TECHNICAL BULLETIN

Name OS/3 BULLETIN NO. 12

Order No. UP-8605.12 Rev. 1

OS/3 ANS'74 COBOL Communications Technical Bulletin

This bulletin contains information on:

COBOL COMMUNICATIONS

This document describes restrictions and provides guidelines to be followed when using the communications facilities of the COBOL language (SEND, RECEIVE, input CD, output CD, ...) to interact with communication devices via ICAM.

Request additional copies by submitting a Sales Help Requisition form (UD1-578) through your local Sperry Univac representative to:

Customer Information Distribution Center (CIDC)
Sperry Univac
555 Henderson Road
King of Prussia, PA 19406

DESTRUCTION NOTICE:

OS/3 ANS'74 COBOL Communications Technical Bulletin UP-8605.12 Rev. 1 replaces Technical Bulletin UP-8605.12 dated August 1981. Please destroy all copies of UP-8605.12 dated August 1981.

Lists:	Bulletin No.:	Date:
18, 19, 20, 21, 28U, 29U, 75, 76 CZ, SBZ, SWE	12	January 1982

The following Technical Bulletins are published for the OS/3 System. Current items are identified with an ``*'' in column 1; scheduled items are identified with an ``**'' in the date column:

SYSTEM	RELEASE #	DATE	ORDER #	ITEM and DESCRIPTION
*OS/3	ALL	12/78	UP-8605.6	OS/3 Technical Bulletin #6 (This document contains information on the use of IMS 90 Multi-Thread.)
*OS/3	ALL	3/78	UP-8605.7	OS/3 Technical Bulletin #7 (This document contains information concerning techniques for processing unordered IRAM files.)
*OS/3	5.2/5.2.1 6.0	5/79	UP-8605.9	OS/3 Technical Bulletin #9 (This document contains information on the use of the IBM 3741 MEDIA COMPATIBILITY UTILITY for the UTS 400; this utility is available with Releases 5.2/5.2.1 and 6.0.)
*OS/3	ALL	7/79	UP-8605.10	OS/3 Technical Bulletin #10 (This document contains information concerning OS/3 FILE CATALOGING.)
*OS/3	ALL	6/81	UP-8605.11	OS/3 Technical Bulletin #11 (This document contains information concerning diskette usage on System 80.)
*OS/3	5.0 and subsequent releases	1/82	UP-8605.12 Rev. 1	OS/3 Technical Bulletin #12 (This document describes restrictions and provides guidelines to be followed when using COBOL communications. This document replaces UP-8605.12 dated 8/81.)

NOTE:

Technical Bulletins are issued as they become available, and may or may not be issued in sequential order.

PAGE

1. INTRODUCTION

1.1. PURPOSE

This technical bulletin describes restrictions and provides guidelines to be followed when using COBOL communications.

1.2. LEVEL OF PRESENTATION

This document assumes that the reader understands assembler macros, declaratives, and correct use of the ICAM standard MCP interface, as well as general use of the COBOL communications verbs and data declarations.

1.3. THE COMMUNICATION USER PROGRAM (CUP)

The OS/3 operating system supports a COBOL compiler that contains a communication capability. It consists of the following COBOL source language features:

- In the Data Division, a communication section that contains input CDs and output CDs

- In the Procedure Division, the communication verbs: SEND, RECEIVE, ACCEPT... message count, ENABLE, and DISABLE

- In the Procedure Division, the verbs STRING and UNSTRING (It is reasonable to characterize the STRING and UNSTRING verbs as communications verbs since they were added to the COBOL language to facilitate message assembly and disassembly.)

To execute the communications verbs of a COBOL object program, the customer's installation must provide a suitable communications environment. The requirements are:

- Communications hardware must be installed.

- The system generation for the installation must declare the existence of local communications hardware.

- An ICAM generation must be performed.

- A COBOL Message Control System object module (CMCS CUP) must be generated and stored in a library on the system. This is done by using the CMCS object module generation procedures that are described in the current version of UP-8552 (ICAM).

Once the requirements have been met, the COBOL object program can then be processed through the linkage editor utility. One of the object modules that will be linked with the COBOL object module is the CMCS object module just mentioned. In COBOL, the CMCS object module is referred to as a run-time routine, while in ICAM, the CMCS object module is called a communication user program (CUP).

On the one hand, the CMCS CUP responds to the communication verbs of the COBOL object program; on the other hand, the CMCS CUP employs the ICAM standard MCP interface macros and declaratives to interact with ICAM on behalf of the COBOL object program.

In summary, the generation of a CMCS object module is actually the generation of an ICAM CUP. Therefore, the purpose of preparing the parameters for the CMCS generation utility is to provide a sufficient description of the COBOL and ICAM environments, so that a CMCS CUP with the appropriate characteristics can be generated.

1.4. ASSEMBLY LANGUAGE CUP AND THE CMCS CUP

A hand-written assembly language CUP that utilizes the ICAM standard MCP interface has access to all of the capabilities of that interface. However, the COBOL programmer is not given access to all of the facilities of the standard MCP interface. The COBOL programmer reference manual, UP-8613 (current version); the ICAM utilities user guide, UP-8552 (current version); and this technical bulletin describe the facilities that are available to the COBOL programmer.

2. GUIDELINES AND RESTRICTIONS

2.1. COBOL INPUT COMES FROM PROCESS FILES

Terminal messages that are destined for a COBOL program must be queued in process files, and it is sufficient to define these files with a low queue. During CMCS CUP generation, a DB#SQT macro is used to designate the ICAM name of each input process file. The DB#SQT macro also designates the symbolic queue names that the COBOL application will use to address that process file (via the input CD). In response to the COBOL application RECEIVE statement, the CMCS uses GETCP-available to retrieve a message from a process file.

2.2. DB#SNT MACRO: SOURCES AND DESTINATIONS

The DB#SNT macro is used in the generation of a CMCS CUP. One of the parameters that DB#SNT accepts is an ICAM terminal name, which is then given an equivalent COBOL symbolic name. When the COBOL application executes a SEND to the symbolic name for a terminal, the CMCS uses the DB#SNT equate to identify the actual ICAM terminal name. The CMCS then executes a PUTCP macro that is addressed to the low priority output queue of the terminal.

Similarly, when the COBOL application executes a RECEIVE, the CMCS addresses a GETCP-available to the implicitly designated process file. (The implicitly designated process file is identified via a DB#SQT macro.) If the GETCP retrieves a message that originated at a terminal, the CMCS uses the DB#SNT equate to obtain the COBOL-equiválent name of the terminal. The CMCS then places the COBOL name in the symbolic source field of the input CD and stores the message in the identifier-1 that was designated by the RECEIVE statement.

The DB#SNT macro also takes a process file name as a parameter, making it possible for the COBOL application to send messages to a process file. (This is unusual.) If this facility is used, the user must be sure that the destination process file is one of the process files named by a DB#SQT macro. In other words, in both a global network and a dedicated network, the COBOL application can send messages only to its own input process files. If the COBOL application actually sends a message to one of its own input process files, the symbolic source of the message (when subsequently retrieved via a RECEIVE statement) will be indicated (in the input CD) to be the symbolic name of the process file, as declared in the DB#SNT macro. If the COBOL application sends a message to one of its own input process files, the user should also note that such a message would never pass through an MPPS. Thus, if terminal messages were being queued to the input process file via an MPPS, and if CUP-originated messages also were being queued to the process file, the messages could have varying formats because an MPPS can alter the format of a message.

2.3. ENABLE INPUT AND DISABLE INPUT VERBS (NO TERMINAL OPTION)

DISABLE INPUT (without the TERMINAL option) maps into a macro that instructs ICAM to destroy all messages that are subsequently received from the designated terminals. ENABLE INPUT (without the TERMINAL option) maps into a macro that instructs ICAM to accept messages once again from the designated terminals. The identity of the terminals to be disabled or enabled is determined by the CMCS as follows:

1. The symbolic queue names in the input CD identify one or more process files. (During CMCS module generation, the symbolic queue name to process file mapping was declared by means of DB#SQT macros.)

2. In turn, the DB#IRT macros declared during CMCS module generation identify the terminals that feed the designated process files. (The terminals may feed the process files directly, via the INPUT= parameter of the TERM macro; or they may feed message processing procedure specifications (MPPSs), which in turn feed the designated process files.)

It is important to notice that the aforementioned ENABLE and DISABLE statements are binary operations; either all traffic is admitted from the designated terminals or no traffic is admitted from the designated terminals. For example, suppose that, through an input MPPS, the messages that are received from a terminal are queued to two different process files - e.g., the MPPS performs ''queue by message content''. Then, in the COBOL program, the statement ''DISABLE INPUT cd-name...'' should not be directed to only one of the process files while the other one gives the appearance of remaining enabled. As explained previously, disabling and enabling are binary, and the messages received from a terminal are either totally retained or totally discarded in response to the aforementioned statements. Therefore, as a practical matter, either both of the process files are unable to receive messages from the terminal or both of the process files are able to receive messages from the terminal.

Consequently, it is not recommended that a COBOL program possess a hierarchy of input queues to which the COBOL statements could be issued in such a way as to give the misleading appearance that some input queues fed by a terminal are enabled while others are disabled.

2.4. ENABLE INPUT TERMINAL AND DISABLE INPUT TERMINAL VERBS

2.4.1. The Disable Verb Causes Destruction of All Messages Received from the Terminal

DISABLE INPUT TERMINAL maps into a macro that instructs ICAM to destroy all messages that are subsequently received from the designated terminal. ENABLE INPUT TERMINAL maps into a macro that instructs ICAM to once again accept messages from the designated terminal. (The CMCS determines the identity of the affected terminal by means of a DB#SNT macro that was declared during CMCS data base generation; the DB#SNT macro relates the COBOL symbolic name of the terminal to the ICAM symbolic name of the terminal.)

Since disabling and enabling are binary operations, it follows that when DISABLE INPUT TERMINAL is in force, all input subsequently received from that terminal will be destroyed by ICAM. Thus, no input flows to the one or several input process files serviced by that terminal.

2.4.2. Change in Status Reported by CMCS in Releases Subsequent to Release 7.1

If CMCS is unable to recognize the terminal name that the COBOL program supplied with an ENABLE INPUT TERMINAL... or DISABLE INPUT TERMINAL... statement, then the CMCS returns error status to the COBOL program. In OS/3 Release 6 (all) and OS/3 Release 7 (all), the CMCS returns status key value 21 for this condition. However, subsequent to Release 7.1, the returned status key value is 20. This change was made to bring CMCS into conformance with the requirements of the 1974 ANSI COBOL Standard. (The change will be reflected through republication of Table 5-11 of the current version of the OS/3 1974 American National Standard COBOL Programmer Reference, UP-8613.)

Optional Correction Number C061946 is available for OS/3 Release 6 (all) if the customer prefers to alter CMCS so that it will return status value 20 instead of 21. Similarly, optional Correction Number C074640 is available for OS/3 Release 7 (all). However, before an OS/3 Release 6 or OS/3 Release 7 customer applies the pertinent correction, it is important to review the logic of all COBOL source programs with which the revised CMCS will run. If the source programs recognize and act on status 21, then they must be modified to instead recognize and act on status 20.

Of course, if an OS/3 Release 6 or OS/3 Release 7 customer wants presently compiled COBOL programs to be affected, it is not enough merely to apply the assembler source correction to the source code of CMCS. Before the CMCS source correction can have an effect, an instance of CMCS must be generated and linked with the COBOL object program that is to be affected.

Customers who write new COBOL communications application programs that are to be compiled under the release following 7.1 are asked to code their programs to expect error status 20 (not 21) for the described condition. In addition, if a COBOL communication application program that has been running with OS/3 Release 6 (all) or OS/3 Release 7 (all) is to be recompiled and relinked under the release following 7.1, its program logic should be reviewed to determine whether it is affected by the change in returned error status.

2.5. ENABLE OUTPUT AND DISABLE OUTPUT VERBS

DISABLE OUTPUT maps into the QHOLD ICAM macro. DISABLE OUTPUT designates the COBOL names of terminals and thus prevents messages in the named output terminal queues from flowing to the terminals. Similarly, ENABLE OUTPUT maps into the QRELSE ICAM macro. ENABLE OUTPUT designates the COBOL names of terminals and thus enables traffic to flow from the named output terminal queues to the terminals. The CMCS determines the ICAM identity of the affected terminals by means of DB#SNT macros that were declared during CMCS module generation.

ENABLE OUTPUT and DISABLE OUTPUT should never be directed to the COBOL symbolic for a process file. QHOLD and QRELSE are not logical operations to perform on a process file.

2.6. DB#SNT MACRO: INIT= PARAMETER

As explained in 2.2, the DB#SNT macro normally accepts a terminal name as a parameter; however, in rare applications, it may also accept a process file name. In addition, the DB#SNT macro accepts the INIT= parameter. INIT= should be used only in a DB#SNT macro that names a terminal (thus, not in a DB#SNT macro that names a process file).

The intent of the INIT= parameter is carried out automatically by the CMCS, as soon as the COBOL application is loaded and before the first statement in the Procedure Division is executed. The COBOL language equivalents of INIT=YES are:

```
ENABLE INPUT TERMINAL...
          and
ENABLE OUTPUT...
```

In other words, INIT=YES instructs the CMCS to immediately begin accepting messages from the designated terminal and to ensure that the terminal output queue is enabled, thus permitting COBOL application messages to flow from the terminal output queue to the terminal. (ENABLE OUTPUT is implemented via the ICAM macro QRELSE.)

The COBOL language equivalents of INIT=NO are:

```
DISABLE INPUT TERMINAL...
          and
DISABLE OUTPUT...
```

In other words, INIT=NO instructs the CMCS to destroy all messages that are received from the designated terminal and to lock the terminal output queue so that COBOL application messages placed in the queue cannot flow to the terminal. (DISABLE OUTPUT is implemented via the ICAM macro QHOLD.)

If INIT=NO is specified, the CMCS will execute this directive before the first statement in the Procedure Division of the COBOL application is executed. The disabling will occur in time to prevent transmission of any messages from the COBOL program to the terminal. However, the disabling activity for input may be late, causing some input messages to be received from the terminal before the disabling actually takes effect. In the case of a dedicated network, the ``window'' is small, and it is reasonable to assume that few or no messages will arrive from the terminal before the directive is put into effect. In the case of a global network, the window can be much larger since the global network may be loaded and the lines activated long before the COBOL application is loaded and attached to the network.

The INIT=NO parameter should not be specified with a DB#SNT macro for a terminal when the parameter FOR INITIAL INPUT is specified with an input CD in the COBOL source program. It is illogical to ask that input messages be discarded (INIT=NO) and at the same time ask (via FOR INITIAL INPUT) that the COBOL program be given the name of the input queue in which the first-arrived message has been stored.

2.7. VIDEO DISPLAY LINE CLEARING

The screen of a video display consists of lines; a message sent to the display can occupy one or more lines. When the SEND verb sends a message to the display, the message may not completely fill the last line that the message occupies. In this case, an erase-to-end-of-line DICE that the CMCS has attached to the end of the message text clears the remainder of the line. However, if the message precisely fills the last line, then the effect of the trailing DICE is to clear the next line of the display. If this latter effect is not desired, then a message that is sent to the display should leave the last character of the last line unfilled.

NOTE:

The erase-to-end-of-line DICE does not pass through a protected field on the screen. Therefore, when using COBOL communications, use of the protected field feature is not recommended.

2.8. RECEIVE...SEGMENT STATEMENT

The RECEIVE verb with the SEGMENT modifier instructs ICAM to return one line of the message that was received from the terminal. For example, if a terminal message occupies three lines on a video screen, then in response to three RECEIVE...SEGMENT statements, the message is returned to the COBOL program in three pieces.

When using the RECEIVE...SEGMENT statement, the programmer must ensure that the size of the identifier-1 receiving data item is at least one byte larger than the line length of the terminal.

2.9. NO INPUT DICE

The CMCS manuals state that, within the network definition, the TERM macros for terminals to be used by CMCS must declare the parameter DICE=(ON). The DICE=(ON) instructs the terminal handlers to convert terminal control character sequences received from the terminals into DICE expressions (where such equivalencies exist). DICE=(ON) thus enables the CMCS to instruct ICAM to find and delete these DICE expressions when ICAM passes the terminal message text to CMCS. The effect is that the terminal message text passed from the CMCS to the COBOL application does not contain DICE expressions. However, terminal control characters in the message text that have no DICE equivalent are passed through to the COBOL program.

2.10. LINE ACTIVATION

No verbs in the COBOL language control communication line activation or deactivation or reactivation. When the COBOL application is loaded in a dedicated network environment, the CMCS requests (via an option of the NETREQ macro) that line activation be performed during network attachment. In a global network environment, line activation is the responsibility of the global user service task (GUST). The only control that the COBOL application can have over line activation/deactivation is through messages directed to the console operator.

2.11. SEND VERB RESTRICTION

An output CD may be used with a SEND statement to transmit message text to one or more COBOL symbolic destinations. However, if multiple SEND statements are used to transmit a single message (e.g., SEND...WITH ESI, SEND...WITH ESI, SEND...WITH EMI), then the output CD must remain dedicated solely to that particular purpose until the final portion of the COBOL message has been transmitted via SEND...WITH EMI. In addition, it is unacceptable for the set of symbolic destinations in the output CD to be altered during the course of those several SEND statements. All portions of the message must be sent to the same set of symbolic destinations.

2.12. DEADLOCK WARNING

When the RECEIVE statement is used without the NO DATA phrase, the COBOL program may go into deadlock waiting for input that never arrives.

2.13. TERM MACRO: OUTPUT QUEUE SUPPORT

UP-8552 (current version) states that the QUEUES= parameter of the TERM macro must be specified for a terminal that is used with the CMCS.

However, the TERM macro parameters LOW=, MEDIUM=, and HIGH= replace the QUEUES= parameter for defining terminal output queue support.

The TERM macro for a terminal that uses CMCS should (at a minimum) contain the LOW= parameter. LOW= generates a low priority output terminal queue for the terminal. This is required because, in response to a SEND statement, CMCS directs the message text to the low priority output queue of the terminal.

2.14. TERMINAL RESTRICTIONS

The COBOL program can exchange messages with the ICAM terminals that are characterized as interactive. The COBOL program cannot exchange messages with the ICAM terminals that are characterized as batch.

2.15. AUXILIARY DEVICE RESTRICTIONS

The COBOL program cannot originate messages that are addressed to the auxiliary devices of interactive terminals or solicit messages from the auxiliary devices of interactive terminals. In other words, the CMCS CUP can only direct the PUTCP macro to a name that has been declared in the symbolic field of a TERM macroinstruction in the CCA. Therefore, the CMCS CUP cannot address the auxiliary devices of an interactive terminal.

2.16. FUNCTION KEYS AND COBOL APPLICATION

The COBOL application is not notified if the terminal operator presses a function key or MESSAGE WAITING.

2.17. NO MULTINODE (VLINE) SUPPORT

COBOL communications has not been qualified for use in a multinode environment, i.e., where the COBOL program is in one node and the terminals are in another node.

2.18. STATIC SESSION SUPPORT (ONLY)

A COBOL program in a global network must exchange traffic with its terminals only over static sessions. Such sessions are designated by means of the network generation SESSION macro (2.19). Therefore, a COBOL program in a global network cannot exchange traffic with its terminals via a dynamic session established by using the SESCON macro.

2.19. NETWORK DEFINITION SESSION MACRO

2.19.1. General Use of the SESSION Macro

The network definition SESSION macro should be used to claim the ICAM resources that the COBOL application LOCAP is using. For example, a session should be declared that relates the LOCAP to each process file from which input messages will be retrieved (by means of the RECEIVE statement). Similarly, sessions should be declared between the terminals and the process files into which the CUP-destined terminal messages are being directly queued. However, it is unnecessary to declare sessions in which one member of the session would be an MPPS.

2.19.2. SESSION Macro and FOR INITIAL INPUT Phrase

In the COBOL source program, when the phrase FOR INITIAL INPUT is associated with an input CD, the programmer informs the compiler that the COBOL program may be loaded and executed automatically when the first message for that application arrives in ICAM. (This capability is available only in global networks.) In order to express this request to ICAM, the programmer must also use special parameters of the network definition SESSION macro, as well as the LOCAP macro.

In the case of the SESSION macro, the programmer must specify the PRIMARY attribute for each session that he declares between the LOCAP and a process file that supplies input messages to the LOCAP. The PRIMARY attribute instructs ICAM to invoke the COBOL program job stream when the first message enters one of those process files.

The LOCAP parameters that must be specified when FOR INITIAL INPUT is designated in the COBOL source program are described in 2.20.

2.20. LOCAP PARAMETERS REQUIRED WHEN USING THE ''FOR INITIAL INPUT'' PHRASE

When the FOR INITIAL INPUT phrase is specified with an input CD in the COBOL source program, special parameters are declared in the network definition LOCAP macro. (Also, a SESSION macro special parameter is required, as discussed in 2.19.) These special LOCAP parameters are:

```
    JOBNAME=jobname
```

The jobname parameter is used to designate the name of the canned job stream that contains the COBOL program. This is the canned job stream that ICAM invokes when the first terminal message enters a program input process file.

```
    JOBINIT=(LOAD,REPORT)
```

This parameter must also be specified and be exactly as shown. LOAD causes the job stream to be loaded and executed. REPORT causes ICAM to notify the CMCS CUP (via a datagram) upon receipt of the first message. (CMCS then posts this information to the input CD containing the FOR INITIAL INPUT phrase.)

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:

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

Cut along line.

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

CUT