

Message Switching/Printout Spooling

This chapter covers the following topics:

- Overview
 - Message Switching
 - Printout Spooling
-

Overview

The Com-plete message switching and printout spooling functions enable online and batch application programs to send messages or printout data sets to one or more terminals. The receiving terminal(s) can be either a hard copy terminal device or a soft copy (display) terminal device.

Messages and printout data sets are both processed by the Com-plete message switching task, although separate functions must be specified for each type of request.

The distinction between a message and a printout is normally the number of output lines and/or the type of terminal device(s) to receive the output.

- *Messages*
Messages are considered to be text that is not column- or line-oriented and that is comprised of words, sentences, and paragraphs. Messages are also assumed to contain a relatively small amount of data. Because of this, Com-plete will sometimes write a message to a terminal in a slightly different format than it was sent. For instance, Com-plete attempts to start a line between words so that words are not broken. 3270-specific WCCs and control attributes are not valid as data.
- *Printouts*
Printouts are assumed to contain formatting and spacing that cannot be altered; they are also assumed to contain a large amount of data. Because of this, lines are always written as is. If the receiving terminal's line length is too short, the output line will be truncated.

When a message switching or printout spool function is initially invoked, Com-plete creates a buffer core queue in order to maintain tracking control over the message or printout. When the message or printout is successfully received, the core queue is freed, and the space occupied becomes available for use. Although core queues are relatively small in size, a large number of messages or printouts requires a large number of core queue elements.

The core queue elements are used by Com-plete to enable message or printout restart/recover in case of an abnormal system failure.

The message switching and printout spooling functions of Com-plete are available to both batch and online programs. The message or printout awaiting receipt may be monitored by use of the Com-plete online utility program USPOOL, which is described in the Com-plete Utilities documentation.

Message Switching

The message switching facilities of Com-plete enable an application program to send messages to one or more terminals in the Com-plete network. The application program can be an online or batch program.

Messages can also be sent from a given terminal to one or more terminals without using an application program. This facility is provided by the Com-plete online utility program UM, which is fully documented in the Com-plete Utilities documentation. Message switching performed by an application program is accomplished by use of the MESGSW function.

When a message is sent to a terminal, it is by default copied onto a disk file containing queues of messages for each terminal. Messages can be recalled from this file by use of the UM utility program at any time, as long as they have not been deleted.

Message Segmentation

Messages are formed in pieces called segments. Each segment cannot exceed 32,767 bytes in length. There is no limit to the number of message segments that comprise a given message, although the size of the message queue file itself is a limiting factor. Before a segmented message is written to the receiving terminal device, the segments are linked together by Com-plete, and the message is sent as a single unit.

All messages, regardless of content, are treated as text when they are displayed at the receiving terminal. They are printed or displayed using the maximum line length of the receiving device. A word that will not fit at the end of a line is moved to the beginning of the next line. All blanks at the end of a message are ignored.

Limited message output formatting can be accomplished by embedding the Com-plete terminal new line symbol in the message text. The new line symbol is X'15'. This symbol is device-independent and produces the same results regardless of the device type to which the message is sent.

Destination Codes

When a message is sent, information stating where the message is to be received must be included. The terminal(s) to receive the message can be specified by referencing one or more destination codes. Each destination code is a TID number, a destination code that represents one or more TIDs, or a user ID. Destination codes are defined by the installation to contain convenient groupings of TIDs. Procedures for changing and adding destination codes are also defined by the installation. Destination codes used in a message switching request are converted into TID numbers, and the message(s) is sent to the appropriate terminals. If the destination code is a user ID, the notation "U=user-id" can be used to distinguish from a group name.

The maximum number of terminals that can receive a message via the MESGSW function is 100, even if a destination code is used. This maximum number restriction can be circumvented by sending multiple copies of the same message, specifying 100 or fewer receiving terminals each time.

Class Codes

Security restrictions and other information about a message are designated by class codes. There are two categories of class codes: security class codes (those numbered 1 through 4), and other class codes (those numbered 8 through 16). Each message must have at least one security class code assigned, or the message request will be unsuccessful.

Each terminal in the Com-plete network has two sets of security class codes assigned to it: one that defines which message classes it can send, and one that defines which message classes it can receive. If the accounting option is used, class codes are assigned through your User ID. If the accounting option is not used or if you have not logged on to Com-plete, class codes are assigned through the terminal definition table.

Messages must have security class codes assigned. Com-plete checks to be sure that the sending terminal is authorized to send a specific message, since the class code(s) assigned to a message must be among the sending class codes assigned to the sending terminal. Com-plete also determines that the assigned class code(s) is(are) included in the receiving class codes of the terminal(s) to which the message is sent. If the class codes are not compatible, the message is not sent, and in some cases, a security violation is logged to the Com-plete logging device.

Several class codes (those numbered 8 through 16) cause certain special operations to be performed as a message is sent. These operations and all the class codes are explained in the following figure.

Class codes 1 through 4 are used for security checking. Note that every message or printout must have at least one of these classes.

Class 1	Standard message class. Messages with this class assigned do not interrupt a terminal while it is in conversation with a program.
Class 2	Urgent message class. Overrides the MESSAGE DISABLED status of a terminal and causes the message to be displayed immediately at the receiving terminal. The receiving terminal is interrupted if it is in conversation with a program. If the receiving terminal has the audible alarm feature, the audible alarm will sound.
Class 3	Special purpose class code. The message interrupts a terminal in conversation. These messages are always restarted from the beginning if Com-plete is reinitialized before they are sent successfully.
Class 4	Reserved for Com-plete logged messages. No application program or terminal can initiate a Class 4 message. These messages are exempt from restart in the event that Com-plete is reinitialized before they have been received successfully.

The following class codes are referred to as other class codes. They cause special services to be performed as a message is sent.

Class 8	Reserved for future usage.
Class 9	Reserved for future usage.
Class 10	Reserved for future usage.
Class 11	Reserved for future usage.
Class 12	Causes a message to be deleted from the message queue file after 30 minutes, if the message cannot be sent to the receiving terminal. These messages are exempt from restart in the event that Com-plete is reinitialized before they have been received successfully.
Class 13	Causes a message to be written without the standard message header that normally accompanies all messages.
Class 14	Standard message class; no special action taken. These messages are exempt from restart in the event that Com-plete is reinitialized before they have been received successfully.
Class 15	Causes the audible alarm to be sounded when the message is written to the terminal, if the receiving terminal has the audible alarm feature.
Class 16	Prevents a message from being queued to the message queue file on disk. Instead, the message remains in main storage until it is successfully sent. The length of a message using this class code is limited to the amount of text that can be contained in one message buffer. This amount can be calculated by doubling the number of terminals to receive the specified message and subtracting this amount from 240. For example, if there were three receiving terminals, a Class 16 message could be a maximum of 234 characters long; however, the message would not be requeued upon restart.

Message Routing

With the exception of Class 13 messages, each message received has a standard header containing the message identification number, date, time the message was written, and TID number of the terminal from which the message was sent. The following display illustrates a typical header message.

```
MSG ID:      133, SENT 10/17/86 AT 1106, FROM TID 17
SAMPLE MESSAGE HEADER DISPLAY
```

If this header is not desired or if the five new line symbols appended to a message sent to a hard copy device are not wanted, the message to be sent must be assigned Class code 13.

Messages sent to hard copy devices are written continuously. Messages sent to CRT devices require acknowledgement after being displayed. Acknowledgement is accomplished by pressing the ENTER key. If the message is not acknowledged immediately, the next null input entry will be considered an acknowledgment. In addition, if an application program is in use at the receiving terminal, any screen formatting in use may be destroyed. The specific procedures for recovering a screen format are application-dependent, and the specific application in use at the time of the interrupt must be referenced for recovery procedures. Note that, regardless of the acknowledgment procedures, the application in use is not terminated, only interrupted.

If a message sent is longer than the buffer size for the receiving terminal and the display is not fully displayed, an asterisk (*) is the last character displayed. The remaining portion of the message can be displayed by pressing the ENETR key.

Messages sent to terminals cannot be received immediately for the reasons indicated below. When this occurs, all messages not received are queued to the destination terminals. Since Com-plete uses buffers to process terminal communication messages, a potential system problem exists. A large queue of messages that cannot be sent could potentially saturate the buffer pool, disabling all terminal communications and causing a system lockout condition. This situation can be avoided if application programs that send messages use the GETCHR function of Com-plete prior to using the MESGSW function to interrogate the message receipt status of the destination terminal(s). If the number of messages or printouts queued to a destination terminal exceeds a specified value, the application program may issue a warning message to the appropriate user, refuse to send additional messages until the number of queued messages is acceptable, or take some other corrective action.

Three common situations in which messages sent to a terminal are *NOT* automatically displayed are:

- The terminal is MESSAGE DISABLED. Occasionally when messages are sent to your terminal, they may interrupt and destroy whatever data is being entered at the time. To prevent this, the you can set the terminal to disabled status for receipt of messages; however, disabled status does not prevent Class 2 messages from being received.
- A conversational program is executing at the terminal. If an active program is currently in use at the receiving terminal, messages are placed in a special message queue for that terminal. The messages are received when the application program terminates. This feature does not apply to Class 2 or 3 messages.
- The previous message was not acknowledged. If a previously sent message has not been acknowledged, additional messages are queued. This condition can occur only with CRT devices.

When a terminal is enabled for receipt of messages (for example, ending a conversational program, removing the MESSAGE DISABLED status), any messages that have been placed in the message queue are available for display. For CRT terminals, acknowledgment of each message queued to the terminal causes the next message to be displayed until all have been displayed and acknowledged. Messages queued to a hard copy terminal are printed automatically when the terminal is made available to receive them.

When a terminal is MESSAGE DISABLED or in conversation, all UM utility commands can be used to display specific messages or message information without altering the disabled or conversational status on termination of any active program.

Alternate Terminals

Alternate terminals are terminals designated to receive message output for terminals that are inoperative, currently in conversation with an online program, or disabled for message receipt. Alternate terminals are assigned either through the terminal definition table (TIBTAB) or via the ALTERNATE command function of UM.

If a terminal has an alternate terminal assigned and cannot immediately receive a message, any message sent to it will automatically be displayed or written to the alternate terminal. Messages rerouted to an alternate terminal are *not* queued to the original receiving terminal.

Disabled Terminals

Disabled terminals are terminals that have been disabled for receipt of messages by use of the DISABLE command function of UM.

When messages are sent to a terminal, they may interrupt and destroy whatever data may have been entered at the time the message was sent. To prevent this, the DISABLE command function of UM is used. When messages (except those sent with Class 2) are sent to a disabled terminal, they are routed to an alternate terminal, if one has been assigned. If an alternate terminal has not been assigned, the messages are placed in the message queue for the disabled terminal. Class 2 messages override a DISABLED status and are displayed immediately.

Messages sent to a disabled terminal, if not rerouted to an alternate terminal, are automatically displayed when the terminal is once again enabled to receive messages.

Inoperative Terminals

Inoperative terminals are terminals that are either turned off, malfunctioning, or write-inhibited via a switch.

Com-plete handles inoperative terminals in a manner similar to the way it handles DISABLED terminals: messages sent to these terminals are rerouted to a designated alternate terminal, if one has been assigned. If an alternate terminal has not been assigned, the messages are placed in the message queue for the inoperative terminal.

Messages sent to an inoperative terminal, if not rerouted to an alternate terminal, are automatically displayed when the terminal is once again able to receive messages.

Message Recovery

Messages sent to hard copy terminals are immediately written, if the terminal is in ready status and no other messages are queued to the terminal.

The Com-plete message switching facility provides automatic message restart in the event that terminal output is interrupted. Message restart is performed on a checkpoint basis. Each message sent to a terminal is initially queued to the message switching file queue that resides on disk. Messages residing on the message file queue are sent, or written, to the destination terminal on an availability basis. As messages are written, a core queue checkpoint is taken to indicate the status of the output being written. For large messages, this checkpoint is taken at the completion of each full page of output. If output is interrupted (for example, by pressing the STOP key), output resumes at the last checkpoint when the terminal is again made ready (for example, by pressing the START key).

If a system failure occurs, all message core queue checkpoint records are destroyed. When Com-plete is again initialized, various message restart options are available, the default being recovery of message output from the beginning of the output message. The system programmer responsible for Com-plete maintenance should be consulted for more details on the message recovery options available after a system failure.

Message Switching Control Block (MESGCB)

Before an application program can send a message, it must create and initialize a Message Switching Control Block (MESGCB). The MESGCB is a working storage area in the program containing information needed by Com-plete to control processing of the message.

Three types of information exist in the MESGCB:

- Class codes;
- Segment flags;
- Message identification numbers.

You can set class codes before sending the message.

The segment flag is also set by the application program before sending the message. If the message is to be treated as an entire message, the segment flag must be initialized to an L. If the message is to be treated as a segment of the entire message, the segment flag must be initialized to a space (X'40'). For segmented messages, the segment flag must be initialized to an L when the last segment has been created and is ready to send.

The message identification number must always be initialized to spaces. When the message is sent, Com-plete uses this field and initializes it with the message number assigned to the message. If the message is being sent in segments, the message number is set by Com-plete in this field after using the MESGSW function to send the first segment.

The format and content of the MESGCB are illustrated in Message Switching Control Block (MESGCB).

A DSECT of the MESGCB can be created using

```
name MCALL MESGCB
```

MESGSW Function

The MESGSW function is used by the application program to send a message or a message segment. The message switching control block must have been defined and initialized prior to execution of the MESGSW function.

Format

The format for using the MESGSW function is:

```
MESGSW (retcode,mesgcb,area,length[,destination][,number])
```

retcode	Required. A fullword where Complete places the return code upon completion of the operation.
mesgcb	Required. The message switching control block to be used when sending this message. This control block must have been previously defined and initialized in the application program.
area	Required. A buffer area in the application program containing the message or message segment to be sent.
length	Required. A binary halfword containing the length of the message or message segment to be sent.
destination	Optional. (Required for the first segment of a message.) Default: None. This keyword is ignored for message segments that are not the first segment. A table defined in the working storage area of the application program. This table contains destination codes and/or TIDs eligible for receipt of messages. Each code in the table must be eight bytes long, left-justified, alphanumeric in format, and padded to the right with blanks. In some cases, you may need to use 10-byte long table entries. For example, to send to 7 or 8-byte user IDs using the U=userid notation. Add 32768 to the actual number of destinations (see the "number" parameter for details) to advise Message Switching to interpret the destination table as having 10-byte entries. This is equivalent to setting the high-order bit of the number halfword to one in Assembler language programs.
number	Optional. (Required for the first segment of a message.) Default: None. A binary halfword containing the number of destination codes (defined in the previous table) that are to be used.

Return Codes

The following return codes are issued by the MESGSW function:

0	Normal return.
4	A message segment was requested. The segment requested was not the first segment, and the first segment cannot be found. Probable MESSAGES control block error.
8	A security violation has occurred, caused by invalid class codes.
12	An unrecoverable I/O error has occurred.
16	Too many receiving terminals were specified.
20	An invalid destination code was specified.
24	A negative segment length was specified.
28	The message text was too long. This return code is provided for Class 16 messages only.

Abends

An abnormal termination may occur during execution of the MESSAGES function. Possible causes include:

- An invalid control block was specified;
- An invalid *area* or *length* argument was specified;
- An invalid *number* argument was specified;
- A Class code of 4 was specified. Application programs cannot use Class code 4.

Printout Spooling

The Complete printout spooling functions enable an online or batch application program to generate print output destined to be printed on a hard copy terminal. More than one program can generate printouts destined for the same device, and individual printout data sets are spooled separately. The outputs are not intermixed.

Print output is created by the application program one statement, or line, at a time; however, the print output itself occupies a logical printout spool data set and is not available for printing on a terminal(s) until the data set has been queued to the terminal(s). The functions available for printout spooling requests enable the application program to allocate the printout spool data set, write the print output statements to the data set, and queue the data set for scheduled printing. The printout spool functions available for use by an application program are summarized in the following table.

Function	Description
PSOPEN	Allocate a printout spool data set.
PSPUT	Write statements to the printout spool data set.
PSCLOS	Queue the printout spool data to the destination terminal(s).

In addition to using the Com-plete printout spool functions to generate print output, the application program must also define a Printout Spool Control Block (PSCB) in the working storage area of the application program.

The information in the PSCB is used to identify the printout and to control the output form and the logical output driver.

Specifying a logical output driver enables you to modify the listing during output. This facility can be used to create extra output lines (for example, header, trailer) for each printout. In the event of changes, you need only to change the logic of the logical output driver instead of changing all their application programs. See the supplied modules LDRVSAMP and IPDSDRV in the Com-plete source for examples.

Since the printout spool facility of Com-plete also utilizes the message switching facility of Com-plete, printout data sets must be assigned class codes in order to determine authorization for both the receipt of printout spool data sets and the ability to generate printout spool data sets. The PSCB is referenced by each of the printout spool functions and is used when defining the required authorization class codes.

The terminal(s) destined to receive a given printout spool data set is also defined in the working storage section of the application program in a table of destination codes or TIDs. Since print output is spooled on a printout spool data set basis, this table is referenced only by the PSOPEN function when defining the printout spool data set.

Destination Codes

The terminals to which spooled printouts are destined are specified using TIDs defined in a table or by referencing one or more destination codes also defined in a table. Both TIDs and destination codes can be used in the same table. Each entry is defined as an alphanumeric entry, left justified, and padded with blanks.

The destination ID "SYSOUT" is a special-purpose destination used to route a printout to the operating system spool. The default output class in the operating system is "A". Note that the output class can be modified by specifying SYSOUT=x as destination, where *x* represents the output class.

Each destination code represents one or more TIDs and is defined by the installation for convenient groupings of terminals. Procedures for changing and adding destination codes are also set by the installation.

Destination codes defined in an application program are converted to TIDs, and spooled printout data sets are sent to the appropriate terminals. Multiple copies of a printout spool data set can be sent by specifying the same destination entry more than once; however, the maximum number of receiving terminals per printout spool function cannot exceed 100 terminals.

Class Codes

Security restrictions for printout spool data sets are designated by the message switching class codes. Each terminal in the Com-plete network has two sets of security class codes assigned to it: one defines which class codes are valid for sending, and one defines which class codes are valid for receipt. These class code assignments are made on a terminal basis and, in the case of conversational type terminals, can be overridden by the user ID of the terminal user after a *ULOG ON request is issued.

A printout spool data set must have a class code associated with it. Complete checks to be sure that the terminal from which the application program is called is authorized to send the printout spool data set. The class code(s) assigned to the printout spool data set must be among the sending class codes of the terminal in use.

Complete also determines that the class code(s) assigned to the printout spool data set is included in the list of receiving class codes of the terminal(s) to which the printout data set is sent. If the class codes are not compatible, the printout data set is not sent.

The class codes available for use with the printout spool facility of Complete are Class codes 1 through 7. These class codes are fully described in the previous section *Message Switching*. Class codes 8 through 16 are reserved exclusively for the message switching facility of Complete and should not be used with printout spooling requests.

Disabled, Inoperative, and Alternate Terminals

A terminal destined to receive a printout spool data set can be disabled, inoperative, or have an alternate assigned. The definition of these conditions and the resulting routing conditions that occur are fully described in the previous section *Message Routing*.

Printout Spool Control Block (PSCB)

The Printout Spool Control Block (PSCB) is defined in the working storage section of the application program. One PSCB must be defined for each printout spool data set to be used by the application program.

The PSCB for a given printout spool data set is referred to by each printout spool function requested for the data set and used to contain information needed by Complete to control the routing of the printout data set. The following six types of information are contained in the PSCB:

- Listname;
The listname must be established by the application program before use of the PSOPEN function.
- Form name.
- Logical output driver
- Class codes;
The class codes settings are established by the application program before use of the PSOPEN function.
- Logical statement lengths;
The logical statement length identifies the length of each print output statement, and must remain constant for the entire printout spool data set. Note that the maximum logical statement length is 220 bytes.
- Printout identification numbers;
The printout identification number is a sequential number assigned by Complete to the printout spool data set. Note that this number must not be modified by the application program or an abnormal termination will occur in the application program.

The format of the PSCB is illustrated in Printout Spool Control Block (PSCB).

A DSECT of the PSCB can be created using

```
name MCALL PSCB,LRECL=nn
```

where nn is the length of the records to be written to the spoolfile.

PSOPEN Function

The PSOPEN function provides the information needed by Com-plete to create a printout spool data set. The application program must initialize a PSCB for each printout spool data set for which the PSOPEN function will be used.

Format

The format for using the PSOPEN function is:

```
PSOPEN (retcode,pscb,codes,numcodes)
```

retcode	Required. A fullword where Com-plete places the return code upon completion of the operation.
pscb	Required. Specifies the printout spool control block defined for the printout spool data set. This control block must have been previously initialized by the application program.
codes	Required. Each entry in the table must be an eight-byte alphanumeric field, left-justified, padded to the right with blanks.Specifies a table of destination codes and/or TIDs to be used to identify which terminal(s) are to receive the printout spool data set when a PSCLOS function is issued.
numcodes	Optional. Default: 1Specifies a binary halfword containing the number of entries to be used from the codes table.

The maximum number of receiving terminals that can be specified by the PSOPEN function is 100. If more than 100 terminals are to receive the printout spool data set, then the PSOPEN function must be reused after the PSCLOS function is used. Alternatively, more than one PSOPEN function can be used simultaneously using an additional PSCB for additional PSOPEN functions.

Return Codes

The following return codes are issued by the PSOPEN function:

0	Normal completion.
4	A PSCB control block error has occurred.
8	A security violation has occurred. The terminal in use is not authorized to send the printout spool data set being requested for creation.
12	The class codes specified were invalid.
16	Too many receiving terminals were specified.
20	One or more destination codes were invalid.
24	An invalid record length was specified.

Abends

An abnormal termination may occur during execution of the PSOPEN function. Possible causes include:

- An invalid PSCB control block was specified;
- An invalid number of destinations was specified.

PSPUT Function

The PSPUT function is used to output, or write, a record from a working storage buffer area in the application program to the printout spool data set named by the PSCB argument. Note that the record is spooled to the data set. The record is not immediately sent to the destination terminal(s).

Format

The format for using the PSPUT function is:

PSPUT (retcode,pscb,area)

retcode	Required. A fullword where Com-plete places the return code upon completion of the operation.								
pscb	Required. Specifies the printout spool control block assigned to the printout spool data set. This control block must have been previously defined by the application program and must have been opened with the PSOPEN function.								
area	Specifies a buffer area in the working storage area of the application program containing the record to be written to the data set. The first character of this field must be one of four carriage control characters: <table border="0" style="margin-left: 20px;"> <tr> <td>blank</td> <td>Advance one line before printing.</td> </tr> <tr> <td>0</td> <td>Advance two lines before printing.</td> </tr> <tr> <td>-</td> <td>Advance three lines before printing.</td> </tr> <tr> <td>1</td> <td>Advance to a new page before printing.</td> </tr> </table>	blank	Advance one line before printing.	0	Advance two lines before printing.	-	Advance three lines before printing.	1	Advance to a new page before printing.
blank	Advance one line before printing.								
0	Advance two lines before printing.								
-	Advance three lines before printing.								
1	Advance to a new page before printing.								

Return Codes

The following return codes are issued by the PSPUT function:

0	Normal return.
4	A PSCB control block error occurred.
12	An unrecoverable I/O error occurred.
24	An invalid logical record length was specified.

Abends

An abnormal termination may occur during execution of the PSPUT function. Possible causes include:

- An invalid PSCB control block was specified;
- The PSCB control block was not opened with the PSOPEN function;
- An invalid *area* argument was specified.

PSCLOS Function

The PSCLOS function is used to logically close the printout spool data set identified by the PSCB argument. Use of the PSCLOS function causes the identified printout spool data set to be queued for printing to the destination terminals.

Format

The format for using the PSCLOS function is:

`PSCLOS (retcode,pscb)`

retcode	Required. A fullword where Com-plete places the return code upon completion of the operation.
pscb	Required. Specifies the printout spool control block that identifies the printout spool data set to be queued for output. This control block must have been previously defined and opened by the application program.

Return Codes

The following return codes are issued by the PSCLOS function:

0	Normal completion.
4	A PSCB control block error occurred.
12	An unrecoverable I/O error occurred.

Abends

An abnormal termination may occur during execution of the PSCLOS function. A possible cause is that the PSCB was not opened.