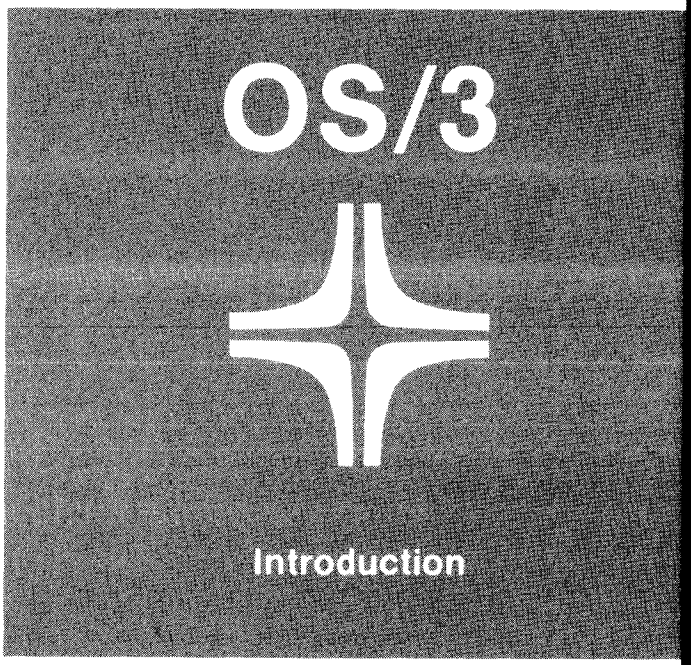
1974 American National Standard COBOL



SPERRY UNIVAC

UP-8804 Rev. 1

H

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 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.

what is COBOL?

In order to use a computer, you must direct it with some system of instructions. These instructions are called programming languages, and many languages were developed to help users handle various data processing needs. One such language is COBOL, which stands for *CO*mmon *B*usiness *O*riented *L*anguage. With COBOL, you can express procedures for solving problems in English-like statements. COBOL is specifically designed for programming computers to do business data processing, and it is one of the most widely used languages for this purpose.

The SPERRY UNIVAC 1974 American National Standard COBOL is based on the American National Standard COBOL, X3.23-1974, developed by the American National Standards Institute (ANSI). This standard is used by the federal government and has gained wide acceptance in industry. SPERRY UNIVAC '74 COBOL conforms in all respects to this standard.

what it can do for you

The '74 COBOL we provide has all the capabilities you need to:

- define and access files;
- define and generate reports;
- define tables and access items within them;
- identify and manipulate indexes;
- monitor data during program testing;
- sort and merge data files;
- transfer data and control to and from sub-programs;
- send messages to and process messages received from local and remote terminals;
- bring predefined and stored COBOL text into a program; and
- divide programs into segments to conserve main storage.

structure

The COBOL programs you write consist of four divisions. These divisions are:

1. Identification Division

   In this division, you label your program. As an option, you may also specify your name, the name of your installation, the date you wrote and compiled the program, and remarks concerning security.

2. Environment Division

   You use the environment division to describe the computer used to execute your program, assign input/output devices to files, and specify the techniques to process files.

3. Data Division

   This division describes the data operated on by the program. You describe the structure of your data files and define how you want your reports to look.

4. Procedure Division

   You use this division to define the logical steps to solve a given problem and to describe the procedures you want executed only under special circumstances (conditions that occur infrequently or what to do when an error occurs).

You write these divisions in the order listed following certain formats and rules.

how to use it

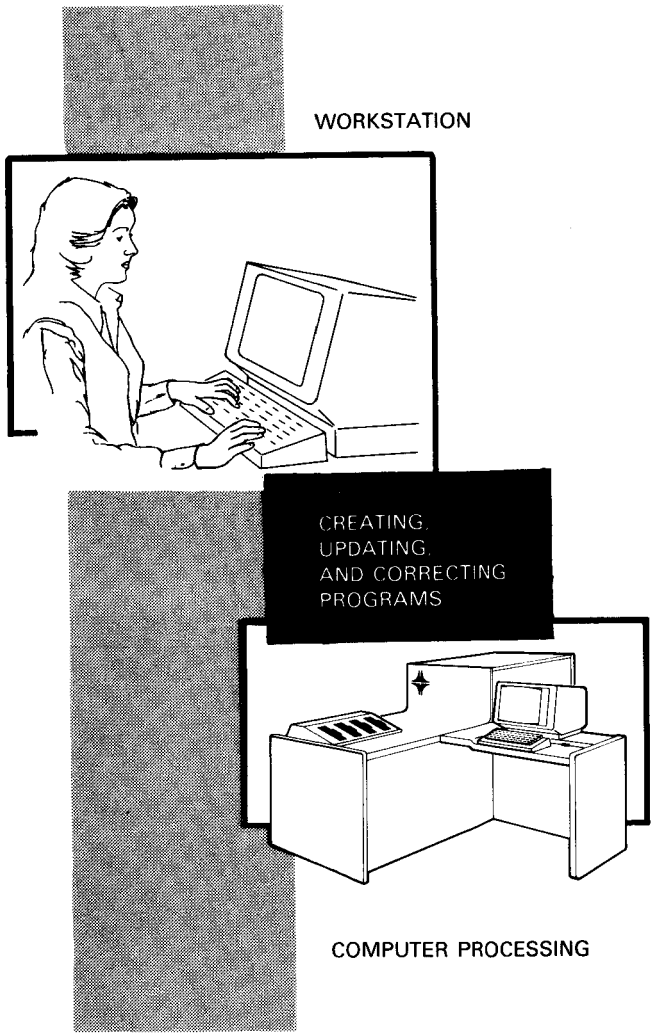
You have a business-oriented problem and you want to solve it. First, you establish the goals of your program to get a clear idea of the procedures that will yield the best answer to your data processing problems. The program you write is called a *source program*; it must contain the four divisions that comprise all COBOL programs. You can enter your source program into the computer either from an interactive terminal called a *workstation (interactive processing)* or by diskette or keypunched cards *(batch processing)*. In either case, you also enter *job control statements* that tell the computer exactly how you want your COBOL programs executed and what resources are needed (tapes, disks, etc).

interactive processing

Traditionally, COBOL programs are written on coding forms and then keypunched. The interactive facility allows you to create and update programs without using coding forms or punched cards.

If you decide to enter your program interactively, you use the keyboard at a workstation to enter your program and job control statements. The workstation is the primary input device; it simplifies coding.

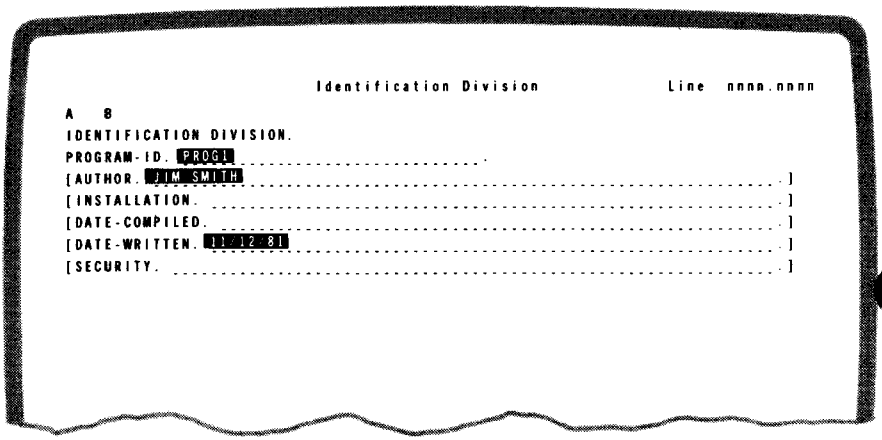
INTERACTIVE PROCESSING



The COBOL editor (COBEDT) provides screen formats that correspond to the COBOL coding divisions. Using these formats, you write your COBOL programs by inserting information on the format. What you enter is displayed on the workstation screen. There are two types of screen formats:

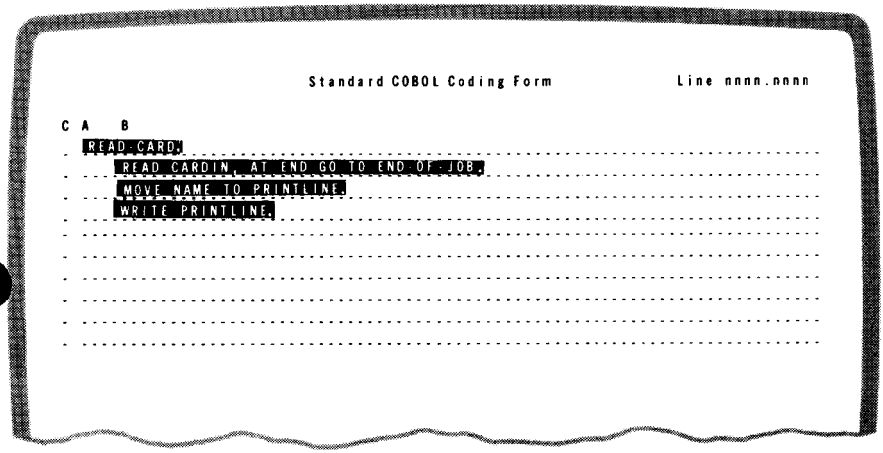
1. Ordered Display

   When you want to create a COBOL program, use the ordered display, which is a series of standard COBOL screens. After each screen appears, you fill in the information needed to complete the statement. You can eliminate any optional lines by not making any entries on them; all required information is automatically included. The following example shows sample entries on an ordered display:



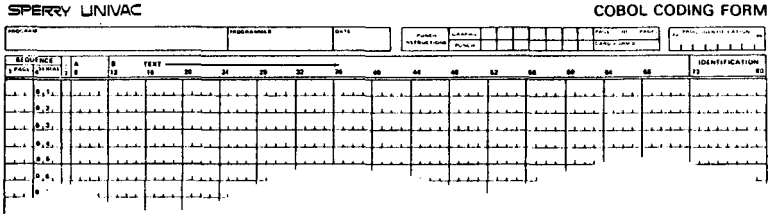
2. Selective Display

When you want to create or update a selected portion of the program, use the selective display, which is similar to a standard COBOL coding form. The following example shows entries on a selective display:



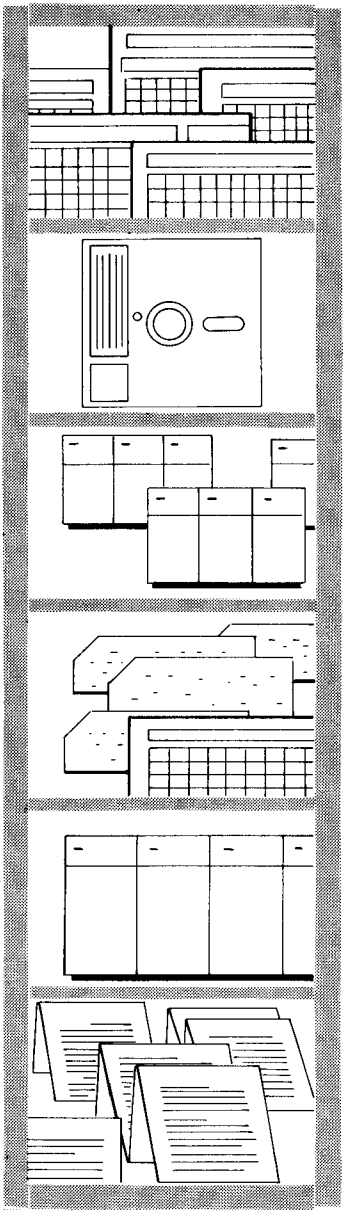
batch processing

If you decide to use batch processing, you write your COBOL statements on coding forms:



Then, you enter your source program and job control language by diskette, punched cards, disk, or tape. If you make any typographical or formatting errors, you can't correct them until after your program is compiled. After all the errors are corrected, you can execute your program.

BATCH PROCESSING



CODING

DISKETTE

COMPUTER
PROCESSING

CORRECTING
ERRORS

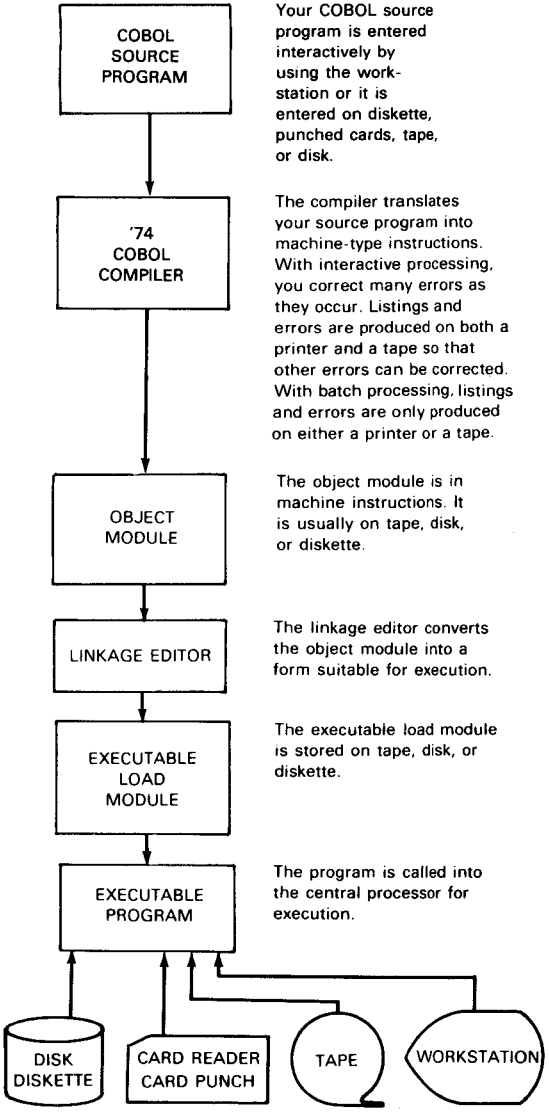
COMPUTER
PROCESSING

OUTPUT
LISTING

generation of a COBOL program

Regardless of whether you use interactive or batch processing, the source program must be translated into machine instructions understood by the computer. This translation is done by the COBOL *compiler* supplied by Sperry Univac, which analyzes the source program and produces an *object module*. The object module is functionally equivalent to the source program, except that it is in machine instructions rather than COBOL statements. You link-edit the object module using a system utility called the *linkage editor*. The resulting load module is now ready for execution.

GENERATION OF A COBOL PROGRAM



Your COBOL source program is entered interactively by using the work-station or it is entered on diskette, punched cards, tape, or disk.

The compiler translates your source program into machine-type instructions. With interactive processing, you correct many errors as they occur. Listings and errors are produced on both a printer and a tape so that other errors can be corrected. With batch processing, listings and errors are only produced on either a printer or a tape.

The object module is in machine instructions. It is usually on tape, disk, or diskette.

The linkage editor converts the object module into a form suitable for execution.

The executable load module is stored on tape, disk, or diskette.

The program is called into the central processor for execution.

When it is executed, the program may use a variety of input/output devices.

advantages

Why is COBOL so popular? Because it has the following advantages:

- *English-like statements* make the programs easy to code, understand, and use. Extra documentation is often not needed.

- *Upward compatibility* with future acquisitions of larger SPERRY UNIVAC computer systems since COBOL is standardized.

- *Low training time* for new programming personnel because COBOL uses English-like statements.

- *COBOL is input/output oriented* so it lends itself to business applications.

- *Programs are easy to test and correct* because the interactive facility allows you to correct many errors as soon as you make them. With both batch and interactive processing, there is a printed listing of the source statements, an error diagnostic listing, and main storage maps and cross-reference listings.

- *COBOL is flexible* so that you can enter your data on diskette or punched cards, store it, then correct any errors using the interactive facility.

All these advantages make COBOL *easy to use*; that's why it's so popular.

programming aids

SPERRY UNIVAC '74 COBOL provides many aids to make programming easier. With both interactive and batch processing, you can direct the compiler to produce several types of listings to reduce your efforts when writing and correcting COBOL programs:

- Source Code Listing

  Shows the source program as it is compiled. All source statements are listed along with a compiler-generated number that identifies each line.

- Diagnostic Error Listing

  Provides a detailed account of the errors the compiler encounters. The statement line number, an error code, and a message are printed for each error.

- Object Code Listing and External References

  Shows the compiler-generated object code along with relative addresses.

- Data/Procedure Division Storage Map and Cross-Reference Listing

  Shows the layout of main storage for the data and procedure divisions. The cross-reference lists the special registers and user-defined names along with compiler-generated line numbers for reference purposes.

With the interactive facility, you actually see your input displayed at the workstation. Consequently, it's much easier to detect and correct typographical and formatting errors as they are made. Compared to the long and more involved batch procedure, the interactive facility saves you time and is a more efficient method of correcting COBOL programs.

extensions to the standard

SPERRY UNIVAC '74 COBOL conforms in all respects to the ANSI standard. To increase the utility of SPERRY UNIVAC '74 COBOL, we've made many extensions to the standard that make our COBOL even more powerful and useful. See your local Sperry Univac sales representative for documentation on these extensions.

sample program

This sample program illustrates the use of some of the SPERRY UNIVAC '74 COBOL language elements. The program processes a very simple problem, so not all of the features of '74 COBOL are used.

Assume that you want to print labels with a client's name, street, city, state, and zip code on them. The program for doing this is:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. LABELS.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. UNIVAC-OS3.
OBJECT-COMPUTER. UNIVAC-OS3.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT CARDIN, ASSIGN TO CARDREADER-ABC-F.
    SELECT PRINTOUT, ASSIGN TO PRINTER-XYZ-FC.
```

This program is named LABELS in the identification division. In the environment division, the source and object computers are identified as UNIVAC-OS3; the input file (CARDIN) is associated with a card reader; and the output file (PRINTOUT) is associated with a printer.

The data division describes the CARDIN and PRINTOUT files and a working storage area is defined. PIC stands for PICTURE, which describes the general characteristics of a data item. For example, 02 NAME PIC X(25) shows you are allowing 25 locations for the name, which can consist of alphabetic characters, digits, or certain special characters.

```
DATA DIVISION.
FILE SECTION.
FD  CARDIN
    LABEL RECORDS ARE OMITTED.
01  CARD-INPUT.
    02  NAME          PIC X(25).
    02  STREET        PIC X(25).
    02  CITY          PIC X(15).
    02  STATE         PIC X(2).
    02  ZIP           PIC 9(5).
    02  FILLER        PIC X(8).
FD  PRINTOUT
    LABEL RECORDS ARE OMITTED.
01  PRINTLINE         PIC X(25).
WORKING-STORAGE SECTION.
01  CITY-STATE-ZIP-LINE.
    02  CITY-OUT      PIC X(15).
    02  FILLER        PIC X(1) VALUE SPACE.
    02  STATE-OUT     PIC X(2).
    02  FILLER        PIC X(2) VALUE SPACES.
    02  ZIP-OUT       PIC 9(5).
```

```
PROCEDURE DIVISION.
BEGIN-JOB.
    OPEN INPUT CARDIN. OUTPUT PRINTOUT.
READ-CARD.
    READ CARDIN, AT END GO TO END-OF-JOB.
    MOVE NAME TO PRINTLINE.
    WRITE PRINTLINE.
    MOVE STREET TO PRINTLINE.
    WRITE PRINTLINE.
    MOVE CITY TO CITY-OUT.
    MOVE STATE TO STATE-OUT.
    MOVE ZIP TO ZIP-OUT.
    WRITE PRINTLINE FROM CITY-STATE-ZIP-LINE.
    GO TO READ-CARD.
END-OF-JOB.
    CLOSE CARDIN, PRINTOUT.
    STOP RUN.
```

The procedure division contains the English-like statements that specify the procedure for solving the problem. First, CARDIN and PRINTOUT are made available for processing. Then, the punched cards (input file) are read and the name is printed on a line, the street is printed on the next line, and the city, state, and zip code are printed on the third line. After all the punched cards are read, the job is finished and the files are closed. This program produces a label that looks like this:

```
John H. Doe
1114 Stirling Drive
Valley Green    NJ  07603
```

summary

SPERRY UNIVAC 1974 American National Standard COBOL is a comprehensive, versatile language that offers flexibility and ease of control through the use of English-like statements. It is easy to learn and efficient to use. The interactive facility greatly simplifies the process of creating and correcting programs. Such power and ease of use have made COBOL a widely used business language.