SPERRY UNIVAC
COMPUTER SYSTEMS

PUBLICATIONS REVISION

Operating System/3 (OS/3)

Dialog Processor

**User Guide/
Programmer Reference**

UP-8858 Rev. 1

This Library Memo announces the release and availability of "SPERRY UNIVAC® Operating System/3 (OS/3) Dialog Processor User Guide/Programmer Reference", UP-8858 Rev. 1.

This revision documents the following corrections:

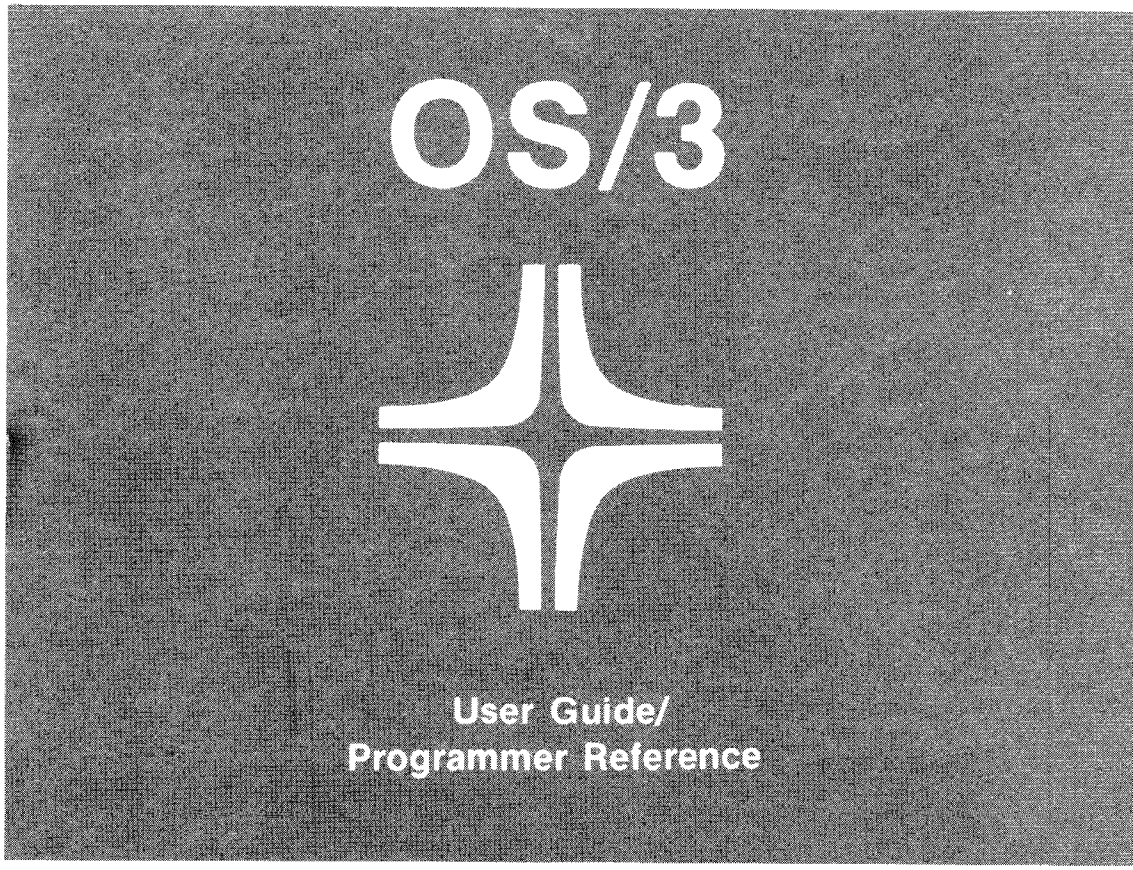
- The DVC-LFD sequences for all of the files referenced in a // USE DP statement must precede the DVC-LFD sequence containing the // USE DP statement.
- The REPLACE function key has been changed to the INSERT function key.

Destruction Notice: This revision supersedes and replaces "SPERRY UNIVAC Operating System/3 (OS/3) Dialog Processor User Guide/Programmer Reference", UP-8858, released on Library Memo dated October, 1980. Please destroy all copies of UP-8858 and/or its Library Memo.

Additional copies may be ordered by your local Sperry Univac representative.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS:
Mailing Lists BZ, CZ and MZ	Mailing Lists A00, A01, B00, B01, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76, and 76U (Cover and 37 pages)	Library Memo for UP-8858 Rev. 1
		RELEASE DATE: September, 1982

Dialog Processor



SPERRY UNIVAC

This document contains the latest information available at the time of preparation. Therefore, it may contain descriptions of functions not implemented at manual distribution time. To ensure that you have the latest information regarding levels of implementation and functional availability, please consult the appropriate release documentation or contact your local Sperry Univac representative.

Sperry Univac reserves the right to modify or revise the content of this document. No contractual obligation by Sperry Univac regarding level, scope, or timing of functional implementation is either expressed or implied in this document. It is further understood that in consideration of the receipt or purchase of this document, the recipient or purchaser agrees not to reproduce or copy it by any means whatsoever, nor to permit such action by others, for any purpose without prior written permission from Sperry Univac.

Sperry Univac is a division of the Sperry Corporation.

This document was prepared by Systems Publications using the SPERRY UNIVAC UTS 400 Text Editor. It was printed and distributed by the Customer Information Distribution Center (CIDC), 555 Henderson Rd., King of Prussia, Pa., 19406.

PRINTED IN U.S.A.

PAGE STATUS SUMMARY

ISSUE: UP-8858 Rev. 1
RELEASE LEVEL: 7.0 Forward

Part/Section	Page Number	Update Level
Cover/Disclaimer		
PSS	1	
Preface	1	
Contents	1, 2	
1	1 thru 7	
2	1 thru 9	
3	1 thru 8	
Glossary	1 thru 3	
Index	1 thru 4	
User Comment Sheet		

Part/Section	Page Number	Update Level

Part/Section	Page Number	Update Level

All the technical changes are denoted by an arrow (→) in the margin. A downward pointing arrow (↓) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow (↑) is found. A horizontal arrow (→) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.

Preface

This manual is one of a series designed to instruct and guide the programmer in the use of the SPERRY UNIVAC Operating System/3 (OS/3). It specifically describes how the dialog processor functions in an interactive processing environment.

The intended audience for this manual is the programmer with a general knowledge of interactive processing and OS/3 job control.

The dialog processor is also described in the introduction to dialog processing services manual, UP-8857 (current version). Related topics are OS/3 job control and the dialog specification language. They are described in the current versions of introductory manuals, user guides, and programmer references.

Contents

GLOSSARY

INDEX

USER COMMENT SHEET

FIGURES

TABLES

1. The Dialog Processor in an Interactive Environment

1.1. INTRODUCTION

The dialog processor is the OS/3 component that manages an interactive dialog during a dialog session at a workstation. It processes an interactive dialog by displaying dialog text at the workstation screen, accepting user responses to the dialog, and routing those responses to an application program. The purpose of a dialog session is to solicit input from the workstation user and pass it to an application program. The dialog processor is the interface between you and the operating system when you conduct a dialog session.

You should be familiar with OS/3 interactive processing terminology to better understand how the dialog processor functions. The glossary in this manual defines the terms you need to know and can be used as a quick-reference dictionary during the course of our discussion.

For a complete picture of how the dialog processor works, you should also be familiar with other elements of an interactive job. For this reason, we'll discuss how dialogs are created, how application programs call a dialog, and the OS/3 job control stream that identifies the dialog processor and the dialog you're using to the operating system. We'll also examine some sample applications.

1.2. GENERAL CONCEPTS

Interactive processing is a computing environment where you communicate directly with the operating system through a workstation. This contrasts with a batch processing environment, where you submit your jobs to a computer operator to be run. Interactive communication between a user and the operating system is accomplished through a dialog session at a workstation. A workstation is an interactive device with a CRT screen for displaying dialog text and a keyboard device for entering user input.

An interactive dialog makes it possible to communicate directly with the operating system. The purpose of an interactive dialog, as we mentioned in the introduction, is to solicit input from the workstation user that is used by another program. Interactive dialogs can be thought of as "scripts" that provide the framework for "conversations" between you and the operating system.

The elements that make interactive dialogs possible are:

- a dialog written in the dialog specification language;
- an OS/3 job control stream;
- the dialog processor; and
- an application program.

Dialogs are either written by you or supplied by Sperry Univac. The Sperry Univac dialogs guide you step by step through the processes of building a job control stream or a system generation parameter stream. The dialog processor is transparent when you're using Sperry Univac dialogs, so we'll discuss it only in conjunction with user-written dialogs.

The dialog specification language is a high-level language designed specifically for writing interactive dialogs. (See 1.3.)

An OS/3 job control stream identifies the devices and files needed for a particular job to the operating system. When a job uses an interactive dialog, the job control stream identifies the workstation, the dialog processor, and the dialog you're using, as well as any other devices and files needed for that job.

The dialog processor displays the questions and messages contained in a dialog at the workstation screen, accepts user input, and routes that input (in the form of output records) to an application program.

The application program that calls the dialog processor can be of any type, in any language. Programs that already exist in your operation, as well as newly written programs, can make effective use of an interactive dialog.

You may be asking yourself at this point: Why use an interactive dialog? The answer is two-fold: ease of use and adaptability. Because you communicate directly with the operating system in a conversational manner, you are less likely to make errors and, if you do, the dialog tells you immediately. You don't have to wait for verification of your input as you do in a batch environment. In addition, a dialog can be written to be used as a tutorial aid, giving novice users immediate hands-on experience; or, it can be used by more experienced personnel who don't need tutorial assistance. Sperry Univac dialogs include HELP screens that explain dialog choices to the workstation user.

The dialog processor is the manager of a dialog session, passing processing control between the dialog and the application program it complements. The audit version of the dialog processor allows you to selectively edit the results of a dialog session. The audit version is explained in Section 3. Figure 1-1 shows how the dialog processor functions with the other elements of a job that uses an interactive dialog.

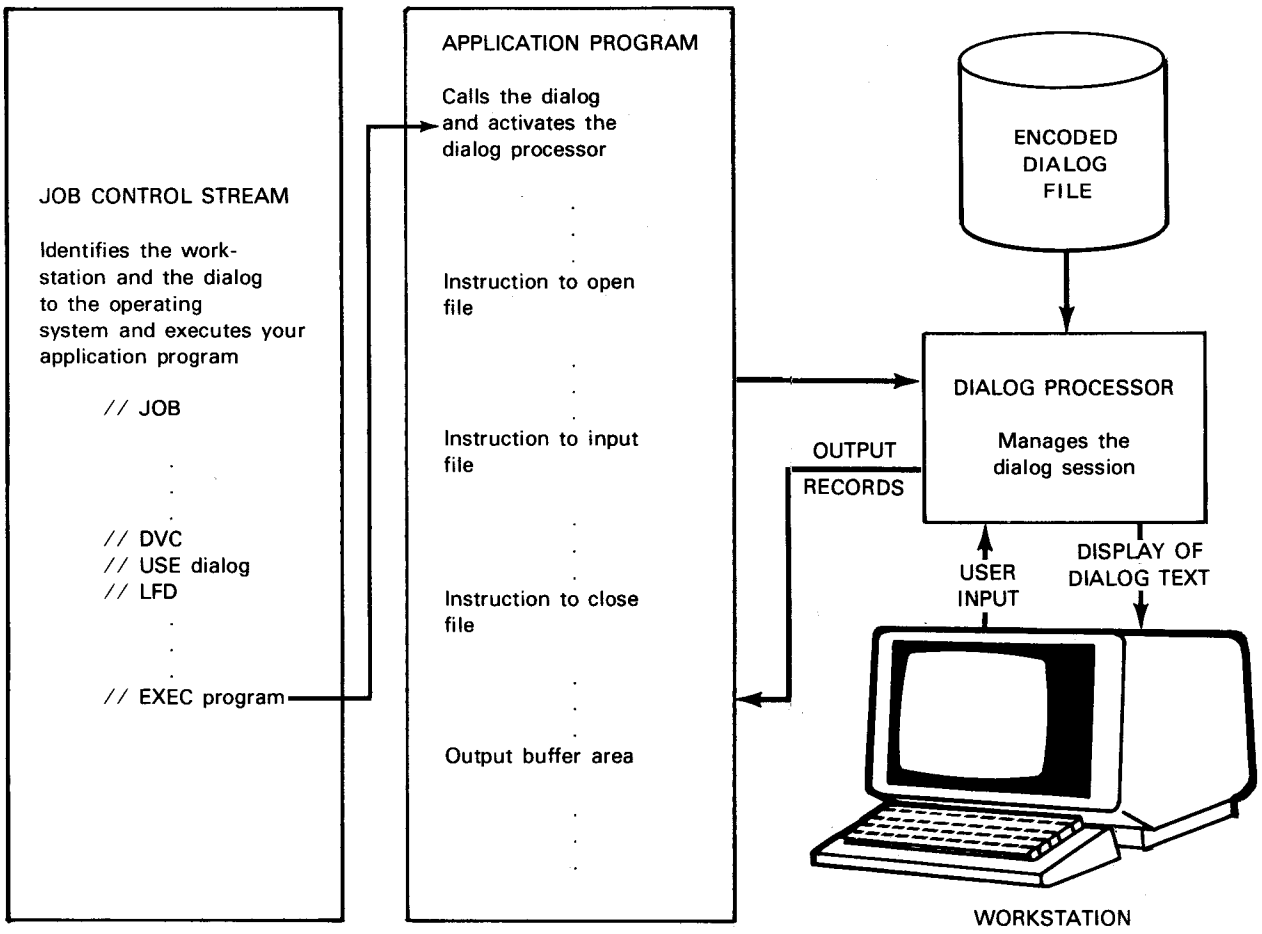


Figure 1—1. The Dialog Processor

1.3. CREATING A DIALOG: DIALOG SPECIFICATION LANGUAGE

Before an interactive dialog is managed by the dialog processor and used by another program, it must, of course, be written. The dialog specification language is a high-level language designed specifically for creating interactive dialogs. It is beyond the scope of this manual to explain how to write a dialog in the dialog specification language, but it is important to understand some basic concepts of the language to better understand how the dialog processor functions. We'll cover those general concepts here. The dialog specification language user guide/programmer reference, UP-8806 (current version) explains how to write an interactive dialog in the dialog specification language.

The dialog specification language allows you to specify dialog structure, messages to be displayed at the workstation screen, input parameters, and the content and format of output records and printed reports.

The dialog specification language employs a block structure to create dialog "trees" (Figure 1-2). A dialog tree is a programming structure having three key elements:

Trunk - dialog commands executed with each branch

Node - a decision point in a dialog; response determines which screen the dialog processor displays next

Branch - a set of dialog specification language commands that request input, display messages, produce output, and perform other processing functions

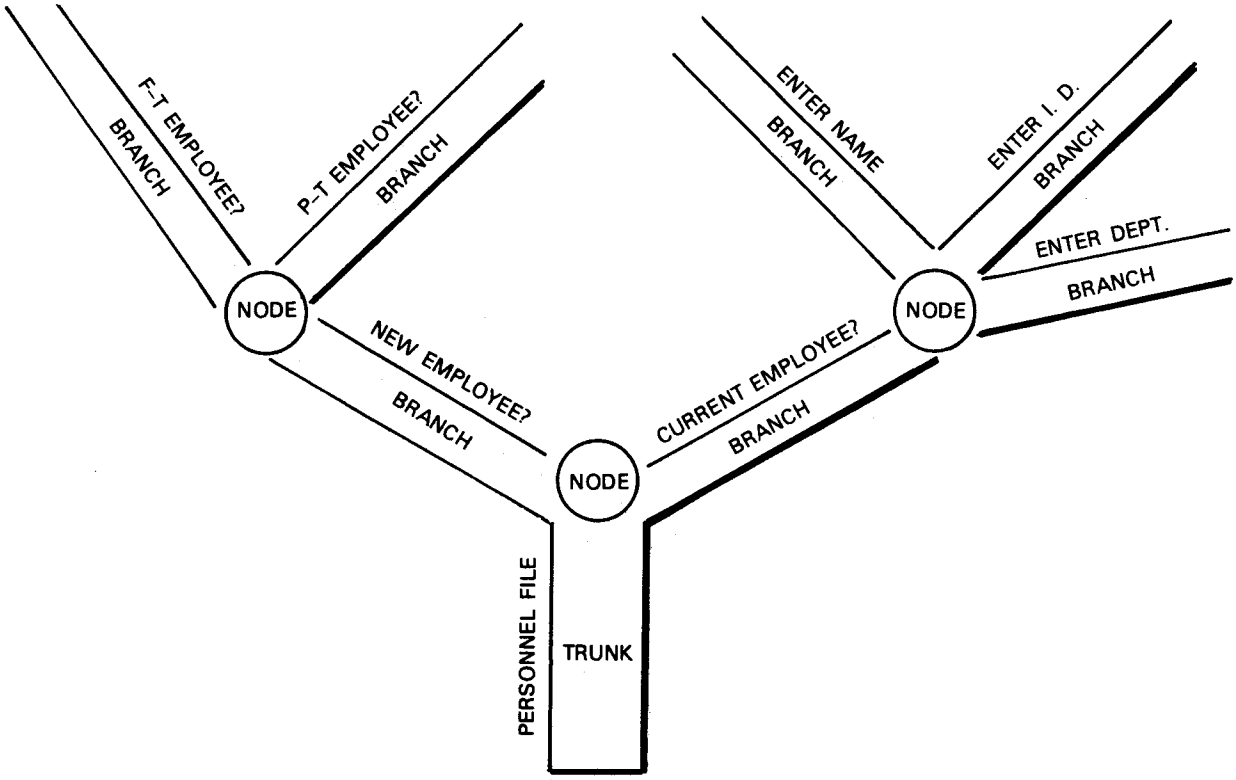


Figure 1—2. A Section of a Tree-Structured Dialog

A dialog specification language block is delimited by the DO and END commands. The DISPLAY and ENTER commands in the dialog specification language direct the dialog processor to display dialog text at the workstation screen and request input from you. The OUTPUT command with EOR (end-of-record) specified directs the dialog processor to create an output record and store it in an output buffer identified in the application program. This output record contains your responses to the dialog.

A dialog paragraph is one execution of a tree. Dialog paragraphs are explained in more detail in Section 3, where we discuss how to change your responses to a dialog in a subsequent session.

The dialog processor handles the display of standard control and error messages, the positioning of headings, the tabulation of choices, and error recovery procedures for an interactive dialog. These procedures do not have to be included in the DSL program.

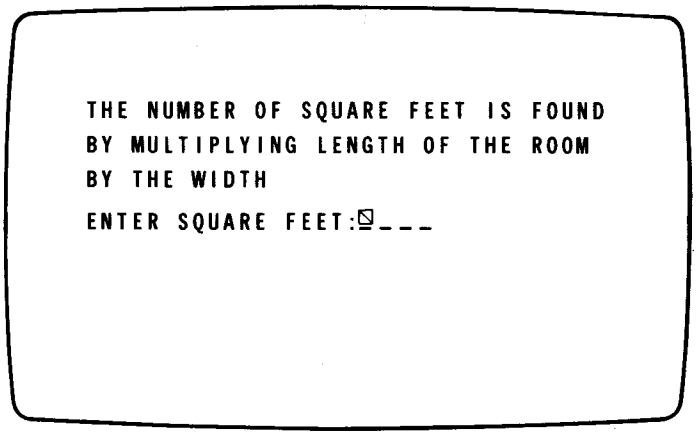
You can see the relationship between a DSL program and the workstation displays it creates in the following example:

DSL Source Code:

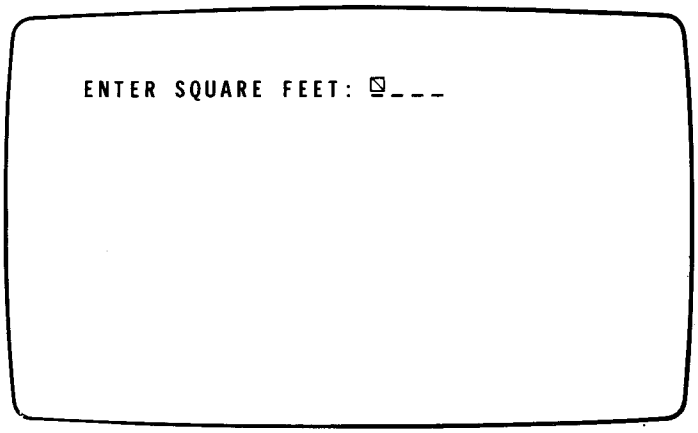
```
          .
          .
          .
SOLUTION: DATA\D4\;

          .
          .
          .
SQFEET:   BRANCH
          IF LEVEL=''2'' THEN DO; /* TUTORIAL SECTION BEGINS HERE.*/
          DISPLAY ''THE NUMBER OF SQUARE FEET IS FOUND'';
                  NEWLINE
          DISPLAY ''BY MULTIPLYING LENGTH OF THE ROOM'';
                  NEWLINE;
          DISPLAY ''BY THE WIDTH'';
                  NEWLINE; NEWLINE;
          END;         /* END OF TUTORIAL SECTION */
          DISPLAY''ENTER SQUARE FEET:'';  ENTER SOLUTION;
          END;              /* END SQFEET BRANCH. */
          .
          .
          .
```

Workstation Display:



This small section of the dialog specification language program is not complete – it is used only as an illustration of how the structure of a dialog specification language program gives it the flexibility to provide you with a HELP screen if you need one. If the workstation user had not requested help, the workstation screen generated by this branch of the dialog specification language program would look like this:



The dialog specification language source code is read from a library file by the dialog specification language translator. The translator produces a dialog, which is stored in an encoded dialog file, and a printer listing of each compilation. Each dialog is identified by a user-supplied name that is specified in the USE job control statement when you execute a program that uses the dialog to solicit workstation input (2.2.)

When a dialog is called by an application program, it is loaded into main storage, where it is available to the dialog processor. The dialog processor manages the subsequent dialog session with the workstation user.

Figure 1-3 illustrates the steps involved in creating an interactive dialog.

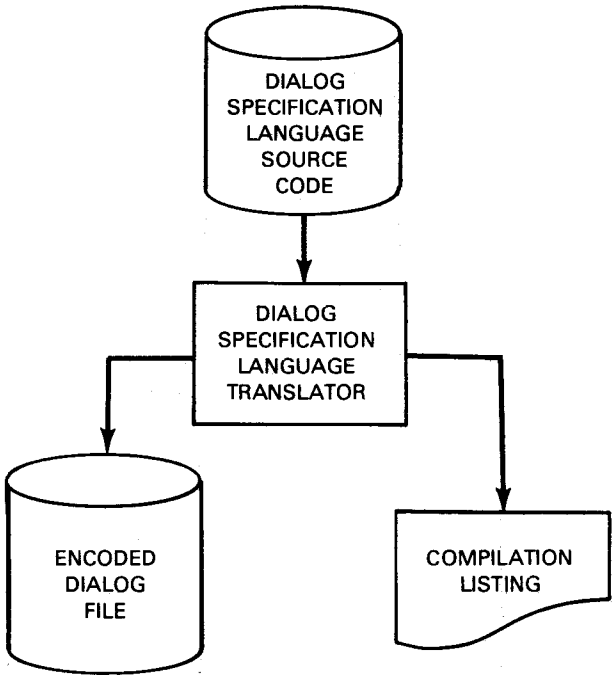


Figure 1—3. Creating an Interactive Dialog

NOTE:

An interactive system recognizes input from both workstations and remote communications terminals. Whenever this manual refers to workstations, those references include both local workstations and remote terminals that use interactive services through a communications network. When using the dialog processor on a communications terminal, be sure to use the field-protect feature on the terminal; otherwise, you will encounter errors.

2. Using the Dialog Processor

2.1. ACTIVATING THE DIALOG PROCESSOR: APPLICATION PROGRAM

Any application program in your operation that would be enhanced by using a dialog to solicit user input can call a dialog. A program that calls a dialog can be written in any language. Existing programs that use another medium, such as cards, for input can be used to call a dialog without changing the programs.

Let's say you have a COBOL program that reads a card file (CARDIN) and prints a report. By using the appropriate job control statements, that application program can be made to solicit input from a dialog session at the workstation rather than from a card reader – without changing the program. When the program is executed, an instruction to open the file CARDIN activates the dialog processor, which opens all the files needed for the dialog session. The dialog processor executes the dialog, accepts user input from the workstation, and routes that input (in the form of dialog output records) to the buffer area identified in the application program. The dialog processor then transfers control back to the application program, and the next program instruction is processed. A program instruction to close the file CARDIN ends the dialog session and turns the dialog processor off.

If your system does not have enough main storage space to process your dialog, the dialog processor will display the message DP082 INSUFFICIENT MEMORY and terminate the job that called the dialog. If you receive this message, wait until all other jobs terminate before re-executing the job.

When running a job that calls an exceptionally large dialog, we suggest you wait until all other jobs terminate before executing that job. This will prevent the dialog processor from displaying DP082 and terminating your job because of insufficient main storage.

Substitution of a dialog session for card input is made possible through OS/3 job control, which we'll discuss in 2.2. Briefly stated, it is made possible through a device assignment set for the workstation that includes the DVC, USE, and LFD job control statements.

2.2. IDENTIFYING THE DIALOG PROCESSOR TO THE SYSTEM: JOB CONTROL

OS/3 requires every job to have a job control stream that directs the execution of the job and identifies the devices and files needed by that job. To better understand this discussion, you should be familiar with the OS/3 job control user guide, UP-8065 (current version). We'll be talking only about job control device assignment sets (DVC through LFD sequences) that identify the workstation, dialog processor, and the dialog you need for a program that calls a dialog.

Let's use the example we outlined in 2.1. You have a program that uses a card file (CARDIN) as input. You decide to enter your input interactively by means of a dialog session, rather than through the card reader. You write a dialog (DIALOG1) to solicit user input. To call DIALOG1 without changing your existing program, you prepare a job control stream that identifies the dialog processor, the dialog file, and the workstation, like this:

```
        .
        .
        .
// DVC 60
// VOL DSK01
// LBL DSLTOUT
// LFD DIALOG1
        .
        .
        .
// DVC 200
// USE DP,DIALOG1
// LFD CARDIN
        .
        .
        .
```

Device assignment set for your dialog file

Device assignment set for the workstation

NOTES:

1. *The device assignment sets for the devices and files referenced in a USE statement must precede the device assignment set that contains that statement.*

2. *Only use USE job control statement may be specified per device assignment set.*

The workstation logical unit number (200) we used in our DVC statement identifies a general type of workstation. Table 2-1 lists all the decimal logical unit numbers for workstations used in OS/3. The OS/3 job control user guide, UP-8065 (current version) contains a complete logical unit number table that lists all the devices that can be used in OS/3, their device type codes, and their decimal logical unit numbers.

*Table 2—1. Workstation Logical Unit Numbers**

Decimal Logical Unit No.	Workstation Type
200 through 215	Any workstation
216 through 219	Any workstation with 24 X 80 screen

**For a complete list of OS/3 logical unit numbers, see the OS/3 job control user guide, UP-8065 (current version).*

The USE statement ties the workstation and dialog file to the application program. Note that the *lfdname* (CARDIN) specified in the LFD statement of the workstation device assignment set must match the file name specified in your application program. You can see the relationship between the application program and the job control stream in Figure 2–1.

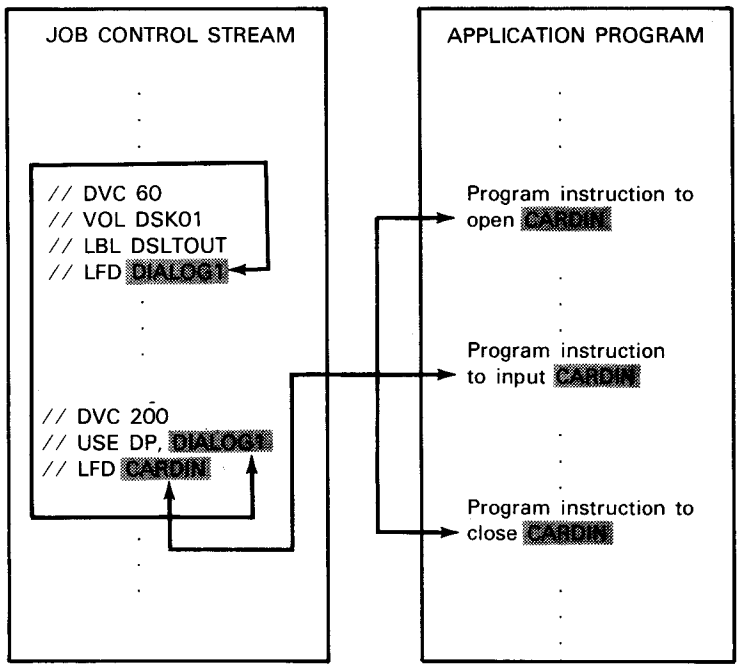


Figure 2—1. Relationship between the Job Control Stream and Application Program

Because of the way we've prepared our control stream for this example, we're able to substitute a dialog session at the workstation for input from the card reader.

The USE statement also allows you to specify a printer and new and old audit files, which we'll cover in our discussion of the audit version of the dialog processor (3.1).

The format of // USE and an explanation of its parameters follow:

```
//[symbol] USE DP,dialog-name[,printer-lfd]
[,new-audit-lfd][,old-audit-lfd]
```

where:

`symbol`
: Is a 1- to 8-character alphanumeric character string, of which the first character must be alphabetic; used as the target of a branching statement.

`DP`
: Tells the operating system that the dialog processor is needed for this job and must be loaded into main storage.

`dialog-name`
: Is the user-specified name for a dialog. One to eight alphanumeric characters in length; the first character must be either an alphabetic or a special character.

`printer-lfd`
Specifies the lfdname of the printer and indicates that a printed summary of the dialog session is to be produced. One to eight alphanumeric characters in length; the first character must be either an alphabetic or a special character.

`new-audit-lfd`
Specifies the lfdname of the new audit file to be produced as output by the audit version of the dialog processor and indicates to the system that the audit version of the dialog processor is to be used. One to eight alphanumeric characters in length; the first character must be either an alphabetic or a special character.

`old-audit-lfd`
Specifies the lfdname of the old audit file to be used as input to the audit version of the dialog processor and indicates to the system that the audit version of the dialog processor is to be used. One to eight alphanumeric characters in length; the first character must be either an alphabetic or a special character.

NOTE:

The USE job control statement is also used to identify the OS/3 screen format services and menu services to the operating system. For the complete format and description of // USE, see the OS/3 job control user guide, UP-8065 (current version).

2.3. HOW THE DIALOG PROCESSOR MANAGES A DIALOG SESSION

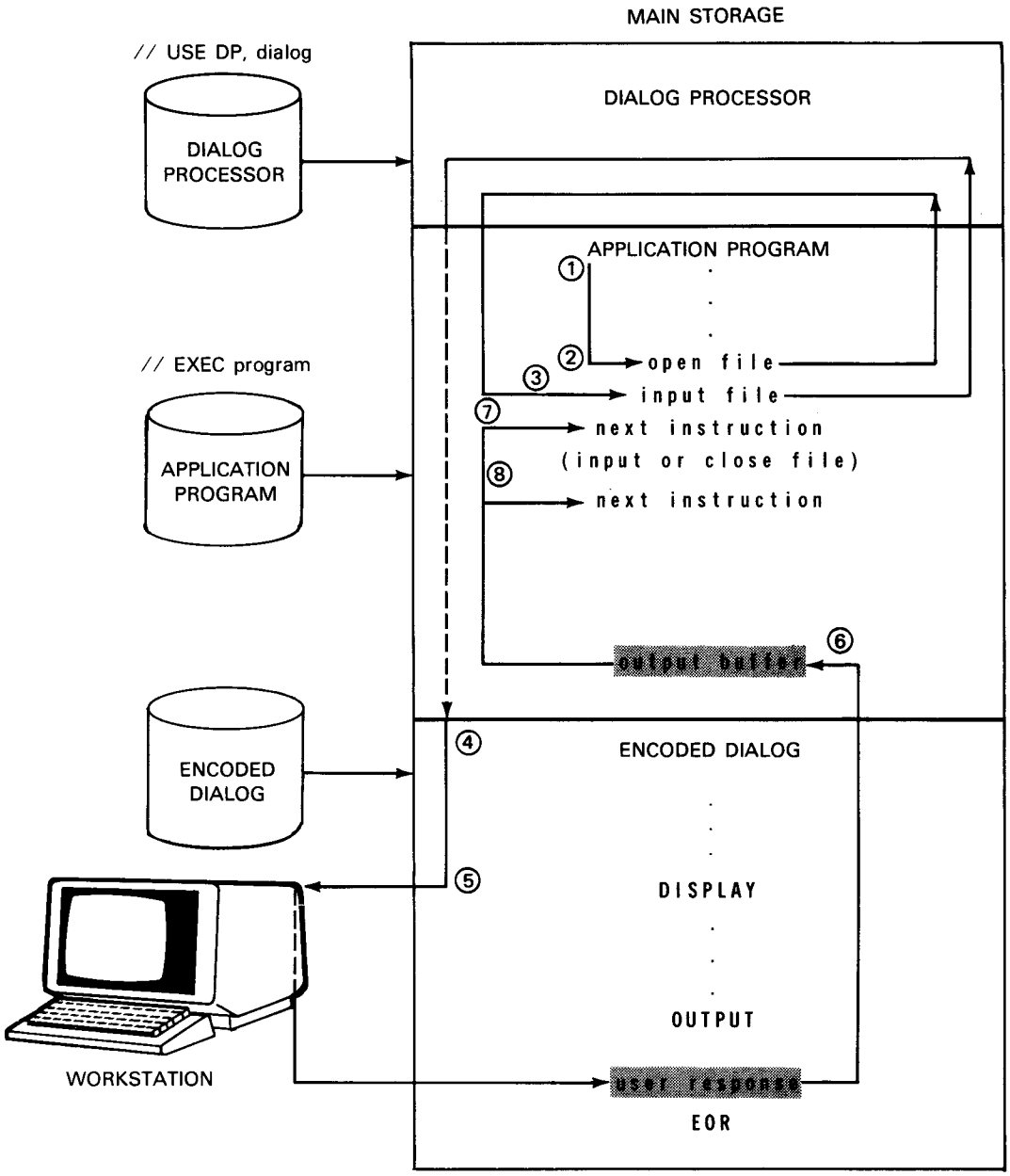
We've already discussed in some detail how the dialog processor passes control between the different elements of a job that uses an interactive dialog. Let's review, step by step, the processing cycle that calls the dialog processor, which then manages the dialog session.

Figure 2–2 is numbered to correspond to the following discussion.

① Your system processes your job control stream first; loads the dialog processor, the application program, and the dialog; and executes the application program. ② When the program encounters an instruction to open files, the system hands control to the dialog processor. The dialog processor then opens the dialog file and all other workstation files in your control stream and returns control to your program. ③ When the program encounters an input instruction, the system again gives control to the dialog processor, which ④ executes the dialog. ⑤ A dialog specification language DISPLAY command in the encoded dialog file directs the dialog processor to display the dialog text at the workstation screen. A dialog specification language OUTPUT command with EOR specified directs the dialog processor to create an output record containing those responses, ⑥ route them to the output buffer area identified in the application program, and then return control to your program. ⑦ Your program processes the next instruction; if it is another input instruction (dialog response), your program repeats the cycle beginning at step 3; if it is an instruction to close the file, the system closes the dialog file, turns the dialog processor off, and ends the session. ⑧ Execution of the program continues and your program processes the next instruction.

NOTE:

A dialog session also ends if the workstation user presses the end-session function key (F4).



NOTE:

Processing steps are numbered to correspond to the discussion in 2.3.

Figure 2—2. Processing Steps for a Program That Calls the Dialog Processor

Figure 2-3 shows dialog processor inputs and outputs.

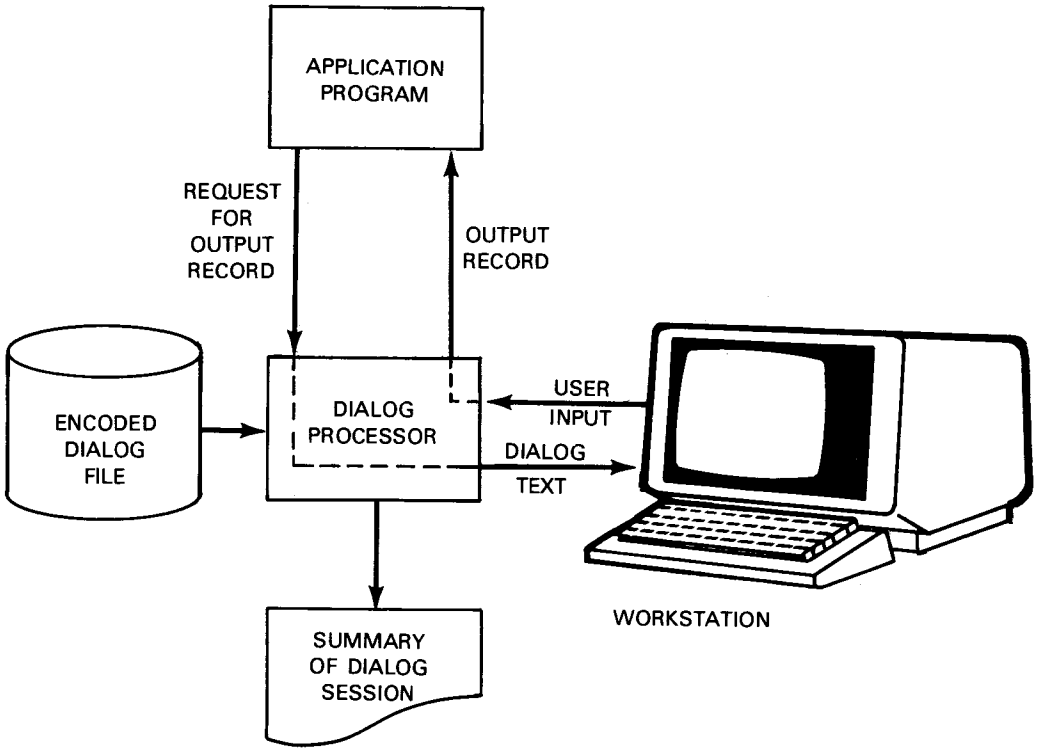


Figure 2—3. The Dialog Processor: Input and Output

2.4. DIALOG SESSION SUMMARY

Besides creating output records that contain your responses to a dialog, the dialog processor also produces a printed summary of the dialog session organized by sequentially numbered paragraphs. A dialog paragraph represents one execution of a dialog specification language tree and may contain many user responses to the dialog. If you decide to change your responses in a subsequent dialog session, the printed summary organized by sequentially numbered paragraphs is your guide to telling the dialog processor which paragraphs you want to change. If you want a printed summary of the dialog session, specify the *printer-lfd* parameter of the USE job control statement and a device assignment set for the printer in the control stream for the job.

Distinct dialog specification language commands in the dialog that you write (or in those Sperry Univac supplies) inform the dialog processor of what should appear in the printed summary listing. The dialog specification language user guide/programmer reference, UP-8806 (current version) explains how to write an interactive dialog.

Figure 2-4 shows a typical example of a printed summary listing for a sample dialog session. Assume that you have already written your dialog and it resides in the encoded dialog file. Notice the relationships among the dialog specification language commands in the sample dialog file, the workstation display screens, your responses, and the printed summary listing paragraphs, numbered in accordance with the dialog file paragraphs. The values in reverse print represent sample responses that you key in.

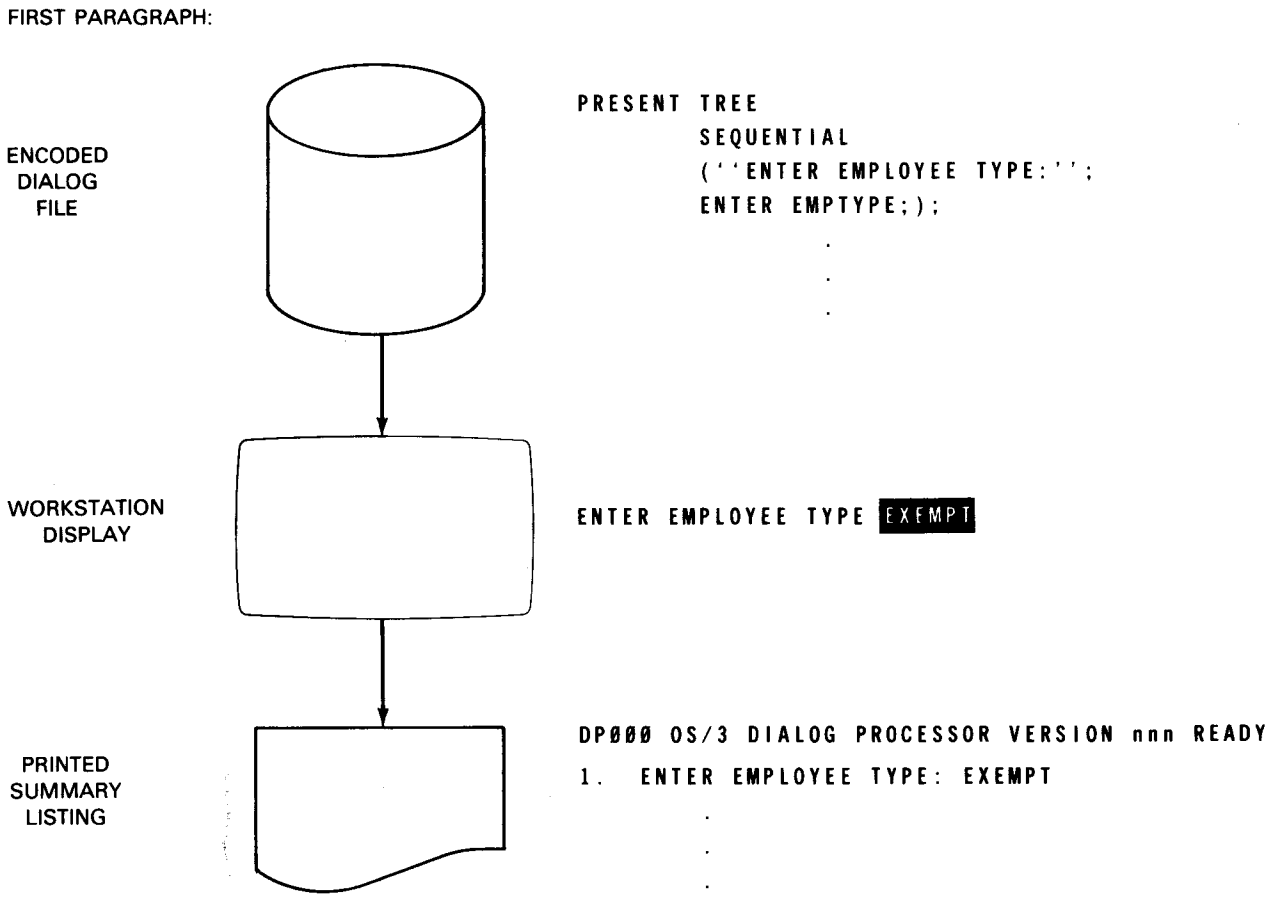
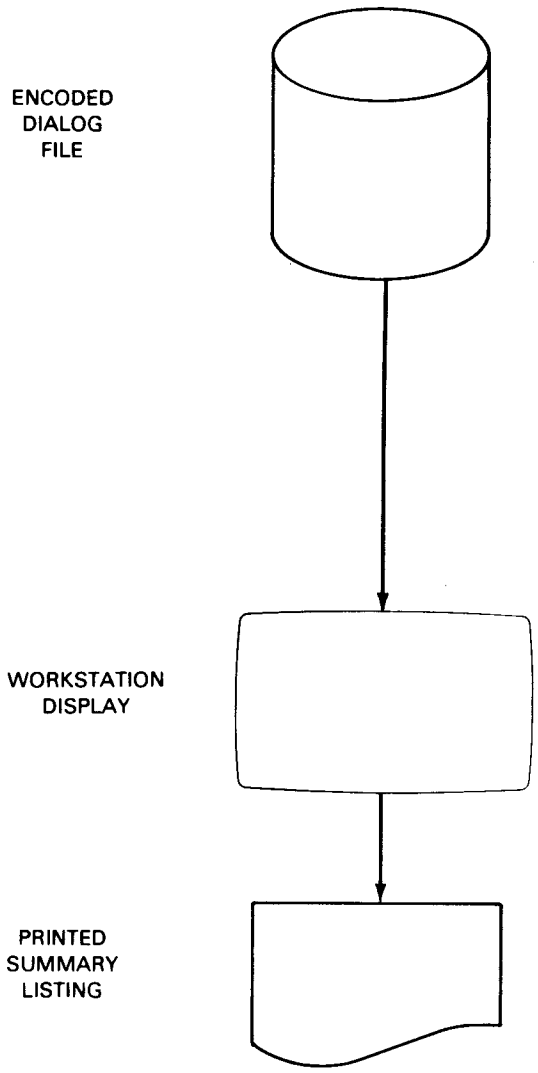


Figure 2—4. Sample Summary Listing (Part 1 of 2)

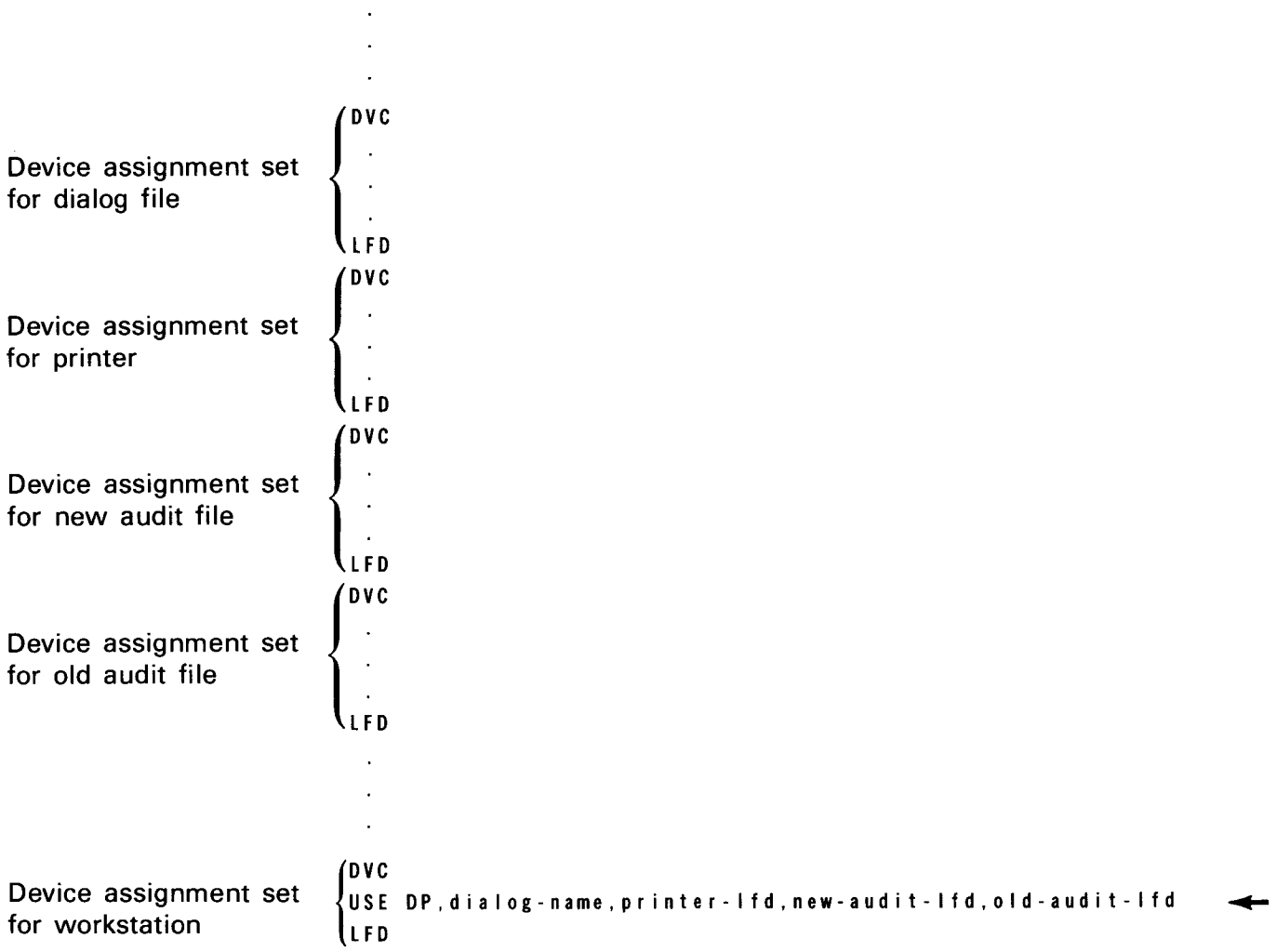
SECOND PARAGRAPH:




```
.
.
.
PRESENT TREE
        PARALLEL
        (DISPLAY ''ENTER EMPLOYEE NAME:'';
        PRINT ''EMPLOYEE NAME:'';
        ENTER EMPNAME;)
        (DISPLAY ''ENTER STREET NAME:'';
        PRINT ''ADDRESS:'';
        ENTER STNAME;)
        (DISPLAY ''ENTER CITY NAME:'';
        ENTER CITYNAME; NEWLINE;
        DISPLAY ''ENTER STATE AND ZIP CODE:'';
        ENTER STATE;
        ENTER ZIPCODE;);
.
.
.
ENTER EMPLOYEE NAME: DAVID SMITH
ENTER STREET NAME:   500 MAIN STREET
ENTER CITY NAME:     NEW YORK
ENTER STATE AND ZIP CODE: NY  10000
.
.
.
2.  EMPLOYEE NAME: DAVID SMITH
                   500 MAIN STREET
                   NEW YORK, NY  10000
```

Figure 2—4. Sample Summary Listing (Part 2 of 2)

When you prepare your job control stream, you must specify the device assignment set that contains the USE statement after the device assignment sets for the devices and files referenced in that USE statement, like this:



3. Changing Dialog Responses

3.1. HOW THE AUDIT VERSION OF THE DIALOG PROCESSOR FUNCTIONS

The audit version of the dialog processor performs all the functions we described in Section 2 and, in addition, provides the means to change the responses you made to a dialog in a previous session. This version of the dialog processor produces an audit file that contains your dialog responses from the previous session. The audit file can then be used as input to the audit version of the dialog processor in a subsequent session that uses the same dialog. As an output file, the audit file is the new audit file; as an input file, it is the old audit file. The audit version of the dialog processor also produces a printed summary of the dialog session if you've requested it through the job control stream.

OS/3 uses the audit version of the dialog processor if you specify that an audit file is to be output (new audit file), or input (old audit file), or both. You specify audit files through device assignment sets in the job control stream and through the USE statement in the device assignment set for the workstation. Again, the format of // USE is:

```
//[symbol] USE DP,dialog-name[,printer-lfd][,new-audit-lfd][,old-audit-lfd]
```

Why would you use the audit version of the dialog processor? You might need it if you have a lengthy dialog used nearly every day in your operation that requires only a few changes each time it's used. With the audit version, you can transact a subsequent dialog session very quickly, automatically outputting user responses from the last dialog session that do not need to be changed and entering new responses to the dialog only where they're needed. The printed summary of a dialog session that contains sequentially numbered paragraphs is your guide to a subsequent editing session. When you're using the audit version of the dialog processor and you've specified that an old audit file is to be used as input, you are asked through a workstation display to key in the numbers of the paragraphs you want to change. The dialog processor then automatically routes dialog responses from the old audit file to the application program and displays text at the workstation only for those paragraphs you've indicated you want to change. At the same time, the dialog processor is creating a new audit file (if you've requested it through the job control stream) that contains a record of the current dialog session, including your responses to the dialog and the responses you chose to keep from the last dialog session. It can then be used as the old audit file for a subsequent editing session. Figure 3-1 illustrates this process.

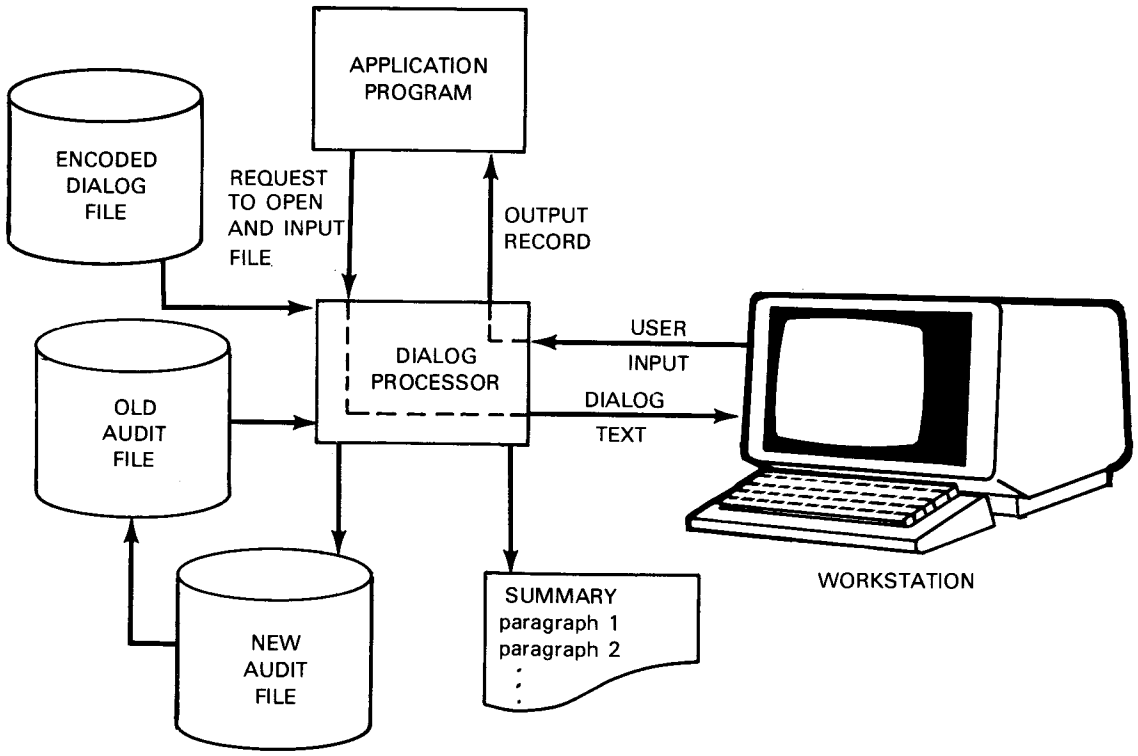


Figure 3—1. Audit Version of the Dialog Processor: Input and Output

Before we explain the operation of the audit version of the dialog processor in more detail, let's discuss the interfaces among it, an application program, a dialog, and the control stream for a job that uses all these elements.

Suppose you have a program (LETTER) that produces a "personalized" form letter. You send the letter to your customers to introduce new products. The variables you can change in the letter are: product description, name of client, client's geographic area, renter status, home-owner status. LETTER opens INPUT1 (workstation file). The dialog processor then executes LETRDLG (the dialog) and accepts your input from the workstation. Suppose you produced 100 letters yesterday and you want to produce an identical run today, except that you want to introduce a different product and remove references in the letter to geographic area and home-owner/renter status. We're assuming, of course, that the dialog is constructed to allow you to make these choices. We are also assuming that when you ran the program yesterday, you used the audit version of the dialog processor and created an audit file and summary of the dialog session. All you need to do today is use the audit version of the dialog processor to change the appropriate dialog responses from yesterday's session.

The control stream for your job could look like this:

```
// JOB NEWLTR
// DVC 20 // LFD PRNTR
// DVC 60 // VOL DSK01 // LBL DSLTOUT // LFD LETRDLG
// DVC 60 // VOL D02345 // LBL AUDIT2 // LFD AUDIT2
// DVC 60 // VOL D02345 // LBL AUDIT1 // LFD AUDIT1
// DVC 200 // USE DP,LETRDLG,PRNTR,AUDIT2,AUDIT1 // LFD INPUT1
// EXEC LETTER
/&
```

The first device assignment set identifies the printer file; the second, the encoded dialog; the third, the new audit file; the fourth, the old audit file; and the last, the workstation. Because you've specified audit files in the control stream, OS/3 knows you need to use the audit version of the dialog processor. You can, of course, create your job control streams interactively by using the SPERRY UNIVAC job control dialog, explained in the interactive job control user guide, UP-8822 (current version).

When LETTER opens INPUT1, the dialog processor displays a message asking you to enter the numbers of the paragraphs you want to change. For unchanged paragraphs, the dialog processor takes dialog responses from the old audit file and routes them to the application program automatically. When the dialog processor encounters a paragraph you've indicated you want to change, it displays your responses from yesterday's dialog session at the workstation screen. You can accept the old paragraph or edit it. If you decide to edit it, you enter your new choices. For our example, you enter new responses for the product description paragraph, and you delete the responses you made yesterday concerning geographic area and home owner/renter status.

At the end of the dialog session, you will have created a new audit file containing a complete record of today's dialog session, and the dialog processor will have routed dialog responses from the old audit file and today's session to the application program. A printed summary of today's session is also produced. The result of the dialog session is that you have created a new letter by changing only a few of the dialog responses you made yesterday.

If you want to transact a dialog session and create a new audit file for a subsequent editing session, but you do not want to use the old audit file as input to the dialog processor for the current dialog session, you specify only the *new-audit-lfd* parameter of the USE job control statement. Using our previous example, the USE statement would look like this:

```
// USE DP,LETRDLG,PRNTR,AUDIT2
```

If, on the other hand, you want to transact a dialog session using the old audit file as input to the dialog processor for the current session, you specify only the *old-audit-lfd* parameter of the USE job control statement. For our example, the USE statement would look like this:

```
// USE DP,LETRDLG,PRNTR,,AUDIT1
```

Note that the comma that precedes the *new-audit-lfd* parameter must be retained in the USE statement even though you haven't specified the parameter itself.

The audit version of the dialog processor operates in three modes. These modes are: automatic, edit, and dialog. They are explained in the following paragraphs.

3.1.1. Automatic Mode

When the audit version of the dialog processor is being used, the dialog processor executes in automatic mode until it encounters a choice that cannot be resolved from the old audit file, encounters a paragraph you've indicated you want to change, or encounters the end of the encoded dialog file. Unresolved choices occur if you've changed one dialog response, but haven't changed the responses that resulted from the original choice.

In automatic mode, user responses from the last dialog session are taken from the old audit file and routed to the application program without your intervention.

When the dialog processor encounters a paragraph you've indicated you want to change or an unresolved choice, it switches to edit mode and displays the paragraph at the workstation screen.

Most of the choices you can make when you change a dialog paragraph are accomplished by pressing function keys. Table 3-1 lists these keys.

Table 3—1. Workstation Function Keys

Edit Option	Workstation Key
Review	F1
Cancel	F2
Reedit	F3
End session	F4
Insert	F5
Delete	F6
Edit	F7

3.1.2. Edit Mode

In edit mode, there are four locations where you can make paragraph changes: at the beginning of a paragraph, at a data entry field, at a choice field, and at the end of the paragraph.

At the beginning of the paragraph, you can:

- proceed by pressing the TRANSMIT key (your responses are routed from the old audit file to the application program – you make no changes to the paragraph);
- end the session by pressing function key F4; or

- edit (change) your responses to the dialog paragraph by pressing function key F7, which moves you to the first data entry field.

When you are at the data entry field in a paragraph, you can:

- Edit – enter new data and then move the cursor to the next data entry field, or you can move the cursor to the next data entry field without changing the first one.
- Proceed – no change; the dialog processor then moves to the next choice field in the paragraph.
- Review – press the F1 function key to see a summary of your changes to the paragraph displayed at the workstation screen. The cursor is positioned at the first choice field, and you can then choose one of the actions available to you when the dialog processor is at a choice field.

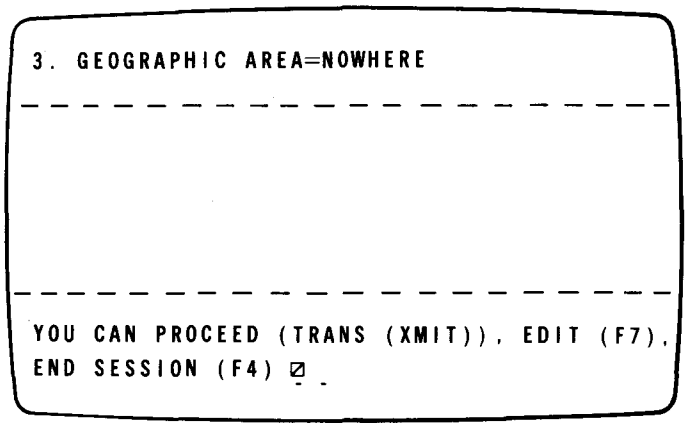
When the dialog processor is at a choice field in the paragraph, you have the widest range of possible actions. You can:

- Proceed – accept the indicated choice, and any lower-level choices automatically, by pressing the TRANSMIT key.
- Edit – accept the indicated choice by pressing function key F7. This allows you to change the choice field within the paragraph.
- Delete – delete the indicated choice by pressing the delete function key (F6). If there are no more acceptable choices in the old audit file, then the dialog processor enters the dialog mode (3.1.3.) to solicit a valid choice.
- Insert – insert a new choice by pressing the insert function key (F5). This causes the dialog processor to enter the dialog mode and solicit new choices.
- Cancel – cancel everything that has been done in the new paragraph by pressing the cancel function key (F2). The dialog is restarted from the beginning of the paragraph with the old choices intact. This function is used if the new paragraph you're creating contains many errors.
- Reedit – if the new paragraph you're creating contains just a few errors, you can press the reedit function key (F3) to correct them. The dialog processor then discards the old paragraph and replaces it with the new paragraph you've created so that you can then edit it. It is edited from the beginning of the paragraph.

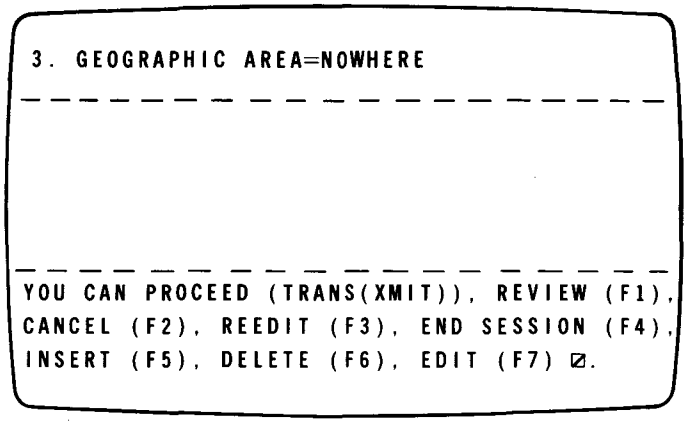
If you make any changes in edit mode, the dialog processor pauses at the end of the paragraph and gives you a final chance to make one of the following choices (which we have already described): proceed, cancel, reedit, or end the session.

Using our previous example of the program LETTER, suppose you want to use the audit version of the dialog processor to change your responses to paragraph 3 of your last dialog session. The response you want to change is that of your client's geographic area. The dialog processor requests the paragraphs you want to change and automatically takes your responses from the old audit file until it reaches the first paragraph you want to change (paragraph 3 in our example). At that point, the dialog processor displays your previous choices to paragraph 3 and gives you the option to proceed, edit, or end the session. If indeed, you need to change one of the choices, you inform the dialog processor that you want to edit the coices, using the appropriate function key described in Table 3-1. The dialog processor then enters edit mode, as shown in the following sample audit session. In this example, your responses are shown as white letters on a black background.

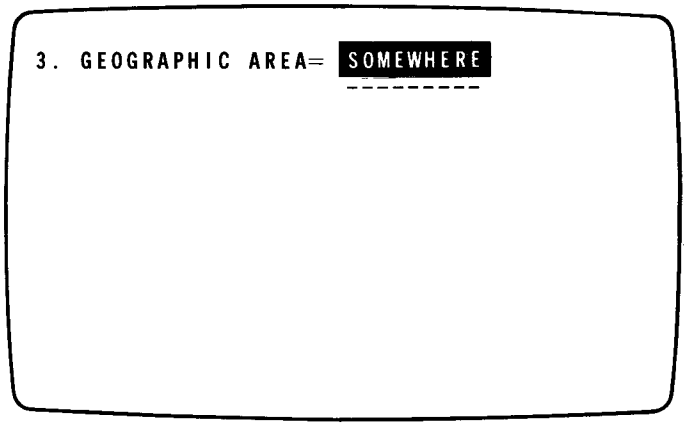
① Response from paragraph 3 of old audit file (NOWHERE) and your list of options with cursor positioned in the correct entry field. You want to edit, so you press the edit function key (F7).



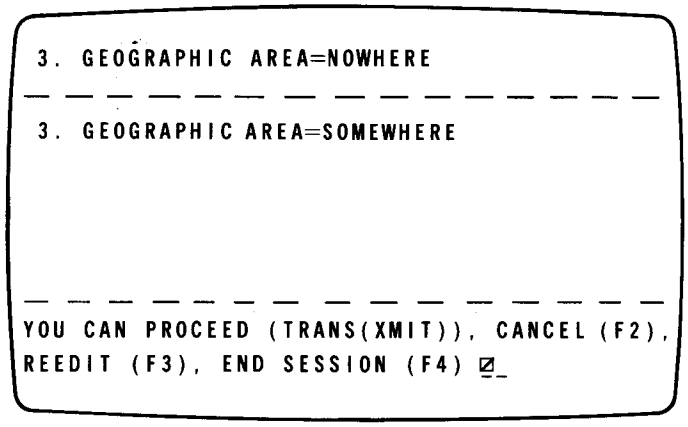
② Dialog processor displays your editing options; you choose to edit an old response, so you press the edit function key (F7).



③ The cursor repositions itself to accept your edited response so you key in your new response (SOMEWHERE).



④ The dialog processor displays your new response and gives you a final chance to cancel, reedit, or end session.



3.1.3. Dialog Mode

Certain situations in automatic mode or edit mode cause the audit version of the dialog processor to switch to dialog mode. At that point, the dialog processor begins processing the encoded dialog file, rather than the old audit file. In general, the dialog processor switches to dialog mode if you request it or if it encounters a choice in the old audit file that can no longer be resolved because you have changed higher-level choices.

The dialog processor remains in dialog mode until:

- a valid choice is made or a valid choice is available from the old audit file (at that point, the dialog processor returns to automatic or edit mode); or
- you press the review function key. In that case, the dialog processor enters edit mode.

3.2. SESSION BREAK

What happens if you are using the audit version of the dialog processor to edit a previously transacted dialog and you're interrupted before you can finish the session? You will not lose the work you have already done. When you press the end-session function key, the dialog processor places an end-session field in the new audit file you are creating, at the point where you break off the editing session. This discontinuity is indicated by a dotted line in the printed summary of the dialog session output by the dialog processor. You must have requested a new audit file and a printer file in the job control stream to use this feature of the audit version of the dialog processor.

To resume the session, specify that the audit file you were producing when the session was interrupted be input to the dialog processor as the old audit file.

Glossary

Italicized terms are cross-references to terms defined in this glossary.

A

audit file
Optional output file produced by the audit version of the *dialog processor.* It contains a complete record of a *dialog session.*

D

dialog
See *interactive dialog.*

dialog paragraph
One execution of a dialog tree.

dialog processor
A shareable software program that manages *interactive dialogs.* It displays dialog text at the workstation screen, accepts user input, and routes that input to the appropriate system or application program. The dialog processor is identified to the system through the USE job control statement.

dialog processor - audit version
The version of the dialog processor that creates an audit file. The audit version is used if you want to change your responses to a dialog in a subsequent session.

dialog session
A single execution of the *dialog processor.*

dialog specification language
A high-level language designed for creating interactive dialogs.

dialog specification language block
The basic structure used in a dialog specification language program. A block is delimited by DO and END commands.

dialog specification language translator
The component that compiles dialog specification language source code and outputs encoded dialogs, which are stored in the encoded dialog file.

dialog tree
A specialized program structure having three key elements:

Trunk – dialog commands executed with each branch
Node – a decision point in a dialog; specifies how branches are selected
Branch – a set of dialog specification language commands that request input, display messages, produce output, and perform other processing functions

E

encoded dialog file
File where encoded dialogs output by the dialog specification language translator are stored.

I

interactive dialog
A "conversation" between the workstation user and the operating system. The purpose of an interactive dialog is to solicit input from the workstation user that is used in a complementary system or user program.

interactive processing
A computing environment where you communicate directly with the operating system through a *workstation.*

N

new audit file
An optional *audit file* output by the audit version of the dialog processor that contains a record of a dialog session.

O

old audit file
An audit file (originally produced as a *new audit file*) input to the audit version of the dialog processor in a subsequent dialog session for the purpose of changing or editing a previously transacted *dialog.*

output record
A record created during a dialog session. The *dialog processor* accepts user input at the *workstation* and routes this input (in the form of an output record) to the appropriate system or application program.

P

paragraph
See *dialog paragraph.*

S

summary report
An optional printed report output by the dialog processor that contains a summary of a dialog session organized by sequentially numbered paragraphs.

W

workstation
A terminal device with a screen for display of dialog text and a keyboard for entry of user input.

Index

SPERRY UNIVAC

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

_____

(Document Title)

_____ _____ _____

(Document No.) *(Revision No.)* *(Update No.)*

Comments:

Cutting line.

From:

_____

(Name of User)

_____

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424

FOLD