Z"D,Q)
*ENDPROC
*INCLUDE MAC1(VV,UU,BV)
*INCLUDE MAC1(XX)
*INCLUDE NORM(3,J,NXTOT,F,FBAR)
*INCLUDE NORM(2015,M,P)
```

the SAIL executive would produce:

```
      BVY=VV+UUX
C     VVUUBV
      CY=XX+BX
C     XXBC
      FBAR = 0
      DO 3    J   = 1,NXTOT
3     FBAR = FBAR + F(J)*F(J)
      Z   = SQRT (FBAR)
      QY=FBAR+DZDY
C     FBARDZDQ
      Y   = 0
      DO 2015 I   1,M
2015  Y   =  Y   + P(I)*P(I)
      Z   =  SQRT(Y)
      QY=Y+DZDX
C     YDZDQ
```

SECTION V

CONSIDERATIONS ABOUT SAIL USAGE

1. LIBRARY GENERATION AND EARLY DEVELOPMENT

A user could convert an existing card-image file into a SAIL library by a GENERATE run. The easiest way to proceed is to simply place a card containing

```
SAIL SYSTEM=name GENERATE
```

for a request area at the beginning of the file. If the file has been maintained by some other editing program in which special blocks that will become PROCs have already been identified, it would save time if these blocks were kept separate, especially if the equivalent of the *INCLUDE directive were kept in the file input to SAIL.

Having created a SAIL library of the file, the user could then examine it for sets of consecutive cards that appear in more than one place throughout the file and consider making them into PROCs. Such PROCs could then be created and the original sets replaced by *INCLUDE directives on one or more runs in UPDATE mode. This idea would apply especially to such sets as COMMON blocks in FORTRAN which are subject change as the routines develop and which must be the same in each routine. The user could also find array dimensions, loop indices, and other parameters for which it would be advantageous to use dynamic string substitution. He could modify and flag cards containing this sort of data and define OPTIONS to be used in the substitution during UPDATE runs. As he developed the structure of the library, the user would always (barring errors) be able to produce the original file on NORMAL runs both for production and for testing of the developing library.

At some point in the development, the need for an alternate version might arise. Without deleting any good lines from the original library, the user could insert new cards for the alternate along with directives to skip the unneeded cards on NORMAL runs meant to produce either file. These directives might be controlled by a new OPTION which identified the version. For instance, READ statements of FORTRAN source code vary in format from one computer to another. In particular, cards like

```
READ (8,100) A
If (EOF(8)  .NE.0.)   GO TO 200
```

where one of the OPTIONs CDC, HONW, or IBM has been set nonzero (TRUE) and the other two zero (FALSE). The same sort of structure, perhaps involving more cards and employing the *LABEL directive, could be used to add a new capability that was only needed for some runs. While the identical file could be produced on a NORMAL run using deletions and insertions, the necessary change deck could soon become cumbersome and the purpose of using SAIL defeated. If programming errors were found in a part of the coding common to both versions, correcting the library would correct both versions so two different change decks would be unnecessary.

As a (somewhat overdone) example of what a routine in a SAIL library could become, consider the following. Sixteen different versions of this routine are contained in this library representing all possible choices of the OPTIONs COMPLEX, GAUSS, DIM (odd or even - the variations due to simple dimension changes are not counted in the sixteen), CDC, IBM, and HONW. The routine is supposed to initialize the array F which might be either complex or real. The array values are to represent either a Gaussian curve or a square. If F is real, the absolute squares of the complex values that would have occupied the same array position are needed. The machine-indicating OPTIONs are set as in the preceding example.

```
*PROC    /FIELD/
*KEEPTO *1 COMPLEX
      COMPLEX F
$     COMMON /FIELD/ F(DIM)
*ENDPROC
*B SAMPLE
        .
        .
        .
       SUBROUTINE SETFLD
*INCLUDE /FIELD/
C         READ POWER AND WIDTH OF BEAM
C           AND WIDTH OF WORKING GRID
C
*KEEPTO *2 CDC
      READ (5,200) POWER,WIDTH,GWIDTH
      IF  (EOF(5)  .NE.0.)   GO TO 100
```

```
*KEEPTO *1 IBM OR HONW
     READ (5,200,END=100) POWER,WDITH,GWIDTH
     IF (WIDTH.LE.GWIDTH)    GO TO 20
     WRITE (6,210) WIDTH, GWIDTH
     STOP
$20  DX=GWIDTH/_DIM
*KEEPTO ENDGAUSS GAUSS
C
C    INTENSITY IS
C       I(X) = POWER*SQRT(2/PI)/WIDTH
C        * EXP(-2*X*X/(WIDTH*WIDTH)
C    .7979 = SQRT(2/PI)
C
*SKIPTO *2 COMPLEX
     FNORM=.7979*POWER*DX/WIDTH
     ALPHA=-2./(WIDTH*WIDTH)
*KEEPTO *2 COMPLEX
     FNORM=SQRT(.7979*POWER*DX/WIDTH)
     ALPHA=1.414/WIDTH
     X=.5*(-GWIDTH+DX)
*LABEL ENDGAUSS
*SKIPTO SQUARE GAUSS
*SKIPTO *1 COMPLEX
     FNORM=POWER*DX/WIDTH
*KEEPTO *1 COMPLEX
     FNORM=SQRT(POWER*DX/WIDTH)
     XSTART=-.5*WIDTH
     X=-.5*GWIDTH
*LABEL SQUARE
$    DO 80 I=1,_DIM/2_
*KEEPTO *1 GAUSS AND NOT COMPLEX
*KEEPTO *1 GAUSS AND     COMPLEX
     F(I)=COMPLX(FNORM8EXP(ALPHA*X),0.)
*SKIPTO SQUARE GAUSS
     IF(X. E.XSTART)   GO TO 60
     IF(X+DX.LE.XSTART)   GO TO 40
```

```
C
C         APPORTION INTENSITY IF BEAM EDGE NOT CELL EDGE
C
*KEEPTO *1 COMPLEX
     F(I)+CMPLX(FNORM*SQRT((X+DX-XSTART)/DX),0.)
*SKIPTO *1 COMPLEX
     F(I)=FNORM*(X+DX-XSTART)/DX
     GO TO 70
*KEEPTO *1 COMPLEX
40   F(I)=(0.,0.)
*SKIPTO *1 COMPLEX
40   F(I)=0.
     GO TO 70
*KEEPTO *1 COMPLEX
60   F(I)=CMPLX(FNORM,0.)
*SKIPTO *1 COMPLEX
60   F(I)=FNORM
*LABEL SQUARE
C
C        TAKE ADVANTAGE OF SYMMETRY ABOUT X=0 (GRID CENTER)
C
$70  F(_DIM+1_-I)=F(I)
80   X=X+DX
*DEFN EVEN DIM/2*2
=
=        THESE ARE SAIL COMMENT CARDS
=        THIS SECTION WILL BE SKIPPED IF DIM IS ODD
             IT SETS THE MIDDLE POINT
=
*SKIPTO ENDODD DIM EQ EVEN
*SKIPTO *1 COMPLEX
$    F(_DIM/2+1_)=FNORM
*KEEPTO *1 COMPLEXT
$    F(_DIM/2+1_)=CMPLX(FNORM,0.)
*LABEL ENDODD
     RETURN
```

```
100     WRITE (6,220)
        STOP
200     FORMAT(3E10.3)
210     FORMAT(14H BEAM WIDTH ( ,E10.3,
       1 30H) IS GREATER THAN GRID WIDTH (,
       1 E10.3,1H))
220     FORMAT(38H END-OF-FILE ENCOUNTERED, NO BEAM INFO)
        END
```

2. FREQUENTLY USED SAIL MODES

The three modes used most frequently during SAIL executions are NORMAL, UPDATE, and LIST. NORMAL mode produces a card-image file to be used for some task. It can be used not only when a well-established file is needed, but also to test the effect of changes to the library before they are made permanent. When UPDATE is requested, SAIL creates a new library, thereby making permanent changes to an older library. In LIST mode, SAIL provides listings of the library, as affected by the changes. It can be used both to discover the line numbers for needed changes and to see the effects changes make on the library.

When a well-established library is used during a NORMAL run, the user will probably need to make no changes. If the default OPTION and PROGRAM selections are the ones that produce the desired file, the request area (in fact the whole INPUT file) could be left empty. If modifications to the OPTION list of a different set of PROGRAMs are needed, the INPUT file would consist of a request area. If an error on the library has been detected, a set of changes would be necessary on the NORMAL runs used to verify the correction. However, it is possible to set the options from the alternate input file. This is used to allow options to be set from other programs.

Development of a library could be handled by a combination of NORMAL, UPDATE, and LIST runs. Until changes become so numerous as to make INPUT burdensome to handle, NORMAL runs could be used to test out additions and corrections. When INPUT becomes too large or when some major portion has been completed and checked, an update run would create a new library. Normally, the line numbers would be resequenced during an update. However, especially when several users are involved in parallel development, it might be convenient to have the lines not involved in the change retain the numbers from an older stable version of the library.

In this case, either no resequencing would be done or only those PROGRAMs on which work was essentially complete could be resequenced using the SEQPROGRAM request.

At times, especially during a development phase, users might need extra listings of a library in addition to the one produced during the last UPDATE run. This can be accomplished by LIST runs. The EDIT feature, as well as the change set, is recognized during runs in LIST mode, so the library can be listed as it would appear after an UPDATE run with the same INPUT file except for the obvious change in the requests from LIST to UPDATE. The only advantages of this over doing a trial update and discarding the new library if further corrections were needed, are that the execution time to copy a temporary file onto the new library is saved and one less file need be assigned. On the other hand, if all went well, an UPDATE run would still be needed to create the new library.

3. USE OF REQUESTS

Most of the information the user needs concerning requests is presented in other sections of this manual. The specifics of what each request does are found in Section 3.1. Much of the test of Section V deals with choosing the operation mode, PROGRAMs, and OPTIONs. This section reemphasizes points about the interactions among the requests and warns about pitfalls that can arise while requesting EDIT.

The first thing to keep in mind is the general structure of the request area. In general, request elements are free-field, that is they do not have to start in any particular column; there can be any number on a card, and they are separated by one or more SAIL delimiters (blank, comma, or equal sign). Any unrecognized request is ignored. A character string that looks like a request may be treated as an OPTION or PROGRAM name if it falls within a specification area or it may be treated as a parameter if it follows a request that is the first member of a request pair. In many cases, a request parameter can be overridden if the same request recurs.

The very first request must be SAIL if any further requests or change cards are to be processed. Following this, any file control requests (listed in Section 3, para. 1) that refer to OLD must appear. As soon as a request that cannot affect OLD is found, the old library is opened and any further file control requests are applied to NEW. Unless the mode is GENERATE or NORMAL it is best to specify the mode next. This is because many function requests, such as EDIT and

those which affect listing, are ignored unless the mode has been already selected. For GENERATE runs, mode selection must be the last request, because SAIL immediately starts reading subsequent cards as lines for the library being created.

Remember the following restrictions. Only one mode can be in effect for a given execution of SAIL with the half-exception of LIST which is considered a function request during an UPDATE run. If two or more mode requests occur, the first remains in effect unless GENERATE occurs, which overrides all other modes. The CONVERT function request automatically selects COPY mode when it refers to OLD.

Finally, following the mode request if needed, the function requests and file control requests referring to NEW may come in any order. Unless the SYSTEM or VERSION requests explicitly appear in the part of the request area where they reset these parameters for NEW, the values from OLD will be used with the VERSION number incremented by one.

EDIT should be used with extreme care. A prior SCAN mode run with the same cards will find and list all the strings that will be substituted for. Remember that all cards are processed by the EDIT routine, including those newly inserted or modified by the column change feature. After an UPDATE or LIST run on which he requests EDIT, the user should check each card from the SCAN listing, as well as each card marked by an asterisk as newly changed, to insure that the desired character changes were made.

4. SETTING UP SELECTION MECHANISMS

Three basic methods are available for selecting which cards from a library will be written to the SAIL file during a NORMAL run. In order of increasing flexibility, these are:

a. PROGRAM requests
b. Use of *KEEPTO and *SKIPTO
c. Combinations of *PROC and *INCLUDE

Not only its inflexibility, but the fact that it is independent of the OPTION selections (although the converse is not true), puts PROGRAM selection into a low priority among the available selection mechanisms. Nevertheless, it has some utility. The original purpose of PROGRAM structure was to indicate the main FORTRAN programs on a library of related programs. In the HULL library, SAIL

itself, a problem initializer, a preprocessor that managed interfaces with a tape library, the main problem solving code, and a graphics package were stored and selected by the PROGRAM request. A second form of library where PROGRAM selection is useful is one consisting of modules, subsets of which can serve as replacements for each other. Each module could be identified as a PROGRAM, allowing it to be selected for individual output on NORMAL, LIST, or PUNCH runs. This form of selection would work best on module libraries where strict module interfacing rules were imposed on module programmers. Strict enforcement of such rules would be less necessary if the modules were not used with ones not supported by SAIL, for by tying OPTIONs to PROGRAMs (see the next section), interfaces could be made dependent on which modules were in use. If OPTION controlled selection were used on such a library instead of, rather than in addition to PROGRAM selection, module separation for use within non-SAIL-supported modules could only be accomplished on NORMAL runs, so only one version of the module, not all those on the library could be obtained.

The remainder of this discussion is concerned with OPTION-controlled selection mechanisms. Notwithstanding the comment at the end of the preceding paragraph, it is in NORMAL mode where selections are usually wanted, so the fact that these selection mechanisms work only in that mode is not a drawback. In fact, if they worked on UPDATE runs, it would be impossible to change lines containing SAIL executive directives.

The flexibility of the OPTION-controlled mechanisms resides basically in the fact that they rely on logical expressions, not just single TRUE-FALSE tests. It is further enhanced by the fact that the logical units in such expressions can be relational tests on OPTIONs. Even more, it is OPTIONs that are used in dynamic substitution so the choice of dimensions, for instance, can be tied to the choice of cards. Finally, OPTION-controlled selection can interconnect with PROGRAM selection due to OPTION revaluing feature explained in the next section.

Like PROGRAM selection, the *KEEPTO and *SKIPTO directives are sequential in their operation. Each line or set of consecutive lines is either skipped without processing or it is not. It is important to remember that *KEEPTO and *SKIPTO either immediately initiate skipping or they are ignored. Once skipping is initiated, it is stopped only if (1) the line count is satisified or a matching or blank *LABLE directive is encountered, (2) the end of the current

PROGRAM is found. In particular, a *KEEPTO with a TRUE operand or a *SKIPTO with a FALSE one will not nullify skipping initiated by a previous *KEEPTO or *SKIPTO.

The great flexibility inherent in selection by *PROC and *INCLUDE manifests itself in several ways. A block of cards can be placed in several places on SAIL while only appearing once on the library. Blocks can be reordered by reordering the *INCLUDE cards. PROCs can be nested within other PROCs. The macro processing ability generalizes the concept of block of cards from strictly identical set to one in which different character strings can occur for each *INCLUDE while retaining the same basic structure. Augmenting all the above features is OPTION control over whether to create a PROC, whether to honor an *INCLUDE, and which PROC should be included.

The user must avoid two basic pitfalls when using PROCs. First, only *INCLUDE (except the conditional expression is processed at PROC creation time) and *E are honored within a PROC when it is included. The other executive directives, in particular *KEEPTO, *SKIPTO, *DEFL, and *DEFN, are honored when it is first created. Second, SAIL aborts if a PROC has not been created at the time an *INCLUDE for it is honored (see Section IV, para. 2b). This problem can arise when using OPTIONs either to select a PROC name or to nullify creation of a PROC. These situations might arise if a library contains several versions of a PROC. with the correct one being selected by OPTIONs.

Having reviewed the properties of the OPTION-controlled selection mechanisms, let us consider how and when to use them. As a general rule, if a block of lines are to appear only one place on the SAIL file, use *KEEPTO or *SKIPTO; if it must or can appear in several different places, define it as a PROC.

It would be possible to create the same final SAIL file by using either only the skipping directives or only PROCs, but either extreme would be cumbersome. With only *KEEPTO and *SKIPTO, a line would have to appear on the library explicitly in each position where it was needed for any single variation of the SAIL file. With only the PROC mechanism a large number of PROCs would have to be defined. On some computers, SAIL will abort if too many (about 500) PROCs are created. By using a combination of selection mechanisms, the user can create an efficient library that is easy to comprehend.

5. REVIEW OF OPTION DEFINITION METHODS

Because OPTIONs play such an important role in producing task-oriented files on NORMAL runs, a review of the means by which they are defined is warranted. Each library file has a default OPTION list which is set during GENERATE, UPDATE, or COPY runs. On such runs, additions, redefinitions, and deletions given in the request area modify the old default list to form the one written out on the new library. Permanent changes of the default list can be made only in this manner.

No action involving OPTIONs, except editing the default OPTION list, occurs during runs in modes other than NORMAL, so the rest of this discussion is confined to that mode. Before any SAIL records are processed, the default OPTION list is modified by the OPTIONS and DELOPTIONS requests just as in any other run. Then SAIL reads the alternate input file (INPUT2) for PROGRAM and OPTION directives. Finally, each selected PROGRAM name is compared with each OPTION name. For each match where the option value is zero, it is reset to one. This changes the OPTION effectively to TRUE if it was FALSE. Once processing begins, new OPTIONs can be added to the list and old ones given new values by the *DEFN and *DEFL directives. Note that changes to the OPTION list by these directives only affect the lines which physically follow them on the library.

The "DEF" operator gives the user a powerful tool for setting OPTIONs. Consider the structure:

```
*SKIPTO *1 "DEF" OPTA
*DEFN OPTA operand
```

This could be more properly or readily handled by the default OPTION list of the numerical operand were a simple constant. However, if one or more OPTION names are used in the operand, these lines establish a default relationship between OPTIONs which can still be explicitly overridden by setting OPTA with the OPTIONS request. Obviously, OPTA should not appear in the default OPTION list or the *SKIPTO would be useless. There is a case where even a constant operand makes sense. That is where these two lines fall within the range of a longer *SKIPTO that depended on another OPTION. An example of this usage occurs within the library for SAIL itself where the OPTIONs are used to handle machine-dependent features. The lengths of various I/O buffers are set by this structure to values best suited for the machine for which a source file of SAIL is being produced.

Another use of "DEF" is possible if the user establishes the convention that among a set of OPTIONs that are used mainly for TRUE-FALSE tests, that one appear on the default OPTION list, that those explicitly defined by the OPTION request for a run (usually just by naming them with no paired value so they are set to zero) should be reset to one (TRUE), and that those no otherwise defined should be set to zero (FALSE). For each such OPTION, a SAIL record of the form:

```
*DEFL OPTA "DEF" OPTA
```

would properly reset the values. Of course, each such OPTION could be left undefined (by the library) and the logical unit "DEF" opta used for each test instead of OPTA, but this might get cumbersome.

A slight modification of the convention might involve defining undefined OPTIONs as zero (FALSE), resetting a-1 zero values to one (TRUE), and leaving any larger values alone, thus allowing the OPTION to convey more information in TRUE cases. A proper set of *SKIPTO or *KEEPTO instructions would have to proceed the *DEFL. In particular,

```
*SKIPTO *1 "DEF" OPTA
*SKIPTO *1 OPTA
*DEFL OPTA "DEF" OPTA
```

6. NAMING OPTIONS

Any string of eight or fewer characters that contains no SAIL delimiter can be used as the name during OPTION definition. Both the OPTIONS request and the DEFL and *DEFN directives will establish or reset OPTIONs with such names. However, there are some practical restrictions on the actual names a user should employ. Except for the special case of a table entry (which must be more general), OPTION names should both begin and end with alphabetic characters.

This restriction is based first upon the concatenated OPTION name-integer form of logical unit within a logical operand. While processing such an element as A1B23, SAIL recognizes it as meaning A1B EQ 23. Thus, if an OPTION were defined as A1B23, SAIL would be unable to test it for TRUE or FALSE or to use it in a comparative test in a logical expression. Therefore, any OPTION name to be used in a logical operand should not end in a number. Also, the letter combinations that form logical or comparative operators should not be used as OPTION names.

Another reason for restricting OPTION names arises from the actions performed by the dynamic substitution processor. This applies mainly to cards identified

for such processing, and the general rule that OPTION names begin with alphabetic characters prevents most problems. What has to be avoided is inadvertent substitution. Because a left parenthesis or slash will initiate construction of a string for possible substitution, and commas can delimit these strings, a field may match an OPTION name and be replaced by its value even though no such action was intended by the user. The fields occurring in FORTRAN FORMAT statements may present such a case if substitution is in effect. Usually, no character string that both begins and ends with a letter will be a legal FORMAT specification, although such things as A3/1X might still appear set off by commas. Another possible inadvertent substitution situation is on cards marked for substitution that contain a simple integer variable as a subscript. Because this is exactly the syntax that occurs on declaration statements where substitution is desired, the user should avoid giving an OPTION the name of a FORTRAN variable.

The use of a table in an OPTION list has been neglected in most of the discussion in this manual. SAIL uses these tables in two processes. One is dynamic substitution where a field set off by the special delimiter begins with a dollar sign. The other is in constructing a PROC name on *INCLUDE directives where the name field is set off by elements consisting of dollar signs. Section IV, para. 2b and 3b., respectively, explain how SAIL processes these fields.

What is of concern in this discussion is how to set up a table or tables to implement these features, that is, how to use the "AFTER" request within the OPTIONS specification area. Especially for dynamic substitution, the user might want a general character string. He must remember and consider the reasons for the restrictions on OPTION names presented at the beginning of this section. An OPTION name, whether or not used as a character string to be substituted for another, can neither exceed eight characters nor contain any of the SAIL delimiters (blank, comma, or equal sign). It also should be chosen so that inadvertent substitution does not occur. One interesting point is that an OPTION name that appears on a table and ends with a numeric character will never be confused with any logical element, because of the scanning rules within logical operands.

Concerning the mechanics of the "AFTER" structure itself, several points should be reemphasized. First, the table-identifying OPTION that appears immediately after the "AFTER" request, must have already been defined. Second, as OPTIONs are placed on the table, if they are already on the OPTION list, they are moved from where they were to the current slot on the table. This applies

even to OPTIONs that have been placed at previous positions on the table, they are moved to the new position and removed from the old one. If an OPTION name appears immediately following itself during table construction, the removal takes place but it is not inserted back on either the table or the entire list. This terminates table construction and the normal processing initiated by the OPTIONS request resumes, that is OPTION whose names already appear in the list are redefined and new OPTIONs are added at the end of the entire list. There is no particular reason to terminate "AFTER" processing, although it can be done by placing an unused string, such as END, twice in a row. The table length is specified when the user sets the value of the table-identifying OPTION, so all the "AFTER" request does is insure that the OPTIONs are in the proper order. In fact, another table or tables can be added by putting the table-identifying OPTION with its value after the "end" of the previous table followed by its table entries. Remember that the first OPTION on the table whose value agrees with the desired value specified in the field that initiates table searching, is used in the substitution or PROC name construction. If no match is found, the original characters, minus the identifying delimiters, are used. The use of *DEFN to modify the values of table members during executive processing gives further flexibility if it is needed.

SECTION VI

MACHINE DEPENDENT INFORMATION

1. CDC CONSIDERATIONS

a. Invoking SAIL

An absolute load of SAIL is stored in a loader library in a permanent file. At AFWL, the permanent file name and ID are HULLIB and DYMXCER. The entry points DYTSAIL, DYTLAMB, DYTHUL, and DYMAST are currently active as initiators of SAIL. Thus, the control sequence

```
ATTACH(HULLIB,ID=DYMXCER)
LIBRARY(HULLIB)
DYTSAIL(...)
```

will execute SAIL. The parameters on the DYTSAIL or DYTLAMB control card are of the form p=aaaa, where p identifies the parameter to be reset and aaaa is the new value. Possible values of p, the default values, and the meanings of the parameters are:

P	Default	Meaning
I	INPUT	Name of input file containing SAIL control cards.
O	OUTPUT	File to print diagnostics (and listings, if requested).
S	SAIL	File for output of NORMAL and PUNCH runs.
N	NEW	File for the new library.
D	entry point used	Base for permanent file names used in internal attaches.
ID	for entry point DYTSAIL - DYTHULL DYTLAMB - DYTHULL DYTHUL - DYTHULL DYMAST - DYMXCER DESHUL - DESHULL	
CY	higher cycle	Cycle number for permanent file attaches.

The D and ID parameters will be discussed more thoroughly in the next section which covers library file access.

A few considerations of the input file are in order. The end-of-file is detected using the FORTRAN function EOF. Unless this file is named INPUT, that is unless it is the card file read in for the job, EOF detects end-of-partition, not end-of-section (7-8-9) cards. It is the mark written by COPYBF or COPYP that is recognized by the EOF function. This permits concatenation of several files into one for use as an input file for SAIL by use of COPYBR or COPYS. For example, suppose a local file named SRS contains the compilable cards for a routine the user wants to make into a SAIL library. The control card sequence

```
COPYBR(INPUT,A)
COPYBR(SRS,A)
ATTACH(HULLIB,ID=DYTHULL)
LIBRARY(HULLIB)
DYTSAIL(I=A)
CATALOG(NEW,...)
```

where the record copied from INPUT was one card long and consisted of

```
SAIL SYSTEM=name GENERATE
```

would set up the GENERATE run.

Bear in mind that SAIL always rewinds its designated input file and that an empty record is valid for SAIL control. Thus, if SAIL is to be executed two or more times during the same job with different requests and/or changes, its control files must be something other than INPUT for all but one, because the first record of SAIL control cards will be used over and over. The easiest way to alleviate this situation is to copy each section of INPUT containing SAIL control cards to its own separate file. Also bear in mind that when SAIL terminates, the file is left positioned just behind the marker recognized by the EOF function.

b. Access to Library Files

On CDC machines, SAIL can go hunting for its old library file and can allocate a device for its new library file.

The request TAPE applied to the OLD file tells SAIL to request a tape from the operating system for OLD with VSN and density as described in Section III, para. 1a. The request LOCAL will prevent SAIL from trying to attach a permanent file even if no local file named OLD is known to the job.

If the TAPE request does not appear and a local file named OLD is known for the job by the CDC-SCOPE operating system, SAIL will first check if it conforms to library format.

When neither the LOCAL nor the TAPE requests appear and OLD either does not conform or is unknown, SAIL tries to attach a permanent file for OLD. The permanent file ID parameter is either the ID parameter from the control card that invoked SAIL or the installation default (DYMXCER at AFWL). The permanent file name is constructed from a base that is defined in one of three ways. If the SYSTEM request appears for OLD, that name is used for the base. If there is no SYSTEM request, the parameter specified after the D= on the invoking control card is used. And if neither of these is present, the base is the default name of the invoking control card. The permanent file name is constructed by concatenating installation dependent characters onto the base. At AFWL, no characters are concatenated on, so the base is the permanent file name for the old library file. The cycle number is the number given by the VERSION request if that was specified for OLD. Otherwise, the highest cycle for that permanent file name and ID are attached.

Once an OLD file has been established, SAIL checks whether the SYSTEM and VERSION parameters stored on the file agree with the requests. If either request has not been made, that check is considered as passed. If the checks fail or if the file is not in library form, a flag is set to abort SAIL unless GENERATE mode is requested. Note that some file will be attached as an OLD candidate if LOCAL is unspecified even on GENERATE runs, so if the checks are passed (or not made), the OPTION list from that library will be set up for modification. To avoid adding unneeded OPTIONs from some other library on a GENERATE run, either the SYSTEM request should come first so the check will be failed, or a LOCAL request made.

On UPDATE, GENERATE, and COPY runs, SAIL will write NEW on a permanent file device unless either TAPE or LOCAL is requested for NEW. If TAPE is requested, the file will be written on the tape specified by the request as explained in Section II, para. 1a.

Because the SCOPE Record Manager routines for the various files have been specified (on CDC 6600 systems) when the absolute load was created, a FILE card will not be recognized for SAIL's files, in particular, SAIL or formatted versions of NEW or OLD. If these files are needed in a different format so as

to comply with the requirements of transporting to a machine of a different manufacturer, as might occur when CONVERT is requested, the SCOPE control card FILE would be useful. Currently, the files must be copied by reading each card and rewriting it on a different file in a FORTRAN program so as to implement the capabilities of Record Manager. There is no such problem on the CDC 7600, for Record Manager is applied by all routines that interact with files, so FILE cards are always recognized.

2. IBM CONSIDERATIONS

a. Invoking SAIL

The SAIL program is invoked on the IBM 360 or 370 by executing the procedure SAIL. The substitutable parameters for this procedure are:

LIBPRE - the prefix for the SAIL library. The current default is 'SAIL'.

LIB specifies the library data set name which contains the SAIL program in a partition named SAIL. The default value of this parameter is HULLIB.

LIBU specifies the unit where the library data set is located. This parameter has a null default and must be specified if the library data set has not been cataloged.

LIBVOL specifies the VOLUME parameter for the library data set. It has a null default value and must be specified if the library data set has not been cataloged.

ALTI - The definition of alternate input file default is 'DUMMY,'.

CHNBLK specifies the block size of the change file data set. It has a default value of 3521.

CHLRL - The logical record length of the change file 3517.

DISPn specifies the disposition field for data set assigned to the nth block of the SAIL file. The defaults are 'DISP=(NEW,PASS)' for all values of n.

EXP specifies the retention period for the NEW file data set. The default is 'RETPD=720' and it should be changed to null if NEW is to be on an unlabeled tape.

FILN specifies the file number on a tape where the NEW file data set is to be written on a tape.

FILO specifies the file position of the OLD file data set on a tape. It is given a null default and need only be specified if the old file data set is on a tape.

GENDUM is a parameter which allows the user to dummy the OLD file DD card for generation runs. It has a null default and should be set to 'DUMMY,' for a generation run.

GENN is the parameter which specifies further qualification of the NEW file data set name. It has a default of '(+1)' so that it will be the next generation of a generation data group. If the new file is the first of the generation data group (i.e., this is the first time this data set name has been placed on the system), then this parameter should be set to '.Gnn.V001' (where nn is the VERSION number of the library). If the data set is not a member of a generation data group this parameter must be set to an appropriate value.

GENO specifies the further qualification of the OLD file data set name. The default value is '(0)' which assumes that the data set is the most current member of a generation data group. This parameter must be set if the OLD file is not a member of a generation data group.

LABN specifies the type of label for the NEW file data set. It has a null default and must be set if the data set is to be written on a tape with other than a standard IBM label.

LABO specifies the label type for the OLD file data set. It must be specified if the OLD file is on a tape with other than an IBM standard label. The default is null.

NEWPRE - the data set prefix for the new data file. Default is 'SAIL'.

NEWDCB is the DCB parameter for the NEW file. It has a default of '(RECFM=VBS,BLKSIZE=7294)'. The block size may be changed to allow better utilization of disk space. If the run is to produce a coded NEW file, this parameter should be set to '(RECFM=FB,LRECL=120,BLKSIZE=1200)'.

NEWDS is the disposition of the NEW file data set. Its default is (NEW,CATLG,DELETE) .

NEWDUM is a parameter which allows allocation of data sets for generation, update, and copy runs. It has a default of DUMMY and is used to dummy out the NEW and TEMPF file DD cards. This parameter must be set to null to allow the TEMPF and NEW files to be allocated to data sets on generation, update and copy runs.

NEWP allows the user to specify password control for the NEW file data set. It has a default of NOPWREAD for read only access.

NEWSPC is the SPACE parameter for allocation of the NEW file data set. It has a default value of '(CYL,(10,20),RLSE)' and can be changed as needed.

NEWU specifies the unit on which the NEW file is to be written. It has a default of TAPE, but may be set to other units.

NEWVOL is the VOLUME parameter for the NEW file data set. Since this one has a null default, it must be specified on any generation, update, or copy run.

OLDPRF - the prefix for the old data set name. Default is 'SAIL.'

OLD specifies the last simple name of the OLD file data set. The default value of this parameter is SAIL to indicate the OLD file contains the SAIL library. This parameter must be specified when running SAIL for any other SAIL library.

OLDDCB is the DCB parameter for the OLD file data set. It has a default value of '(RECFM=VBS,BLKSIZE=7294)'. This parameter may be changed to allow for a better block size. When converting from a coded OLD file, it should be set to '(RECFM=FB,LRECL=120,BLKSIZE=1200)'.

OLDDS specifies the disposition of the OLD file data set. The default is SHR.

OLDU specifies the unit on which the OLD file data set resides. The default for this parameter is null and it must be given if the old file data set is not cataloged.

OLDVOL specifies the VOLUME parameter for the OLD file data set. It has a null default and must be given if the OLD file has not been cataloged.

PROG - the member name for SAIL in the library. Default is SAIL.

PRCL - the number of bytes in random file used for PROCs. Defaulted to length of record for SAIL generation.

PRCN - the number of records in the PROC file. Defaulted from SAIL generation.

P1 specifies the routing of the normal output from the SAIL program. The default is 'SYSOUT=A'.

P2 specifies the routing for the nonfatal error messages from the SAIL program. The default is 'SYSOUT=A'.

REG is the region size for the SAIL execution step. Its default is 160K.

SAILBLK specifies the block size for the card image data sets which are produced on the SAIL file during a NORMAL run. The default is 800.

SAILR - the logical record length of the SAIL file used during NORMAL runs. Default is 80.

SAILn specifies the allocation for the data set assigned to the nth block of the SAIL file during a NORMAL run (n runs from 1 to 15). 'DSN=SAILCD, UNIT=SYSDA,SPACE=(TRK,(10,20),RLSE)' is the default for SAIL1. The default value is DUMMY for all other values of n.

SCRTC specifies the generic unit name for the scratch disk area. The default is SYSDA.

STIME is the TIME parameter for the SAIL execution step. Its default is '(2,0)'.

TEMBLK specifies the block size used by the TEMPF file data set on a generation or update run. The default is 7924.

TEMPLRL - the logical record length of the TEMP file set used in generation or update run. Default is 7290.

b. Examples of SAIL Runs

To copy the SAIL library from an unlabeled tape to a generation data group member, the following would be used:

```
//CPY  EXEC  SAIL, GENN= '.G50V00 ,NEWVOL= SER=SAILVOL' ,
//           OLDU=TAPE9,OLDVOL= 'SER=SAILTAP ',
//           OLDDS= (NEW,KEEP) ,
//           LABO=NL,FILO=1,GENO= '.TAPE ',NEWDUM=
//SAIL.INPUT  DD  *
```

```
SAIL CONVERT COPY
/*
        To perform a NORMAL run from SYSTEM LAMB the card would be:
//NRML   EXEC   SAIL,OLD=LAMB
//SAIL.INPUT    DD  *
SAIL
      (SAIL change cards)
/*
```

As can be seen from the examples, the procedure has one step name SAIL and the input DD name is INPUT.

c. The SAIL Procedure

```
//* * * * * * * * * * * * * * S A I L * * * * * * * * * * * * * * * *
//*
//* NAME
//*   SAIL
//*
//* FORMAT
//* //  EXEC  SAIL,...(OPTIONAL PARAMETERS AS REQUIRED)
//*  //SAIL.INPUT  DD  *
```

3. HONEYWELL CONSIDERATIONS

Invoking SAIL

A load module of SAIL is stored on permanent file. This load module is invoked by the Honeywell control card PROGRAM. An example of the control cards for a NORMAL run are:

```
$       PROGRAM RLHA,DUMP
$       LIMITS  ,33K
$       PRMFL   H*,R,R,USER/SAILHSTR
$       FILE    01,NULL
$       TAPE9   02,A2D,,tape number of SAIL library
```

```
$       FILE    03,A3R,20L
$       FILE    04,A4R,10L
$       FILE    10,B0R,20R
$       FILE    11,B1R,20L
$       FILE    12,B2R,40R
$       FILE    15,NOSLEW
$       FILE    15,B5S,50R
$       DATA    I*,,COPY,ENDFC
      SAIL control cards
$       ENDCOPY 1*
      Cards for utilizing file 15
$       ENDJOB
***EOF
```

The control cards for files 3, 4, 10, 11, and 12 should remain unchanged for any runs in any execution mode except for possible variation in the file-size parameter. The control card for file I* should also remain unchanged. No control card is needed for P* which is commonly referred to elsewhere in this manual as OUTPUT. Refer to Section II, para. 2 for the correspondence between other file names appearing in this manual and the Honeywell file number.

The control cards for files 1, 2, and 15-23 (NEW, OLD, and SAIL) depend on the SAIL execution mode. During NORMAL runs, the SAIL file begins on file 15 and is incremented by one each time a *E directive (if any) is written. Most NORMAL runs require only file 15. If additional files are required, control cards for files 16 onward should be defined in a manner similar to file 15 in the above example except that each file must have its unique logical unit designator.

The following example shows how to set up the control cards for a run in UPDATE mode. In this example the entire library is to be resequenced and the default OPTION list altered.

```
$       PROGRAM RLHS,DUMP
$       LIMITS  ,33K
```

```
$       PRMFL   H*,R,R,USER/SAILHSTR
$       TAPE9   01,A1D,,tape no. of new library
$       TAPE9   02,A2D,,tape no. of old library
$       FILE    03,A3R,20L
$       FILE    04,A4R,10L
$       FILE    10,B0R,20R
$       FILE    11,B1R,20L
$       FILE    12,B2R,40R
$       FILE    15,B5R,20L
$       DATA    I*,,COPY,ENDFC
SAIL UPDATE SEQ
OPTIONS ... ENDOPTIONS
  SAIL change cards
$       ENDCOPY I*
$       ENDJOB
***EOF
```

Finally, here is an example of creating a new library from cards. The SYSTEM identifier for the library is to be NEWLIB. SECOND and PLOTTER are the PROGRAMs to be written to file 15 by default on NORMAL runs. A default OPTION list is also being stored on the library.

```
$       PROGRAM RLHS,DUMP
$       LIMITS  ,33K
$       PRMFL   H*,R,R,USER/
$       TAPE9   01,A1D,,tape no. of new library
$       FILE    02,NULL
$       FILE    03,A3R,20L
$       FILE    04,A4R,10L
$       FILE    10,B0R,20R
```

```
$       FILE    11,B1R,20L
$       FILE    12,B2R,40R
$       FILE    15,NULL
$       DATA    I*,,COPY,ENDFC
SAIL SYSTEM NEWLIB
        PROGRAM SECOND PLOTTER ENDPROGRAM
OPTIONS ... ENDOPTIONS   GENERATE
        Lines to be placed in library go here.
$       ENDCOPY I*
$       ENDJOB
***EOF
```

SECTION VII

HISTORY AND PHILOSOPY OF SAIL

1. HISTORY OF SAIL

Sail grew from the merger of two closely related programs used to manage the HULL system of hydrodynamic programs at the Air Force Weapons Laboratory. The older of these was the progenitor of the current SAIL executive. Its objective was to produce card-image source code tailored to each problem to be solved. In recognition of the fact that a particular section of code or a particular routine could be used for several related problems, a library file was created that held all coding for all treatments of each physical effect related to the system of routines. The outstanding characteristic of both the original executive and the current SAIL is that this file was structured so that the accessing program could produce efficient compliable code.

The construction of a library file that could handle a range of problems involving similar coding that differed in a few details could have been achieved in several ways. Let us consider the approach found in SAIL and compare it to two alternatives that might have been used. On any particular execution, the original executive processor set dimensions in FORTRAN declaration statements and defined blocks of cards called PROCs, each of which could be inserted into the resultant card-image file at any places where the *INCLUDE directive with the matching name occurred. Dimension values and control over whether or not to include a PROC were provided through pairs of parameters, consisting of a character string and an integer value, called OPTIONs, SAIL reflects upgrades to this processor so it now effectively selects only the cards appropriate to the task at hand while producing a card-image file. Once all the needed portions of code have been placed in a library, new programs can be constructed merely by choosing the proper combinations of OPTION values.

Among the alternative approaches to library construction is a general program that consists of all coding needed for any task with flags that select the appropriate portions during execution. To deal with varying array-length requirements, the used could: (1) set all dimensions to their largest possible values, (2) change all the array declaration statements and recompile, or

(3) employ a complicated bookkeeping scheme to store this data in one or two large arrays. A second alternative approach is to maintain a file containing the version of the program oriented toward one of the tasks along with change decks for each other task so an editing utility can build the appropriate version.

SAIL has advantages over both these alternatives and, in a sense, combines them in its library file structure which, like the general program, contains all coding needed for any task, while its selection directives essentially define built-in change decks. General programs suffer from overheads in both memory space and computation time. Routines and sections of routines unneeded for a particular task, and possibly unused array positions, require memory beyond that needed by the tailored code produced by the other approaches. Extra time is spent checking flags to determine which portions to skip and which to execute. If a bookkeeping system is used to hold down array lengths, the coding becomes less transparent to the users and it may take more time to calculate addresses. The change deck approach, although it can produce code tailored to a problem, often causes extra work. When coding errors are found or new features that apply to several tasks are added, the user must not only modify several change decks, he must first identify and locate those which are affected. Also, the change decks can become obsolete if the line numbers on the basic file are resequenced. These problems are relieved in well-managed SAIL libraries, for corrections can be made (after testing) directly to the library and the OPTIONs which selected cards appropriate to a particular task will select the corrected cards if they dealt with that task. If additions are made the selection mechanisms can be set so that the OPTIONs either keep or skip the new cards, as appropriate to each old task as wll as any new ones the additions make possible.

Even with only the two features mentioned in the above comparison, the original executive processor provided a powerful tool for program management. However, its libraries were fairly static. As users were presented with different problems to solve, completely new portions of code were needed which had not yet been added to the library. Thus, in December 1973, the need for an updating program became evident. The program was required to update the file in such a way that the identifiers of the existing card images in the filed need not be changed. It was desired that the order of the change sets could be random rather than that in which the records they modified where found in the file. The CDC system UPDATE did not satisfy the exact need, so the development of SAIL began.

The first version of SAIL was simply a FORTRAN program which updated a file of SAIL records. Each SAIL record contained one card image and an identifier which specified its block and card location. During the initial generation of a file or during an update where resequencing was requested, the records were assigned sequential block numbers and card numbers of zero. During other updates, they retained their previous identifiers and added records were given the block numer of the previous existing record with a unique sequential card number.

To allow SAIL to identify instructions given to it, the convention was adopted that the directive verb would immediately follow an asterisk (*) in column one. Cards with an asterisk in column one but no understandable SAIL directive were processed as ordinary card images to be added to the file. The file was divided into groups of records called PROGRAMs which were identified by a special directive in the file. These PROGRAMs could be selected for individual processing.

During execution of SAIL, the change sets were read and any cards to be inserted were written onto a random file with information about the changes and a pointer to the cards kept in central memory. The allowed modifications were insertion of new cards and deletion with or without insertion. A set of records in the existing file could be copies to any place in the change set that a card could be inserted. To accomplish this the original file was read and the sets requested were copied to the random file while pointers were saved to allow these records to be included in the correct change set. The insertion and deletion pointers were then sorted into the order that the records they modified were found in the original file. Then, all the changes were written in that order onto a sequential file in a special packed format. It was then a simple matter to modify the original file by merging it with the sequential change file.

While this first version of SAIL was being used, the capabilities of the executive processor were being expanded. The ability to create PROCs was enhanced by making the creation and inclusion depend on the result of a logical string, rather than on the value of a single variable. In addition, while producing the card-image file, blocks of card images could now be skipped or kept depending on the result of the same type of logical string. The dynamic dimensioning routine was expanded to allow numbers to be calculated from OPTION

values, encoded, and inserted in card images other than FORTRAN DIMENSION statements. The routine was then described as a dynamic value substitution routine.

The next requirement was to include the executive processor in the SAIL program proper. Although when this was first done there were exactly four OPTIONs fixed in the program, the executive portion was rewritten to allow the user to add more OPTIONs with a default set provided in the header of the original file being processed.

Next, the ability to resequence selective PROGRAMs during an update was added. (This required the additional convention that each PROGRAM begin with a block number which was a multiple of 10000.) The card images found in the file before the first PROGRAM were designated as the PROLOGUE which was set to be processed regardless of the PROGRAMs selected, thereby allowing PROCs to be created and used by several PROGRAMS. Finally, the program was expanded to perform automatic internal attaches of any CDC permanent files it needed.

In November of 1974, it became apparent that SAIL should run on computers other than those with CDC SCOPE systems, so work began on a new phase of SAIL development. The dependence on the word length of the CDC computers was removed from the program as well as the use of such CDC system features as ENCODE and DECODE.

At the same time, new capabilities were added bringing SAIL to its present form. OPTION values could be set during processing using simple logical or arithmetic expressions involving OPTIONs and constants. A new SAIL directive controlled modification of just a portion of a card image (this is called the column change feature). Finally, the ability to scan for or replace character strings was added to the program.

2. SAIL SUPPORT AND MAINTENANCE CONCEPT

One of the main strengths of SAIL has been its slow growth under the support and control of users. This should continue in the future. If anything, further development will probably be slower, for most of the really useful features have already been incorporated into the program. In any case, the following criteria should be strictly adhered to:

a. Any changes made to SAIL will be such that all existing libraries can be processed with the current results except for cases where obvious errors exist in SAIL.

b. No existing feature will be removed from the program without unanimous consent from the SAIL user community.

c. Any new feature will be added only with the approval of a majority of the community and only if a significant number of users need the feature.

d. The official versions (as identified by the VERSION number) of each SAIL library will be identical on all machines.

e. New VERSIONs of SAIL will be developed as new computers and operating systems are added to the user community, or as better coding is written to support the current features.

APPENDIX A

SAIL GENERATION

1. GENERATION OPTIONS

Sail is maintained on a SAIL library and is generated with options which control its function. These options are:

INST - This defines the installation at which SAIL is being generated. The default is in the sys SAIL library file header. The values are

INST	
1	Air Force Weapons Laboratory Kirtland AFB, New Mexico
2	Air Force Armaments Laboratory Eglin AFB, Florida
3	McDonnell-Douglas *Los Angeles, California*
4	ABEMDA Research Center (Brown Engineering)
5	ABEMDA Research Center (SAI)
6	Atomic Weapon Research Establishment United Kingdom

SYS - This option defines the computer system for which SAIL is to be generated. The default is in the SAIL library file header. The values are:

SYS	Computer
66	CDC 6600
76	CDC 7600
176	CDC Cyber 176
360	IBM 360

SYS	Computer
370	IBM 370
6080	Honeywell 6080

VER - This option defines the version of the operating system being used. The default is in the SAIL library file header and the values are:

VER	Operating System
1	IBM OS
2	IBM OS/VS2
3	CDC SCOPE 3.3
4	CDC SCOPE 3.4
20	CDC SCOPE 2.0

LBUFF - This option defines the length of the random file buffer in character words. The defaults are:

CDC - 510

IBM - 430

Honeywell - 160

NUMREC - This option defines the number of records in the random file on IBM system. The default is 3000.

LENC - This option defines the size of the change file buffer in character words. The defaults are:

CDC - 512

IBM - 450

Honeywell - 160

CARDBUF - This option defines the number of SAIL lines in a SAIL library block. The defaults are:

CDC - 113

IBM - 82

Honeywell - 20

COMBUF - This option defines the size of the change command buffer. On IBM Honeywell systems the maximum number of changes (i.e., commands) is COMBUF/2.

HX - This option defines the H extended FORTRAN compiler on IBM systems if set to non-zero. Default = 0.

LOPT - This option defines the maximum size of the option table. Default = 150.

FILMPR - This option defines the film output ability for listing if non-zero. Defaults are:

INST	FILMPR
1	1
Others	0

2. IBM GENERATION PROCEDURE

The control procedure for generating a new SAIL from the SAIL library file is shown in the listing in paragraph 1. The procedure parameters are:

LIB - The partition data set name for current library. The full name is LIBPRE/LIB. Default = HULLIB.

LIBPRE - The prefix for current library data set. Default = 'SAIL.'

LIBU - The unit for current library. Default is null.

LIBVOL - The volume parameter for the current library. Default is null.

NLIB - Primary name for new library data set. Full name is NLIBPRE/NLIB. Default = HULLIBN.

NLIBU - UNIT field for new library data set. Default = SYSDA.

NLIBVOL - VOLUME field for new library data set. Default is null.

NLIBDS - Disposition of the new library data set for the Link Edit step for generation of SAIL. Default = '(NEW,CATLG)'.

LDSPACE - Space for the new library data set when disposition is NEW. Default = '(CYL,(20,5,5))'.

APROG - The name of the assembler. Default = IFOX00.

AREG - The REGION size for the assembly step. Default = 187K.

APARM - Assembly step parameters. Default = 'LOAD,NODECK'.

ATIME - Assembly step time limit. Default = '(2,0)'.

AMACL - The data set name for system assembler macro library. Default = 'SYS1.MACLIB'.

AP1 - The disposition of the assembler output. Default = 'SYSOUT=A'.

CHNBLK - The BLKSIZE for the SAIL change file. Default = 3521.

CHNLRL - The LRELL for the SAIL change file. Default = 3517.

FILO - The file number for the old SAIL file if on tape. Default is null.

FPARM - The function step parameters in addition to 'NAME=SAIL'. Default = 'MAP'.

FPROG - The name of the FORTRAN compiler. Default = IFEAAB.

FREG - The REGION size for the compiler step. Default = 512K.

FP1 - The disposition of the compiler output. Default = 'SYSOUT=A'.

FSPACE - The space for the object output from the compiler. Default = '(CYL,(10,5),RLSE)'.

FTIME - The time limit for the compiler step. Default = '(1,0)'.

GENO - The version suffix for the old SAIL file. Total data set name is OLDPRE/OLD/GENO. Default = '(0)'.

LABO - This defines the label tape for a tape old SAIL file. Default is null.

LPARM - This defines the LINK-EDITOR step parameters in addition to 'OVLY'. Default = 'MAP'.

LREG - This defines the REGION size for the linkage editor step. Default = 250K.

LP1 - The disposition of linkage editor printed output. Default = 'SYSOUT=A'.

LTIME - This defines the time limit for the linkage editor step. Default = '(0,45)'.

SGENSP - This defines the space for the source output from SAIL for the compiler. Default = '(CYL,(5,5),RLSE)'.

OLD - This is the primary data set name of the old SAIL file. Default = SAIL.

OLDDCB - This is the DCB field for the old SAIL file. Default is null.

OLDDS - This is the disposition for the old SAIL file. Default = SHR.

OLDPRF - This is the data set name prefix for the old SAIL file. Default = 'SAIL'.

OLDU - The UNIT field for the old SAIL file. Default is null.

OLDVOL - The VOLUME field for the old SAIL file. Default is null.

PRCN - The number of records in the SAIL random file. Default set for current SAIL.

PRCL - Number of bytes in SAIL random file. Default set for current SAIL.

PS1 - Disposition for the primary printed output from SAIL. Default = 'SYSOUT=A'.

PS2 - The disposition for the error output from SAIL. Default = 'SYSOUT=A'.

SATLBLK - The block size of the SAIL source data sets. Default = 800.

SAILR - The record length of the SAIL source data sets. Default = 80.

SCRTC - The UNIT class for the scratch disk. Default = SYSDA.

SPROG - The member name for the SAIL program in the current library. Default = SAIL.

SREG - The REGION size for the SAIL execution step. Default = 175K.

STIME - The time limit for SAIL execution step. Default = '(2,0)'.

WORKSP - The SPACE field for scratch data sets used by the assembler and compiler. Default = '(CYC,(5,5))'.

APPENDIX B

SAIL PROCEDURES

```
30000       *B CDC CONTROL CARDS                                                    7APR75
30001       =                                                                       7APR75
30002       =          THE FOLLOWING CONTROL CARDS ARE USED ONA CDC 6000 OR 7000    7APR75
30003       =                                                                       7APR75
30004       =        MACHINE WITH SCOPE 3.4 OR 2.1                                  7APR75
30005       =                                                                       7APR75
30006       =                                                                       7APR75
30007       =                                                                       7APR75
30008       =          NORMAL RUN                                                   7APR75
30009       =                                                                       7APR75
30010       REQUEST SAIL,HY.                                                        7APR75
30011       SAIL.                                                                   7APR75
30012       *E                                                                      7APR75
30013       SAIL TAPE  VSN  OPTIONS ENDOPTIONS                                      7APR75
30014       SAIL OPTIONS      ENDOPTIONS                                            7APR75
30015       *E                                                                      7APR75
30016       =                                                                       7APR75
30017       =                                                                       7APR75
30018       =          UPDATE RUN                                                   7APR75
30019       =                                                                       7APR75
30020       REQUEST SAIL,HY.                                                        7APR75
30021       SAIL.                                                                   7APR75
30022       *E                                                                      7APR75
30023       SAIL TAPE OLDVSN UPDATE TAPE NEWVSN                                     7APR75
30024       *E                                                                      7APR75
30025       =                                                                       7APR75
30026       =                                                                       7APR75
30027       =          CONVERT COPY  TO TAPE                                        7APR75
30028       =                                                                       7APR75
30029       REQUEST SAIL,HY.                                                        7APR75
30030       SAIL.                                                                   7APR75
30031       *E                                                                      7APR75
30032       SAIL TAPE OLDVSN COPY CONVERT NEWVSN                                    7APR75
30033       *E                                                                      7APR75
```

```
*  40000        *B IBM-PROCS                                                          12JUN78
*  40001        =                                                                     12JUN78
*  40002        =       THE FOLLOWING PROC IS USED TO RUN SAIL ON AN IBM 360 OR 370    12JUN78
*  40003        =       SYSTEM                                                        12JUN78
*  40004        =                                                                     12JUN78
*  40005        //* * * * * * * * * * * * * * S A I L * * * * * * * * * * * * * * * * 12JUN78
*  40006        //*                                                                   12JUN78
*  40007        //* NAME                                                             12JUN78
*  40008        //*   SAIL                                                          12JUN78
*  40009        //*                                                                   12JUN78
*  40010        //* FORMAT                                                           12JUN78
*  40011        //*   //        EXEC    SAIL,...(OPTIONAL PARAMETERS AS REQUIRED)   12JUN78
*  40012        //*   //SAIL.INPUT  DD  *                                           12JUN78
*  40013        //*                                                                   12JUN78
*  40014        //* FUNCTION                                                         12JUN78
*  40015        //*   THIS PROCEDURE EXECUTES THE SAIL PREPROCESSOR TO PERFORM ANY   12JUN78
*  40016        //*   OF THE SAIL FUNCTIONS (UPDATE,GENERATE,PUNCH,LIST,SCAN,COPY,   12JUN78
*  40017        //*   OR NORMAL RUN TYPES) COMPILATION AND EXECUTION OF RESULTANT    12JUN78
*  40018        //*   PROGRAM ARE NOT PART OF THIS PROCEDURE                         12JUN78
*  40019        //*                                                                   12JUN78
*  40020        //* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 12JUN78
*  40021        //SAIL    PROC LIB=NULLIB,                                        X 12JUN78
*  40022        //             LIBPRE='SAIL.',                                    X 12JUN78
*  40023        //             LIBU=,                                             X 12JUN78
*  40024        //             LIBVOL=,                                           X 12JUN78
*  40025        //             ALTI=DUMMY,                                        X 12JUN78
*  40026        //             CHNBLK=3521,                                       X 12JUN78
*  40027        //             CHNLRL=3517,                                       X 12JUN78
*  40028        //             DISP1='DISP=(NEW,PASS)',                           X 12JUN78
*  40029        //             DISP2='DISP=(NEW,PASS)',                           X 12JUN78
*  40030        //             DISP3='DISP=(NEW,PASS)',                           X 12JUN78
*  40031        //             DISP4='DISP=(NEW,PASS)',                           X 12JUN78
*  40032        //             DISP5='DISP=(NEW,PASS)',                           X 12JUN78
*  40033        //             DISP6='DISP=(NEW,PASS)',                           X 12JUN78
*  40034        //             DISP7='DISP=(NEW,PASS)',                           X 12JUN78
*  40035        //             DISP8='DISP=(NEW,PASS)',                           X 12JUN78
*  40036        //             DISP9='DISP=(NEW,PASS)',                           X 12JUN78
*  40037        //             DISP10='DISP=(NEW,PASS)',                          X 12JUN78
*  40038        //             DISP11='DISP=(NEW,PASS)',                          X 12JUN78
*  40039        //             DISP12='DISP=(NEW,PASS)',                          X 12JUN78
*  40040        //             DISP13='DISP=(NEW,PASS)',                          X 12JUN78
*  40041        //             DISP14='DISP=(NEW,PASS)',                          X 12JUN78
*  40042        //             DISP15='DISP=(NEW,PASS)',                          X 12JUN78
*  40043        //             EXP='RETPD=720',                                   X 12JUN78
*  40044        //             FILN=,                                             X 12JUN78
*  40045        //             FILO=,                                             X 12JUN78
*  40046        //             GENDUM=,                                           X 12JUN78
```

```
*  40047        //          GENN='(+1)',                                                    X 12JUN78
*  40048        //          GENO='(0)',                                                     X 12JUN78
*  40049        //          LABN=,                                                          X 12JUN78
*  40050        //          LABO=,                                                          X 12JUN78
*  40051        //          NEW=SAIL,                                                       X 12JUN78
*  40052        //          NEWDCB='(RECFM=VBS,LRECL=7290,BLKSIZE=7294)',                   X 12JUN78
*  40053        //          NEWDS='(NEW,CATLG,DELETE)',                                     X 12JUN78
*  40054        //          NEWDUM='DUMMY,',                                                X 12JUN78
*  40055        //          NEWPRE='SAIL.',                                                 X 12JUN78
*  40056        //          NEWP=NOPWREAD,
```

```
* 40057  //          NEWSPC='(CYL,(10,2),RLSE)',                    X 12JUN78
* 40058  //          NEWU=SYSDA,                                    X 12JUN78
* 40059  //          NEWVOL=,                                       X 12JUN78
* 40060  //          OLD=SAIL,                                      X 12JUN78
* 40061  //          OLDDCB=,                                       X 12JUN78
* 40062  //          OLDDS=SHR,                                     X 12JUN78
* 40063  //          OLDPRE='SAIL.',                                X 12JUN78
* 40064  //          OLDU=,                                         X 12JUN78
* 40065  //          OLDVOL=,                                       X 12JUN78
* 40066  *AUTO                                                       12JUN78
* 40067  //          PRCN=_NUMREC_,                                 X 12JUN78
* 40068  //          PRCL=_8*LBUFF_,                                X 12JUN78
* 40069  *MAN                                                        12JUN78
* 40070  //          PROG=SAIL,                                     X 12JUN78
* 40071  //          P1='SYSOUT=A',                                 X 12JUN78
* 40072  //          P2='SYSOUT=A',                                 X 12JUN78
* 40073  //          REG=175K,                                      X 12JUN78
* 40074  //          SAILBLK=800,                                   X 12JUN78
* 40075  //          SAILR=80,                                      X 12JUN78
* 40076  //          SAIL1='DSN=&&SAILCD,UNIT=SYSDA,SPACE=(TRK,(10,20))',  X 12JUN78
* 40077  //          SAIL2=DUMMY,                                   X 12JUN78
* 40078  //          SAIL3=DUMMY,                                   X 12JUN78
* 40079  //          SAIL4=DUMMY,                                   X 12JUN78
* 40080  //          SAIL5=DUMMY,                                   X 12JUN78
* 40081  //          SAIL6=DUMMY,                                   X 12JUN78
* 40082  //          SAIL7=DUMMY,                                   X 12JUN78
* 40083  //          SAIL8=DUMMY,                                   X 12JUN78
* 40084  //          SAIL9=DUMMY,                                   X 12JUN78
* 40085  //          SAIL10=DUMMY,                                  X 12JUN78
* 40086  //          SAIL11=DUMMY,                                  X 12JUN78
* 40087  //          SAIL12=DUMMY,                                  X 12JUN78
* 40088  //          SAIL13=DUMMY,                                  X 12JUN78
* 40089  //          SAIL14=DUMMY,                                  X 12JUN78
* 40090  //          SAIL15=DUMMY,                                  X 12JUN78
* 40091  //          SCRTC=SYSDA,                                   X 12JUN78
* 40092  //          STIME='(2,0)',                                 X 12JUN78
* 40093  //          TEMPBLK=7294,                                  X 12JUN78
* 40094  //          TEMPLRL=7290                                     12JUN78
* 40095  //SAIL    EXEC PGM=&PROG,TIME=&STIME,REGION=&REG             12JUN78
* 40096  //*                                                         12JUN78
* 40097  //STEPLIB   DD DSN=&LIBPRE&LIB,                           X 12JUN78
* 40098  //             UNIT=&LIBU,                                 X 12JUN78
* 40099  //             VOL=&LIBVOL,                                X 12JUN78
* 40100  //             DISP=SHR                                      12JUN78
* 40101  //*                                                         12JUN78
* 40102  //FT01F001  DD &NEWDUM.DSN=&NEWPRE.&NEWAGEN,              X 12JUN78
* 40103  //             UNIT=&NEWU.                                 X 12JUN78
```

```
*  40104        //              LABEL=(&FILN,&LABN,&NEWP,OUT,&EXP),                X 12JUN78
*  40105        //              DISP=&NEWDS,                                       X 12JUN78
*  40106        //              VOL=&NEWVOL,                                       X 12JUN78
*  40107        //              SPACE=&NEWSPC,                                     X 12JUN78
*  40108        //              DCB=&NEWDCB                                          12JUN78
*  40109        //*                                                                  12JUN78
*  40110        //FT02F001  DD &GENDUM.DSN=&OLDPRE&OLD&GENO,                       X 12JUN78
*  40111        //              UNIT=&OLDU,                                        X 12JUN78
*  40112        //              LABEL=(&FILO,&LABO,,IN),                           X 12JUN78
*  40113        //              DISP=&OLDDS,                                       X 12JUN78
```

```
*  40114        //              VOL=&OLDVOL,                                    X 12JUN78
*  40115        //              DCB=&OLDDCB                                       12JUN78
*  40116        //*                                                               12JUN78
*  40117        //FT03F001  DD UNIT=&SCRTC,                                    X 12JUN78
*  40118        //              DISP=(NEW,DELETE),                              X 12JUN78
*  40119        //              DCB=(RECFM=VBS,LRECL=&CHNLRL,BLKSIZE=&CHNBLK),  X 12JUN78
*  40120        //              SPACE=(TRK,(20,20))                               12JUN78
*  40121        //*                                                               12JUN78
*  40122        //FT04F001  DD &P2                                                12JUN78
*  40123        //*                                                               12JUN78
*  40124        //FT05F001  DD DDNAME=INPUT                                       12JUN78
*  40125        //*                                                               12JUN78
*  40126        //FT06F001  DD &P1                                                12JUN78
*  40127        //*                                                               12JUN78
*  40128        //FT0CF001  DD &SAIL1,                                         X 12JUN78
*  40129        //              &DISP1,                                         X 12JUN78
*  40130        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40131        //*                                                               12JUN78
*  40132        //FT08F002  DD &SAIL2,                                         X 12JUN78
*  40133        //              &DISP2,                                         X 12JUN78
*  40134        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40135        //*                                                               12JUN78
*  40136        //FT08F003  DD &SAIL3,                                         X 12JUN78
*  40137        //              &DISP3,                                         X 12JUN78
*  40138        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40139        //*                                                               12JUN78
*  40140        //FT08F004  DD &SAIL4,                                         X 12JUN78
*  40141        //              &DISP4,                                         X 12JUN78
*  40142        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40143        //*                                                               12JUN78
*  40144        //FT08F005  DD &SAIL5,                                         X 12JUN78
*  40145        //              &DISP5,                                         X 12JUN78
*  40146        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40147        //*                                                               12JUN78
*  40148        //FT08F006  DD &SAIL6,                                         X 12JUN78
*  40149        //              &DISP6,                                         X 12JUN78
*  40150        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40151        //*                                                               12JUN78
*  40152        //FT08F007  DD &SAIL7,                                         X 12JUN78
*  40153        //              &DISP7,                                         X 12JUN78
*  40154        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40155        //*                                                               12JUN78
*  40156        //FT08F008  DD &SAIL8,                                         X 12JUN78
*  40157        //              &DISP8,                                         X 12JUN78
*  40158        //              DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40159        //*                                                               12JUN78
*  40160        //FT08F009  DD &SAIL9,                                         X 12JUN78
```

```
*  40161        //             &DISP9,                                                    X 12JUN78
*  40162        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)                 12JUN78
*  40163        //*                                                                        12JUN78
*  40164        //FT08F010  DD &SAIL10,                                                   12JUN78
*  40165        //             &DISP10,                                                   X 12JUN78
*  40166        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)                 12JUN78
*  40167        //*                                                                        12JUN78
*  40168        //FT08F011  DD &SAIL11,                                                   X 12JUN78
*  40169        //             &DISP11,                                                   X 12JUN78
*  40170        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)                 12JUN78
```

```
*  40171        //*                                                                  12JUN78
*  40172        //FT08F012  DD &SAIL12,                                            X 12JUN78
*  40173        //             &DISP12,                                            X 12JUN78
*  40174        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)          12JUN78
*  40175        //*                                                                  12JUN78
*  40176        //FT08F013  DD &SAIL13,                                            X 12JUN78
*  40177        //             &DISP13,                                            X 12JUN78
*  40178        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)          12JUN78
*  40179        //*                                                                  12JUN78
*  40180        //FT08F014  DD &SAIL14,                                             12JUN78
*  40181        //             &DISP14,                                            X 12JUN78
*  40182        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)          12JUN78
*  40183        //*                                                                  12JUN78
*  40184        //FT08F015  DD &SAIL15,                                            X 12JUN78
*  40185        //             &DISP15,                                            X 12JUN78
*  40186        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)          12JUN78
*  40187        //*                                                                  12JUN78
*  40188        //FT09F001  DD &ALTI                                                12JUN78
*  40189        //*                                                                  12JUN78
*  40190        //FT10F001  DD UNIT=&SCRTC,                                        X 12JUN78
*  40191        //             DISP=(NEW,DELETE),                                  X 12JUN78
*  40192        //             SPACE=(&PRCL,(&PRCN))                                 12JUN78
*  40193        //*                                                                  12JUN78
*  40194        //FT11F001  DD &NEWDUM.UNIT=&SCRTC,                                X 12JUN78
*  40195        //             DISP=(NEW,DELETE),                                  X 12JUN78
*  40196        //             DCB=(RECFM=VBS,LRECL=&TEMPLRL,BLKSIZE=&TEMPBLK),    X 12JUN78
*  40197        //             SPACE=&NEWSPC                                         12JUN78
*  40198        //*                                                                  12JUN78
*  40199        //FT12F001  DD UNIT=&SCRTC,                                        X 12JUN78
*  40200        //             DISP=(NEW,DELETE),                                  X 12JUN78
*  40201        //             DCB=(RECFM=FB,LRECL=120,BLKSIZE=1200),              X 12JUN78
*  40202        //             SPACE=(TRK,(2,2))                                     12JUN78
*  40203        //*                                                                  12JUN78
```

```
*  40204        *P SGEN                                                                  12JUN78
*  40205        =                                                                        12JUN78
*  40206        =       THE FOLLOWING PROC IS USED TO GENERATE A NEW VERSION OF SAIL     12JUN78
*  40207        =       ON AN IBM 360 OR 370 SYSTEM                                      12JUN78
*  40208        =                                                                        12JUN78
*  40209        //* * * * * * * * * * * * * * S G E N * * * * * * * * * * * * * * * * * 12JUN78
*  40210        //*                                                                      12JUN78
*  40211        //* NAME                                                                 12JUN78
*  40212        //*   SGEN                                                               12JUN78
*  40213        //*                                                                      12JUN78
*  40214        //* FORMAT                                                               12JUN78
*  40215        //*   //         EXEC    SGEN,...(OPTIONAL PARAMETERS AS REQUIRED)     12JUN78
*  40216        //*   //SAIL.INPUT  DD  *                                              12JUN78
*  40217        //*                                                                      12JUN78
*  40218        //*       SAIL CHANGES                                                   12JUN78
*  40219        //*                                                                      12JUN78
*  40220        //*   /*                                                                 12JUN78
*  40221        //*                                                                      12JUN78
*  40222        //* FUNCTION                                                             12JUN78
*  40223        //*   THIS PROCEDURE EXECUTES THE SAIL PROGRAM AND GENERATES A NEW      12JUN78
*  40224        //*   SAIL LIBRARY                                                       12JUN78
*  40225        //*                                                                      12JUN78
*  40226        //* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 12JUN78
*  40227        //SGEN    PROC LIB=HULLIB,                                          X 12JUN78
*  40228        //             LIBPRE='SAIL.',                                      X 12JUN78
*  40229        //             LIBU=,                                               X 12JUN78
*  40230        //             LIBVOL=,                                             X 12JUN78
*  40231        //             NLIB=HULLIBN,                                        X 12JUN78
*  40232        //             NLIBPRE='SAIL.',                                     X 12JUN78
*  40233        //             NLIBU=SYSDA,                                         X 12JUN78
*  40234        //             NLIBVOL=,                                            X 12JUN78
*  40235        //             NLIBDS='(NEW,CATLG)',                                X 12JUN78
*  40236        //             LDSPACE='(CYL,(20,5,5))',                            X 12JUN78
*  40237        //             APROG=IFOX00,                                        X 12JUN78
*  40238        //             AREG=187K,                                           X 12JUN78
*  40239        //             ATIME='(2,0)',                                       X 12JUN78
*  40240        //             APARM='LOAD,NODECK',                                 X 12JUN78
*  40241        //             AMACL='SYS1.MACLIB',                                 X 12JUN78
*  40242        //             AP1='SYSOUT=A',                                      X 12JUN78
*  40243        //             CHNBLK=3521,                                         X 12JUN78
*  40244        //             CHNLRL=3517,                                         X 12JUN78
*  40245        //             FILO=,                                               X 12JUN78
*  40246        //             FPARM=MAP,                                           X 12JUN78
*  40247        //             FPROG=IFEAAB,                                        X 12JUN78
*  40248        //             FREG=512K,                                           X 12JUN78
*  40249        //             FP1='SYSOUT=A',                                      X 12JUN78
*  40250        //             FSPACE='(CYL,(10,5),RLSE)',                          X 12JUN78
```

```
* 40251    //          FTIME='(1,0)',                                  X 12JUN78
* 40252    //          GENO='(0)',                                     X 12JUN78
* 40253    //          LABO=,                                          X 12JUN78
* 40254    //          LPARM='MAP',                                    X 12JUN78
* 40255    //          LPROG=IEWL,                                     X 12JUN78
* 40256    //          LREG=250K,                                      X 12JUN78
* 40257    //          LPI='SYSOUT=A',                                 X 12JUN78
* 40258    //          LTIME='(0,45)',                                 X 12JUN78
* 40259    //          SGENSP='(CYL,(5,5),RLSE)',                      X 12JUN78
* 40260    //          OLD=SAIL,                                       X 12JUN78
```

```
*  40261        //              OLDDCB=,                                                X 12JUN78
*  40262        //              OLDDS=SHR,                                              X 12JUN78
*  40263        //              OLDPRE='SAIL.',                                         X 12JUN78
*  40264        //              OLDU=,                                                  X 12JUN78
*  40265        //              OLDVOL=,                                                X 12JUN78
*  40266        *AUTO                                                                     12JUN78
*  40267        //              PRCN=_NUMREC_,                                          X 12JUN78
*  40268        //              PRCL=_8*LBUFF_,                                         X 12JUN78
*  40269        *MAN                                                                      12JUN78
*  40270        //              PS1='SYSOUT=A',                                         X 12JUN78
*  40271        //              PS2='SYSOUT=A',                                         X 12JUN78
*  40272        //              SAILBLK=800,                                            X 12JUN78
*  40273        //              SAILR=80,                                               X 12JUN78
*  40274        //              SCRTC=SYSDA,                                            X 12JUN78
*  40275        //              SPROG=SAIL,                                             X 12JUN78
*  40276        //              SREG=175K,                                              X 12JUN78
*  40277        //              STIME='(2,0)',                                          X 12JUN78
*  40278        //              WORKSP='(CYL,(5,5))'                                      12JUN78
*  40279        //*-------------------------------------------------------------------*  12JUN78
*  40280        //SAIL    EXEC PGM=&SPROG,TIME=&STIME,REGION=&SREG                       12JUN78
*  40281        //*                                                                      12JUN78
*  40282        //STEPLIB   DD DSN=&LIBPRE&LIB,                                       X 12JUN78
*  40283        //             UNIT=&LIBU,                                            X 12JUN78
*  40284        //             VOL=&LIBVOL,                                           X 12JUN78
*  40285        //             DISP=SHR                                                 12JUN78
*  40286        //*                                                                      12JUN78
*  40287        //FT01F001  DD DUMMY                                                    12JUN78
*  40288        //*                                                                      12JUN78
*  40289        //FT02F001  DD DSN=&OLDPRE&OLD&GENO,                                  X 12JUN78
*  40290        //             UNIT=&OLDU,                                            X 12JUN78
*  40291        //             LABEL=(&FILO,&LABO,,IN),                               X 12JUN78
*  40292        //             DISP=&OLDDS,                                           X 12JUN78
*  40293        //             VOL=&OLDVOL,                                           X 12JUN78
*  40294        //             DCB=&OLDDCB                                              12JUN78
*  40295        //*                                                                      12JUN78
*  40296        //FT03F001  DD UNIT=&SCRTC,                                           X 12JUN78
*  40297        //             DISP=(NEW,DELETE),                                     X 12JUN78
*  40298        //             DCB=(RECFM=VBS,LRECL=&CHNLRL,BLKSIZE=&CHNBLK),         X 12JUN78
*  40299        //             SPACE=(TRK,(20,20))                                      12JUN78
*  40300        //*                                                                      12JUN78
*  40301        //FT04F001  DD &PS2,                                                  X 12JUN78
*  40302        //             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)                   12JUN78
*  40303        //*                                                                      12JUN78
*  40304        //FT05F001  DD DDNAME=INPUT                                             12JUN78
*  40305        //*                                                                      12JUN78
*  40306        //FT06F001  DD &PS1,                                                  X 12JUN78
*  40307        //             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)                   12JUN78
```

```
*  40308       //*                                                                    12JUN78
*  40309       //FT08F001  DD DSN=&&FSAIL,                                          X 12JUN78
*  40310       //             DISP=(NEW,PASS),                                      X 12JUN78
*  40311       //             UNIT=&SCRTC,                                          X 12JUN78
*  40312       //             SPACE=&SGENSP,                                        X 12JUN78
*  40313       //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)            12JUN78
*  40314       //*                                                                    12JUN78
*  40315       //FT08F002  DD DSN=&&ASAIL,                                          X 12JUN78
*  40316       //             DISP=(NEW,PASS),                                      X 12JUN78
*  40317       //             UNIT=&SCRTC,                                          X 12JUN78
```

```
*  40318        //             SPACE=&SGENSP,                                  X 12JUN78
*  40319        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40320        //*                                                             12JUN78
*  40321        //FT08F003  DD DSN=&&LSAIL,                                    X 12JUN78
*  40322        //             DISP=(NEW,PASS),                                X 12JUN78
*  40323        //             UNIT=&SCRTC,                                    X 12JUN78
*  40324        //             SPACE=&SGENSP,                                  X 12JUN78
*  40325        //             DCB=(RECFM=FB,LRECL=&SAILR,BLKSIZE=&SAILBLK)      12JUN78
*  40326        //FT09F001  DD DUMMY                                            12JUN78
*  40327        //*                                                             12JUN78
*  40328        //*                                                             12JUN78
*  40329        //FT10F001  DD UNIT=&SCRTC,                                    X 12JUN78
*  40330        //             DISP=(NEW,DELETE),                              X 12JUN78
*  40331        //             SPACE=(&PRCL,(&PRCN))                             12JUN78
*  40332        //*                                                             12JUN78
*  40333        //FT11F001  DD DUMMY                                            12JUN78
*  40334        //*                                                             12JUN78
*  40335        //FT12F001  DD DUMMY                                            12JUN78
*  40336        //*-------------------------------------------------------------* 12JUN78
*  40337        //ASM     EXEC PGM=&APROG,PARM='&APARM',REGION=&AREG,           X 12JUN78
*  40338        //             TIME=&ATIME,                                    X 12JUN78
*  40339        //             COND=(8,LT,SAIL)                                  12JUN78
*  40340        //*                                                             12JUN78
*  40341        //SYSLIB    DD DSN=&AMACL,DISP=SHR                              12JUN78
*  40342        //*                                                             12JUN78
*  40343        //SYSIN     DD DSN=&&ASAIL,DISP=(OLD,DELETE)                    12JUN78
*  40344        //*                                                             12JUN78
*  40345        //SYSPRINT  DD &AP1,                                            12JUN78
*  40346        //             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)           12JUN78
*  40347        //*                                                             12JUN78
*  40348        //SYSUT1    DD UNIT=&SCRTC,SPACE=&WORKSP                        12JUN78
*  40349        //*                                                             12JUN78
*  40350        //SYSUT2    DD UNIT=&SCRTC,SPACE=&WORKSP                        12JUN78
*  40351        //*                                                             12JUN78
*  40352        //SYSUT3    DD UNIT=&SCRTC,SPACE=&WORKSP                        12JUN78
*  40353        //*                                                             12JUN78
*  40354        //SYSPUNCH  DD SYSOUT=B                                         12JUN78
*  40355        //*                                                             12JUN78
*  40356        //SYSGO     DD  DDNAME=SYSLIN                                   12JUN78
*  40357        //*                                                             12JUN78
*  40358        //SYSLIN    DD DSN=&&LOADSET,                                 X 12JUN78
*  40359        //             UNIT=&SCRTC,                                    X 12JUN78
*  40360        //             DISP=(MOD,PASS),                                X 12JUN78
*  40361        //             SPACE=&FSPACE,                                  X 12JUN78
*  40362        //             DCB=(RECFM=FB,LRECL=80,BLKSIZE=1600)              12JUN78
*  40363        //*-------------------------------------------------------------* 12JUN78
*  40364        //FORT    EXEC PGM=&FPROG.                                    X 12JUN78
```

```
*  40365      //             REGION=&FREG,                                  X 12JUN78
*  40366      //             PARM='NAME=SAIL,&FPARM',                       X 12JUN78
*  40367      //             TIME=&FTIME,                                   X 12JUN78
*  40368      //             COND=(8,LT,SAIL)                                 12JUN78
*  40369      //*                                                              12JUN78
*  40370      //SYSPRINT  DD &FP1,                                          X 12JUN78
*  40371      //             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)           12JUN78
*  40372      //*                                                              12JUN78
*  40373      //SYSLIN    DD DSN=&&LOADSET,                                 X 12JUN78
*  40374      //             DISP=(MOD,PASS),                               X 12JUN78
```

```
*  40375       //             UNIT=&SCRTC,                                          X 12JUN78
*  40376       //             SPACE=&FSPACE,                                        X 12JUN78
*  40377       //             DCB=(RECFM=BF,LRECL=80,BLKSIZE=1600)                    12JUN78
*  40378       //*                                                                   12JUN78
*  40379       //SYSIN     DD DSN=&&FSAIL,DISP=(OLD,DELETE)                         12JUN78
*  40380       //*                                                                   12JUN78
*  40381       //SYSUT1    DD UNIT=&SCRTC,SPACE=&WORKSP                             12JUN78
*  40382       //*                                                                   12JUN78
*  40383       //SYSUT2    DD UNIT=&SCRTC,SPACE=&WORKSP                             12JUN78
*  40384       //*                                                                   12JUN78
*  40385       //SYSTERM   DD DUMMY                                                 12JUN78
*  40386       //*-------------------------------------------------------------------* 12JUN78
*  40387       //LKED    EXEC PGM=&LPROG,REGION=&LREG,TIME=&LTIME,                  X 12JUN78
*  40388       //             COND=((4,LT,FORT),(8,LT,SAIL),(8,LT,ASM)),            X 12JUN78
*  40389       //             PARM='OVLY,&LPARM'                                      12JUN78
*  40390       //*                                                                   12JUN78
*  40391       //SYSPRINT  DD &LP1,                                                 X 12JUN78
*  40392       //             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)                  12JUN78
*  40393       //*                                                                   12JUN78
*  40394       //SYSLIB    DD DSN=SYS1.FORTLIB,DISP=SHR                             12JUN78
*  40395       //*                                                                   12JUN78
*  40396       //SYSUT1    DD UNIT=&SCRTC,SPACE=(1024,(200,20))                     12JUN78
*  40397       //*                                                                   12JUN78
*  40398       //SYSLMOD   DD DSN=&NLIBPRE&NLIB,                                    X 12JUN78
*  40399       //             UNIT=&NLIBU,                                          X 12JUN78
*  40400       //             VOL=&NLIBVOL,                                         X 12JUN78
*  40401       //             DISP=&NLIBDS,                                         X 12JUN78
*  40402       //             SPACE=&LDSPACE                                          12JUN78
*  40403       //*                                                                   12JUN78
*  40404       //SYSLIN    DD DSN=&&LOADSET,DISP=(OLD,DELETE)                       12JUN78
*  40405       //             DD DSN=&&LSAIL,DISP=(OLD,DELETE)                        12JUN78
*  40406       //*-------------------------------------------------------------------* 12JUN78
```

```
*  50000        *B IBM-EXAMPLES                                    12JUN78
```

```
    50001        *P  EXAMPLES                                                    16MAY75
    50002        =                                                               16MAY75
    50003        =       GENERATE A NEW SAIL PROGRAM                             16MAY75
    50004        =                                                               16MAY75
*   50005        //NEW EXEC SGEN                                                 12JUN78
*   50006        //SAIL.INPUT  DD  *                                             12JUN78
*   50007        #                                                               12JUN78
*   50008        #       SAIL CHANGES                                            12JUN78
*   50009        #                                                               12JUN78
*   50010        /*                                                              12JUN78
    50011        =                                                               16MAY75
    50012        =                                                               16MAY75
```

```
50013       *P                                                          16MAY75
50014       =                                                           16MAY75
50015       =       NORMAL RUN WITH ONLY ONE FILE WRITTEN OF CARDS      16MAY75
50016       =                                                           16MAY75
* 50017     //RN EXEC SAIL                                             12JUN78
50018       //SAIL.INPUT DD   *                                        16MAY75
50019         SAIL OPTIONS    ...         ENDOPTIONS                    16MAY75
50020       =                                                           16MAY75
50021       =       SAIL CHANGE CARDS                                   16MAY75
50022       =                                                           16MAY75
50023       /*                                                          16MAY75
```

```
     50024        *P                                                          16MAY75
     50025        =                                                           16MAY75
     50026        =       COPY A SAIL FILE AND CHANGE DEFAULT OPTIONS          16MAY75
     50027        =                                                           16MAY75
*    50028        //COPY  EXEC SAIL,GENN='.NEW',NEWPRE=,NEWDUM=,              12JUN78
*    50029        //        NEWVOL='SER=SAILVOL',NEWU=2314                    12JUN78
     50030        //SAIL.INPUT DD   *                                         16MAY75
     50031          SAIL COPY  OPTIONS  ...        ENDOPTIONS                 16MAY75
     50032        /*                                                          16MAY75
```

```
  50033        *P                                                                16MAY75
  50034        =                                                                 16MAY75
  50035        =        CONVERT A PACKED FILE TO A CODED FILE                    16MAY75
  50036        =        CODED FILE IS FIRST FILE ON A TAPE                       16MAY75
  50037        =                                                                 16MAY75
* 50038        //CONV EXEC SAIL,FILN=1,LABN=NL,NEWDS='(NEW,KEEP)',               12JUN78
  50039        //       NEWDCB='(RECFM=FB,LRECL=120,BLKSIZE=5040)',              16MAY75
  50040        //       GENN='.TAPE',NEWDUM=,                                    16MAY75
  50041        //       NEWVOL='SER=SAILCD',NEWU=TAPE9                           16MAY75
  50042        //SAIL.INPUT DD   *                                               16MAY75
  50043            SAIL COPY CONVERT                                            16MAY75
  50044        /*                                                                16MAY75
```

AD-A068 519 AIR FORCE WEAPONS LAB KIRTLAND AFB NM F/G 9/2
SAIL, AN AUTOMATED APPROACH TO SOFTWARE DEVELOPMENT AND MANAGEM--ETC(U)
JAN 79 L P GABY, D C GRAHAM, C E RHOADES

UNCLASSIFIED AFWL-TR-78-80 SBIE-AD-E200 258 NL

2 OF 2
AD
A068519

END
DATE
FILMED
6-79
DDC

```
50045       *P                                                              16MAY75
50046       =                                                               16MAY75
50047       =       CONVERT A CODED FILE TO PACKED FILE                     16MAY75
50048       =       CODED FILE IS FIRST FILE ON A TAPE                      16MAY75
50049       =                                                               16MAY75
*50050      //CONV EXEC SAIL,FILO=1,LABO=NL,OLDDS='(OLD,KEEP)',             12JUN78
50051       //       OLDDCB='(RECFM=FB,LRECL=120,BLKSIZE=5040)',            16MAY75
50052       //       GENO='.TAPE',GENN='.G0044V00',NEWDUM=,                 16MAY75
50053       //       OLDVOL='SER=SAILCD',OLDU=TAPE9,NEWVOL='SER=SAIL'       16MAY75
50054       //SAIL.INPUT DD   *                                             16MAY75
50055          SAIL CONVERT COPY                                            16MAY75
50056       /*                                                              16MAY75
```

```
    50057       *P                                                          16MAY75
    50058       =                                                           16MAY75
    50059       =       UPDATE OF A SAIL FILE                               16MAY75
    50060       =                                                           16MAY75
*   50061       //UPDATE EXEC SAIL,NEWDUM=,                                 12JUN78
    50062       //       NEWVOL='SER=SAIL',NEWU=2314                        16MAY75
    50063       //SAIL.INPUT DD   *                                         16MAY75
    50064         SAIL UPDATE SE                                            16MAY75
    50065       =                                                           16MAY75
    50066       =       SAIL CHANGE CARDS                                   16MAY75
    50067       =                                                           16MAY75
    50068       /*                                                          16MAY75
```

```
**************************
*                        *
*      END OF LIST       *
*                        *
**************************
```

GLOSSARY OF TERMS

block - On the SAIL file(s) output during a NORMAL run, a set of card images that belong to one logical portion of the file(s). On CDC, this is a logical section; on IBM, those cards identified by FT08F00n for some n; on Honeywell, one of the files 15, 16, 17, etc.

On a library, a group of cards with the same block number.

There is no relationship between these two meanings of block.

card - One physical card or one line of information corresponding to a physical card on INPUT.

One line appearing on the SAIL file from a NORMAL run.

In general, the set of characters read or written as a single record by formatted FORTRAN I/O statements.

The part of a SAIL record containing data or a SAIL directive.

card-image - The same as card, but usually with the implication that the information is stored on some device other than physical cards.

change set - A consecutive set of cards appearing on INPUT, the first of which is one of the SAIL directives *A, *C, *D, or *I, and the last, the card preceding the next such card or the end-of-file.

COPY - A SAIL execution mode in which an old library is copied to a new one, usually with a conversion between the coded format used for intermachine transport and the packed format used for SAIL records on a particular machine.

delimiter - A character used to mark the beginning or end of a character string.

On free-field cards, one of the characters blank, comma, or equal sign, which are referred to as SAIL delimiters.

One of the characters (" or =), $, or (_ or →), which are used as delimiters on some data cards and directives.

directive - A character string following an asterisk in column one, which is recognized by SAIL as a command to perform an action. The command may be modified or ignored due to the effect of SAIL requests. Directives fall into two classes: general and executive.

directory - Those SAIL records occurring after the *DIR directive and before the *EDIR directive.

A Type of listing in which only SAIL records occurring in directory sections are printed.

element - On free-field chards, a contiguous string of characters appearing after the beginning of the card or a SAIL delimeter and before the end of the scanned portion of the card (column 72 by default) or another SAIL delimiter.

ERROR - The name used in this manual for a file on which SAIL writes error messages.

executive - The processor section that operates during NORMAL runs to select cards within PROGRAMs to be written to the SAIL file.

field - A contiguous set of columns on a card or card-image. Field is used as a general term to describe such a set that has been set aside for a particular purpose by a computer program.

free-field - The characteristic of the request area on INPUT and of the cards or SAIL records which are recognized as SAIL directives, that elements can be separated by any number of SAIL delimiters without having either to appear in particular columns or a certain number of columns after the previous element.

GENERATE - A SAIL execution mode in which the cards appearing on INPUT following the request area are used to create a new library.

global list type - On LIST runs, either directory or full. (The latter causes SAIL to list all cards in the selected PROGRAMs.) This is used as the list type for any PROGRAM that is not explicitly selected opposite.

identifier - A pair of integers used to identify SAIL records. One is the block number and the other the card number within the block.

include - On a NORMAL run, to place the contents of a PROC on the SAIL file.

INPUT - The name used in this manual for the file SAIL uses for control information and changes to the library.

library - A file containing SAIL records which together contain each possible data card that can appear on the SAIL file during a NORMAL run along with directives to select among them under OPTION control. Default lists of PROGRAMs and OPTIONs are also stored on a library. SYSTEM and VERSION parameters allow the user to verify the identity of the library.

line - In this manual, a term used synonymously with SAIL record.

LIST - A SAIL execution mode that lists selected PROGRAMs from a library.

list type - For each PROGRAM on a library during a LIST run, the choice between listing the entire PROGRAM (full) or just the directory cards (directory).

macro processing - During inclusion of a PROC, the substitution of new character strings given on the *INCLUDE directive for those indicated on the *PROC directive. The characters to be replaced must be flagged inside the PROC.

mode - The choice of processing function to be accomplished during a SAIL execution.

name field - The second element or set of elements on a card interpreted as a SAIL executive directive.

NEW - The name used in this manual for the new library file created on GENERATE, UPDATE, and COPY runs.

NORMAL - A SAIL execution mode in which the executive processor selects cards from the library under OPTION control and writes them to the SAIL file.

OLD - The name used in this manual for the library file used in all modes as a basis for processing.

operand field - The third set of elements on a card or cards containing a SAIL executive directive. If it is not omitted, it is usually a logical expression combining logical units composed of comparison tests or TRUE-FALSE determinations on OPTIONs. For the *DEFN directive, it is an arithmetic expression involving OPTIONs and constants.

OPTION - A pair of parameters consisting of a character string (the name) and a nonnegative integer value used to control card selections and character string substitutions during NORMAL runs.

OUTPUT - The name used in this manual for the file on which printable output is placed.

PROC - A set of SAIL records identified by the SAIL executive directive *PROC so they can be included at any point by an *INCLUDE directive with a matching name field.

PROCF - The name used in this manual for a file used to store the PROCs from the PROLOGUE during NORMAL runds on CDC machines.

PROGRAM - The SAIL records beginning with one *B directive and ending with either the end of the library or the card preceding the next *B.

PROLOGUE - The SAIL records preceding the first PROGRAM on a library.

PUNCH - A SAIL execution mode in which the card portion from SAIL records within the selected PROGRAMs are written to the SAIL file ready to punch or to use source cards or changes for another library.

request - An element in the request area used to control processing during SAIL execution.

request area - The cards on INPUT preceding the first card (if any) with an asterisk in column one, except on GENERATE runs where the card with the GENERATE request terminates the request area.

request pair - Two consecutive requests, which must occur on the same card, the first of which specifies a parameter to be set, and the second, the value to be given it.

SCAN - A SAIL execution mode in which the library is searched for SAIL records containing any of a set of character strings designated by the user.

SAIL - The program described in this manual. The name used in this manual for the file on which task-oriented card images are written during a NORMAL run or on which source cards are written during a PUNCH run.

SAIL delimiter - One of three characters (blank, comma, or equal sign) used to separate elements on cards or SAIL records written in free-format.

SAIL directive (general) - A directive recognized by SAIL that is processed not only on NORMAL runs, but during executions in at least one other mode.

SAIL executive directive - A directive recognized by SAIL that is processed (by the executive processor) only on NORMAL runs. In other modes, such a SAIL record is trated as data.

SAIL record - One card-image along with its identifier stored on a library or printed on OUTPUT.

specification area - Those requests that fall between the requests OPTIONS and ENDOPTIONS, DELOPTIONS and ENDOPTIONS, or PROGRAM and ENDPROGRAM. If the terminating directive does not appear sooner, the specification area ends at the end of the request area.

SYSTEM - A character string stored on a library that identifies it by name.

The request that specifies the name of a library.

table - A subset of the OPTION list starting with the OPTION after the one designated as the head and consisting of as many consecutive OPTIONs as the value of the head.

task - A problem for which the user needs a card-image file either in the form of input to a compiler or as data.

UPDATE - A SAIL execution mode in which changes to an old library are made permanent by creating a new library.

verb field - The first element of an executive directive. The first character must be an asterisk and it must appear in column one of the card or card-image portion of a SAIL record.

VERSION - An integer stored on the library to distinguish it from other libraries with the same SYSTEM name.

The request that specifies this integer for a library.