LY28-1740-1
File No. S370-36

Program Product

MVS/Extended Architecture System Logic Library: Scheduler JCL Facility

MVS/System Product:

JES3 Version 2	5665-291
JES2 Version 2	5740-XC6

IBM

Second Edition (September, 1989)

This is a major revision of, and obsoletes, LY28-1740-0. See the Summary of Amendments following the Contents for a summary of the changes made to this manual. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

This edition applies to Version 2 Release 2 of MVS/System Product program numbers 5665-291 and 5740-XC6 and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370 Bibliography, GC20-0001, for the editions that are applicable and current.

References in this publication to IBM products or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product in this publication is not intended to state or imply that only IBM's product may be used. This statement does not expressly or implicitly waive any intellectual property right IBM may hold in any product mentioned herein. Any functionally equivalent product may be used instead.

Publications are not stocked at the address given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Information Development, Department D58, Building 921-2, PO Box 950, Poughkeepsie, NY 12602. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PREFACE

The MVS/Extended Architecture System Logic Library is intended for people who debug or modify the MVS control program. It describes the logic of most MVS control program functions that are performed after master scheduler initialization completes. For detailed information about the MVS control program prior to this point, refer to MVS/Extended Architecture System Initialization Logic. For general information about the MVS control program and the relationships among the components that make up the MVS control program, refer to the MVS/Extended Architecture Overview. To obtain the names of publications that describe some of the components not in the System Logic Library, refer to the section Corequisite Reading in the Master Preface in MVS/Extended Architecture System Logic Library: Master Index.

TRADEMARKS

The following are trademarks of International Business Machines Corporation.

- DFSMS(TM)
- Enterprise Systems Architecture/370(TM)
- ESA/370(TM)
- MVS/ESA(TM)
- MVS/DFP(TM)
- MVS/SP(TM)
- MVS/XA(TM)
- Processor Resource/Systems Manager(TM)
- PR/SM(TM)
- ES/3090(TM)
- Enterprise Systems/3090(TM)
- VM/XA(TM)
- Virtual Machine/Extended Architecture(TM)

HOW THE LIBRARY IS ORGANIZED

SET OF BOOKS

The System Logic Library consists of a set of books. Two of the books provide information that is relevant to the entire set of books:

1. The MVS/Extended Architecture System Logic Library: Master Index contains the master preface master index for the other books in the set.

2. The MVS/Extended Architecture System Logic Library: Module Descriptions contains module descriptions for all of the modules in the components documented in the System Logic Library and an index.

Each of the other books (referred to as component books) in the set contains its own table of contents and index, and describes the logic of one of the components in the MVS control program.

ORGANIZATION OF THE COMPONENTS

Most component books contain information about one component in the MVS control program. However, some component books (such as System Logic Library: Initiator/Terminator) contain more than one component if the components are closely related, frequently referenced at the same time, and not so large that they require a book of their own.

A three or four character mnemonic is associated with each component book and is used in all diagram and page numbers in that book. For example, the mnemonic ASM is associated with the book MVS/Extended Architecture System Logic Library: Auxiliary Storage Management. All diagrams in this book are identified as Diagram ASM-n, and all pages as ASM-n, where n represents the specific diagram or page number. Whenever possible, the existing component acronym is used as the mnemonic for the component book. The Table of Book Titles in the Master Preface in MVS/Extended Architecture System Logic Library: Master Index lists the book titles, the components included in each book (if a book contains more than one component), the mnemonics for the books, and the order number for each book.

HOW TO USE THE LIBRARY

To help you use this library efficiently, the following topics cover

- How to find information using book titles and the master index
- What types of information are provided for each component
- How to obtain further information about other books in the System Logic Library

FINDING INFORMATION USING THE BOOK TITLES

As you become familiar with the book titles, MVS component names and mnemonics, and the book contents, you will be able to use the System Logic Library as you would an encyclopedia and go directly to the book that you need. We recommend that you group the books in alphabetical order for easy reference, or, if you are familiar with MVS, that you group the books by related functions.

The Table of Book Titles in the Master Preface in MVS/Extended Architecture System Logic Library: Master Index contains a list of book titles and mnemonics. It provides a quick reference to all the books, and their corresponding components, in the System Logic Library.

FINDING INFORMATION USING THE MASTER INDEX

If you are not sure which book contains the information you are looking for, you can locate the book and the page on which the information appears by using the master index in System Logic Library: Master Index. For the component books, the page number in an index entry consists of the mnemonic for the component and the page number; for System Logic Library: Module Descriptions, the page number consists of the mnemonic "MOD" and the page number.

For example:

ASM-12	refers to MVS/Extended Architecture System Logic Library: Auxiliary Storage Management, page ASM-12.
MOD-245	refers to MVS/Extended Architecture System Logic Library: Module Descriptions, page MOD-245.

INFORMATION PROVIDED FOR MOST COMPONENTS

The following information is provided for most of the components described in the System Logic Library.

1. An introduction that summarizes the component's function

2. Control block overview figures that show significant fields and the chaining structure of the component's control blocks

3. Process flow figures that show control flow between the component's object modules

4. Module information that describes the functional organization of a program. This information can be in the form of:

   - Method-of-Operation diagrams and extended descriptions.

   - Automatically-generated prose. The automated module information is generated from the module prologue and the code itself. It consists of three parts: module description, module operation summary, and diagnostic aids.

5. Module descriptions that describe the operation of the modules (the module descriptions are contained in System Logic Library: Module Descriptions)

Some component books also include diagnostic techniques information following the Introduction.

FURTHER INFORMATION

For more information about the System Logic Library, including the order numbers of the books in the System Logic Library, see the Master Preface in MVS/Extended Architecture System Logic Library: Master Index.

CONTENTS

FIGURES

Summary of Amendments for LY28-1740-1 MVS/System Product Version 2 Release 2.3

This major revision contains changes to support MVS/System Product Version 2 Release 2.3. Changes include MVS/XA support for MVS/Data Facility Product Version 3 Release 1.0, which introduces the storage management subsystem (SMS). SMS provides new function for data and storage management.

- Modifications to the Introduction section, including new and changed information on:
  - SJF Initialization
  - SJF Control Blocks
  - SJF Services
  - SJF Parameter List Conventions
  - SJF SWA Token
  - Invoking SJF Routines
  - Requirements For Using SJF Services
  - Recovery Processing
  - SJF Return Code Conventions
  - SJF Reason Code Conventions
  - SJF Abend Code Conventions
- New information in the Process Flow section.
- Method of Operation diagrams for the following new modules:

  IEFSJACC IEFSJVER
  IEFSJERS
- Changes to the following modules:

  IEFSJBLD IEFSJFND
  IEFSJCNL IEFSJHTB
  IEFSJDEF IEFSJINT
  IEFSJDEL IEFSJUPD
  IEFSJEXT IEFSJWRT

The Preface has been updated to include the new title for the <u>MVS/XA System Logic Library: Master Index</u> and the deletion of the index from the <u>MVS/XA System Logic Library: Module Descriptions</u>.

Summary of Amendments for LY28-1740-0 for MVS/System Product Version 2 Release 2.0

This publication is new for MVS System Product Version 2 Release 2.0. It contains information that was reorganized from the Scheduler JCL Facility section in <u>MVS/XA System Logic Library Volume 12</u>, LY28-1250, which applies to MVS/System Product Version 2 Release 1.7.

This publication contains changes to support MVS/System Product Version 2 Release 2.0. The changes include:

- Information for the new module IEFSJHTB.
- A new topic, SJF Initialization, in the Introduction.
- Minor technical and editorial changes throughout the publication.

INTRODUCTION

The scheduler JCL facility (SJF) is a set of routines used to interface with the converter, interpreter, dynamic allocation, storage management subsystem, the job entry subsystems (JES2 and JES3), printer support facility (PSF), and TSO. SJF routines interface with them in:

- Referencing information in the JCL definition tables (JDTs) and the statement definition table (SDT).
- Storing information in scheduler work blocks (SWBs).
- Retrieving information from the SWBs.
- Retrieving and updating information in other scheduler work area (SWA) blocks.

The converter uses the SJF services to verify JCL verbs and keywords. SJF obtains this information about the verbs and keywords from the JDTs. The interpreter uses the SJF services to verify JCL verbs and keywords and to create and update the SWBs that will contain the keyword subparameter information. Dynamic allocation invokes the SJF services to verify the dynamic allocation text units and to store the text unit parameter information in the SWBs. JES uses SJF services to obtain output characteristics from the SWBs, and recreates the output SWBs in the PSF address space using SJF GET and PUT services. PSF uses the SJF services to retrieve output information from the SWBs to determine how the output is to be printed. The storage management subsystem invokes SJF services to retrieve and update information residing in SWA control blocks to determine if a dataset is eligible to reside on system-managed storage. TSO uses the SJF services to validate commands and operands defined in the JDTs, and obtains text units built by SJF in order to dynamically create a DD statement.

SJF INITIALIZATION

SJF initialization occurs during master scheduler base initialization (IEEVIPL) processing. Module IEFSJLOD receives control from IEEVIPL, and initializes SJF entry points JESSJF and JSSJCNL in the JESCT extension. Immediately following this, IEFSJLOD invokes IEFSJINT which calls IEFSJDEF (SJF define JDVT routine) indicating that a default JDVT be built. IEFSJINT uses IEFJSIMW (a message writing routine) to issue error messages. Upon return from IEFSJINT, IEFSJLOD initializes the mutual exclusivity checker entry point JESMECHK in the JESCT extension. See MVS/XA SPL: System Initialization Logic for more detailed information.

SJF CONTROL BLOCKS

The following text describes control blocks created and maintained by the scheduler JCL facility. Access to these are supported only via the appropriate SJF service.

JCL DEFINITION TABLE (JDT)

The JCL definition tables (JDTs) define statement types, keywords for each statement type, and keyword subparameters. Each JDT contains:

- Statement type (verb) and the keywords for each statement type.
- The owner name (JDT name).
- For each keyword, the corresponding TSO commands and operands supported through SJF, if any exist.
- A text unit key for each keyword subparameter.
- Rules for the keyword subparameters such as data type and length.
- An identifier for each keyword subparameter. This identifier will be specified on a request for SJF to update the keyword subparameter.
- SWB block ID of the SWB in which the keyword subparameter should be stored.

STATEMENT DEFINITION TABLE (SDT)

A statement definition table (SDT) contains information about JCL statement types, keywords and keyword subparameters that reside in control blocks other than SWBs, such as the Job Control Table (JCT), Step Control Table (SCT), Step I/O Table (SIOT), etc. There is currently one SDT, and it is pointed from the JCL definition vector table (JDVT).

HASH TABLES

For performance reasons, SJF uses a hashing algorithm when searching the JDTs for a JCL verb, keyword, keyword subparameter, command or operand. The hash tables are built during SJF initialization and are pointed from the JDVT.

JCL DEFINITION VECTOR TABLE (JDVT)

A JCL definition vector table (JDVT) logically groups one or more JDTs. It contains a JDVT name and the names and addresses of the associated JDTs. It also contains the addresses of the SDT, verb hash table, and command hash table. The JDVT is anchored off the job entry subsystem control table (JESCT) extension. There is currently one JDT for the system and its name is CNTLJDVT.

Figure 1 on page SJF-5 shows the JDVT structure.

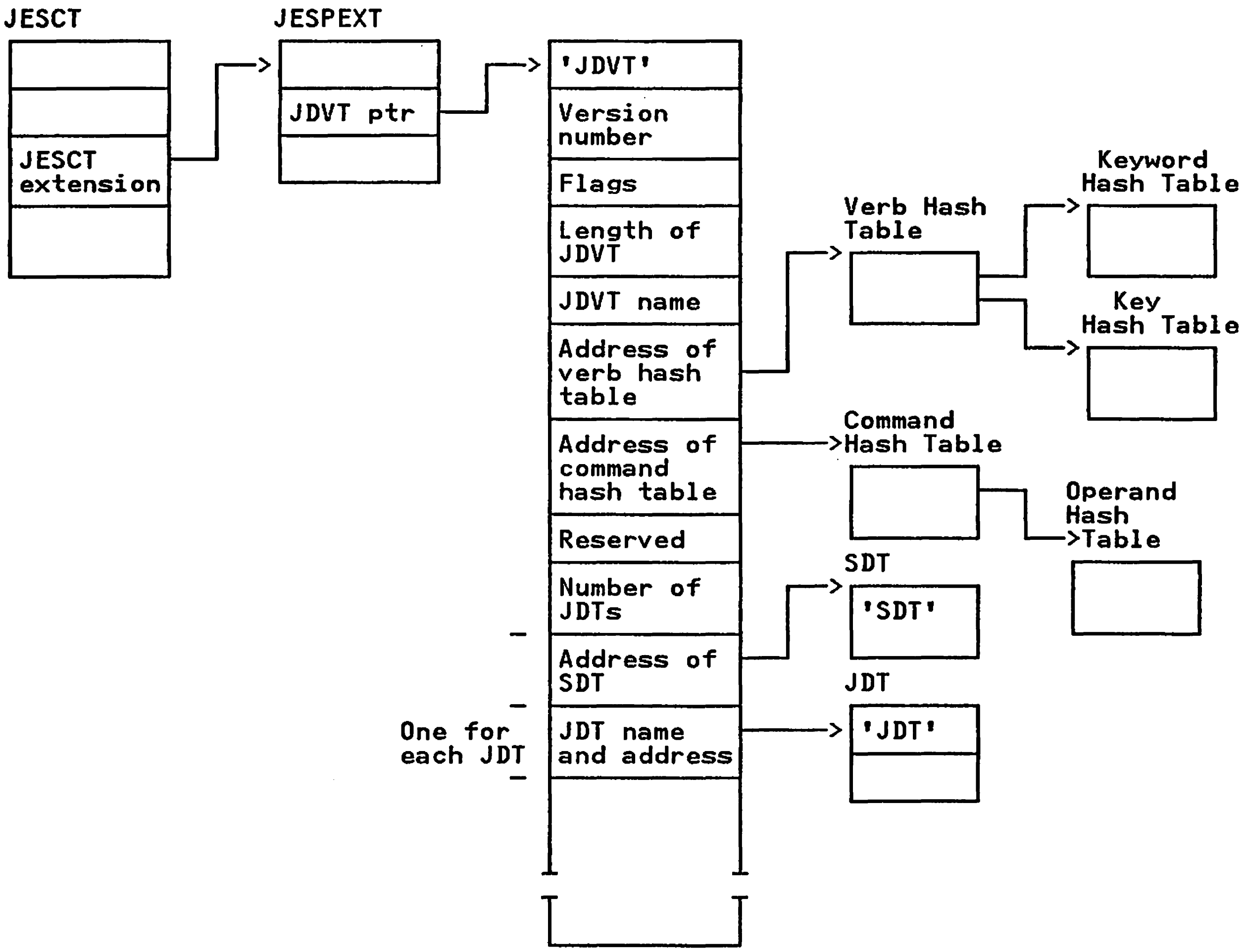


Figure 1. JCL Definition Vector Table (JDVT) Structure

SCHEDULER WORK BLOCKS (SWBS)

The scheduler work block (SWB) is a SWA control block that is used to save JCL keyword subparameter information. A SWB is identified by an owner name (JDT name), control block ID, statement type (verb), and a statement label (the 1-8 character name following the // on the JCL statement). Each SWB chain represents a JCL statement. The SWB will contain all information derived from the new JCL for a particular verb and label.

SWBs are created at the following three levels:

- Job level: For statements within a job prior to the first EXEC statement, the SWBs are located off the job control table extension (JCTX).
- Step level: For statements specified within a step, the SWBs are located off the step control table (SCT).
- DD level: For keywords on a DD statement, the SWBs are located off the step input output table (SIOT).

Figure 2 is an example of a SWB structure created from the following JCL.

```
//JOBX    JOB
//NV1     NEWVERBA  KEYWORD1=X,KEYWORD2=(1,2,3)
//NV2     NEWVERBA  KEYWORD1=Z
//STEP1   EXEC
//NV3     NEWVERBB  KEYWORD3=(X,Y)
//DD1     DD        KEYWORD4=A
//
```

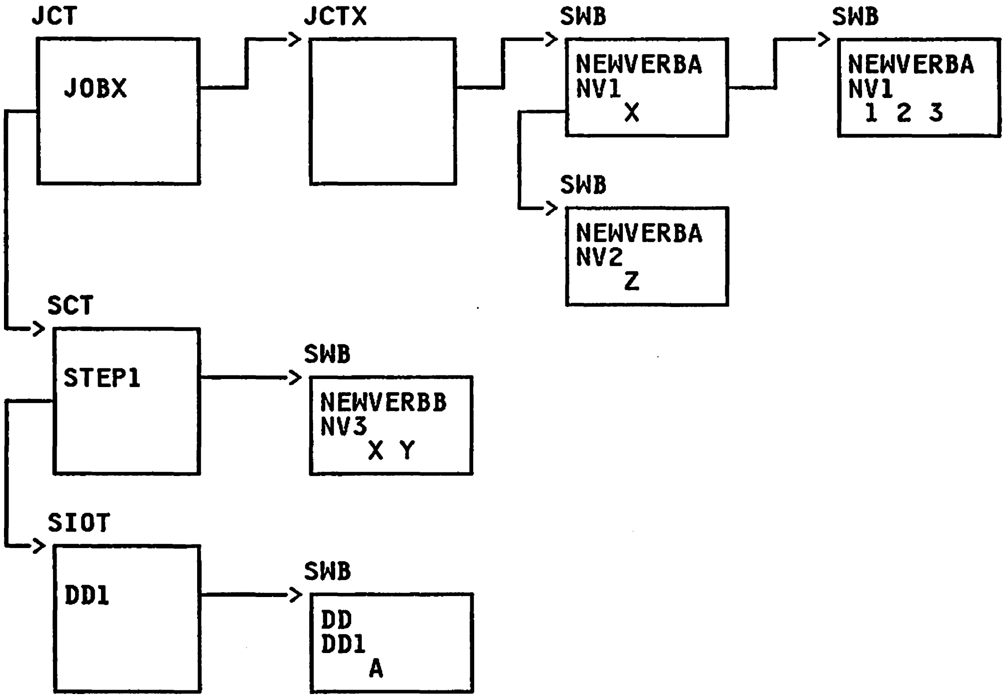


Figure 2. Scheduler Work Block (SWB) Structure

ADDRESSING AND RESIDENCY MODE OF SJF MODULES

Most of the SJF modules have an addressing and residency mode of 31, but SJF can handle callers in either 24 or 31-bit addressing mode. Input data can reside at 24 or 31 bit addresses.

- IEFSJRTE has a residency mode of 24 and provides an interface between 24-bit callers and IEFSJCNL.

SJF SERVICES

The scheduler JCL facility (SJF) provides the following services.

Delete: Deletes a SWB chain.

Erase: Erases a keyword (or key) subparameter in a specified SWB chain.

Extract: Extracts information from the JDT associated with a verb, a verb and keyword, a verb and key, or subparameters of a keyword and key.

Find JDVT: Locates the specified JDVT.

Find SWA: Locates a SWB chain or a SWA block at a particular level of the SWA structure.

Get: Copies selected keywords from a SWB chain in text unit format into a storage area specified by the caller. The keywords obtained are those whose JDT flags match the qualifier flags set in the input parameter list.

Initialize JDVT: Creates the system default JDVT.

Access: Locates a particular level of the SWA control block structure, and retrieves/updates information in the SWA control blocks.

Put: Rebuilds a SWB chain from SWB keyword data found in text unit format.

Retrieve: Retrieves parameter information from a scheduler work block (SWB) chain associated with a keyword or keyword for a particular verb and label, and uses that information to build text units.

Terminate: Frees all SJF working storage and deletes the recovery environment, if necessary. This SJF service is invoked as the SJF function.

Update: Verifies the text units specified by the caller and if requested, updates the SWB chain with the information specified in text unit format.

Verify: Provides an unauthorized interface to build text units for SJF defined keywords for use by callers such as TSO.

Write: Locates a specific SWB and updates the data portion of the SWB.

NAMING CONVENTIONS OF SJF MODULES

Each SJF module has the following format:

IEFSJ___

The letters IEF indicate that the routine is part of the scheduler. The letters SJ indicate that the routine is further classified as a scheduler JCL facility module. The last three characters are any meaningful string that further describe the module (for example - IEFSJCNL, where CNL is a shortened form of control).

SJF PARAMETER LIST CONVENTIONS

The first 16 bytes of all SJF input parameter lists have the following format:

LENGTH	MASK	DESCRIPTION
4		Parameter list identifier
1		Version number
1		Control flags:
	X'80'	No recovery processing
	X'40'	No cleanup processing
	X'20'	Unauthorized caller
2		Parameter list length
4		Local storage pointer (returned)
4		Reason code (returned; include the SJF Reason code mapping, IEFSJRC, to analyze)

SJF SWA TOKEN

Many SJF routines require a SWA block "token" as part of their input. The token is a way to identify to SJF which level of the SWA control block structure to perform the requested service on (eg., JOB, STEP, DD). Use the SJF Find routine to obtain the token.

INVOKING SJF ROUTINES

The SJF Request Macro (SJFREQ) is used to invoke SJF. For an Access request, the SJF Access macro (SJFACC) is used. The local storage pointer in the parameter list must be zero on the first invocation.

Callers invoking SJF services multiple times may find it advantageous to utilize the multiple invocation feature of SJF. This allows SJF to retain its resources and recovery environment over multiple calls, thereby eliminating the overhead for each invocation. This can be accomplished by doing the following:

1. On the first invocation specify "NO CLEANUP" in the parameter list. If an ESTAE environment is not desired, then indicate "NO RECOVERY" as well. (The default is for SJF to establish an ESTAE environment.) The address of SJF's storage area will be returned in the local storage pointer field in the input parameter list (see SJF Parameter List Conventions).

2. On subsequent invocations, supply the local storage pointer returned in the parameter list, as well as specifying "NO CLEANUP" and "NO RECOVERY" (if a recovery environment was established) in the input parameter list.

3. On the last invocation, specify a request type of "TERMINATE", or do NOT specify "NO CLEANUP". This will free any resources held by SJF, including the SJF local storage, and delete the ESTAE environment, if one was previously established.

REQUIREMENTS FOR USING SJF SERVICES

The following is required to invoke SJF services:

- With the exception of SJF Verify and Terminate requests, the caller must be in supervisor state and run in key 0-7.

- If the caller's storage area is referenced by SJF, then it must not be fetch protected.

- SJF services are NOT available in cross-memory mode.

- Use of the multiple invocation facility of SJF is limited to one task. This is due to the recovery processing of SJF, as well as obtaining access to the SJF local storage over multiple invocations.
- If the caller is using SJF to access DD level control blocks, such as SIOTs and DD-level SWBs, then an enqueue for the SYSZTIOT resource MUST be issued. See MVS/XA SLL: Vol 2. for more information.

RECOVERY PROCESSING

The SJF Control Routine (IEFSJCNL) establishes a recovery environment for the SJF functions, if requested by the caller. The caller can request that a recovery environment not be established by specifying "NO RECOVERY" in the input parameter list. The default is for SJF to set up a recovery environment. It is desirable for SJF to establish a recovery environment to ensure that the resources held by SJF, including the local storage, are freed when an abend occurs while an SJF environment is outstanding. Once a recovery environment is established, it will remain active until the SJF environment is terminated. This can be accomplished by not specifying "NO CLEANUP" on the last SJF request, or issuing a "TERMINATE" request via the SJFREQ or SJFACC macros.

A return code from SJF of decimal 20 (X'14') indicates that a system error has occurred, and recovery processing has been performed by SJF. SJF will have freed up its resources as a result. Thus, the caller should NOT attempt to terminate the SJF environment after an abend takes place. Also, because the SJF local storage is freed during recovery processing, the caller should NOT supply the local storage pointer on subsequent SJF invocations. SJF will zero out the local storage pointer in the parameter list used by the caller on the first SJF invocation. Thus, the caller should ensure that SJF can address this storage area at any time during SJF processing.

See the module description for IEFSJCNL for more detailed information on SJF recovery processing.

SJF RETURN CODE CONVENTIONS

The following decimal codes will be returned in register 15 for the indicated situations:

0 SJF request successfully completed: in some instances, a non-zero reason code will be set in the input parameter list.

4 SJF request not processed: a non-zero reason code indicative of the error is set in the input parameter list.

8 Error in the input parameter list passed to SJF: Check the first 16 bytes of the input parameter list.

16 SJF ESTAE could not be established.

20 A system error occurred while an SJF environment was active.

24 Error in SJF initialization.

If the return code specified in register 15 is greater than decimal 16 (X'10'), then the caller should NOT attempt to terminate the SJF environment, or pass the local storage pointer obtained on previous calls, as this will cause SJF to attempt to access resources that no longer exist.

SJF REASON CODE CONVENTIONS

The first 2 decimal digits of the SJF reason code corresponds to the function code of the SJF function requested.

SJF ABEND CODE CONVENTIONS

SJF issues abends when certain unexpected conditions are encountered. See the documentation in MVS/XA Message Library: System Codes for abend 'X'054'.

PROCESS FLOW

Figure 3 on page SJF-12 shows the process flow for the following SJF modules.

```
IEFSJACC
IEFSJBLD
IEFSJCNL
IEFSJDEF
IEFSJDEL
IEFSJERS
IEFSJEXT
IEFSJFND
IEFSJGET
IEFSJHTB
IEFSJINT
IEFSJJDV
IEFSJPUT
IEFSJRET
IEFSJRTE
IEFSJUPD
IEFSJVER
IEFSJWRT
```

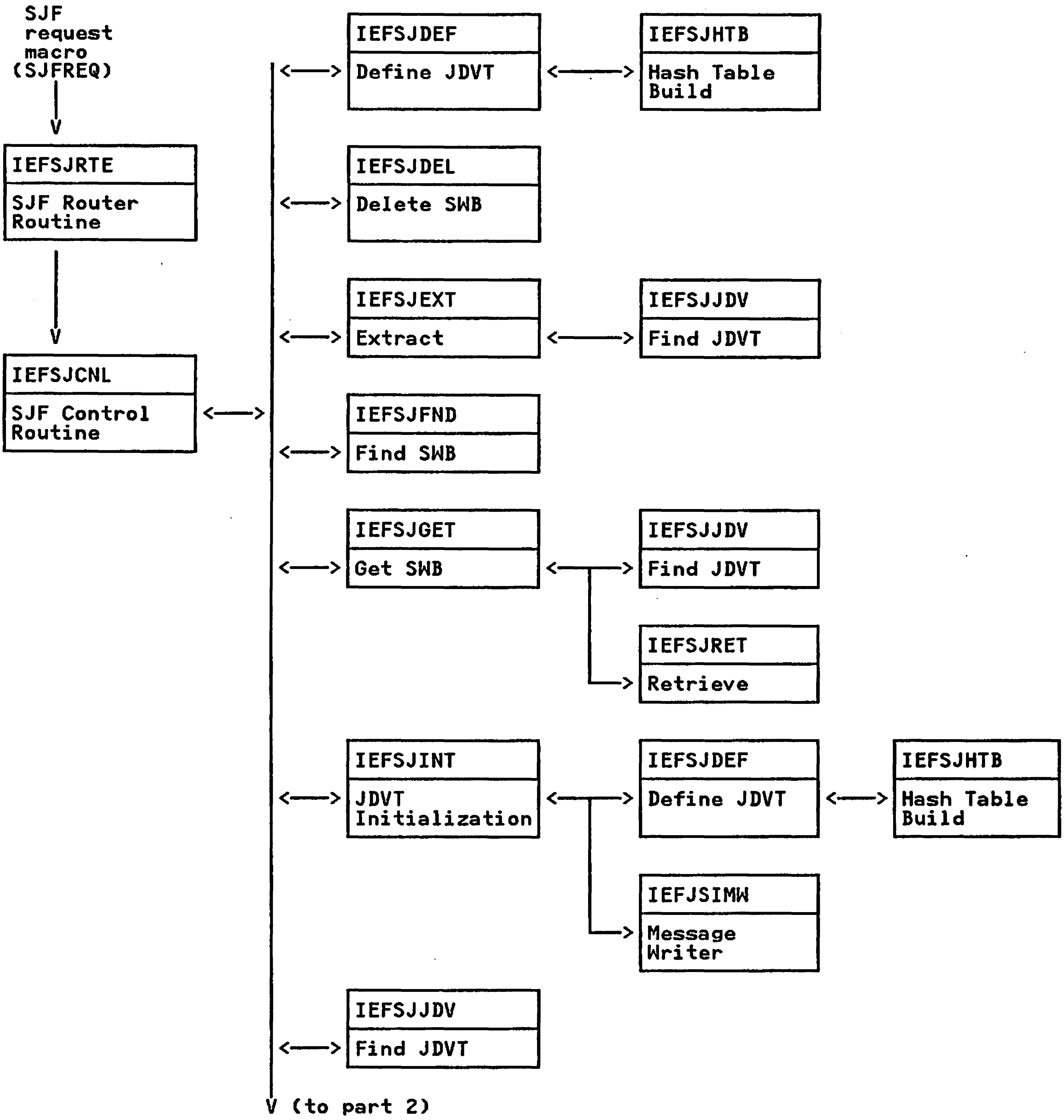


Figure 3 (Part 1 of 3). Scheduler JCL Facility Process Flow Overview

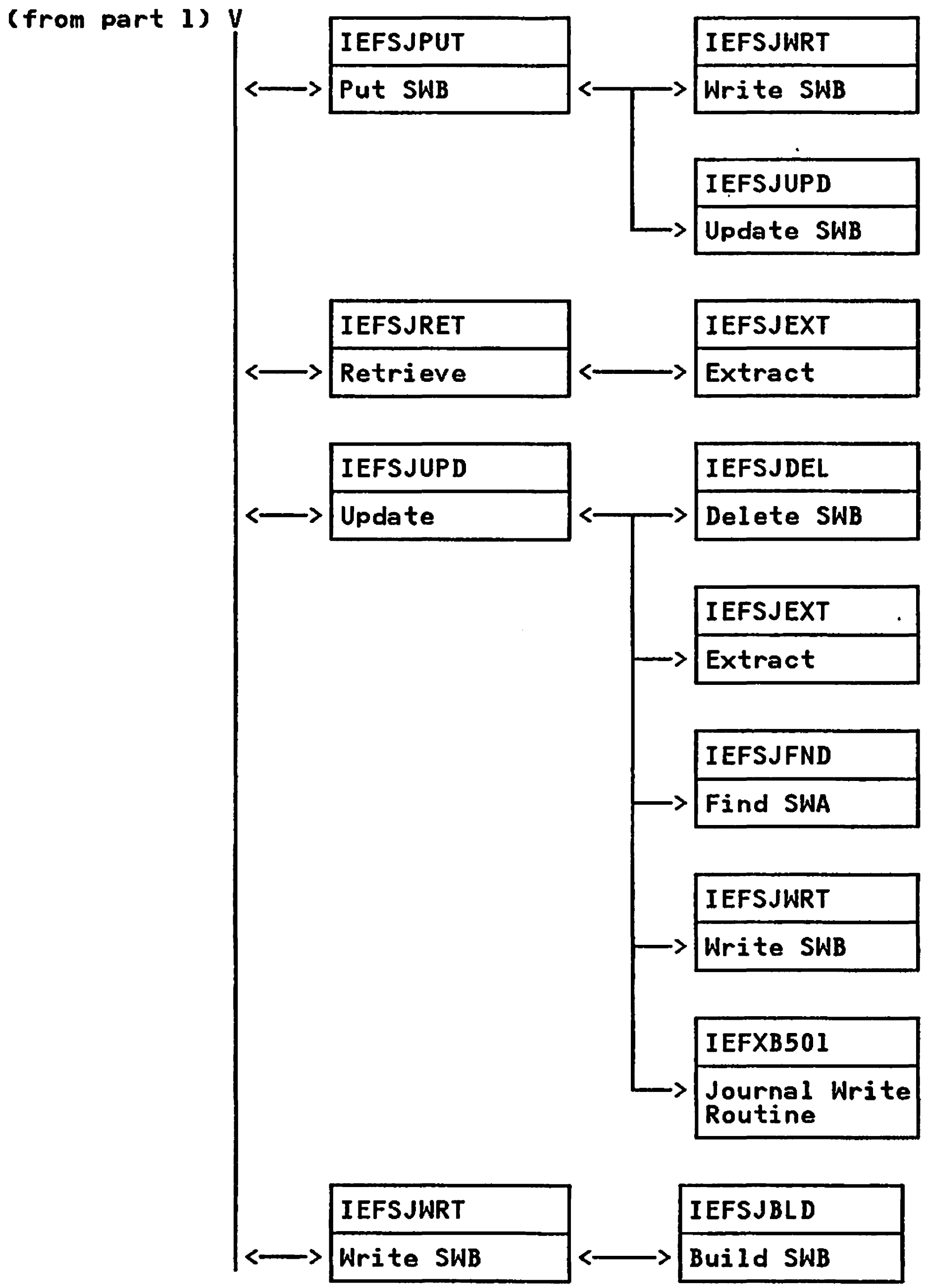


Figure 3 (Part 2 of 3). Scheduler JCL Facility Process Flow Overview

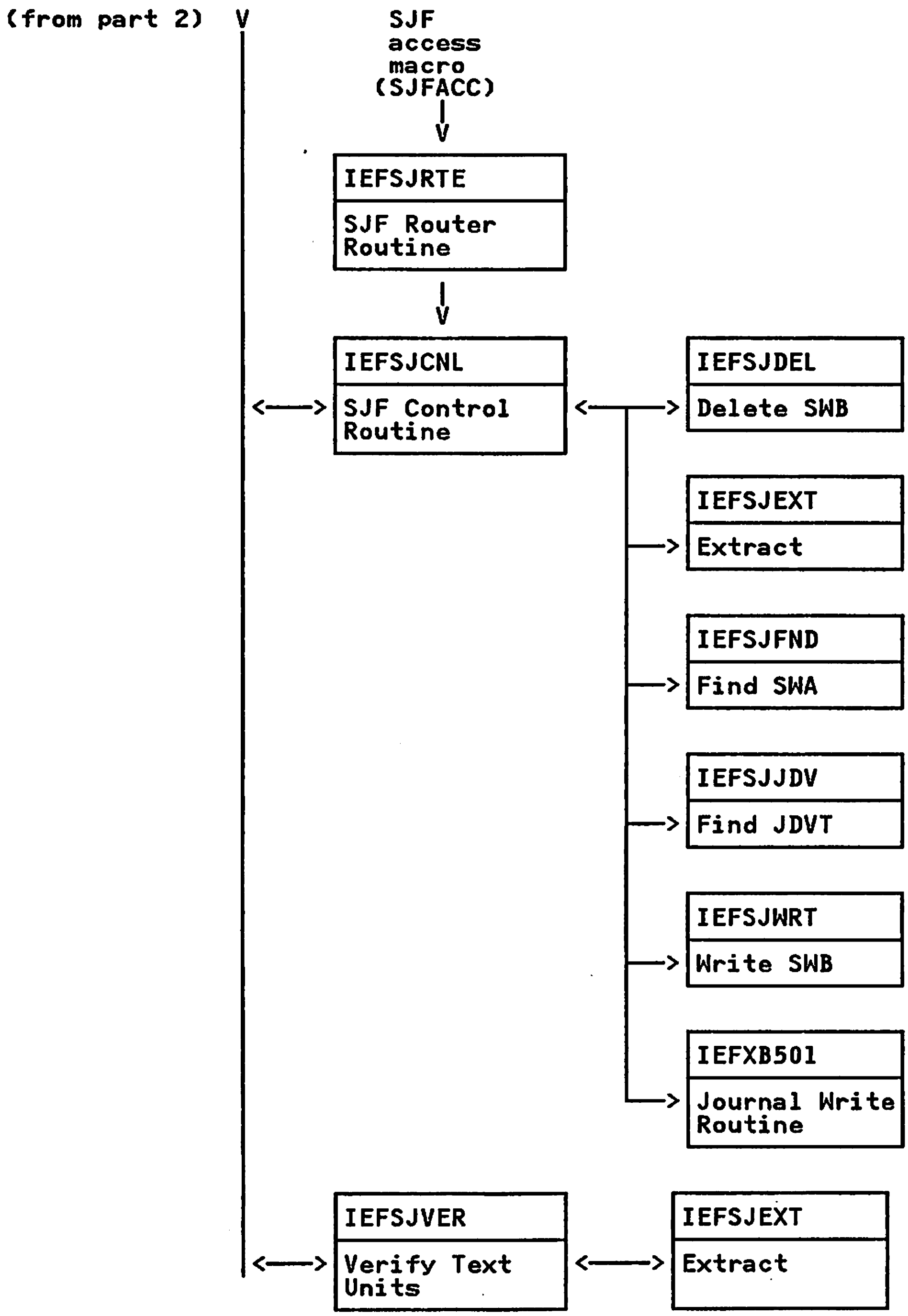


Figure 3 (Part 3 of 3). Scheduler JCL Facility Process Flow Overview

METHOD OF OPERATION

This section has detailed information for every SJF module. These modules are in alphabetic order. This detailed information is broken down into four different headings. The headings and the topics they document are:

Module Description, which includes:

- Descriptive name
- Function (of the entire module)
- Entry point names
- External references
- Tables
- Serialization

Note: Brief SJF module descriptions appear in MVS/Extended Architecture System Logic Library: Module Descriptions, which contains module descriptions for all the MVS/Extended Architecture components described in the System Logic Library.

Module Operation, which includes:

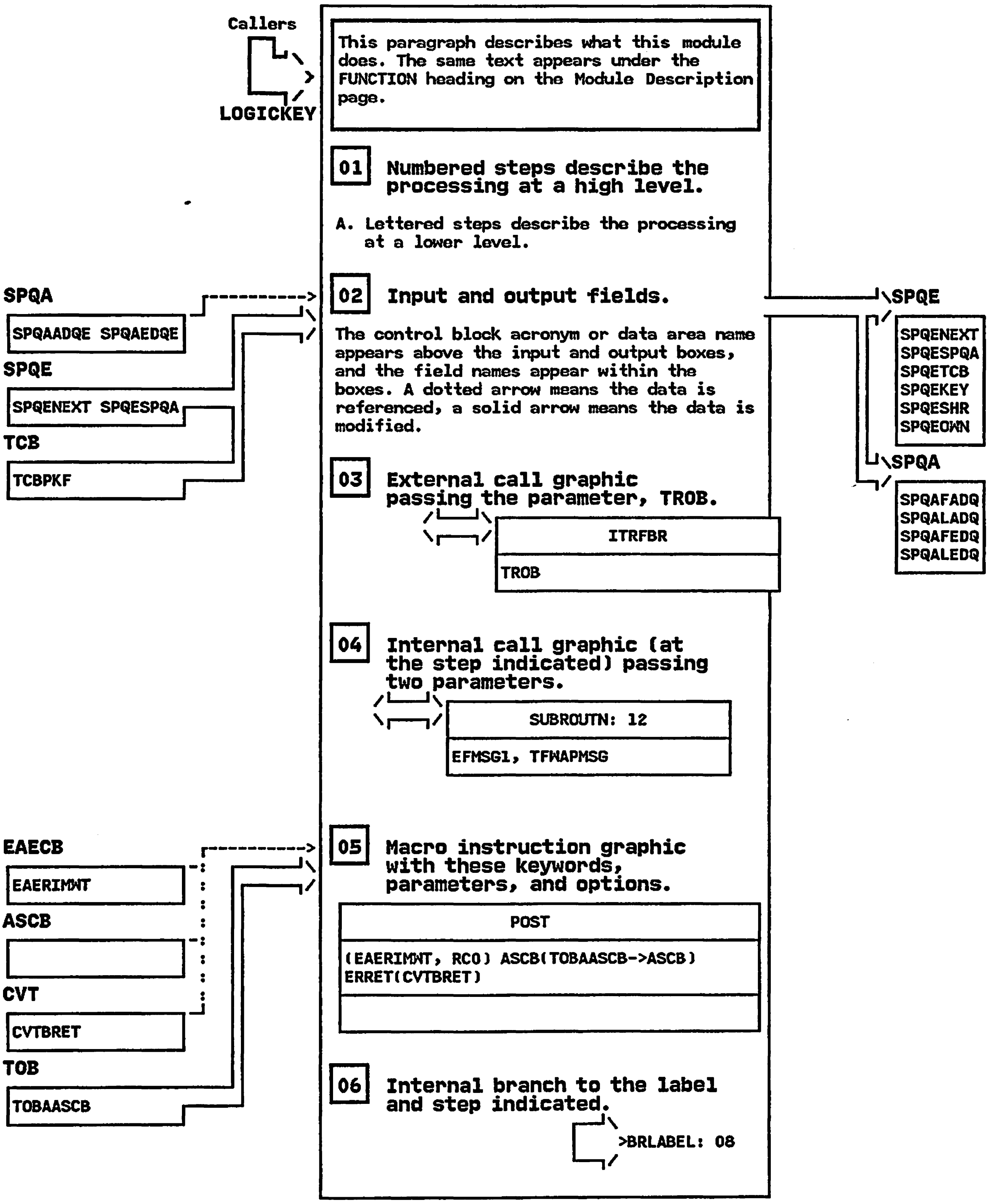
- Operation, which explains how the module performs its function.
- Recovery operation, which explains how the module performs any recovery.

Diagnostic aids, which provide information useful for debugging program problems; this includes:

- Entry point names
- Messages
- Abend codes
- Wait state codes
- Return codes for each entry point. Within each entry point, return codes might be further categorized by exit-normal and exit-error.
- Entry register contents for each entry point
- Exit register contents for each entry point

Logic Diagram, which illustrates the processing of the module, the input it uses, the output it produces, and the flow of control. Many modules do not have a logic diagram because the processing is sufficiently explained in the module description, the module operation, and the diagnostic aids sections. Figure 4 on page SJF-16 illustrates the graphic symbols and format used in the logic diagrams.

LOGICKEY - Key to the Logic Diagrams STEP 01



This paragraph describes what this module does. The same text appears under the FUNCTION heading on the Module Description page.

01 Numbered steps describe the processing at a high level.

A. Lettered steps describe the processing at a lower level.

02 Input and output fields.

The control block acronym or data area name appears above the input and output boxes, and the field names appear within the boxes. A dotted arrow means the data is referenced, a solid arrow means the data is modified.

03 External call graphic passing the parameter, TROB.

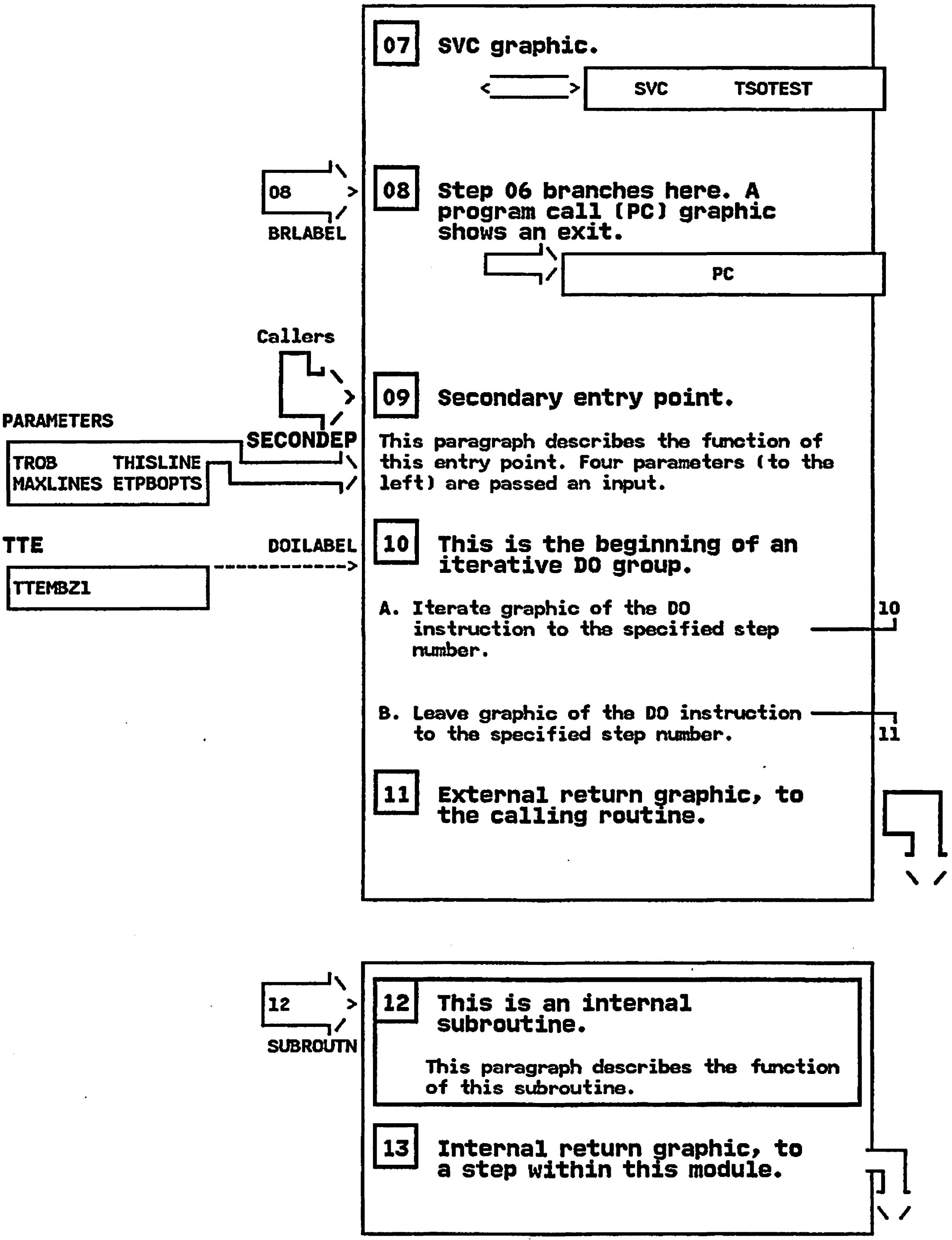
04 Internal call graphic (at the step indicated) passing two parameters.

05 Macro instruction graphic with these keywords, parameters, and options.

(EAERIMWT, RC0) ASCB(TOBAASCB->ASCB)
ERRET(CVTBRET)

06 Internal branch to the label and step indicated.

Figure 4. Key to the Logic Diagrams (Part 1 of 2)



07 SVC graphic.

08 Step 06 branches here. A program call (PC) graphic shows an exit.

09 Secondary entry point.

This paragraph describes the function of this entry point. Four parameters (to the left) are passed an input.

10 This is the beginning of an iterative DO group.

A. Iterate graphic of the DO instruction to the specified step number.

B. Leave graphic of the DO instruction to the specified step number.

11 External return graphic, to the calling routine.

12 This is an internal subroutine.

This paragraph describes the function of this subroutine.

13 Internal return graphic, to a step within this module.

Figure 4. Key to the Logic Diagrams (Part 2 of 2)

IEFSJACC - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Access SWA Routine

FUNCTION:
This module allows callers to locate a particular level of the SWA control block structure, and retrieve or update information contained in these blocks.

ENTRY POINT: IEFSJACC

PURPOSE: See Function

LINKAGE: CALL

CALLERS: SJF control routine (IEFSJCNL)

INPUT:
SJF Access Parameter List (IEFSJACP)

FIELD	LENGTH/MASK	DESCRIPTION
SJAC	96	
SJACID	4	identifier 'SJAC'
SJACVERS	1	version number
SJACFLAG	1	function flags
SJACNREC	x'80'	no recovery,
SJACNOCU	x'40'	no cleanup,
SJACLEN	2	length of parm list
SJACSTOR	4	local storage pointer,
SJACREAS	4	reason code (returned)
SJACTOKN	8	SJF token
SJACFLDS		
SJACRQST	1	request type
SJACUPD	X'80'	update
SJACRET	X'40'	retrieve
SJACFIND	X'20'	find
SJACFUNC	1	flag field
SJACSYST	X'80'	system input
SJACUNAU	X'40'	request is from an invoker whose caller is unauthorized
SJACCNT	X'20'	continue processing after acceptable errors have occurred
SJACJRNL	X'10'	journaling requested
SJACREQ#	2	number of individual requests
SJACRPTR	4	pointer to request table, address of variable length storage acquired for positional parameters
SJACCHID	16	SWB chain identification
SJACVERB	8	Verb (optional if not DD)
SJACLABL	8	statement label (optional)
SJACFNP		SJF Find parameters
SJACFLG2	1	
SJACNEXT	X'80'	find next SWB processing,
SJACNJST	X'40'	JOB token supplied
SJACJBTK	X'20'	JOB token requested
SJACCSTK	X'10'	Current Step token requested@D2A
SJACFUN1	1	non-master scheduler flag byte
SJACJOB	X'80'	job level
SJACCST	X'40'	current step level
SJACST	X'20'	step level and procname and step
SJACRSV0	2	reserved
SJACSTPN	8	step name
SJACPRLB	8	Label on the proc statement

IEFSJACC - MODULE DESCRIPTION (Continued)

SJACSTMT	4	statement number(returned)
SJACALT	4	Address of alternate SWA Manager
SJACRSV2	4	Reserved
SJRQT		request table
SJACENTY	16	request table entry
SJACRSN	4	reason code (returned)
SJACADDR	4	address of area
SJACLNTH	2	length of area
SJACKEY	2	key
SJACPARM	1	parameter number
SJACRSV3	3	reserved

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:

SJF Access Parameter List (IEFSJACP)

FIELD	LENGTH	DESCRIPTION
SJACREAS	4	Reason code of first error
SJACRSN	4	Reason code for each entry In the request table
SJACVERB	8	Verb
SJACLABL	8	Label
SJACSTMT	4	SJF statement number
SJACTOKN	8	SJF token
SJACSTOR	4	SJF local storage pointer

The following control blocks may be updated:

IEFASIOT - Step I/O Table
IEFJFCBN - Job File Control Block
IEFJFCBX - Job File Control Block Extension
IEFSWB - Scheduler Work Block

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: ACCRETRY

PURPOSE:
Performs cleanup processing when an abend occurs during the SJF Access routine's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES:
IEFSJCAS - SJF Check ASIS Type Routine (included)
IEFSJCCH - SJF Check Character Type Routine (included)

IEFSJACC - MODULE DESCRIPTION (Continued)

IEFSJCBM - SJF Check Bytemask Type Routine (included)
IEFSJCHX - SJF Check Hexadecimal Type Routine (included)
IEFSJCIN - SJF Check Integer Type Routine (included)
IEFSJCBL - SJF Check Boolean Type Routine (included)
IEFSJTOK - SJF Token Build Routine (included)
IEFSJDEL - SJF Delete
IEFSJEXT - SJF Extract
IEFSJFND - SJF Find
IEFSJJDV - SJF Find JDVT
IEFSJWRT - SJF Write
IEFXB501 - Journal Write Routine

DATA AREAS:
IEFQMIDS - SWA Block ID and Acronym Constants
IEFSJACP - SJF Access Parameter List
IEFSJCNW - SJF Control Work Area
IEFSJDLP - SJF Delete Parameter List
IEFSJEXP - SJF Extract Parameter List
IEFSJFNP - SJF Find parameter list
IEFSJJDP - SJF Find JDVT parameter list
IEFSJKEY - SJF Key Mapping
IEFSJRC - SJF Reason Codes
IEFSJWRP - SJF Write Parameter List
IEFZB502 - SWA Prefix Mapping
IEFZB505 - EPA Mapping for Locate Mode SWA Manager
IEFZB507 - Journal Write Parameter List
IEZJSCB - Job/Step Control Block
IHAPSA - Prefix Save Area
IKJTCB - Task Control Block

CONTROL BLOCKS: None

TABLES: IEFSJSDT

IEFSJACC - MODULE OPERATION

This module allows callers to locate a particular level of the SWA control block structure, and retrieve or update information contained in these blocks. It does the following:

1. If the caller has requested to find a particular level of the SWA control block structure:

   A. If the request is for the JOB or Current Step token, calls IEFSJTOK to build the token. The Active JSCB is used as a starting point in order to obtain the token requested.

   B. If the FIND request is NOT for the JOB or Current Step token, initializes the parameter list to SJF Find SWA block and invokes this routine.

   C. If SJF Find processing is successful or if the request is for a DD level search and SJF Find returns a return code of 4 and a reason code indicating that no SWB chain has been found, then processing continues. Otherwise, the return and reason code from SJF Find are returned to the caller.

2. Evaluates the SJF token passed by the caller or, if SJF Find has been invoked, the SJF token returned from SJF Find. If the token is invalid, a return code 4 and a reason code indicating that the token is invalid are returned to the caller.

3. Initializes pointers to the SWA control blocks which represent the statement being processed.

4. Obtains a minimum of 4K of storage. If more than 4K of storage is needed to process the request, then the larger amount is obtained instead. For retrieve requests, storage is obtained for a table to contain information about each request in the request table. For update requests, storage is obtained and the SWA blocks to be updated are copied into the storage.

   - If storage was obtained on a previous call to SJF Access, then determines if it is large enough to accomodate the current storage needed. If so, then reuses the existing storage area. Otherwise, frees the existing storage area, obtains a larger storage area, and anchors it out of the SJF Control Workarea (SJCNW).

   - If storage was not previously obtained, then getmains the storage area and anchors it out of the SJF Control Workarea (SJCNW).

   If the needed storage is unavailable, a return code of 4 and a reason code indicating that storage is unavailable are returned to the caller.

5. Determines if the JCL Definition Vector Table (JDVT) pointed to by the control workarea is the system default JDVT. If not, then invokes the SJF Find JDVT routine to locate the system default JDVT. If the SJF Find JDVT routine processing was not successful, then the return and reason code from SJF Find JDVT are returned to the caller.

6. Determines if a Statement Definition Table (SDT) exists and if so, whether it contains the verb specified by the

IEFSJACC - MODULE OPERATION (Continued)

caller.

7. For each entry in the request table, does the following:

   A. Moves (via MVCK) the key, parameter number, address of caller's area and length of caller's area into local storage.

   B. If the verb was found in the SDT then searches the SDT for the key and parameter number. If the key and parameter are found in the SDT then obtains addressability to the control block which contains the data for this request. If the key is found, but the parameter is not found in the SDT, sets a temporary reason code and an indicator that an error has occurred.

   C. If neither the key or parameter were found in the SDT, invokes the SJF Extract routine to locate the key and parameter in the JCL Definition Tables (JDTs). If SJF Extract processing was unsuccessful and the reason code from SJF Extract indicates that either the key or parameter were not found, sets a temporary reason code and an indicator that an error has occurred. If SJF Extract processing was unsuccessful for a reason other than the key or parameter was not found, the return and reason code from SJF Extract are returned to the caller.

   D. Ensures that the key may be accessed by the caller. If the key is defined in the JDT as a system use only key and the caller has not specified that this is a system use invocation, sets a temporary reason code and an indicator that an error has occurred. If the key is defined in the SDT as an authorized use only key and the caller has indicated that this invocation is on behalf of an unauthorized caller, sets a temporary reason code and an indicator that an error has occurred.

   E. Ensures that the caller has specified a non-zero address and a valid length for the field into which data will be retrieved or from which the SWA blocks will be updated. If the address is zero or the length is invalid, a return code of 4 and a reason code describing the error are returned to the caller.

   F. For retrieve requests:

      1. Ensures that the data is valid. If the key is JDT defined, checks the validity bit in the SWB. If the key is SDT defined, performs validity checks as defined in the SDT. If the data is not valid, sets a temporary reason code and an indicator that an error occurred.

      2. Stores information about the request in the local retrieve table.

   G. For update requests:

      1. If the key is SDT defined, ensures that the key may be updated. If the key can not be updated, sets a temporary reason code and an indicator that an error occurred.

      2. Moves (via MVCK) the data to be updated into local storage.

IEFSJACC - MODULE OPERATION (Continued)

3. Performs validity checking on the data to be updated. If the data is invalid, a return code 4 and a reason code indicative of the error are returned to the caller.

4. Updates the SWA block copies with the data to be updated. If the key is JDT defined, invokes SJF Write SWB to update an existing chain or create a new SWB chain. If SJF Write is unsuccessful, the return and reason code from SJF Write are returned to the caller. If the key is SDT defined, updates the SWA block and sets validity indicators as defined in the SDT.

H. Determines if processing should continue with the next request in the request table. If no errors have occurred or if an "allowable" error has occurred and the caller has specified to continue after certain "allowable" errors have occurred, then processing will continue with the next request in the request table. If the caller has not specified to continue processing after "allowable" errors occurred, a return code of 4 and the reason code previously set are returned to the caller. See RETURN CODES below for a list of reason codes which describe "allowable" error conditions.

8. After all requests in the request table have been processed, performs the operation requested. For retrieve requests, copies the SWA block information for each key into the fields specified by the caller. For update requests, copies the SWA block copies into the original SWA blocks or anchors a new SWB chain and if journalling was requested, invokes IEFXB501, the journal routine.

9. If the caller did NOT specify "NOCLEANUP" (SJACNOCU), then frees the storage area that has been obtained.

10. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the SJF control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (ACCRETRY) in the SJF control workarea. When ACCRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate a SJF system error.

2. If any new SWBs were created, invokes SJF Delete to delete the new SWBs.

3. Frees storage that has been obtained.

4. Returns to the caller.

IEFSJACC - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJACC
ACCRETRY

MESSAGES: None

ABEND CODES:

'054'X (decimal 84) and the following
reason codes in decimal

10 - Invalid control block acronym encountered in SDT
11 - Invalid validity type in SDT
12 - Invalid data type for this parameter in SDT
13 - Invalid special key defined in SDT
15 - Invalid data type for this parameter - no parameter checking routine exists

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJACC:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully

Reason codes in SJACREAS

SJRCNOER (0)	- Request completed successfully
SJRCALLW (1304)	- Allowable errors occurred - see SJACRSN to evaluate the error

Reason codes in SJACRSN for SJRCALLW

SJRCNKEY (202)	- Key not defined
SJRCNPRM (203)	- Parameter not defined
SJRCIVKY (504)	- Invalid key, system specification only
SJRCNDAT (1300)	- No data exists for this parameter
SJRCNATH (1301)	- Not authorized to access this data
SJRCNUPD (1303)	- Key not updateable
SJRCNOCB (1306)	- Control block does not exist

EXIT ERROR: Return to caller

Register 15 = 4 - Request was not processed

Reason codes in SJACREAS:

SJRCNOER (0)	- No errors detected
SJRCIVTK (2)	- Invalid token
SJRCNKEY (202)	- Key not defined
SJRCNPRM (203)	- Parameter not defined
SJRCIVLN (500)	- Invalid length of parameter
SJRCIVCH (501)	- Invalid choice specified for parameter
SJRCGMAX (502)	- Parameter exceeds maximum
SJRCLMIN (503)	- Parameter less than minimum
SJRCIVKY (504)	- Invalid key, system specification only
SJRCIVRB (508)	- Verb not specified in the parameter list
SJRCIVLB (509)	- Label not specified in the parameter list
SJRCNLLN (510)	- Length of level exceeds the maximum
SJRCNLNM (511)	- number of levels exceeds the maximum
SJRCNFCH (512)	- Invalid first character of level

IEFSJACC - DIAGNOSTIC AIDS (Continued)

SJRCNOCH (513) - Invalid character other than first in level
SJRCNLIV (514) - Invalid specification of level
SJRCSTRA (603) - No address specified for storage area
SJRCNDAT (1300) - No data exists for this parameter
SJRCNATH (1301) - Not authorized to access this information
SJRCNSTG (1302) - Unable to obtain storage for Internal retrieve table or temporary SWA blocks
SJRCNUPD (1303) - Key not updateable
SJRCPLST (1305) - Error in parameter list
SJRCNOCB (1306) - Control block does not exist
SJRCLSTG (1307) - Storage area exceeds required amount
SJRCSSTG (1308) - Storage area less than required amount

EXIT ERROR: Return to caller

Register 15 = 20 - SJF System error

ENTRY POINT ACCRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJACC:

Register 0 = Undefined
Register 1 = Address of two words that contain the address of the input parameter list and the address of the control work area.
Registers 2-12 = Undefined
Register 13 = Address of 18-word save area
Register 14 = Return address
Register 15 = Entry point address

ENTRY POINT ACCRETRY:

Register 0 = Undefined
Register 1 = Address of ESTAE parameter list
Register 2-14 = Undefined
Register 15 = Entry point address

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJACC:

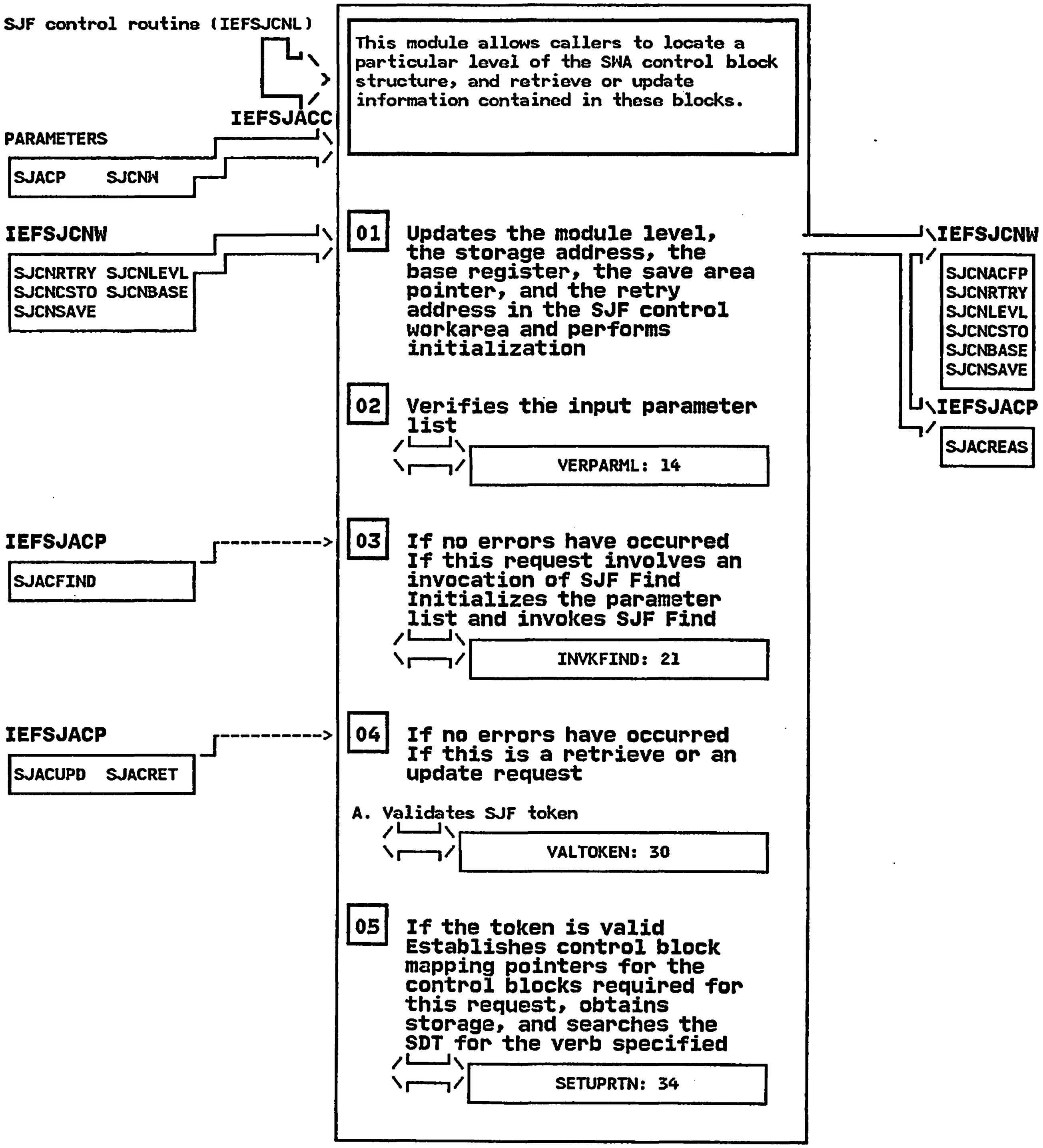
Register 0 = Restored
Register 1 = Address of two words that contain the address of the input parameter list and the address of the control work area.
Registers 2-12 = Restored
Register 13 = Address of 18-word save area
Register 14 = Return address
Register 15 = Return code

ENTRY POINT ACCRETRY:

IEFSJACC - DIAGNOSTIC AIDS (Continued)

```
Registers 0-14 = Restored
Register  15   = Return code
```

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 01



This module allows callers to locate a particular level of the SWA control block structure, and retrieve or update information contained in these blocks.

01 Updates the module level, the storage address, the base register, the save area pointer, and the retry address in the SJF control workarea and performs initialization

02 Verifies the input parameter list

VERPARML: 14

03 If no errors have occurred If this request involves an invocation of SJF Find Initializes the parameter list and invokes SJF Find

INVKFIND: 21

04 If no errors have occurred If this is a retrieve or an update request

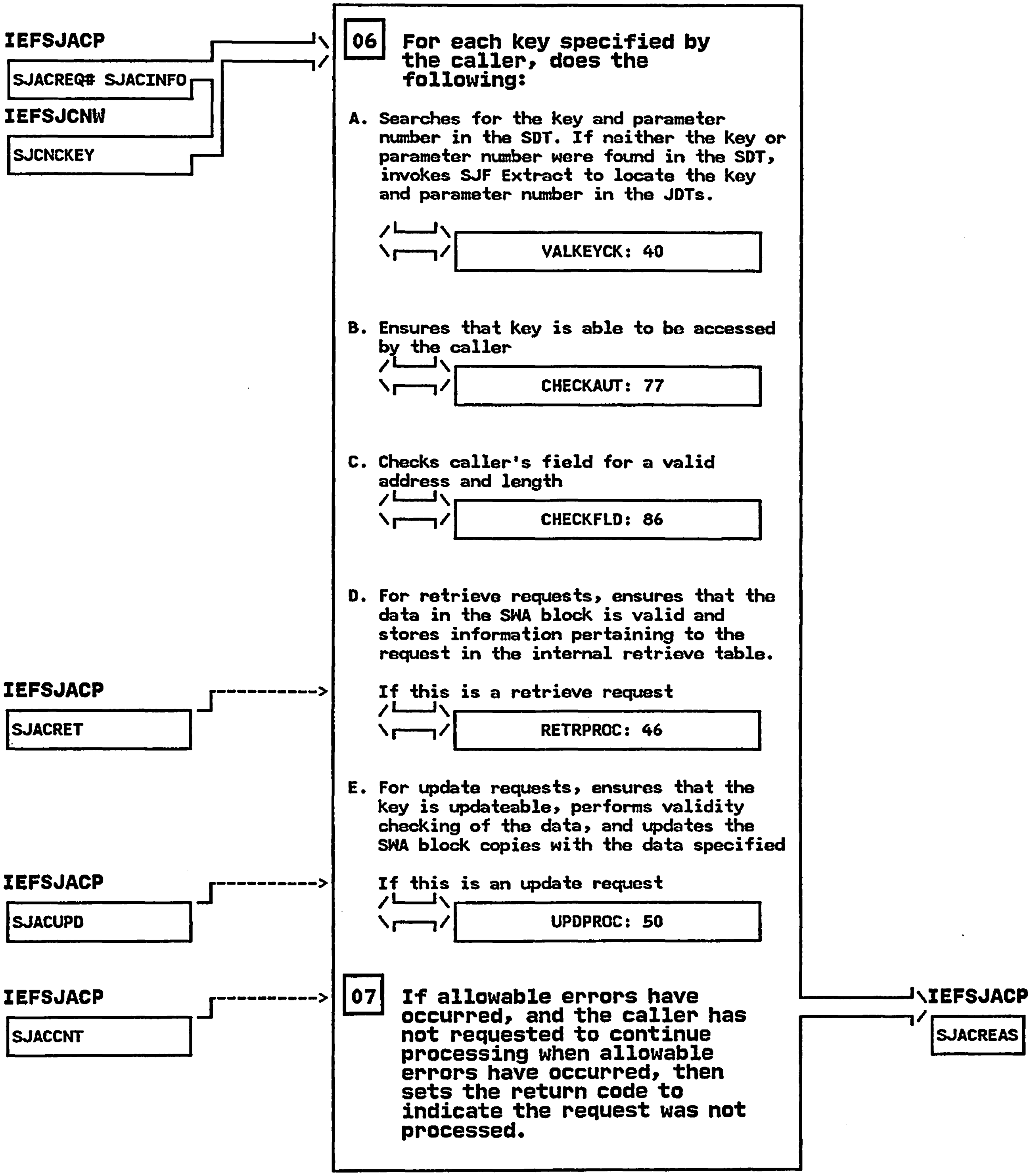
A. Validates SJF token

VALTOKEN: 30

05 If the token is valid Establishes control block mapping pointers for the control blocks required for this request, obtains storage, and searches the SDT for the verb specified

SETUPRTN: 34

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 06



06 For each key specified by the caller, does the following:

A. Searches for the key and parameter number in the SDT. If neither the key or parameter number were found in the SDT, invokes SJF Extract to locate the key and parameter number in the JDTs.

VALKEYCK: 40

B. Ensures that key is able to be accessed by the caller

CHECKAUT: 77

C. Checks caller's field for a valid address and length

CHECKFLD: 86

D. For retrieve requests, ensures that the data in the SWA block is valid and stores information pertaining to the request in the internal retrieve table.

If this is a retrieve request

RETRPROC: 46

E. For update requests, ensures that the key is updateable, performs validity checking of the data, and updates the SWA block copies with the data specified

If this is an update request

UPDPROC: 50

07 If allowable errors have occurred, and the caller has not requested to continue processing when allowable errors have occurred, then sets the return code to indicate the request was not processed.

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 08

IEFSJCNW

SJCNCKEY

IEFSJACP

SJACREAS

08 Otherwise, indicates that at least one allowable error has occurred during this invocation

IEFSJACP

SJACRSN
SJACVLKY

09 If no terminating errors have occurred or if allowable errors have occurred and the caller has requested to continue processing, then does the following:

IEFSJACP

SJACRET

If retrieve request, then retrieves information into the caller's fields

RETRINFO: 125

IEFSJACP

SJACUPD

If update request, then copies the contents of the temporary SWA blocks into the actual SWA blocks

WRITBLKS: 128

10 If entering from RTM after an ABEND, restores the data register, code register and savearea pointer, and sets the return code to indicate an SJF system error.

RTM

ACCRETRY

IEFSJCNW

SJCNCSTO SJCNBASE
SJCNSAVE

11 Performs cleanup processing. Restores the caller's module level, storage address, base register, savearea address and retry routine address.

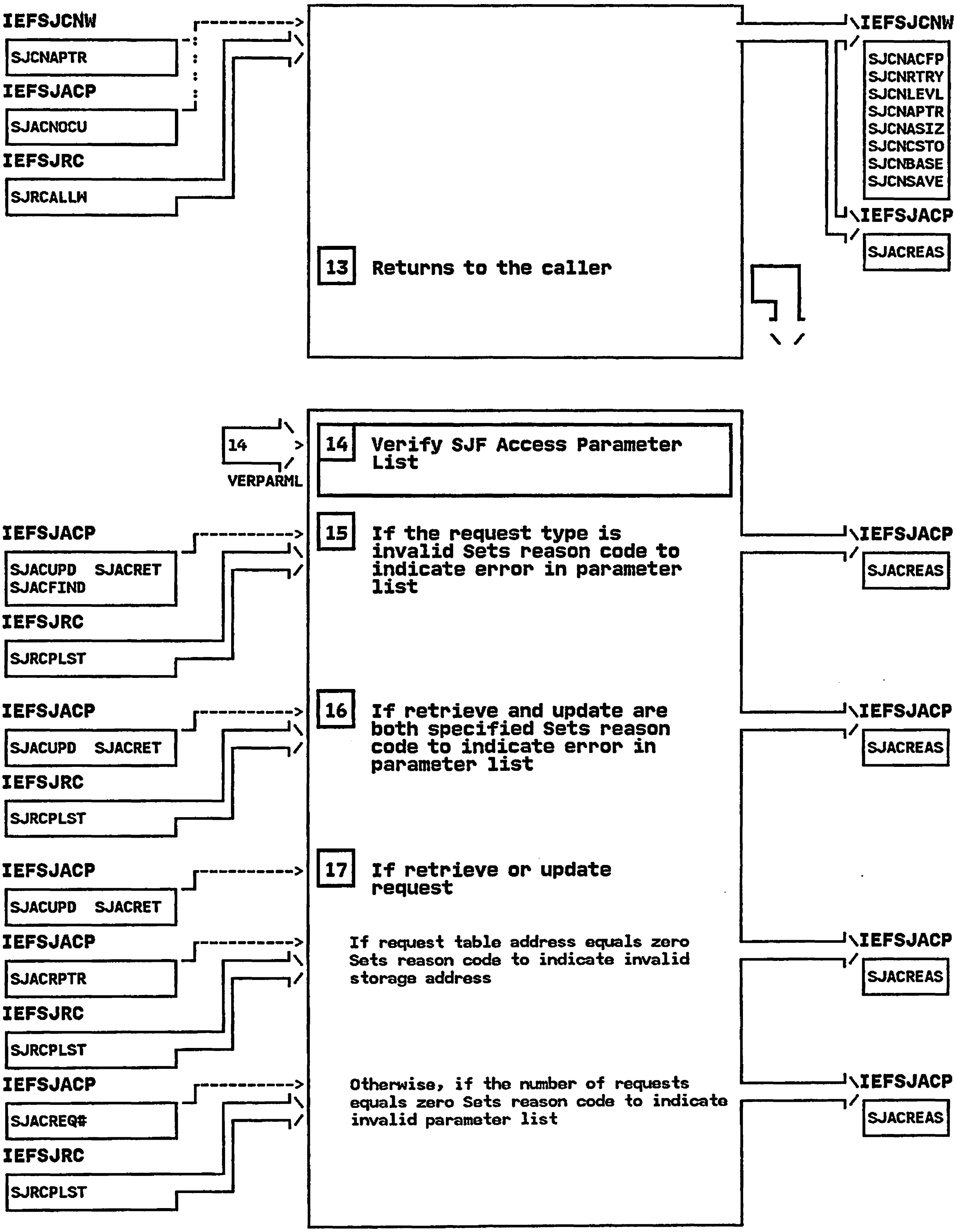
IEFSJACP

SJACUPD

12 If this is an update request If the SWB chain has been copied into local storage, but has not been copied back into the original SWB chain Determines if new SWBs have been created and if so invokes SJF Delete to delete them

DELETSWB: 136

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 13



IEFSJCNW
SJCNAPTR
IEFSJACP
SJACNOCU
IEFSJRC
SJRCALLW

13 Returns to the caller

IEFSJCNW
SJCNACFP
SJCNRTRY
SJCNLEVL
SJCNAPTR
SJCNASIZ
SJCNCSTO
SJCNBASE
SJCNSAVE
IEFSJACP
SJACREAS

14
VERPARML

14 Verify SJF Access Parameter List

IEFSJACP
SJACUPD SJACRET
SJACFIND
IEFSJRC
SJRCPLST

15 If the request type is invalid Sets reason code to indicate error in parameter list

IEFSJACP
SJACREAS

IEFSJACP
SJACUPD SJACRET
IEFSJRC
SJRCPLST

16 If retrieve and update are both specified Sets reason code to indicate error in parameter list

IEFSJACP
SJACREAS

IEFSJACP
SJACUPD SJACRET

17 If retrieve or update request

IEFSJACP
SJACRPTR
IEFSJRC
SJRCPLST

If request table address equals zero Sets reason code to indicate invalid storage address

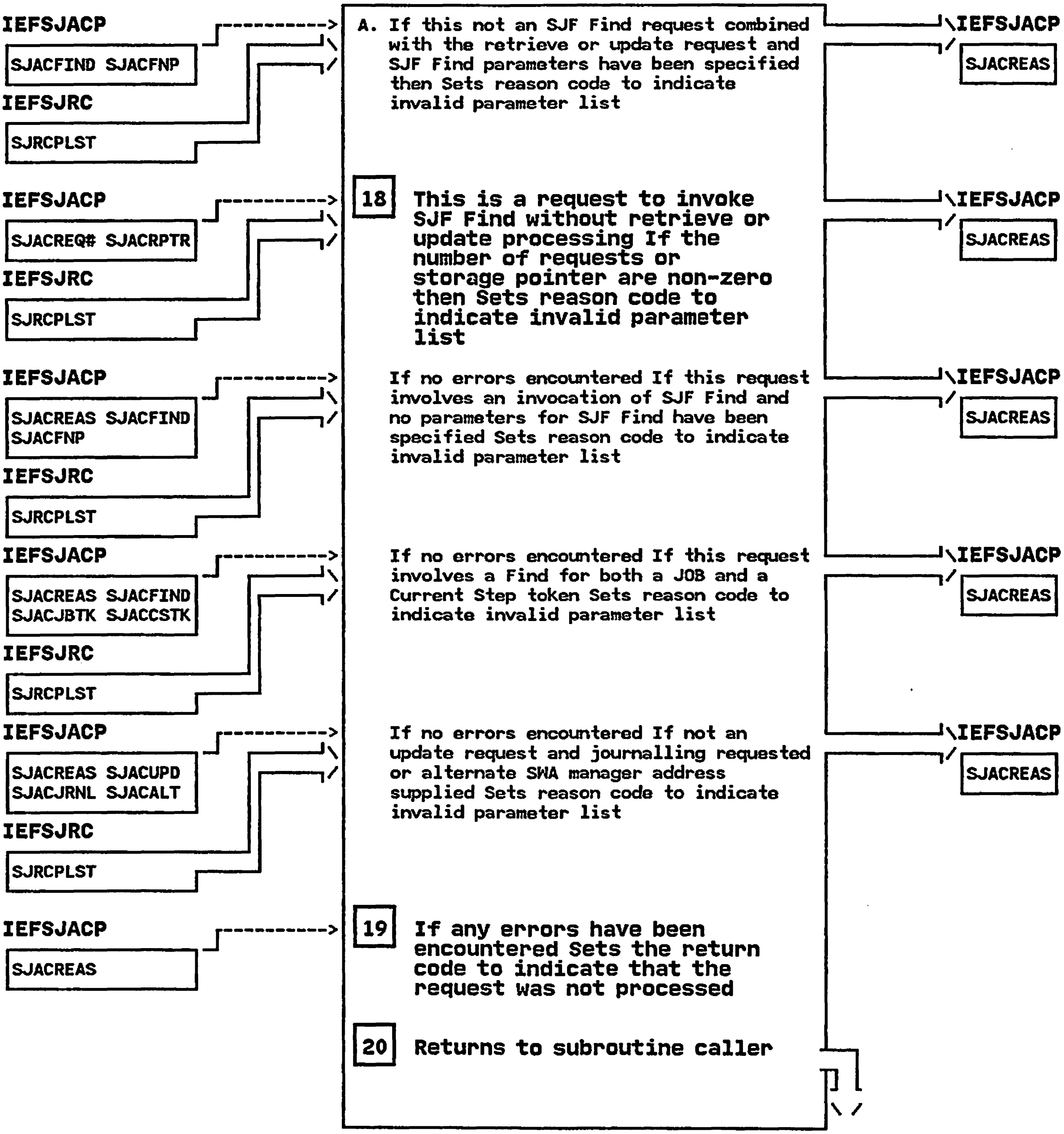
IEFSJACP
SJACREAS

IEFSJACP
SJACREQ#
IEFSJRC
SJRCPLST

Otherwise, if the number of requests equals zero Sets reason code to indicate invalid parameter list

IEFSJACP
SJACREAS

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine   STEP 17A



IEFSJACP
SJACFIND SJACFNP
IEFSJRC
SJRCPLST

A. If this not an SJF Find request combined with the retrieve or update request and SJF Find parameters have been specified then Sets reason code to indicate invalid parameter list

IEFSJACP
SJACREAS

IEFSJACP
SJACREQ# SJACRPTR
IEFSJRC
SJRCPLST

18 This is a request to invoke SJF Find without retrieve or update processing If the number of requests or storage pointer are non-zero then Sets reason code to indicate invalid parameter list

IEFSJACP
SJACREAS

IEFSJACP
SJACREAS SJACFIND SJACFNP
IEFSJRC
SJRCPLST

If no errors encountered If this request involves an invocation of SJF Find and no parameters for SJF Find have been specified Sets reason code to indicate invalid parameter list

IEFSJACP
SJACREAS

IEFSJACP
SJACREAS SJACFIND SJACJBTK SJACCSTK
IEFSJRC
SJRCPLST

If no errors encountered If this request involves a Find for both a JOB and a Current Step token Sets reason code to indicate invalid parameter list

IEFSJACP
SJACREAS

IEFSJACP
SJACREAS SJACUPD SJACJRNL SJACALT
IEFSJRC
SJRCPLST

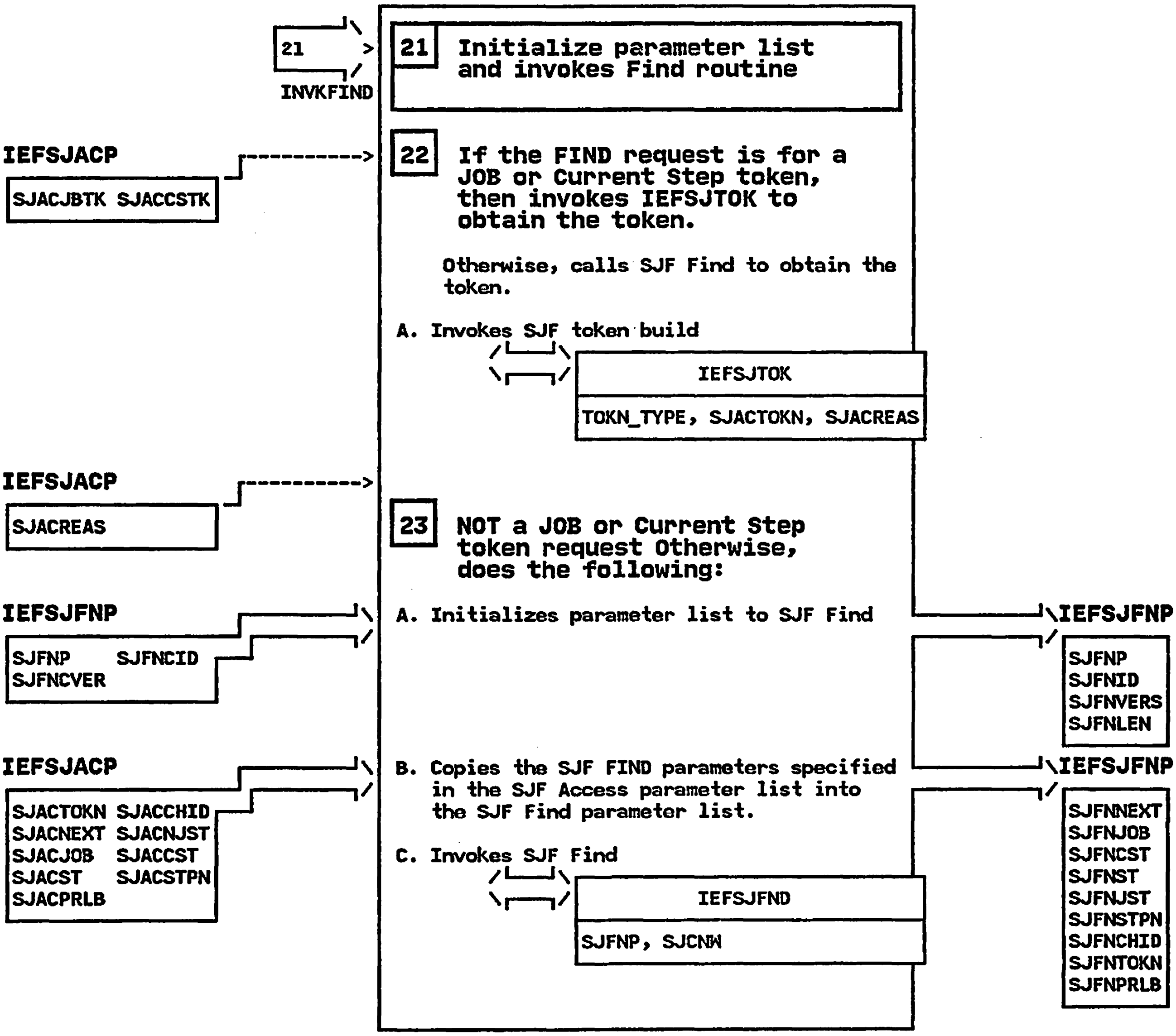
If no errors encountered If not an update request and journalling requested or alternate SWA manager address supplied Sets reason code to indicate invalid parameter list

IEFSJACP
SJACREAS

IEFSJACP
SJACREAS

19 If any errors have been encountered Sets the return code to indicate that the request was not processed

20 Returns to subroutine caller



21 Initialize parameter list and invokes Find routine

IEFSJACP: SJACJBTK SJACCSTK

22 If the FIND request is for a JOB or Current Step token, then invokes IEFSJTOK to obtain the token.

Otherwise, calls SJF Find to obtain the token.

A. Invokes SJF token build

IEFSJTOK: TOKN_TYPE, SJACTOKN, SJACREAS

IEFSJACP: SJACREAS

23 NOT a JOB or Current Step token request Otherwise, does the following:

IEFSJFNP: SJFNP SJFNCID SJFNCVER

A. Initializes parameter list to SJF Find

IEFSJFNP: SJFNP SJFNID SJFNVERS SJFNLEN

IEFSJACP: SJACTOKN SJACCHID SJACNEXT SJACNJST SJACJOB SJACCST SJACST SJACSTPN SJACPRLB

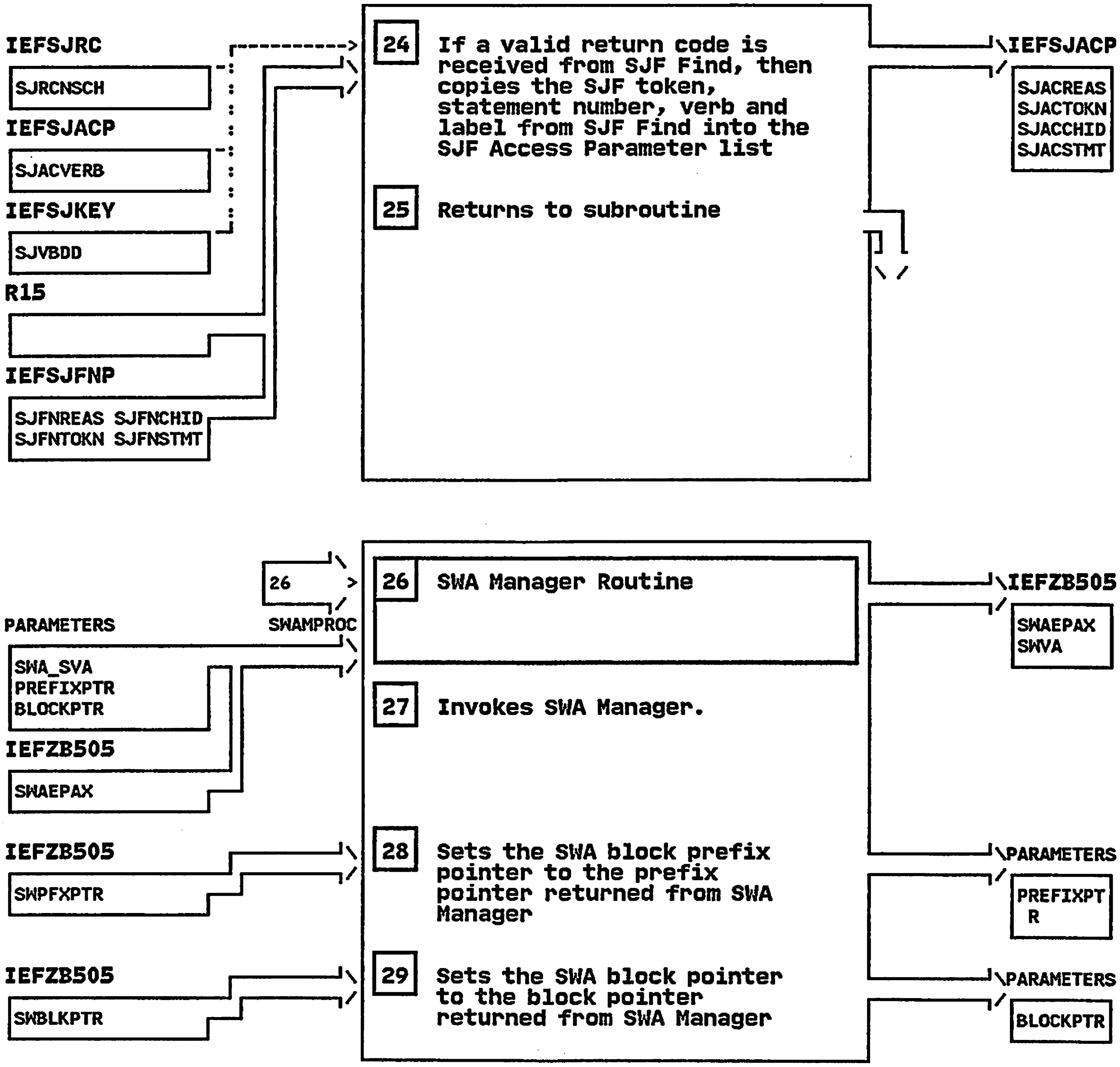
B. Copies the SJF FIND parameters specified in the SJF Access parameter list into the SJF Find parameter list.

IEFSJFNP: SJFNNEXT SJFNJOB SJFNCST SJFNST SJFNJST SJFNSTPN SJFNCHID SJFNTOKN SJFNPRLB

C. Invokes SJF Find

IEFSJFND: SJFNP, SJCNW

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 24



24 If a valid return code is received from SJF Find, then copies the SJF token, statement number, verb and label from SJF Find into the SJF Access Parameter list

25 Returns to subroutine

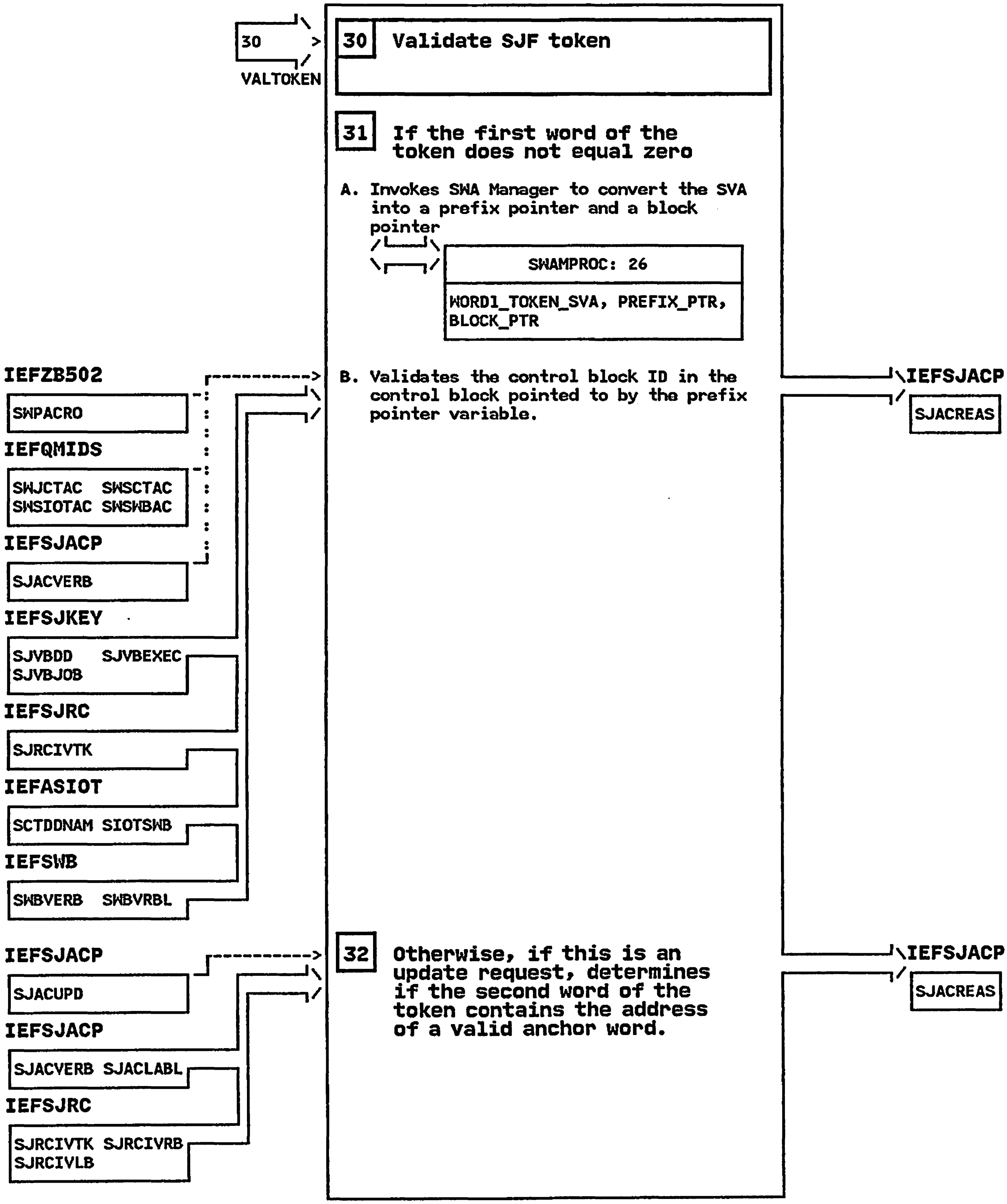
26 SWA Manager Routine

27 Invokes SWA Manager.

28 Sets the SWA block prefix pointer to the prefix pointer returned from SWA Manager

29 Sets the SWA block pointer to the block pointer returned from SWA Manager

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 30



30 VALTOKEN

30 Validate SJF token

31 If the first word of the token does not equal zero

A. Invokes SWA Manager to convert the SVA into a prefix pointer and a block pointer

SWAMPROC: 26

WORD1_TOKEN_SVA, PREFIX_PTR, BLOCK_PTR

B. Validates the control block ID in the control block pointed to by the prefix pointer variable.

IEFZB502: SWPACRO

IEFQMIDS: SWJCTAC SWSCTAC SWSIOTAC SWSWBAC

IEFSJACP: SJACVERB

IEFSJKEY: SJVBDD SJVBEXEC SJVBJOB

IEFSJRC: SJRCIVTK

IEFASIOT: SCTDDNAM SIOTSWB

IEFSWB: SWBVERB SWBVRBL

IEFSJACP: SJACREAS

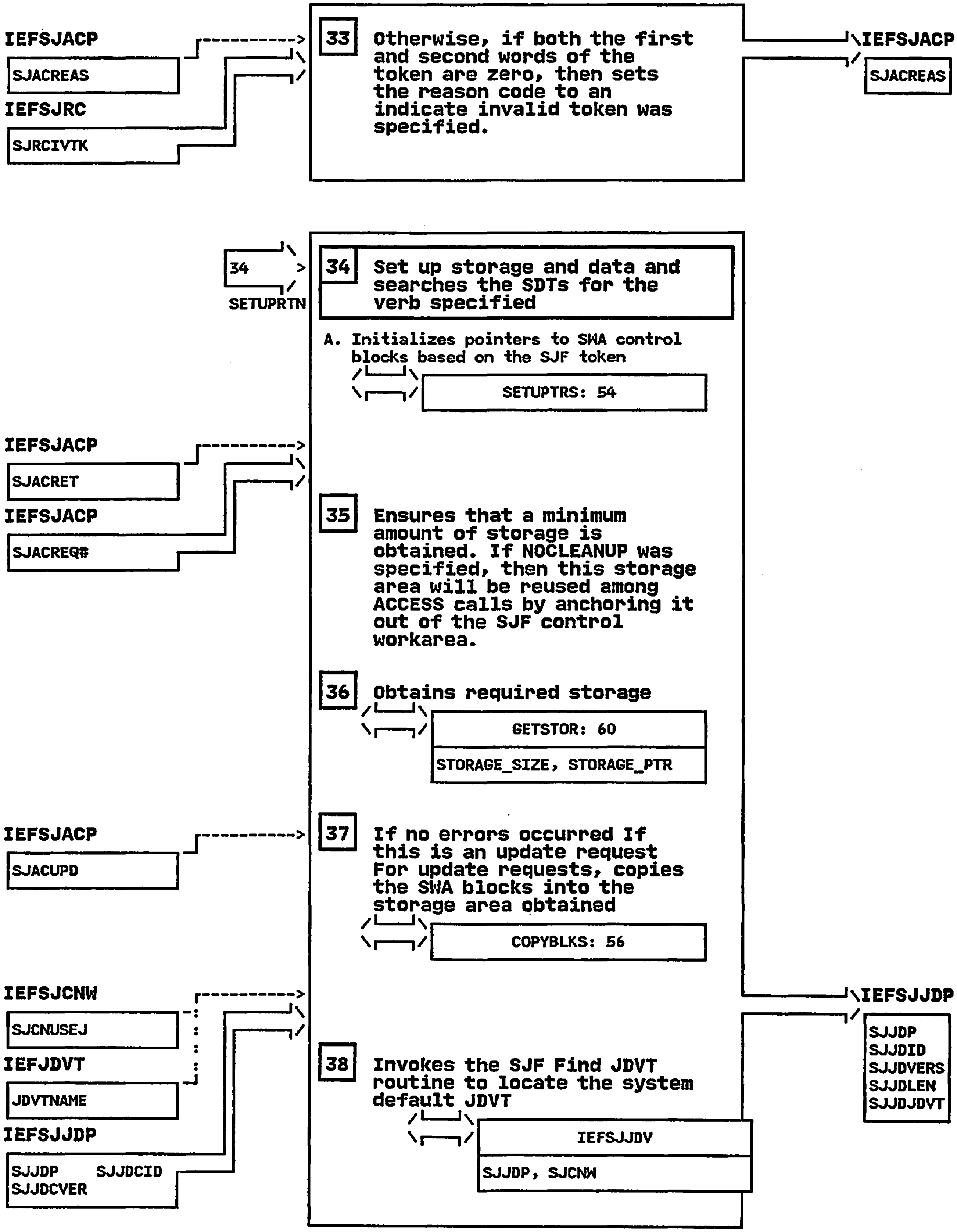
32 Otherwise, if this is an update request, determines if the second word of the token contains the address of a valid anchor word.

IEFSJACP: SJACUPD

IEFSJACP: SJACVERB SJACLABL

IEFSJRC: SJRCIVTK SJRCIVRB SJRCIVLB

IEFSJACP: SJACREAS



33 Otherwise, if both the first and second words of the token are zero, then sets the reason code to an indicate invalid token was specified.

34 Set up storage and data and searches the SDTs for the verb specified

A. Initializes pointers to SWA control blocks based on the SJF token

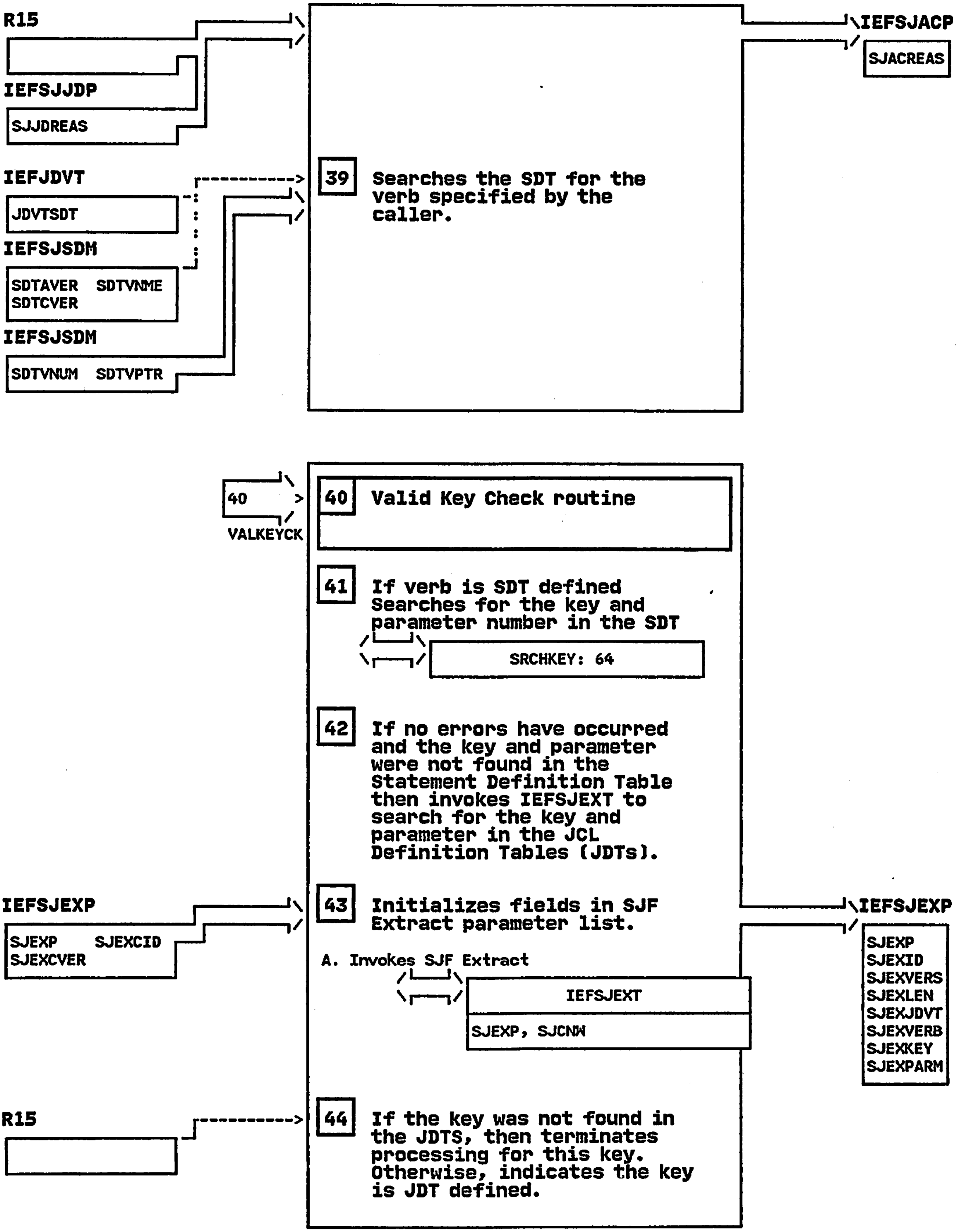
35 Ensures that a minimum amount of storage is obtained. If NOCLEANUP was specified, then this storage area will be reused among ACCESS calls by anchoring it out of the SJF control workarea.

36 Obtains required storage

37 If no errors occurred If this is an update request For update requests, copies the SWA blocks into the storage area obtained

38 Invokes the SJF Find JDVT routine to locate the system default JDVT

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 39



39 Searches the SDT for the verb specified by the caller.

40 Valid Key Check routine

41 If verb is SDT defined Searches for the key and parameter number in the SDT

SRCHKEY: 64

42 If no errors have occurred and the key and parameter were not found in the Statement Definition Table then invokes IEFSJEXT to search for the key and parameter in the JCL Definition Tables (JDTs).

43 Initializes fields in SJF Extract parameter list.

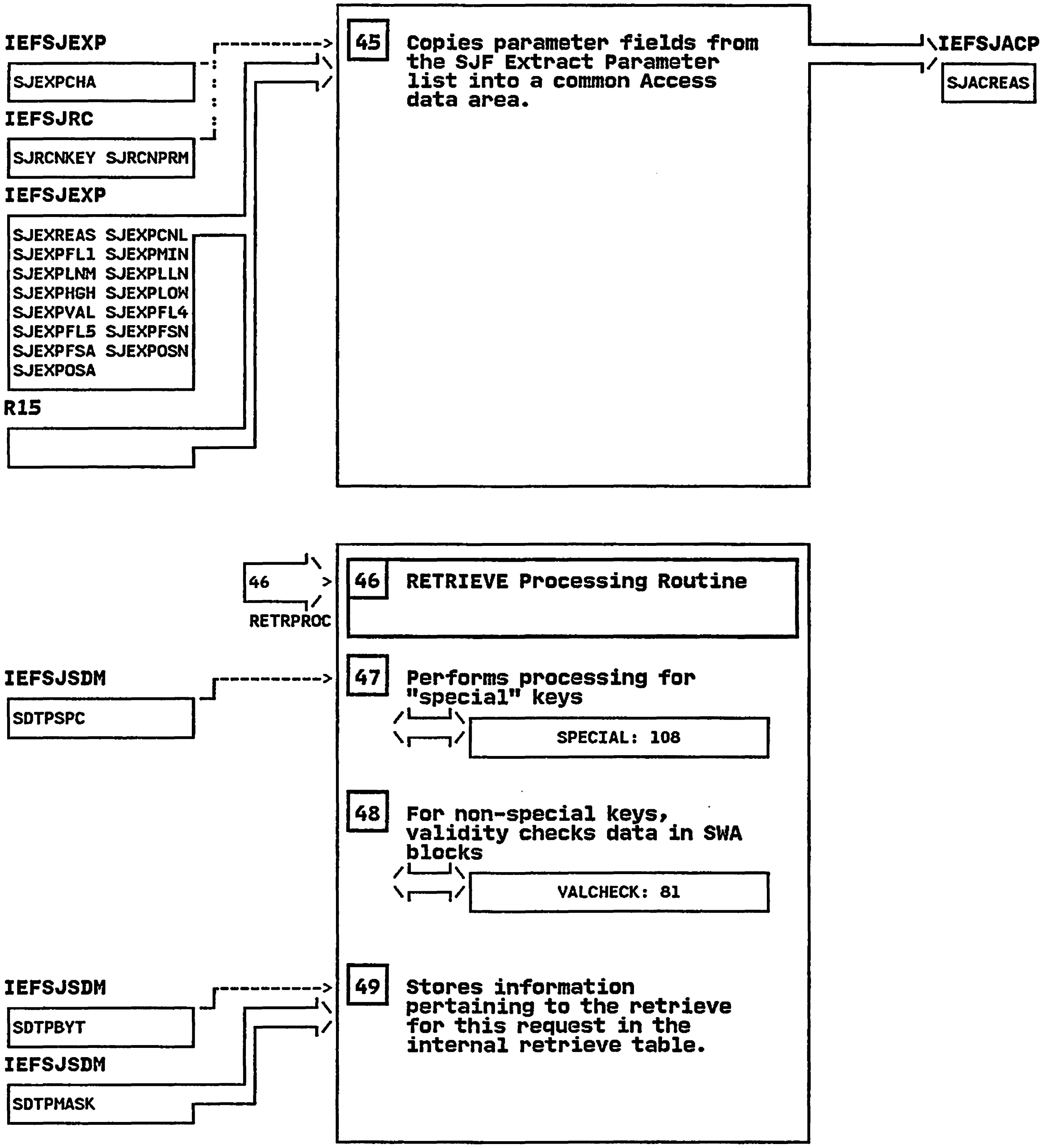
A. Invokes SJF Extract

IEFSJEXT

SJEXP, SJCNW

44 If the key was not found in the JDTS, then terminates processing for this key. Otherwise, indicates the key is JDT defined.

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 45



IEFSJEXP

SJEXPCHA

IEFSJRC

SJRCNKEY SJRCNPRM

IEFSJEXP

SJEXREAS SJEXPCNL
SJEXPFL1 SJEXPMIN
SJEXPLNM SJEXPLLN
SJEXPHGH SJEXPLOW
SJEXPVAL SJEXPFL4
SJEXPFL5 SJEXPFSN
SJEXPFSA SJEXPOSN
SJEXPOSA

R15

45 Copies parameter fields from the SJF Extract Parameter list into a common Access data area.

IEFSJACP

SJACREAS

46

RETRPROC

46 RETRIEVE Processing Routine

IEFSJSDM

SDTPSPC

47 Performs processing for "special" keys

SPECIAL: 108

48 For non-special keys, validity checks data in SWA blocks

VALCHECK: 81

IEFSJSDM

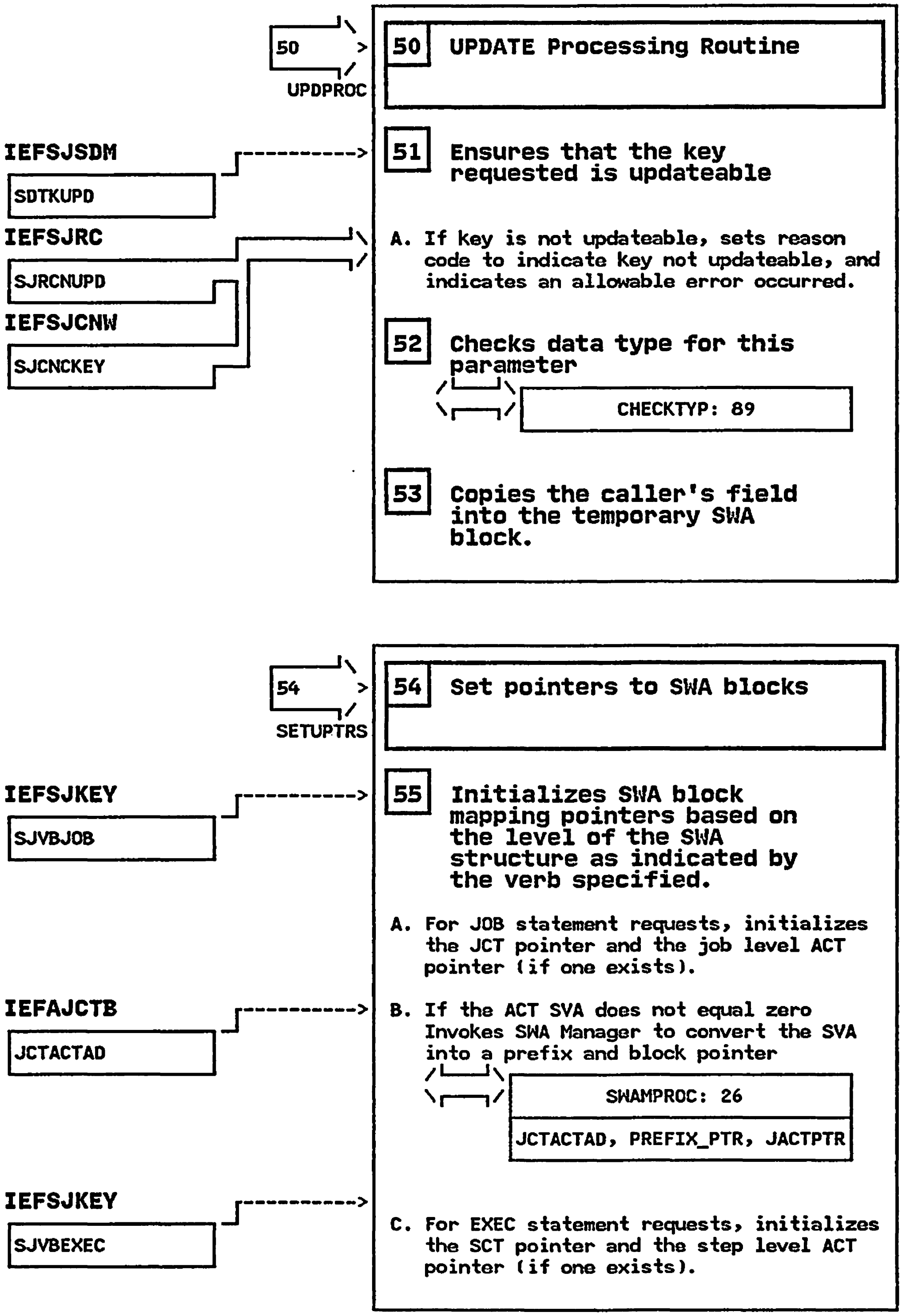
SDTPBYT

IEFSJSDM

SDTPMASK

49 Stores information pertaining to the retrieve for this request in the internal retrieve table.

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine   STEP 50



50 UPDPROC

50 UPDATE Processing Routine

IEFSJSDM SDTKUPD

51 Ensures that the key requested is updateable

IEFSJRC SJRCNUPD
IEFSJCNW SJCNCKEY

A. If key is not updateable, sets reason code to indicate key not updateable, and indicates an allowable error occurred.

52 Checks data type for this parameter

CHECKTYP: 89

53 Copies the caller's field into the temporary SWA block.

54 SETUPTRS

54 Set pointers to SWA blocks

IEFSJKEY SJVBJOB

55 Initializes SWA block mapping pointers based on the level of the SWA structure as indicated by the verb specified.

A. For JOB statement requests, initializes the JCT pointer and the job level ACT pointer (if one exists).

IEFAJCTB JCTACTAD

B. If the ACT SVA does not equal zero Invokes SWA Manager to convert the SVA into a prefix and block pointer

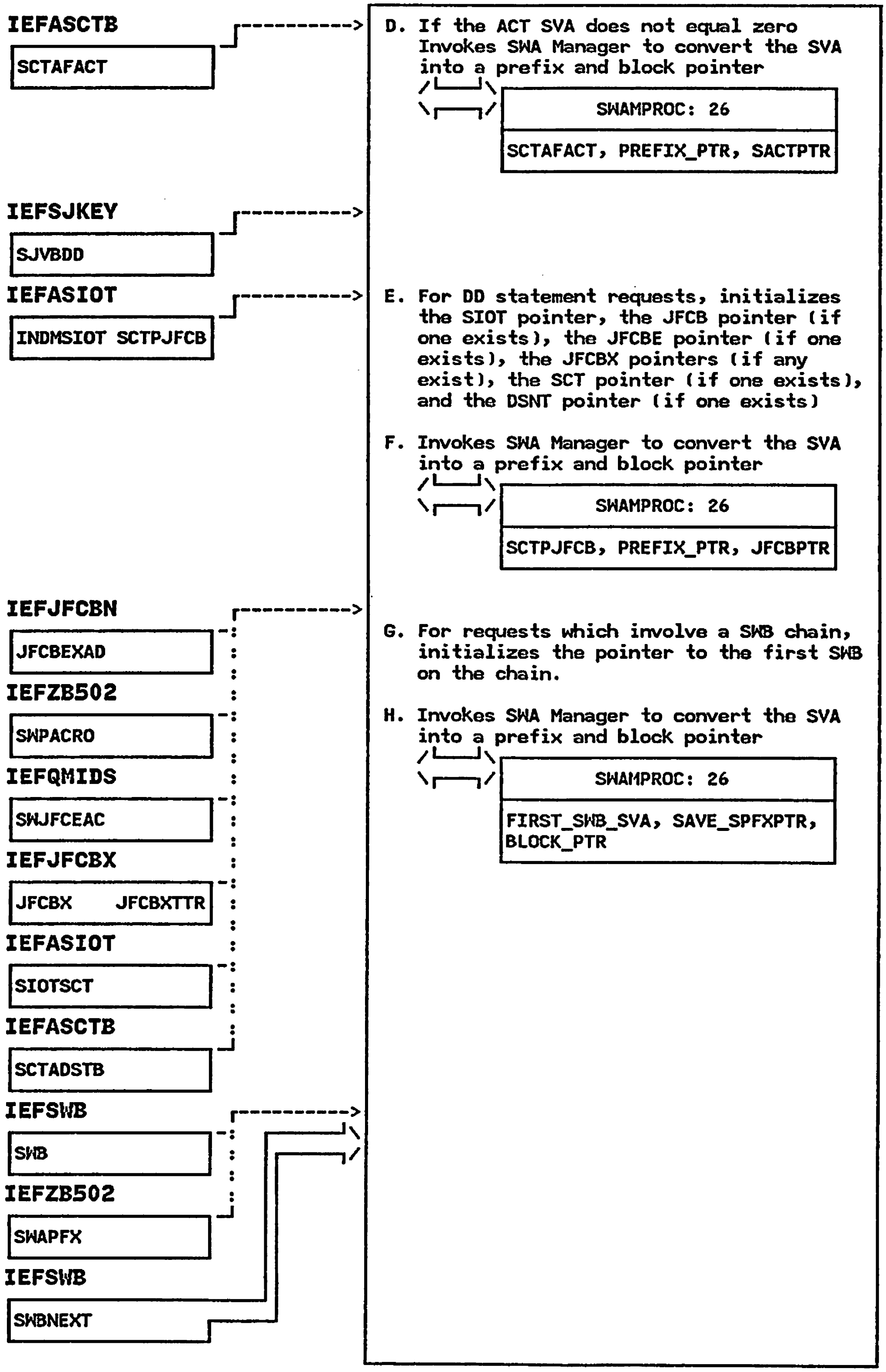
SWAMPROC: 26
JCTACTAD, PREFIX_PTR, JACTPTR

IEFSJKEY SJVBEXEC

C. For EXEC statement requests, initializes the SCT pointer and the step level ACT pointer (if one exists).

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine

STEP 55D



D. If the ACT SVA does not equal zero Invokes SWA Manager to convert the SVA into a prefix and block pointer

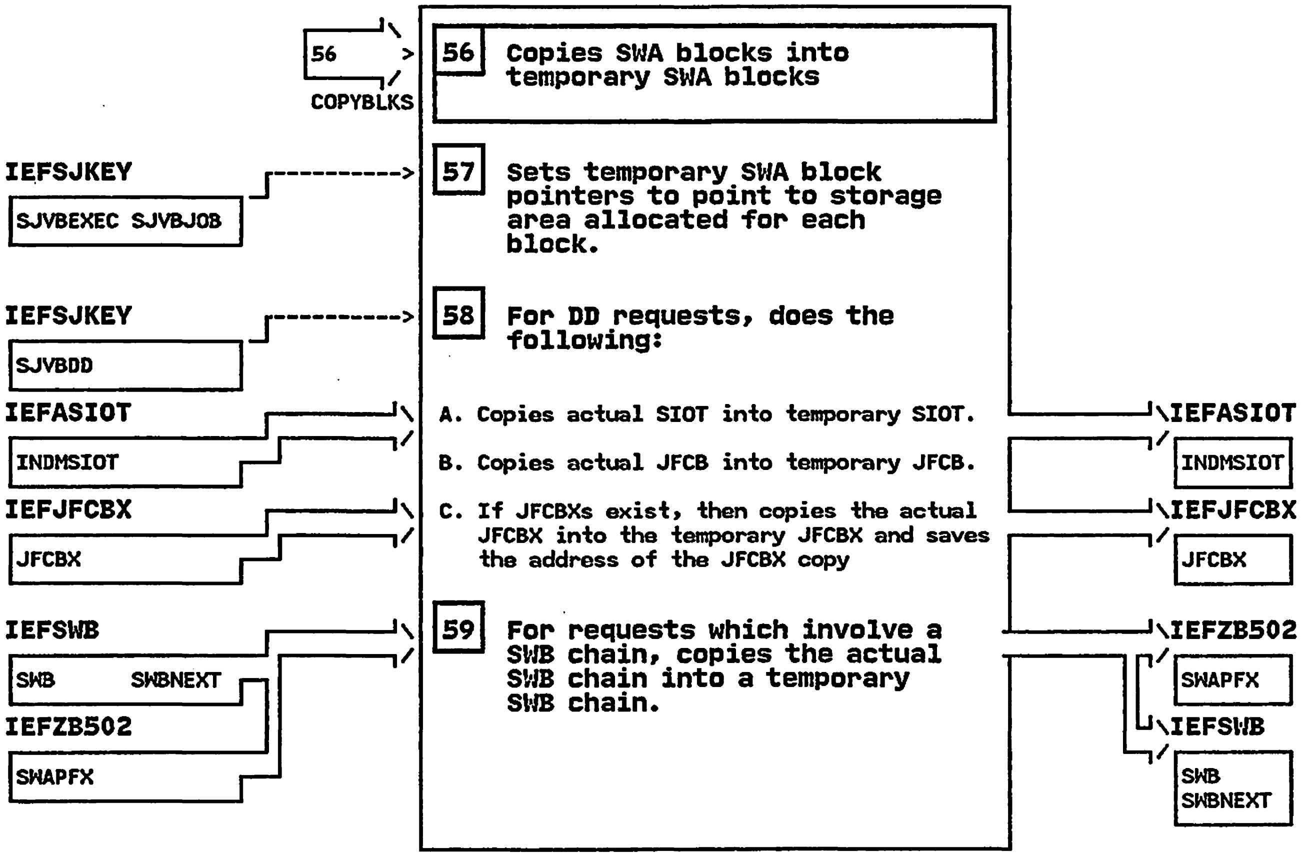
E. For DD statement requests, initializes the SIOT pointer, the JFCB pointer (if one exists), the JFCBE pointer (if one exists), the JFCBX pointers (if any exist), the SCT pointer (if one exists), and the DSNT pointer (if one exists)

F. Invokes SWA Manager to convert the SVA into a prefix and block pointer

G. For requests which involve a SWB chain, initializes the pointer to the first SWB on the chain.

H. Invokes SWA Manager to convert the SVA into a prefix and block pointer

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 56



56 Copies SWA blocks into temporary SWA blocks

57 Sets temporary SWA block pointers to point to storage area allocated for each block.

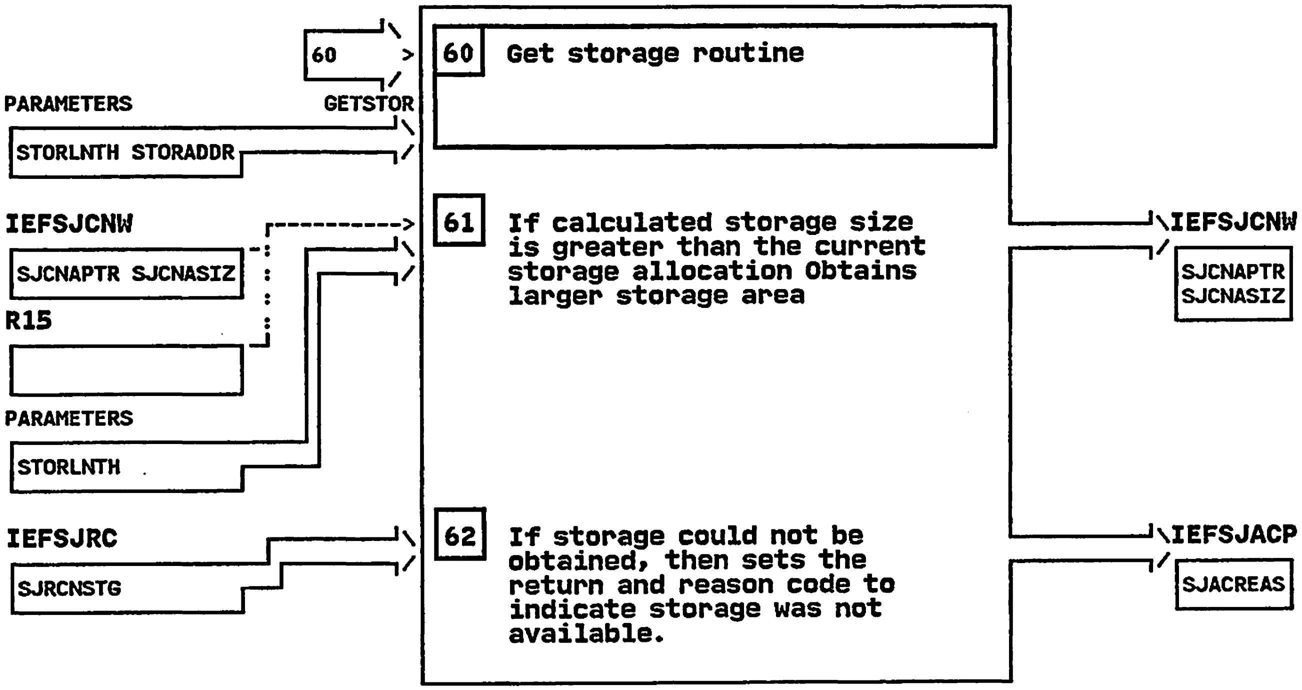
58 For DD requests, does the following:

A. Copies actual SIOT into temporary SIOT.

B. Copies actual JFCB into temporary JFCB.

C. If JFCBXs exist, then copies the actual JFCBX into the temporary JFCBX and saves the address of the JFCBX copy

59 For requests which involve a SWB chain, copies the actual SWB chain into a temporary SWB chain.



60 Get storage routine

61 If calculated storage size is greater than the current storage allocation Obtains larger storage area

62 If storage could not be obtained, then sets the return and reason code to indicate storage was not available.

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine

STEP 63

IEFSJCNW

SJCNAPTR SJCNASIZ

63 If GETMAIN was successful, or no GETMAIN was necessary, then zeroes the storage area obtained.

PARAMETERS

STORADDR

64

SRCHKEY

64 Search Statement mapping for Key

Searches statement mapping for the key and parameter number specified.

IEFSJSDM

SDTVKEYB SDTVKEYM

IEFSJSDM

SDTVKPTR

65 Checks if key specified in the parameter list is in the range of keys allowed for this verb.

IEFSJSDM

SDTKPNUM

IEFSJSDM

SDTKPPTR

66 Searches statement mapping for the key and parameter number specified

67 If the key and parameter number are found in the SDT, obtains addressability to the control block that contains the information to be retrieved or updated.

68

SET_BLOCK_PTR: 73

SDTPCBLK, SWA_BLOCK_PTR, COPY_SWA_BLOCK_PTR

IEFSJACP

SJACRET

IEFSJSDM

SDTPDOFF SDTPMAX SDTPMIN

69 Obtains addressability to the field in SWA by adding the offset of the SWA field specified in the SDT to the pointer to the control block containing the field

IEFSJSDM

SDTPDOFF

70 Gains addressability to the temporary SWA field by adding the offset of the SWA field (in stmt mapping) to the pointer to the temporary SWA block

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 71

IEFSJSDM

SDTPCHCA

IEFSJACP

SJACRET

IEFSJSDM

SDTPTYP SDTPBYT
SDTPSPC SDTPCRC
SDTPINC SDTPCVAL

IEFSJRC

SJRCNPRM SJRCNDAT
SJRCNOCB

71 Sets fields in the common parameter area to be used by the common include routines to perform parameter checking

72 Returns to subroutine caller

73

SET_BLOCK_PTR

PARAMETERS

CBACRO CBPTR
COPYPTR

73 Set control block pointer

FUNCTION: Obtains addressability to the control block associated with a particular SWA block acronym

ENTRY: From SRCHKEY, VALCHECK, UPDBLKS

INPUT: CBACRO = Control block acronym

OUTPUT: CBPTR = Pointer to the SWA control block COPYPTR = Pointer to the SWA control block copy

EXIT: To SRCHKEY, VALCHECK, UPDBLKS

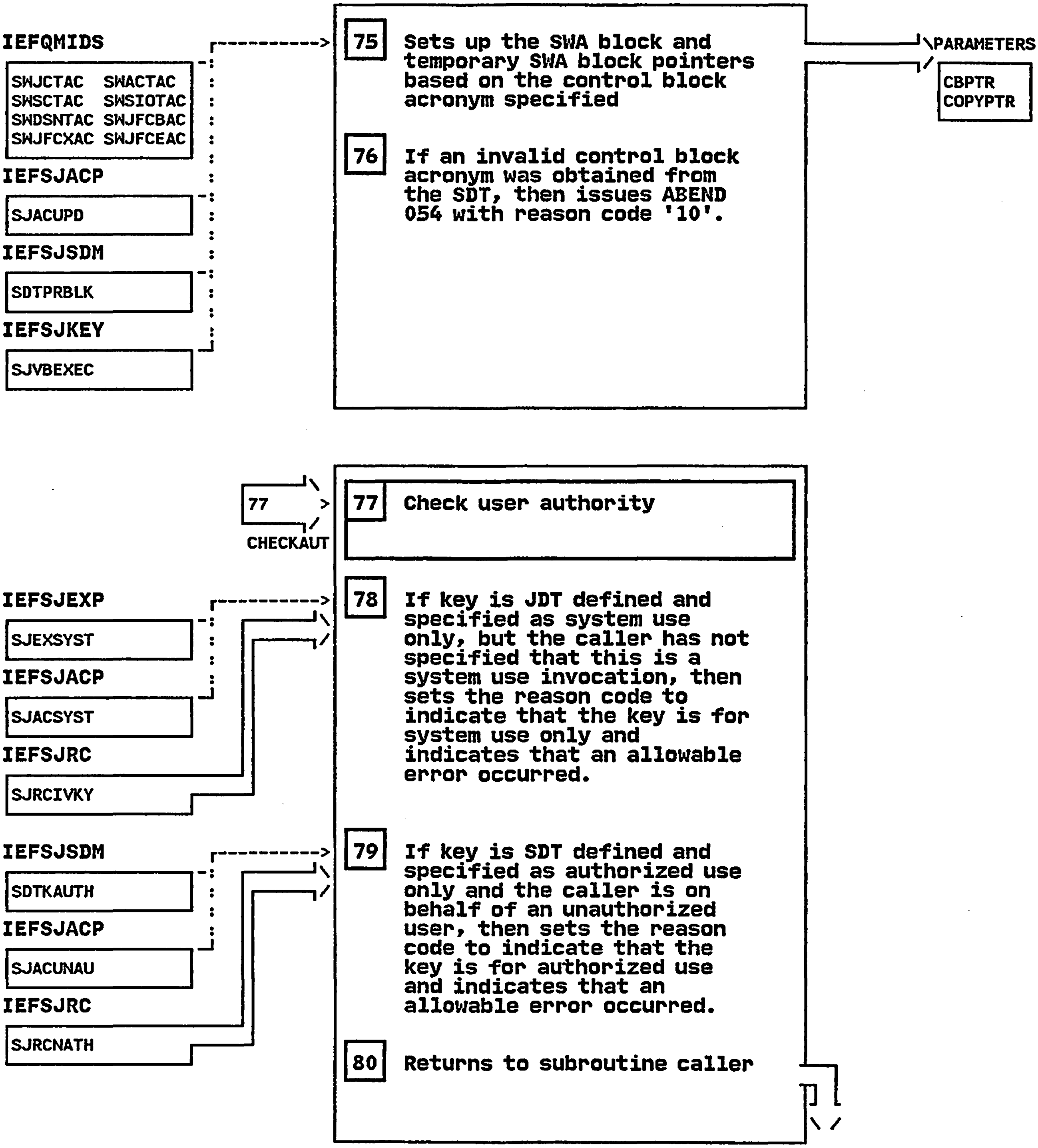
RETURN CODE: RETURN_CODE

PARAMETERS

CBACRO

74 Checks the SWA acronym specified in the SDT

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 75



75 Sets up the SWA block and temporary SWA block pointers based on the control block acronym specified

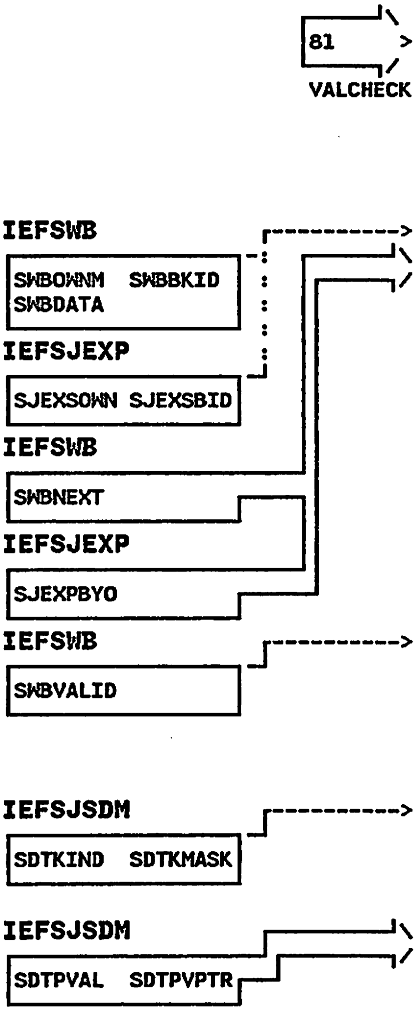
76 If an invalid control block acronym was obtained from the SDT, then issues ABEND 054 with reason code '10'.

77 Check user authority

78 If key is JDT defined and specified as system use only, but the caller has not specified that this is a system use invocation, then sets the reason code to indicate that the key is for system use only and indicates that an allowable error occurred.

79 If key is SDT defined and specified as authorized use only and the caller is on behalf of an unauthorized user, then sets the reason code to indicate that the key is for authorized use and indicates that an allowable error occurred.

80 Returns to subroutine caller



81 Validity Check Data

82 For JDT defined data does the following:

A. Checks if the SWB containing the data is on the SWB chain

B. If the SWB is found, checks the validity bit for the data requested.

83 For SDT defined data does the following:

A. For a select set of keys, determines if the key was user-specified (via JCL or Dynamic Allocation), or defaulted by the system.

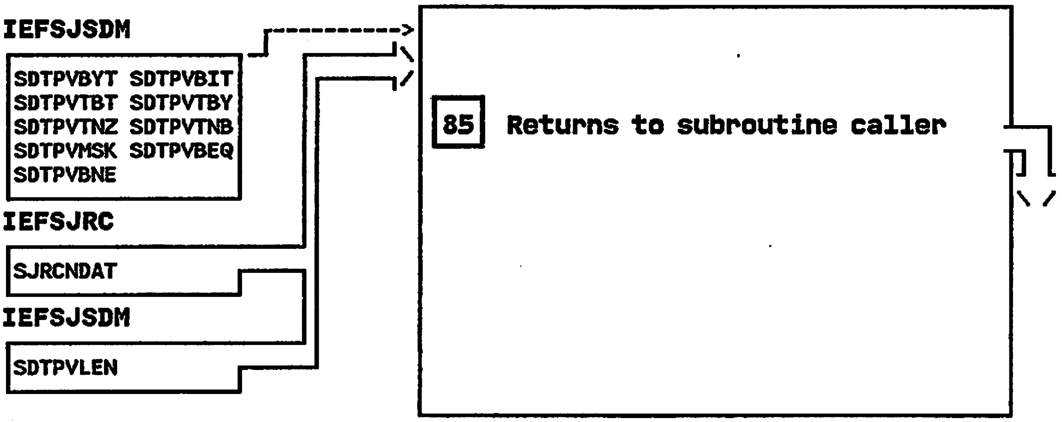
B. Performs the validity checking as defined in the SDT

84 Obtains addressability to the control block containing the validity indicator

SET_BLOCK_PTR: 73

VALIDPTR->SDTPVCB,
VAL_BLOCK_PTR,
COPY_VAL_BLOCK_PTR

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 85



IEFSJSDM

SDTPVBYT SDTPVBIT
SDTPVTBT SDTPVTBY
SDTPVTNZ SDTPVTNB
SDTPVMSK SDTPVBEQ
SDTPVBNE

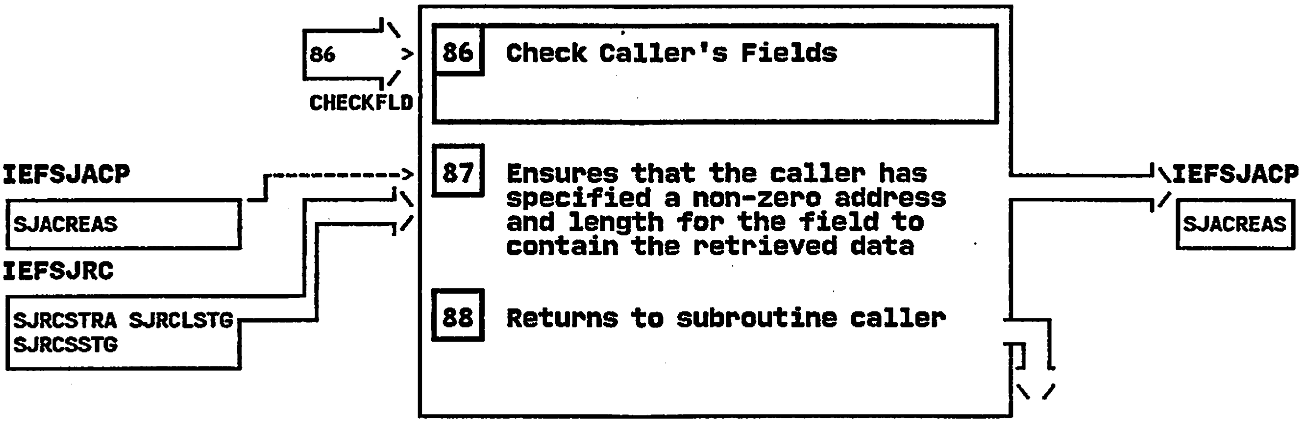
IEFSJRC

SJRCNDAT

IEFSJSDM

SDTPVLEN

85 Returns to subroutine caller



86 CHECKFLD

86 Check Caller's Fields

IEFSJACP

SJACREAS

IEFSJRC

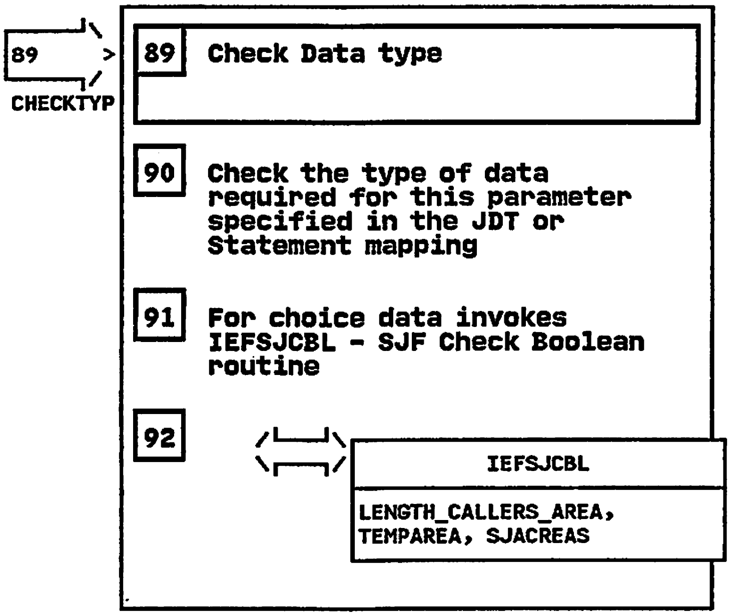
SJRCSTRA SJRCLSTG
SJRCSSTG

87 Ensures that the caller has specified a non-zero address and length for the field to contain the retrieved data

IEFSJACP

SJACREAS

88 Returns to subroutine caller



89 CHECKTYP

89 Check Data type

90 Check the type of data required for this parameter specified in the JDT or Statement mapping

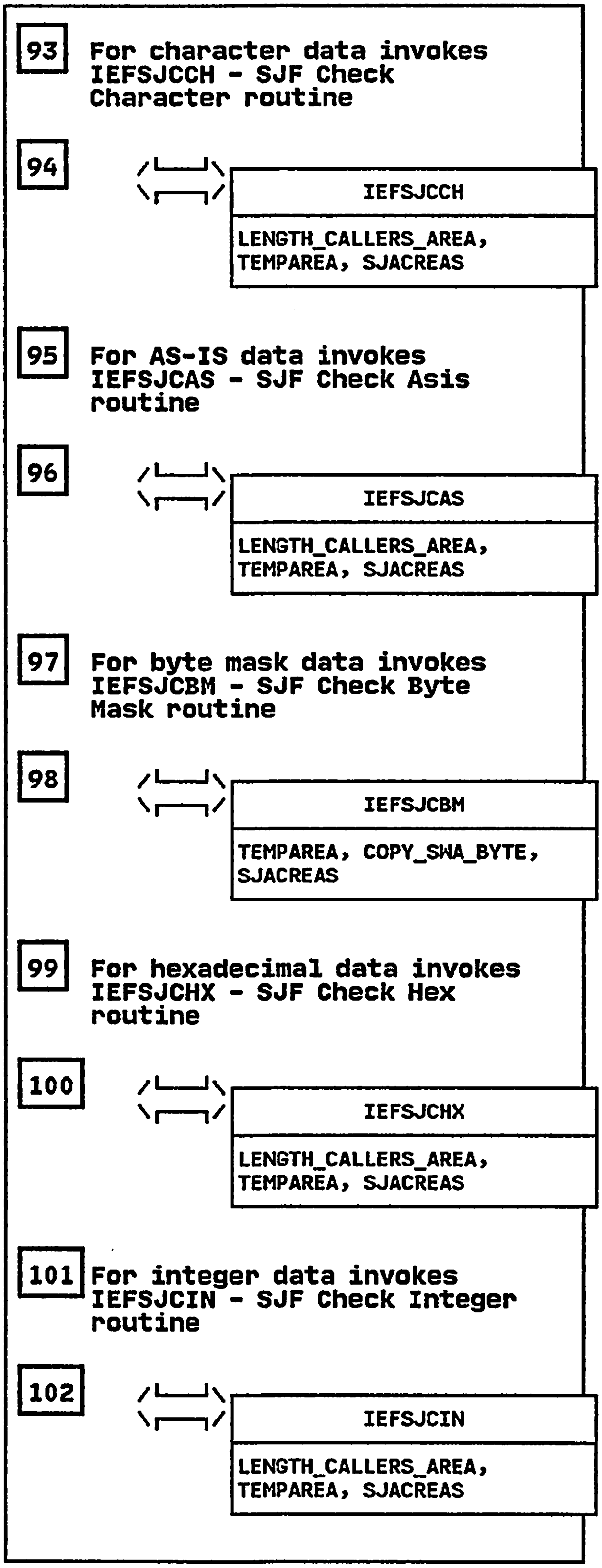
91 For choice data invokes IEFSJCBL - SJF Check Boolean routine

92

IEFSJCBL

LENGTH_CALLERS_AREA,
TEMPAREA, SJACREAS

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 93



93 For character data invokes IEFSJCCH - SJF Check Character routine

94 IEFSJCCH
LENGTH_CALLERS_AREA, TEMPAREA, SJACREAS

95 For AS-IS data invokes IEFSJCAS - SJF Check Asis routine

96 IEFSJCAS
LENGTH_CALLERS_AREA, TEMPAREA, SJACREAS

97 For byte mask data invokes IEFSJCBM - SJF Check Byte Mask routine

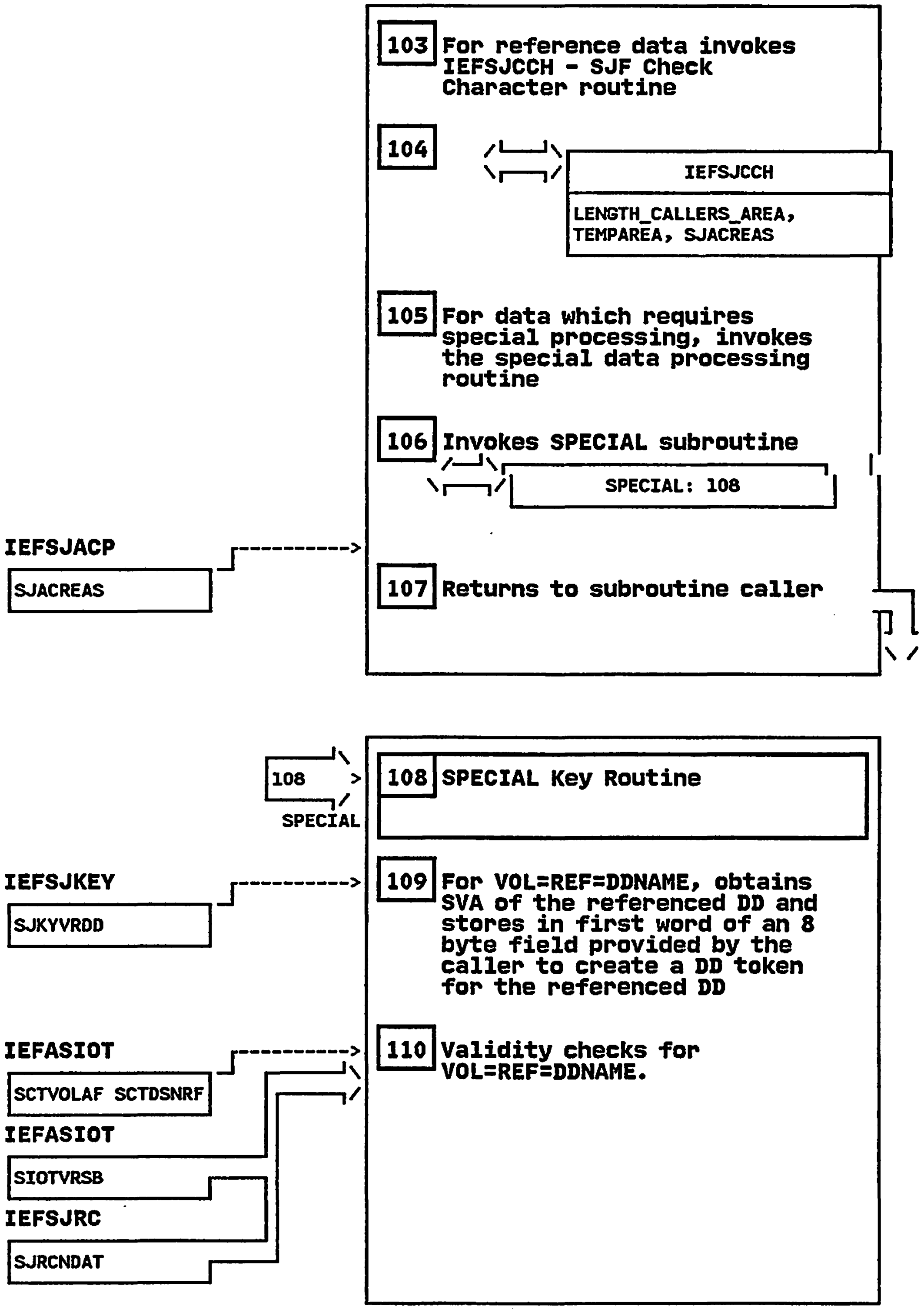
98 IEFSJCBM
TEMPAREA, COPY_SWA_BYTE, SJACREAS

99 For hexadecimal data invokes IEFSJCHX - SJF Check Hex routine

100 IEFSJCHX
LENGTH_CALLERS_AREA, TEMPAREA, SJACREAS

101 For integer data invokes IEFSJCIN - SJF Check Integer routine

102 IEFSJCIN
LENGTH_CALLERS_AREA, TEMPAREA, SJACREAS



103 For reference data invokes IEFSJCCH - SJF Check Character routine

104 IEFSJCCH
LENGTH_CALLERS_AREA, TEMPAREA, SJACREAS

105 For data which requires special processing, invokes the special data processing routine

106 Invokes SPECIAL subroutine
SPECIAL: 108

IEFSJACP
SJACREAS

107 Returns to subroutine caller

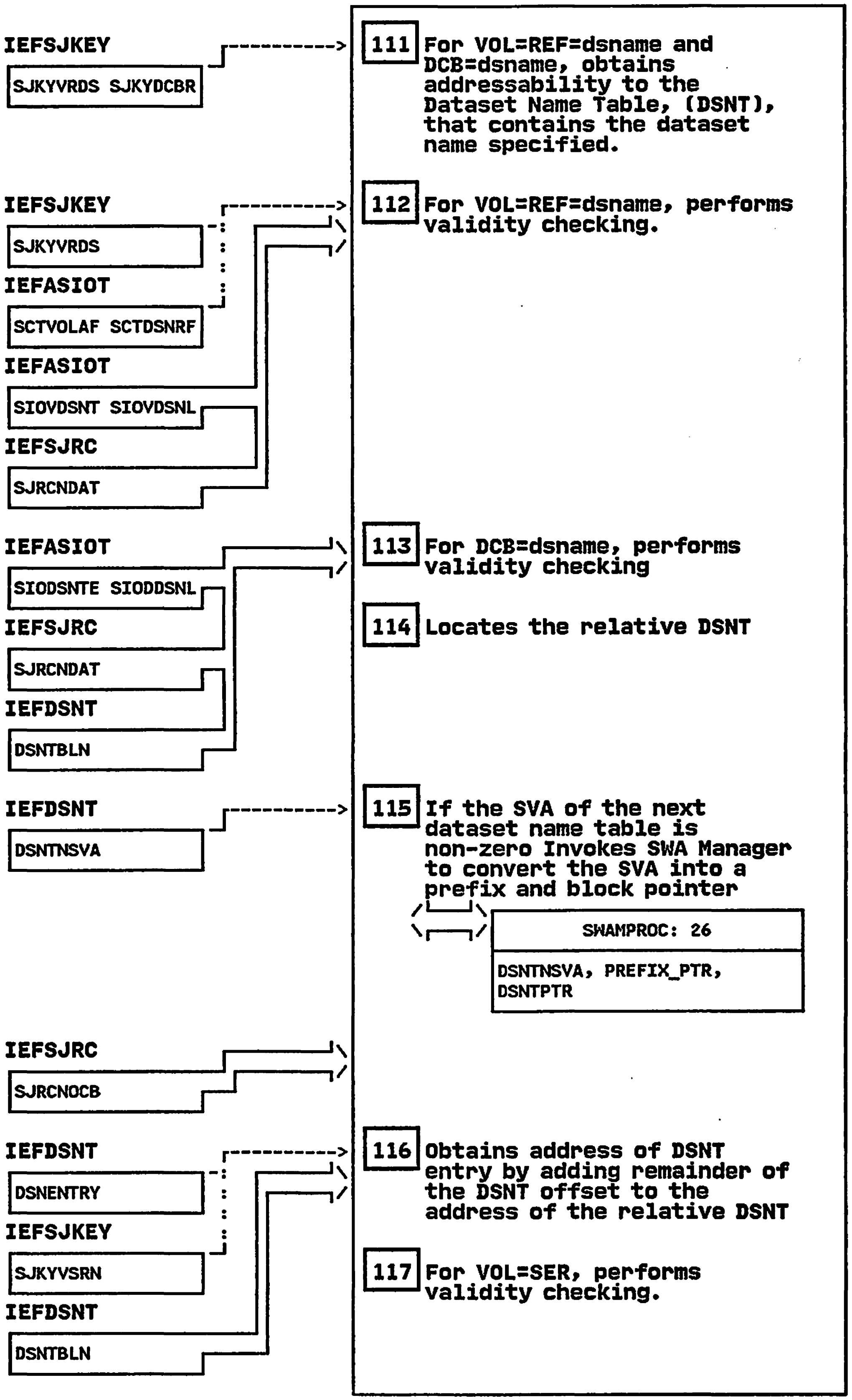
108 SPECIAL
108 SPECIAL Key Routine

IEFSJKEY
SJKYVRDD

109 For VOL=REF=DDNAME, obtains SVA of the referenced DD and stores in first word of an 8 byte field provided by the caller to create a DD token for the referenced DD

IEFASIOT
SCTVOLAF SCTDSNRF
IEFASIOT
SIOTVRSB
IEFSJRC
SJRCNDAT

110 Validity checks for VOL=REF=DDNAME.



IEFSJKEY
SJKYVRDS SJKYDCBR

111 For VOL=REF=dsname and DCB=dsname, obtains addressability to the Dataset Name Table, (DSNT), that contains the dataset name specified.

IEFSJKEY
SJKYVRDS
IEFASIOT
SCTVOLAF SCTDSNRF
IEFASIOT
SIOVDSNT SIOVDSNL
IEFSJRC
SJRCNDAT

112 For VOL=REF=dsname, performs validity checking.

IEFASIOT
SIODSNTE SIODDSNL
IEFSJRC
SJRCNDAT
IEFDSNT
DSNTBLN

113 For DCB=dsname, performs validity checking

114 Locates the relative DSNT

IEFDSNT
DSNTNSVA

115 If the SVA of the next dataset name table is non-zero Invokes SWA Manager to convert the SVA into a prefix and block pointer

SWAMPROC: 26
DSNTNSVA, PREFIX_PTR, DSNTPTR

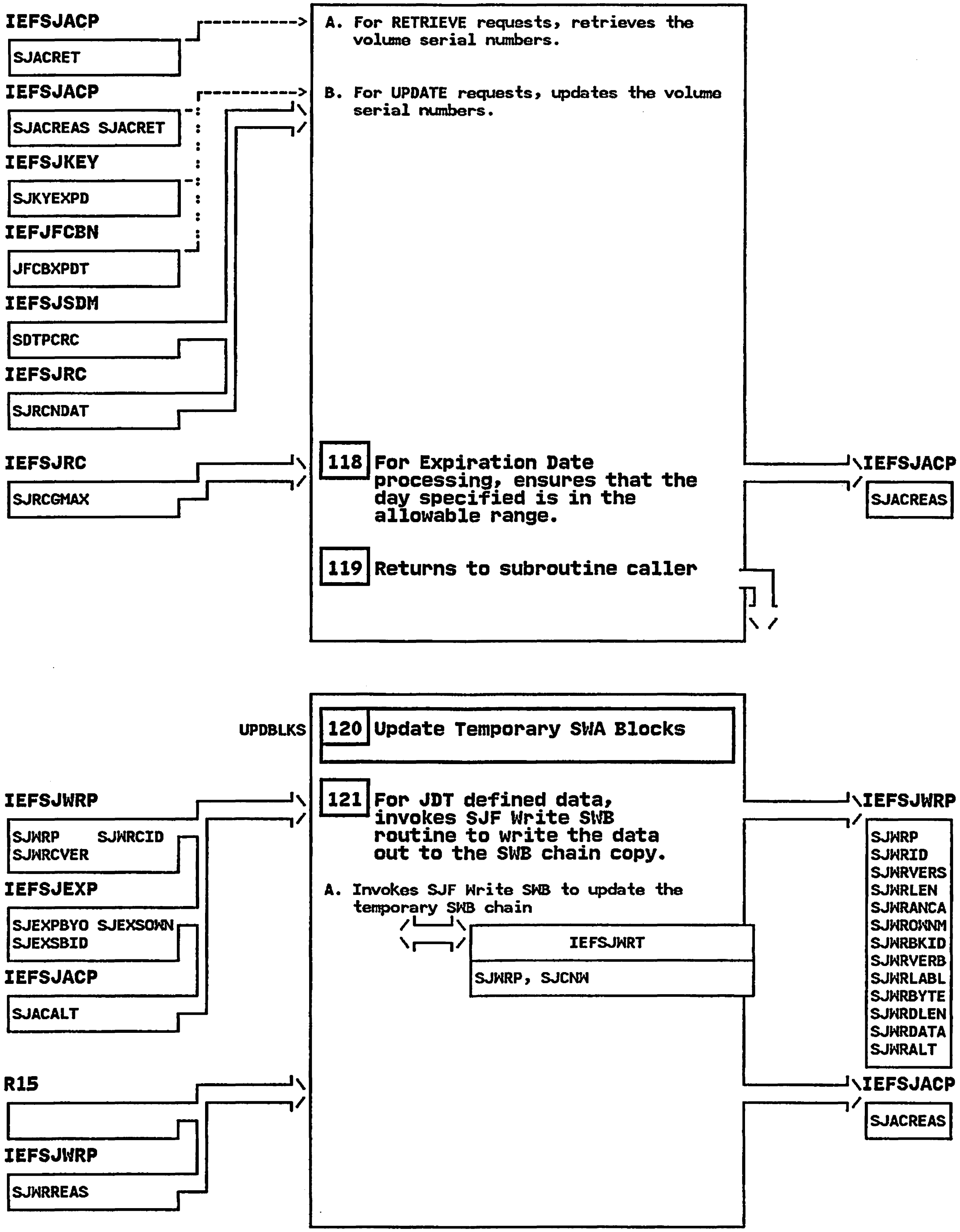
IEFSJRC
SJRCNOCB

IEFDSNT
DSNENTRY
IEFSJKEY
SJKYVSRN
IEFDSNT
DSNTBLN

116 Obtains address of DSNT entry by adding remainder of the DSNT offset to the address of the relative DSNT

117 For VOL=SER, performs validity checking.

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 117A



A. For RETRIEVE requests, retrieves the volume serial numbers.

B. For UPDATE requests, updates the volume serial numbers.

118 For Expiration Date processing, ensures that the day specified is in the allowable range.

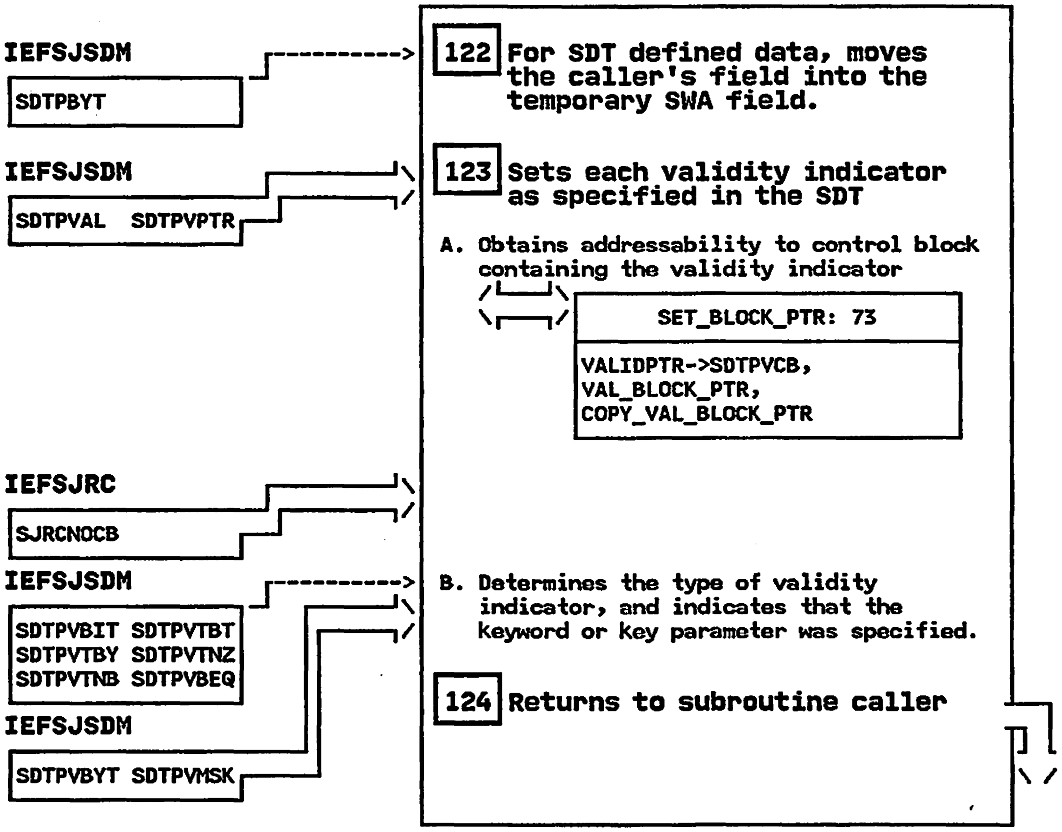
119 Returns to subroutine caller

120 Update Temporary SWA Blocks

121 For JDT defined data, invokes SJF Write SWB routine to write the data out to the SWB chain copy.

A. Invokes SJF Write SWB to update the temporary SWB chain

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine   STEP 122



IEFSJSDM
SDTPBYT

122 For SDT defined data, moves the caller's field into the temporary SWA field.

IEFSJSDM
SDTPVAL SDTPVPTR

123 Sets each validity indicator as specified in the SDT

A. Obtains addressability to control block containing the validity indicator

SET_BLOCK_PTR: 73
VALIDPTR->SDTPVCB,
VAL_BLOCK_PTR,
COPY_VAL_BLOCK_PTR

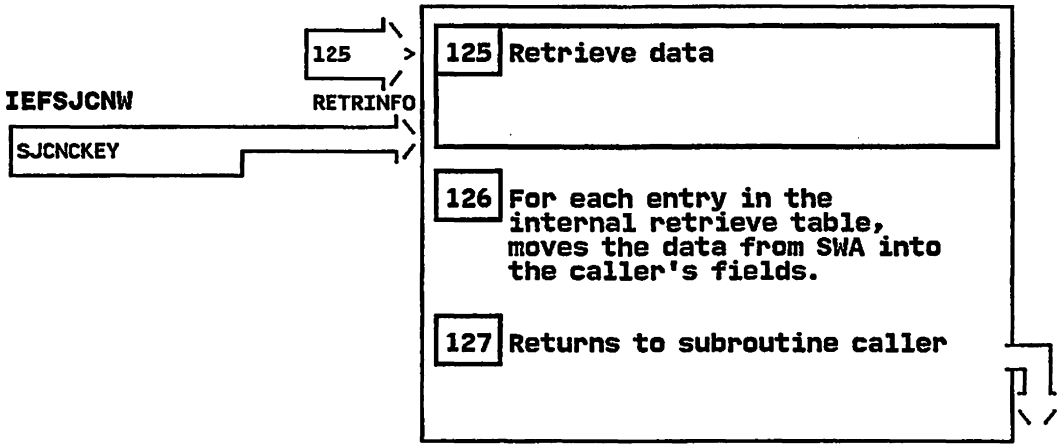
IEFSJRC
SJRCNOCB

IEFSJSDM
SDTPVBIT SDTPVTBT
SDTPVTBY SDTPVTNZ
SDTPVTNB SDTPVBEQ

IEFSJSDM
SDTPVBYT SDTPVMSK

B. Determines the type of validity indicator, and indicates that the keyword or key parameter was specified.

124 Returns to subroutine caller



125
RETRINFO

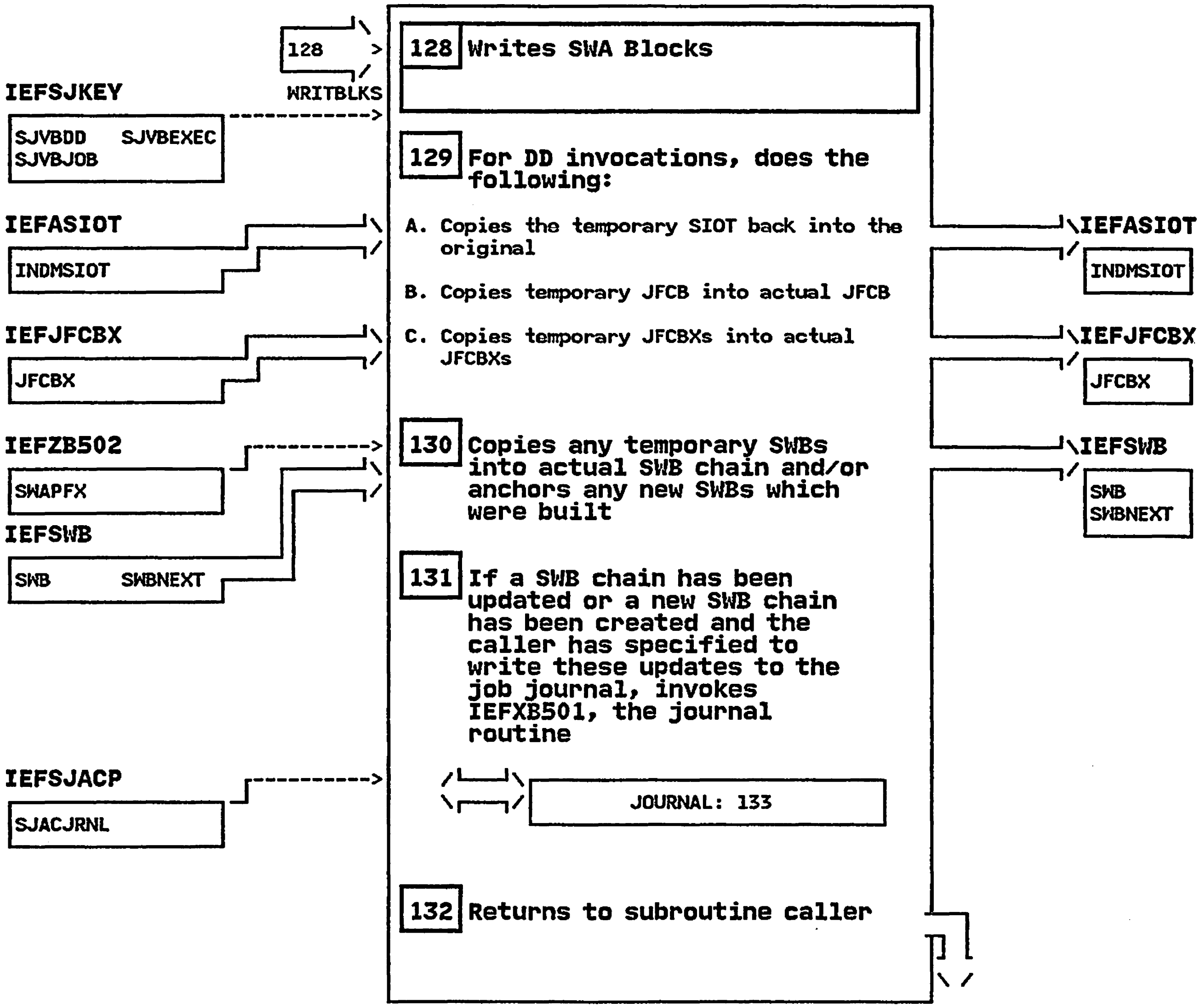
IEFSJCNW
SJCNCKEY

125 Retrieve data

126 For each entry in the internal retrieve table, moves the data from SWA into the caller's fields.

127 Returns to subroutine caller

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 128



128 Writes SWA Blocks

129 For DD invocations, does the following:

A. Copies the temporary SIOT back into the original

B. Copies temporary JFCB into actual JFCB

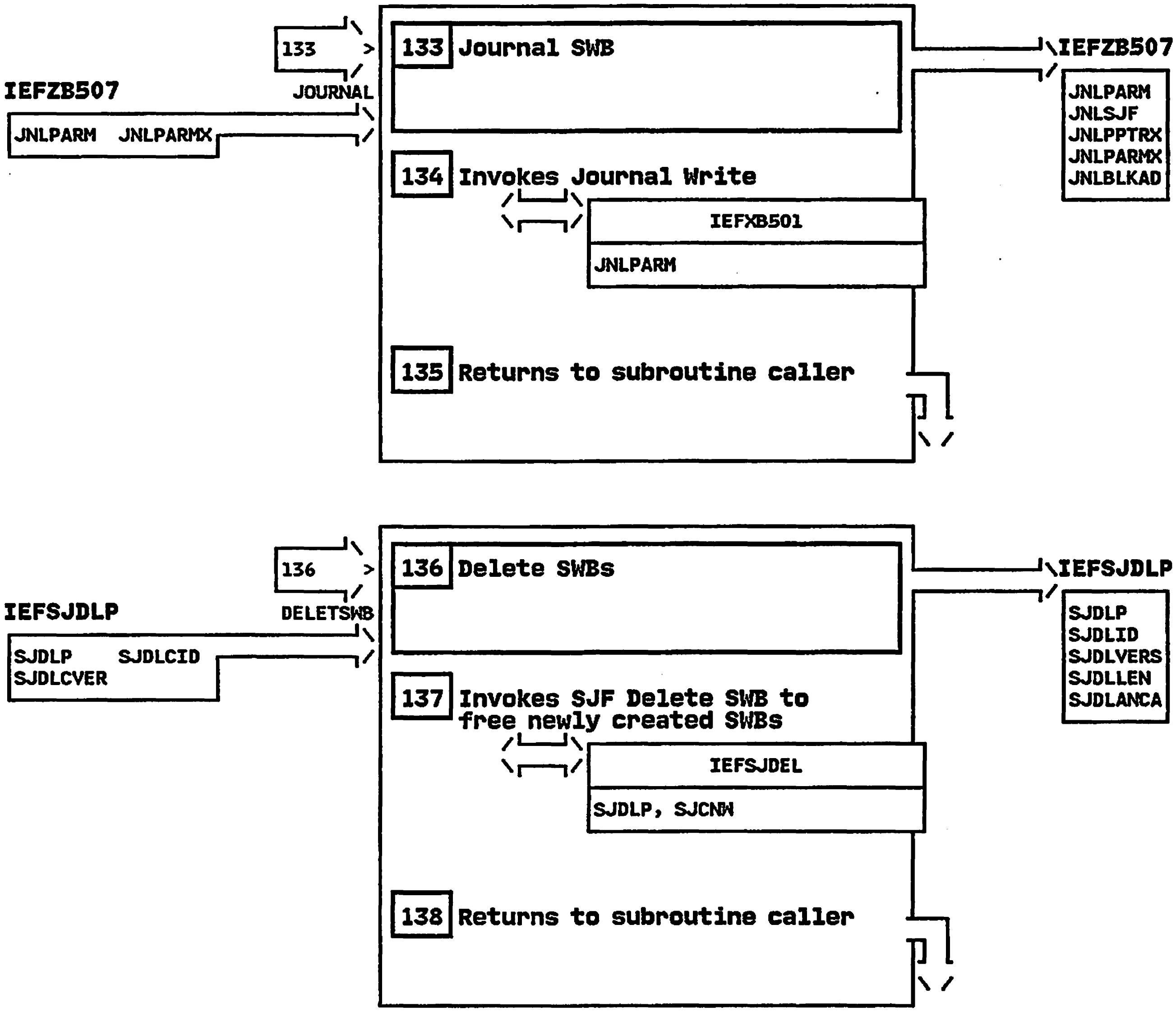
C. Copies temporary JFCBXs into actual JFCBXs

130 Copies any temporary SWBs into actual SWB chain and/or anchors any new SWBs which were built

131 If a SWB chain has been updated or a new SWB chain has been created and the caller has specified to write these updates to the job journal, invokes IEFXB501, the journal routine

132 Returns to subroutine caller

IEFSJACC - Scheduler JCL Facility (SJF) Access SWA Routine STEP 133



IEFSJBLD - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Build SWB Routine

FUNCTION:

This module builds a SWB for the following requests:

1. If the request is for a SWB that is to be part of a job's control structure (a SWA SWB), invokes the SWA manager to obtain the SWB from a SWA subpool and to initialize the SWA prefix.
2. Otherwise, the request is to build a non-SWA SWB. Issues a GETMAIN for storage from subpool 230 and initializes a dummy prefix (formatted like the SWA prefix) in the new SWB.

ENTRY POINT: IEFSJBLD

PURPOSE: See Function

LINKAGE: CALL

CALLERS: Scheduler JCL Facility Write SWB (IEFSJWRT)

INPUT:

SJF Build SWB parameter list, IEFSJBLP:

FIELD	LENGTH/MASK	DESCRIPTION
SJBLP		Parameter list
SJBLID	4	Identifier 'SJBL'
SJBLVERS	1	Version number
SJBLFLAG	1	Control flags
SJBLLEN	2	Length of parameter list
SJBLSTOR	4	Local storage pointer
SJBLREAS	4	Reason code (returned)
SJBLSWBI		Data to identify SWB
SJBLOWNM	8	Owner name
SJBLBKID	2	Block ID
SJBLRSV1	2	Reserved
SJBLCHNI		Data to identify SWB chain
SJBLVERB	8	Verb
SJBLLABL	8	Label
SJBLFLG2	1	Flag byte
SJBLNSWA	X'80'	Build is for a non-SWA SWB
SJBLDYNS	X'40'	Dynamically created SWB
SJBLRSV2	3	Reserved
SJBLALT	4	Address of alternate SWA manager
SJBLNSWB	4	Prefix address of the new SWB (returned)
SJBLNSVA	4	SVA or address of the assigned SWB (returned)
SJBLSTMT	4	JCL statement number

The input to this module also includes the Scheduler JCL Facility control workarea (IEFSJCWA).

OUTPUT:

SJF Build SWB parameter list, IEFSJBLP:

SJBLNSWB = The (SWA prefix) address of the new SWB
SJBLNSVA = The SVA of the new SWB, or the address
of the non-SWA SWB
SJBLREAS = reason code

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

IEFSJBLD - MODULE DESCRIPTION (Continued)

ENTRY POINT: BLDRETRY

PURPOSE:
Performs clean up processing when an ABEND occurs during SJF Build's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES:
SWA Manager Locate Mode (IEFQB556)
Alternate SWA Manager Routine

DATA AREAS:

IEFSJBLP	- SJF Build SWB Parameter List
IEFSJCNW	- SJF Control Workarea
IEFSJRC	- SJF Reason Codes
IEFZB502	- SWA Prefix
IEFZB505	- Extended External Parameter Area

CONTROL BLOCKS:

CVT	- Communications Vector Table
JESCT	- JES Communication Table
PSA	- Prefix Save Area
SWB	- Scheduler Work Block

SERIALIZATION:

Obtains the local lock during a branch entry GETMAIN/FREEMAIN for a non-SWA SWB.

IEFSJBLD - MODULE OPERATION

This module receives control when a SWB needs to be built and performs the following functions:

1. If the request is for a non-SWA SWB:

   - Issues a GETMAIN for storage from subpool 230 for the SWB (including the dummy SWA prefix).

   - If the GETMAIN failed, sets register 15 to 4, sets a reason code of SJRCGETS (1100) in SJBLREAS, and returns.

   - If the GETMAIN was successful, initializes the dummy SWA prefix of the new SWB and sets the non-SWA indicator (SWBNSWA) in the SWB prefix.

2. Otherwise, the request is for a SWA SWB:

   - Builds the extended external parameter area (extended EPA mapped by IEFZB505) for the SWA Manager Assign/Conditional function. The extended EPA contains the length of the block to be obtained (192 bytes, excluding the length of the SWA prefix) and the block id of the SWB.

   - If an Alternate SWA Manager was not specified (SJBLALT is 0), then invokes the SWA Manager Assign/Conditional function to obtain the SWB.
     . If the assign was successful, invokes SWA Manager Write/Locate to initialize the SWA prefix of the new SWB.
     . If the write was successful, invokes SWA Manager Locate/All to get a pointer to the new SWA SWB's prefix.

   - If an alternate SWA Manager was specified (SJBLALT not 0), then invokes the Alternate SWA Manager Assign/Conditional function to obtain the SWB.
     . If the assign was successful, invokes Alternate SWA Manager Write/Locate to initialize the SWA prefix of the new SWB.
     . If the write was successful, invokes Alternate SWA Manager Locate/All to get a pointer to the new SWA SWB's prefix.

3. Initializes the SWB with data from the input parameter list: SJBLSWBI (owner name and block ID), SJBLCHNI (verb and label), SJBLSTMT (statement number), and SJBLDYNS (Dynamic SWB indicator)

4. Stores the address of the SWA prefix for the new SWB in SJBLNSWB and the SVA of the new SWB in SJBLNSVA of the parameter list.

5. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery will receive control from RTM. The recovery routine specifies the retry address in the SJF workarea (BLDRETRY) to RTM. When the retry segment (BLDRETRY) receives control from RTM, it does the following:

1. If a SWB has been obtained, then frees the SWB:
   - If the SWB is a SWA SWB, then calls SWA

IEFSJBLD - MODULE OPERATION (Continued)

   Manager to free the SWB.
   - If the SWB is a non-SWA SWB, then issues a FREEMAIN for the SWB.
2. Sets the return code to indicate an SJF system error.
3. Returns to the caller.

IEFSJBLD - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJBLD
BLDRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJBLD:

EXIT NORMAL:

Register 15 = 0 - Processing successful

Reason codes in SJBLREAS:
SJRCNOER (0) - Processing successful

EXIT ERROR:

Register 15 = 4 - Request not processed

Reason codes in SJBLREAS:
SJRCGETS (1100), GETMAIN for a SWB failed, or some other SWA Manager error

NOTE: An 0B0 ABEND occurs for all errors in the SWA manager except when a request is made for a conditional GETMAIN.

ENTRY POINT BLDRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJBLD:

Register 0	= Undefined
Register 1	= Address of a two word parameter list. The first word contains the address of the build SWB parameter list (IEFSJBLP), and the second word contains the address of the SJF control workarea (IEFSJCNW)
Registers 2-12	= Undefined
Register 13	= Address of 18 word savearea
Register 14	= Return address
Register 15	= Entry point address

ENTRY POINT BLDRETRY:

Register 1	= Address of ESTAE parameter list
Registers 0,2-14	= Undefined
Register 15	= Entry point address

IEFSJBLD - DIAGNOSTIC AIDS (Continued)

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJBLD:

```
Registers 0-14 = Restored
Register    15 = Return code
```

ENTRY POINT BLDRETRY:

```
Registers 0-14 = Restored
Register  15   = Return Code
```

IEFSJCNL - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Control Routine

FUNCTION:
This module performs common initial processing for the SJF functions, routes the request to the specified SJF function, and upon return from the specified function, performs common cleanup processing.

ENTRY POINT: IEFSJCNL

PURPOSE: See Function

LINKAGE: BSM

CALLERS: SJF router routine (IEFSJRTE)

INPUT:
There is a different input parameter list for each SJF function. The first 16 bytes of each parameter list contains the information mapped by the SJF control parameter list (IEFSJCNP).

FIELD	LENGTH/MASK	DESCRIPTION
SJCNP	16	Control parameter list
SJCNID	4	Identifier for requested SJF function
SJCNVERS	1	Version number
SJCNFLAG	1	Control flags
SJCNNREC	X'80'	No recovery
SJCNNOCU	X'40'	No cleanup
SJCNUNAU	X'20'	Unauthorized caller
SJCNLEN	2	Length
SJCNSTOR	4	Local storage pointer or zero
SJCNREAS	4	Reason code

OUTPUT:
SJF control parameter list (IEFSJCNP)

FIELD	LENGTH/MASK	DESCRIPTION
SJCNSTOR	4	Local storage pointer or zero
SJCNREAS	4	Reason code

EXIT NORMAL: Return to the issuer of SJFREQ macro

EXIT ERROR: Return to the issuer of SJFREQ macro

ENTRY POINT: RECOVERY

PURPOSE:
To recover from an error that caused the exit to RTM.

LINKAGE: SYNCH

CALLERS: RTM

INPUT:
Estae parameters
System diagnostic work area (SDWA)

OUTPUT: SVC dump and a record written in LOGREC data set.

EXIT NORMAL:
Return to RTM specifying the retry address stored in the SJF control

IEFSJCNL - MODULE DESCRIPTION (Continued)

workarea.

EXIT ERROR:
Percolate to the caller's recovery routine if the abend did not occur while SJF was processing or a previous ABEND occurred.

ENTRY POINT: RECCLEAN

PURPOSE:
To perform cleanup processing when an abend occurred during the SJF control routine's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: Estae parameters

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to the issuer of SJFREQ macro

EXTERNAL REFERENCES:

ROUTINES:

IEFSJACC	- SJF Access Function
IEFSJDEF	- SJF Define JDVT
IEFSJDEL	- SJF Delete SWB
IEFSJERS	- SJF Erase
IEFSJEXT	- SJF JDT Extract
IEFSJFND	- SJF Find SWB
IEFSJGET	- SJF Get SWB
IEFSJINT	- SJF JDVT Initialization
IEFSJJDV	- SJF Find JDVT
IEFSJPUT	- SJF Put SWB
IEFSJRET	- SJF Retrieve
IEFSJUPD	- SJF Update
IEFSJVER	- SJF Verify
IEFSJWRT	- SJF Write SWB

DATA AREAS:

IEFSJCNP	- SJF Control Parameter List
IEFSJCNW	- SJF Control Work Area
IEFSJRC	- SJF Reason Codes

CONTROL BLOCKS:

CVT	- Communications Vector Table
JESCT	- Job Entry Subsystem Communications Table
SDWA	- System Diagnostic Work Area

IEFSJCNL - MODULE OPERATION

This module performs common initial processing for the SJF functions, routes the request to the specified SJF function, and upon return from the specified function, performs common cleanup processing. It does the following:

1. Checks the addressability of the input parameter list, validates the parameter list length and version number, verifies that the parameter list identifier matches the requested function, and verifies that if the caller is unauthorized (SJCNUNAU = on), the function is a Verify or Terminate request.

2. If the caller is an authorized caller, issues a MODESET to change to key 1. For unauthorized callers SJF executes in the key of the caller.

3. If this is the first invocation of SJF (SJCNSTOR=0), obtains the local storage for SJF.

4. Copies the caller's parameter list into the SJF local storage.

5. If this is the first invocation of SJF (SJCNSTOR=0) and the caller requests recovery (SJCNNREC='0'B), establishes a recovery environment.

6. If this request is not to terminate the scheduler JCL facility (SJF) (register 0 not 0), invokes the requested SJF function.

7. Copies the parameter list in the SJF local storage into the caller's parameter list.

8. If cleanup processing was requested by the caller (SJCNNOCU='0'B), cancels the recovery environment (if established) and frees the local storage that was obtained.

9. If the caller is an authorized caller, issues a MODESET to change back to the key of the caller.

10. Returns to the issuer of SJFREQ macro.

RECOVERY OPERATION:

The recovery segment (RECOVERY) provides recovery for all SJF function routines and the SJF control routine.

If the error occurred in a SJF function routine or the SJF control routine:

1. Stores diagnostic information in the system diagnostic work area (SDWA).

2. Writes an entry in the LOGREC dataset and if the caller is authorized, takes an SVC dump.

3. Specifies the retry address stored in the SJF control workarea to RTM.

4. Returns to RTM.

If the error did not occur in a SJF routine or a previous ABEND occurred:

1. Frees the local storage that was obtained.

2. Specifies that RTM percolate to the caller's

IEFSJCNL - MODULE OPERATION (Continued)

recovery routine.

3. Returns to RTM.

If the error occurred in the SJF control routine, the retry segment (RECCLEAN) in the SJF control routine receives control from RTM and does the following:

1. Sets the return code to indicate an SJF system error.

2. Copies the parameter list in the SJF local storage into the caller's parameter list.

3. Cancels the recovery environment and frees the local storage that was obtained.

4. If the caller is an authorized caller, issues a MODESET to change back to the key of the caller.

5. Returns to the issuer of the SJFREQ macro.

IEFSJCNL - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJCNL
RECOVERY
RECCLEAN

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJCNL:

EXIT NORMAL:

```
Register 15 = 0 - Request completed successfully.
                  (Return code 0 returned from the
                  requested function.)
```

EXIT ERROR:

```
Register 15 =  4 - Did not process request.
                   (Return code 4 returned from the
                   requested function. See the reason
                   code in the parameter list.)
Register 15 =  8 - Parameter list invalid
Register 15 = 12 - GETMAIN for local storage failed
Register 15 = 16 - ESTAE could not be established
Register 15 = 20 - SJF system error
```

ENTRY POINT RECOVERY:

EXIT NORMAL:

```
Register 15 = 4 - Retry to mainline cleanup processing
```

EXIT ERROR:

```
Register 15 = 0 - Do not retry
```

ENTRY POINT RECCLEAN:

EXIT ERROR:

```
Register 15 = 20 - SJF system error
```

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJCNL:

```
Register  0     = Requested function mask
Register  1     = Address of a word that contains
                  the address of the input
                  parameter list
Registers 2-12  = Undefined
Register  13    = Address of 18-word save area
Register  14    = Return address
Register  15    = Entry point address
```

ENTRY POINT RECOVERY:

IEFSJCNL - DIAGNOSTIC AIDS (Continued)

Register 0 = Indicates whether a SDWA was obtained
Register 1 = Pointer to the SDWA if a SDWA was obtained
Register 2 = Pointer to the ESTAE parameter list if a SDWA was not obtained
Registers 3-13 = Undefined
Register 14 = Return address to RTM
Register 15 = Entry point address

ENTRY POINT RECCLEAN:

Register 0 = Undefined
Register 1 = Address of the ESTAE parameter list
Registers 2-14 = Undefined
Register 15 = Entry point address

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJCNL:

Register 0 = Requested function mask
Register 1 = Address of a word that contains the address of the input parameter list
Registers 2-12 = Restored
Register 13 = Address of 18-word save area
Register 14 = Return address
Register 15 = Entry point address

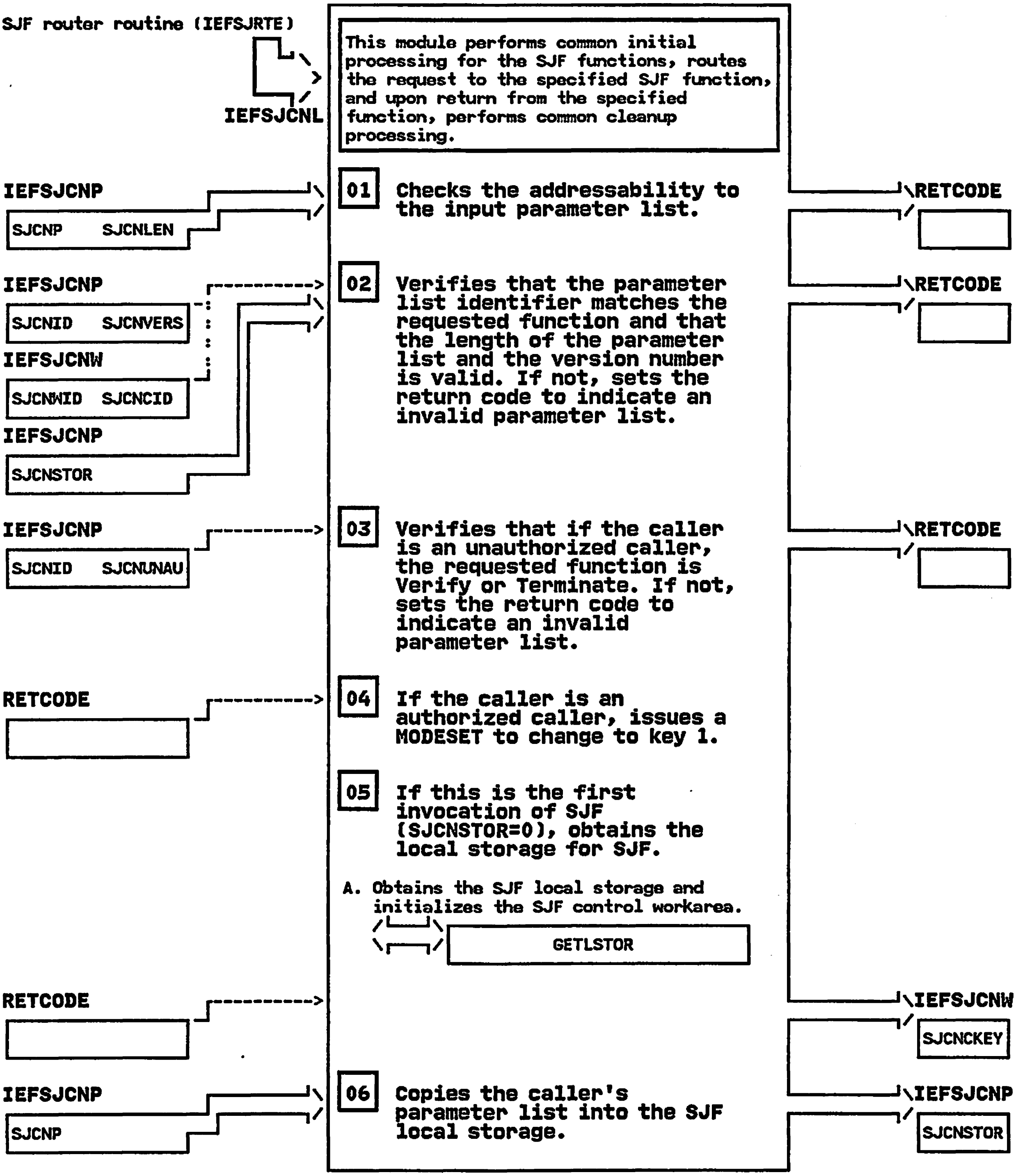
ENTRY POINT RECOVERY:

Registers 0-13 = Undefined
Register 14 = Return address
Register 15 = Retry address

ENTRY POINT RECCLEAN:

Register 0 = Requested function mask
Register 1 = Address of a word that contains the address of the input parameter list
Registers 2-12 = Undefined
Register 13 = Address of 18-word save area
Register 14 = Return address
Register 15 = Entry point address

IEFSJCNL - Scheduler JCL Facility (SJF) Control Routine STEP 01



This module performs common initial processing for the SJF functions, routes the request to the specified SJF function, and upon return from the specified function, performs common cleanup processing.

01 Checks the addressability to the input parameter list.

02 Verifies that the parameter list identifier matches the requested function and that the length of the parameter list and the version number is valid. If not, sets the return code to indicate an invalid parameter list.

03 Verifies that if the caller is an unauthorized caller, the requested function is Verify or Terminate. If not, sets the return code to indicate an invalid parameter list.

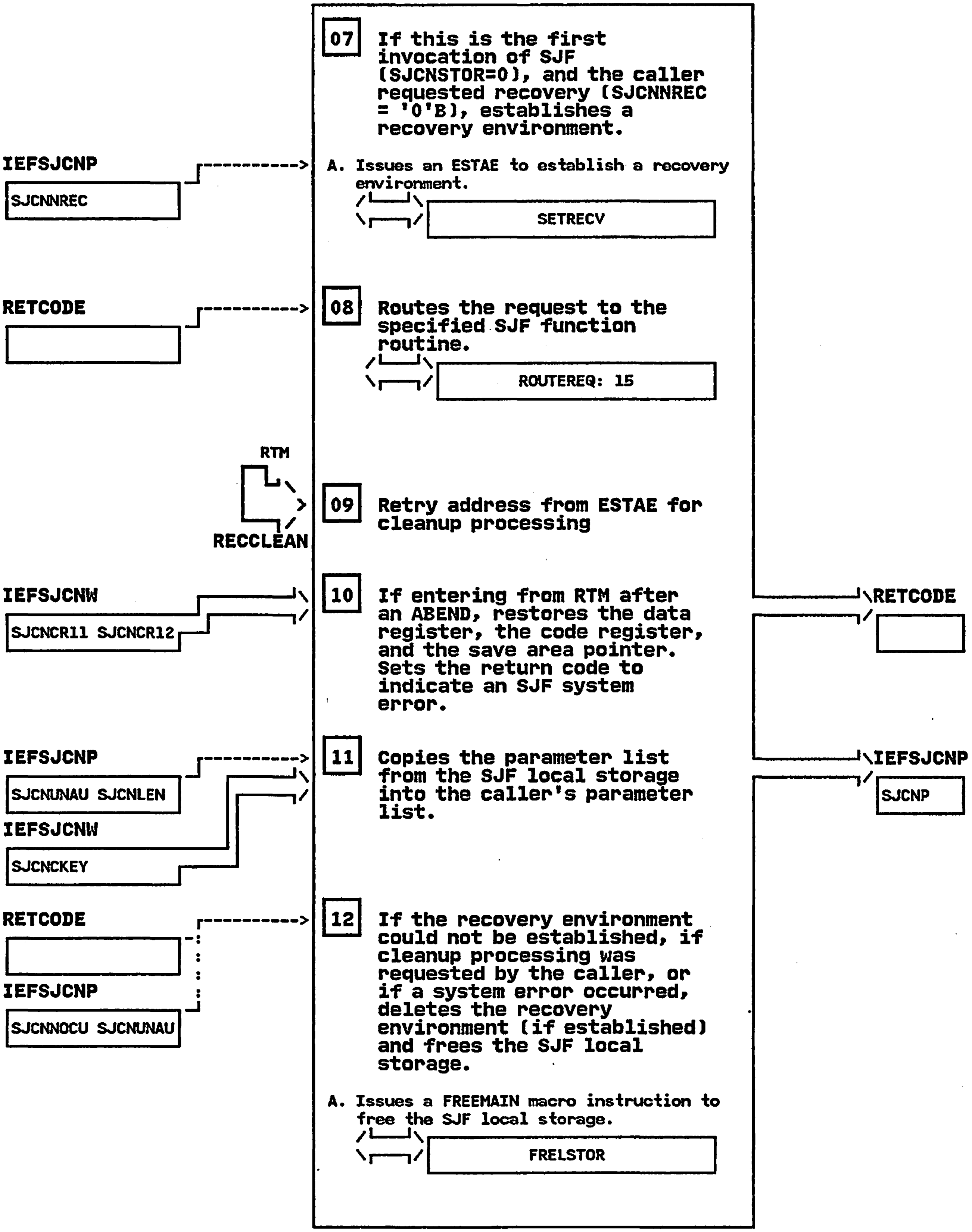
04 If the caller is an authorized caller, issues a MODESET to change to key 1.

05 If this is the first invocation of SJF (SJCNSTOR=0), obtains the local storage for SJF.

A. Obtains the SJF local storage and initializes the SJF control workarea.

06 Copies the caller's parameter list into the SJF local storage.

IEFSJCNL - Scheduler JCL Facility (SJF) Control Routine STEP 07



IEFSJCNP: SJCNNREC

07 If this is the first invocation of SJF (SJCNSTOR=0), and the caller requested recovery (SJCNNREC = '0'B), establishes a recovery environment.

A. Issues an ESTAE to establish a recovery environment.

SETRECV

RETCODE

08 Routes the request to the specified SJF function routine.

ROUTEREQ: 15

RTM

RECCLEAN

09 Retry address from ESTAE for cleanup processing

IEFSJCNW: SJCNCR11 SJCNCR12

10 If entering from RTM after an ABEND, restores the data register, the code register, and the save area pointer. Sets the return code to indicate an SJF system error.

RETCODE

IEFSJCNP: SJCNUNAU SJCNLEN

IEFSJCNW: SJCNCKEY

11 Copies the parameter list from the SJF local storage into the caller's parameter list.

IEFSJCNP: SJCNP

RETCODE

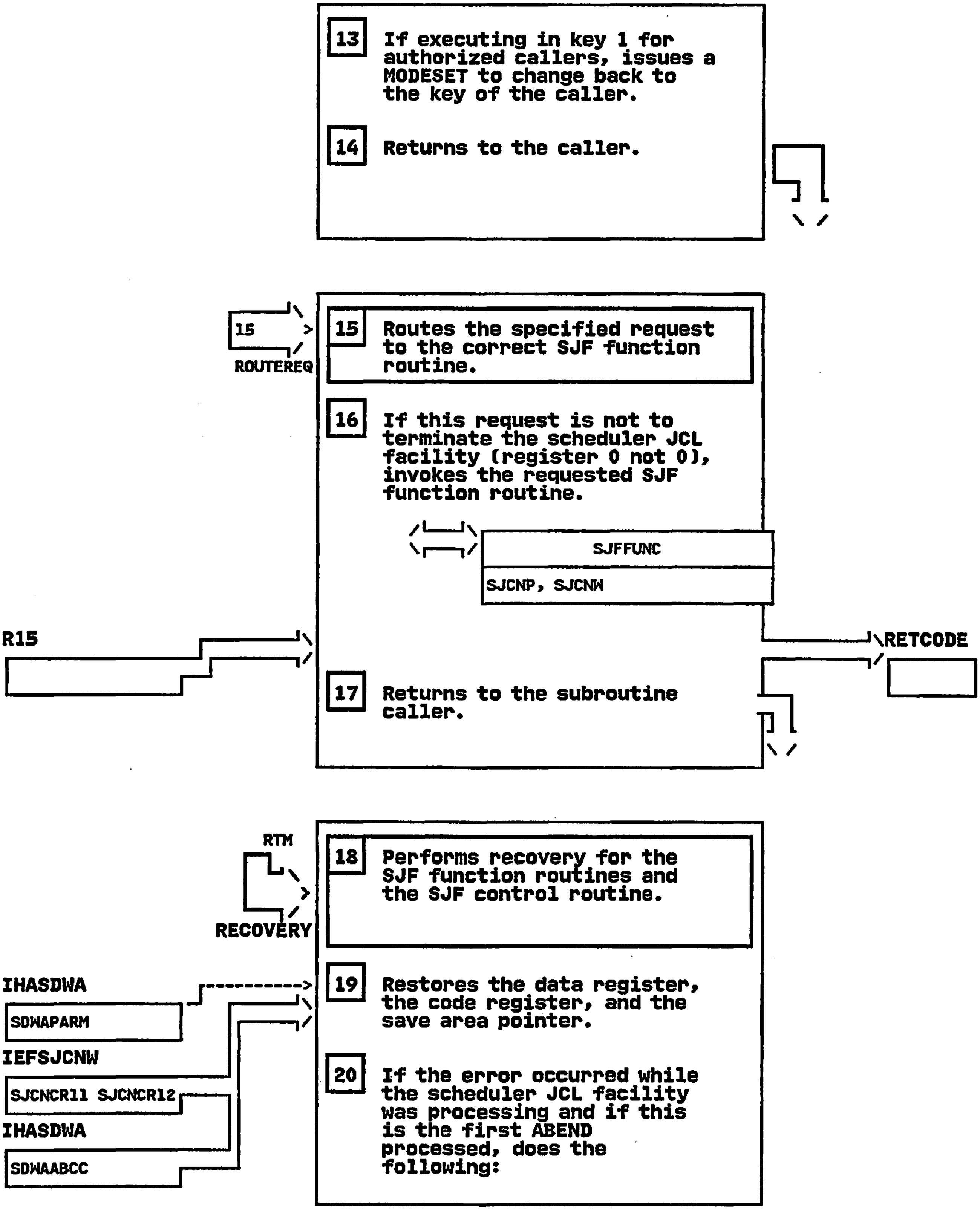
IEFSJCNP: SJCNNOCU SJCNUNAU

12 If the recovery environment could not be established, if cleanup processing was requested by the caller, or if a system error occurred, deletes the recovery environment (if established) and frees the SJF local storage.

A. Issues a FREEMAIN macro instruction to free the SJF local storage.

FRELSTOR

IEFSJCNL - Scheduler JCL Facility (SJF) Control Routine



13 If executing in key 1 for authorized callers, issues a MODESET to change back to the key of the caller.

14 Returns to the caller.

15 Routes the specified request to the correct SJF function routine.

16 If this request is not to terminate the scheduler JCL facility (register 0 not 0), invokes the requested SJF function routine.

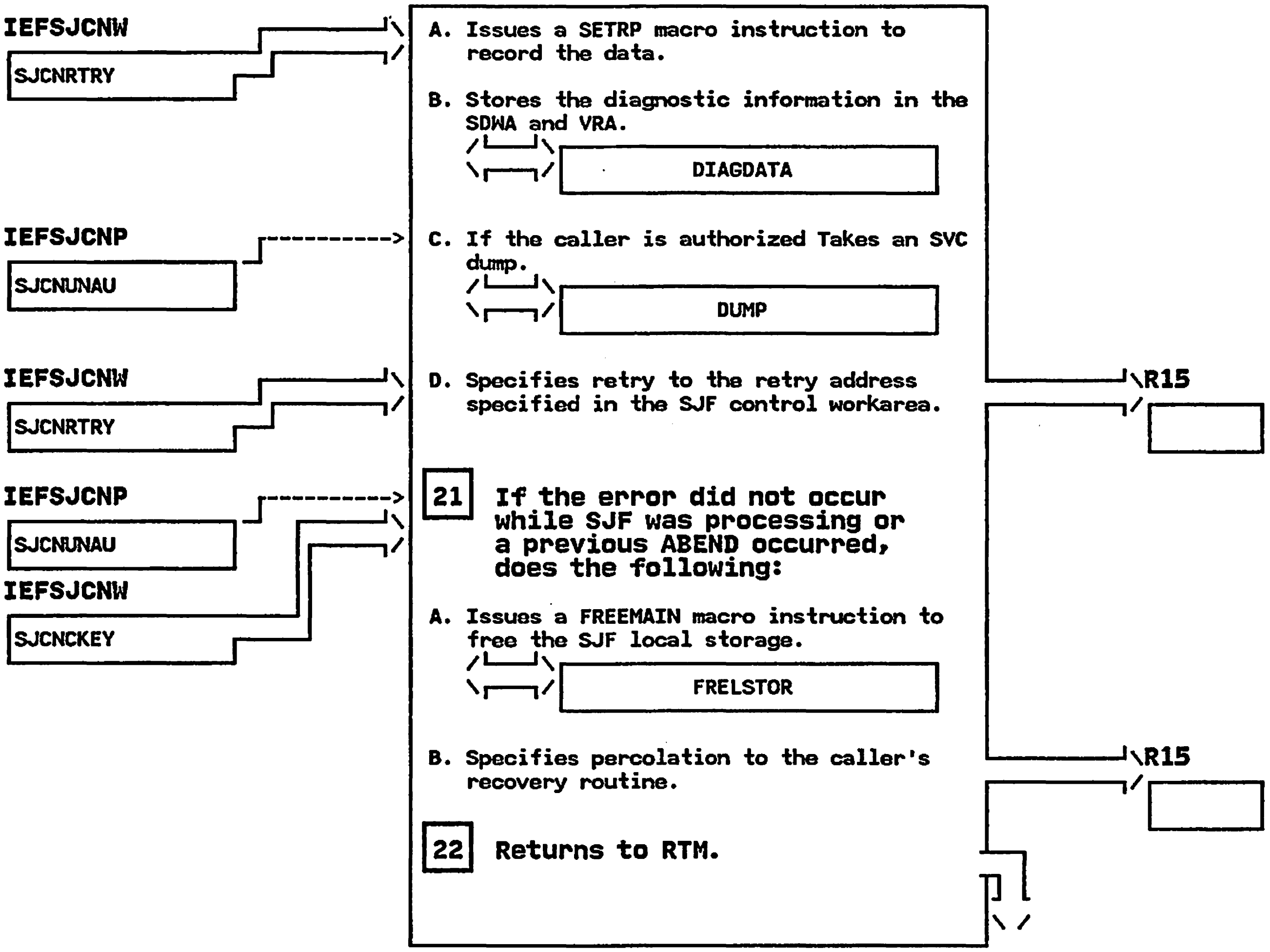
17 Returns to the subroutine caller.

18 Performs recovery for the SJF function routines and the SJF control routine.

19 Restores the data register, the code register, and the save area pointer.

20 If the error occurred while the scheduler JCL facility was processing and if this is the first ABEND processed, does the following:

IEFSJCNL - Scheduler JCL Facility (SJF) Control Routine STEP 20A



A. Issues a SETRP macro instruction to record the data.

B. Stores the diagnostic information in the SDWA and VRA.

C. If the caller is authorized Takes an SVC dump.

D. Specifies retry to the retry address specified in the SJF control workarea.

21 If the error did not occur while SJF was processing or a previous ABEND occurred, does the following:

A. Issues a FREEMAIN macro instruction to free the SJF local storage.

B. Specifies percolation to the caller's recovery routine.

22 Returns to RTM.

IEFSJDEF - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility Define JDVT Routine

FUNCTION:

This module creates a JCL definition vector table (JDVT) and adds this table to the JDVT chain anchored off the JES control table (JESCT).

ENTRY POINT: IEFSJDEF

PURPOSE: See Function

LINKAGE: CALL

CALLERS:
SJF control routine (IEFSJCNL)
SJF JDVT initialization (IEFSJINT)

INPUT:
SJF Define JDVT parameter list IEFSJDFP

FIELD	LENGTH/MASK	DESCRIPTION
SJDFP	Variable	Parameter list
SJDFID	4	Identifier 'SJDF'
SJDFVERS	1	Version number
SJDFFLAG	1	Function flags
SJDFNREC	x'80'	No recovery
SJDFNOCU	x'40'	No cleanup
SJDFLEN	2	Length of parameter list
SJDFSTOR	4	Local storage pointer
SJDFREAS	4	Reason code (returned)
SJDFJDVT	8	JDVT name
SJDFFUNC	1	Flag byte
SJDFDFLT	x'80'	This JDVT is the system default
SJDFRSV1	1	Reserved
SJDF#JDT	2	Number of JDTs for this JDVT
SJDFJLNF	2	Number of the JDT that failed to LOAD (returned)
SJDFSDTN	8	Statement Definition Table (SDT) name
SJDFJDTN(*)	8	JDT names, number of occurrences dependent on SJDF#JDT

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:
Define JDVT parameter list (IEFSJDFP)

FIELD	LENGTH/MASK	DESCRIPTION
SJDFREAS	4	Reason code (returned)

EXIT NORMAL: Return to caller.

EXIT ERROR: Return to caller.

ENTRY POINT: DEFRETRY

PURPOSE:
Performs clean up processing when an ABEND occurs during SJF define JDVT processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

IEFSJDEF - MODULE DESCRIPTION (Continued)

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller.

EXTERNAL REFERENCES:

ROUTINES: None

DATA AREAS:

IEFSJDFP	- SJF Define JDVT Parameter List
IEFSJCNW	- SJF Control Work Area
IEFSJHTP	- SJF Hash Table Build Parameter List
IEFSJRC	- SJF Common Reason Codes

CONTROL BLOCKS:

CVT	- Communications Vector Table
JDVT	- JCL Definition Vector Table
JESCT	- JES Control Table

IEFSJDEF - MODULE OPERATION

Entry point IEFSJDEF creates a JCL definition vector table (JDVT) and adds this table to the JDVT chain anchored off the JES control table. IEFSJDEF does the following:

1. If a JDVT already exists with the specified JDVT name (SJDFJDVT), sets the return code and reason code to indicate that a duplicate JDVT name was found. Sets an error switch to end processing.

2. If this JDVT is to be the system default JDVT (SJDFDFLT=ON), checks if there already is a default JDVT. If so, sets the return code and reason code to indicate that a default JDVT already exists. Sets an error switch to end processing.

3. Obtains storage for the JDVT. If no storage is available, sets the reason code to indicate that storage was not available for the JDVT. Sets an error switch to end processing.

4. Initializes the JDVT with the control block identifier ('JDVT'), length, macro version number, and JDVT name.

5. For each JDT name specified in the parameter list, issues a LOAD to get the address of the JDT, stores the name and address in the JDVT, and issues a DELETE. If a specified JDT is not found, sets the reason code to indicate that a JDT was not found. Sets an error switch to end processing.

6. Issues a LOAD to get the address of the Statement Definition Table (SDT), and stores the address of the SDT in the JDVT, and issues a DELETE. If the SDT was not found, then sets the reason code to indicate the SDT was not found. Sets an error switch to end processing.

7. If no errors occur, invokes IEFSJHTB to build the hash table structure, and adds the JDVT to the chain that is anchored off the JESCT.

8. If there are no JDVTs on the chain, issues a MODESET to change to key zero and chains the JDVT to the JESCT. Issues a MODESET to change back to key one.

9. Returns to the caller.

RECOVERY OPERATION:

If an ABEND occurs in this module, the scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (DEFRETRY) in the SJF control work area. When DEFRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.
2. If storage had already been obtained, then frees it.
3. Returns to the caller.

IEFSJDEF - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJDEF
DEFRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJDEF:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully
Reason code in SJDFREAS:
SJRCNOER (0) - Define successful

EXIT ERROR:

Register 15 = 4 - Request was not processed.
Reason code in SJDFREAS (in decimal):
SJRCNJDT (300) - JDT not found
SJRCDUPJ (301) - Duplicate JDVT name
SJRCDFTJ (302) - Default JDVT already exists
SJRCGETJ (303) - GETMAIN for JDVT failed
SJRCSUHT (304) - Storage unavailable to build hash table structure
SJRCNSDT (306) - SDT not found

ENTRY POINT DEFRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJDEF:

Register 0 = Undefined
Register 1 = Address of two words that contain the address of the input parameter list (IEFSJDFP) and the address of the control work area (IEFSJCNW).
Registers 2-12 = Undefined
Register 13 = Address of 18-word save area
Register 14 = Return address
Register 15 = Entry point address

ENTRY POINT DEFRETRY:

Register 0 = Undefined
Register 1 = Address of the ESTAE parameter list
Registers 2-14 = Undefined
Register 15 = Entry point address

REGISTER CONTENTS ON EXIT:

IEFSJDEF - DIAGNOSTIC AIDS (Continued)

ENTRY POINT IEFSJDEF:

```
Registers 0-14 = Restored
Register    15 = Return code
```

ENTRY POINT DEFRETRY:

```
Registers 0-14 = Restored
Register    15 = Return code
```

IEFSJDEL - MODULE DESCRIPTION

**DESCRIPTIVE NAME: Scheduler JCL Facility
Delete SWB Chain Routine**

FUNCTION:
This module deletes a scheduler work block (SWB) chain.

ENTRY POINT: IEFSJDEL

PURPOSE: See Function

LINKAGE: CALL

CALLERS:
Scheduler JCL facility (SJF) control routine (IEFSJCNL)
Scheduler JCL facility (SJF) update routine (IEFSJUPD)

INPUT:
SJF Delete SWB parameter list, IEFSJDLP:

FIELD	LENGTH/MASK	DESCRIPTION
SJDLP	32	Parameter list
SJDLID	4	Identifier 'SJDL'
SJDLVERS	1	Version number
SJDLFLAG	1	Function flags
SJDLNREC	x'80'	No recovery
SJDLNOCU	x'40'	No cleanup
SJDLLEN	2	Length of parameter list
SJDLSTOR	4	Local storage pointer
SJDLREAS	4	Reason code (returned)
SJDLRSV1	4	Reserved
SJDLTOKN	8	SWB chain token
SJDLANBK	4	Address of the anchor control block or of the first control block for a JCL statement
SJDLANCA	4	Address of a word pointing to a SWB chain or zero
SJDLFUNC	1	Delete Function byte
SJDLLDEL	x'80'	Logically delete indicator
SJDLRSV2	3	Reserved

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:
Data returned in the input parameter list:

FIELD	LENGTH/MASK	DESCRIPTION
SJDLREAS	4	Reason code

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: DELRETRY

PURPOSE:
Performs clean up processing when an ABEND occurs during SJF delete processing.

LINKAGE: SYNCH

CALLERS: RTM

IEFSJDEL - MODULE DESCRIPTION (Continued)

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES: SWA Manager

DATA AREAS:

IEFZB502	- Scheduler Work Area Prefix
IEFZB505	- SWA Manager Parameter List
IEFSJRC	- SJF Common Reason Codes
IEFSJDLP	- SJF Delete SWB Parameter List
IEFSJCNW	- SJF Control Work Area
IEFSJSWP	- IEFSJSWA parameter list

CONTROL BLOCKS:

CVT	- Communication Vector Table
JCT	- Job Control Table
JCTX	- Job Control Table Extension
JESCT	- JES Communication Table
PSA	- Prefix Save Area
SCT	- Step Control Table
SIOT	- Step Input Output Table
SWB	- Scheduler Work Block

SERIALIZATION:

Holds the local lock during branch entry FREEMAIN of a non-SWA SWB.

IEFSJDEL - MODULE OPERATION

Entry point IEFSJDEL deletes a scheduler work block (SWB) chain. It does the following:

1. Calls the IEFSJSWA routine to translate the SJF token into the address of the first SWB on the chain. Saves the next SWB chain SVA in the anchor word pointed to by the second word of the SWB token. If an invalid token was found, sets the return code and reason code to indicate this.

2. Determines if this call is for a logical SWB chain deletion and if so validates that the SWB chain is dynamically created and not already logically deleted. Marks the SWB chain logically deleted.

3. If this call is for physical deletion then If the SWB prefix indicates that this is a non-SWA SWB (SWBNSWA is on), deletes the SWB from subpool 230. Otherwise, calls SWA manager to delete the SWBs.

4. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery will receive control from RTM. The recovery routine specifies the retry address in the SJF work area (DELRETRY) to RTM. When the retry segment (DELRETRY) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.
2. Returns to the caller.

IEFSJDEL - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJDEL
DELRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJDEL:

EXIT NORMAL:

```
Register 15 = 0 - Request completed successfully.
Reason code in SJDLREAS:
  ZERO (0) - Delete of a SWB chain
```

EXIT ERROR:

```
Register 15 = 4 - Request was not processed.
Reason code in SJDLREAS (decimal):
  SJRCIVTK (0002) - Invalid SWB token
  SJRCALDL (0700) - SWB Chain already logically
                    deleted
  SJRCNDYN (0701) - Only dynamically created SWB
                    chains may be logically deleted
```

ENTRY POINT DELRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJDEL:

```
Register    0 = Undefined
Register    1 = Address of two words that contain the
                address of  the input  parameter list
                (IEFSJDLP)  and  the  address  of the
                control work area (IEFSJCNW).
Register 2-12 = Undefined
Register   13 = Address of 18-word save area
Register   14 = Return address
Register   15 = Entry point address
```

ENTRY POINT DELRETRY:

```
Register     0 = Undefined
Register     1 = Address of the ESTAE parameter list
Registers 2-14 = Undefined
Register    15 = Entry point address
```

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJDEL:

IEFSJDEL - DIAGNOSTIC AIDS (Continued)

```
Registers 0-14 = Restored
Register    15 = Return code
```

ENTRY POINT DELRETRY:

```
Registers 0-14 = Restored
Register    15 = Return code
```

IEFSJERS - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Erase SWB Parameter Routine

FUNCTION:

This module erases SWB information for a specified parameter on a JCL keyword or text unit key.

ENTRY POINT: IEFSJERS

PURPOSE: See Function

LINKAGE: CALL

CALLERS: Scheduler JCL Control Routine (IEFSJCNL)

INPUT:

SJF Erase SWB parameter list, IEFSJERP:

FIELD	LENGTH/MASK	DESCRIPTION
SJERP		Parameter list
SJERID	4	Identifier 'SJER'
SJERVERS	1	Version number
SJERFLAG	1	Control flags
SJERNREC	X'80'	No recovery
SJERNOCU	X'40'	No cleanup
SJERLEN	2	Length of parameter list
SJERSTOR	4	Local storage pointer
SJERREAS	4	Reason code (returned)
SJERTOKN		Token identifying SWB chain
SJERANBK	4	
SJERANCA	4	
SJERFUNC	1	Function byte
SJERJOUR	X'80'	Journalling requested
SJERRSV1	3	Reserved
SJERJDVT	8	JDVT name for keyword to erase
SJERVERB	8	Verb
SJERKEYW	8	Keyword
SJERPARM	2	Parameter number
SJERSUBL	2	Sublist element number
SJERKEY	2	Key

The input to this module also includes the Scheduler JCL Facility control workarea (IEFSJCNW).

OUTPUT: SJF Erase SWB parameter list, IEFSJERP:

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: ERSRETRY

PURPOSE:

Performs clean up processing when an ABEND occurs during SJF Erase's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

IEFSJERS - MODULE DESCRIPTION (Continued)

EXTERNAL REFERENCES:

ROUTINES:

IEFSJEXT	-	SJF Extract Routine
IEFXB501	-	Journal Write Routine

DATA AREAS:

IEFSJERP	-	SJF Erase SWB Parameter List
IEFSJCNW	-	SJF Control Workarea
IEFSJSWP	-	SJF SWA Block routine
IEFSJEXP	-	SJF Extract Parameter List
IEFSJRC	-	SJF Reason Codes

CONTROL BLOCKS: SWB - Scheduler Work Block

IEFSJERS - MODULE OPERATION

This module erases SWB information for a specified parameter on a JCL keyword or text unit key.

1. Validates the SWB token passed in using IEFSJSWA common include segment.

2. Validates the key or keyword passed by calling SJF Extract (this returns the location of the data in the SWB also).

3. Erases the data in the SWB so that it looks like the Keyword was never specified.

4. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery will receive control from RTM. The recovery routine specifies the retry address in the SJF workarea (ERSRETRY) to RTM. When the retry segment (ERSRETRY) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.
2. Returns to the caller.

IEFSJERS - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJERS
ERSRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJERS:

EXIT NORMAL:

Register 15 = 0 - Processing successful

Reason codes in SJERREAS:
SJRCNOER (0) - Processing successful

EXIT ERROR:

Register 15 = 4 - Request not processed

Reason codes in SJERREAS (decimal):

SJRCPRMN (1400) - Subparameter information does not exist in the SWB

Also, the reason codes returned by IEFSJEXT and IEFSJSWA.

ENTRY POINT ERSRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJERS:

Register 0 = Undefined
Register 1 = Address of a two word parameter list. The first word contains the address of the Erase SWB parameter list (IEFSJERP), and the second word contains the address of the SJF control workarea (IEFSJCNW)
Registers 2-12 = Undefined
Register 13 = Address of 18 word savearea
Register 14 = Return address
Register 15 = Entry point address

ENTRY POINT ERSRETRY:

Register 1 = Address of ESTAE parameter list
Registers 0,2-14 = Undefined
Register 15 = Entry point address

IEFSJERS - DIAGNOSTIC AIDS (Continued)

REGISTER CONTENTS ON EXIT:

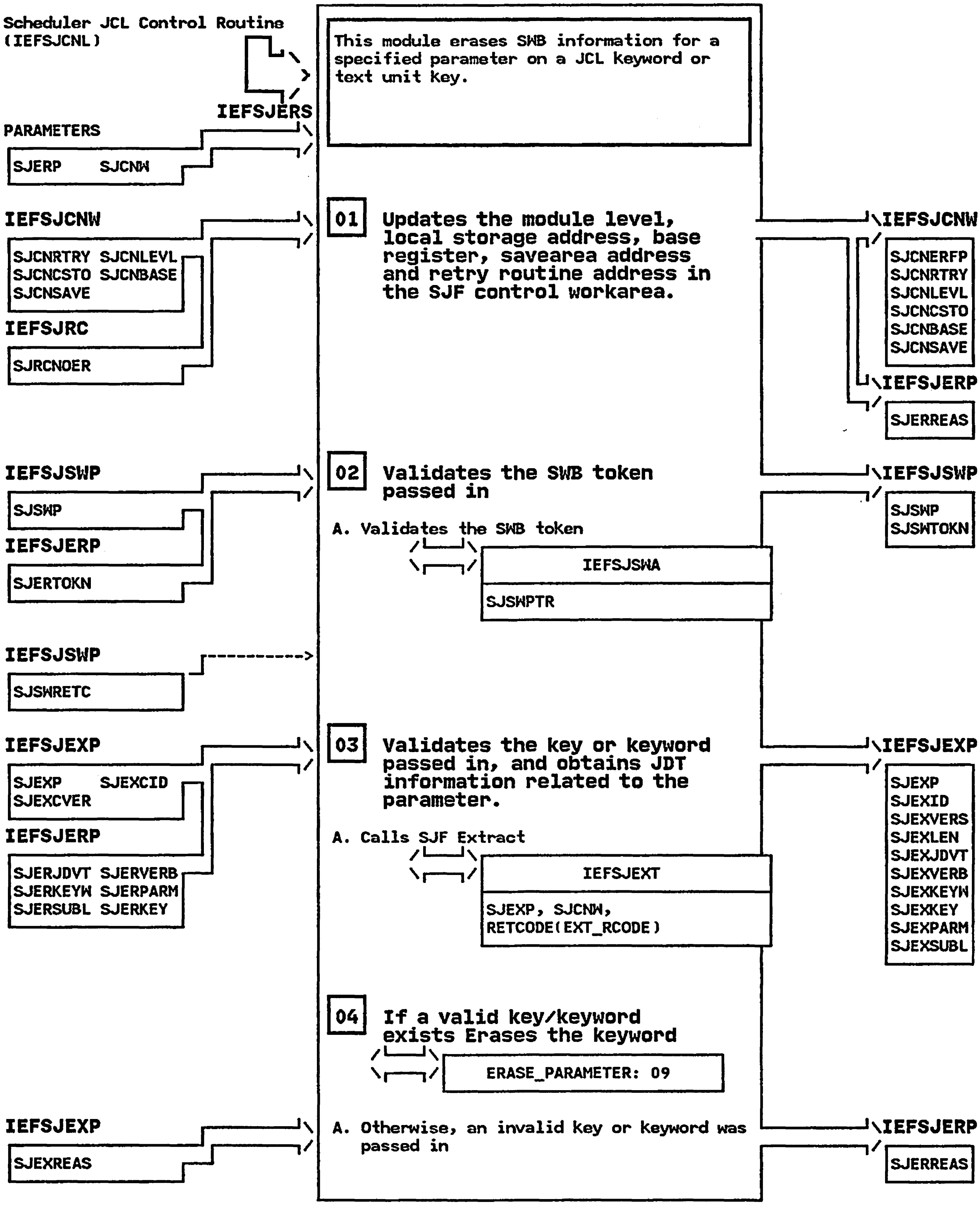
ENTRY POINT IEFSJERS:

```
Registers 0-14 = Restored
Register    15 = Return code
```

ENTRY POINT ERSRETRY:

```
Registers 0-14 = Restored
Register  15   = Return Code
```

IEFSJERS - Scheduler JCL Facility (SJF) Erase SWB Parameter Routine STEP 01



Scheduler JCL Control Routine (IEFSJCNL)

IEFSJERS

PARAMETERS

SJERP SJCNW

This module erases SWB information for a specified parameter on a JCL keyword or text unit key.

IEFSJCNW

SJCNRTRY SJCNLEVL
SJCNCSTO SJCNBASE
SJCNSAVE

IEFSJRC

SJRCNOER

01 Updates the module level, local storage address, base register, savearea address and retry routine address in the SJF control workarea.

IEFSJCNW

SJCNERFP
SJCNRTRY
SJCNLEVL
SJCNCSTO
SJCNBASE
SJCNSAVE

IEFSJERP

SJERREAS

IEFSJSWP

SJSWP

IEFSJERP

SJERTOKN

02 Validates the SWB token passed in

A. Validates the SWB token

IEFSJSWA

SJSWPTR

IEFSJSWP

SJSWP
SJSWTOKN

IEFSJSWP

SJSWRETC

IEFSJEXP

SJEXP SJEXCID
SJEXCVER

IEFSJERP

SJERJDVT SJERVERB
SJERKEYW SJERPARM
SJERSUBL SJERKEY

03 Validates the key or keyword passed in, and obtains JDT information related to the parameter.

A. Calls SJF Extract

IEFSJEXT

SJEXP, SJCNW,
RETCODE(EXT_RCODE)

IEFSJEXP

SJEXP
SJEXID
SJEXVERS
SJEXLEN
SJEXJDVT
SJEXVERB
SJEXKEYW
SJEXKEY
SJEXPARM
SJEXSUBL

04 If a valid key/keyword exists Erases the keyword

ERASE_PARAMETER: 09

IEFSJEXP

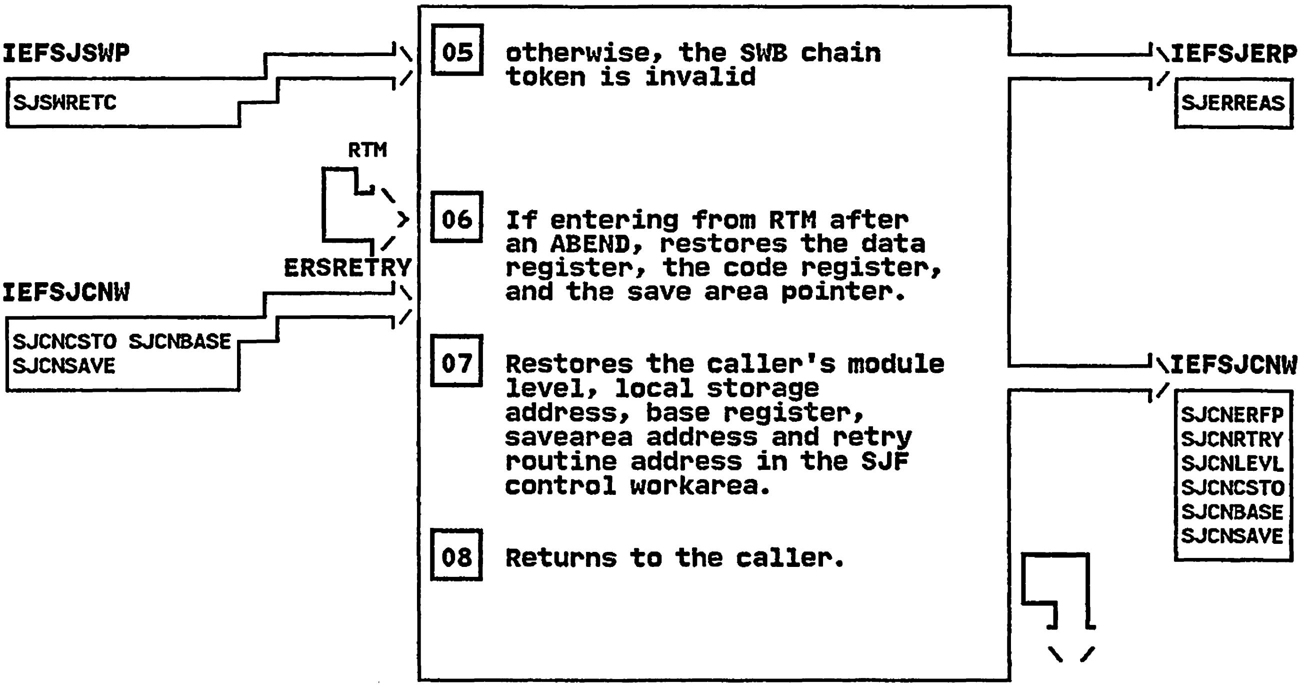
SJEXREAS

A. Otherwise, an invalid key or keyword was passed in

IEFSJERP

SJERREAS

IEFSJERS - Scheduler JCL Facility (SJF) Erase SWB Parameter Routine STEP 05

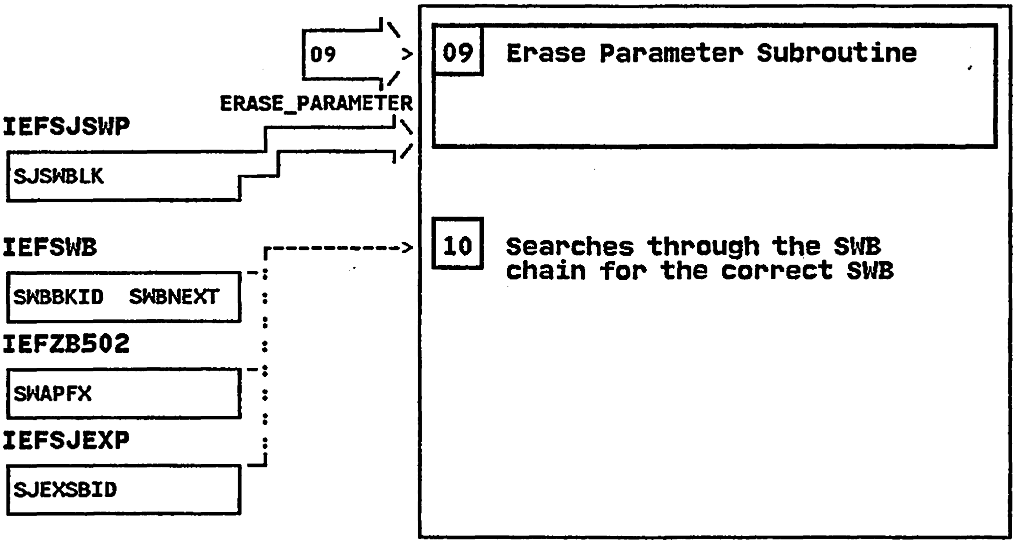


05 otherwise, the SWB chain token is invalid

06 If entering from RTM after an ABEND, restores the data register, the code register, and the save area pointer.

07 Restores the caller's module level, local storage address, base register, savearea address and retry routine address in the SJF control workarea.

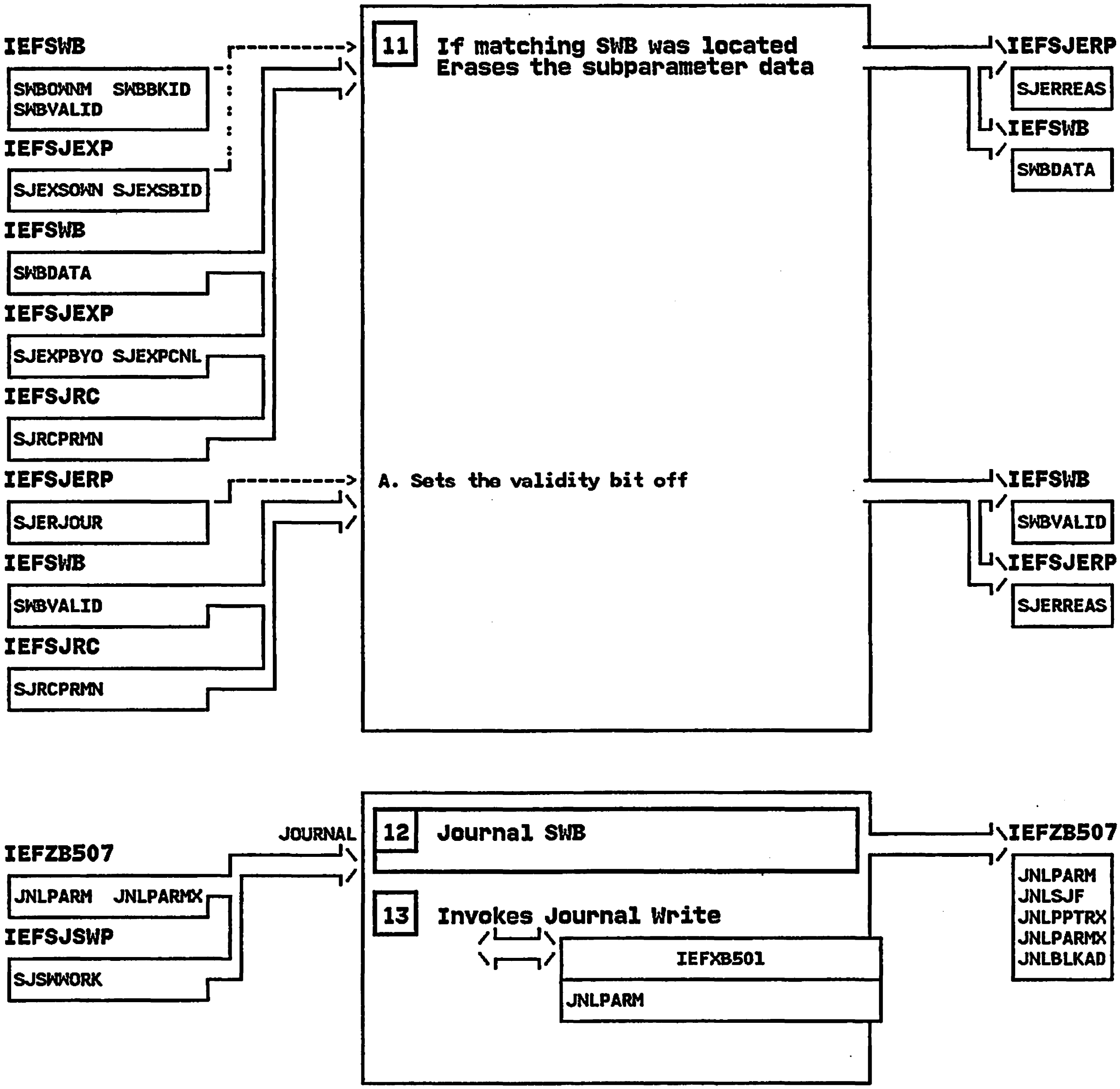
08 Returns to the caller.



09 Erase Parameter Subroutine

10 Searches through the SWB chain for the correct SWB

IEFSJERS - Scheduler JCL Facility (SJF) Erase SWB Parameter Routine STEP 11



11 If matching SWB was located
Erases the subparameter data

A. Sets the validity bit off

12 Journal SWB

13 Invokes Journal Write

IEFSJEXT - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Extract Routine

FUNCTION:
This module extracts information from the JCL definition table (JDT) associated with a verb, a verb and keyword, a verb and key, subparameters of a keyword or key, a command, a command and operand, and the subparameters of an operand.

ENTRY POINT: IEFSJEXT

PURPOSE: See Function

LINKAGE: CALL

CALLERS:
SJF control routine (IEFSJCNL)
SJF update routine (IEFSJUPD)
SJF retrieve routine (IEFSJRET)

INPUT:
SJF Extract Parameter List (IEFSJEXP)

FIELD	LENGTH/MASK	DESCRIPTION
SJEXP	232	Extract parameter list
SJEXID	4	Identifier 'SJEX'
SJEXVERS	1	Version number
SJEXFLAG	1	Control flags
SJEXNREC	X'80'	No recovery
SJEXNOCU	X'40'	No cleanup
SJEXUNAU	X'20'	Unauthorized caller
SJEXLEN	2	Length
SJEXSTOR	4	Local storage pointer or zero
SJEXREAS	4	Reason code
SJEXJDVT	8	Name of JDVT or zero
SJEXVERB	8	Verb
SJEXKEYW	8	Keyword
SJEXKEY	2	Key
SJEXPARM	1	Number of subparameter
SJEXSUBL	1	Number of sublist element
SJEXRSV4	4	Reserved
SJEXVFLG	1	Verb flags (returned)
SJEXVCTL	X'80'	Control statement
SJEXRSV3	1	Reserved
SJEXKSTM	8	Statement type for referral (returned)
SJEXSFLG	1	Keyword flags (returned)
SJEXSPOL	X'80'	Keyword to be spooled
SJEXKFLG	1	Keyword flags (returned)
SJEXSYST	X'80'	System input only
SJEXKJOB	X'40'	Keyword on job level statement only
SJEXKSTP	X'20'	Keyword on step level statement only
SJEXKREF	X'08'	Referral keyword
SJEXINFO	138	Parameter information (Returned)
SJEXPKEY	2	Key
SJEXDFLT	1	Default value for key
SJEXPMLN	1	Maximum length of parameter
SJEXPBYO	1	Byte offset into SWB
SJEXPCNL	1	Length of converted parameter in the SWB
SJEXPFL1	1	Parameter flags
SJEXPBOL	X'80'	Choice/boolean data
SJEXPCHR	X'40'	Character data

IEFSJEXT - MODULE DESCRIPTION (Continued)

SJEXPINT	X'20'	Integer data
SJEXPHEX	X'10'	Hexadecimal data
SJEXPREF	X'08'	Reference data
SJEXFL2	1	Parameter flags
SJEXPSUB	X'80'	Sublist data
SJEXPSFR	X'40'	First element of sublist
SJEXPFL3	1	Parameter flags
SJEXPDDF	X'80'	Key default choice specified
SJEXPMIN	1	Minimum length of parameter
SJEXPLNM	1	Maximum number of levels for name data
SJEXPLLN	1	Maximum length of level for name data
SJEXPHGH	4	High range of integer or hexadecimal data
SJEXPLOW	4	Low range of integer or hexadecimal data
SJEXPCHC	72	Choice data
SJEXPCHO	8	Choice
SJEXPVAL	1	Value of choice
SJEXSOWN	8	SWB owner
SJEXSBID	2	SWB block ID
SJEXPFL4	1	First character type flag
SJEXPFAL	X'80'	Any character
SJEXPFAP	X'40'	Alphabetic character
SJEXPFNU	X'20'	Numeric character
SJEXPFNA	X'10'	National character
SJEXPFSP	X'08'	Special character
SJEXPFL5	1	Other character type flag
SJEXPOAL	X'80'	Any character
SJEXPOAP	X'40'	Alphabetic character
SJEXPONU	X'20'	Numeric character
SJEXPONA	X'10'	National character
SJEXPOSP	X'08'	Special character
SJEXPFSN	1	Number of special characters defined for the first character
SJEXPFSA	16	Special characters defined for the first character
SJEXPOSN	1	Number of special characters defined for characters other than the first
SJEXPOSA	16	Special characters defined for characters other than the first
SJEXCMND	8	Command
SJEXOPER	10	Operand
SJEXPOIP	4	Address of area to contain the operand information
SJEXPOLN	1	Length of area to contain the operand information
SJEXORSV	3	Reserved
SJEXKEWD	8	Keyword
SJEXOINF	104	Operand information (returned)
SJEXOID	4	Identifier
SJEXROPR	10	Operand
SJEXOCHC	22	Operand choices
SJEXOCHA	(2)	Operand choice array
SJEXOCHO	10	Operand choice
SJEXOVAL	1	Value of operand choice
SJEXODLN	1	Length of descriptive name
SJEXODRS	3	Reserved
SJEXODES	64	Descriptive name for operand

IEFSJEXT - MODULE DESCRIPTION (Continued)

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:
SJF Extract Parameter List (IEFSJEXP)

FIELD	LENGTH/MASK	DESCRIPTION
SJEXREAS	4	Reason code
SJEXVFLG	1	Verb flags
SJEXKSTM	8	Statement type for referral
SJEXKFLG	1	Keyword flags
SJEXSFLG	1	Keyword flags
SJEXKEWD	8	Keyword name
SJEXINFO	138	Parameter information
SJEXOINF	104	Operand information

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: EXTRETRY

PURPOSE:
Performs cleanup processing when an abend occurs during the SJF extract routine's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES: IEFSJJDV - Scheduler JCL Facility (SJF) Find JDVT

DATA AREAS:
IEFSJCNW - Scheduler JCL Facility Control Workarea
IEFSJEXP - Scheduler JCL Facility Extract Parameter List
IEFSJJDP - Scheduler JCL Facility Find JDVT Parameter List
IEFSJRC - Scheduler JCL Facility Reason Codes

CONTROL BLOCKS:
JDT - JCL Definition Table
JDVT - JCL Definition Vector Table
SJCDENTY - JDT Command Entry
SJCDT - JDT Command Hash Table
SJKWENTY - JDT Keyword Entry
SJKWT - JDT Keyword Hash Table
SJKYENTY - JDT Key Entry
SJKYT - JDT Key Hash Table
SJOPENTY - JDT Operand Entry
SJOPT - JDT Operand Hash Table
SJVBENTY - JDT Verb Entry
SJVBT - JDT Verb Hash Table

IEFSJEXT - MODULE OPERATION

This module extracts information from the JCL definition table (JDT). It does the following:

1. If the JCL definition vector table (JDVT) name was not found on a previous invocation or if the JDVT name specified in the parameter list does not match the JDVT name found on a previous invocation, invokes the SJF find JDVT routine (IEFSJJDV) to find the JDVT to use. If the JDVT was not found, sets the reason code (SJEXREAS) and the return code to the reason code and the return code returned by find JDVT and returns.

2. If a verb was specified in the parameter list without a keyword or key, uses the hashing algorithm to locate the verb that matches the verb in the parameter list. If the verb is not found, sets the reason code (SJEXREAS) to indicate that the verb was not found.

3. If a verb and a keyword were specified in the parameter list (SJEXVERB and SJEXKEYW not zeros), uses the hashing algorithm to locate the verb and keyword that match the verb and keyword in the parameter list. If the verb and keyword are found, places the keyword information and the keyword flags from the JDT into the parameter list (SJEXKSTM, SJEXSFLG, and SJEXKFLG) and sets the JDT token field (SJCNJTKN) to point to the JDT verb entry and the JDT keyword entry. If the verb or the keyword is not found in the JDT, sets the reason code (SJEXREAS) to indicate that the verb was not found or the keyword was not found.

4. If a verb and a key were specified in the parameter list (SJEXVERB and SJEXKEY not zeros), uses the hashing algorithm to locate the verb and the key that match the verb and key in the parameter list. If the verb and the key are found, places the keyword information and the keyword flags from the JDT into the parameter list (SJEXKSTM, SJEXSFLG, and SJEXKFLG) and sets the JDT token field (SJCNJKN) to point to the JDT verb entry and the first JDT subparameter entry for the key. If the verb or key is not found in the JDT, sets the reason code (SJEXREAS) to indicate that the verb was not found or the key was not found.

5. If a command was specified in the parameter list without an operand, uses the hashing algorithm to locate the command that matches the command in the parameter list. If the command is not found, sets the reason code (SJEXREAS) to indicate that the command was not found.

6. If a command and operand were specified in the parameter list (SJEXCMND and SJEXOPER not zeroes), uses the hashing algorithm to locate the command and operand in the parameter list. If the command and operand are found, places the keyword information, keyword flags, the operand, operand descriptor and operand descriptor length into the parameter list (SJEXKSTM, SJEXSFLG, SJEXKFLG, SJEXROPR, SJEXODES, and SJEXODLN) and sets the JDT token field (SJCNJTKN) to point to the JDT verb entry for the command and the JDT keyword entry for the operand. If the command or the operand is not found in the JDT, sets the reason

IEFSJEXT - MODULE OPERATION (Continued)

code (SJEXREAS) to indicate that the command or the operand was not found.

7. If a subparameter number was specified in the parameter list (SJEXPARM), finds the JDT subparameter entry for the subparameter specified by the subparameter number (SJEXPARM) and the sublist element number (SJEXSUBL) for the keyword, key or operand. If the subparameter entry exists in the JDT, copies the subparameter information from the JDT entry into the parameter list(SJEXINFO). If the subparameter entry is not defined in the JDT for the subparameter specified, sets the reason code (SJEXREAS) to indicate the subparameter or sublist element is not defined for this keyword, key or operand.

8. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (EXTRETRY) in the SJF work area. When EXTRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.

2. Returns to the caller.

IEFSJEXT - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJEXT
EXTRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJEXT:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully

Reason codes in SJEXREAS:

SJRCNOER (0)	- All requested information returned

EXIT ERROR:

Register 15 = 4 - Request was not processed

Reason codes in SJEXREAS (in decimal):

SJRCNJDV (4)	- JDVT not found
SJRCNJCH (5)	- JDVT chain does not exist
SJRCNVRB (200)	- Verb not found
SJRCNKWD (201)	- Keyword not found
SJRCNKEY (202)	- Key not found
SJRCNPRM (203)	- Subparameter is not defined for this keyword or key
SJRCBKK (204)	- Both keyword and key specified
SJRCNSLE (206)	- Sublist element is not defined for this keyword
SJRCNCMD (207)	- Command not defined in JDT
SJRCNOPE (208)	- Operand not defined in JDT
SJRCBVC (209)	- Both verb and command specified
SJRCNOIP (210)	- No operand information pointer
SJRCVAOC (211)	- Verb and/or command not specified

ENTRY POINT EXTRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJEXT:

Register 0	= Undefined
Register 1	= Address of 2 words that contain the address of the input parameter list and the control work area
Registers 2-12	= Undefined
Register 13	= Address of 18-word save area
Register 14	= Return address
Register 15	= Entry point address

IEFSJEXT - DIAGNOSTIC AIDS (Continued)

ENTRY POINT EXTRETRY:

```
Register  0    = Undefined
Register  1    = Address of ESTAE parameter list
Registers 2-14 = Undefined
Register  15   = Entry point address
```

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJEXT:

```
Register  0    = Restored
Register  1    = Address of 2 words that contain the
                 address of the input parameter list and
                 the control work area.
Registers 2-12 = Restored
Register  13   = Address of 18-word savearea
Register  14   = Return address
Register  15   = Return code
```

ENTRY POINT EXTRETRY:

```
Registers 0-14 = Restored
Register  15   = Return code
```

IEFSJFND - MODULE DESCRIPTION

**DESCRIPTIVE NAME: Scheduler JCL Facility
Find SWB Chain Routine**

FUNCTION:

- Locates a scheduler work block (SWB) chain at a particular level of the scheduler work area (SWA) structure.
- Enables the caller to specify a search for the next SWB on the chain from where the last call of a SWB chain left off.
- Enables the caller to specify a starting address from which to start the SWB chain searches.
- Enables the caller to search for a verb and label within a control group (CNTL and ENDCNTL).
- Enables the caller to search for a SIOT or to search for the next SIOT or SCT on the chain.

ENTRY POINT: IEFSJFND

PURPOSE: See Function

LINKAGE: CALL

CALLERS:
SJF control routine (IEFSJCNL)
SJF update routine (IEFSJUPD)

INPUT:
SJF find SWB parameter list (IEFSJFNP)

FIELD	LENGTH/MASK	DESCRIPTION
SJFNP	72	Parameter list
SJFNID	4	Identifier 'SJFN'
SJFNVERS	1	Version number
SJFNFLAG	1	Function flags
SJFNNREC	x'80'	No recovery
SJFNNOCU	x'40'	No cleanup
SJFNLEN	2	Length of parameter list
SJFNSTOR	4	Local storage pointer
SJFNREAS	4	Reason code (returned)
SJFNINFO		Parameter information
SJFNFLG2	1	
SJFNNEXT	x'80'	Find next SWB processing
SJFNCNTL	x'40'	Search for a statement within a control group
SJFNSASP	x'20'	Starting address specified (Except for verb=DD)
SJFNIDSW	2	Identify the SWB to be found
SJFNFUN1	1	Non-master scheduler flag byte
SJFNJOB	x'80'	Job level
SJFNCST	x'40'	Current step level
SJFNST	x'20'	Step level or procname and step
SJFNFUN2	1	Master scheduler flag byte
SJFNMSTJ	x'80'	Job level
SJFNMSTS	x'40'	Current step level
SJFNFLG3	1	
SJFNJST	x'80'	Job token indicator
SJFNSTPN	8	Step name
SJFNCHID	16	SWB chain identification
SJFNVERB	8	Verb (Optional if not DD)
SJFNLABL	8	Statement label (Optional)
SJFNTOKN	8	SWB chain token
SJFNANBK	4	Address of control block for a JCL statement (JCT, SCT, SIOT or SWB) or the address of a SWB chain
SJFNANCA	4	Zero or address of a word

IEFSJFND - MODULE DESCRIPTION (Continued)

		pointing to a SWB chain
SJFNCNLB	8	Label on the CNTL statement
SJFNPRLB	8	Label on the PROC statement
SJFNSTMT	4	Statement number returned in hexadecimal.

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:
SJF find SWB parameter list (IEFSJFNP)

FIELD	LENGTH/MASK	DESCRIPTION
SJFNREAS	4	Reason code
SJFNTOKN	8	SWA block token
SJFNSTMT	4	Statement number

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: FNDRETRY

PURPOSE:
Performs clean up processing when an ABEND occurs during SJF find SWB processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES: None

DATA AREAS:

IEEBASEA	- Master Scheduler Resident Data Area
IEFZB502	- SWA Prefix
IEFSJRC	- SJF Common Reason Codes
IEFSJFNP	- SJF Find SWB Parameter list
IEFSJCNW	- SJF Control Work Area

CONTROL BLOCKS:

CVT	- Communications Vector Table
PSA	- Prefix Storage Area
TCB	- Task Control Block
JCT	- Job Control Table
JSCB	- Job Step Control Block
JCTX	- Job Control Table Extension
SCT	- Step Control Table
SCTX	- Step Control Table Extension
SIOT	- Step Input/Output Table
SWB	- Scheduler Work Block

TABLES: QMAT - SWA Manager Address Table

IEFSJFND - MODULE OPERATION

1. For all requests except those for the master scheduler's SWA or when the job token is specified, locates the active job step control block (JSCB) via:

   PSA -> TCB -> JSCB -> active JSCB

   The JSCB contains pointers to the JCT (JSCBJCT) and to the current SCT (JSCSCT). Processing depends on the level and input supplied by the caller:

   For requests when the job token is specified, the JSCB is not used to get addressability to the Scheduler Work Area (SWA). Addressability to SWA is gained through the JCT in the parameter list token.

LEVEL	DATA SUPPLIED	CHAINING STRUCTURE TO BE SEARCHED	RETURNED TO CALLER
Job	Verb&Label	JSCB -> JCT -> JCTX -> SWB Chain	For old token: SVA of anchor for SWB chain. Address of a word pointing to a SWB chain. For new token: SVA of SWB chain.
Current Step	Verb&Label (verb¬=DD)	JSCB -> SCT for current step -> SWB chain	For old token: SVA of anchor for SWB chain. Address of a word pointing to a SWB chain. For new token: SVA of SWB chain.
Step	Stepname, verb&label (verb¬=DD)	JSCB -> JCT -> SCT chain -> SCT for specified stepname -> SWB chain	For old token: SVA of anchor for SWB chain. Address of a word pointing to a SWB chain. For new token: SVA of SWB chain.
Step	Stepname and verb=DD	Job token (indicated by SJFNJST) -> JCT -> SCT chain -> SCT for specified step name -> SIOT for specified DD block	SVA of SIOT in the first word and the address of the SWB chain in the second word.
Step	Stepname and verb=DD	JSCB -> JCT -> SCT chain -> SCT	For old token: SVA of SIOT

IEFSJFND - MODULE OPERATION (Continued)

		for specified stepname -> SIOT chain -> SIOT for specified DD label -> SWB chain	for the SWB. Address of word pointing to the SWB. For new token: SVA of SIOT for SWB chain.
Current Step	verb=DD	JSCB -> SCT for current step -> SIOT chain -> SIOT for specified DD label -> SWB chain	SVA of SIOT in the first word and the address of a word pointing to the SWB chain in the second word.

2. For the requests referring to the master scheduler's SWA, the chaining structure is different:

LEVEL	CHAINING STRUCTURE TO BE SEARCHED
Job	CVT -> BASEA -> JSCB (for Master Scheduler) -> JCT -> JCTX
Step	CVT -> BASEA -> JSCB (for Master Scheduler) -> SCT
Step & verb=DD	CVT -> BASEA -> JSCB (for Master Scheduler) -> SCT -> SIOT -> SIOT for specified DD label.

3. The input token may be in the old or new format. The new format uses the first word of the token to point to the SWA block. The second word is not used. For the old token format, both words are used.
   For a find next or starting address call to SJF Find, the first word of the input token may point to the first control block for a JCL statement (JCT, SCT, SIOT, or SWB).

4. If the find next SWA block indicator is on (SJFNNEXT) finds the SWA block with the verb and label requested at the job level, current step level, or the stepname level.
   Locates the SWA block to be returned as follows:

   - If the verb and label are specified, finds the first matching SWA block and returns the address of its chain.
   - If only the verb is specified (label=0), finds the first matching SWA block by its verb and returns its label and the address of its chain.
   - If only label is specified (verb=0), finds the first matching SWA block by its label and returns its verb and the address of its chain.
   - If neither verb nor label is specified (both

IEFSJFND - MODULE OPERATION (Continued)

are zeroes), returns the verb and label, and the address of the SWA block at the level specified.

5. If the starting address specified indicator is on (SJFNSASP), then SJFNANCA is used for an old token as the start address. For a new token SJFNANBK is used as the initial starting address. This type of search does not support verb=DD invocations.

6. If a search within a control group is indicated (SJFNCNTL), then the SWB chains will be scanned until a CNTL SWB is found. Searching is then done the same way as usual. If the group label is zero (SJFNCNLB), then all the control groups at the level specified will be searched for a verb and label match. The find next function may also be specified with a control group search.

7. The module returns to the caller the address of the SWB chain.

RECOVERY OPERATION:

If an ABEND occurs in this module, the scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (FNDRETRY) in the SJF control workarea. When FNDRETRY (in this module) receives control, it does the following:

1. Sets the return code to indicate an SJF system error.
2. Returns to the caller.

IEFSJFND - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJFND
FNDRETRY

MESSAGES: None

ABEND CODES:

X'054' (decimal 84) and a reason code of 100 in register 15 occurs when an SVA for the JCT, JCTX, SCT, SCTX, SIOT, or SWB can not be translated successfully.

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJFND:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully
Reason code in SJFNREAS:
SJRCNOER (0) - Find SWB successful

EXIT ERROR:

Register 15 = 4 - Request not completed successfully
Reason code in SJFNREAS (decimal):
SJRCIVTK (2) - Invalid SWB token
SJRCNSCH (400) - Specified SWB chain not found (set if the verb and label are not found)
SJRCSTEP (401) - Specified STEP or PROC name not found
SJRCDDNM (402) - Specified DD label not found
SJRCNBIT (403) - No search bits specified in parm list
SJRCEBIT (404) - Undefined bits specified in parm list
SJRCNGRP (405) - Control group not found
SJRCNOST (406) - No Step Name specified in parm list
SJRCINAN (407) - Invalid starting address specified in parameter list
SJRCINVJ (408) - Invalid job or step token specified in parameter list

ENTRY POINT FNDRETRY:

EXIT ERROR:

Register 15 = 20 (decimal) - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJFND:

Register 0 = Undefined
Register 1 = Address of two words that contain the address of the input parameter list (IEFSJFNP) and the address of the control work area (IEFSJCNW).
Registers 2-12 = Undefined
Register 13 = Address of 18-word save area
Register 14 = Return address

IEFSJFND - DIAGNOSTIC AIDS (Continued)

```
  Register    15 = Entry point address
```

ENTRY POINT FNDRETRY:

```
  Register     0 = Undefined
  Register     1 = Address of the ESTAE parameter list
  Registers 2-14 = Undefined
  Register    15 = Entry point address
```

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJFND:

```
    Registers 0-14 = Restored
    Register    15 = Return code
```

ENTRY POINT FNDRETRY:

```
    Registers 0-14 = Restored
    Register    15 = Return code
```

IEFSJGET - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Get SWB Chain Routine

FUNCTION:

This module copies selected keywords from a SWB chain in text unit format into a storage area specified by the caller. The keywords obtained are those keywords whose JDT flags match the qualifier flags set in the input parameter list.

ENTRY POINT: IEFSJGET

PURPOSE: See Function

LINKAGE: CALL

CALLERS: SJF control routine (IEFSJCNL)

INPUT:

Get SWB parameter list, IEFSJGEP:

FIELD	LENGTH/MASK	DESCRIPTION
SJGEP		Parameter list
SJGEID	4	Identifier 'SJGE'
SJGEVERS	1	Version number
SJGEFLAG	1	Control flags
SJGENREC	x'80'	No recovery
SJGENOCU	x'40'	No clean up
SJGELEN	2	Length of parameter list
SJGESTOR	4	Local storage pointer
SJGEREAS	4	Reason code (returned)
SJGETOKN	8	SWB token
SJGEANBK	4	Address of anchor control block or of the first control block for a JCL statement
SJGEANCA	4	Address of word pointing to a SWB chain or zero
SJGEQUAL	2	Bit qualifiers for SWB
SJGEPOSI	1	Attributes requested
SJGESPL	x'80'	Keywords spooled for output processing
SJGENEGA	1	Attributes not requested
SJGENSPL	x'80'	Keywords not spooled for output processing
SJGERSV2	2	Reserved
SJGESWBA	4	Address of area to copy keyword data
SJGEALEN	2	Length of keyword data area
SJGERSV4	2	Reserved
SJGEJDVT	8	JDVT name

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:

Data returned in the get SWB parameter list, IEFSJGEP:

SJGEREAS = Reason code

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: GETRETRY

IEFSJGET - MODULE DESCRIPTION (Continued)

PURPOSE:
Performs clean up processing when an abend occurs during SJF get's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES:

IEFSJJDV	- SJF Find JDVT Routine
IEFSJRET	- SJF Retrieve Routine

DATA AREAS:

IEFSJCNW	- SJF Control Workarea
IEFSJRC	- SJF Reason Codes
IEFJDT	- JCL Definition Table
IEFJDVT	- JCL Definition Vector Table
IEFSJGEP	- SJF Get SWB Parameter List
IEFSJJDP	- SJF Find JDVT Parameter List
IEFSJREP	- SJF Retrieve Parameter List
IEFSJPFX	- NJE Prefix
IEFSJSWP	- IEFSJSWA Parameter List
IEFZB502	- SWA Prefix

CONTROL BLOCKS:

CVT	- Communication Vector Table
IEFAJCTB	- Job Control Table
IEFASCTB	- Step Control Table
IEFASIOT	- Step Input/Output Table
IEFJCTX	- Job Control Table Extension
IEFJESCT	- JES Communication Table
IEFSWB	- Scheduler Work Block

SERIALIZATION:

No locks or resources are obtained by this module.

IEFSJGET - MODULE OPERATION

IEFSJGET copies the keywords from a SWB chain whose JDT flags match the qualifier flags in the input parameter list. The routine places the keywords in a storage area provided by the caller. IEFSJGET performs the following functions:

1. Gets the address of the SWB chain by using the IEFSJSWA procedure to interpret the SWB token and verify that it is pointing to a valid SWB chain. If validation is not successful, sets register 15 to 4, sets a reason code of SJRCIVTK (2) in SJGEREAS, and returns.

2. Ensures that a keyword data area address (SJGESWBA) was specified in the parameter list.
   If SJGESWBA is zero, sets register 15 to 4, sets a reason code of SJRCGSWB (1000) in SJGEREAS, and returns.

3. The length of the keyword data area (SJGEALEN) is checked to make sure it is non-zero and large enough to contain at least the NJE prefix. If these conditions are not met, sets a reason code of SJRCGLEN (1002) in SJGEREAS, and returns.

4. If the SWB chain address has changed from the last invocation of IEFSJGET or there was no previous invocation, then this module determines whether the JDVT pointer in the SJF control workarea is different than was specified in the input parameter list. If the JDVT name is different, invokes SJF find JDVT to obtain the JDVT and its associated JDTs that correspond to the JDVT name specified by the caller.

5. Obtains the verb name from the SWB chain specified by the caller. This verb name is used as a search argument through each of the JDTs. (Note: The same verb may be specified multiple times in the same JDT or in one or more JDTs). Each time IEFSJGET finds a match in the JDT for the verb name, it counts the keywords and subparameters and keeps a total of the sizes of subparamater data. This is used to determine the maximum amount of storage needed for the keyword list (SJREKWDL) and text unit area (SJREAREA) passed to the SJF retrieve routine.

6. If keywords were found in the JDTs for the verb specified, obtains storage for the keyword list and text unit area. Makes a second pass for the verb name through the JDTs in order to move the keywords for each verb entry into the keyword list, omitting any duplicate keywords found. If a match on the verb name was found in the JDTs, but no keyword entries were found for the verb, builds a NJE prefix with no keyword data.

7. If keywords were found, invokes the SJF retrieve routine to obtain text unit information for the keywords specified in the keyword list. If SJF retrieve was successful, copies an NJE prefix and text unit information for those keywords found on the SWB chain into the area speicified by the caller (SJGESWBA). If not enough storage is available to contain all the keyword text unit data, sets register 15 to 4, sets a reason code of SJRCMORE (1001) in SJGEREAS, and returns. Another invocation of SJF get will be necessary to obtain the remainder of the

IEFSJGET - MODULE OPERATION (Continued)

text unit data.

8. If this is a multiple invocation to obtain the remainder of text unit data, the address of the keyword list, the index into the keyword list, the index into the text unit pointer list, the number of parameters already processed, and the amount of the text unit already processed exists in the SJF control workarea. The copying of text unit data continues from where it left off in the previous invocation.

9. Returns to caller.

RECOVERY OPERATION:

If an abend occurs in this module, the Scheduler JCL Facility control routine's (IEFSJCNL) recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (GETRETRY) in the SJF control workarea. When GETRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.

2. Determines if any storage has been obtained via a GETMAIN macro and frees it if it has not already been freed.

3. Returns to caller.

IEFSJGET - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJGET
GETRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJGET:

EXIT NORMAL:

Register 15 = 0 - Processing successful

Reason Code in SJGEREAS =
SJRCNOER (0) - Processing successful

EXIT ERROR:

Register 15 = 4 - Request cannot be processed

Reason Codes in SJGEREAS =
SJRCIVTK (2) - Invalid SWB chain address
SJRCGSWB (1000) - Invalid SWB get keyword area address
SJRCMORE (1001) - More keyword data to be obtained
SJRCGLEN (1002) - Invalid length for keyword data area
SJRCGEGM (1003) - Unable to GETMAIN storage for keyword list or text or text unit area

ENTRY POINT GETRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJGET:

Register 0 = Undefined
Register 1 = Address of a two word parameter list. The first word contains the address of the get SWB parameter list (IEFSJGEP) and the second word contains the address of the SJF control workarea (IEFSJCNW)
Registers 2-12 = Undefined
Register 13 = Address of an 18-word savearea
Register 14 = Return address
Register 15 = Entry point address

ENTRY POINT GETRETRY:

Register 1 = Address of ESTAE parameter list
Registers 0,2-14 = Undefined

IEFSJGET - DIAGNOSTIC AIDS (Continued)

```
Register      15 = Entry point address
```

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJGET:

```
Registers 0-14 = Restored
Register    15 = Return code
```

ENTRY POINT GETRETRY:

```
Registers   0-14 = Restored
Register      15 = Return code
```

IEFSJHTB - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Hash Table Build Routine

FUNCTION:

This module builds hash tables to provide access to information in the JDT's given a verb and a keyword, a verb and a key, or a command and an operand.

ENTRY POINT: IEFSJHTB

PURPOSE: See Function

LINKAGE: CALL

CALLERS: SJF define JDVT (IEFSJDEF)

INPUT:

SJF Hash Table Build Parameter List (IEFSJHTP)

FIELD	LENGTH/MASK	DESCRIPTION
SJHTP	20	Parameter list
SJHTID	4	Identifier 'SJHT'
SJHTVERS	1	Version number
SJHTFLAG	1	Function flags
SJHTNREC	X'80'	No recovery
SJHTNOCU	X'40'	No cleanup
SJHTLEN	2	Length of parameter list
SJHTSTOR	4	Local storage pointer
SJHTREAS	4	Reason code (returned)
SJHTJDVP	4	Pointer to the JDVT
SJHTCID	C'SJHT'	Identifier
SJHTCVER	X'01'	Current version of macro

The input to this module also includes the SJF control work area (IEFSJCNW).

OUTPUT:

SJF hash table build parameter list (IEFSJHTP)

FIELD	LENGTH/MASK	DESCRIPTION
SJHTREAS	4	Reason code

The hash table structure consisting of the hash tables and entries is also output.

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: HTBRETRY

PURPOSE:

Performs cleanup processing when an abend occurs during the SJF hash table build routine's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: None

OUTPUT: None

IEFSJHTB - MODULE DESCRIPTION (Continued)

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES: None

DATA AREAS:

IEFSJCNW	- Scheduler JCL Facility Control Workarea
IEFSJHTP	- Scheduler JCL Facility Hash Table Build Parameter List
IEFSJRC	- Scheduler JCL Facility Reason Codes

CONTROL BLOCKS:

JDT	- JCL Definition Table
JDVT	- JCL Definition Vector Table
SJCDENTY	- JDT Command Entry
SJCDT	- JDT Command Hash Table
SJKWENTY	- JDT Keyword Entry
SJKWT	- JDT Keyword Hash Table
SJKYENTY	- JDT Key Entry
SJKYT	- JDT Key Hash Table
SJOPENTY	- JDT Operand Entry
SJOPT	- JDT Operand Hash Table
SJVBENTY	- JDT Verb Entry
SJVBT	- JDT Verb Hash Table

TABLES:

SJVBT	- JDT Verb Hash Table
SJCDT	- JDT Command Hash Table
SJKWT	- JDT Keyword Hash Table
SJKYT	- JDT Key Hash Table
SJOPT	- JDT Operand Hash Table

IEFSJHTB - MODULE OPERATION

This module builds hash tables to provide access to information in the JDT's. It does the following:

1. Builds the verb hash table and the command hash table for the JDT's defined in the JDVT.

2. For each verb in the JDTs, does the following:

   a. If a verb entry does not exist, builds and initializes a verb entry, a Keyword hash table, and a Key hash table. Anchors the Keyword hash table and the key hash table in the verb entry. Anchors the verb entry at the index into the verb hash table found by using the hashing algorithm or at the end of the verb entry synonym chain.

   b. For each Keyword specified for the verb does the following:

      - If a Keyword entry does not exist, builds and initializes a Keyword entry. Anchors the Keyword entry at the index into the Keyword hash table found by using the hashing algorithm or at the end of the Keyword entry synonym chain. If a Keyword entry already exists, issues abend x'054' with a reason code of 04.

      - For each command: if a command entry does not exist, builds and initializes a command entry and an operand hash table. Anchors the operand hash table in the command entry. Anchors the command entry at the index into the command hash table found by using the hashing algorithm or at the end of the command entry synonym chain.

      - For each operand, its valid abbreviations, and each operand hash table: if an operand entry does not exist, builds and initializes an operand entry. Anchors the operand entry at the index into the operand hash table found by using the hashing algorithm or at the end of the operand entry synonym chain. If an operand entry exists for a different Keyword, issues abend x'054' with a reason code of 06.

      - For each key: if a key entry does not exist, builds and initializes a Key entry. Anchors the Key entry at the index into the Key hash table found by using the hashing algorithm or at the end of the Key entry synonym chain. If a Key entry exists for a different Keyword, issues abend x'054' with a reason code of 05.

3. If any errors occur in obtaining storage before the hash table structure is in a usable state, frees all of the storage that was previously obtained.

4. If the structure can be used in its current state, anchors the verb hash table and command hash table in the JDVT and frees any remaining storage.

5. Returns to the caller.

IEFSJHTB - MODULE OPERATION (Continued)

RECOVERY OPERATION:

An ESTAE routine exists for IEFSJHTB. If an abend occurs in this module, the Scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (HTBRETRY) in the SJF work area. When HTBRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.

2. If the structure can not be used in its current state, frees all storage obtained.

3. If the structure can be used, anchors the verb hash table and the command hash table in the JDVT and frees any remaining storage.

4. Returns to the caller.

IEFSJHTB - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJHTB
HTBRETRY

MESSAGES: None

ABEND CODES:

Abend code hex 054 (dec 084) and a reason code of 4, 5, or 6 in register 15 occurs when an error in the JDTs is detected.

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJHTB:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully

Reason codes in SJHTREAS:
SJRCNOER (0)   - Hash table build successful

EXIT ERROR:

Register 15 = 4 - Request not processed

Reason codes in SJHTREAS:
SJRCSUHT (304) - Storage unavailable for hash tables

ENTRY POINT HTBRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJHTB:

Register	0	= Undefined
Register	1	= Address of 2 words that contain the address of the input parameter list and the control workarea
Register	2-12	= Undefined
Register	13	= Address of 18-word save area
Register	14	= Return address
Register	15	= Entry point address

ENTRY POINT HTBRETRY:

Register	0	= Undefined
Register	1	= Address of ESTAE parameter list
Register	2-14	= Undefined
Register	15	= Entry point address

REGISTER CONTENTS ON EXIT:

IEFSJHTB - DIAGNOSTIC AIDS (Continued)

ENTRY POINT IEFSJHTB:

Register	0	= Restored
Register	1	= Address of 2 words that contain the address of the input parameter list and the control workarea
Register	2-12	= Restored
Register	13	= Address of 18-word save area
Register	14	= Return address
Register	15	= Return code

ENTRY POINT HTBRETRY:

Register	0-14	= Restored
Register	15	= Return code

IEFSJINT - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Faciltiy
JDVT Initialization Routine

FUNCTION:
This module builds the system default JCL definition vector table and the hash table structure by supplying information in the SJF define JDVT parameter list (IEFSJDFP) and invoking the SJF define JDVT routine to process the request.

ENTRY POINT: IEFSJINT

PURPOSE: See Function

LINKAGE: CALL

CALLERS: SJF control routine (IEFSJCNL)

INPUT:
JDVT initialization parameter list, IEFSJINP:

FIELD	LENGTH/MASK	DESCRIPTION
SJINP		Parameter list
SJINID	4	Identifier 'SJIN'
SJINVERS	1	Version number
SJINFLAG	1	Control flags
SJINNREC	x'80'	No recovery
SJINNOCU	x'40'	No clean up
SJINLEN	2	Length of parameter list
SJINSTOR	4	Local storage pointer
SJINREAS	4	Reason code (returned)

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:
Data returned in the SJF JDVT initialization parameter list, IEFSJINP:

SJINREAS = Reason code

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: INTRETRY

PURPOSE:
Performs clean up processing when an abend occurs during SJF JDVT initialization processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

IEFSJINT - MODULE DESCRIPTION (Continued)

ROUTINES:

IEFSJDEF - SJF Define JDVT Routine
IEFJSIMW - System Initialization Message Writer

DATA AREAS:

IEFSJCNW - Scheduler JCL Facility Control Workarea
IEFSJDFP - Scheduler JCL Facilty Define JDVT Parameter List
IEFSJINP - Scheduler JCL Facility JDVT Initialization Parameter List
IEFSJRC - Scheduler JCL Facilty Reason Codes

CONTROL BLOCKS:

CVT - Communications Vector Table
IEFJESCT - JES Control Table

IEFSJINT - MODULE OPERATION

IEFSJINT builds the system default JDVT. It does the following:

1. Fills in the SJF define JDVT parameter list.

2. Invokes SJF define JDVT routine (IEFSJDEF) to build the system default JDVT.

3. Checks that a JDVT was successfully built by IEFSJDEF and that the hash table structure was successfully built by IEFSJHTB. If the request could not be processed, IEFSJINT determines the type of error by checking the reason code and issues the corresponding version of message IEF818E to the operator. If an unexpected return code or reason code was returned by IEFSJDEF, IEFSJINT issues an abend code of 054 and a reason code of 3 in register 15.

4. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (INTRETRY) in the SJF workarea. When INTRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.
2. Returns to the caller.

IEFSJINT - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJINT
INTRETRY

MESSAGES:

IEF818E JCL USAGE LIMITED - MODULE name NOT FOUND

IEF818E JCL USAGE LIMITED - STORAGE UNAVAILABLE

IEF818E JCL USAGE LIMITED - UNABLE TO SET UP RECOVERY ENVIRONMENT

IEF818E JCL USAGE LIMITED - SYSTEM ERROR IN JCL INITIALIZATION

ABEND CODES:

Abend code X'054' (dec 084) and a reason code of 3 in register 15 occurs when an unexpected reason code or return code is returned from the SJF Define JDVT routine.

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJINT:

EXIT NORMAL:

Register 15 = 0 - Processing successful

Reason Code in SJINREAS =
SJRCNOER (0) - Processing successful

EXIT ERROR:

Register 15 = 4 - Request cannot be processed

Reason Codes in SJINREAS =
SJRCNJDT (300) - JDT not found
SJRCGETJ (303) - Getmain for JDVT failed
SJRCSUHT (304) - Storage unavailable to build hash table structure
SJRCNSDT (306) - Statement Definition Table (SDT) not found

ENTRY POINT INTRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJINT:

Register 0 =	Undefined
Register 1 =	Address of a two word parameter list. The first word contains the address of the JDVT initialization parameter list (IEFSJINP) and the

IEFSJINT - DIAGNOSTIC AIDS (Continued)

```
                    second word contains the address
                    of the SJF control workarea
                    (IEFSJCNM).
   Registers 2-12 = Undefined
   Register    13 = Address of an 18-word savearea
   Register    14 = Return address
   Register    15 = Entry point address
```

ENTRY POINT INTRETRY:

```
   Register       1 = Address of ESTAE parameter list
   Registers 0,2-14 = Undefined
   Register      15 = Entry Point Address
```

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJINT:

```
     Registers 0-14 = Restored
     Register    15 = Return code
```

ENTRY POINT INTRETRY:

```
     Registers   0-14 = Restored
     Register      15 = Return code
```

IEFSJJDV - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Find JDVT Routine

FUNCTION:

This module locates a JCL definition vector table (JDVT) identified by one of the following:

1. A JDVT name specified in the input parameter list
2. The JDVT specified in the JCTX
3. The default JDVT for the system.

ENTRY POINT: IEFSJJDV

PURPOSE: See Function

LINKAGE: CALL

CALLERS:
SJF control routine (IEFSJCNL)
SJF extract routine (IEFSJEXT)

INPUT:
SJF Find JDVT parameter list, IEFSJJDP:

FIELD	LENGTH/MASK	DESCRIPTION
SJJDP	24	Parameter list
SJJDID	4	Identifier 'SJJD'
SJJDVERS	1	Version number
SJJDFLAG	1	Control flags
SJJDNREC	x'80'	No recovery
SJJDNOCU	x'40'	No clean up
SJJDLEN	2	Length of parameter list
SJJDSTOR	4	Local storage pointer
SJJDREAS	4	Reason code (returned)
SJJDJDVT	8	Name of JDVT or zero

The input to this module also includes the Scheduler JCL Facility control workarea (IEFSJCNW).

OUTPUT:
SJF Find JDVT parameter list, IEFSJJDP:

SJJDJDVT = JDVT name (if zero on entry)
SJJDREAS = reason code

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: JDVRETRY

PURPOSE:
Performs cleanup processing when an ABEND occurs during SJF find JDVT's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

IEFSJJDV - MODULE DESCRIPTION (Continued)

EXTERNAL REFERENCES:

ROUTINES: None

DATA AREAS:

IEFSJCNW	-	Scheduler JCL Facility Control Workarea
IEFSJJDP	-	Scheduler JCL Facility Find JDVT Parameter List
IEFSJRC	-	Scheduler JCL Facility Reason Codes
IEFZB502	-	SWA Prefix

CONTROL BLOCKS:

CVT	-	Communications Vector Table
JCT	-	Job Control Table
JCTX	-	Job Control Table Extension
JDVT	-	JCL Definition Vector Table
JESCT	-	Job Entry Subsystem Control Table
JSCB	-	Job Step Control Table
PSA	-	Prefixed Save Area
TCB	-	Task Control Block

SERIALIZATION:

No locks or resources are obtained by this module.

IEFSJJDV - MODULE OPERATION

IEFSJJDV locates a JDVT that is either specified in the input parameter list, specified in the JCTX, or the default JDVT for the system. It performs the following functions:

1. If a JDVT chain does not exist off the JESCT, sets a reason code of SJRCNJCH (5) in SJJDREAS and returns to the caller.

2. Otherwise, checks the input parameter list for a specified JDVT name (SJJDJDVT). If a JDVT name is specified, searches the chain of JDVTs, anchored off the JESCT, for a JDVT with a matching name.

3. If a JDVT name is not specified in the input parameter list (SJJDJDVT = 0), then:
   - If a JDVT name exists in the job control table extension (JCTXJVTN not 0), searches the JDVT chain, anchored off the JESCT, for a JDVT with a matching name.
   - If there is no JDVT name in the JCTX, searches the JDVT chain, anchored off the JESCT, for the system default JDVT (bit JDVTDFLT is on).

4. If a JDVT was located, returns the JDVT name in SJJDJDVT in the parameter list and stores the JDVT address in the SJF control workarea (SJCNUSEJ).

5. If the JDVT was not located, sets a reason code of SJRCNJDV (4) in SJJDREAS to indicate that condition.

6. Returns to the caller.

RECOVERY OPERATION:

If an ABEND occurs in this module, the scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (JDVRETRY) in the SJF control workarea. When JDVRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.

2. Returns to the caller.

IEFSJJDV - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJJDV
JDVRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJJDV:

EXIT NORMAL:

Register 15 = 0 - Processing successful

Reason codes in SJJDREAS:
SJRCNOER (0) - Processing successful

EXIT ERROR:

Register 15 = 4 - Request cannot be processed

Reason codes in SJJDREAS:
SJRCNJDV (4) - The JDVT does not exist
SJRCNJCH (5) - The JDVT chain does not exist.

ENTRY POINT JDVRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJJDV:

```
Register  0     = Undefined
Register  1     = Address of a two word parameter list.
                  The first word contains the address
                  of the Find JDVT parameter list
                  (IEFSJJDP), and the second word con-
                  tains the address of the Scheduler JCL
                  Facility control workarea (IEFSJCNM)
Registers 2-12 = Undefined
Register  13    = 18 word savearea
Register  14    = Return address
Register  15    = Entry point address
```

ENTRY POINT JDVRETRY:

```
Register  1       = Address of ESTAE parameter list
Registers 0,2-14 = Undefined
Register  15      = Entry point address
```

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJJDV:

IEFSJJDV - DIAGNOSTIC AIDS (Continued)

```
Registers 0-14 = Restored
Register    15 = Return code
```

ENTRY POINT JDVRETRY:

```
Registers 0-14 = Restored
Register  15   = Return Code
```

IEFSJPUT - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Put SWB Chain Routine

FUNCTION:

IEFSJPUT rebuilds a SWB chain from SWB keyword data found in text unit format.

ENTRY POINT: IEFSJPUT

PURPOSE: See Function

LINKAGE: CALL

CALLERS: SJF control routine (IEFSJCNL)

INPUT:

Put SWB parameter list, IEFSJPUP:

FIELD	LENGTH/MASK	DESCRIPTION
SJPUP		Parameter list
SJPUID	4	Identifier 'SJPU'
SJPUVERS	1	Version number
SJPUFLAG	1	Control flags
SJPUNREC	x'80'	No recovery
SJPUNOCU	x'40'	No clean up
SJPULEN	2	Length of parameter list
SJPUSTOR	4	Local storage pointer
SJPUREAS	4	Reason code (returned)
SJPUTOKN	8	SWB token
SJPUANBK	4	Address of anchor control block or of the first control block for a JCL statement
SJPUANCA	4	Address of word pointing to a SWB chain or zero
SJPUSWBA	4	Address of keyword data area
SJPUALEN	2	Length of area containing keyword data
SJPUFLG2	2	Flags
SJPUNSWA	x'80'	SWBs to be built in non-SWA
SJPUWARN	x'40'	Continue processing after an ignorable error is encountered. Ignorable errors are due to changes in the JDTs from release to release
SJPURSV2	4	Reserved
SJPUJDVT	8	JDVT name

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:

Put SWB parameter list, IEFSJPUP:

SJPUREAS = Result reason code

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: PUTRETRY

PURPOSE:

Performs clean up processing when an abend occurs during SJF put's processing.

LINKAGE: SYNCH

IEFSJPUT - MODULE DESCRIPTION (Continued)

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES:
SJF Update Routine (IEFSJUPD)
SJF Write SWB Routine (IEFSJWRT)

DATA AREAS:

IEFSJCNW	- SJF Control Workarea
IEFSJRC	- SJF Reason Codes
IEFSJPFX	- NJE Prefix
IEFSJPUP	- SJF Put Parameter List
IEFSJRUP	- SJF Update Parameter List
IEFSJSWP	- IEFSJSWA Parameter List
IEFSJWRT	- SJF Write Parameter List
IEFZB502	- SWA Prefix
IEFZB4D1	- Dynamic Allocation Text Unit Pointer List

CONTROL BLOCKS:

CVT	- Communication Vector Table
IEFAJCTB	- Job Control Table
IEFASCTB	- Step Control Table
IEFASIOT	- Step Input/Output Table
IEFJCTX	- Job Control Table Extension
IEFJESCT	- JES Communication Table
IEFSWB	- Scheduler Work Block

SERIALIZATION:

No locks or resources are obtained by this module.

IEFSJPUT - MODULE OPERATION

This module rebuilds a SWB chain from SWB keyword data found in text unit format. It performs the following functions:

1. Gets the address of the SWB chain by using the IEFSJSWA procedure to interpret the SWB token.

2. Verifies that the parameter list (IEFSJPUP) contains valid entries:
   a. Checks to make sure that the IEFSJSWA procedure successfully verified the SWB token and found an address to a valid SWB. If not, sets register 15 to 4, sets a reason code of SJRCIVTK (2) in SJPUREAS, and returns.

   b. Checks that the address of the keyword data area was specified (SJPUSWBA). If this field is zero, sets register 15 to 4, sets a reason code of SJRCPSWB (900) in SJPUREAS, and returns.

   c. Checks that the length of the keyword data area was specified. If this field is zero, sets register 15 to 4, sets a reason code of SJRCPLEN (902) in SJPUREAS, and returns.

3. Verifies that the verb and label of the SWB chain matches the verb and label of the NJE prefix found in the keyword data area. If the verb and label do not match, sets register 15 to 4, sets reason code of SJRCIVTK (2) in SJPUREAS, and returns.

4. While processing in the caller's key, counts the number of text units found in the keyword data area. IEFSJPUT uses the total number of text units to determine the amount of storage needed for the text unit pointer list.

5. If no text unit data was found in the keyword data area, invokes the SJF write SWB routine to build and chain a SWB. Otherwise, obtains storage for the text unit pointer list. If the GETMAIN was unsuccessful, sets register 15 to 4, sets a reason code of SJRCPUGM (901) in in SJPUREAS, and returns.

6. Builds the text unit pointer list by scanning the keyword data area and storing a pointer to each text unit into the text unit pointer list.

7. Fills in the SJF update parameter list (IEFSJRUP) and invokes SJF update to build and chain a SWB chain containing the information in the text units in the keyword data area. If the SJF update routine had any problems while processing the request, sets the return code and the reason code to SJF update's return code and reason code.

8. Returns to caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (PUTRETRY) in the SJF control workarea. PUTRETRY (in this module) receives control from RTM, it does the following:

IEFSJPUT - MODULE OPERATION (Continued)

1. Sets the return code to indicate an SJF system error.

2. Determines if any storage has been obtained, and frees it if it has not already been freed.

3. Returns to caller.

IEFSJPUT - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJPUT
PUTRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJPUT:

EXIT NORMAL:

```
    Register 15 = 0 - Processing successful

  Reason codes in SJPUREAS =
   SJRCNOER (0) - Processing successful
```

EXIT ERROR:

```
    Register 15 = 4 - Request cannot be processed

  Reason codes in SJPUREAS =
    SJRCIVTK (2)   - Invalid SWB chain address
    SJRCPSWB (900) - Address of keyword data
                     area to be put not
                     specified
    SJRCPUGM (901) - Unable to obtain storage
                     for the text unit pointer and
                     for a local copy of the keyword
                     data area
    SJRCPLEN (902) - Length of keyword data area not
                     specified
```

ENTRY POINT PUTRETRY:

EXIT ERROR:

```
  Register 15 = 20 - SJF system error
```

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJPUT:

```
Register  0     = Undefined
Register  1     = Address of a two word parameter
                  list. The first word contains the
                  address of the Put SWB parameter
                  list (IEFSJPUP) and the second
                  word contains the address of
                  the SJF control workarea (IEFSJCNW)
Registers 2-12 = Undefined
Register  13   = Address of an 18 word savearea
Register  14   = Return address
Register  15   = Entry point address
```

ENTRY POINT PUTRETRY:

```
Register  0    = Undefined
Register  1    = Address of ESTAE parameter list
```

IEFSJPUT - DIAGNOSTIC AIDS (Continued)

```
Registers 2-14 = Undefined
Register  15   = Return code
```

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJPUT:

```
Registers 0-14 = Restored
Register    15 = Return code
```

ENTRY POINT PUTRETRY:

```
Registers 0-14 = Restored
Register    15 = Return code
```

IEFSJRET - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Retrieve Routine

FUNCTION:

This module retrieves parameter information from a scheduler work block (SWB) chain, associated with a keyword or keywords for a particular verb and label, and uses that information to build text units.

ENTRY POINT: IEFSJRET

PURPOSE: See Function

LINKAGE: CALL

CALLERS: SJF control routine (IEFSJCNL)

INPUT:

SJF Retrieve Parameter List (IEFSJREP)

FIELD	LENGTH/MASK	DESCRIPTION
SJREP	48	Retrieve parameter list
SJREID	4	Identifier 'SJRE'
SJREVERS	1	Version number
SJREFLAG	1	Control flags
SJRENREC	X'80'	No recovery
SJRENOCU	X'40'	No cleanup
SJRELEN	2	Length of the parameter list
SJRESTOR	4	Local storage pointer or zero
SJREREAS	4	Reason code (returned)
SJREJDVT	8	Name of JDVT or zeroes
SJRETOKN		SWB chain token
SJREANBK	4	Address of anchor control block or of the first control block for a JCL statement
SJREANCA	4	Address of word pointing to SWB chain or zero
SJREAREA	4	Storage area address
SJRESIZE	2	Size of storage area
SJRENKWD	2	Number of keywords passed
SJREKWDL	4	Keyword list address
SJREKERR	4	Address of keyword causing error (returned)

SJF Retrieve Keyword List (Pointed to by SJREKWDL)

FIELD	LENGTH/MASK	DESCRIPTION
SJRELIST (*)	12	Keyword list
SJREKEYW	8	Keyword for retrieve
SJRETPAD	4	Address of a list of text unit pointers (returned)

OUTPUT:

SJF Retrieve Parameter List (IEFSJREP)

FIELD	LENGTH/MASK	DESCRIPTION
SJREREAS	4	Reason code
SJREKERR	4	Address of keyword causing the error or zero

SJF Retrieve Keyword List

FIELD	LENGTH/MASK	DESCRIPTION

IEFSJRET - MODULE DESCRIPTION (Continued)

SJRETPAD (*)	4	Address of a list of text pointers or zero

EXIT NORMAL: Return to caller.

EXIT ERROR: Return to caller.

ENTRY POINT: RETRETRY

PURPOSE:
Performs cleanup processing when an ABEND occurs during SJF Retrieve's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES: IEFSJEXT - SJF Extract Routine

DATA AREAS:
IEFSJCNW - SJF Control Work Area
IEFSJEXP - SJF Extract Parameter List
IEFSJRC - SJF Reason Codes
IEFSJREP - SJF Retrieve Parameter List
IEFSJSWP - IEFSJSWA Parameter List
IEFZB502 - SWA Prefix
IEFZB505 - SWA Manager Extended External Paramater Area

CONTROL BLOCKS:
CVT - Communication Vector Table
JCT - Job Control Table
JCTX - Job Control Table Extension
JESCT - JES Communication Table
SCT - Step Control Table
SIOT - Step Input/Output Table
SWB - Scheduler Work Block

IEFSJRET - MODULE OPERATION

This module receives control when a retrieve request is issued to the scheduler JCL facility. It does the following:

1. Checks the validity of the caller's parameter list values. If an error is detected, sets the reason code (SJREREAS) and return code, then returns.

2. Sets to zero the caller's storage area and address fields (SJRETPAD) in the keyword list passed by the caller.

3. Initializes the SJF extract parameter list fields.

4. For each keyword in the keyword list (SJREKEYW), does the following:

   a) Places the keyword in the extract parameter list (SJEXKEYW).

   b) Invokes the SJF extract routine (IEFSJEXT) to obtain JCL definition table (JDT) information about the keyword. (See IEFSJEXT for a description of the information returned.)

   c) If the SJF extract routine had an error, examines its reason code (SJEXREAS). If a nonzero reason code is returned by the SJF extract routine, copies the reason code into the retrieve parameter list (SJREREAS), sets the address of the keyword causing the error (SJREKERR) in the parameter list, sets the return code and returns.

   d) If the SJF extract routine was successful, sets a pointer to the first subparameter/ sublist element definition in the JDT for the keyword.

   e) For each subparameter/sublist element of each keyword, does the following:

      . Searches the SWB chain to find the SWB identifier (owner name and block ID specified in the JDT). If the SWB was found:

        - Calculates the space required to build the text unit and the text unit pointer list

        - If there is not enough space to store the text unit data, copies the text unit pointers created thus far into the caller's area, sets the reason code (SJREREAS), sets the return code, sets the address of the keyword causing the error (SJREKERR), and returns.

        - Checks the validity flag associated with the subparameter /sublist element in the SWB.

        - If the validity flag is on,

IEFSJRET - MODULE OPERATION (Continued)

copies the subparameter/sublist element data into the text unit and sets the length field in the text unit.

- If the validity flag is off and the subparameter/ /sublist element is not a list item, sets the length of the subparameter/sublist element data to zero in the text unit. If the subparameter is a list item, stops processing this keyword. If the sublist element is a list item, stops processing the sublist element and starts processing the next subparameter.

. If the specified SWB was not found and the parameter is not a list item, sets the length of the subparameter/ sublist element to zero in the text unit. If the parameter is a list item, stops processing this keyword. If the sublist element is a list item, stops processing the sublist element and starts processing the next subparameter.

. If another subparameter/sublist element definition exists, sets a pointer to that definition in the JDT.

f) If no data was found for the keyword, sets the text unit address in the keyword list (SJRETPAD) to zero. If data was found for the keyword, copies the text unit pointer list from the retrieve work area to the caller's area.

g) Zeroes the retrieve work area.

5. Continues with the next keyword in the keyword list.

6. Returns to the caller.

RECOVERY OPERATION:

If an ABEND occurs in this module, the SJF control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (RETRETRY) in the SJF control workarea. When RETRETRY (in this module) receives control from RTM it does the following:

1. Sets the return code to indicate an SJF system error.

2. Returns to the caller.

IEFSJRET - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJRET
RETRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJRET:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully.

EXIT ERROR:

Register 15 = 4 - Request was not processed.

Reason codes in SJREREAS:

SJRCIVTK	(2)	- Invalid SWB token
SJRCNJDV	(4)	- JDVT not found
SJRCNJCH	(5)	- JDVT chain does not exist
SJRCNVRB	(200)	- Verb not found in JDT (See Note 1)
SJRCNKWD	(201)	- Keyword not found in JDT (See Note 1)
SJRCSTRS	(600)	- Not enough space in data area
SJRCWSPC	(601)	- Not enough space for text unit pointer list in work area
SJRCSTRA	(603)	- No address specified for storage area
SJRCIVKN	(604)	- Zero specified for number of keywords
SJRCIVKL	(605)	- No keyword list address specified
SJRCIVKW	(607)	- Keyword not specified

Note 1: These reason codes are generated by SJF Extract (IEFSJEXT) and caused by user error. See the Extract module for additional Extract reason codes.

ENTRY POINT RETRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJRET:

Register 0	= Undefined
Register 1	= Address of two words that contain the address of the input parameter list (IEFSJREP) and the address of the control work area (IEFSJCNW).
Registers 2-12	= Undefined
Register 13	= Address of an 18-word save area
Register 14	= Return address
Register 15	= Entry point address

ENTRY POINT RETRETRY:

IEFSJRET - DIAGNOSTIC AIDS (Continued)

Register 1 = Address of ESTAE parameter list
Registers 0,2-14 = Undefined
Register 15 = Entry point address

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJRET:

EXIT ERROR:

Register 0 = Undefined
Register 1 = Address of two words that contain the address of the input parameter list (IEFSJREP) and the address of the control work area (IEFSJCNW).
Registers 2-12 = Undefined
Register 13 = Address of an 18-word save area
Register 14 = Return address
Register 15 = Return code

ENTRY POINT RETRETRY:

EXIT ERROR:

Registers 0-14 = Restored
Register 15 = Return code

IEFSJRTE - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Router Routine

FUNCTION:

This module provides an addressing mode interface between the issuer of the SJFREQ macro and the Scheduler JCL Facility control routine (IEFSJCNL).

ENTRY POINT: IEFSJRTE

PURPOSE: See function

LINKAGE:
Standard PLS linkage via SJFREQ macro
. Entry address is in JESCT at location JESSJCNL

CALLERS: Issuers of the SJFREQ macro

INPUT:
IEFSJRTE passes on the input parameter list to IEFSJCNL. There is a different input parameter list for each SJF function.

OUTPUT: None

EXIT NORMAL: Exit to IEFSJCNL via BSM

EXTERNAL REFERENCES:

ROUTINES: IEFSJCNL - SJF Control Routine

CONTROL BLOCKS:

CVT	- Communications Vector Table
JESCT	- Job Entry Subsystem Communication Table

IEFSJRTE - MODULE OPERATION

This module passes control to the Scheduler JCL Facility control routine (IEFSJCNL) in 31-bit addressing mode in the following manner:

1. Saves the addressing mode of the caller in register 14.
2. If the caller is running in 24-bit addressing mode, clears the high order byte of register 13 (save area register).
3. If the caller is running in 31-bit addressing mode, clears only the high order bit of register 13.
4. Gets the address of IEFSJCNL from the JESCT.
5. Sets the addressing mode for IEFSJCNL and branches to IEFSJCNL.

IEFSJRTE - DIAGNOSTIC AIDS

ENTRY POINT NAME: IEFSJRTE

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES: None

REGISTER CONTENTS ON ENTRY:

Register 0	= Requested function mask
Register 1	= Address of a word that contains the address of the input parameter list
Register 2-12	= Irrelevant
Register 13	= Address of an 18-word savearea
Register 14	= Return address to issuer of SJFREQ
Register 15	= Entry point address of this module

REGISTER CONTENTS ON EXIT:

Register 0-12	= Unchanged
Register 13	= If the caller is running in 24-bit mode, the high order byte is zeroed out. If the caller is running in 31-bit mode, the high order bit is zeroed out.
Register 14	= Return address to issuer of SJFREQ
Register 15	= Entry point address of IEFSJCNL

IEFSJUPD - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Update Routine

FUNCTION:
This module verifies the text units specified by the caller and if requested, updates the SWB chain with the information specified in text unit format.

ENTRY POINT: IEFSJUPD

PURPOSE: See Function

LINKAGE: CALL

CALLERS:
SJF control routine (IEFSJCNL)
SJF put SWB routine (IEFSJPUT)

INPUT:
SJF Update Parameter List (IEFSJRUP)

FIELD	LENGTH/MASK	DESCRIPTION
SJRUP	56	Control parameter list
SJRUID	4	Identifier 'SJRU'
SJRUVERS	1	Version number
SJRUFLAG	1	Control flags
SJRUNREC	X'80'	No recovery
SJRUNOCU	X'40'	No cleanup
SJRULEN	2	Length of parameter list
SJRUSTOR	4	Local storage pointer or zero
SJRUREAS	4	Reason code (returned)
SJRUINPT	4	Pointer to the list of text unit pointers
SJRUJDVT	8	Name of JDVT or zeros
SJRUVERB	8	Verb
SJRULABL	8	Label
SJRUTOKN	8	SWB chain token
SJRUANBK	4	Address of anchor control block or of the first control block for a JCL statement
SJRUANCA	4	Address of word pointing to SWB chain or zero
SJRUFUNC	1	Flag field
SJRUSYST	X'80'	System input
SJRUNSWA	X'40'	Request for a non-SWA SWB
SJRUVERF	X'20'	Verification only
SJRUNREF	X'10'	Do not check reference
SJRUCONT	X'08'	Continuation text unit
SJRUJRNL	X'04'	Journaling requested
SJRUWARN	X'02'	Continue processing after an ignorable error is encountered. Ignorable errors are due to changes in the JDTs from release to release
SJRUDYNS	X'01'	Request is for a dynamic SWB chain
SJRUPARM	1	Number of parameters already processed in the first text unit
SJRUERRK	2	Key in error (returned)

The input to this module also includes the SJF control workarea (IEFSJCNW).

IEFSJUPD - MODULE DESCRIPTION (Continued)

OUTPUT:
SJF Update Parameter List (IEFSJRUP)

FIELD	LENGTH	DESCRIPTION
SJRUREAS	4	Reason code
SJRUERRK	2	Key in error

The SWB chain is updated, if requested.

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: UPDRETRY

PURPOSE:
Performs cleanup processing when an abend occurs during the SJF update routine's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES:
IEFSJDEL - SJF Delete SWB
IEFSJEXT - SJF JDT Extract
IEFSJFND - SJF Find SWB
IEFSJWRT - SJF Write SWB
IEFXB501 - Journal Write Routine

DATA AREAS:
IEFSJCNW - SJF Control Work Area
IEFSJDLP - SJF Delete SWB Parameter List
IEFSJEXP - SJF JDT Extract Parameter List
IEFSJFNP - SJF Find SWB Parameter List
IEFSJRC - SJF Reason Codes
IEFSJRUP - SJF Update Parameter List
IEFSJSWP - IEFSJSWA Parameter List
IEFSJWRP - SJF Write SWB Parameter List
IEFZB4D1 - Dynamic Allocation Text Unit
IEFZB507 - Journal Write Parameter List

CONTROL BLOCKS: None

IEFSJUPD - MODULE OPERATION

This module verifies the text units specified by the caller and if requested, updates the SWB chain with the information specified in text unit format.
It does the following:

1. If the request is to update the SWB chain, calls IEFSJSWA to validate the input SWB token and makes a copy of the SWB chain to be updated.

2. For each text unit specified, IEFSJUPD does the following:
   - Invokes the SJF extract routine (IEFSJEXT) to verify that the verb specified in the parameter list and the key specified in the text unit are defined in the JDT.
   - For each parameter in the text unit, invokes the SJF extract routine to retrieve the parameter information from the JDT, performs the checking specified in the JDT for this data type, and if the request is to update the SWB chain, invokes the SJF write SWB routine (IEFSJWRT) to update the SWB specified in the JDT with the parameter data.
   - If no parameter was specified in the text unit, invokes the SJF extract routine to retrieve the parameter information from the JDT and checks if a default value is specified in the JDT. If a default value is specified and the request is to update the SWB chain, invokes the SJF write SWB routine to update the SWB specified in the JDT with the default value.
   - If the warning indicator (SJRUWARN) in the parameter list is on, then processing will continue after an error in a text unit caused by changes in the JDTs from release to release. The erroneous text unit or text unit parameter is ignored and all valid information is stored in the SWB chain.
   - If the warning indicator (SJRUWARN) is off, processing will stop after the first error in a text unit and the SWB chain will not be updated.

3. If no errors have occurred and the request is to update the SWB chain, copies the updated SWB chain back into the original SWB chain.

4. If no errors have occurred and some SWBs have been updated, invokes the journal routine (IEFXB501) to journal the SWBs.

5. If an error was detected and some new SWBs were created, invokes the SJF delete routine (IEFSJDEL) to delete the SWBs that were newly built.

6. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (UPDRETRY) in the SJF control workarea. When UPDRETRY (in this module) receives control from RTM, it does the following:

1. Sets the return code to indicate an SJF system error.

2. If some new SWBs were created, invokes SJF delete (IEFSJDEL) to delete the SWBs that were newly built.

3. Frees the storage obtained to copy the SWBs and

IEFSJUPD - MODULE OPERATION (Continued)

the text units.

4. Returns to the caller.

IEFSJUPD - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJUPD
UPDRETRY

MESSAGES: None

ABEND CODES:

'054'X (84 decimal) with reason code '0E'X (14 decimal)
- indicates an invalid data type was encountered when checking the parameter.

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJUPD:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully

Reason codes in SJRUREAS
SJRCNOER (0) - Request completed successfully

The following non-zero reason codes may be returned if an error due to changes in the JDTs from release to release was detected and SJRUWARN bit is on.

SJRCNKEY (202) - Key not defined in JDT
SJRCNPRM (203) - Subparameter not defined in JDT
SJRCIVLN (500) - Invalid length of parameter
SJRCIVCH (501) - Invalid choice specified for parameter
SJRCGMAX (502) - Integer parameter exceeds maximum
SJRCLMIN (503) - Integer parameter less than minimum
SJRCNLLN (510) - Length of level exceeds the maximum
SJRCNLNM (511) - Number of levels exceeds the maximum
SJRCNFCH (512) - Invalid first character of level in parameter
SJRCNOCH (513) - Invalid character other than the first in level in parameter

EXIT ERROR:

Register 15 = 4 - Request was not processed

Reason codes in SJRUREAS:
SJRCIVID (1) - Invalid SWB ID
SJRCIVTK (2) - Invalid SWB token
SJRCNJDV (4) - JDVT not found
SJRCNJCH (5) - JDVT chain does not exist
SJRCNVRB (200) - Verb not defined in JDT
SJRCNKEY (202) - Key not defined in JDT
SJRCNPRM (203) - Subparameter not defined in JDT
SJRCNSCH (400) - Specified SWB chain not found (invalid referral)
SJRCSTEP (401) - Specified step or proc not found (invalid referral)
SJRCDDNM (402) - Specified DD label not found (invalid referral)
SJRCIVLN (500) - Invalid length of parameter

IEFSJUPD - DIAGNOSTIC AIDS (Continued)

SJRCIVCH (501) - Invalid choice specified for parameter
SJRCGMAX (502) - Integer parameter exceeds maximum
SJRCLMIN (503) - Integer parameter less than minimum
SJRCIVKY (504) - Invalid key, system specification only
SJRCDUPK (505) - Duplicate key
SJRCNNUM (506) - No parameter specified and no default defined
SJRCCOPY (507) - No storage could be obtained in which to update the SWBs
SJRCIVRB (508) - Verb not specified in the parameter list
SJRCIVLB (509) - Label not specified in the parameter list
SJRCNLLN (510)- Length of level exceeds the maximum
SJRCNLNM (511) - Number of levels exceeds the maximum
SJRCNFCH (512) - Invalid first character of level in parameter
SJRCNOCH (513) - Invalid character other than the the first in level in parameter
SJRCNLIV (514) - Invalid specification of level in parameter
SJRCIVRF (515) - Invalid specification of referral
SJRCIREF (517) - Invalid referral. This is due to a reference to a dynamic SWB chain outside of the current step

ENTRY POINT UPDRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJUPD:

Register 0 = Undefined
Register 1 = Address of two words that contain the address of the input parameter list and the address of the control work area.
Registers 2-12 = Undefined
Register 13 = Address of 18-word save area
Register 14 = Return address
Register 15 = Entry point address

ENTRY POINT UPDRETRY:

Register 0 = Undefined
Register 1 = Address of ESTAE parameter list
Registers 2-14 = Undefined
Register 15 = Entry point address

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJUPD:

Register 0 = Restored
Register 1 = Address of two words that contain the address of the input parameter list and the address of the

IEFSJUPD - DIAGNOSTIC AIDS (Continued)

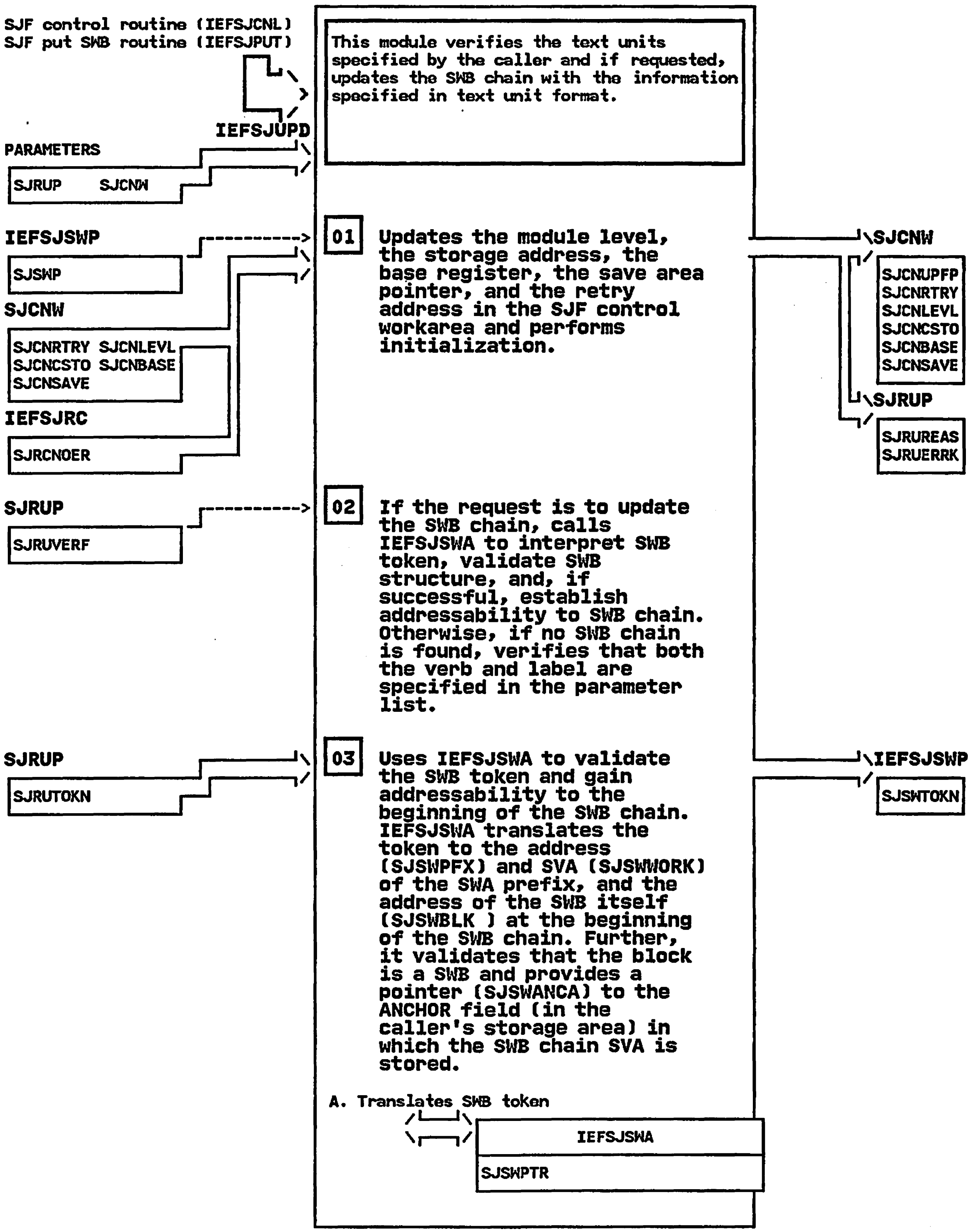
```
                  control work area.
Registers 2-12 = Restored
Register  13   = Address of 18-word save area
Register  14   = Return address
Register  15   = Return code
```

ENTRY POINT UPDRETRY:

```
Registers 0-14 = Restored
Register  15   = Return code
```

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine

STEP 01



This module verifies the text units specified by the caller and if requested, updates the SWB chain with the information specified in text unit format.

01 Updates the module level, the storage address, the base register, the save area pointer, and the retry address in the SJF control workarea and performs initialization.

02 If the request is to update the SWB chain, calls IEFSJSWA to interpret SWB token, validate SWB structure, and, if successful, establish addressability to SWB chain. Otherwise, if no SWB chain is found, verifies that both the verb and label are specified in the parameter list.

03 Uses IEFSJSWA to validate the SWB token and gain addressability to the beginning of the SWB chain. IEFSJSWA translates the token to the address (SJSWPFX) and SVA (SJSWWORK) of the SWA prefix, and the address of the SWB itself (SJSWBLK) at the beginning of the SWB chain. Further, it validates that the block is a SWB and provides a pointer (SJSWANCA) to the ANCHOR field (in the caller's storage area) in which the SWB chain SVA is stored.

A. Translates SWB token

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 03B

IEFSJSWP
SJSWRETC

IEFSJRC
SJRCNOER

IEFSJSWP
SJSWPFX

SWB
SWBVERB SWBVRBL

IEFSJSWP
SJSWRETC

IEFSJRC
SJRCNSWB

IEFSJRC
SJRCIVTK SJRCIVRB
SJRCIVLB

SJRUP
SJRUVERB SJRULABL

B. If a valid SWB chain isn't found, sets a return code and a reason code indicating that the SWB token is invalid. Otherwise, a SWB chain does not exist, so ensures that both verb and label are specified in the input parameter list. If this is the case, saves the verb and label and if not, sets the return code and reason code to indicate that the verb/label were not specified.

SJRUP
SJRUREAS

IEFSJSWP
SJSWPFX

04 Makes a copy of the SWB chain to be updated, the text unit pointer list, and text units in key 1 storage.

COPY

TEXTUNIT
TXTPLENT

05 For each text unit specified by the caller, does the following:

A. Verifies that the verb and key are defined in the JDTs.

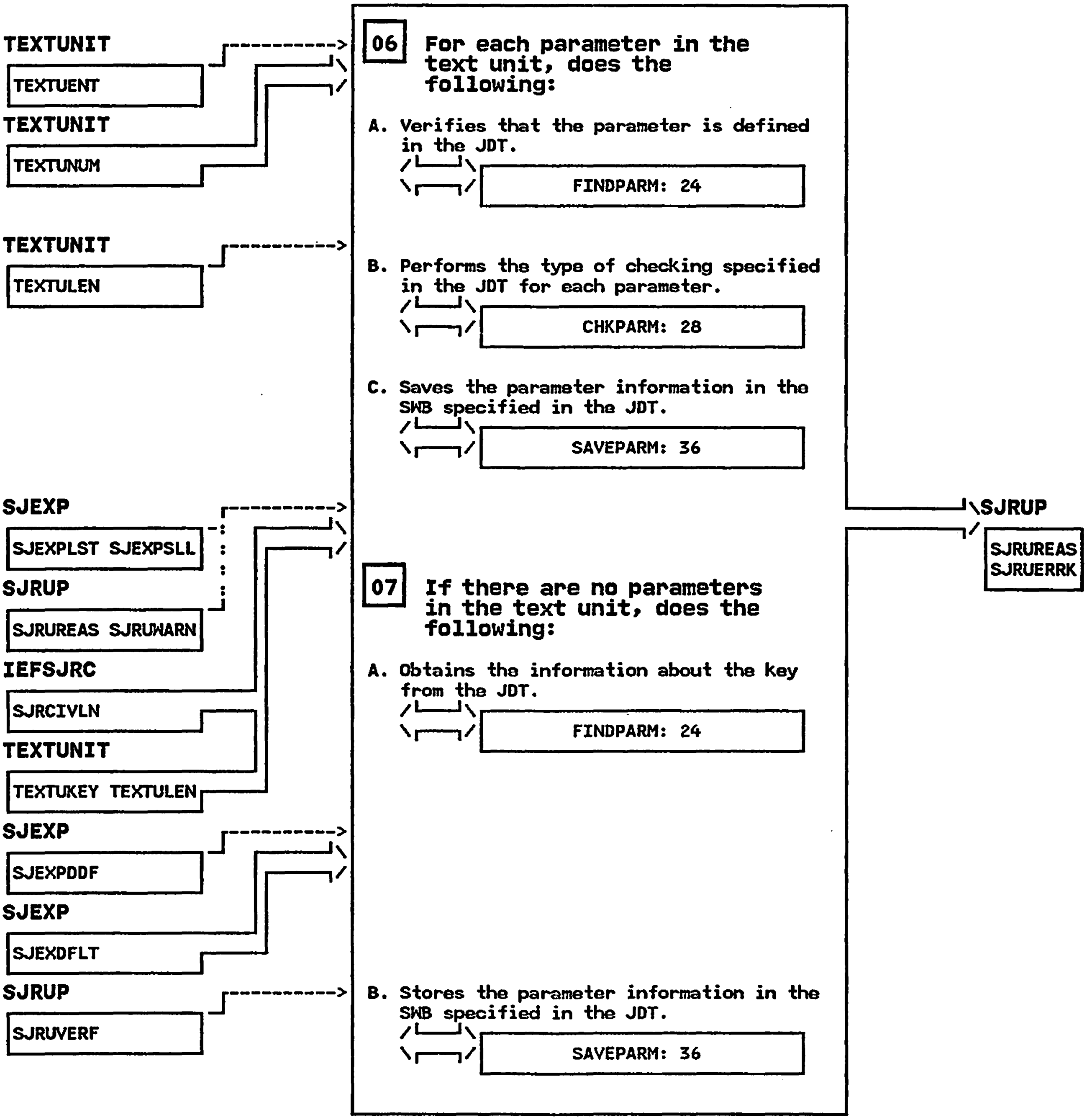
FINDKEY: 19

B. Verifies that the key has not already been specified in a previous text unit.

DUPKEYCK

TEXTUNIT
TEXTUNUM

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 06



06 For each parameter in the text unit, does the following:

A. Verifies that the parameter is defined in the JDT.

FINDPARM: 24

B. Performs the type of checking specified in the JDT for each parameter.

CHKPARM: 28

C. Saves the parameter information in the SWB specified in the JDT.

SAVEPARM: 36

07 If there are no parameters in the text unit, does the following:

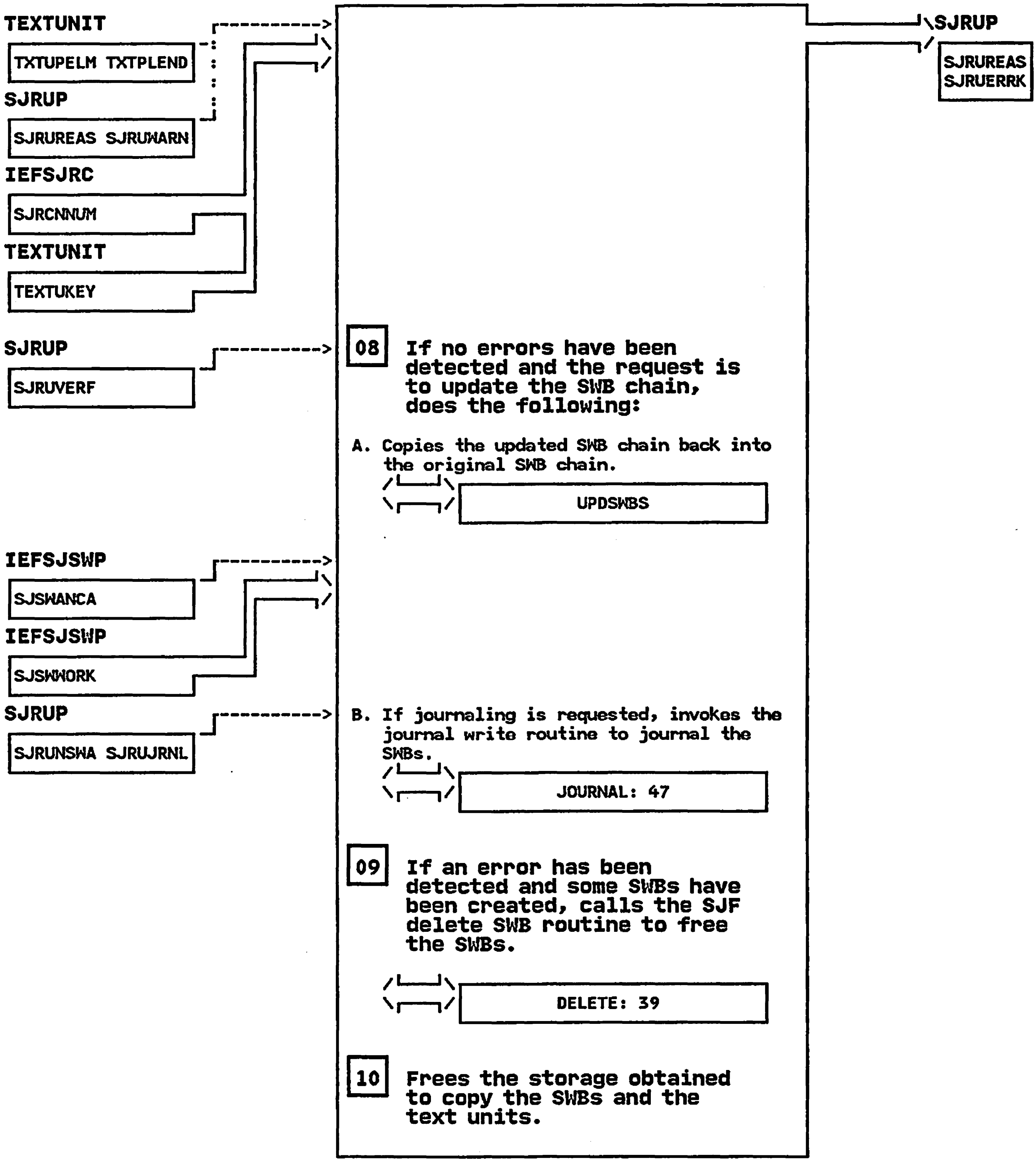
A. Obtains the information about the key from the JDT.

FINDPARM: 24

B. Stores the parameter information in the SWB specified in the JDT.

SAVEPARM: 36

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 08



08 If no errors have been detected and the request is to update the SWB chain, does the following:

A. Copies the updated SWB chain back into the original SWB chain.

B. If journaling is requested, invokes the journal write routine to journal the SWBs.

09 If an error has been detected and some SWBs have been created, calls the SJF delete SWB routine to free the SWBs.

10 Frees the storage obtained to copy the SWBs and the text units.

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 11

11 Restores the caller's module level, storage pointer, base register, save area pointer, and retry address in the SJF control workarea.

SJCNW

SJCNUPFP
SJCNRTRY
SJCNLEVL
SJCNCSTO
SJCNBASE
SJCNSAVE

12 Returns to the caller.

RTM

UPDRETRY

SJCNW

SJCNCSTO SJCNBASE
SJCNSAVE

13 If entering from RTM after an ABEND, restores the data register, the code register, and save area pointer and sets the return code to indicate a SJF system error.

14 Sets the return code to indicate a SJF system error.

15 If some SWBs have been created and the original SWB chain has not been updated, calls the SJF delete SWB routine to free the SWBs.

SJRUP

SJRUVERF

DELETE: 39

16 Frees the storage obtained to copy the SWBs and the text units.

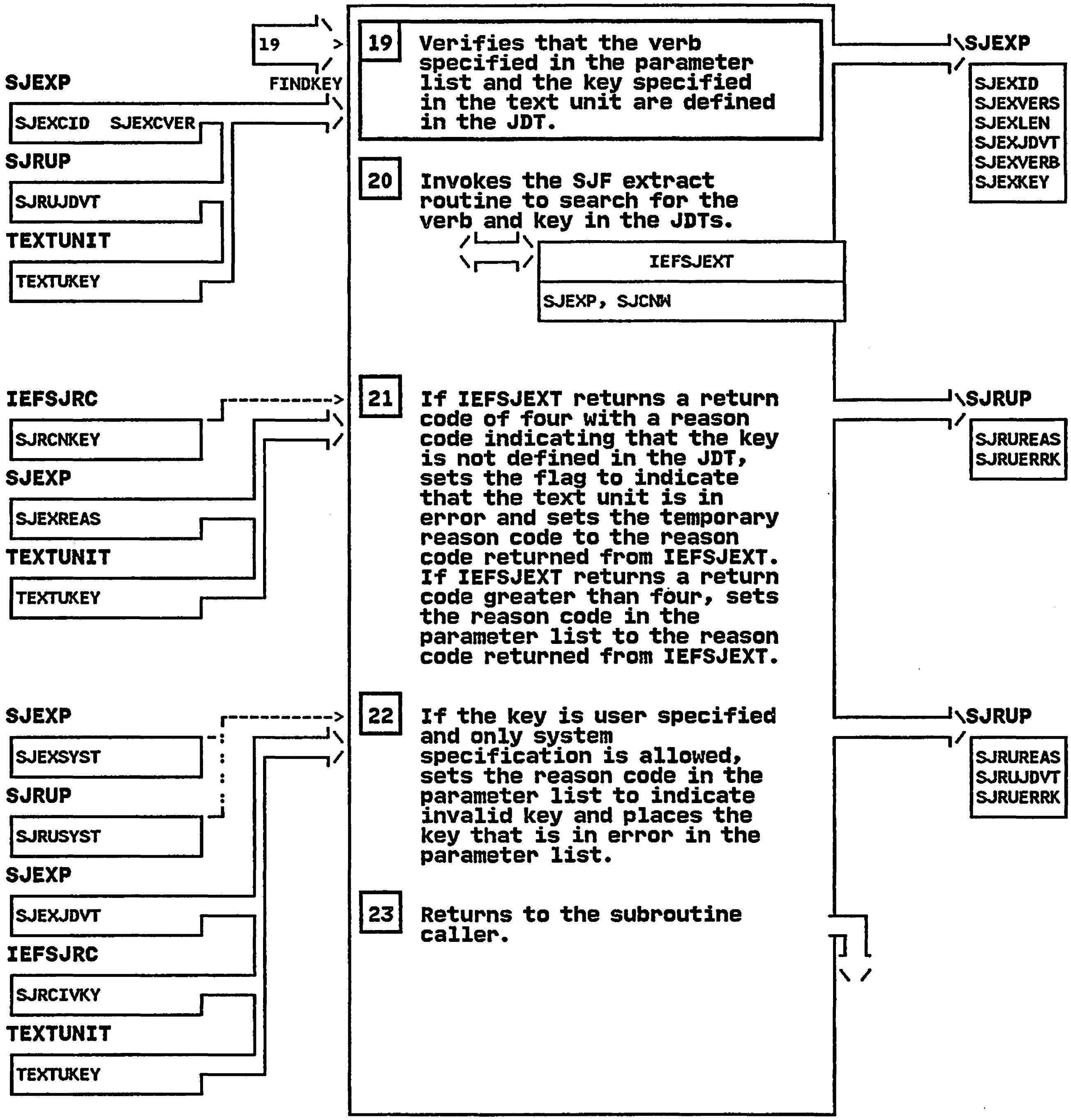
17 Restores the caller's module level, storage pointer, base register, save area pointer, and retry address in the SJF control workarea.

SJCNW

SJCNUPFP
SJCNRTRY
SJCNLEVL
SJCNCSTO
SJCNBASE
SJCNSAVE

18 Returns to the caller.

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 19



19 Verifies that the verb specified in the parameter list and the key specified in the text unit are defined in the JDT.

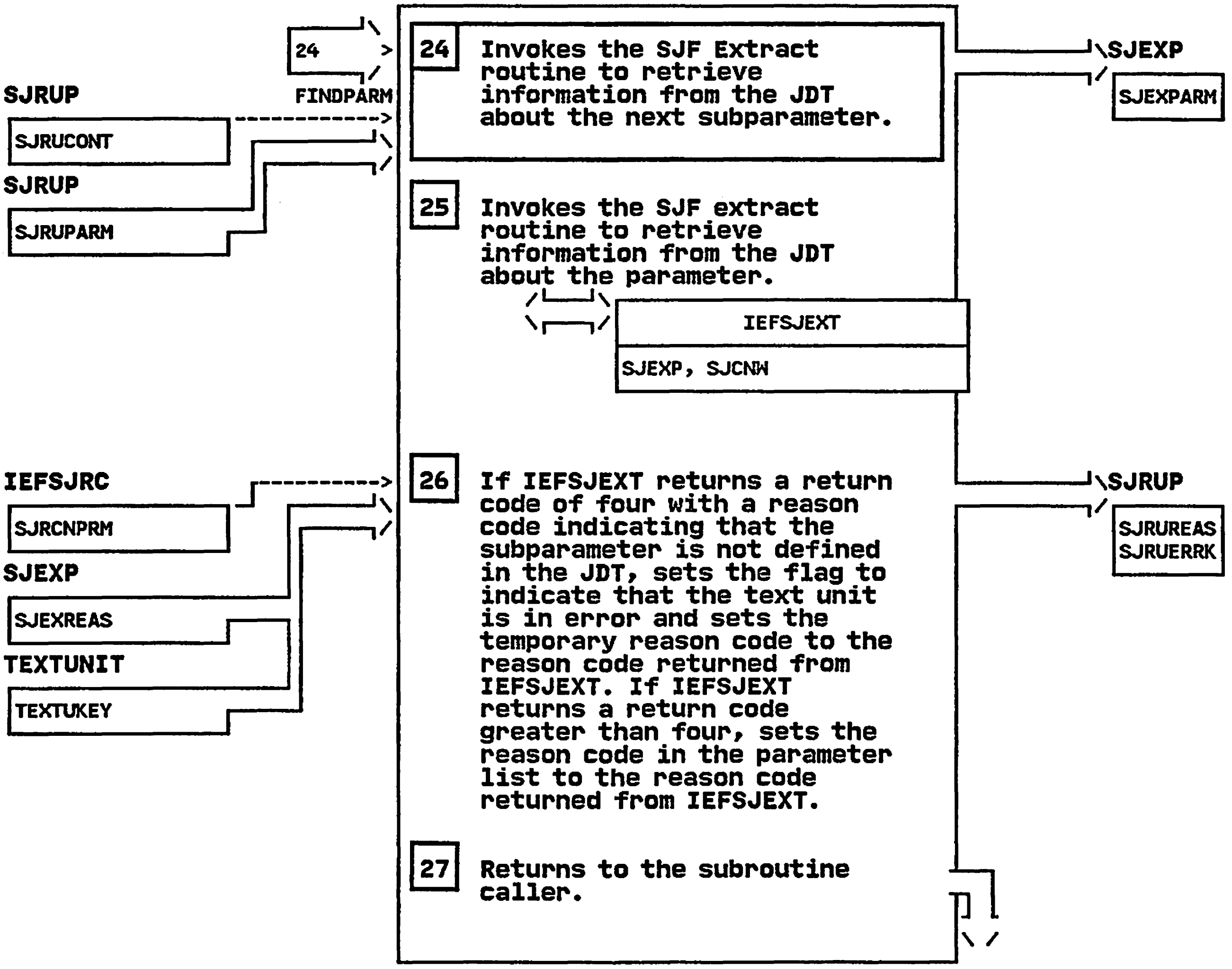
20 Invokes the SJF extract routine to search for the verb and key in the JDTs.

21 If IEFSJEXT returns a return code of four with a reason code indicating that the key is not defined in the JDT, sets the flag to indicate that the text unit is in error and sets the temporary reason code to the reason code returned from IEFSJEXT. If IEFSJEXT returns a return code greater than four, sets the reason code in the parameter list to the reason code returned from IEFSJEXT.

22 If the key is user specified and only system specification is allowed, sets the reason code in the parameter list to indicate invalid key and places the key that is in error in the parameter list.

23 Returns to the subroutine caller.

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 24

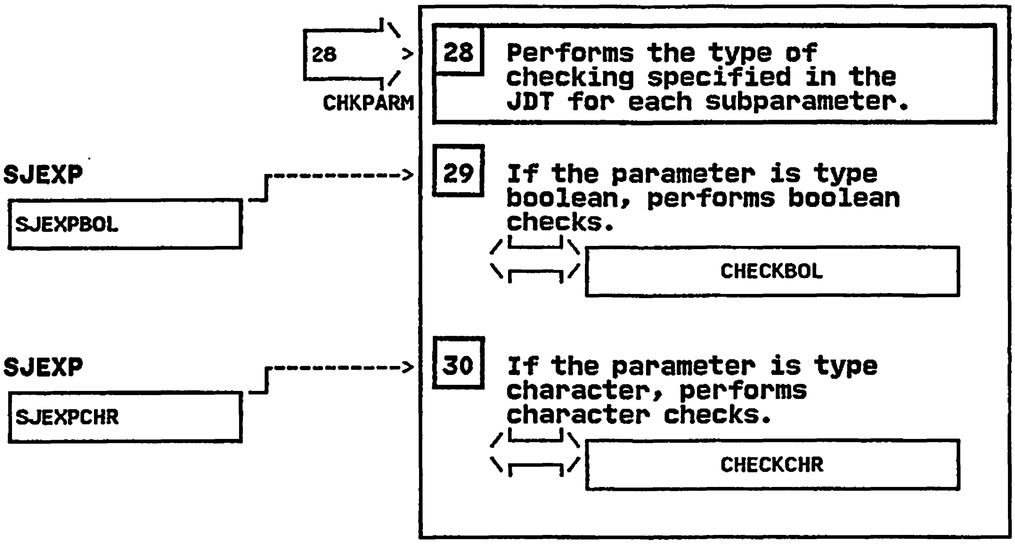


24 Invokes the SJF Extract routine to retrieve information from the JDT about the next subparameter.

25 Invokes the SJF extract routine to retrieve information from the JDT about the parameter.

26 If IEFSJEXT returns a return code of four with a reason code indicating that the subparameter is not defined in the JDT, sets the flag to indicate that the text unit is in error and sets the temporary reason code to the reason code returned from IEFSJEXT. If IEFSJEXT returns a return code greater than four, sets the reason code in the parameter list to the reason code returned from IEFSJEXT.

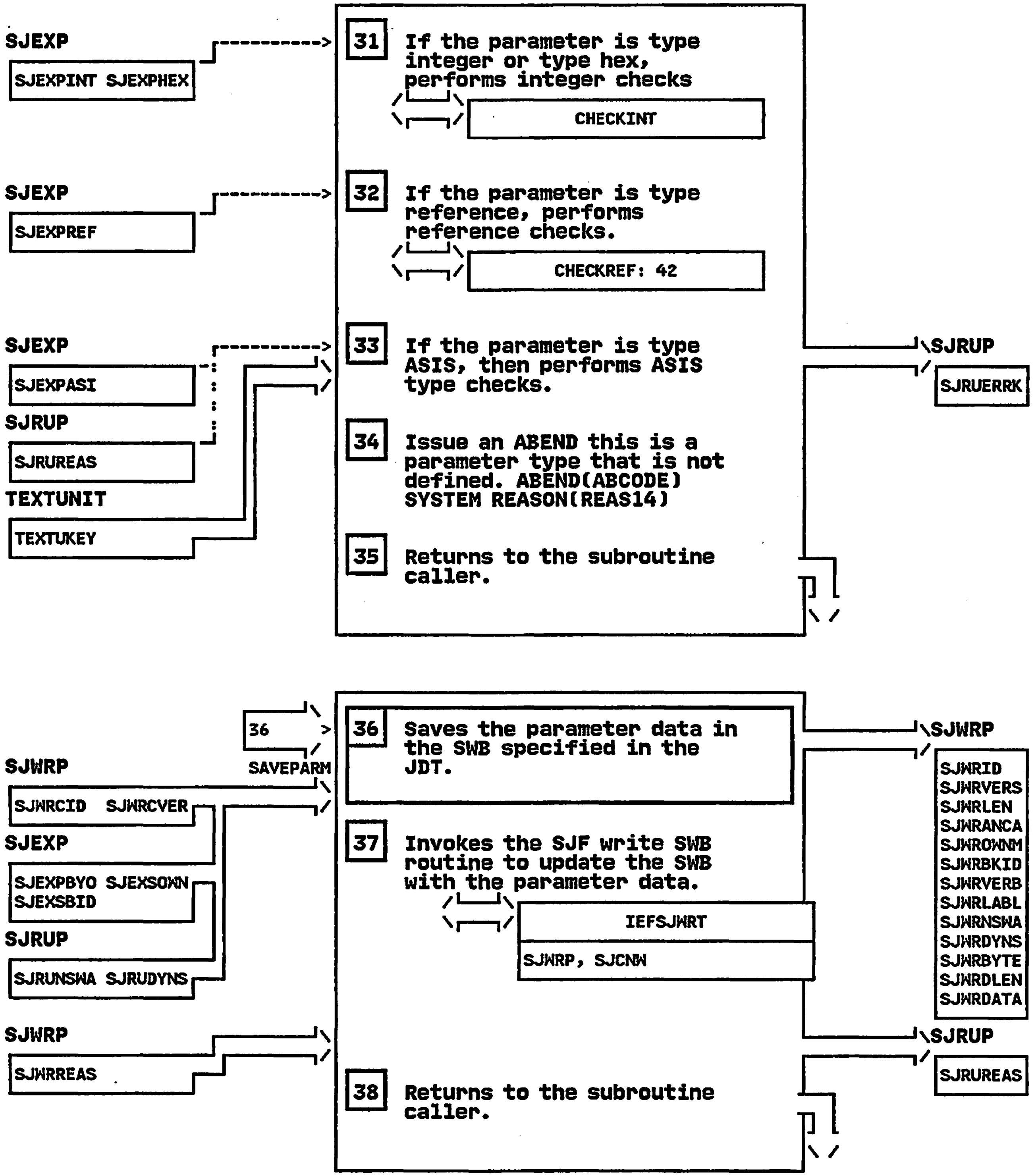
27 Returns to the subroutine caller.



28 Performs the type of checking specified in the JDT for each subparameter.

29 If the parameter is type boolean, performs boolean checks.

30 If the parameter is type character, performs character checks.



SJEXP
SJEXPINT SJEXPHEX

31 If the parameter is type integer or type hex, performs integer checks

CHECKINT

SJEXP
SJEXPREF

32 If the parameter is type reference, performs reference checks.

CHECKREF: 42

SJEXP
SJEXPASI
SJRUP
SJRUREAS
TEXTUNIT
TEXTUKEY

33 If the parameter is type ASIS, then performs ASIS type checks.

SJRUP
SJRUERRK

34 Issue an ABEND this is a parameter type that is not defined. ABEND(ABCODE) SYSTEM REASON(REAS14)

35 Returns to the subroutine caller.

36
SAVEPARM

SJWRP
SJWRCID SJWRCVER
SJEXP
SJEXPBYO SJEXSOWN
SJEXSBID
SJRUP
SJRUNSWA SJRUDYNS

36 Saves the parameter data in the SWB specified in the JDT.

SJWRP
SJWRID
SJWRVERS
SJWRLEN
SJWRANCA
SJWROWNM
SJWRBKID
SJWRVERB
SJWRLABL
SJWRNSWA
SJWRDYNS
SJWRBYTE
SJWRDLEN
SJWRDATA

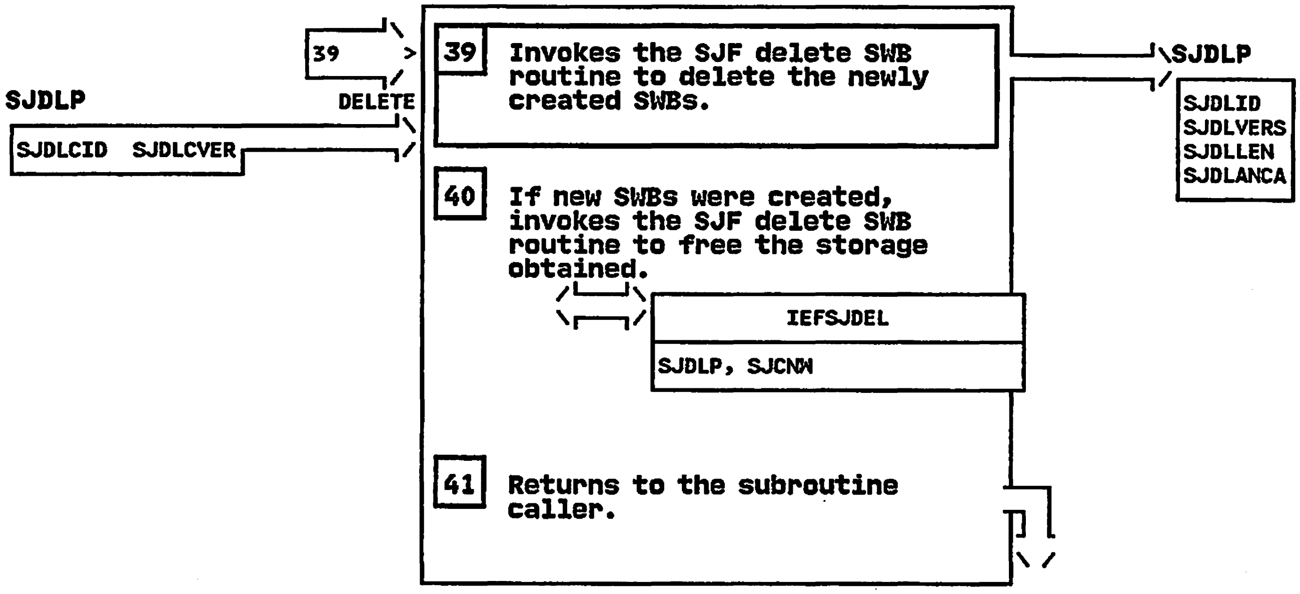
37 Invokes the SJF write SWB routine to update the SWB with the parameter data.

IEFSJWRT
SJWRP, SJCNW

SJWRP
SJWRREAS

SJRUP
SJRUREAS

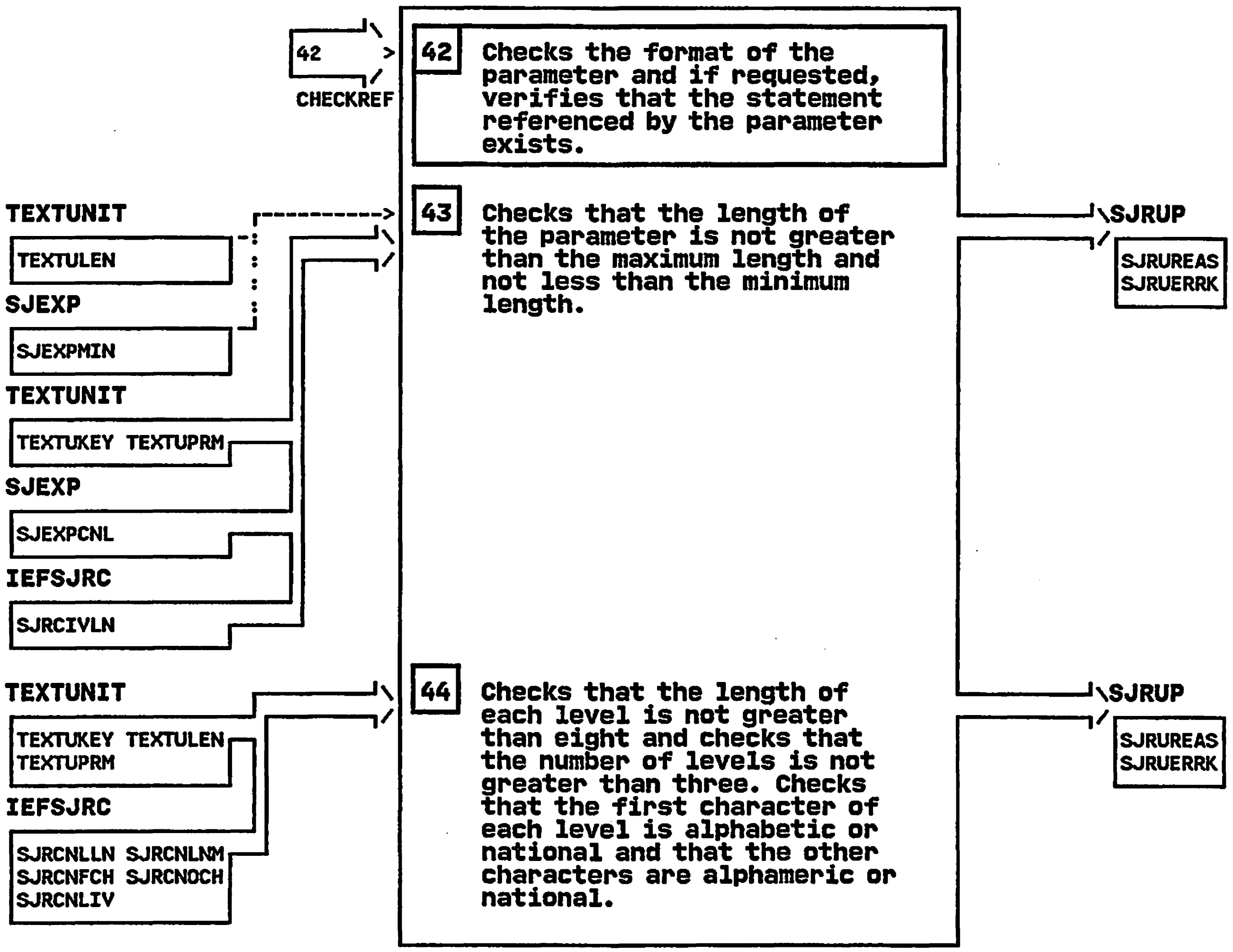
38 Returns to the subroutine caller.



39 Invokes the SJF delete SWB routine to delete the newly created SWBs.

40 If new SWBs were created, invokes the SJF delete SWB routine to free the storage obtained.

41 Returns to the subroutine caller.

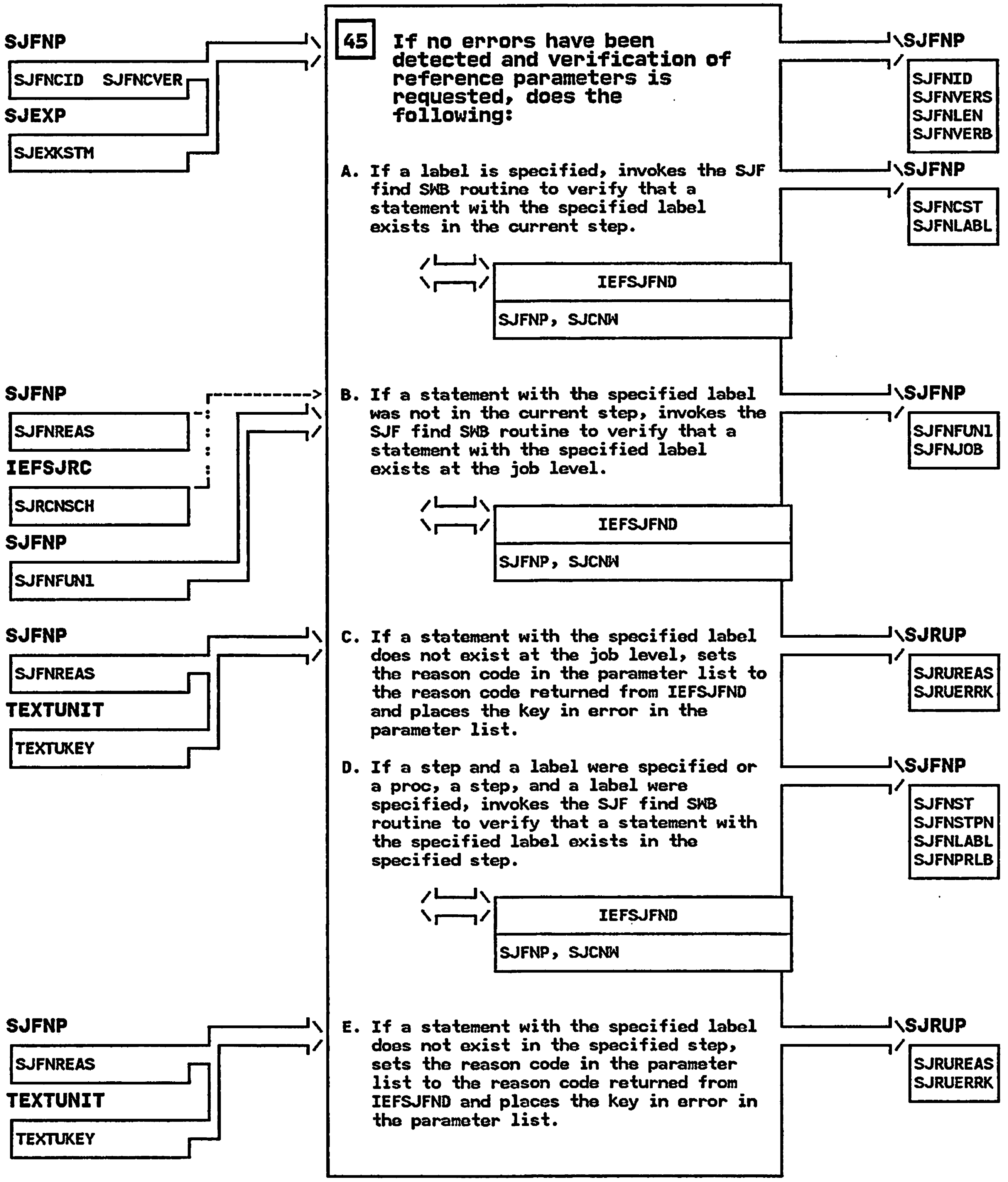


42 Checks the format of the parameter and if requested, verifies that the statement referenced by the parameter exists.

43 Checks that the length of the parameter is not greater than the maximum length and not less than the minimum length.

44 Checks that the length of each level is not greater than eight and checks that the number of levels is not greater than three. Checks that the first character of each level is alphabetic or national and that the other characters are alphameric or national.

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 45



45 If no errors have been detected and verification of reference parameters is requested, does the following:

A. If a label is specified, invokes the SJF find SWB routine to verify that a statement with the specified label exists in the current step.

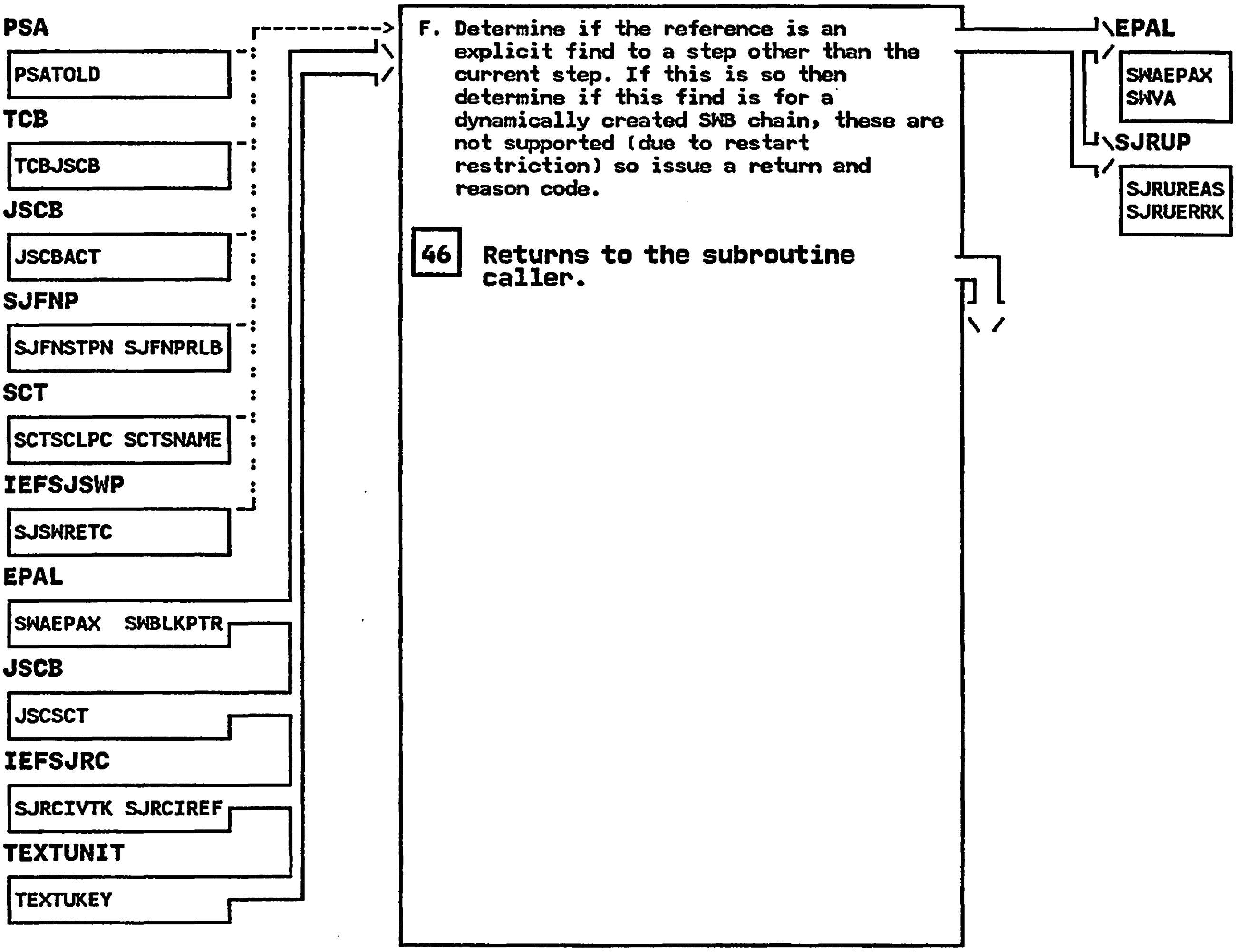
B. If a statement with the specified label was not in the current step, invokes the SJF find SWB routine to verify that a statement with the specified label exists at the job level.

C. If a statement with the specified label does not exist at the job level, sets the reason code in the parameter list to the reason code returned from IEFSJFND and places the key in error in the parameter list.

D. If a step and a label were specified or a proc, a step, and a label were specified, invokes the SJF find SWB routine to verify that a statement with the specified label exists in the specified step.

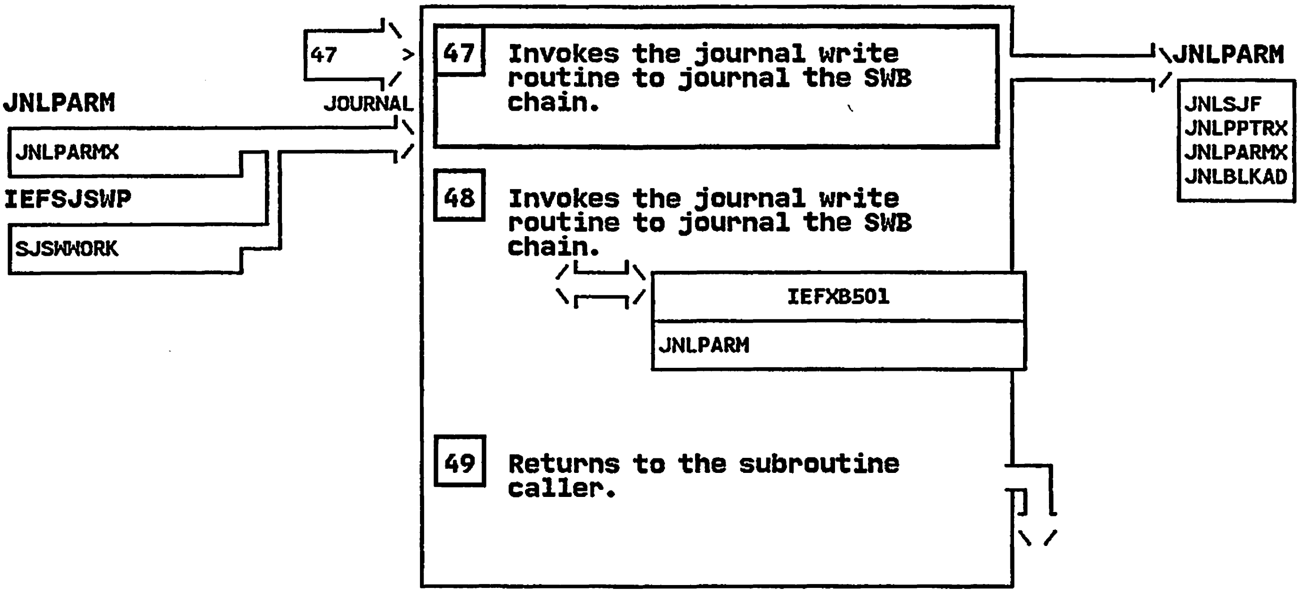
E. If a statement with the specified label does not exist in the specified step, sets the reason code in the parameter list to the reason code returned from IEFSJFND and places the key in error in the parameter list.

IEFSJUPD - Scheduler JCL Facility (SJF) Update Routine STEP 45F



F. Determine if the reference is an explicit find to a step other than the current step. If this is so then determine if this find is for a dynamically created SWB chain, these are not supported (due to restart restriction) so issue a return and reason code.

46 Returns to the subroutine caller.



47 Invokes the journal write routine to journal the SWB chain.

48 Invokes the journal write routine to journal the SWB chain.

49 Returns to the subroutine caller.

IEFSJVER - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Verify Routine

FUNCTION:

This module verifies the command, operand, and subparameters specified by thé caller and builds text units containing the subparameter information.

ENTRY POINT: IEFSJVER

PURPOSE: See Function

LINKAGE: CALL

CALLERS: SJF control routine (IEFSJCNL)

INPUT:

SJF Verify Parameter List (IEFSJVEP)

FIELD	LENGTH/MASK	DESCRIPTION
SJVEP	256	SJF Verify parameter list
SJVEID	4	Identifier 'SJVE'
SJVEVERS	1	Version number
SJVEFLAG	1	Function flags
SJVENREC	X'80'	No recovery
SJVENOCU	X'40'	No cleanup
SJVEUNAU	X'20'	Unauthorized caller
SJVELEN	2	Length of parameter list
SJVESTOR	4	Local storage pointer
SJVEREAS	4	Reason code (returned)
SJVEJDVT	8	Name of JDVT or zeroes
SJVECMND	8	Command
SJVEOPEP	4	Pointer to operand
SJVEOPEL	2	Length of operand
SJVEPARM	1	Subparameter number
SJVESUBL	1	Sublist element number
SJVEPRMP	4	Pointer to subparameter data
SJVEPRML	2	Length of subparameter data
SJVETUBL	2	Length of text unit buffer
SJVETUBP	4	Pointer to text unit buffer
SJVEFLG1	1	Verify Option flags
SJVELSTC	X'80'	Last call bit, used to remove null text units
SJVERSV1	3	Reserved
SJVETUPL	4	Pointer to text unit pointer list (returned)
SJVEOPD	64	Operand description (returned)
SJVEOPDL	2	Length of operand description (returned)
SJVEMSGL	2	Length of message information (returned)
SJVEMSG	120	Message information (returned)
SJVERSV2	8	Reserved

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:

SJF Verify Parameter List (IEFSJVEP)

FIELD	LENGTH	DESCRIPTION
SJVEREAS	4	Reason code
SJVETUPL	4	Pointer to text unit pointer list
SJVEOPD	64	Operand description
SJVEOPDL	2	Length of operand description

IEFSJVER - MODULE DESCRIPTION (Continued)

SJVEMSGL	2	Length of message information
SJVEMSG	120	Message information

A text unit pointer list and text units are built in the text unit buffer.

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: VERRETRY

PURPOSE:
Performs cleanup processing when an abend occurs during the SJF verify routine's processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES:
IEFSJEXT - SJF Extract
IEFSJMSG - SJF Message Module (data only)

DATA AREAS:
IEFSJCNW - SJF Control Work Area
IEFSJEXP - SJF Extract Parameter List
IEFSJRC - SJF Reason Codes
IEFSJVEP - SJF Verify Parameter List
IEFSJMSP - SJF Message Parameter List
IEFZB4D1 - Dynamic Allocation Text Unit Pointer List

CONTROL BLOCKS: None

SERIALIZATION: No serialization

IEFSJVER - MODULE OPERATION

This module verifies the command, operand, and subparameters specified by the caller and builds text units containing the subparameter information. It does the following:

1. If the last call option was specified (SJVELSTC) then check the last text unit to determine if it is null. If the last text unit is null then it is removed and Verify related parameters in the SJF Control Work area are zero'd.

2. If a command was specified with no operand:

   A. Invokes the SJF extract routine (IEFSJEXT) to verify that the command specified in the parameter list is defined in the JDT.

   B. If the SJF extract is not successful, sets the return code and the reason code to the return code and reason code returned from the SJF extract.

3. If a command with an operand and no subparameter was specified:

   A. Invokes the SJF extract routine (IEFSJEXT) to verify that the command and operand specified in the parameter list are defined in the JDT.

   B. If the SJF extract is not successful, sets the return code and the reason code to the return code and reason code returned from the SJF extract.

4. If a command with an operand and a subparameter was specified:

   A. Invokes the SJF extract routine (IEFSJEXT) to verify that the command and operand specified in the parameter list are defined in the JDT and to retrieve the subparameter information from the JDT.

   B. If the SJF extract is not successful, sets the return code and the reason code to the return code and reason code returned from SJF extract, and puts the error message information in the parameter list.

   C. If there are no errors, performs the checking specified in the JDT for this subparameter.

   D. If there are no errors, builds a text unit containing the subparameter information or adds on to an existing text unit. For each new text unit, adds a pointer to the text unit pointer list. If a duplicate text unit key exists, the new text unit will replace the previous one.

   E. If there is an error in the parameter, builds message text containing the parameter rules.

5. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the SJF control routine's recovery (entry point RECOVERY in IEFSJCNL) receives control from RTM. The recovery routine specifies to RTM the retry address (VERRETRY) in the SJF control workarea. When VERRETRY (in this module)

IEFSJVER - MODULE OPERATION (Continued)

receives control from RTM, it does the following:

1. Sets the return code to indicate a SJF system error.
2. Returns to the caller.

IEFSJVER - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJVER
VERRETRY

MESSAGES: None

ABEND CODES:

'054'X (084 decimal) - SJF system error abend

Reason code - 7 - Text unit buffer full
8 - Invalid data type during SJF Verify data checking
9 - Invalid data type during SJF Verify message building

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJVER:

EXIT NORMAL:

Register 15 = 0 - Request completed successfully

Reason codes in SJVEREAS
SJRCNOER (0) - Request completed successfully

EXIT ERROR:

Register 15 = 4 - Request was not processed

Reason codes in SJVEREAS:
SJRCNJDV (4) - JDVT not found
SJRCNJCH (5) - JDVT chain does not exist
SJRCNPRM (203) - Subparameter not defined in JDT
SJRCNCMD (207) - Command not defined in JDT
SJRCNOPE (208) - Operand not defined in JDT
SJRCIVLN (500) - Invalid length of parameter
SJRCIVCH (501) - Invalid choice specified for parameter
SJRCGMAX (502) - Integer parameter exceeds maximum
SJRCLMIN (503) - Integer parameter less than minimum
SJRCNNUM (506) - No parameter specified and no default defined
SJRCNLLN (510)- Length of level exceeds the maximum
SJRCNLNM (511) - Number of levels exceeds the maximum
SJRCNFCH (512) - Invalid first character of level in parameter
SJRCNOCH (513) - Invalid character other than the first in level in parameter
SJRCNLIV (514) - Invalid specification of level in parameter
SJRCIHEX (515) - Characters other than hex specified
SJRCINUM (516) - Nonnumeric characters specified
SJRCIVCM (1200)- Command not specified
SJRCIVTP (1201)- No address specified for the text unit buffer
SJRCIVTL (1202)- Not enough storage in the text unit buffer

IEFSJVER - DIAGNOSTIC AIDS (Continued)

ENTRY POINT VERRETRY:

EXIT ERROR:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJVER:

Register 0	= Undefined
Register 1	= Address of two words that contain the address of the input parameter list and the address of the control work area.
Registers 2-12	= Undefined
Register 13	= Address of 18-word save area
Register 14	= Return address
Register 15	= Entry point address

ENTRY POINT VERRETRY:

Register 0	= Undefined
Register 1	= Address of ESTAE parameter list
Register 2-14	= Undefined
Register 15	= Entry point address

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJVER:

Register 0	= Restored
Register 1	= Address of two words that contain the address of the input parameter list and the address of the control work area.
Registers 2-12	= Restored
Register 13	= Address of 18-word save area
Register 14	= Return address
Register 15	= Return code

ENTRY POINT VERRETRY:

Registers 0-14	= Restored
Register 15	= Return code

IEFSJVER - Scheduler JCL Facility (SJF) Verify Routine   STEP 01

SJF control routine (IEFSJCNL)

IEFSJVER

PARAMETERS

SJVEP   SJCNW

IEFSJVEP

SJCNCSTO R11

SJEXP

SJVELSTC

This module verifies the command, operand, and subparameters specified by the caller and builds text units containing the subparameter information.

01 Updates the module level, the storage address, the base register, the save area pointer, and the retry address in the SJF control workarea and performs initialization.

02 If this is not the last call, then does the following:

03 If a command was specified, does the following:

A. Calls the routine to check if the command, or operand, or subparameter, as specified in the parameter list, is defined in the JDT.

B. Calls the routine to invoke SJF extract

FINDCOS: 14

C. If a subparameter number was specified, does the following:

D. Checks the validity of the subparameter.

E. Calls the routine to perform the checking specified in the JDT.

CHKPARM: 21

F. If no errors were found in the subparameter, does the following:

G. Calls the routine to build the text unit.

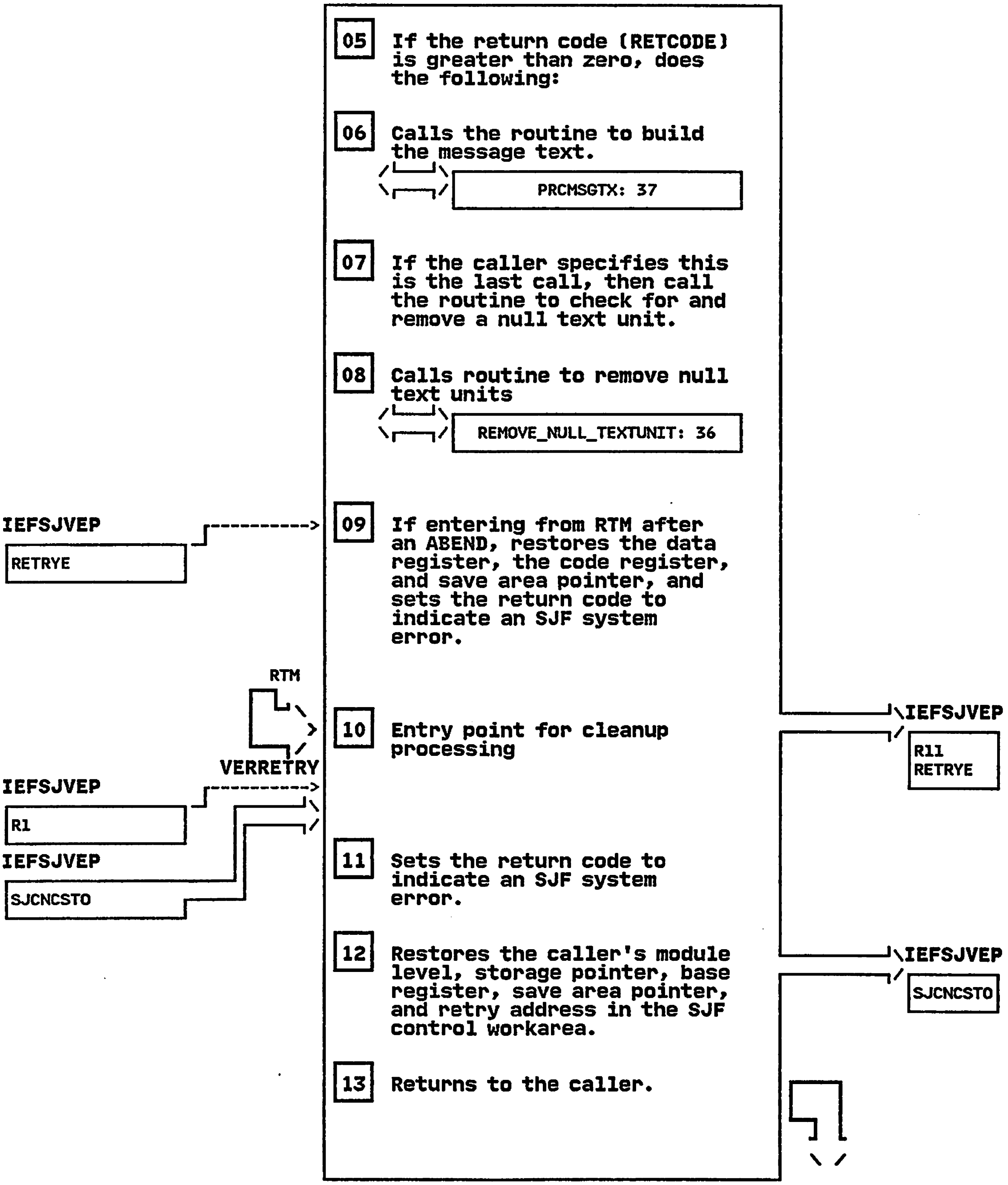
BLDTXUNT: 29

04 If no command was specified, sets the return code and reason code to indicate an error condition.

IEFSJVEP

SJCNCSTO

IEFSJVER - Scheduler JCL Facility (SJF) Verify Routine STEP 05



05 If the return code (RETCODE) is greater than zero, does the following:

06 Calls the routine to build the message text.

PRCMSGTX: 37

07 If the caller specifies this is the last call, then call the routine to check for and remove a null text unit.

08 Calls routine to remove null text units

REMOVE_NULL_TEXTUNIT: 36

09 If entering from RTM after an ABEND, restores the data register, the code register, and save area pointer, and sets the return code to indicate an SJF system error.

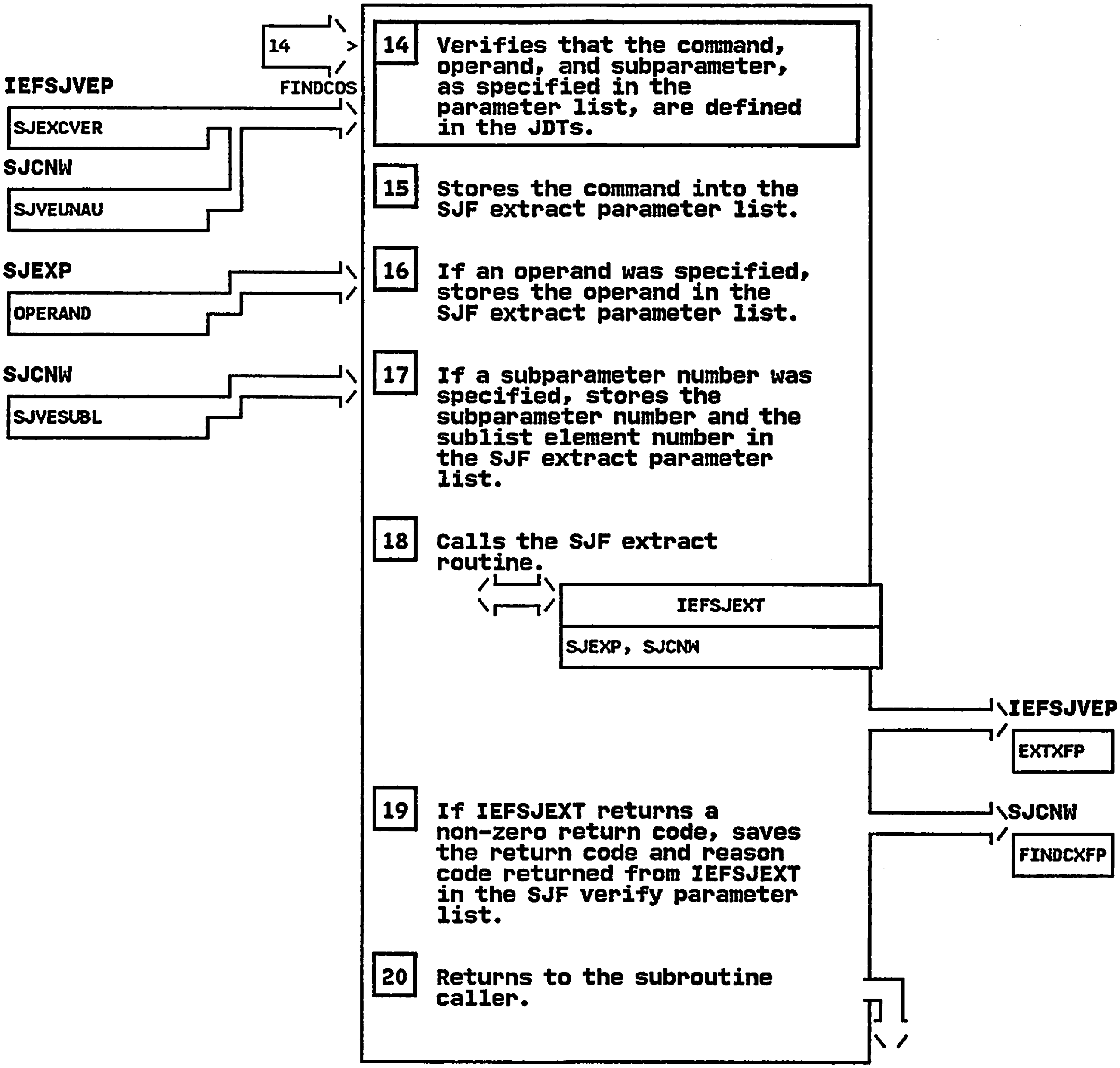
10 Entry point for cleanup processing

11 Sets the return code to indicate an SJF system error.

12 Restores the caller's module level, storage pointer, base register, save area pointer, and retry address in the SJF control workarea.

13 Returns to the caller.

IEFSJVER - Scheduler JCL Facility (SJF) Verify Routine STEP 14



14 Verifies that the command, operand, and subparameter, as specified in the parameter list, are defined in the JDTs.

15 Stores the command into the SJF extract parameter list.

16 If an operand was specified, stores the operand in the SJF extract parameter list.

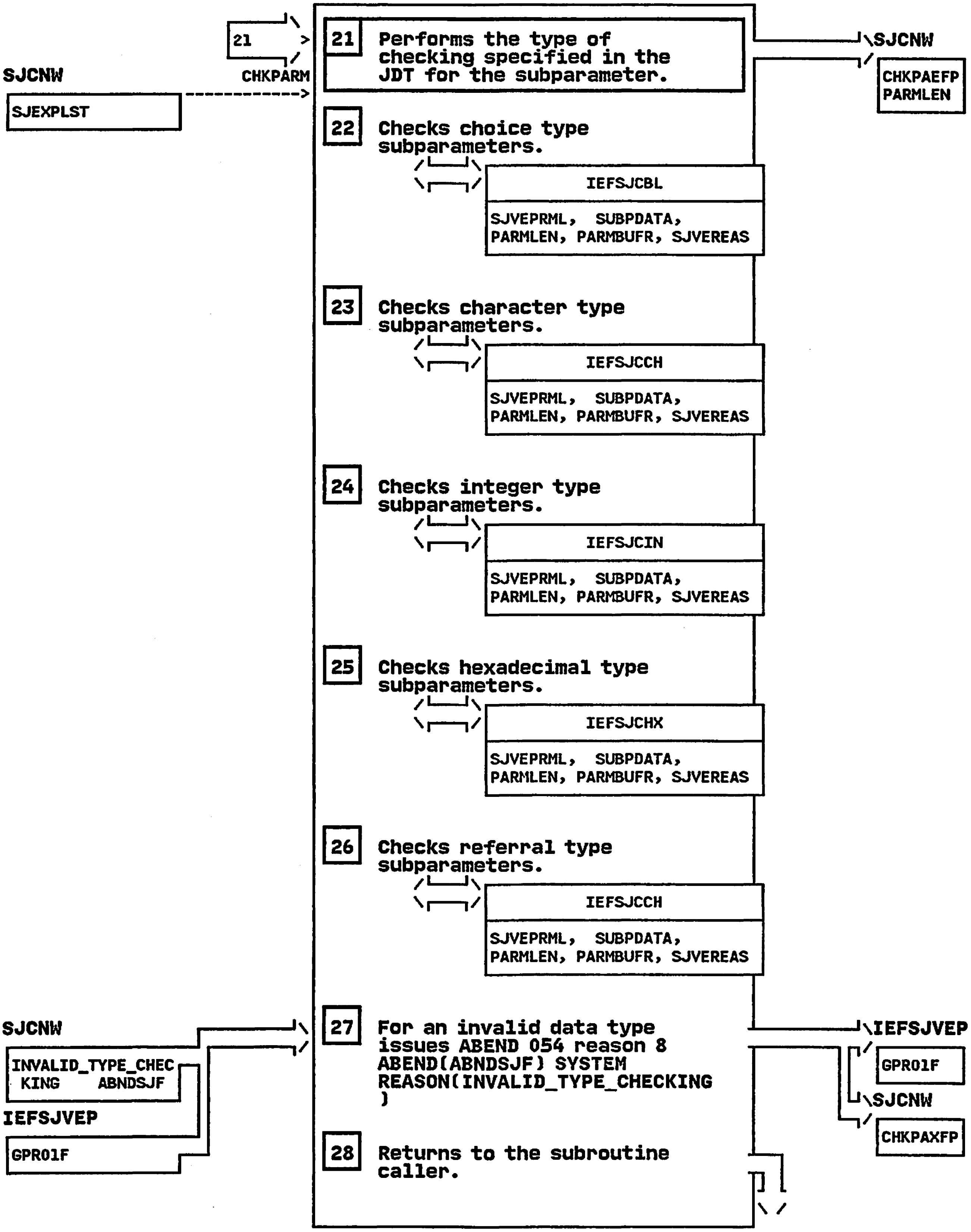
17 If a subparameter number was specified, stores the subparameter number and the sublist element number in the SJF extract parameter list.

18 Calls the SJF extract routine.

19 If IEFSJEXT returns a non-zero return code, saves the return code and reason code returned from IEFSJEXT in the SJF verify parameter list.

20 Returns to the subroutine caller.

IEFSJVER - Scheduler JCL Facility (SJF) Verify Routine STEP 21



21 Performs the type of checking specified in the JDT for the subparameter.

22 Checks choice type subparameters.

23 Checks character type subparameters.

24 Checks integer type subparameters.

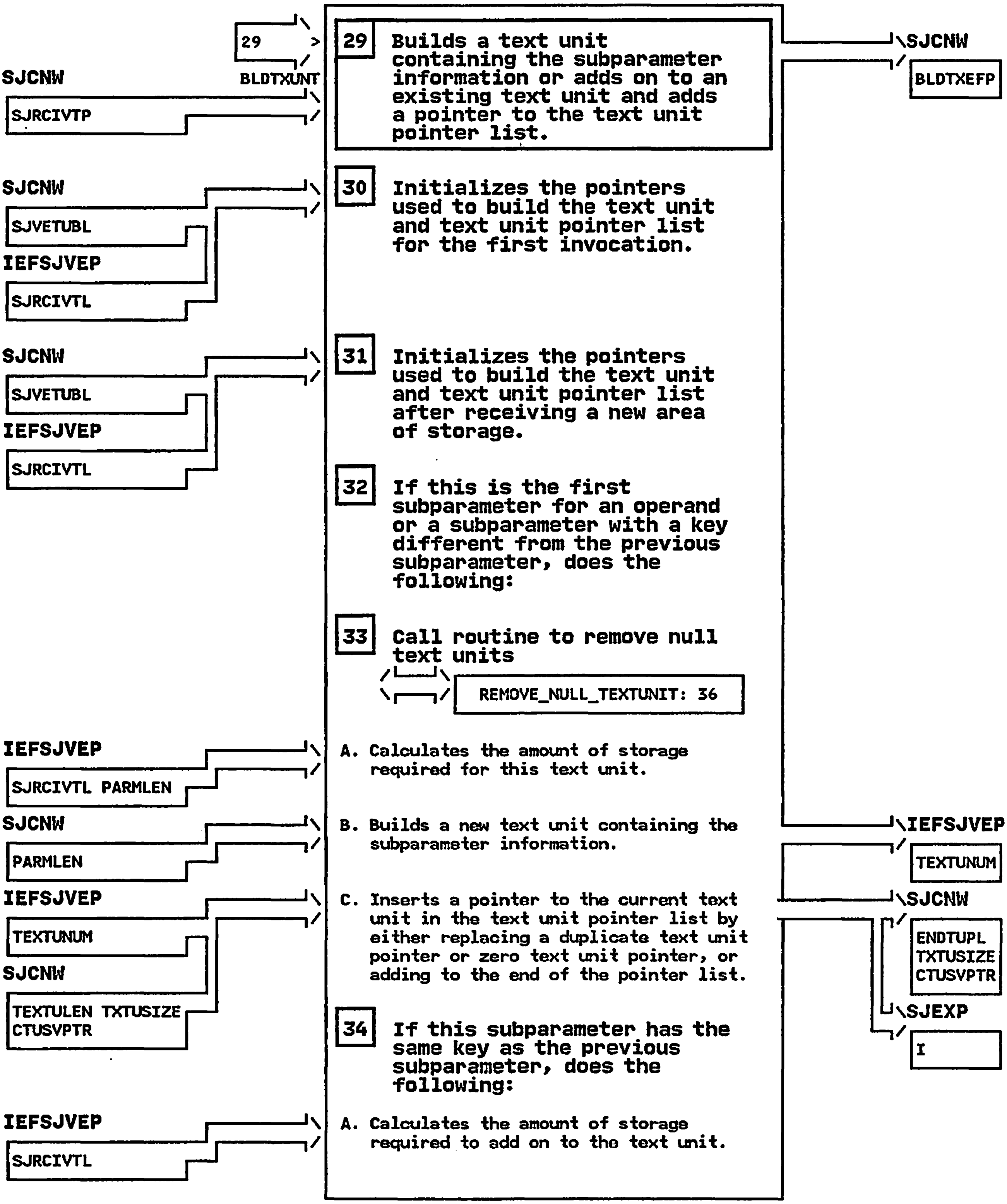
25 Checks hexadecimal type subparameters.

26 Checks referral type subparameters.

27 For an invalid data type issues ABEND 054 reason 8 ABEND(ABNDSJF) SYSTEM REASON(INVALID_TYPE_CHECKING)

28 Returns to the subroutine caller.

IEFSJVER - Scheduler JCL Facility (SJF) Verify Routine STEP 29



29 Builds a text unit containing the subparameter information or adds on to an existing text unit and adds a pointer to the text unit pointer list.

30 Initializes the pointers used to build the text unit and text unit pointer list for the first invocation.

31 Initializes the pointers used to build the text unit and text unit pointer list after receiving a new area of storage.

32 If this is the first subparameter for an operand or a subparameter with a key different from the previous subparameter, does the following:

33 Call routine to remove null text units

REMOVE_NULL_TEXTUNIT: 36

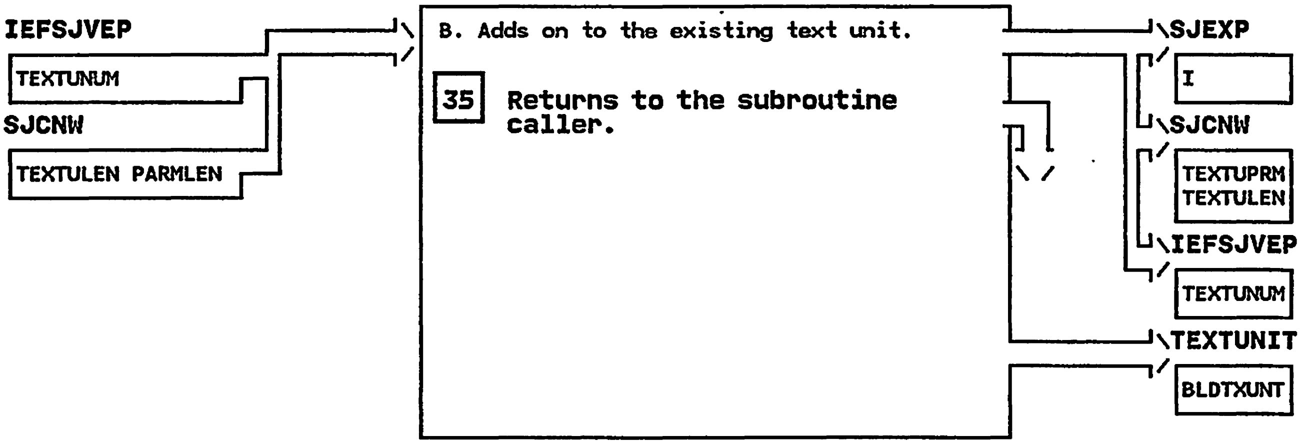
A. Calculates the amount of storage required for this text unit.

B. Builds a new text unit containing the subparameter information.

C. Inserts a pointer to the current text unit in the text unit pointer list by either replacing a duplicate text unit pointer or zero text unit pointer, or adding to the end of the pointer list.

34 If this subparameter has the same key as the previous subparameter, does the following:

A. Calculates the amount of storage required to add on to the text unit.

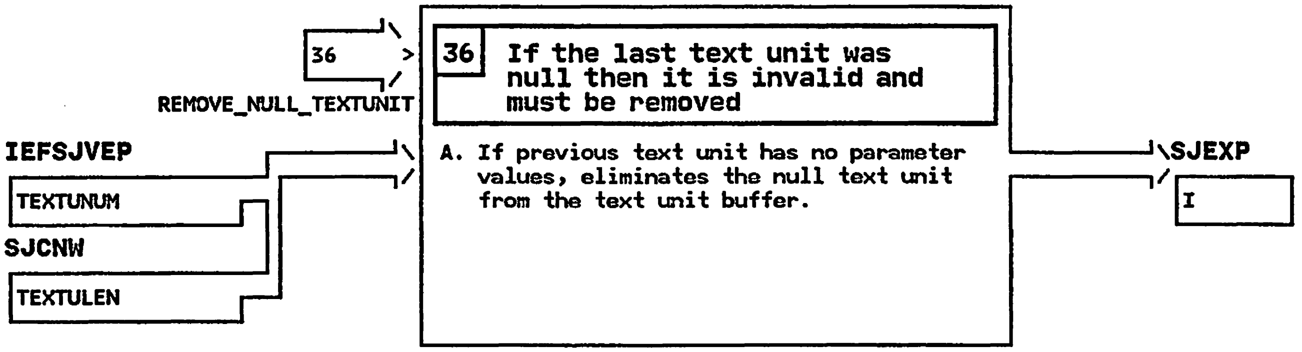


IEFSJVEP
TEXTUNUM
SJCNW
TEXTULEN PARMLEN

B. Adds on to the existing text unit.

35 Returns to the subroutine caller.

SJEXP
I
SJCNW
TEXTUPRM
TEXTULEN
IEFSJVEP
TEXTUNUM
TEXTUNIT
BLDTXUNT



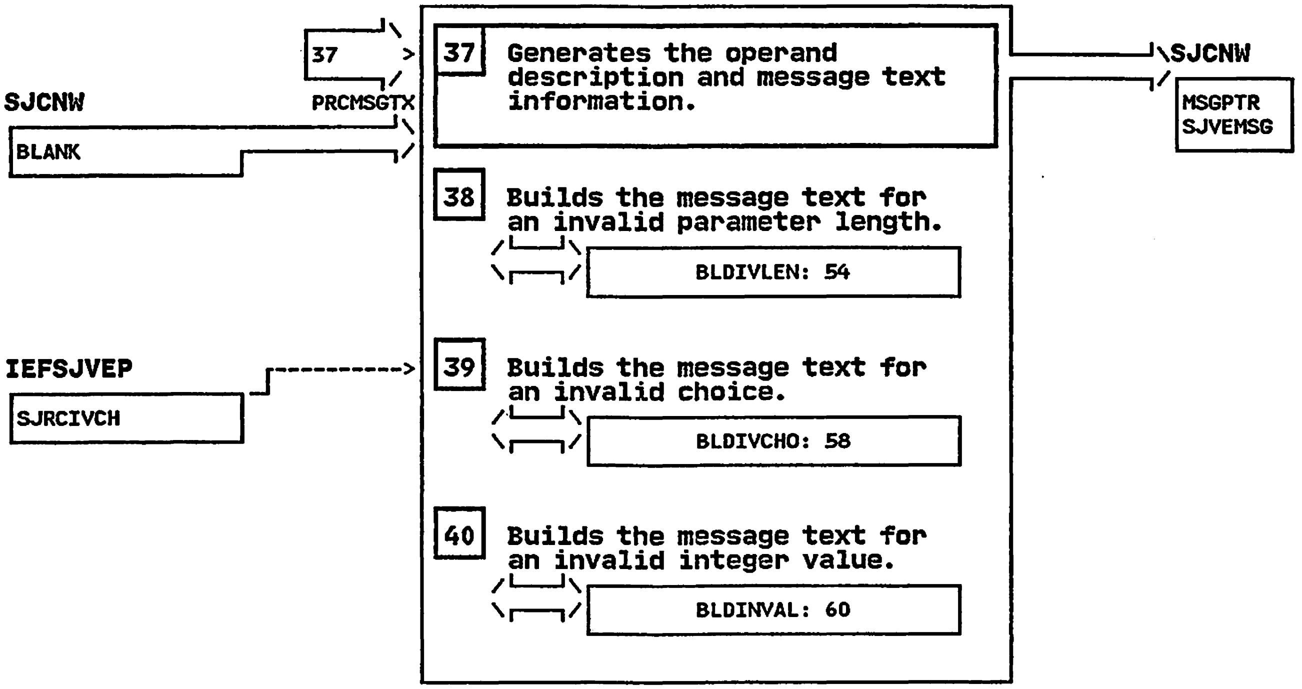
36
REMOVE_NULL_TEXTUNIT

36 If the last text unit was null then it is invalid and must be removed

IEFSJVEP
TEXTUNUM
SJCNW
TEXTULEN

A. If previous text unit has no parameter values, eliminates the null text unit from the text unit buffer.

SJEXP
I



37
PRCMSGTX

37 Generates the operand description and message text information.

SJCNW
BLANK

38 Builds the message text for an invalid parameter length.

BLDIVLEN: 54

IEFSJVEP
SJRCIVCH

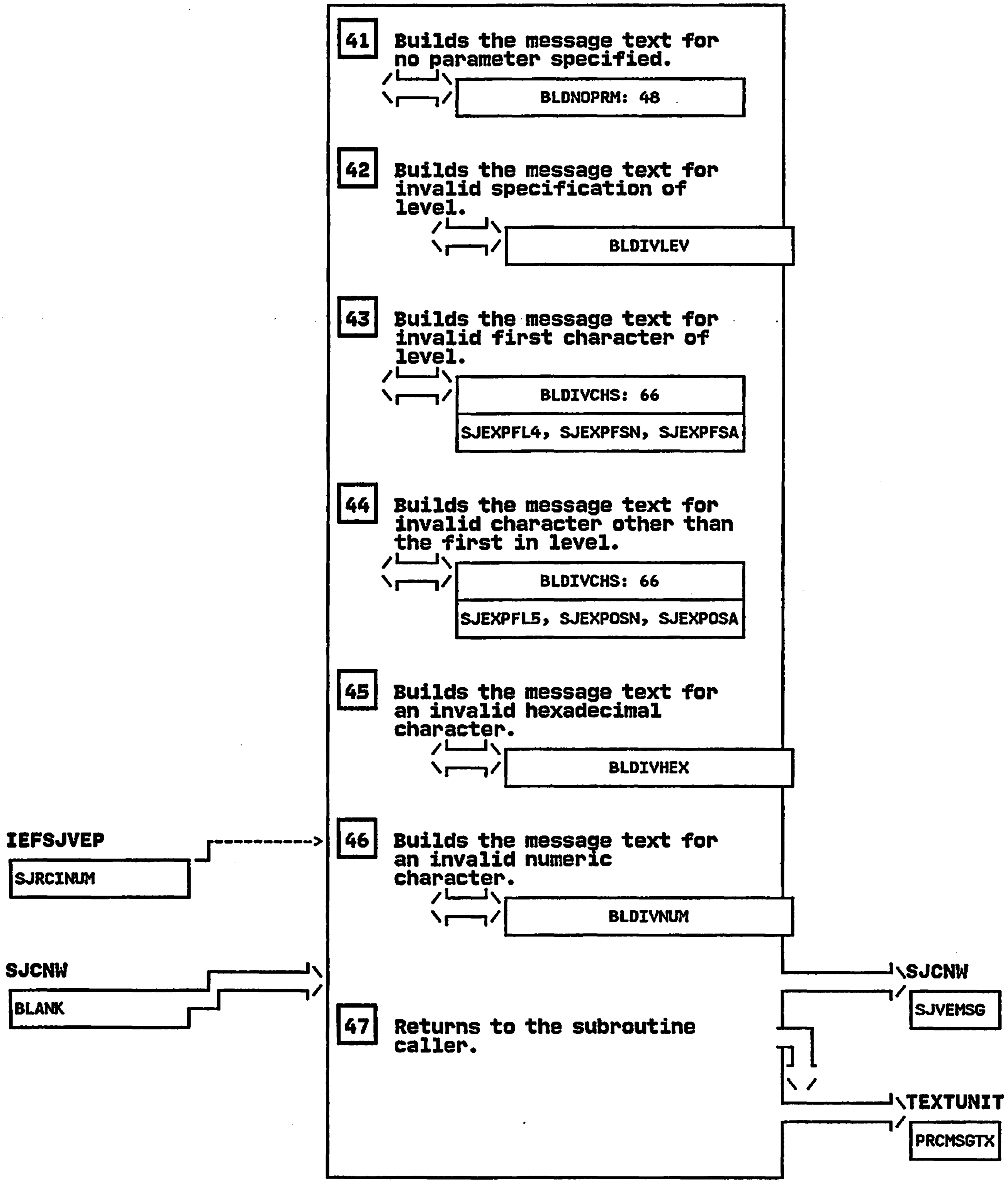
39 Builds the message text for an invalid choice.

BLDIVCHO: 58

40 Builds the message text for an invalid integer value.

BLDINVAL: 60

SJCNW
MSGPTR
SJVEMSG



41 Builds the message text for no parameter specified.

42 Builds the message text for invalid specification of level.

43 Builds the message text for invalid first character of level.

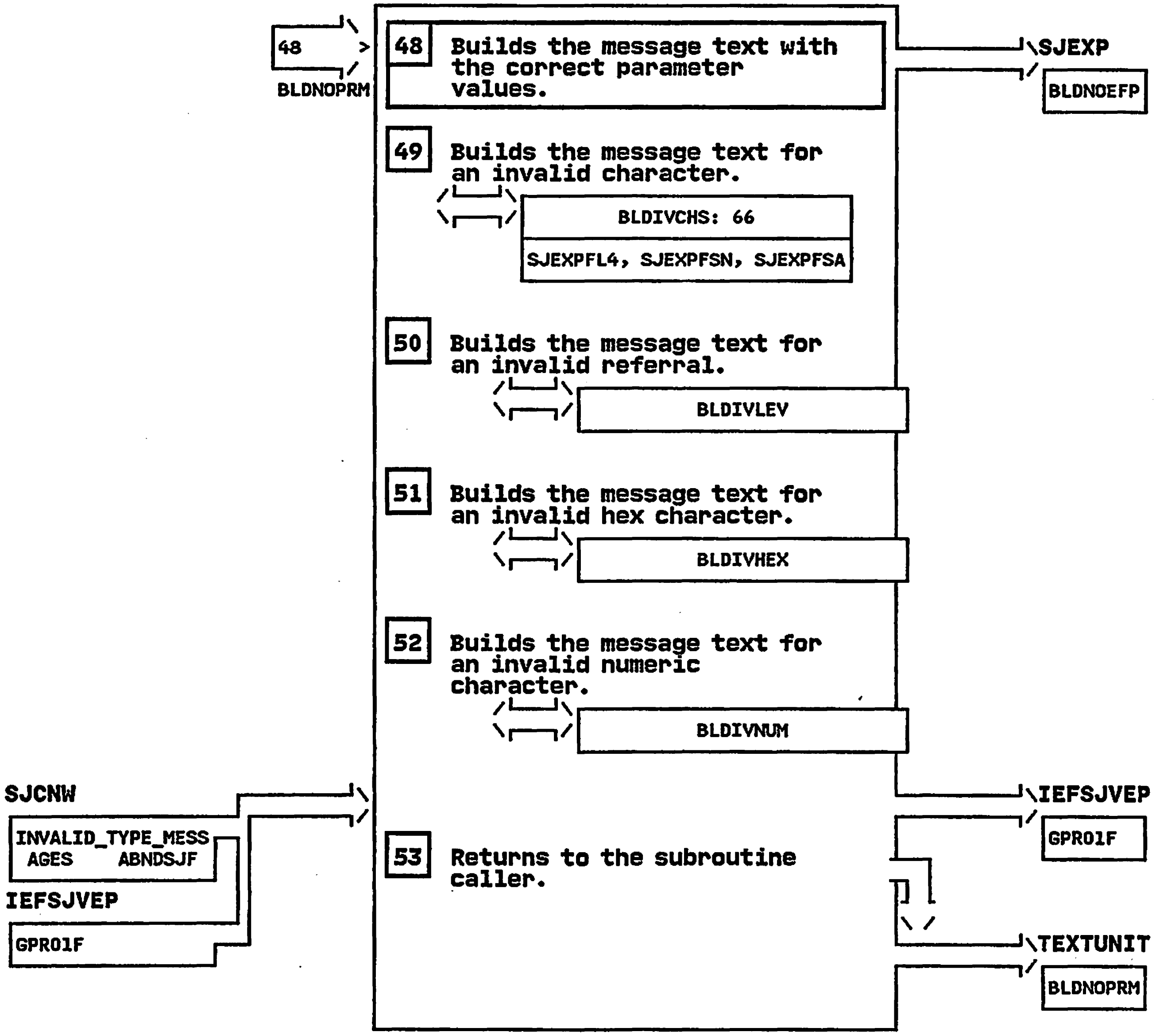
44 Builds the message text for invalid character other than the first in level.

45 Builds the message text for an invalid hexadecimal character.

46 Builds the message text for an invalid numeric character.

47 Returns to the subroutine caller.

IEFSJVER - Scheduler JCL Facility (SJF) Verify Routine STEP 48



48 Builds the message text with the correct parameter values.

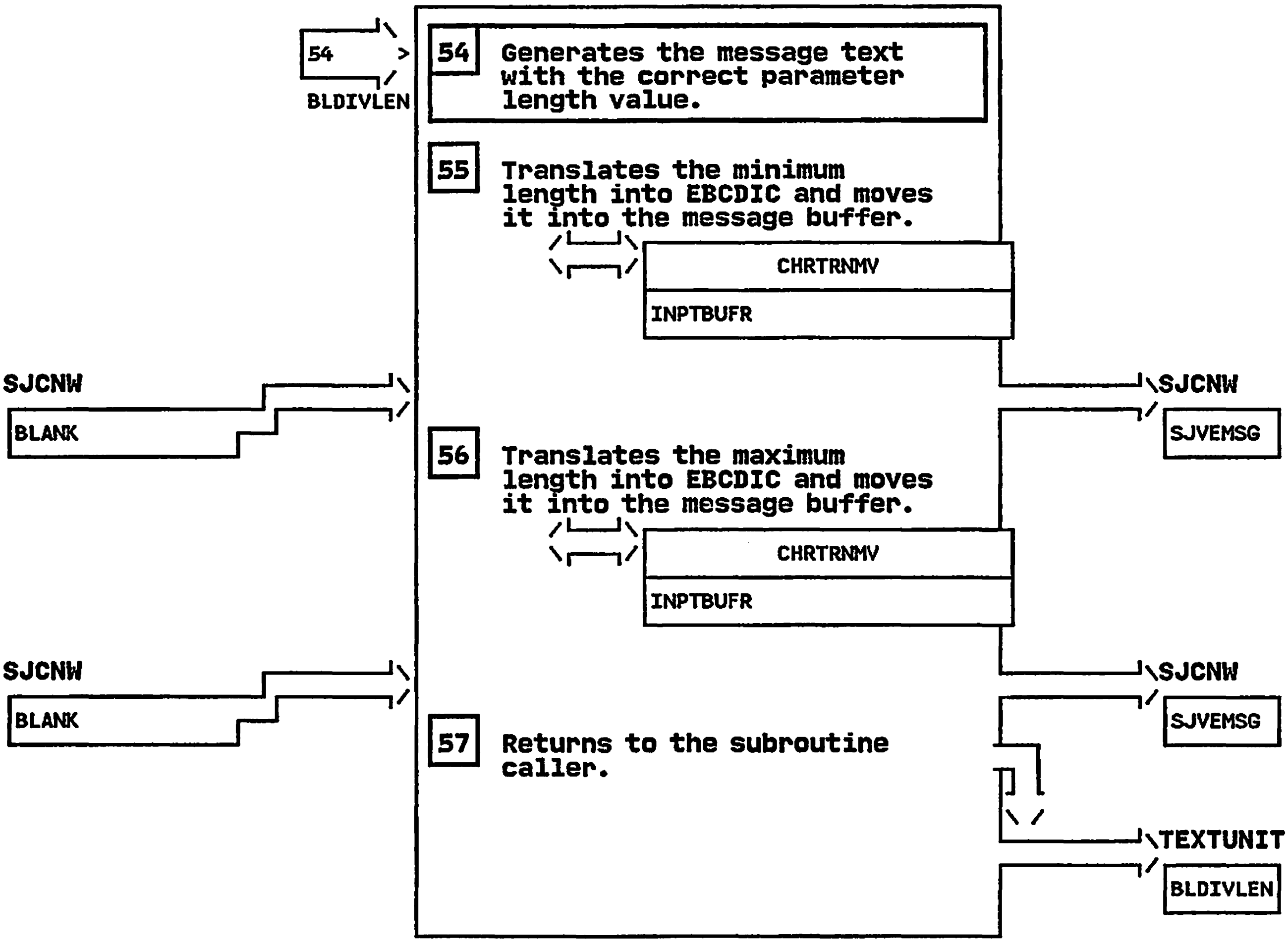
49 Builds the message text for an invalid character.

50 Builds the message text for an invalid referral.

51 Builds the message text for an invalid hex character.

52 Builds the message text for an invalid numeric character.

53 Returns to the subroutine caller.

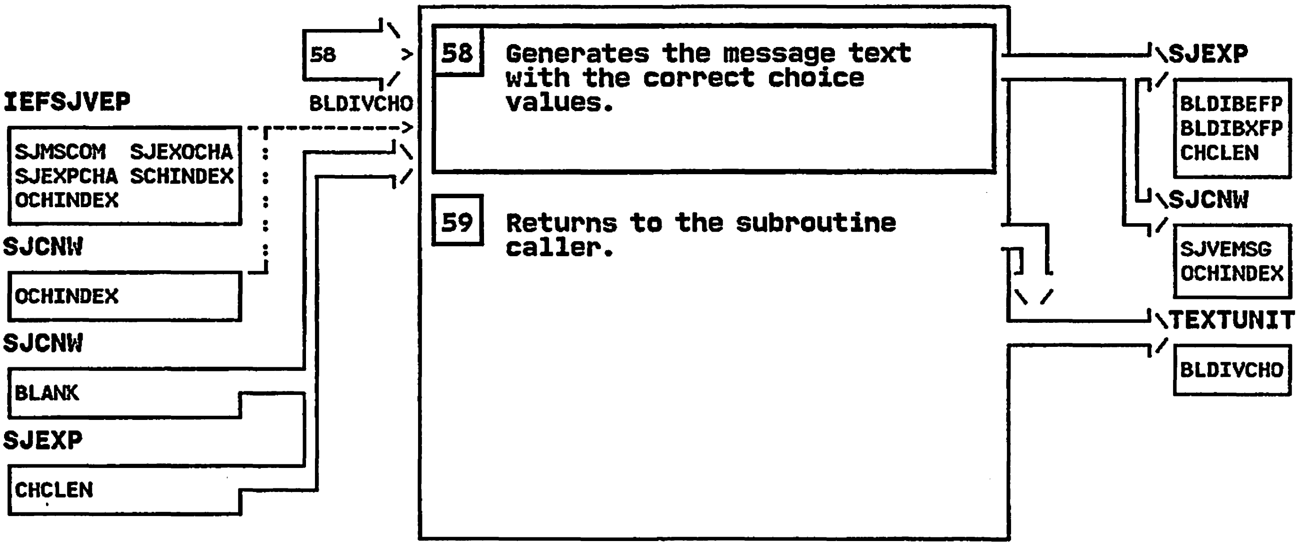


54 Generates the message text with the correct parameter length value.

55 Translates the minimum length into EBCDIC and moves it into the message buffer.

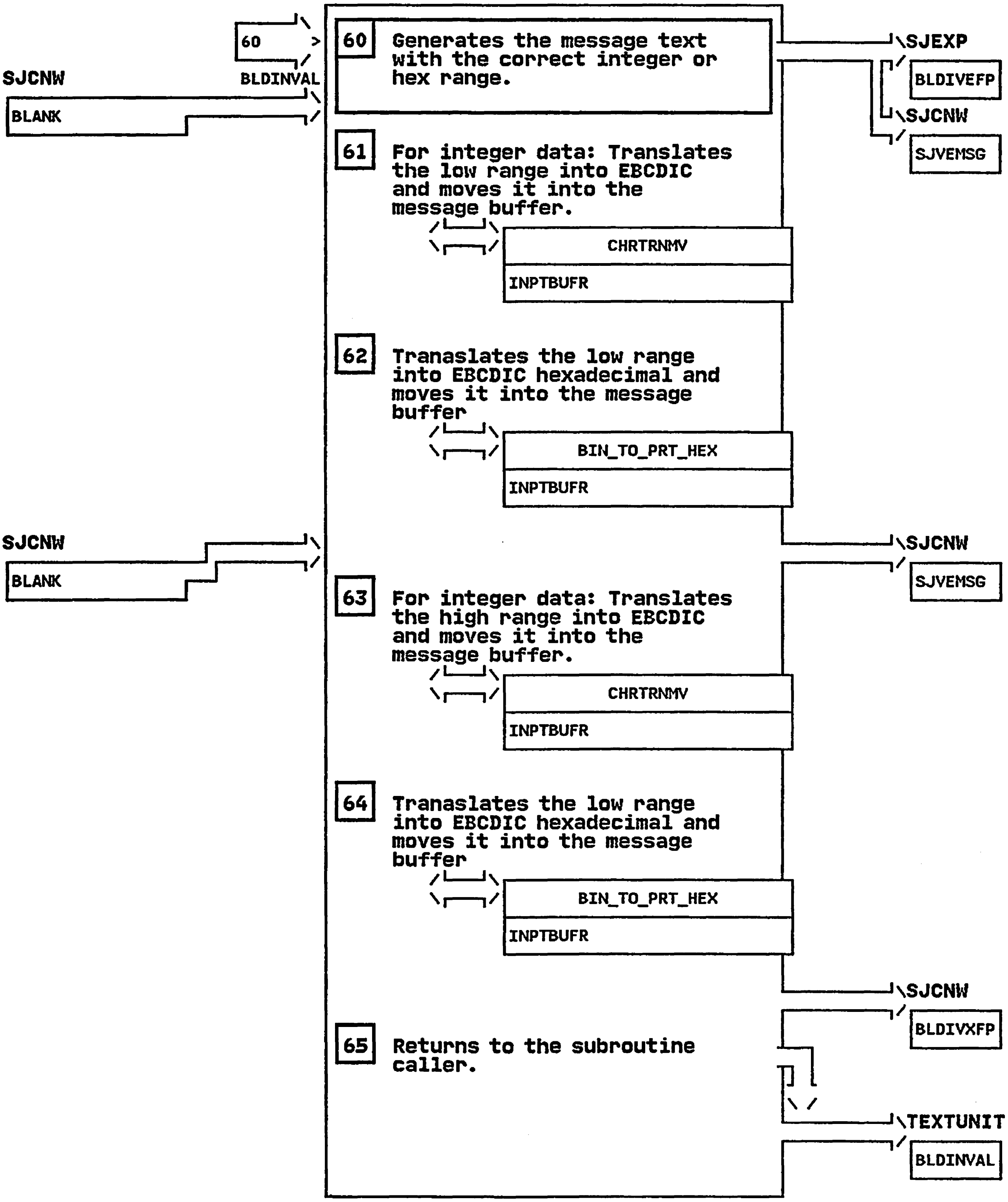
56 Translates the maximum length into EBCDIC and moves it into the message buffer.

57 Returns to the subroutine caller.



58 Generates the message text with the correct choice values.

59 Returns to the subroutine caller.



60 Generates the message text with the correct integer or hex range.

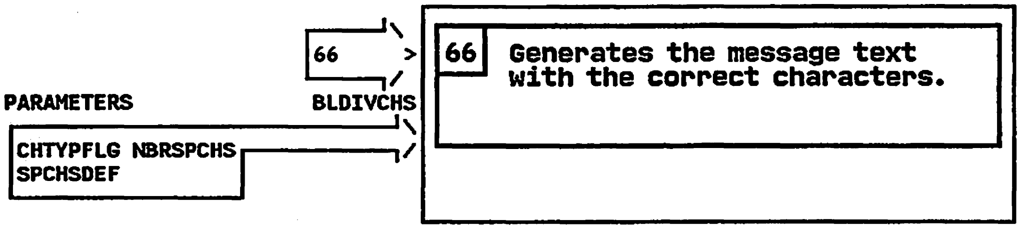
61 For integer data: Translates the low range into EBCDIC and moves it into the message buffer.

62 Tranaslates the low range into EBCDIC hexadecimal and moves it into the message buffer

63 For integer data: Translates the high range into EBCDIC and moves it into the message buffer.

64 Tranaslates the low range into EBCDIC hexadecimal and moves it into the message buffer

65 Returns to the subroutine caller.



66

PARAMETERS

BLDIVCHS

CHTYPFLG NBRSPCHS
SPCHSDEF

66 Generates the message text with the correct characters.

IEFSJWRT - MODULE DESCRIPTION

DESCRIPTIVE NAME: Scheduler JCL Facility (SJF) Write SWB Routine

FUNCTION:
This module locates a specific SWB and updates its data portion.

ENTRY POINT: IEFSJWRT

PURPOSE: See Function

LINKAGE: CALL

CALLERS:
Scheduler JCL Facility control routine (IEFSJCNL)
Scheduler JCL Facility update routine (IEFSJUPD)

INPUT:
SJF Write SWB parameter list, IEFSJWRP:

FIELD	LENGTH/MASK	DESCRIPTION
SJWRP		Parameter list
SJWRID	4	Identifier 'SJWR'
SJWRVERS	1	Version number
SJWRFLAG	1	Control flags
SJWRNREC	x'80'	No recovery
SJWRNOCU	x'40'	No clean up
SJWRLEN	2	Length of parameter list
SJWRSTOR	4	Local storage pointer
SJWRREAS	4	Reason code (returned)
SJWRTOKN	8	SWB token
SJWRANBK	4	Address of the anchor control block or of the first control block for a JCL statement
SJWRANCA	4	Address of a word pointing to the SWB chain or zero
SJWRSWBI		Data to identify SWB
SJWROWNM	8	Owner name
SJWRBKID	2	Block ID
SJWRRSV1	2	Reserved
SJWRCHNI		Data to identify SWB chain
SJWRVERB	8	Verb
SJWRLABL	8	Label
SJWRFUNC	1	Flag byte
SJWRNCHN	x'80'	A new SWB chain is to be built
SJWRNSWA	x'40'	Request is for a non-SWA SWB
SJWRDYNS	x'20'	Request is for a dynamic SWB chain
SJWRRSV2	3	Reserved
SJWRSTMT	4	JCL statement number
SJWRBYTE	2	Byte offset for data portion update
SJWRDLEN	2	Length of data to be stored
SJWRDATA	4	Address of data to be stored
SJWRALT	4	Address of Alternate SWA Manager routine

The input to this module also includes the SJF control workarea (IEFSJCNW).

OUTPUT:
SJF Write SWB parameter list, IEFSJWRP:

SJWRREAS = reason code

IEFSJWRT - MODULE DESCRIPTION (Continued)

EXIT NORMAL: Return to caller

EXIT ERROR: Return to caller

ENTRY POINT: WRTRETRY

PURPOSE:
Performs clean up processing when an ABEND occurs during SJF Write processing.

LINKAGE: SYNCH

CALLERS: RTM

INPUT: ESTAE parameter list

OUTPUT: None

EXIT NORMAL:

EXIT ERROR: Return to caller

EXTERNAL REFERENCES:

ROUTINES: (IEFSJBLD) Scheduler JCL Facility (SJF) Build SWB

DATA AREAS:

IEFSJCNW	- SJF Control Workarea
IEFSJRC	- SJF Reason Codes
IEFSJSWP	- IEFSJSWA Parameter List
IEFSJWRP	- SJF Write SWB Parameter List
IEFSJBLP	- SJF Build SWB Parameter List
IEFZB502	- SWA Prefix
IEFZB505	- External Parameter Area Extended

CONTROL BLOCKS:

CVT	- Communication Vector Table
JCT	- Job Control Table
JCTX	- Job Control Table Extension
JESCT	- JES Communication Table
PSA	- Prefix Save Area
SCT	- Step Control Table
SIOT	- Step Input Output Table
SWB	- Scheduler Work Block

SERIALIZATION:

Holds the local lock during branch entry FREEMAIN of a non SWA SWB.

IEFSJWRT - MODULE OPERATION

This module receives control to store data into a SWB. It performs the following functions:

1. Checks if a verb (SJWRVERB) is specified in the input parameter list. If it is not, sets register 15 to 4, sets a reason code of SJRCIVRB (508) in SJWRREAS, and returns.

2. Validates the SWB structure address:
   - Calls IEFSJSWA to determine validity of the token and to gain addressability to the SWB chain.

3. Verifies that the length of the data to be stored in the SWB data portion will not exceed the length of the SWB. If it will, sets register 15 to 4, sets a reason code of SJRCIVDT (800) in SJWRREAS, and returns.

4. If the SWB structure address is zero, builds the parameter list (IEFSJBLP) and invokes the Scheduler JCL Facility (SJF) build routine to build a new SWB.

5. If the SWB structure address is not zero, locates the specified SWB chain (with the matching verb (SJWRVERB) and label (SJWRLABL)). If the SWB chain is located, checks if this is an explicit build new SWB chain request. If it is and the label is non zero then a SWB chain already exists with the specified verb and label. The module then sets register 15 to 4, sets a reason code of SJRCDUPV (801) in SJWRREAS and returns to caller.

   If this is not an explicit build new SWB chain request, finds the specified SWB (with the matching owner name (SJWROWNM) and Block ID (SJWRBKID)) on the located SWB chain.

   If the SWB is not found, builds the parameter list, IEFSJBLP, and invokes the Scheduler JCL Facility Build routine to build a new SWB.

6. If there is parameter data to be stored (its length (SJWRDLEN) is not zero), moves the data into the SWB data portion and sets the corresponding validity bits in the SWB prefix.

7. Returns to the caller.

RECOVERY OPERATION:

If an abend occurs in this module, the scheduler JCL facility control routine's recovery will receive control from RTM. The recovery routine specifies the retry address in the SJF workarea (WRTRETRY) to RTM. When the retry segment (WRTRETRY) receives control from RTM, it does the following:

1. If a SWB has been built but not anchored to the SWB chain, then:
   - If the SWB is a SWA SWB, calls SWA Manager to free the SWB.
   - If the SWB is a non-SWA SWB, issues a FREEMAIN for the SWB.

2. Sets the return code to indicate an SJF system error.

IEFSJWRT - MODULE OPERATION (Continued)

3. Returns to the caller.

IEFSJWRT - DIAGNOSTIC AIDS

ENTRY POINT NAMES: IEFSJWRT
WRTRETRY

MESSAGES: None

ABEND CODES: None

WAIT STATE CODES: None

RETURN CODES:

ENTRY POINT IEFSJWRT:

EXIT NORMAL:

Register 15 = 0 - Processing successful

Reason codes in SJWRREAS:
SJRCNOER (0) - Processing successful

EXIT ERROR:

Register 15 = 4 - Request cannot be processed

Reason codes in SJWRREAS:
SJRCIVRB (508) - Invalid Verb
SJRCIVTK (2) - Invalid SWB token
SJRCIVDT (800) - Invalid SWB data
SJRCDUPV (801) - SWB chain already exists for specified verb and label

See SJF Build SWB for additional reason codes.

ENTRY POINT WRTRETRY:

EXIT NORMAL:

Register 15 = 20 - SJF system error

REGISTER CONTENTS ON ENTRY:

ENTRY POINT IEFSJWRT:

Register 0	= Undefined
Register 1	= Address of a two word parameter list. The first word contains the address of the Write SWB parameter list (IEFSJWRP) and the second word contains the address of the SJF control workarea (IEFSJCNW)
Registers 2-12	= Undefined
Register 13	= Address of an 18 word savearea
Register 14	= Return address
Register 15	= Entry point address

ENTRY POINT WRTRETRY:

Register 1	= Address of ESTAE parameter list
Registers 0,2-14	= Undefined
Register 15	= Entry point address

IEFSJWRT - DIAGNOSTIC AIDS (Continued)

REGISTER CONTENTS ON EXIT:

ENTRY POINT IEFSJWRT:

```
    Registers 0-14 = Restored
    Register    15 = Return code
```

ENTRY POINT WRTRETRY:

```
    Registers 0-14 = Restored
    Register 15    = Return code
```

MVS/Extended Architecture
System Logic Library:
Scheduler JCL Facility

READER'S
COMMENT
FORM

LY28-1740-1

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comment are:

Clarity   Accuracy   Completeness   Organization   Coding   Retrieval   Legibility

If you wish a reply, give your name, company, mailing address, and date:

____

____

____

____

What is your occupation? ________________________________

How do you use this publication? ____________________________

Number of latest Newsletter associated with this publication: ________________

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

MVS/Extended Architecture System Logic Library: Scheduler JCL Facility

LY28-1740-1 S370-36

Reader's Comment Form

Cut or Fold Along Line

Fold and tape Please Do Not Staple Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D58, Building 921-2
PO Box 950
Poughkeepsie, New York 12602

Fold and tape Please Do Not Staple Fold and tape

Printed in U.S.A.

IBM®

LY28-1740-01

INDEX

MVS/Extended Architecture System Logic Library: Scheduler JCL Facility

LY28-1740-1 S370-36

Reader's Comment Form

Cut or Fold Along Line

Fold and tape Please Do Not Staple Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D58, Building 921-2
PO Box 950
Poughkeepsie, New York 12602

Fold and tape Please Do Not Staple Fold and tape

Printed in U.S.A.

IBM®

MVS/Extended Architecture System Logic Library: Scheduler JCL Facility

LY28-1740-1

S370-36

Printed in U.S.A.