ROPES

Remote Online Print Executive System
Version 16.0

Programmer’s Guide

PREFACE

This publication contains information necessary for the installation, maintenance and operation of the Remote Online Print Executive System (ROPES), a proprietary program product used to manage a network of printers in the CICS interactive environment. It provides data processing managers, system programmers, application programmers and operating personnel with the information required to effectively use this product.

Information in this publication is subject to significant change.

Tenth Edition (June 2010)

This edition applies to Version 16.0 of the program product Remote Online Print Executive System (ROPES) and to all subsequent versions and modifications until otherwise indicated in new editions or newsletters.

Contents

Introduction

What This Manual Contains

This manual is intended for programmers that will be interfacing application programs to ROPES through the ROPES Application Programming Interface. The ROPES sample programs are documented. The major applications and application related features are also documented here.

Each chapter in the manual addresses one or more of these subjects.

Other ROPES related activities are documented in the other ROPES manuals. The Related Publications lists those manuals and summarizes their contents.

Just a brief note on a notation used in this manual. Wherever you see <hlq> in a data set name we are referring to the **h**igh **l**evel **q**ualifier you choose to use when installing the ROPES data sets. You should always substitute that value when you actually refer to the data sets in JCL, and you will need to make this information available to any programmers or users that might need this information in the course of their work.

Related Publications

In addition to this manual, there are three other ROPES publications. Your installation was supplied with at least one copy of each of these manuals in Adobe Acrobat (PDF) format. You may print any copies you require for your company's use, but remember that this material is licensed to your company and may not be distributed outside of your organization.

General Information Manual

This manual contains an introduction to ROPES with an overview of the functionality ROPES provides.

Administrator's Guide

This manual is intended to guide the ROPES installer and administrator through the steps required to install ROPES, customize ROPES for your installation, establish and maintain the ROPES control files, archive report data, audit ROPES activities, account for ROPES usage, secure ROPES, and tailor and use the ROPES exits.

User's Guide

This manual is intended for end-users of ROPES. It explains the various ROPES status displays, the ROPES commands and transactions used to control ROPES, the JES transfer facility used to retrieve output from JES for printing by ROPES, and it documents the report browsing transaction.

Utilities Guide

This manual is intended for programmers that will be using the ROPES utility programs.

Messages and Codes

This manual is intended for administrators, programmers and end users that will be using the various functions of ROPES. It contains a complete list of the ROPES messages and codes with descriptions and suggested user actions when appropriate.

Report Distribution

Introduction

Report distribution is the process of breaking down large reports into smaller sections for distribution to various locations. Report sections are separated by an owner identification, such as a department code, or an address, or perhaps a user identification. To perform the distribution, this identification information must appear somewhere on the report. In many cases this identification information can be found in the heading of a report, or a banner page which marks the separation point.

For companies that produce many large reports, report distribution can be a time consuming manual process.

The Report Distribution Facility provides a solution to this problem. While a large report is still in machine readable form, the Report Distribution Facility can read and separate the report into smaller sections. Under ROPES, the separated report sections are copied to new or existing ROPES reports. If the receiving ROPES reports are already assigned to printers at various locations, the distributed reports become immediately available for printing.

Report Distribution Input

The ROPES Report Distribution Facility can accept two sources of input at this time. The input report may be in the form of an existing ROPES report, or it may be a sequential input file accessible through the CICS Extra-Partition Transient Data facility.

In a future release of ROPES, additional sources of report input will become available. Indirect input from the JES spool is possible now by using the ROPES Spool Transfer Utility.

ROPES Report Distribution requires that the input report contain American National Standards Institute (ANSI) control characters. These control characters mark the spacing and page breaks in the report. A skip to channel one ANSI control character (“1”) must be placed in the first column of the first line of every new page. ROPES Report Distribution uses this control character to start the processing of report heading information.

Report Distribution Processing

ROPES Report Distribution uses the input report’s heading information to determine the target ROPES report. This is accomplished by inspecting a maximum of 20 lines of the heading text, and using segments of the text, or translations of the text to build an output ROPES report name. Once the output ROPES report name is determined, the input page is copied to the output report. If multiple pages of input report data contain the same heading text, then these pages will be copied to the same output report.

Each page, marked by the ANSI control character, is read and copied to an output report until the entire input report is processed.

Distribution Processing Options

When making a request to distribute a report, several options are available governing the disposition of the input report and the creation or use of the output report(s).

Using Existing Output Reports

ROPES output reports should be pre-defined before performing a report distribution. This is the most efficient use of the facility. Output reports may be assigned to printers in advance making the output reports immediately available for printing. Defining the output reports and printers can be accomplished by using the ROPES report and printer maintenance transaction (ROMT).
Occasionally, it will not make sense to define all output reports in advance. The ROPES Report Distribution facility offers several ways to handle an input report that will not have one or more destination reports pre-defined.
The Unassigned Report Name feature allows input reports that don’t have pre-defined destination reports to be copied to a single output report. This single output report is called the **Unassigned** report name.

When making the distribution request, unassignable input report data may also be ignored. This means an input report page will be skipped if it does not have a pre-defined output ROPES report destination.

Distribution processing options also allow the distribution request to be terminated if an unassignable input report section is encountered.

Creating New Output Reports

New ROPES output reports can be created during report distribution processing. This is done by using the create option. If a new input report is being set up for distribution, it may be time consuming to build all of the destination reports using the ROPES maintenance transaction. The create option will cause all missing output reports to be created before the input report sections are copied to them. If the create option is used again for the same input report, the output reports will not be created again unless they are manually deleted by the ROPES maintenance transaction.

Report Distribution Definitions

Generation of the output or destination report name is controlled by the Report Distribution Definition. Multiple distribution definitions can be created or updated using the ROPES report distribution transaction RODM, and selecting **Distribution Definition Maintenance**.

Each distribution definition contains **line**, **column**, and **length** specifications which are used to select text from the heading of an input report page. Up to three line, column, and length selections can be made per distribution definition. Selected heading text can then be used as part of the output report name, or translated into a constant which is then used as a component of the output report name. Each component or report **key segment**, is concatenated or assembled together to make up the final output ROPES report name. This is known as Report **Key Generation**.

Selected heading text may also be appended to by using a character constant. For example, a distribution definition selects heading text at line 3, column 27, for a length of 4. The text found at that location is "**DEPT**". If the character constant "**0001**" was appended to this text, it would result in an output report name of **DEPT0001**.

Text attributes may also be enforced. For example, the input report can be rejected if the selected input text is not numeric. This feature will notify the user of a problem with the report data, and allow the problem to be corrected.

Left or Right Justification may also be used on selected heading text. This feature would allow a segment of selected text to contain a floating numeric value (that is it may have blanks preceding or trailing it). The text could then be right justified so that only the numeric characters are selected.

Once all report key segments have been selected, user exits may be employed to further translate or modify the segments. Exit names are specified in each distribution definition, and therefore may be different in each distribution definition.

Report key segments may also be translated by specifying a translation table name for the key segment in question. For example, a segment of text at line 3, column 14, and a length of 4 has been selected. The text segment contains "**EXPE**". If a translation table is employed for this segment, the text "**EXPE**" will be looked up in the translation table. If "**EXPE**" is found in the table, it will be replaced by the translation table value. The length of the replacing value will be determined by the **Key Generation** rules for that segment. If the look-up value for "**EXPE**" was "**DEPT**", and the key generation length was 3, then the new key segment value would be "**DEP**".

After each key segment is processed, the output ROPES report name is assembled from the key segments. The key segments are concatenated in the same order as the line, column and length specifications.

An illustration of the Report Distribution process can be found in Figure 1 on page 25. It shows how each page of report input is processed to determine the output report name for that page. Near the bottom of the figure is a heading line which indicates the **Line**, **Column**, **Length**, **Source Data**, **Operation**, and **Result** key segment. Under the operation heading, you will see **TR** which indicates the source data was translated, and **USE** which indicates the input data was selected and used as a key segment value.

Installation Requirements

If you have completed the ROPES base system installation process, you will be able to use the Report Distribution Facility. If you wish to use an Extra-Partition Transient Data file as input, you must define an Extra-Partition Transient Data destination first. This can be done by modifying the CICS Destination Control Table (DFHDCT). The input file must have a logical record length of 255, and should be blocked for better performance.

Preparing For Distribution

Before an input report can be processed by Report Distribution, the following steps must be taken to set up for processing.

1. The input report must be examined to determine which heading information will be used for generating the output report names.

2. The Report Distribution Definition must be created to define the line, column and length controls which are used to select the heading text. Key generation controls must also be specified to control the assembly of the output ROPES report name.

3. If any translation tables are specified in the Key Generation controls, they must be created using Report Distribution's Translation Table maintenance.

4. If the input report is a sequential file instead of a ROPES report, then an Extra-Partition Transient Data Destination must be defined to CICS. The destination entry can be added by modifying the CICS Destination Control Table (DCT).

5. All output reports should be defined to ROPES before running Report Distribution. This can be accomplished by using the ROPES maintenance transaction ROMT.

If there are too many output reports to define them manually, then the Report Distribution **Create Option** can be used to create the required output reports. Before using the create option, be certain the Report Distribution Definition is producing the desired output report names.

6. It is recommended that the first attempt to distribute a report be done using the **Terminate** option. If Report Distribution does not find the desired output report, it will terminate, and display the report it could not find. This is useful when trying to determine if the Distribution Definition parameters are correct.

The above steps should be followed when setting up a new report for distribution.

Using Report Distribution

The Report Distribution Facility is initiated at a CICS terminal using the **RODM** transaction. To use the **RODM** transaction, clear the screen, enter "RODM" and press the ENTER key. In response, the Report Distribution Primary Menu is displayed.

Report Distribution Screens

The first line of the screen contains the date, time, title, panel type, and panel name. The two last significant lines of the screen contain the ROPES message line, and the program function key definitions. All screens in the Report Distribution Facility have this format.

The Report Distribution Facility uses Basic Mapping Support maps with extended attributes. To achieve the best screen appearance, it is recommended that you use these screens on terminals defined with extended attributes and color.

Report Distribution screens employ an additional data transmission saving feature. If the same screen is displayed more than one time, the screen will not be cleared and re-displayed in its entirety. The DATAONLY option is used to send the data, and not the entire map.

All Report Distribution transactions are pseudo-conversational. Although this is beneficial from a performance standpoint, you should be aware that Report Distribution screens may be interrupted by programs that send messages to CICS terminals.

Function Key Definitions

The following function keys are supported in this version of the ROPES Report Distribution Facility.

PF1 = HELP This key will invoke the help facility. Help information is displayed for the field pointed to by the cursor. The cursor may be in any position of the field. If help information is not available, the table of contents is displayed.

PF3 = EXIT This key will save the last update and exit, or just exit from the current screen to the previous screen level.

PF4 = RETN This key will cause a return to the highest screen level. For the Report Distribution Facility, that is the Primary Menu screen.

PF7 = PREV This key will display the previous page of data items in a **LIST** screen, or the previous record definition in a data entry type screen.

PF8 = NEXT This key will display the next page of data items in a **LIST** screen, or the next record definition for a data entry type screen.

PF9 = CNCL This key will cancel an outstanding update, add, or delete request. When responding to a confirmation message, the CNCL key may be used to abort the current operation. The cancel key does not cause a return to the previous screen level.

Other function keys are not supported at this time. Function key assignments may be changed, however, during the installation of ROPES. Please consult the installation section of the Administrator's Guide.

The Help Information Screen

The "Help Information Screen" is displayed whenever the HELP function key is pressed (PF1), and when the help file is allocated and defined to ROPES.

The "Help Information Screen" can display the contents of a complete manual including a table of contents. Help is also context sensitive. If you press the HELP key while the cursor is within the bounds of a field, help will attempt to retrieve help information about that field. If the field's help information cannot be found help will display the table of contents. *The help facility is available for Report Distribution only at this time.*

Using selection controls at the top of the screen, you can position to any chapter or heading. Up to 4 levels of heading can be accessed directly. Location can also be performed by page number using the locate command. A sample of the Help Information Screen appears in 2 on page 26.

Protected Help Screen Fields

The only protected area of the help screen is the help text display area which is located in the lower half of the screen. Help information text cannot be changed.

Unprotected Help Screen Fields

The unprotected help screen fields occur at the upper half of the screen. These fields can be used to position by chapter, heading, or page number. A detailed description of each input field can be found below.

Command

The command input field allows entry of two commands at this time. These commands are **LOCATE** and **RESET**. Locate is used with the **Page** field which is also an input field. To locate a page simply enter "**LOCate**" in the command input field, and the page number in the "Page" field.

The "**RESET**" command is used to clear or reset the memory pointer for the table of contents. Help maintains the last location you examined in the table of contents so that you can go back to that location after examining the help text. This feature is helpful if you go back and forth between the table of contents and the help text numerous times. After using the RESET command, help will position to the start of the table of contents if you access the table of contents.

A sample of the help information screen positioned at the table of contents is shown in the figure below.

Page

The "Page" field displays the current and highest page numbers for the help text being displayed. The current page number can also be modified to locate a specific page in the help document. This is accomplished by using the "**LOC**ate" command.

Scroll

The "Scroll" field specifies the scroll value that will be used when the scroll keys (PREV/NEXT) are pressed. The possible values for the scroll field are; MAX, TOP, END, and FULL. The default scroll value is "FULL". Note: *The number of lines of a "FULL" scroll is determined by the LPP parameter of the load help utility function. The LPP parameter specifies the number of lines each* *page of the help document will contain. The default number of lines is 11 which is the size of the help display area on the help screen.*

Chapter

The "Chapter" field can be used to locate a chapter heading in the help text. This function is almost equivalent to a find request except it only positions to a chapter heading. The value entered into this field is CASE sensitive. If the chapter heading you wish to locate is in upper case, then you must enter the chapter heading in upper case.

Heading (1-4)

The "Heading" fields can be used to locate a heading level in the help document. The function of this field is similar to a find request except that it finds a heading level only. This heading field and all other heading level fields must be entered alone, and not in conjunction with another heading level field or chapter field. The heading level fields are case sensitive. If you wish to locate a level 3 heading, then you must enter just the heading text in the level 3 heading field.

In the table of contents each heading level is indented 2 spaces to the right of the previous heading level. The chapter heading is always aligned to the left boundary in the table of contents.

Returning To ROPES

To return to a ROPES screen press the EXIT key (PF3).

The Primary Menu Screen

The Report Distribution Primary Menu screen is initiated by entering the **RODM** transaction at a CICS terminal. This can be accomplished by clearing the screen, entering "**RODM**" and pressing the ENTER key.

Report Distribution will respond by displaying the Primary Menu shown in **Figure 3** on page 26.

The Primary Menu offers three selections. The first, **Perform A Report Distribution**, allows the distribution request to be entered. The second, **Distribution Definition Maintenance**, allows the distribution definitions to be created, updated, or deleted. The last, **Translation Table Maintenance**, allows translation tables to be created, updated, or deleted. The selections are made by entering the number of the selection near the bottom of the screen after the "===>" literal. If you enter an invalid selection number, a message will ask you to re-enter the selection.

When using the Primary Menu, pressing the **END** key causes a return to CICS. Pressing the **RETN** key displays the primary menu again.

Distribution Definition Index Screen

The Distribution Definition Index screen presents a list of distribution definitions to the user. At the top line of the screen, the panel type field shows the word **LIST**. All list screens have a similar format. A sample of the distribution definition index screen appears in **Figure 4** on page 27.

Command Input Field

Below the title line appears the **Command** input field. The Command input field is not functional in this version of Report Distribution. It is present however, to facilitate future enhancement.

The Option Field

On the extreme left side of the screen, an **OPTION** field will be present for every record. The option field allows you to perform various operations on the record being displayed. Normally, these operations will be add, change, delete, locate, or perform. An information line at the top of the screen displays the available options. If an invalid option is selected, a message will appear on the message line indicating the error. This message will appear in red for terminals which display in color.

The Distribution Definition Index screen will allow the **(L)**ocate, **(A)**dd, **(C)**hange, and **(D)**elete options.

The Locate Option

The locate option allows a distribution definition to be located. This is accomplished by entering an "**L**" in the option field of any definition display line, and over-typing the **Distribution Name** with a complete or partial name. The locate function will reposition the list starting with the definition name most closely matching the name you requested. For example, to reposition the definition list to all definitions starting with the letter "**D**", enter the locate option (**L**) under the **Opt** heading, enter the letter "**D**" in the first column of the **Distribution Name** field, erase the rest of the distribution name, and then press ENTER.

The Add Option

The add option works in a similar manner as the locate option. However, when entering the **Distribution Name** field, the entire distribution name must be entered. The distribution name you enter will become the name of the definition added. When using the add option, the cursor does not have to be positioned at an empty distribution definition line. You can over-type an existing distribution definition with the distribution name you wish to add. Multiple add requests may also be made at the same time. Once the add request is entered, a data entry screen is presented allowing you to enter the fields of the distribution definition. If you requested multiple additions, a blank data entry screen will appear in sequence for each definition you requested. Messages will appear in the message area requesting you to confirm the addition of every record, and to indicate the completion of all additions.

Once the add requests are completed, you may use the scroll keys to scroll forward and backward through the recently added definitions. Note however, the scroll keys are not active until all requested records have been added. To return to the distribution definition index, you must press the END key.

The Delete Option

The delete option allows existing distribution definitions to be deleted. For the delete option, you would not over-type the **Distribution Name** field. Simply locate the entry you wish to delete, and then place a "**D**" under the **Opt** heading for that line. When you press ENTER, report distribution will present the data entry screen for that distribution definition. A message will appear in the message area asking you to press ENTER to confirm the delete. If you decide not to delete the distribution definition, press the CNCL key. This will cancel the delete.

You may delete more than one definition at a time. If you select multiple distribution definitions for deletion, each definition will be displayed in sequence by a data entry screen. You must either press the ENTER key to confirm the delete, or press the CNCL key to cancel the delete. When all requested deletes are processed, a message will appear in the message line confirming this. You must press the END key to return to the index screen.

The Change Option

The change option allows existing distribution definitions to be modified. You can select the definition to be updated by locating it first, and then entering a "**C**" under the **Opt** heading next to the located definition. After you press enter, a data entry screen will present the definition to you for update. You may now modify fields, and press enter to update the record. If you wish to cancel the update you must press the CNCL key in response to the confirmation message.
If you selected more than one definition for update, each definition will be presented to you in sequence. When all updates are completed, a message will appear in the message line confirming this. When the updates are completed, you may either scroll forward or backward using the NEXT and PREV keys, or return to the index screen by pressing the END key.

Initially, the Distribution Definition Index Screen will have two sample entries in it. After you have tested with these entries, and added more of your own, you may delete these entries.

The Distribution Definition Screen

The Distribution Definition Screen is the data entry screen that is displayed after the Distribution Definition index item is selected for update, add, or delete. A sample of this screen is shown in **Figure 5** on page 27.

Field Definitions

Each field is described below proceeding from left to right, and starting at the line below the title line of the screen.

Distribution Name

The Distribution Name field identifies the Distribution Definition record. This field may contain up to 12 alphanumeric characters.

Description

The Description field allows a short description to be entered about this distribution definition. This field may contain up to 28 characters.

Created

This field is automatically filled in by Report Distribution when a definition is added. This field contains the date the distribution definition was added.

User Id

This field is not used by Report Distribution at this time.

Data Location (Heading)

This heading defines a section of the screen where report heading text locations can be defined. Allowance is made for three separate heading text locations. These are numbered from left to right, 1 through 3.

Line (Data Locations 1, 2 and 3)

The line field defines the line number where an occurrence of heading text can be found. The line number is relative to 1, and cannot exceed a maximum of 20. This limitation is indicated on the right hand side of the screen. The line number must be numeric. Care must be taken when examining heading text to determine the correct line number. Improper definitions may cause incorrect report names to be generated or used.

Column (Data locations 1, 2 and 3)

This field defines the column location for the heading text that will be selected. This field is used in conjunction with the line location field to create a coordinate for the heading text that is to be selected. This number may be in the range of 1 through 255 and must be numeric.

Length (Data locations 1, 2 and 3)

This field defines the length of the heading text selected by the **Line** and **Column** fields of the same number. The length field must be numeric and must be in the range of 1 to 35.

Data Attributes (Heading)

The data attribute fields correspond to the **Line**, **Column**, and **Length** fields of the same number. The data attribute fields define the attributes of the heading text selected by the data location coordinates. Data attribute fields cause the selected heading text to be checked or justified before use by Report Distribution. If a selected heading text field is marked as numeric here, and the input report contains non-numeric data, the distribution will be terminated. An error message will be displayed to report the error. Field justification may also be used to justify text within the selected heading text area. This feature is useful when a report heading contains text that floats within a certain range of columns, and this text is important criteria for generating the destination (distributed to) report name.

Type (Data Attributes 1, 2 and 3)

The type field controls whether the input report data at the specified coordinate, must contain alpha-numeric, or numeric characters. The type field must contain either A(lphanumeric) or N(umeric). Data that does not conform to the attribute will be rejected during report distribution processing.

Justify (Data Attributes 1, 2 and 3)

The justify field controls justification on the input heading text at the specified coordinate during report distribution. The justification is performed internally, and passed on as a result to the **Key Generation** phase where the destination report name is constructed. For example, if a section of heading text located at line 3, column 24 contained the value " 147-89   ", right justification would cause the field to be changed to "   147-89". This would enable the Key Generation process to extract the value "89" and use it as part of the destination report name. Values permitted for this field are **L**(eft), **R**(ight), and **A**(sis).

Key Generation (Heading)

The **Key Generation** area of the screen contains parameter fields for controlling the construction and assembly of the destination (output) report name(s). For each page of input report, a new output report name may be constructed. This is entirely dependent on the content of the heading text. If the heading text on several pages remains exactly the same, these pages will be copied into a single destination or output ROPES report.

Key Length (Key Generation 1, 2 and 3)

The Key Length field controls the size of one of three report key segments. If heading text were selected, that always contained the word "DEPARTMENT", a KEY LENGTH of 3 would cause this report key segment to be truncated to a value of "**DEP**". If the field were **RIGHT JUSTIFIED**, the key segment would be "**ENT**". If the remaining two key segments were "**014**" and "**89**", a final destination report name of "**DEP01489**" would be generated. On each page of input the "**014**" and "**89**" key segments might change. This would cause other destination reports names to be generated. The sum of all **KEY LENGTH** fields cannot exceed eight). This is due to the fact that the ROPES report name cannot exceed a maximum of 8 characters.

Constant (Key Generation 1, 2 and 3)

The Constant field allows you to append a constant text value to any of the three key segments you have selected. The constant will be appended to the heading text selected by the **Data Location** specifications. The Key Generation **Constant** must not be used if you are going to specify a value for the Key Generation **Lookup Table**. If you use the **Constant** parameter, the Key Generation **Length** field must equal the combined length of the heading text selected, and the constant. Remember, the total size of all key segments must not exceed eight.

Lookup (Key Generation 1, 2 and 3)

The **Lookup** field specifies the name of a look-up table to be used for translating the selected heading text into another value. The selected heading text is used as an argument key to access the table entry directly. If a match is found, the selected heading text will be replaced by the translation value from the table.

It is important to remember that the selected heading text, and resulting translation value, reside in a field that is internal to the Report Distribution process. We have been calling this internal field the **Key Segment**.

When a lookup table is used, the Key Generation **Key Length** must be less than or equal to the length of the translation value. Once again, remember the total of all **Key Length** fields cannot exceed eight.

If the selected heading text has been right or left justified under the control of the **Data Attributes** Justify parameter, then the translation table entry key must also be justified. This is done when you add the table entry, and enter the **Data Value Used As Input** field. Please refer to the section on Translation Table Maintenance for more information on adding translation table entries.

User Exit (Key Generation 1, 2 and 3)

The **User Exit** field specifies the program name of a user exit to be used in further modification of a **Key Segment**. The user exit is invoked only if an exit name is specified. The user exit is invoked after the **Constant** value or **Translation** value has been set. In other words, the user exit is invoked after the Key Generation process.

The user exit is called using an **EXEC CICS LINK** command. The LINK request passes the standard ROPES COMMAREA. For assembler language exits, the following fields are used to pass the key segment value, key segment length, return code, and message text fields.

COMINPT - Key Segment text value (up to 100 characters).

COMTLEN - Actual Key Segment Length (4 byte binary field).

COMRETC - Return Code field. A non-zero binary value returned in this field indicates no user exit modification took place (2 byte binary field).

COMRETD - A message text field, which can be used by the user exit to return a message (78 characters).

If you are writing a user exit in a language other than assembler, you will have to copy the **COMMAREA** definition from the appropriate ROPES language support library. In some cases, the COMMAREA field names may be slightly different, but you will be able to associate these names with the assembler names by examining the field descriptions.

When your exit receives control from the Report Distribution module, the **COMINPL** and **COMINPT** fields will contain the length and text of the selected key segment. You may modify the key segment text (COMINPT), and the key segment length (COMINPL), but you must not set a new key segment length that exceeds the original length. If you do, it will result in a final output report name which exceeds eight (8) characters in length. If the output report name exceeds a length of eight, it will produce an error, and terminate the report distribution process.

Key Segment Assembly

The **Key Segment Assembly** fields control the final phase which produces the output, or destination, report name for the input page being processed.

The **Key Segments** which have been selected (numbers 1, 2 and or 3), are concatenated together in the same order they were selected. For example, if the internal key segment fields contained "**REP**", "**D1**", and "**A14**" respectively, then the final output report name would be "**REPD1A14**". If key segments 1 and 2 were the only key segments selected, then the output report name would be "**REPD1**".
After the output report name is assembled, it is possible to change the name one last time by using the **Key Segment Assembly** parameter fields.

Use As Is (Key Segment Assembly)

The **Use As Is** parameter directs Report Distribution to use the final output report name or to allow further modification by employing a translation table or user exit. The **Use As Is** parameter may contain one of two values, either **Y**(es) or **N**(o). The value **Y** means the output report name will not undergo further modification. The value **N** means the output report name will either be translated to another value or modified by a user exit.

Use This Exit (Key Segment Assembly)

This parameter specifies the user exit program used for final modification of the output report name. The exit is not used if the name is absent.

The user exit is called using an **EXEC CICS LINK** command. The standard ROPES COMMAREA is passed to the user exit. The final output report name is passed in field **COMINPT**, and the length of the output report name is passed in field **COMINPL**. The user exit may modify the output report name to any new value. If the output report name's size has changed, the new length must be placed in the **COMINPL** field.

If the user exit fails to modify the output report name, it may either terminate the process by setting a non-zero binary value in the **COMRETC** field, or return normally without changing the input values. The field **COMRETD** may be used to set an error message if the user exit has terminated the process with an abnormal return code.

Lookup Value In This Table (Key Segment Assembly)

This parameter specifies the **Lookup Table** name to be used for translating the final output report name to a new value. Note, the translated report name will retain the length of the assembled output report name which was used as input to the translation.

Program Function Keys

The Program Function key assignments are displayed at the bottom of the Distribution Definition screen. Please refer to the section **Function Key Definitions** for an explanation of their use.

Translate Table Index Screen

The Translate Table Index screen is a **LIST** type screen. The Translation Table Index screen is provided to display and maintain the Report Distribution translate tables.

Under each translate table are translate table entries. The translate table entries are displayed under another **LIST** screen called the **Report Distribution Translate Table** screen. A sample of the Translate Table Index screen is shown in **Figure 6** on page 28.

Below the screen **title** line is the **Command** input line. The command input line is not functional in this version of Report Distribution, but is present for future enhancement.

Below the command input line is an options information line, indicating the options available for this **LIST** screen. A heading line appears after the options information line. The heading line indicates the **Opt**, **Translation Name**, **Added** (date), **Updated** (date), and **Description** fields will be listed for each table definition below. Initially there are two translate tables defined; ROPETABLE01, and ROPETABLE02. These translate tables are provided as samples for testing the Report Distribution Facility. Once you have run the installation test and defined your own translation tables you may delete the samples.

For each translate table entry on the screen, the **Translation Name**, and **Description** fields can be modified. When you add a translate table, you will need to over-type both of these fields to enter the translate table name, and the description.

Translate Table Index Options

The options available to you on the Translate Table Index screen are; **L**(ocate), **A**(dd), **C**(hange), and **D**(elete). Any of these options may be entered under the **Opt** heading, to the left of the **Translation Name**.

Locate Option

The **Locate** option allows you to position the index list to the translate table entered in the **Translation Name** field. The **Translation Name** entered may be partial or complete. If you enter a partial name, the index list will be positioned to the translate table name most closely matching the name you entered. For example, if you enter "**D**" in the **Translation Name** field, the list will be repositioned to the first translate table beginning with the letter "D". One locate request should be made at a time.

Add Option

Translate tables can be added using the add option. To add a table, enter the **A** option under the **Opt** heading next to any **Translation Name** entry in the list. Over-type the **Translation Name** field with the name of the translate table you wish to add. The translation name may be up to 15 characters in length. Next, tab to the **Description** field on the same line, and enter the desired description. Once you have typed in the description, press the ENTER key. If the add was successful, a confirmation message will be displayed.

Depending on the number of translate tables you have defined, it is possible that the table entry will not appear on the current screen. This is a result of the translate table names being presented to you in alphabetical order. If you cannot see your table entry after adding it, then do a **Locate** on your table name. This will reposition the list starting with your table name.

Unlike adding the Report Distribution Definition record, a data entry screen will not be presented to you after entering the Add option. This is due to the fact that the translate table is added immediately from the Translate Table Index screen, and no additional fields are required to complete the definition. This makes it possible to add multiple translate tables in the same screen iteration. Do this by simply placing an "**A**" on multiple lines under the **Opt** heading, then typing in the appropriate translate table names under the **Translation Name** heading. Once you have entered all the desired names, press ENTER. If you receive a message indicating the add(s) were successful, then all tables have been added. If you receive a message indicating a failure, one or more tables were not added.

Delete Option

To delete one or more translate tables and their entries, simply enter the "**D**" option next to the desired translate table name(s). After you press the enter key, a message will ask you to **confirm** the delete request. If you wish to delete the selected table(s), press the ENTER key. If you wish to abort the delete(s), press the **CNCL** key. If the delete was successful, the table entries will immediately vanish from the index list. The confirmation message will also indicate the number of table entries deleted for the last table deleted. If a message returns indicating an error deleting one or more entries, check your request and try again.

Change Option

The **Change** option performs two operations. It updates the translation table description field, and displays another **LIST** screen containing translate table entries. This screen is called the **Report Distribution Translate Table** screen. Since detailed information is given about this screen in the next section, it will not be discussed here. Once you have performed all necessary maintenance on the table entries from the Translate Table screen, you must press **END** key to return to the Translate Table Index screen.

While in this and other **LIST** screens, you may press the **PREV** and **NEXT** keys to scroll back and forth through the index list. Please refer to the section on **Function Key Definitions** for more information on the use of program function keys.

Report Distribution Translate Table Screen

The Report Distribution Translate Table screen is provided to maintain translate table entries. All translate table entries belong to a translate table. Translate table entries contain the argument and translate values required to translate report key segments from one value to another.

The Report Distribution Translate Table screen is a **LIST** type screen. It displays a list of translate table entries which can be updated, added or deleted. A sample of this screen is shown in Figure 7 on page 28.

Translate Table Screen Fields

Just below the title line are fields displaying information about the translate table that owns the translate table entries. These fields are **Table Name**, **Created** (date), **Updated** (date), **Description**, and **User Id**. These fields describe the table which was created under the **Translate Table Index** screen. Note, the User Id field is not implemented in Report Distribution, but is present to facilitate future enhancement.

Next is the options information line. The options information line shows the options which can be performed on the translate table entries. The valid options are **L**(ocate), **A**(dd), **C**(hange), and **D**(elete).

Immediately below the options line is a heading line. The heading line contains headings for the fields that will appear in the list. The headings are **Opt**, "**This Data Value Used as Input……**", and "**Will Yield this key…**". The fields under the headings "**Data Value Used as Input**", and "**Will Yield this key**", are the fields which make up a translate table entry. Both of these fields can be modified for the purpose of adding or changing a table entry.

Option Rules

Similarly to the Translate Table Index screen, one or more table entries may be added, changed, or deleted at the same time. Add and change operations are completed in one screen iteration. Delete requests are completed in two screen iterations. A combination of add, change, or delete requests may also be made at the same time.

Translate Table Entries

Each table entry is made up of a translation argument (under **This Data Value Used as Input**), and a translation value (under **Will Yield this key**).

Translate Argument Field

The **translation argument** should be equal to a segment of input report text which has been selected using the criteria in the **Distribution Definition**. The length and justification of this text must also match the selection criteria and data attributes set in the **Distribution Definition**.

If the **Distribution Definition** selects the input report text using a length of 14 characters, then the translation argument must also be 14 characters long. If the same segment is right justified within its 14 character length, then the translation argument must also be right justified. For example, the **Distribution Definition** selects input report text at line 2, column 21, for a length of 14. The selected input report text is "Dept 41   ". The **Data Attributes** in the **Distribution Definition** also call for this field to be right justified. The resulting key segment would then be "   Dept 41". To achieve a matching entry in the translation table for this text would require that you enter "   Dept 41" in the translation argument field. Note, the proceeding blanks would have to be entered in the translation argument field before entering "Dept 41".

Translate Value

The translation value is entered under the heading "**Will Yield this key**". The value entered here will replace the key segment data that was used to look up the table entry. The resulting key segment is then passed on to the **Key Segment Assembly** phase.

Adding A Translate Table Entry

To add a translation table entry, place an **A**(dd) under the **Opt** heading next to any blank or occupied table entry. Enter the desired translation argument, and translation value, over-typing an existing entry if necessary. After you have entered the required fields, press the ENTER key. If the addition was successful, a message will appear in the message line confirming this. Any failure to add the table entry will also be indicated in the message area.

Changing A Translate Table Entry

To change one or more translate table entries, enter a **C**(hange) under the **Opt** heading next to the desired table entry or entries. Over-type the translate argument and translate value fields with new values. Press the enter key when you have completed typing in your changes. If all changes were applied successfully, a message will appear in the message area confirming this. A message will also indicate when an update has failed. If a request has been made to update multiple table entries, and one of the updates has failed, the remaining updates will not be applied.

Deleting Table Entries

One or more translate table entries can be deleted by placing a "D" next to the desired table entry or entries. Once you have selected the entry or entries you wish to delete, press the ENTER key. In response, a message will return asking you to CONFIRM the delete(s). If you wish to confirm, press the ENTER key. If you wish to abort the delete(s), press the **CNCL** key.

Locating A Table Entry

To locate a translate table entry, place an "**L**" at any table entry line under the **Opt** heading. Then over-type the translate argument field with the text you are searching for. The argument text can be partially entered if you are not sure of the complete translate argument value. When you are done press the ENTER key. Locate will respond by repositioning the table entries to the
entry most closely matching the requested argument value.

Only one locate request can be made per screen iteration.

Report Distribution Requests

A Report Distribution request can be made by selecting option "**1**" from the Report Distribution primary menu. Option **1** on the primary menu is "**Perform A Report Distribution**". After you select option 1, the system will respond by displaying the **Report Distribution Definition Index** screen. Please see the section on the **Distribution Definition Index Screen** for detailed information about the screen layout.

The Distribution Definition Index screen will have one difference after selecting the "Perform A Report Distribution" option. This difference concerns the **options** that are available on the index screen. Only two options are available on the index screen. They are **L**(ocate) and **P**(erform). This is indicated on the "**Options:**" information line.

Selecting The Distribution Definition

To perform a report distribution, you must locate and select the distribution definition you wish to use. It will be helpful to examine the description field to make sure you have selected the proper definition. You may also use the scroll program function keys to locate the distribution definition. Only one perform request can be made at a time. Once you have located the definition, you must enter a **P**(erform) under the **Opt** heading next to the distribution definition you selected. Now press enter. The system will display the **Report Distribution Request Panel**.

Report Distribution Request Screen

The Report Distribution Request screen allows you to enter specific controls regarding the distribution request and how it will be processed. From this screen you may define the input source, processing options, and the use or creation of output reports. A sample of the Report Distribution Request Panel is shown in **Figure 8** on page 29.

Field Definitions

Each field is described below proceeding from left to right, and starting at the line below the title line of the screen.

Distribution Name

The distribution name field contains the name of the distribution definition being used for this report distribution. This field is filled in by the system after a perform request is made against the distribution name from the Distribution Definition Index screen. This field cannot be modified.

Created (date)

This field displays the date the distribution definition was created. This field cannot be modified.

User Id

The user identification field is not being used by the Report Distribution system at this time. However, the field is present for future enhancement.

Processing Options (Heading)

The processing option fields control the input report source, and the final disposition of the input report data if the input source is a ROPES report.

ROPES Report (Input Source)

This field specifies the name of the ROPES report that will be used as input to report distribution. This field must contain a valid ROPES report name, and must not exceed eight characters in length.

All pages of the report data must contain ANSI control characters in column one. A skip to channel one page control must be present at the top of each new page of input.

Spooled Job And Job Id (Input Source)

The spooled job and job Id fields are not functional in this release of report distribution, but are present on the screen for future enhancement. If you attempt to enter a spool job name, and job Id, Report Distribution will return a message in the message area indicating this input source is not available at this time.

TD Destination (Input Source)

This field must contain the name of a valid Extra-partition Transient Data destination. The extra-partition destination Id must be defined in the CICS Destination Control Table (DCT).

The extra-partition data set must exist as a physical sequential file with a logical record length of 255. The data set must also contain report data which contains ANSI controls in column one of each report page.

Open/Close Destid (Input Source)

This field allows you to specify whether the input transient data destination should be dynamically opened and closed by report distribution. This is useful if the extra-partition data set must be closed to CICS to allow re-population of new report data. The only possible values for this field are **Y**(es) and **N**(o).

Transient Data set (Input Source)

This field allows you to specify the name of the extra-partition data set to be dynamically allocated and deallocated by report distribution. If a valid data set name is entered in this field, report distribution will dynamically allocate and deallocate this file during the initialization and termination phases of report distribution. The dynamic **Open/Close** feature must be set on (**Yes**) if the dynamic allocation feature is used.

If the **Transient Data set** field is left blank, dynamic allocation will not be performed. If you do not use dynamic allocation, you must add the **DD** card specifying the extra-partition data set name in the CICS JCL.

The dynamic allocation feature uses SVC 99. If the **RUNAWAY** time interval in CICS is set too low, it may result in an **AICA** abend during dynamic allocation. Please be certain that the **ICVR** parameter in the CICS System Initialization Table (DFHSIT) is set to a value that will accommodate a single file dynamic allocation request.

If you receive an error message from report distribution indicating dynamic allocation has failed, you will need to look up the dynamic allocation return and reason codes to determine the cause. These codes can be found in the IBM **System Programming Library: Job Management** manual.

Delete Input Data

If you selected a ROPES report as your input source, it is possible to purge the input ROPES report by using this option. To purge the input report data, code a **Y**(es) in the **Delete Input Data** field. If you do not wish to delete the input data, code a **N**(o). This option is not valid if your input source is a transient data file. The only correct values for this field are **Y**(es) and **N**(o).

Error Handling Options (Heading)

The next set of parameter fields allow for different processing paths to be taken in the event the output (distribution) reports do not exist.

Terminate If Errors Occur

This parameter directs report distribution to terminate or continue processing if an output report name is not defined to ROPES, and report distribution is attempting to write to the report.

If the value for this field is set to **N**(o), then report distribution will continue processing. The output report data will either be placed into an unassigned report bucket, skipped entirely, or placed into a dynamically created report. If the value for this field is set to **Y**(es), then report distribution will terminate, and an error message will be displayed indicating the output report definition that was missing. Only two values are correct for this parameter, **Y**(es) and **N**(o).

You will find it helpful to test your Report Distribution Definitions by running with the terminate option set on (**Y**es). This will allow you to verify if the correct output report name is being generated by report distribution. If the output report name (indicated in the error message) is not close to what you expect it to be, there is an error in your Report Distribution Definition. Perhaps the line, column, and length definitions are incorrect, or a translation table you may be using does not have the correct translate argument value. There is also a possibility that you may be using the wrong input report data.

Ignore Undefined Reports

This control field may be used depending on the value of the **Terminate** control field. The **Terminate** and **Ignore Undefined Reports** parameters are mutually exclusive and cannot be set to **Y**(es) at the same time. The Ignore Undefined Reports flag may only contain a **Y**(es) or **N**(o) value.

If the ignore flag is set to yes, report distribution will either completely skip the report, or write its unassigned pages to the **Unassignable Report Id**.

Create Undefined Reports

The create undefined reports parameter is mutually exclusive with the **Terminate**, and **Ignore Undefined Reports** parameters. The Create Undefined Reports flag must be off (set to No) if either the terminate or ignore undefined reports flags are on (set to Yes). Otherwise, the Create Undefined Reports flag may be turned on. This is done by entering a **Y**(es) in the **Create Undefined Reports** field. The only other correct value for this parameter is **N**(o).

If the create flag is set to yes, then the **Report To Use As Model** must also be entered. This field is discussed next.

Report To Use As Model

This parameter specifies the name of an existing ROPES report which can be used as a model for creating a new report definition. Entering this parameter is required when the **Create Undefined Reports** parameter is set to yes. Otherwise, it should be left blank.

Spool Unassignable Data To Report

This field establishes the name of a ROPES report which can be used as a bucket for all input report data that cannot be placed in an existing destination ROPES report. This parameter may be entered only when the **Ignore Undefined Reports** parameter is set to yes.

[redacted]

Figure 1 Distribution Process

```
Jul 11, 2002 01:14pm      ROPES Device Control Block Index    INPUT    S10HLP0
------------------------------------------------------------------------------
Command ==>                                                Page:    69 Of 00165
Chapter:                                                        Scroll: FULL
Heading:
   Heading:
      Heading:
         Heading:
------------------------------------------------------------------------------
Line (Data Locations 1, 2 and 3)

          The line field  defines the  line number where  an occurrence  of
          heading text can be found.  The line number is relative to 1, and
          cannot  exceed a maximum of 20.   This limitation is indicated on
          the  right hand  side of  the screen.   The  line number  must be
          numeric.    Care must  be taken  when  examining heading  text to
          determine  the correct  line  number.   Improper definitions  may
          cause incorrect report names to be generated or used.



ROPES Version 8.0                (C) Copyright 1991-2002 Axios Products, Inc.
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN 07 PREV 08 NEXT 10 TBOC
```

Figure 2 The Help Information Screen

```
Jul 11, 2002 01:15pm    ROPES Report Distribution Main Menu   MENU     R51RD01
------------------------------------------------------------------------------
                                 Primary Menu



                    1.   Perform A Report Distribution

                    2.   Distribution Definition Maintenance

                    3.   Translation Table Maintenance







  Select an option from those listed ===> 01

ROPES Version 8.0                (C) Copyright 1991-2002 Axios Products, Inc.
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN
```

Figure 3 The Primary Menu Screen

```
Jan 26, 1998 09:21am    Report Distribution Definition Index  LIST     R51RD02
------------------------------------------------------------------------------
Command ===>

Options: L=Locate, P=Perform.
Opt   Distribution Name   Added       Updated     Description
      DISTDEF01           04/23/91    12/30/94    Sample Distribution Def. 1
      DISTDEF02           04/23/91    12/29/94    Sample Distribution Def. 2
      EMPRPT00            04/28/93    12/28/94    Test













ROPES09066 - Bottom of list reached. (f)
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN 07 PREV 08 NEXT
```

Figure 4 Distribution Definition Index Screen

```
Jan 26, 1998 09:25am    Report Distribution Definition Panel  DISPLAY  R51RD04
------------------------------------------------------------------------------
Distribution Name: DISTDEF01      Description: Sample Distribution Def. 1
 Created: 04/23/91                                    User id: QAX006
Data Location:             1         2         3   Data Segment Number
                 Line:     1         3         3   Line to find data (1-20)
               Column:    29        15        19   Column to find data (1-255)
               Length:    19         3         2   Length of data (1-35)

Data Attributes:
                 Type:     A         N         N   A(lphanumeric) or N(umeric)
              Justify:     L         R         R   L(eft), R(ight) or A(s is)

Key Generation:
   Key Length:           3           3           2   Segment Length (1-8)
     Constant:                                         Append to constant
       Lookup: ROPETABLE01                                   Use this table
    User Exit:                                         Use this exit routine

Key Segment Assembly:  Use as is: Y  (Y or N)  or  Use this exit:           or
                       Look up value in this table:
ROPES09043 - Please press the ENTER key to MODIFY the current record.
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN 07 PREV 08 NEXT 09 CNCL
```

Figure 5 Distribution Definition Screen

```
Jan 26, 1998 09:23am   Report Distribution Translation Index  LIST     R51RD03
------------------------------------------------------------------------------
Command ===>

Options: L=Locate, A=Add, C=Change, D=Delete.
Opt   Translation Name    Added       Updated     Description
      IDMSPMON01          06/17/93    04/21/94    Test Table
      LTPLKUP             04/28/93    04/29/93    Department Test
      ROPETABLE01         04/23/91    01/24/95    Translt for Extra-TD input.
      ROPETABLE02         04/23/91    01/25/95    Sample Translate Table 2











ROPES09068 - Bottom of list reached.
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN 07 PREV 08 NEXT 09 CNCL
```

Figure 6 Translation Index Screen

```
Jan 26, 1998 09:26am    Report Distribution Translate Table   LIST     R51RD05
------------------------------------------------------------------------------

Translation Table Name: LTPLKUP          Created: 04/28/93  Updated: 01/26/95
Description: Department Of Justice Test                     User id: QAX006

Options: L=Locate, A=Add, C=Change, D=Delete
Opt    This Data Value Used as Input......       Will yield this key...
       BUN                                            LTQEMP00
       POM                                            LTPEMP00











ROPES09068 - Bottom of list reached.
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN 07 PREV 08 NEXT 09 CNCL
```

Figure 7 Translate Table Screen

```
Jan 26, 1998 09:27am     Report Distribution Request Panel    INPUT    R51RD06
------------------------------------------------------------------------------

Distribution Name: DISTDEF01         Created: 04/23/91   User Id: QAX006


Processing Options:
        Input source:
              1.   ROPES Report: REPORT01                    Enter report name
           or 2.    Spooled Job:            Job Id:          Enter Job name/id
           or 3. TD Destination:                             Enter TD Dest. id
              Open/Close Destid: N                           Y(es) or N(o)
              Transient Data set:
              Delete Input Data: N  (with options 1 and 2)   Y(es) or N(o)

Error Handling Options:
                   Terminate if errors occur: N              Y(es) or N(o)
                    Ignore undefined reports: Y              Y(es) or N(o)
                    Create undefined reports: N              Y(es) or N(o)
                    Report to use as a model:                Enter report name
           Spool unassignable data to report: UNASREPT       Enter report name
ROPES38023 - Please enter distribution request parameters.
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN
```

Figure 8 Distribution Request Panel

Job Submission (ROJB)

Introduction

The ROJB transaction will provide you with a facility to create, manage and submit job streams on-line under CICS.

The Job Submission facility has the following features:

- provides you with an SPF-like interface to modify your JCL and data.
- JCL streams are saved in a VSAM file thereby permitting repeated use.
- a member index screen is provided which enables members to be easily located and selected for update.
- space on the JCL data file is reused when members are deleted.
- JCL members may be submitted to the Job Entry System.
- a batch utility is provided to allow the copying of existing JCL members from a PDS to the JCL data file.
- the maximum number of lines that can be contained in a member can be controlled at the member level.
- existing JCL members can be included with your job stream during submit processing by using the '@@INCLUDE' control card.
- access to JCL members may be protected using the standard security mechanism provided by the SECUPD/SECEXIT installation options.
- your CICS userid can be included anywhere in your JCL during submit processing by using the '@@USERID' substitution variable in your JCL.

JCL members can be accessed through the member index screen. The member index screen is invoked by entering transaction **ROJB** at a cleared CICS terminal. JCL members can be located, added, changed, or deleted from the member index screen. A sample of the JCL member index screen can be seen in **Figure 9** on page 35.

Near the top of the JCL Member Index screen, command options are listed. These command options are entered along the left side of the member name under the '**Opt**' heading.

JCL Member Index Screen Options

The **L**(ocate) option can be used to locate members in the JCL member index list. To locate a member enter '**L**' under the '**Opt**' heading, and then over-type the member name with the member name you are trying to locate. The system will respond by repositioning the index list to the closest matching member name you selected.

The **D**(elete) option can be used to delete an existing member from the index list. The delete command removes all records associated with the member. To delete a member, enter a '**D**' under the heading '**Opt**' next to the member name you wish to delete, and press the enter key. The system will respond by asking you to confirm the delete. To confirm the delete, press the enter key. To cancel the delete, press the '**CNCL**' PF key. The cancel PF key is usually define as PF9.

The **A**(dd) option can be used to add/create a JCL member. To add a member, enter '**A**' under the heading '**Opt**' on any line of the index, and then enter the member name you wish to create under the member name heading. It is okay to over-type an existing member listed on the index screen. When you press enter, the system will respond by showing the Job Submission Facility Editor screen. This screen will allow you to enter the JCL statements that make up your member.

The **C**(hange) option can be used to modify the contents of a JCL member. Change can be used against existing members only. To change a member, enter '**C**' under the '**Opt**' heading next to the member name of the member you wish to change, and then press the enter key. You may also over-type the name of an existing member if you remember the name of the member you wish to edit. The system will respond by displaying the Job Submission Facility Editor screen. A sample of the Job Submission Facility Editor screen appears in **Figure 10** on page 35.

Job Submission Facility Editor

The editor menu shown in the figure above gives you the ability to add, delete or change lines in a job stream. Many of the commands and functions of this editor are SPF-like. The editor provides many of the primary commands necessary to the maintenance of your JCL, as well as most of the line commands provided by the SPF editor.

Primary Editor Commands

Near the top of the editor screen is the 'Command Input' field. This is where all editor primary commands must be entered. Supported commands are; **SAVE**, **CAN**cel, **RESET**, **L**ocate, **RENUM**, **TOP**, **BOT**tom, **COLS ON**, **COLS OFF**, **CAPS|CAPS ON**, **CAPS OFF**, **F**ind, and **C**hange. The bold highlighted portions of these commands represent the minimum characters that need to be entered to issue the command. A detailed description of each primary command appears below.
Command line input supports the following functions:

- **SAVE** Saves any updates made to JCL member content.
- **CAN**cel Cancels updates made to JCL members content.
- **L**ocate Locates a line of text, and causes repositioning of the JCL member's text so that the selected line number is the first line displayed in the text area of the editor screen. The locate command is entered with a line sequence number. The line sequence number may be up to 5 digits in length, and must be numeric. The line sequence number is entered one space after the locate command. Example: **L 2500**.
- **RESET** Resets any line commands in progress, or clears any find/change command parameters from memory.
- **TOP** Repositions the text so that the first line of the member is positioned at the first line of the text area.
- **BOT**tom Repositions the text so that the last line of the member is positioned at the first line of the text area.
- **RENUM** This command renumbers the sequence numbers by the increment value you select. The increment value is entered after the command and one space. The increment value must be numeric, and is limited to 3 digits. Renumbering is not really necessary since the editor does this automatically after every line command. The default line increment is 10.
- **COLS ON** Changes the heading line which immediately precedes the text area to a column scale line. The column scale line will remain until the COLS OFF command is used. The column scale line is useful when trying to determine the column range of a FIND or CHANGE command.
- **COLS OFF** Turns the column scale heading line off. The heading line reverts back to dashes.
- **CAPS** ON Turns upper case translation on. Text that is entered in lower case will be translated to upper case by the editor. If text has already been entered in lower case, you will have to press enter twice to modify the current page of text to upper case. Only data visible on the screen will be translated.
- **CAPS OFF** Turns upper case translation off. Text that is entered in lower case will remain in lower case, and text that is entered in upper case remains in upper case.
- **F**ind Allows text to be located in the JCL member. Found text is positioned at the beginning of the text display area, and the cursor is positioned at the start of the found text. The Find command may be entered with column ranges, and may search going forward or backward. Some examples of the find command appear below.
  FIND 'TEXT STRING' FIRST
  F "ANY STRING" 10 21 NEXT
  F 'STRING OF TEXT' LAST
  F ANYSTRING 10 16 PREV

The find command permits almost any special character to be used as a delimiter for enclosing the text string. Column range numbers must be numeric, and 1 to 3 digits in length. Numeric strings **MUST** be enclosed in string delimiters. Find commands are memorized so that a *repeat find* function key (PF5/RFND) can be used to find the next occurrence of the text.

- Change The change command allows a string of text to be changed. Change can apply to one occurrence, many, or **ALL** occurrences of a text string. Like the find command, change can also operate within a range of columns, and can search forwards and backwards. The change command's parameters are also memorized so that the *repeat change* key (PF6/RCHG) can be used to change the next occurrence of the string. Below are some examples of the change command.

  CHANGE 'STRING1' 'TEXT2' ALL
  C 'STRING1' 'TEXT2' FIRST
  C 'STRING1' 'TEXT2' NEXT
  C 'STRING1' 'TEXT2' 10 17 PREV

Other Editor Screen Fields

Several other input fields on the JCL editor screen can affect the operation of this screen, or provide input for the JCL member. These fields are; **Scroll**, **Description**, **Max Lines**, and **Text Display Area**. Explanations of these fields appear below.

- **Scroll** The 'Scroll' field controls the number of lines that will be moved forward/backward when a "scroll" function key (PF7/PF8) is pressed. The scroll value may be set to **FULL**, **HALF**, and **MAX**. Fourteen (14) lines of text can be displayed on the editor screen, so FULL will move 14 lines at a time. The HALF scroll value will move 7 lines at a time, and the MAX scroll value will move to the top or bottom of the member depending on which direction was used.

- **Description** The description field is used to set a description for the member being edited. This description will show up in the member index screen to the right of the member name. The description may contain up to 40 characters of descriptive text.
- **Max Lines** The max lines field is used to control the size of the JCL member being edited. When a JCL member is selected for modification, an internal work area is obtained which is equivalent in size to the maximum number of lines (from this field) times the size of each line of text (78 characters). The max lines value is either set by the JCL loader utility, or set to 500 lines by default when using the online editor. If the value in this field is too small for the JCL you are editing, you will receive a message indicating the internal work area has been exceeded. To correct this, increase the Max Lines value and save the member. Then re-open the member, and continue adding the statements that were required.

- **Text Area** There are fourteen (14) lines of text display area just below the description and max lines fields. On the left side of the text area are sequence numbers for each line. Sequence numbers cannot be turned off. To the right of the sequence numbers, text for your JCL member may be entered. Each line of text is a maximum of 78 characters. Updates to text are registered when the enter key is pressed.

Edit Line Commands

Edit line commands are entered over the sequence numbers next to the text lines they are intended to modify. Like the SPF editor, line commands can affect a single line, or a group (block) of lines. The commands and their single and block forms are listed below.

- **A** Specifies the target location for the line(s) being copied/moved will be **after** the sequence number this line command is entered into.

- **B** Specifies the target location for the line(s) being copied/moved will be **before** the sequence number this line command is entered into.
- **C** Copies a single line of text to the target location marked by the **A** or **B** line commands.
- **Cnnn** Copies nnn lines starting at the current sequence number location to the target location marked by the **A** or **B** commands.
- **D** Deletes a single line of text at the current sequence number location.
- **Dnnn** Deletes nnn lines starting at the current sequence number location.
- **M** Moves a single line at the current sequence number location to the target location marked by the **A** or **B** line commands.
- **Mnnn** Moves nnn lines starting at the current sequence number location to the target location specified by the **A** or **B** line commands.
- **R** Repeats the line at the current sequence number location 1 time.
- **Rnnn** Repeats the line at the current sequence number location nnn times.
- **I** Inserts one blank line at the current sequence number location.
- **Innn** Inserts nnn blank lines at the current sequence number location.
- **CC** Copies a group of lines starting at the location marked by the first 'CC' line command, and ending with the last 'CC' line command to the location marked by the **A** or **B** line command.
- **DD** Deletes a group of lines starting at the location marked by the first 'DD' line command, and ending with the last 'DD' line command.
- **MM** Moves a group of lines starting at the location marked by the first 'MM' line command, and ending with the last 'MM' line command to the target location marked by the **A** or **B** line commands.
- **RR** Repeats a group of lines starting at the location marked by the first 'RR' line command, and ending with the last 'RR' line command after the location of the first 'RR' line command.
- **RRnnn** Repeats nnn times the group of lines marked by the first 'RR' command, and ending with the last 'RR' line command.

Those line commands which permit a repeat or occurrence value to be used must follow this important rule. ***The repeat/occurrence number must follow the line command immediately, and it must also be followed by a single space.*** If the repeat/occurrence number is not followed by a space, it will revert back to a value of one (1).

Block lines commands may span more than one page of text. This is accomplished by entering a line command at the starting location, and then scrolling down one or more pages to the target location. The message move/copy command is pending will be displayed as you scroll to the target location. A primary find command may also be used to locate the target location.

Substitution and Include Options

You can embed the following control statement in your job stream to include other JCL streams when you submit your job.

@@INCLUDE member,firstline,lastline
where:
memberis the name of the JCL member to insert.
firstline is the number of the first line to insert.
lastline is the number of the last line to insert.

You can use any number of INCLUDE statements in the job stream. A blank must separate the command and the member name. Separate the firstline and lastline options from the member name and each other by commas. Do not embed any blanks in the statement.

@@USERIDwill include the logonid of the person logged on to CICS at any occurrence of the variable '@@USERID'. Substitution occurs during SUBMIT processing.

```
Nov 19, 1999 02:28pm       ROPES JCL Member Index Screen      LIST     R51JCI0
------------------------------------------------------------------------------

Options:  (L)ocate, (A)dd, (C)hange, (D)elete.
Opt    Name     Update    Userid                  Description
---  --------- ---------- --------   ----------------------------------------
 _   RUNBQOR  11/19/1999 BOB        BATCH QUEUE REORGANIZATION JCL
 _   RUNBUILD 11/19/1999 BOB        SETUP ROPES PARAMETER FILE JCL
 _   RUNLDJCL 11/19/1999 BOB        LOAD JCL MEMBERS UTILITY
 _   RUNRPIB  11/19/1999 BOB        RUN REPORT TO PRINTER INDEX UTILITY JCL
 _   TEST1    11/19/1999 BOB        TEST SUPERZAP JOB
 _
 _
 _
 _
 _
 _
 _
 _
 _

ROPES09066 - Bottom of list reached. (f)
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN 07 PREV 08 NEXT 09 CNCL
```

Figure 9 JCL Member Index Screen

```
Jul 11  2002 01:15pm    ROPES JCL Member Maintenance Screen   LIST     R51JCM0
------------------------------------------------------------------------------
Command ===>                                                       Scroll FULL
Job Name: RUNBQOR   Updated  Date: 11/19/1999  Time: 14:24:47   User: BOB
Description: BATCH QUEUE REORGANIZATION JCL              Max. Lines:      229
------------------------------------------------------------------------------
01500 //*        RUN THE ROPES REORGANIZATION UTILITY.                  *
01600 //*                                                               *
01700 //*****************************************************************
01800 //ROPEBQOR EXEC  PGM=ROPEBQOR
01900 //STEPLIB  DD    DSN=ROPES.LOADLIB,DISP=SHR
02000 //ROPEARC  DD    DSN=ROPES.ARCHIVE,DISP=OLD        ARCHIVE
02100 //ROPERIB  DD    DSN=ROPES.RIBFILE.BASE,DISP=OLD   ROPES
02200 //ROPEPCR  DD    DSN=ROPES.PCRFILE.BASE,DISP=OLD      DATA
02300 //ROPEQBR  DD    DSN=ROPES.QBRFILE.BASE,DISP=OLD         BASE
02400 //SYSLST   DD    SYSOUT=A               MESSAGE DATA SET
02500 //SYS006   DD    SYSOUT=A               PRINT REQUEST DATA SET
02600 //SYSUDUMP DD    SYSOUT=A               JUST IN CASE
02700 //SYSIPT   DD    *                      CONTROL CARDS
02800 /*

ROPES Version 8.0                (C) Copyright 1991-2002 Axios Products, Inc.
------------------------------------------------------------------------------
01 HELP 03 EXIT 04 RETN 05 FIND 06 CHNG 07 PREV 08 NEXT 09 CNCL
```

Figure 10 Job Submission Facility Editor

Liberty/Soft Output Interface

The Liberty/Soft product line provides a powerful set of graphic features which can dramatically improve the appearance of your reports. This can be accomplished without changing your reports or the applications that generate them.

Reports destined to be output on CICS printers can be transformed into graphically rich documents, and routed to output devices managed by VTAM, JES, CICS, or TCP/IP.

If you have purchased the Liberty/Soft product line to enhance the appearance of your reports, then ROPES can easily take advantage of the graphics features of Liberty/Soft by using the Liberty/Soft output interface of ROPES.

Control of the interface is managed through a series of parameter options in the ROPES Forms Control Block (FCB). ROPES printer definitions receiving output from Liberty/Soft must also be modified to set the SCSEJECT parameter value to 'N' (No). Samples of the ROPES FCB and DCB can be found in the ROPES distribution SOURCE library under the names RO$FLSPD and RO$DLSPD.

ROPES reports that are output using the Liberty/Soft tool must have a "Liberty/Soft" ROPES FCB associated with them. This assignment is made when modifying or creating the report definition under ROPES using the ROMT transaction. The report definition's **FCB Suffix** must be set to the name of the Liberty/Soft ROPES FCB (Example 'LSPD'). Multiple Liberty/Soft FCB's can also be created which have different Liberty/Soft output options. These new FCBs may then be associated with other selected ROPES reports.

The Liberty/Soft FCB has a new type of control statement in it. This macro control statement starts with "**TYPE=LSDEF**". The LSDEF macro control statement specifies the parameter options necessary for controlling the output of the ROPES report by Liberty/Soft. Please see the **ROPES Administrator's Guide** for a complete description of the LSDEF control statement parameters.

The ROPES E-mail Interface

Introduction

The ROPES e-mail interface provides a mechanism for routing ROPES report output to Internet or Intranet e-mail destinations. E-mail destinations are only limited by the TCP/IP and SMTP mail server configurations you have installed.

The ROPES E-mail Interface also supports attaching the selected report to the e-mail as a separate file. Two file attach formats are provided, plain text, HTML or PDF document formats. Reports delivered in HTML or PDF format are automatically converted to HTML or PDF format prior to delivery. Reports delivered in plain text format are delivered without conversion. File attachments are setup through new parameters in the ROPES Device Characteristics Blocks (DCBs). Attachments are delivered using the MIME 1.01 standard.

The ROPES e-mail interface requires TCP/IP for MVS, SMTP Mail Server for MVS, CICS Sockets, and CICS version 3.3 or higher. A minimum NJE configuration with at least the host node explicitly defined is also required. ***The name of the host node must match the name of the host as it appears in the TCP/IP configuration.***

APPLDATA Generation

Beginning with ROPES Version 14, the TCP/IP based data transmission functions, including this function, now generate the APPLDATA information made available in z/OS Communications Server V1R9 (and by PTF, rolled back to V1R7 and V1R8).

This data allows you to have a better understanding of the purpose of the ROPES active sockets connections. The associated application data is made available on Netstat ALL/-A, ALLConn/-a and COnn/-c reports, in SMF 119 TCP Connection Termination records and though the Network Management Interface (NMI) on the GetTCPListeners and GetConnectionDetail poll requests.

The APPLDATA created by ROPES consists of 40 bytes defined as shown in Figure 13 on page 46.

Installation

The e-mail interface is automatically installed by the primary ROPES installation process. All necessary programs, and table entries are defined as part of the main installation process. If additional Intra-partition destination queue (DCT) entries are required, they can be duplicated from the SMTP entry provided in the ROPESDCT source member. The conditions for needing additional DCT entries will be described later. The definition of DCT entries can be avoided entirely by coding **SIGNAL=TS** in the DCB on the TYPE=CONTROLS macro statement. This causes ROPES to use Temporary Storage instead of Transient Data to signal the printing task. Please refer to the ROPES Administrator's Guide in the chapter on 'Device Characteristics Blocks' for more information on this parameter.

Implementation

The ROPES e-mail interface is implemented through the SMTPCTL macro statement specified in the ROPES DCB (Device Characteristics Block). Modification of the ROPES DCB is described in the **ROPES Administrator’s Guide** under the section **Device Characteristics Blocks**, and the ROPESDEV macro.

ROPES uses a special output device (ROPES printer) to route e-mail. The device characteristics of the printer is determined by the DCB it points to. The ROPES printer definition for an e-mail routing printer, must specify the name of a ROPES DCB generated with the appropriate e-mail configuration options. The DCB name’s suffix must be specified in the ROPES printer definition under the field heading “Device Block Suffix”.

An e-mail supporting DCB must specify **DEVTYPE=SMTP** on the ROPESDEV TYPE=CONTROLS macro statement, and it must also specify the **ROPESDEV TYPE=SMTPCTL** macro statement. The **SMTPCTL** macro statement permits the specification of a number of e-mail related parameters for the DCB. In addition to the SMTPCTL macro statement, a ROPESDEV

TYPE=SOCKETS macro statement must be added after the ROPESDEV TYPE=CONTROLS macro statement. The TYPE=-SOCKETS macro statement establishes certain device level options for components of ROPES that use the CICS Sockets Interface. The E-mail delivery feature of ROPES is one such component. If the TYPE=-SOCKETS macro statement is not coded in the DCB, then the SOCKETS segment will be automatically generated by the assembly process when the DEVTYPE is SMTP. Details of these parameters are described in the ROPES Administrator's Guide.

The new ROPES E-mail File Attach feature is controlled by a number of new parameters added to the TYPE=SMTPCTL macro statement. These new parameters are optional, and need not be coded if file attachment is not required. The new file attach SMTPCTL parameters are named as follows; ATCHTYP, ATCHDMT, ATCHTST, ATCHTSN, ATCHFLN, ATCHFSL, ATCHFSF, ATCHFXT, ATCHSTA, ATCHNT1, ATCHNT2, ATCHNT3, ATCHNT4, and ATCHNT5. These parameters define the format of the file to be attached, the name of the file to be attached, and optional elements of an identifying e-mail body which precedes and accompanies the file attachment. Please consult the ROPES Administrator's Guide in the chapter 'Device Characteristics Blocks' and the section that describes the TYPE=SMTPCTL macro statement for more information about these parameters.

Controlling The E-mail Destination

The e-mail destination for a ROPES report can either be determined by several parameter options in the DCB, or it can be specified directly in the body of the report data. The DCB parameter that specifies where the e-mail address will come from is **SMTPCMD**. A value of "R" means the e-mail address information (sender, receiver, subject) will come from the first few records of the report data. The value "M" means the e-mail address and subject information will come from the **MAILFRM**, **RCPTTO**, **COPYTO**, and **SUBJECT** parameters of the DCB. This will be described in detail by the ROPES Administrator's Guide.

When using the SMTPCMD=R setting, please note that the lines containing the SMTP control information (the lines starting with "(#") must be formatted like any other ROPES print line. The first character of the line must be a carriage control character (blank is fine), and the control text should begin in column 2. If the control information begins in column 1 it will not be recognized.

When specifying the e-mail address information and subject in the body of the report data the following command sequences must be used:

(#SENDHST=sending host name) - Where sending host name is the name of the host as it is specified in the TCP/IP configuration via parameter *HOSTNAME*. The sending host name must not exceed 24 characters in length.

(#MAILFROM=e-mail address) - Where the e-mail address of the sender is specified.

(#RCPTTO=e-mail address) - Where the e-mail address of the recipient is specified.

(#COPYTO=e-mail address) - Where the e-mail address of a secondary recipient is specified. More than one '(#COPYTO' control statement can be generated by the application. The number of '(#COPYTO' statements is limited by the value of the "MAXCPYE" parameter in the Device Characteristics Block (DCB). The maximum value of MAXCPYE is 95. Note that the DCB parameters COPYTO through COPYTO5 are ignored when the '(#COPYTO' statement is generated by the application. The amount of storage allocated to hold the COPYTO entries can be calculated by multiplying the value of MAXCPYE by 81. If the e-mail addresses generated are smaller than 81 bytes, then the actual number of COPYTO e-mail addresses may actually exceed the MAXCPYE parameter.

(#SUBJECT=subject text) - Where subject text is some text that describes the content of the e-mail. The subject text is limited to 80 characters.

The '(#SENDHST' and '(#SUBJECT' commands are *positional commands*. The '(#SENDHST' command must be specified first, and the '(#SUBJECT' command must be specified last. The other commands may occur in any sequence between '(#SENDHST' and '(#SUBJECT'. All the commands are *required* except '(#COPYTO'.

When specifying the e-mail addresses, and subject in the body of the report data, it becomes possible to include multiple e-mails under the same ROPES report id. This is accomplished by simply starting a new set of e-mail (header) commands at the beginning of the next logical e-mail. When ROPES encounters the new e-mail header in the report text, it closes the previous e-mail, and starts the new one. Using this approach lends itself well to sending many e-mails to different e-mail destinations. *Using this method requires changes to your application.*

Controlling the e-mail destination from the ROPES DCB is less efficient when sending e-mail to a large number of e-mail addresses, but does allow reports to be routed to e-mail without application changes. It is also useful when sending a large number of e-mails to the same static e-mail destination.

A third possibility uses the e-mail addresses and subject information from the DCB, but allows the e-mail addresses and subject text to be changed via an exit program. The exit program is named ROPEMRDX. It is called every time a new e-mail header is processed. The user's exit program can change the values of the SENDHST, MAILFROM, RCPTTO, COPYTO, and SUBJECT options. This enables the exit to modify the default values provided by the DCB to suit application requirements.

A sample of the ROPEMRDX exit program can be found in the ROPES SOURCE data set in member ROPEMRDX. This exit sample is provided in ASSEMBLER source, and it uses the communication area macro contained in member ROPEMDXA. This exit program is described in detail in the Administrator's Guide.

Reports delivered as text, HTML or PDF file attachments are also directed to their E-mail destinations using the same controls described in this section. For E-mails sending attached files, destination addresses may be statically defined in the DCB, or dynamically defined in the report data using the E-mail header commands. Report attach options may also be modified by using the exit program ROPEMRDX.

Performance Considerations

The ROPES E-mail Interface uses CICS/Sockets. The CICS/Sockets API is not a command level API, but rather uses OS calls. OS calls cause the CICS region to wait for the duration of the call. This makes it important to ensure these calls are completed quickly. One strategy for ensuring quick turnaround of these calls is to make sure the TCP/IP and SMTP started tasks are running at a dispatching priority that is higher than CICS. You may also want to make sure these started tasks are not contending with other address spaces too often.

If it is imperative that other tasks not be impacted by CICS/Sockets, then you may wish to consider isolating the e-mail routing printers and applications to a dedicated CICS region.

ROPES employs the command level "CHANGE TASK" command to release control after each sockets call so that other tasks can run in CICS. Nevertheless, there may still be some impact on performance if a lot of e-mail is being routed by the ROPES e-mail interface. Careful planning will prevent unexpected performance problems.

Minimizing the number of e-mail printer definitions, and the number of concurrent e-mail tasks in your CICS region will help performance. Therefor, it is recommended that the approach to changing the e-mail address be done through the application using the ROPES e-mail option commands, and that these commands be embedded in the application's report data. This would required the use of the **SMTPCMD=R** option in the DCB.

While no conversion is required for ROPES reports that are attached as text files, HTML and PDF attached reports must be converted prior to sending the E-mail. The conversion process uses temporary storage as an interim storage facility for reading and storing the converted report. One parameter option, ATCHTST, permits the selection of either Auxiliary or Main Temporary Storage for this purpose. For installations that have abundant memory resources, Main Temporary Storage will be the faster and more efficient designation. For installations that have limited memory resources, Auxiliary Temporary Storage will probably be the better selection. Auxiliary Storage is set as the default. However, it is possible to make this selection for every ROPES printer definition/DCB that serves as an E-mail router.

Another factor which should be considered when selecting Auxiliary or Main Temporary Storage concerns the size of the report(s) that will be processed. If very large reports are to be processed as HTML or PDF file attachments, then demands on memory might be excessive if Main Temporary Storage is used. For most of these cases, it is recommended that Auxiliary Temporary Storage be used. It should be noted, however, that a tradeoff will be experienced and large reports may take slightly longer to process. The increase in processing time will largely depend on other factors like the placement of the Temporary Storage data set, and whether other measures have been taken to optimize performance of the Auxiliary Temporary Storage data set.

The amount of Main Temporary Storage used at any given time will also be affected by the number of ROPES printers setup to deliver HTML or PDF attached reports. The amount of storage used can be reduced by assigning more reports to a fewer number of printers rather than defining a large number of printers to serve the same number of reports. This approach will probably reduce the throughput of E-mails being delivered, and so should be considered as part of the overall strategy.

E-mail Application Scenarios

Here are a few examples to provide some recommended approaches for implementing your ROPES/E-mail applications.

1. My application will generate e-mail to only a few e-mail destinations, and my application cannot be changed at this time.

Create one or more DCB modules that specify **SMTPCMD=M**, and specify the static e-mail address(es) via the RCPTTO and MAILFRM DCB parameters. If more than one DCB is required, each DCB must be associated with its own ROPES printer definition. The DCB suffix name is specified in the ROPES printer definition. Each printer definition should also be associated with its own Intra-Partition Transient Data Queue. The **SMTP** definition provided by source member ROPESDCT can be used to clone multiple queues. Using this approach, each printer sends any reports assigned to it to the same e-mail destination without application changes.

2. My batch application will be sending many e-mails to different recipients, and the e-mail addresses of these recipients may change from run to run of the application.

For this scenario, the application must be modified to send ROPES e-mail headers for each logical e-mail. One ROPES DCB and printer definition would be used for this scenario. The DCB would specify the option **SMTPCMD=R**. This option directs ROPES to use the e-mail header commands that are embedded in the body of the report data to determine the e-mail destinations. The e-mail header commands that can be included in the report data were mentioned earlier, but are provided here for convenience; (#SENDHST, (#MAILFROM, (#RCPTTO, (#COPYTO, and (#SUBJECT. The occurrence of a new set of e-mail header commands tells ROPES to send the report data following it to a new e-mail destination. The number of e-mail headers that may be contained in a single report is essentially unlimited.

3. My online application will be generating multiple reports to different recipients, but there will only be one logical e-mail per report generated. The destination of each report can be determined by the name of the report. However, the application cannot be changed at this time.

In this case, a single ROPES DCB/printer can be created. The DCB must use the e-mail parameter **SMTPCMD=M**. This specification will tell ROPES to use the e-mail destinations specified by the DCB parameters **RCPTTO**, **MAILFRM**, and **COPYTO**. Since the e-mail addresses provided by the DCB are static and cannot be changed, the user exit program ROPEMRDX must be used to modify the e-mail addresses.

ROPEMRDX is called at the time the e-mail header information is processed. ROPEMRDX can be modified to suit the user's requirements for processing different e-mail destinations. For example, one method for changing the e-mail destinations might employ an e-mail address lookup table. The ROPEMRDX exit could be modified to use the e-mail address lookup table and dynamically change the RCPTTO, MAILFRM, and COPYTO DCB option values to the appropriate e-mail addresses.

4. A salesman in the field uses various online transactions to generate reports which are all destined to be returned to him. The salesman will pick up the reports as separate e-mails generated to his e-mail address. It would be preferable if application changes were not required.

For this case, a new ROPES printer definition and DCB should be created which statically defines the e-mail address of the salesman. Using this method, any data sent to the ROPES report id (which is associated with the salesman's printer) will be sent to the salesman's e-mail address. It is also possible for multiple ROPES reports to be assigned to the same printer id. When creating the DCB, the **SMTPCMD=M** option must be used on the **SMTPCTL** DCB macro statement. This directs ROPES to obtain the e-mail address information from the DCB.

5. My online application generates many e-mails to many customers. Different transactions are used to generate varying information which must be sent to the customer directly by e-mail. Sometimes information must be collected and then sent together to the same e-mail destination.

First it is assumed that the necessary configuration changes have been implemented to permit TCP/IP, and the mail server (SMTP) to send mail to the Internet. Next, the application would have to be designed and implemented to include the ROPES e-mail header commands in the report data. These e-mail header commands specify the e-mail destination, sender, and subject text that is to be associated with the report data that follows. Multiple sets of e-mail header commands can be generated and included by the application into the same ROPES report. The application can also append more report data to the same report id, and continue to use the same e-mail destination.

Since the e-mail destination is controlled by the application, the DCB **SMTPCTL** macro statement must specify **SMTPCMD=R**. This directs ROPES to obtain the e-mail destination from the report data (not the DCB). The manner in which the data is collected and transmitted is purely a matter of application design. The manner in which e-mail addresses are obtained and used is also a matter of application design.

This concludes the discussion on the ROPES E-mail Interface. Please consult the ROPES Administrator's Guide for detailed information on the coding of the **SMTPCTL** macro statement and all of its parameters.

6. Any of the application scenarios mentioned above could be used to send a ROPES report as an attached HTML or PDF file. To accomplish this, a new set of DCB parameters have been added to the DCB TYPE=SMTPCTL macro statement. In the DCB code sample below (see **Figure 11** on page 45), the e-mail destination is controlled by the e-mail header commands located in the body of the report data. Using this method, it is possible to send multiple logical reports and e-mails from the same ROPES report id. In this scenario however, the reports will be sent as attached HTML or PDF files.

In the above macro statements, the SENDHST, MAILFRM, RCPTTO and SUBJECT parameters are ignored. The values for these parameters will be obtained from the report e-mail header commands which are contained in the body of the report data (i.e. "(#SENDHST= ", "(#RCPTTO= ", "(#MAILFROM=", etc.). The SMTPCMD=R parameter option indicates that the e-mail header options (sending host, mail from, recipient, subject, etc.) will come from the report data.

The report will be sent as an attached HTML or PDF file by control of the parameters that all begin with the prefix "ATCH". The "ATCH" parameters are optional, but once the ATCHTYP parameter is specified, then all other required "ATCH" parameters must be specified. The ATCHTYP parameter specifies that the attached report file will either be in report/text-plain format, HTML format or PDF format. The ATCHTST parameter is required and specifies that the type of temporary storage that will be used is "Main" Temporary Storage. This is a requirement for converting the report into HTML or PDF format. The ATCHTSN parameter specifies the name of the Temporary Storage Queue to be used. This is required when ATCHTST/ATCHTYP has been specified.

The next set of ATCH parameters pertains to naming the report file to be attached to the e-mail. The full file name is built from a combination of ATCH parameters. The first parameter is ATCHFLN which specifies the file name prefix. The value for this prefix can be any length up to 50 characters. Care should be taken not to exceed the maximum file name size for the platform that serves as the e-mail client. The ATCHFSF parameter is set to "Y" in this example, which means that the file suffix has been selected. The file suffix is just a time stamp which serves to give the file name a unique name. The time stamp is made up of the date and time in the format YYMMDDHHMMSS. Any number of digits from the time stamp may be selected by the ATCHFSL parameter, and the selection is made starting from the right. In our example, the value of ATCHFSL is "12" which means the entire time stamp value is appended to the file prefix to build the first node of the file name. The result is "ROPRPTyymmddhhmmss" where "yymmddhhmmss" is the current date and time obtained from the system. The last parameter pertaining to the file name is ATCHFXT which specifies the file extension. In this example, the file extension is set to "PDF". Note that the file extension should match the type of file that is being attached to the e-mail.

The remaining parameters control what will be present in the preamble identifying e-mail body. When ROPES sends the file attachment, it sends it as a "multipart/mixed" type of e-mail. This provides for an identifying e-mail body which occurs at the top and indicates where the e-mail came from and how it was delivered. This identifying e-mail or preamble e-mail is received by the recipient like any normal text e-mail and serves as a kind of notification about the source of the e-mail. The first parameter, ATCHSTA, specifies that status or identifying information containing the date/time of delivery, the ROPES printer id, and the report id should be included in the preamble e-mail.

At least one message line will appear in the preamble e-mail. That message line indicates the source of the E-mail. By default this message line will contain "This E-mail delivered by ROPES Version 8.0". The ATCHDMT parameter may be used to modify the content of the delivery message if it is not suitable for the customer's needs. The ATCHNT1 through ATCHNT5 parameters specify optional message lines which can be included in the preamble e-mail. Their content is strictly up to the designer, and may be different for each DCB. Using the parameters specified in our example, the preamble e-mail would contain the text shown in **Figure 12** on page 45.

```
          ROPESDEV TYPE=SMTPCTL,                                          X
                TCPIPNM=TCPIP,                                            X
                SMTPCMD=R,                                                X
                USELHST=L,                                                X
                MAILLOG=Y,                                                X
                MAILLGQ=CSMT,                                             X
                SENDHST=YOURHOST,                                         X
                MAILFRM=MYNAME@DOMAIN.COM,                                X
                RCPTTO=MYNAME@DOMAIN.COM,                                 X
                MAXCPYE=50,                                               X
                SUBJECT='THIS ROPES REPORT DELIVERED BY E-MAIL.',         X
                ATCHTYP=PDF,                                              X
                ATCHTST=M,                                                X
                ATCHTSN=RPDFTEST,                                         X
                ATCHFLN=ROPRPT,                                           X
                ATCHFSF=Y,                                                X
                ATCHFSL=12,                                               X
                ATCHFXT=PDF,                                              X
                ATCHSTA=Y,                                                X
                ATCHDMT='This E-mail delivered by ROPES Version 14.0.',   X
                ATCHNT1='********************************************',   X
                ATCHNT2='* This report was delivered by ROPES V14.0 *',   X
                ATCHNT3='* as a PDF attachment using a file suffix, *',   X
                ATCHNT4='* and the date/time statistics options.    *',   X
                ATCHNT5='********************************************'
```

Figure 11 E-mail DCB Example

```
This E-mail delivered by ROPES Version 14.0.
Date/Time Sent: Mar 13, 2008 10:34:27
Printer Id: SMTP
Report Id: ROWE02

********************************************
* This report was delivered by ROPES V14.0 *
* as a PDF attachment using a file suffix, *
* and the date/time statistics options.    *
********************************************
```

Figure 12 E-mail preamble for attachments

```
----+----1----+----2----+----3----+----4
ROPES mmmm prtr devb strm reportid yyddd
|     |    |    |    |    |        |
|     |    |    |    |    |        Task start date
|     |    |    |    |    Report Name
|     |    |    |    Data stream type
|     |    |    Device Block Suffix
|     |    Printer id
|     Module id
Literal
```

For example:
`"ROPES LPRP DELL LPRD PPDS COPYGD01 07002"`

Module Id:

LPRP	- from ROPELPRP	- LPR printer output
MRDP	- from ROPEMRDP	- Mail (SMTP) printer output
SOCP	- from ROPESOCP	- Direct socket output

Printer Id:
the four character ROPE PCR name

Device Block Suffix:
the last four bytes of the device block module name

Data Stream Type:
the four character DSTYPE value from the Device Block

Report Id:
Current report name

Task Start Date:
Task start date in Julian yyddd notation.

Figure 13 APPLDATA

The TCP/IP LPR Remote Print Feature

Introduction

The ROPES TCP/IP LPR Remote Print Feature provides an interface to remote Line Printer Daemons (LPDs) via the internet or intranet to print ROPES reports. Limitations on the location of the LPD servers is determined by network configuration and security.

The ROPES LPR Remote Print Feature requires TCP/IP for MVS, CICS Sockets, and CICS version 3.3 or higher.

APPLDATA Generation

Beginning with ROPES Version 14, the TCP/IP based data transmission functions, including this function, now generate the APPLDATA information made available in z/OS Communications Server V1R9 (and by PTF, rolled back to V1R7 and V1R8).

This data allows you to have a better understanding of the purpose of the ROPES active sockets connections. The associated application data is made available on Netstat ALL/-A, ALLConn/-a and COnn/-c reports, in SMF 119 TCP Connection Termination records and though the Network Management Interface (NMI) on the GetTCPListeners and GetConnectionDetail poll requests.

The APPLDATA created by ROPES consists of 40 bytes defined as shown in **Figure 13** on page 46.

Installation

The ROPES LPR Remote Print Feature is automatically installed by the primary ROPES installation process. All necessary programs, and table entries are defined as part of the main installation process. If additional Intra-Partition destination queue (DCT) entries are required, they can be duplicated from the LPRP entry provided in the ROPESDCT source member. Additional DCT entries are required for each LPR printer defined to ROPES unless the printer task (R#O1) signal method has been changed to use Temporary Storage. This is accomplished by setting the TYPE=CONTROLS parameter SIGNAL=TD to SIGNAL=TS in the ROPES Device Characteristics Block (DCB). If the signal method is changed to use Temporary Storage, then no DCT entries are required. This is described in more detail in the Administrators Guide under the chapter dealing with Device Characteristics Blocks.

Implementation

The ROPES TCP/IP LPR Remote Print Feature is implemented by specifying a Device Characteristics Block in the ROPES printer definition that specifies a device type of "LPRD". The DEVTYPE parameter of the ROPESDEV TYPE=CONTROLS macro statement must specify "LPRD". In addition to modifying the DEVTYPE parameter, a ROPESDEV TYPE=SOCKETS macro statement must be added after the ROPESDEV TYPE=CONTROLS macro statement. The TYPE=SOCKETS macro statement establishes certain device level options for components of ROPES that use the CICS Sockets Interface. Options provided by the TYPE=SOCKETS macro statement are SOKTMEO, SOKRTRY, and SOKDLAY. These parameter options provide a READ/WRITE socket timeout value, a failed request retry value, and a retry delay option expressed in seconds. The TCP/IP LPR Remote Printer Feature is one such component. If the TYPE=SOCKETS macro statement is not coded in the DCB, then the SOCKETS segment will be automatically generated by the assembly process when the DEVTYPE is LPRD. Modification of the ROPES DCB (Device Characteristics Block) is described in the ROPES Administrator's Guide under the section Device Characteristics Blocks, and the ROPESDEV macro.

ROPES uses a special output device (ROPES printer) to route output to an LPD server. The device characteristics of the printer is determined by the DCB it points to. The ROPES printer definition for an LPR printer, must specify the name of a ROPES DCB generated with the appropriate DEVTYPE value (LPRD) to support LPR. The DCB name suffix must be specified in the ROPES printer definition under the field heading "Device Block Suffix". This can be accomplished by executing the ROPES Maintenance transaction, ROMT, to update or add a printer definition.

In addition, a CICS Intra-partition destination queue must be defined which uses the same name as the ROPES printer defined to send output using LPR. The Intra-partition queue definition must be setup to initiate the printer driver transaction "R#O1" using a trigger level of 1. A sample Intra-partition destination can be found in source member ROPESDCT called LPRP.

If preferred, adding a DCT definition can be avoided entirely by changing the method in which the printer task is signaled. Normally, the printer task is signaled using transient data. It can be signaled using Temporary Storage. This is accomplished by setting the TYPE=CONTROLS macro statement parameter SIGNAL=TD to SIGNAL=TS. More information is available on this in the Administrator's Guide under the section on Device Characteristics Blocks.

An online transaction is also provided to setup the LPR options that will be used to send the output to the required LPD server. The transaction can be run on a CICS terminal by entering "ROLI" at a clear screen. Transaction ROLI will bring up an index of LPR printer options. One ore more entries can exist for a printer. There can be a default entry for the printer which all reports use, or there can be individual entries for each report assigned to a given printer. The ROLI transaction is described in more detail in the ROPES Administrator's Guide.

Controlling the LPD Server Destination

The LPD server is determined by the LPR options record defined by the ROLI transaction. The remote LPD server is defined using the remote host/server name value. In addition, the port on which the LPD server listens is defined to be port number 515. The LPR options does permit this port number to be changed, but it is not expected that any other port number will be valid unless certain LPD configuration items have changed. Please consult your LPD server documentation for more information on the LPD "CAP" configuration file.

Components Of The ROPES LPR Feature

The LPR Remote Print feature of ROPES uses CICS Sockets and the protocol described by RFC1179 to interface with a Line Printer Daemon server. The Line Printer Daemon receives the print job through a series of commands/file transfers, and then places the job on a print queue. The print queue name is predefined through the configuration of the LPD server. Limitations on the location of the LPD server is determined through the network configuration. The LPD server may be accessed over the Internet, or may be localized to an intra-net. In some cases, the LPD server may reside in a physical printer capable of receiving LPR commands.

Once the ROPES report has been transferred to the LPD, control over the report is determined by the operator(s) managing the LPD server. If the LPD server has already started the print queue, then the output will be printed. If the operator has stopped the print queue, then the output will remain in the queue until the operator releases the print queue. ROPES does not exercise any control over the output once it has been transmitted to the LPD server. At this time, some additional control can be obtained by using the LPR, LPC, LPQ, and LPRM commands provided with OS/390 eNetwork Communication Server.

Security

The LPR Remote Print feature of ROPES does not provide any authentication or job data transfer security other than that provided by CICS and RACF facilities. The LPR protocol used by ROPES does not support encryption or secured sockets layer protection at this time.

LPR Device Controls And Usage Notes

Many ROPES Device Characteristic Block (DCB) parameter option settings are possible with the new ROPES LPR feature. Many ROPES Forms Control Block (FCB) parameter option settings are also possible. For example, the ROPES FCB "TYPE=CONTROLS" CPI, LPI, and ORIENT parameters can be coded in the user mode to change the font size, orientation, and LINE density for PCL type printers. However, some changes to the FCB/DCB can cause failures in the LPR process. One such example is the LINEFMT parameter of the ROPES DCB. The LINEFMT parameter must be set to "V" (variable) in order for LPR processing to work correctly. Changing the LINEFMT parameter to "F" (fixed) will cause the output driver module not to send the data to the LPR program correctly, and this will result in an abend.

The LPR Printer/Report Options parameters must also be considered when changing the ROPES FCB/DCB parameters. For example, using certain control codes may not be compatible with the "f" (formatted print) option defined by the LPR Options screen. It may be necessary to use the "l" option instead which allows all control characters to be passed on to the printer. For these reasons it is recommended that all LPR Option and FCB/DCB changes for LPR printers/reports be tested before implementation into your production environment. This will prevent unexpected failures, and provide a higher degree of reliability for your IP printers.

It is recommended that you use the ROPES distributed FCB/DCB members as a starting point for your LPR implementation. These members are located in the source distribution library, and are named RO$DLPRD, RO$F, and RO$FLAZR.

LPR Transmission Logging And Problem Diagnosis

In most cases it will be sufficient to examine the error messages produced in the ROPES error message log (usually extra-partition queue CSMT), yet in some cases additional information may be required to determine the cause of a problem. The LPR Options definition screen has an option called "Log Xmits To TDQ". If this option is set to "Y" (yes), all transmissions sent and received will be logged to the "Log TD Queue" destination (also specified on the same LPR Options Definition screen). Since each LPR option definition specifies the options to be used at the printer or report level, it is possible to turn logging on and off for a specific printer or report. All logging messages are prefixed with a constant "SOCKDATI-" (input) and "SOCKDATO-" (output) to indicate the direction of the transmission. The data produced by the logging option can often be used in conjunction with error messages to resolve the problem.

Sockets Options For LPR Printers

The ROPES Device Characteristics Block (DCB) must specify the TYPE=SOCKETS macro statement for LPRD devices. The SOKTMEO parameter on the TYPE=SOCKETS macro statement prevents blocked READ/WRITE operations by specifying a timeout value for the socket read/write call. This parameter option is required if you wish to prevent transactions from stalling when LPR Daemons or SMTP servers do not respond.

Two new parameters have been added to the TYPE=SOCKETS macro statement. They are the SOKRTRY and SOKDLAY parameters. These parameters can be used for Remote Print Feature or for DEVTYPE=LPRD printer definitions (see the DCB ROPESDEV TYPE=CONTROLS macro statement in the Administrator's Guide).

The SOKRTRY parameter specifies the number of times the print operation should be retried in the event of a connection loss with the Line Printer Daemon or abend due to some other communication failure. The maximum number of retries that can be specified is 12. The recommended number of retries is 3, and zero is the default if no value is specified. If a positive retry value is specified and a failure occurs, one or more ROPES00031 messages will appear in the ROPES message log to indicate the retries being performed and to show the number of retries remaining. If a delay is required after the retry to permit automatic recovery processes to run, then the SOKDLAY parameter should be used. The SOKDLAY parameter works in conjunction with the SOKRTRY parameter. The SOKDLAY parameter is specified only when there is a retry value coded for SOKRTRY. The SOKDLAY parameter can be supplied by specifying the delay time in seconds. The SOKDLAY parameter will accept any value in the range of 0 to 61439. Please consult the Administrator's Guide in the chapter about Device Characteristics Blocks for more information about the TYPE=SOCKETS macro statement.

The TCP/IP Direct Socket Remote Print Feature

Introduction

The ROPES TCP/IP Direct Socket Remote Print Feature provides an interface to remote IP attached printers via the internet or intranet to print ROPES reports. Limitations on the location of the printers is determined by network configuration and security.

The ROPES Direct Socket Remote Print Feature requires TCP/IP for MVS, CICS Sockets, and CICS version 3.3 or higher.

APPLDATA Generation

Beginning with ROPES Version 14, the TCP/IP based data transmission functions, including this function, now generate the APPLDATA information made available in z/OS Communications Server V1R9 (and by PTF, rolled back to V1R7 and V1R8).

This data allows you to have a better understanding of the purpose of the ROPES active sockets connections. The associated application data is made available on Netstat ALL/-A, ALLConn/-a and COnn/-c reports, in SMF 119 TCP Connection Termination records and though the Network Management Interface (NMI) on the GetTCPListeners and GetConnectionDetail poll requests.

The APPLDATA created by ROPES consists of 40 bytes defined as shown in **Figure 13** on page 46.

Installation

The ROPES Direct Socket Remote Print Feature is automatically installed by the primary ROPES installation process. All necessary programs, and table entries are defined as part of the main installation process. If additional Intra-Partition destination queue (DCT) entries are required, they can be replicated from the LPRP entry provided in the ROPESDCT source member. Additional DCT entries are required for each Direct Socket printer defined to ROPES unless the printer task (R#O1) signal method has been changed to use Temporary Storage. This is accomplished by setting the TYPE=CONTROLS parameter SIGNAL=TD to SIGNAL=TS in the ROPES Device Characteristics Block (DCB). If the signal method is changed to use Temporary Storage, then no DCT entries are required. This is described in more detail in the Administrators Guide under the chapter dealing with Device Characteristics Blocks.

Implementation

The ROPES TCP/IP Direct Socket Remote Print Feature is implemented by specifying a Device Characteristics Block in the ROPES printer definition that specifies a device type of "SOKT". The DEVTYPE parameter of the ROPESDEV TYPE=CONTROLS macro statement must specify "SOKT". In addition to modifying the DEVTYPE parameter, a ROPESDEV TYPE=SOCKETS macro statement must be added after the ROPESDEV TYPE=CONTROLS macro statement. The TYPE=SOCKETS macro statement establishes certain device level options for components of ROPES that use the CICS Sockets Interface. Options provided by the TYPE=SOCKETS macro statement are SOKTMEO, SOKRTRY, and SOKDLAY. These parameter options provide a READ/WRITE socket timeout value, a failed request retry value, and a retry delay option expressed in seconds. The TCP/IP Direct Socket Remote Printer Feature is one such component. If the TYPE=SOCKETS macro statement is not coded in the DCB, then the SOCKETS segment will be automatically generated by the assembly process when the DEVTYPE is SOKT. Modification of the ROPES DCB (Device Characteristics Block) is described in the ROPES Administrator's Guide under the section Device Characteristics Blocks, and the ROPESDEV macro.

ROPES uses a special output device (ROPES printer) to route output to a Direct Socket printer. The device characteristics of the printer is determined by the DCB it points to. The ROPES printer definition for a Direct Socket printer must specify the name of a ROPES DCB generated with the appropriate DEVTYPE value (SOKT) to support Direct Socket printing. The DCB name suffix must be specified in the ROPES printer definition under the field heading "Device Block Suffix". This can be accomplished by executing the ROPES Maintenance transaction, ROMT, to update or add a printer definition.

In addition, a CICS Intra-partition destination queue must be defined which uses the same name as the ROPES printer defined to send output using Direct Socket printing. The Intra-partition queue definition must be setup to initiate the printer driver transaction "R#O1" using a trigger level of 1. A sample Intra-partition destination can be found in source member ROPESDCT called LPRP.

If preferred, adding a DCT definition can be avoided entirely by changing the method in which the printer task is signaled. Normally, the printer task is signaled using transient data. It can be signaled using Temporary Storage. This is accomplished by setting the TYPE=CONTROLS macro statement parameter SIGNAL=TD to SIGNAL=TS. More information is available on this in the Administrator's Guide under the section on Device Characteristics Blocks.

An online transaction is also provided to setup the Direct Socket options that will be used to send the output to the IP attached printer. The transaction can be run on a CICS terminal by entering "ROLI" at a clear screen. Transaction ROLI will bring up an index of LPR printer options. This facility is also used for the Direct Socket printers. One ore more entries can exist for a printer. There can be a default entry for the printer which all reports use, or there can be individual entries for each report assigned to a given printer. The ROLI transaction is described in more detail in the ROPES Administrator's Guide.

Controlling the Direct Socket Printer Destination

The IP printer destination is determined by the LPR options record defined by the ROLI transaction. The remote printer is defined using the remote host/server name value. In addition, the port on which the printer listens is defined to be port number 9100. The LPR options does permit this port number to be changed, but it is not expected that any other port number will be valid unless certain printer configuration items have changed. Please consult your printer documentation for more information on the printer port configuration.

Security

The Direct Socket Remote Print feature of ROPES does not provide any authentication or job data transfer security other than that provided by CICS and RACF facilities. The protocol used by ROPES does not support encryption or secured sockets layer protection at this time.

LPR Device Controls And Usage Notes

Many ROPES Device Characteristic Block (DCB) parameter option settings are possible with the new ROPES Direct Socket printing feature. Many ROPES Forms Control Block (FCB) parameter option settings are also possible. For example, the ROPES FCB "TYPE=CONTROLS" CPI, LPI, and ORIENT parameters can be coded in the user mode to change the font size, orientation, and LINE density for PCL type printers. However, some changes to the FCB/DCB can cause failures in the printing process. One such example is the LINEFMT parameter of the ROPES DCB. The LINEFMT parameter must be set to "V" (variable) in order for Direct Socket printing processing to work correctly. Changing the LINEFMT parameter to "F" (fixed) will cause the output driver module not to send the data to the printer correctly, and this will result in an abend.

It is recommended that you use the ROPES distributed FCB/DCB members as a starting point for your Direct Socket printing implementation. These members are located in the source distribution library, and are named RO$DSOKT, RO$F, and RO$FLAZR.

Direct Socket Printing Transmission Logging And Problem Diagnosis

In most cases it will be sufficient to examine the error messages produced in the ROPES error message log (usually extra-partition queue CSMT), yet in some cases additional information may be required to determine the cause of a problem. The LPR Options definition screen has an option called "Log Xmits To TDQ". If this option is set to "Y" (yes), all transmissions sent and received will be logged to the "Log TD Queue" destination (also specified on the same LPR Options Definition screen). Since each LPR option definition specifies the options to be used at the printer or report level, it is possible to turn logging on and off for a specific printer or report. All logging messages are prefixed with a constant

"SOCKDATI-" (input) and "SOCKDATO-" (output) to indicate the direction of the transmission. The data produced by the logging option can often be used in conjunction with error messages to resolve the problem.

Sockets Options For Direct Socket Printers

The ROPES Device Characteristics Block (DCB) must specify the TYPE=SOCKETS macro statement for SOKT devices. The SOKTMEO parameter on the TYPE=SOCKETS macro statement prevents blocked READ/WRITE operations by specifying a timeout value for the socket read/write call. This parameter option is required if you wish to prevent transactions from stalling when remote IP printers do not respond.

Two new parameters have been added to the TYPE=SOCKETS macro statement. They are the SOKRTRY and SOKDLAY parameters. These parameters can be used for Remote Print Feature or for DEVTYPE=SOKT printer definitions (see the DCB ROPESDEV TYPE=CONTROLS macro statement in the Administrator's Guide).

The SOKRTRY parameter specifies the number of times the print operation should be retried in the event of a connection loss with the Line Printer Daemon or abend due to some other communication failure. The maximum number of retries that can be specified is 12. The recommended number of retries is 3, and zero is the default if no value is specified. If a positive retry value is specified and a failure occurs, one or more ROPES00031 messages will appear in the ROPES message log to indicate the retries being performed and to show the number of retries remaining. If a delay is required after the retry to permit automatic recovery processes to run, then the SOKDLAY parameter should be used. The SOKDLAY parameter works in conjunction with the SOKRTRY parameter. The SOKDLAY parameter is specified only when there is a retry value coded for SOKRTRY. The SOKDLAY parameter can be supplied by specifying the delay time in seconds. The SOKDLAY parameter will accept any value in the range of 0 to 61439. Please consult the Administrator's Guide in the chapter about Device Characteristics Blocks for more information about the TYPE=SOCKETS macro statement.

Extended Device Controls Feature

Introduction

The Extended Device Controls Feature permits large streams of Printer Control Language device codes to be sent to a ROPES printer. Previously, ROPES had a limited capability to send PCL Code strings, and the code sequences could not exceed 254 bytes in length. PCL codes can be sent to ROPES devices using the "X" carriage control character in the first position of the print line. This tells ROPES that the data on the rest of the line is not to be translated, and can be used to send printer control language codes. Another facility for sending special printer codes involves using the ROPES escape character. All data following the escape character up to the next escape character, or for the remainder of the line is sent ASIS.

Using the ROPES Extended Device Controls Feature, the PCL code sequences can be unlimited in size. Any number of PCL code sequences may be established during device initialization and device reset or termination. Large sequences can be broken down into smaller sequences and labeled and used accordingly. Other compatible devices can reuse PCL sequences.

PCL Code sequences are stored in tables. In a future release of this feature, PCL codes may also be stored on one or more "PCL" files. The PCL table is a simple module which contains a fullword pointer to the beginning of the PCL code, a fullword containing the size of the PCL code, a descriptive area which can be variable in size, and the PCL code area which can be virtually unlimited in size. The PCL code table must be generated manually. However, any program capable of producing PCL code, which often includes word processing type programs, may generate the PCL code data itself. The PCL code will then require some reformatting and possibly filtering to put it in PCL Code table form.

Individual PCL code tables are identified and loaded based on two new macro statements in the ROPES Device Characteristics Block (DCB). The two new macro statements are TYPE=DEVINIT and TYPE=DEVEXIT. Each DEVINIT and DEVEXIT statement identifies a PCL code table, which must be sent to the printer. Please see the ROPES Administrator's Guide under the section about customizing the ROPES DCB for more information on these new macro statements.

The Extended Device Controls Feature is only available for TCP/IP LPR Remote printers at the present time. In the next release, this feature will be made available for most if not all ROPES supported printers.

The Extended Device Controls Feature requires the following software: CICS 3.3 or higher, OS/390, ROPES Version 7.0 and all ROPES LPR feature maintenance, TCP/IP for MVS, and CICS Sockets.

ROPES version 8.0 PTF level 1 supports a new enhancement to the Extended Device Controls Feature. The new feature is called the EDC Override Feature (Extended Device Controls Override Feature). This feature permits one part of a PCL Overlay table/module to be dynamically overridden with a parameter/option value from another part of the ROPES system. Five different types of override are possible with the EDC Override feature. One of the override types permits the ROPES printer or report definition copies value to be used as an overriding value for a PCL code sequence that sets the number of printer copies. Other override types enable the Characters Per Inch, Lines Per Inch, and Orientation settings found in the ROPES FCB to override appropriate PCL code sequences.

Several new DEVINIT/DEVEXIT macro statement parameters have been added to the ROPES DCB by the EDC Override Feature. These parameters are; DVCOVRD, DVCOVOF, DVCOVLN, DVCOVDF, and DVCRULE. These parameters are described in detail in the ROPES Administrator's Guide under the chapter 'Customizing ROPES For Your Installation' and in section 'Device Characteristics Blocks'. However, this chapter will provide an overview and guide to using the new parameters of the EDC Override Feature.

Installation

The Extended Device Controls Feature does not require separate installation. Some customization is required for the DCB(s), FCB(s), and PCL Code tables. CICS program definitions will also be required for the new DCB(s), FCB(s) and PCL Code tables.

Implementation

The Extended Device Controls Feature is implemented through the ROPES Device Characteristics Block. The ROPES DCB defines the characteristics of the printer that it is associated with. The Extended Device Controls Feature is also implemented through the ROPES Forms Control Block. The ROPES FCB defines the characteristics of a ROPES report.

The ROPES DCB defines which PCL Code tables are to be used for the associated printer. Each PCL Code table is defined by a separate TYPE=DEVINIT or TYPE=DEVEXIT macro statement. Please see the Administrator's Guide under the section on customization of the ROPES DCB for more information on these macro statements. The DEVINIT statement establishes codes sequences to be sent for device initialization, and the DEVEXIT statement defines codes sequences to be sent for device reset or termination.

The name of the PCL Code table is specified on the DVCNAME parameter of the DEVINIT/DEVEXIT statements. The DVCSRCE parameter specifies that the PCL code sequences are contained in a PCL Code table or a file. The DVCTYPE parameter specifies a PCL Code type name to be associated with the table. The DVCTYPE name is 20 characters in length and may contain any descriptive name. However, it should contain a name that corresponds with the type of code sequence being sent to the printer. For example, if the code sequence is a type of form overlay, then it should be labeled “OVERLAY” or “OVERLAY1”, etc. If the code sequence changes the Characters Per Inch setting in the printer, then you should use a DVCTYPE value of CPI or CPI1. It is recommended that a naming standard be established so those duplicate DVCTYPE names are not created for different printer models.

If a ROPES DCB contains one or more enabled DEVINIT or DEVEXIT statements and there are no DEVINIT or DEVEXIT statements in the FCB associated with the report, then all PCL Code tables will be sent to the printer. If a user supplied FCB is associated with a report, then DEVINIT and DEVEXIT statements may be added to the FCB to suppress or allow sending specific PCL Code tables. The FCB DEVINIT and DEVEXIT macro statements also contains a DVCTYPE parameter, which must contain the same name, used in the DCB's DEVINIT or DEVEXIT statement. So if the DVCTYPE value in the DCB is “OVERLAY1”, then to suppress this PCL Code table the FCB DVCTYPE value must also be set to “OVERLAY1”, and the DVCUSEF value must be set to “N” (No).

Sample DCB Statements:

```
ROPESDEV  TYPE=DEVINIT,
      DVCNAME=PCLMOD1,
      DVCSRCE=M,
      DVCTYPE=OVERLAY1,
      DVCBUFS=1024,
      DVCRTRN=N,
      DVCSTAT=Y
ROPESDEV TYPE=DEVINIT,
      DVCNAME=PCLMOD2,
      DVCSRCE=M,
      DVCTYPE=CPILPI,
      DVCBUFS=1024,
      DVCRTRN=N,
      DVCSTAT=Y
```

Sample FCB Statements:

```
ROPESFRM TYPE=DEVINIT,DVCTYPE=OVERLAY1,DVCUSEF=Y
ROPESFRM TYPE=DEVINIT,DVCTYPE=CPILPI,DVCUSEF=N
```

In the above example, the FCB overrides the use of the “CPILPI” code sequence in the DCB, and forces only the “OVERLAY1” code sequence in PCL Code table “PCLMOD1” to be sent. Samples of the FCB and DCB using the Extended Device Controls feature can be found in members RO$FESSX and RO$DLPRO in the ROPES distribution source library.

A DEVINIT/DEVEXIT macro statement may have an initial setting of enabled or disabled by using the DVCSTAT parameter. PCL codes for a disabled definition are not sent to the printer, and PCL codes for an enabled definition will be sent to the printer unless overridden by the FCB. Using the DVCSTAT parameter makes it possible to create a DEVINIT/DEVEXIT statement that will be initially disabled so that it can be enabled or activated by the FCB DVCUSEF parameter. If the DVCSTAT parameter is not coded, the default value is 'Y' or enabled. The FCB DVCUSEF parameter always overrides the setting of the DVCSTAT parameter.

PCL Code Tables

Before a PCL Code table can be used, it must be created, assembled and link edited, and then defined to CICS as an assembly language module. A sample PCL Code Table can be found in the ROPES distribution source library in member name BOX.

This member is provided as a sample for examination, and is not really intended for use. However, for purposes of testing the Extended Device Controls feature, it can be used as a test overlay.

The format of a PCL Code table can be seen in the sample code below:

```
BOX   CSECT
BOX   AMODE ANY
BOX   RMODE ANY
     DC    AL4(BEGIN) *POINTER TO BEGINNING OF PCL CODE
      DC    AL4(ENDING-BEGIN)  *LENGTH OF PCL CODE DATA
      DC    CL8'BOX     ' *NAME OF PCL CODE TABLE
      DC    C'&SYSDATE'   *DATE OF MODULE ASSEMBLY
BEGIN EQU     *
      DC    X'1B451B266C32613068316F307331581B'
      DC    X'2666317930783053 1B2A7230461B391B'
      *
      *          (PCL code data continues)
      *
      DC    X'1B266C3545'
ENDING  EQU     *
      END     BEGIN
```

The first 4 byte (fullword) field of the module points to the beginning of the PCL code data. The second 4 byte (fullword) field contains the length of all the PCL code data. The next two fields are descriptive and provide the module name and assembly date of the PCL code table. The descriptive fields can be variable in size, so more descriptive information or fields may be added before label “BEGIN” if so desired without affecting the functionality of the module.

Generation and creation of PCL Code tables is a manual process that must be performed by the user. In some cases, word processing tools may be used to generate PCL Code data, which can then be reformatted into the assembler module format described above. In other cases, the PCL Code data may have to be created by an entirely manual process.

Future enhancements to this feature will include a utility program for generating the PCL Code table from raw PCL code data, the generation of PCL Code records to a PCL Code file, and an online maintenance facility for updating and modifying PCL Code files.

PCL Code File Tables

Support for PCL Code File Tables will be announced in a future release of ROPES when it is available.

EDC Override Feature

The Extended Device Controls Override Feature permits a single specific PCL command sequence in a PCL Overlay module to be dynamically overridden with an existing parameter value from ROPES. Only one override is possible per DCB DEVINIT or DEVEXIT macro statement.

The Extended Device Controls Feature permits PCL code sequences, or PCL forms overlays to be sent to a ROPES printer prior to sending the report data. This feature is useful because it sets within the printer certain options, or forms that will be applied to each page of the report as it is printed. However, the PCL code sequences contained within a PCL overlay module cannot be changed to meet the requirements of a different report unless a different PCL overlay module is used. For example, if a PCL overlay module contains a PCL code sequence that sets the characters per inch printer setting to 12, then all reports using that PCL overlay will be printed at 12 CPI. The Extended Device Codes Override Feature allows the PCL code within a PCL overlay module to be changed at the report level. Using the EDC Override Feature, it would be possible for multiple reports using the same PCL overlay module to have different CPI settings.

Since the EDC Override Feature also supports changing the LPI, Orientation, and Copies printer settings, it would be possible to change these at the report level also. However, all of these settings cannot be changed for the same PCL Overlay module at the same time. Since the EDC Override Feature allows only one setting to be changed, the PCL Overlay module must be broken up into smaller PCL code sequences that contain only the PCL code necessary to specify one of the printer settings. For example, a PCL overlay module may contain the PCL code sequences for CPI, LPI, and Orientation. This PCL overlay module would have to be broken into smaller PCL code sequences each specifying a single PCL parameter like CPI, or LPI. In this example, the original module would be broken into 4 PCL overlay modules each containing the PCL code necessary to specify one printer setting. Using this method, all override types can be implemented for the same PCL code stream.

Since the override values are obtained from standard ROPES controls like the Forms Control Block or the copies setting from the Report or Printer definition records, it is possible to change the override values using familiar interfaces and options. Some override data values can be changed immediately using an online transaction (ROMT) while others can be changed using a ROPES control table called the Forms Control Block (FCB).

Override Type Parameter

Five possible types of override can be selected. The possible types of override are: Printer Definition Copies; Report Definition Copies; FCB Characters Per Inch; FCB Lines Per Inch and FCB Orientation. The override type is specified by the DVCOVRD para-meter on the ROPESDEV TYPE=DEVINIT/-DEVEXIT macro statement. The actual parameter values for each type are given below:

DVCOVRD **Override Type**	**Description**
PrinterCopies or PrtCopies	Copies value from the Printer Control Record.
ReportCopies or RptCopies	Copies value from the Report Information Block Record.
FCBCharactersPerInch or FCBCPI	Characters Per Inch value from the ROPES FCB.
FCBLinesPerInch or FCBLPI	Lines Per Inch value from the ROPES FCB.
FCBOrientation or FCBOrient	Orientation value from the ROPES FCB.

The override type values are case sensitive and must be specified by the DVCOVRD parameter exactly as shown in the table above. FCB values are obtained from the FCB TYPE=CONTROL macro statement. Values specified by the FCB TYPE=CONTROL macro statement must be specified using CTL-MODE=NORMAL and not CTLMODE=USER. If the values are specified using CTLMODE=USER, then the EDC Override Feature will treat the FCB values as missing and won't perform the override.

If the ROPES FCB cannot be loaded or the required parameters are missing from the FCB, then the EDC Override Feature will not perform the override.

Override Parameters

The EDC Override Feature also uses parameters that specify the offset, length and data format of the PCL code override. The following DEVINIT/DEVEXIT macro statement parameters describe all the para-meters that can be used to specify a PCL code override. Note again that the DVCRULE parameter is optional.

Parameter	**Description**
DVCOVRD	Specifies the override type name. Please see the Override Types table above.
DVCOVOF	Specifies the override offset within the PCL Overlay table/module.
DVCOVLN	Specifies the length of the data that will override PCL code in the PCL module.
DVCOVDF	Specifies the format of the data that will override PCL code in the PCL module.
DVCRULE	Specifies the name of a validation rule that will be used to validate the data overriding the PCL code.

Detailed descriptions for the coding of each of the override parameters can be found in the ROPES Administrator's Guide in the section on Device Characteristics Blocks, and the DEVINIT/DEVEXIT macro control statements.

TYPE RULES Macro Statement

The last parameter (DVCRULE) in the 'Override Parameters' table above specifies the name of a validation list that can be used to validate the overriding data values. The validation list is estab-lished by another Device Characteristics Block (DCB) macro control statement. That macro statement is 'ROPESDEV TYPE=RULES'. A sample of the TYPE=RULES macro statement appears below.

```
ROPESDEV TYPE=RULES,
         RULENME=CPIRULE,
         RULELST='10.00,12.00,15.00,16.67'
```

The RULENME parameter shown above identifies the name of the validation rule. The RULELST parameter specifies a list of valid data values for this validation rule. The TYPE=RULES macro statement is **optional**. If no validation rule is used, the EDC Override Feature will not perform a validation check on the override data. If a validation rule is specified in the DCB, and the same rule name is provided by the DVCRULE parameter on the DEVINIT/DEVEXIT macro statement, then a validation check will be made against the overriding data. If the overriding data fails the validation process, then the EDC Override Feature will issue an error message and terminate the print request abnormally.

Most of the time it will not be necessary to use a validation rule. The reason lies in the fact that most of the values are already validated by the ROPES FCB macro and the assembly process. However, as new override types are added to ROPES it may be necessary to use validation rules to verify the accuracy of the overriding data for some of the override types.

Below are several DCB sample macro statements showing the use of the TYPE=RULES and override parameters.

```
ROPESDEV TYPE=RULES,
         RULENME=CPIRULE,
         RULELST='10.00,12.00,15.00,16.67'


ROPESDEV TYPE=DEVINIT,
         DVCNAME=OVRDTEST,
         DVCKEYN=NONE,
         DVCRTRN=N,
         DVCSRCE=M,
         DVCTYPE=OVRDTEST,
         DVCBUFS=1024,
         DVCSTAT=Y,
         DVCOVRD=FCBCPI,
         DVCOVOF=10,
         DVCOVLN=5,
         DVCOVDF=Z,
         DVCRULE=CPIRULE
```

In the above sample, ROPES will load a PCL overlay module named OVRDTEST. The type name associated with this module (DVCTYPE) is also called 'OVRDTEST'. In order for the FCB to override the initial status of this overlay module, it would have to do so by using the name specified by the DVCTYPE parameter ('OVRDTEST'). In addition, the PCL overlay module's PCL code stream will be overridden by an FCB Characters Per Inch type of override. The PCL code stream contained in the overlay module will be overridden at offset 10 for a length of 5 bytes, and the format of the data will be numeric zoned decimal.

Note that the DVCRTRN parameter indicates that the PCL overlay code stream is already in ASCII format. This will also cause the overriding data to be converted from EBCDIC to ASCII.

The CPI override obtained from the FCB must contain one of the values specified by the RULELST parameter on the TYPE=RULES statement named 'CPIRULE'. If the DVCRULE parameter did not specify the name of an existing and valid TYPE=-RULES macro statement, then an assembly error would occur when the DCB is assembled. Please notice that the RULELST parameter values are enclosed in single quotes, and each item in the list is the same length. These are requirements when coding the validation list for the RULELST parameter.

Special Considerations - Printer/-Report Copies Overrides

Some EDC override types may require special consideration when combining their effects with other ROPES controls. Two such examples are the "PrinterCopies" and "ReportCopies" override types. Since the override values for these override types come from existing ROPES controls, the aggregate effect of the overrides and the normal controls will produce predictable but different results than what might be expected by using the ROPES controls alone.

First, both the "PrinterCopies" and "ReportCopies" override types change the number of device controlled copies (printer copies) that will be produced by the printer. This means the number of copies actually set in the printer's memory, and produced by the printer and not the application. If the number of override copies is set to 3, then the printer will produce 3 copies of each page of a given report. If the report were two pages long, then 3 copies of page 1 and 3 copies of page 2 would be produced.

Now consider the effect if we tell ROPES we want 2 copies of the entire report (collated). This can be accomplished by setting a value of two (2) in the copies parameter for the report definition, or it can be accomplished by setting a value of two (2) in the copies parameter for the printer definition (ROMT menu option 9). If the number of override copies is still 3, and there are two pages in the report we would receive the following output:

3 copies of page 1
3 copies of page 2 (1 complete collated copy)

3 copies of page 1
3 copies of page 2 (1 complete collated copy)

Remember that the override copy value is being obtained from the printer definition, or the report definition. These are exactly the same controls that are being used to specify the number of copies that ROPES will produce. That is why it is useful to use one of the ROPES controls for the override value, and one for the ROPES value. The one you select would be based on whether you want the ROPES copies controlled at the printer level, or at the report level. If you decide you want the copies value controlled at the printer level, you would probably select the "ReportCopies" override type so that you could set the number of override copies based on the value supplied by the report definition, and then use the printer definition copies to control the number of copies ROPES would produce. Please remember, using the native ROPES copy controls, the number of printer defined copies is always honored over the number of report definition copies. This may seem confusing, but actually can work to your advantage if you ever want to produce device copies (uncollated), and ROPES copies (collated) at the same time.

The table on page 64 describes the output produced by varying copy controls and EDC copy override types. Please use this table to setup your copy parameters if you wish to produce uncollated device copies along with your collated ROPES copies. Please note, that your PCL overlay module must also contain a PCL code sequence that sets the device level copies value in order to use a copies override.

In the table on page 64, X is the number of Report Definition Copies and Y is the number of Printer Definition Copies. Report definition copies is changed via transaction ROMT menu option 4, and Printer Definition Copies is changed via transaction ROMT menu option 9.

Other override types like the FCBCPI and LPI override types do not require any special considerations. They simply implement the specific-ations of the ROPES FCB for the PCL Overlay module.

Printer/Report Copies Override Examples

In this example, the Report Information Block (RIB or Report Definition) provides the copies override value which is 4. The Printer Control Record (PCR) con-tains a copy count of one (1).

```
ROPESDEV TYPE=RULES,
          RULENME=CPYRULE,
          RULELST='01,02,03,04,05,06,08,09,10'

ROPESDEV TYPE=DEVINIT,
          DVCNAME=OVRDTEST,
          DVCKEYN=NONE,
          DVCRTRN=N,
          DVCSRCE=M,
          DVCTYPE=OVRDTEST,
          DVCBUFS=1024,
          DVCSTAT=Y,
          DVCOVRD=ReportCopies,
          DVCOVOF=24,
          DVCOVLN=2,
          DVCOVDF=Z,
          DVCRULE=CPYRULE
```

The presence of the RULES macro statement indicates that the override value obtained from the RIB will be validated against the data items specified by the RULELST parameter. In this case, the valid number of copies can range from 1 to 10. This kind of limitation would probably not be imposed on the copies override, but it is provided here only as an example.

The PCL code module is named OVRDTEST. This sample PCL code (overlay) module can be found in the ROPES distribution source. Five PCL code sequences can be found in the OVRDTEST module. The five PCL code sequences perform a printer reset, set the printer's Orientation (Portrait/Landscape), set the CPI, LPI, and printer copies. The very first PCL code sequence is a printer reset sequence. The second sequence is the orientation sequence at offset 6. The third sequence is the CPI sequence at offset 11. The fourth PCL code sequence is the LPI code sequence at offset 20. The last PCL code sequence is the printer's copies sequence at offset 25. Note that the offset value specified by DVCOVOF is set to 24. The actual offset value is relative to zero, and must be 1 less than the actual count into the OVRDTEST module.

The length of the PCL code override is set by the parameter DVCOVLN. The length set in this example is 2. This means that the PCL code sequence in the PCL code module has a two digit data value that can be overridden at the offset specified by DVCOVOF. The length specified by the DVCOVLN parameter must match the length of the data value portion of the PCL code sequence in the PCL code (overlay) module. The copies PCL code sequence in EBCDIC is:

Esc & l nn X

The same PCL code fragment in ASCII is:

1B 26 6C 30 31 58

Note that the data value represented by the 'nn' or '30 31' is the same data that will be overridden by our example. The OVRDTEST module's current PCL code sequence sets a printer copies value of '01' (30 31). The 'ReportCopies' override will replace that data value with the value contained in the RIB record (report definition). The number of copies set in the report definition or RIB record is 4. Therefor, the printer will produce 4 uncollated copies of each page of the report. As we had stated at the very beginning of our example, the Printer Control Record has a copy count of 1. This means that ROPES will produce one collated copy of the entire report. If the report has 3 pages, each page will be printed 4 times, and the entire report will be printed once. This example corresponds with the first entry in the 'COPIES OVERRIDES RESULTS TABLE' described above.

The DVCOVDF parameter specifies how the final override data will be formatted. Internal data conversions may be necessary to put the override data value in the correct format for overriding the PCL code sequence. In our example, the RIB copies parameter (2 bytes binary) will be converted to a numeric Zoned Decimal format ('Z'). This means that the binary copies value is converted to Zoned Decimal format and placed in a work field (right justified) as a staging area for the override. Starting at the right-most byte, ROPES will move left the number of bytes specified by the DVCOVLN parameter (in this case 2), and then perform the override for the same length (2 bytes). The 'Z' format type specifies the data is numeric, right justified, and in zoned decimal (printable/character) format.

If the example had a PCL code sequence of:

1B 26 6C 30 30 31 58 or Esc & l 001 X

then the override length (DVCOVLN) would have to be 3. This would not present a problem, because the data value is assumed to be numeric and contains leading zeros. Even if the override copies value from the RIB is 4, the final converted override value would contain leading zeros ('004'). This is due to the fact that the DVCOVDF parameter is set to 'Z'. If the DVCOVDF parameter were 'C' (character only data), then the same assumption cannot be made. The 'C' data format is used when the final data format must be character data which can contain alpha-numeric data, or symbols. Therefor, such a data type would not contain leading zeros, and would not be right justified. In fact, this data would probably be padded with blanks and the blanks would probably not be suitable for use inside of a PCL code sequence.

In the next example, the copies override is coming from the Printer Control Record (PCR/Printer Definition). The number of printer copies is set at 2. The number of Report Copies (RIB/Report Definition) is 1.

```
ROPESDEV TYPE=DEVINIT,
         DVCNAME=OVRDTEST,
         DVCKEYN=NONE,
         DVCRTRN=N,
         DVCSRCE=M,
         DVCTYPE=OVRDTEST,
         DVCBUFS=1024,
         DVCSTAT=Y,
         DVCOVRD=PrinterCopies,
         DVCOVOF=24,
         DVCOVLN=2,
         DVCOVDF=Z
```

Note that the RULES macro statement is not present, and so no validation will occur against the final override copies value. The DVCRULE parameter on the DEVINIT macro statement has also been correctly omitted. Otherwise, everything is the same except for the DVCOVRD parameter which specifies the value 'PrinterCopies'. This value indicates that the copies override will be obtained from the Printer Control Record (PCR/Printer Definition). In our second ex-ample, the PCR record contains a copies value of 2. This means the PCL code sequence will contain the final data value:

1B 26 6C 30 32 58 or Esc & l 02 X

In this example, the printer will produce 2 uncollated copies of each page. Since the Printer Copies field is also the controlling copies value for ROPES, two (2) full copies of the entire report will also be produced. Since the PCR copies value always overrides the RIB copies value, the PCR copies value is used by ROPES to produce the collated copies (full report copies).

Detailed descriptions can be found for each of the DEVINIT/DEVEXIT override parameters in the ROPES Administrator's Guide in the section covering 'Device Characteristics Blocks'.

Minimum Length Override Example - CPI

In this example, the FCB's CPI value is used to override a PCL code sequence in the OVRDTEST module that sets the printer's characters per inch control. This example is called the 'Minimum Length Override Example' because the minimum length for the CPI override is 5. The FCB CPI parameter provides for CPI settings of; 10, 12, 15, and 16. However, the CPI setting of 16 really means '16.67'. Since an FCB CPI value of 16 really means 16.67, all CPI settings are converted to the same 5 digit format (nn.nn) so that all the parameters can be stored in a consistent format. For this example, the FCB CPI value is 16.

```
ROPESDEV TYPE=DEVINIT,
         DVCNAME=OVRDTEST,
         DVCKEYN=NONE,
         DVCRTRN=N,
         DVCSRCE=M,
         DVCTYPE=OVRDTEST,
         DVCBUFS=1024,
         DVCSTAT=Y,
         DVCOVRD=FCBCPI,
         DVCOVOF=10,
         DVCOVLN=5,
         DVCOVDF=Z
```

The DVCOVDF format is 'Z' which means the override data value format is numeric Zoned Decimal, right justified, and filled with leading zeros. The length of the override is determined by the DVC-OVLN parameter. Since this value is set to 5, the resulting override will be '16.67'. However, it would be possible to use a length of 6 provided the PCL code sequence could accommodate a value that is 6 digits long. If the DVCOVLN length were set to 6, the final override data value would be '016.67'. However, a DVCOVLN length of 4 would not be long enough. It would cause the value to be truncated on the left resulting in the value '6.67'. The CPI override automatically inserts a decimal point two digits from the right, and this decimal counts as one position. So, using a DVCOVLN length of 4 would result in an incorrect PCL code sequence value ('6.67'). This is the reason why the 'FCBCPI' value has a minimum override length of 5.

Please see the ROPES Administrator's Guide for more information on the minimum length requirements for various override types.

PCL Code Module Fragments

The EDC Override Feature supports one override per DEVINIT/DEVEXIT macro control statement in the DCB. This means one override can be applied to a given PCL code module per DEVINIT/DEVEXIT macro statement. Since coding multiple DEVINIT/-DEVEXIT macro statements using the same PCL code module would cause that PCL code module to be sent to the device multiple times, the correct approach to applying more than one override to a PCL code module would be to break that module up into smaller fragments.

If the FCBCPI, FCBLPI, and FCBOrient override types were to be applied to the OVRDTEST module, it would have to be broken up in a number of smaller PCL code modules. One module would contain the reset and orientation PCL code sequence, another would contain the CPI PCL code sequence, and one final module would have to be created to contain the LPI PCL code sequence. Each PCL code module would be defined on a separate DEVINIT/DEVEXIT macro statement. Each DEVINIT/DEVEXIT macro statement would then have separate PCL code overrides defined corresponding with the type of override required for each PCL code module. In addition, the status of each PCL code module could be determined by setting the DVCSTAT parameter on each DEVINIT/DEVEXIT macro statement. In this manner, each PCL code module could have an initial status of enabled or disabled. PCL code modules defined to be disabled, can be enabled later by using the DEVINIT/DEVEXIT macro statements of the FCB. In this manner, overrides can be applied or not depending on the report being processed.

An example of breaking up the OVRDTEST module into separate (fragment) PCL code modules can be seen in the example below.

```
ROPESDEV TYPE=DEVINIT,
         DVCNAME=OVRDORIN,
         DVCKEYN=NONE,
         DVCRTRN=N,
         DVCSRCE=M,
         DVCTYPE=OVRDORIN,
         DVCBUFS=1024,
         DVCSTAT=Y,
         DVCOVRD=FCBOrient,
         DVCOVOF=5,
         DVCOVLN=1,
         DVCOVDF=C
```

```
ROPESDEV TYPE=DEVINIT,
         DVCNAME=OVRDCPIM,
         DVCKEYN=NONE,
         DVCRTRN=N,
         DVCSRCE=M,
         DVCTYPE=OVRDCPIM,
         DVCBUFS=1024,
         DVCSTAT=Y,
         DVCOVRD=FCBCPI,
         DVCOVOF=3,
         DVCOVLN=5,
         DVCOVDF=Z

ROPESDEV TYPE=DEVINIT,
         DVCNAME=OVRDLPIM,
         DVCKEYN=NONE,
         DVCRTRN=N,
         DVCSRCE=M,
         DVCTYPE=OVRDLPIM,
         DVCBUFS=1024,
         DVCSTAT=Y,
         DVCOVRD=FCBLPI,
         DVCOVOF=3,
         DVCOVLN=1,
         DVCOVDF=Z
```

Each new PCL code module would contain just the PCL code sequence for doing orientation, CPI, or LPI as indicated by the module name. In the samples above, all the PCL code modules are enabled (DVCSTAT=Y) and will be sent to the printer device. If the DVCSTAT parameter were changed to 'DVCSTAT=N', each of the PCL code modules could be enabled or activated by the associated report's FCB. The FCB DEVINIT/DEVEXIT macro statements can be used to enable/activate each PCL code module by specifying the DVCTYPE name, and specifying a new status. Each report's FCB can activate all, one, or a subset of the available PCL code modules with the corresponding overrides.

Copies Override Results Table

Override Type	Report Definition Copies	Printer Definition Copies		Collated	Uncollated
ReportCopies	If X >= 1	If Y >= 1	=Y	=X*Y	
ReportCopies	If X >= 1	If Y = 0	=X	=X*X	
ReportCopies	If X = 0	If Y >= 1	=Y	=X or 0	
ReportCopies	If X = 0	If Y = 0	=1	=0	
PrinterCopies	If X >= 1	If Y >= 1	=Y	=Y*Y	
PrinterCopies	If X >= 1	If Y = 0	=X	=Y or 0	
PrinterCopies	If X = 0	If Y >= 1	=Y	=Y*Y	
PrinterCopies	If X = 0	If Y = 0	=1	=0	

Figure 14 Copies Override Results

Special Considerations

Translation tables are now fully supported with the Extended Device Controls Feature. Translation tables may be implemented through the use of the DCB macro statement 'ROPESDEV TYPE=TRANSLATE,TRANTAB='. Translation tables permit the translation of unprintable characters to printable characters. Please see the ROPES Administrator's Guide for more information on the 'Device Characteristics Blocks' macro statement 'TYPE=TRANSLATE'.

The DVCRTRN parameter of the DCB DEVINIT and DEVEXIT macro statements controls character translation for the PCL code sequence data. The PCL code sequence data is often generated using a word processing tool, which results in ASCII PCL code sequences. If the PCL control code data is in ASCII format, and it is being sent to an ASCII device, then the DVCRTRN parameter value must be set to “N” (no translation required). For example, a TCP/IP LPR type printer will require the PCL code data to be in ASCII format. If the PCL Code data is already in ASCII format, then no translation is required and the DVCRTRN parameter value should be set to “N”. If the PCL code sequence data were in EBCDIC format, then translation would be required and the DVCRTRN parameter would be set to “Y”. Note that PCL Code sequences cannot be in both ASCII and EBCDIC formats in the same PCL Code table or file. The DVCRTRN parameter effects translation for TCP/IP LPR printers only.

Performance Considerations

The DEVINIT and DEVEXIT statements in the DCB contain a parameter named DVCBUFS. This parameter can be used to reduce the number of calls made to ROPEDVCP when large PCL code sequences are transported to the ROPES printer driver module. However, the value of DVCBUFS cannot exceed the value of BUFSIZE minus 4. The BUFSIZE parameter is another DCB buffer size parameter that controls the maximum amount of data that can be sent to the device. For more information on the use of the BUFSIZE and DVCBUFS parameters, please consult the ROPES Administrator's Guide under the section on DCB customization.

TCP/IP Translation Facility

Introduction

ROPES supports several data translation options for TCP/IP attached printers in LPR or Direct Socket mode. In addition to the existing translations using the CICS Sockets routines EZACIC04 and EZACIC05, we have added support for EZACIC14, EZACIC15 and EZACICTR, including the EZACIA2E and EZACIE2A entry points.

The EZACICTR routine is provided as a sample routine in z/OS Communications Server for z/OS V1R9.0, and is available on the web for earlier releases. The EZACICTR routine and its entry points allow an installation to:

- select a translation table dynamically;
- add their own translation tables; or
- pass the translation table to use to the translation routine.

ROPES supports all of these uses. The implementation is described below.

Preparation of the EZACICTR Module

It is the installation's responsibility to obtain, tailor, assemble and link-edit the EZACICTR module and make this module available to ROPES. Once this module is available, a ROPES source library member LINKCTR and it's input member LINKEZAC, are used to connect the EZACICTR routine to the ROPES modules that need to access the routines. It is a requirement of the EZACICTR routine that it be statically linked to CICS programs as it must be CALLed and cannot be invoked with a CICS LINK command.

User Controls

You control the translation process through one of several methods.

1. You can continue to use the defaults of EZACIC04 (output translation) and EZACIC05 (input translation). To do this, you need not do anything at all, as ROPES will use these defaults in the absence of any other instructions.

2. You can specify the translation to be used by default in those cases where you have not supplied any printer or report specific translation rules. To do this, you must run the ROPES Batch Utility program ROPETRAN (described elsewhere) to set the default translation methods, and if appropriate, translation tables to be used. You set separate default values for the LPR method or the Direct Socket method.

3. You can specify the translation to be used for a specific printer and for a specific report on a specific printer, if this level of specification is required. The ROLI transaction allows you to define the TCP/IP characteristics used for LPR and Direct Socket printing. Five new fields have been added to the data update panel to allow you to specify the Output Translation routine, the Input Translation routine, the Output Translation Table Name, when the E2A method is used, the Input Translation Table Name, when the A2E method is used, the User Input and output Translation Table names, when the CTR method is used, and the mode (Single or Multiple) used to prepare the EZACICTR module. Please refer to the ROLI transaction documentation in the Administrator's Guide.

Command Level Programming

Introduction

This manual describes the Remote Online Print Executive System (ROPES) command level application programming services. Application programmers request ROPES services by means of ROPES commands. These commands are sets of instructions that can be included at appropriate points in an application program.

ROPES commands can be used in application programs written in Assembler, American National Standard (ANS) COBOL, and in PL/I. The ROPES commands are essentially the same in each language, differing only as the language syntax requires.

ROPES Command Syntax

This chapter describes the general rules governing the coding of the ROPES commands. ROPES commands are distributed as sets of instructions residing in the appropriate programming language library (books for DOS, members for OS). The ROPES commands can be included in Assembler, ANS COBOL, and PL/I programs anywhere that an executable statement can be included. All ROPES commands follow a naming convention outlined by the rules listed below (book names for DOS or member names for OS):

- All command names start with a prefix "ROPE";
- then the letter "O" or "B" indicates the type of application. The letter "O" identifies commands to be used by online application programs. The letter "B" identifies commands to be used by batch application programs; and
- finally, a two letter identifier of the command function (e.g., the letters "PR" indicate a Prepare function).

For example:

ROPE<u>O</u>PR . . . Online Prepare command.

ROPE<u>B</u>SL . . . Batch Sendline command.

Coding the ROPES Commands - COBOL

COBOL application programs must include the ROPES commands by using the COPY statement. The default ROPES command options such as data names and paragraph names can be changed by using the REPLACING option of the COPY statement.

For example:

Using the default options . . .

```
                     .
ROPES-FUNCTION. COPY  ROPEOPR.
                     .
ERR-PARA.
                     .
```

Changing the default options . . .

```
                     .
                     .
ROPES-FUNCTION. COPY  ROPEOPR
           REPLACING ERR-PARA BY MY-ERROR.
                     .
MY-ERROR.
                     .
```

Coding the ROPES Commands - PL/I

PL/I application programs issue ROPES commands by using the %INCLUDE statement. The default ROPES command options such as data names and paragraph names may be changed by using the preprocessor facilities of the PL/I Compiler.

For example:

Using the default options . . .

```
              .
              .
%INCLUDE ROPEOPR;
```

Changing the default options . . .

```
              .
              .
%DECLARE (LINE_AREA,ERR_PARA) CHAR;
%LINE_AREA = 'MY_LINE';
              .
%ERR_PARA = 'BAD_PREPARE';
%INCLUDE ROPEOPR;
              .
BAD_PREPARE:
              .
```

Coding the ROPES Commands - Assembler

Assembler application programs must specify the ROPES commands by coding macro instructions. The assembler programmer can change the options of the commands by giving values to the keywords provided with the commands.

For example:

Using the default options . . .

```
                .
                .
     ROPEOPR
                .
```

Changing the default options . . .

```
                .
                .
     ROPEOPR ERRPARA=BADPREP
                .
 BADPREP DS 0H
                .
```

ROPES Programming

This chapter contains the information necessary for the application programmer to process ROPES reports. This chapter does not contain any guidance on the use of programming language statements or program techniques that are unrelated to ROPES. Such information is given in the appropriate language publications.

Definitions

The application programmer should be familiar with the following terms and symbols used throughout this section:

ROPES Communication Area
ROPES Report Line
ROPES Report Name
ROPES Report
ROPES Error Paragraph
ROPES Line Group Area

These terms are defined below.

ROPES Communication Area

The ROPES Communication Area contains information required for all ROPES report processing. It is distributed in each of the programming language libraries under the name ROPESPRM. In general, this area contains fields such as request code, response code, report line, report name area, etc. (For a detailed description of the fields in the ROPES Communication Area refer to Appendix A on page 88 in this section.)

In order to include the ROPES Communication Area definition into the application program, the following must be coded:

For COBOL programs . . .

```
           .
 01 ROPESPRM  COPY ROPESPRM.
           .
```

For PL/I programs . . .

```
           .
 %INCLUDE ROPESPRM;
           .
```

For Assembler programs . . .

```
   .
   .
 ROPESPRM
```

Additionally, the default names of certain fields in the ROPES Communication Area may be changed in the manner shown for the ROPES commands under the heading The ROPES Commands on page 81. The default names of these fields are:

Assembler:
LINAREA
LINCC
LINDATA
RPTNAME

COBOL:
LINE-AREA
LINE-CC
LINE-DATA
REPORT-NAME

PL/I:

LINE_AREA
LINE_CC
LINE_DATA
REPORT_NAME

Online application programs must define the ROPES Communication Area in non-static, non-volatile storage to maintain CICS pseudo-reentrancy. For COBOL, the WORKING-STORAGE SECTION should be used. For PL/I programs, AUTOMATIC-class storage is dynamically obtained. For Assembler programs, the ROPES Communication Area must be defined after statement "DFHEISTG DSECT" (use of the "DFHEISTG" reserved name is described in the Customer Information Control System Application Programmer's Reference Manual (Command Level) under "Command Language Translator").

ROPES Report Line

A report line is a 255 character area used to send or receive a line of text between the application program and ROPES by the Sendline, Readline, Forwardspace and Backspace commands. A default report line area is provided in the ROPES Communication Area. The default name of this area is:

Assembler: LINAREA
COBOL: LINE-AREA
PL/I: LINE_AREA

The report line must adhere to the following rules:

- The first character must contain a valid carriage control character as defined in Appendix C. The default name of this area is:

  Assembler: LINCC
  COBOL: LINE-CC
  PL/I: LINE_CC

- Characters 2 through 255 contain text characters or blanks. The default name of this area is:

  Assembler: LINDATA
  COBOL: LINE-DATA
  PL/I: LINE_DATA

ROPES Report Name

A report name is an eight character name which uniquely identifies a ROPES report. A ROPES report name must be defined to ROPES (usually by a system programmer) before it may be used by an application program. A default ROPES report name area is provided in the ROPES Communication Area. The ROPES report name area must be initialized with a ROPES report name before report processing is begun. The default name of this area is:

Assembler: RPTNAME
COBOL: REPORT-NAME
PL/I: REPORT_NAME

ROPES Report

A ROPES report consists of report lines stored under a report name. After generation, a report is usually routed to ROPES-controlled printers by ROPES system modules using a combination of system tables and terminal operator commands.

Appendable Reports (APPEND Facility)

The append facility allows an application program to add report lines to the end of a report after it has been successfully generated. The append facility may be selectively disabled through the ROPES control tables.

ROPES Error Paragraph

The ROPES error paragraph will be given control by any ROPES command which generates a response code which indicates that an exceptional condition (see ROPES Command Exceptional Conditions on page 86) was detected during processing of the ROPES command. The default routine name is:

Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

ROPES Line Group Area

The ROPES Line Group Area is used only by the Findreport, Sendlinegroup, Readlinegroup, Forwardspace and Backspace commands. This usage is explained in "ROPES Commands." It is distributed in each of the programming language libraries under the name ROPESLGA. In general, this area contains fields such as the group of report lines, the count of lines to send, space or retrieve, control codes, etc. (For detailed description of the fields in the ROPES Line Group Area refer to Appendix B on page 88.)

In order to include the ROPES Line Group Area definition into the application program, the following must be coded:

For COBOL programs . . .

```
        .
        .
    * BUILD 22 LINES IN LINE GROUP AREA
     01 ROPESLGA  COPY ROPESLGA REPLACING
                    LINE-COUNT BY 22.
        .
        .
```

For PL/I programs . . .

```
                    .
                    .
    /* BUILD 22 LINES IN LINE GROUP AREA */
       %DECLARE LINE_COUNT CHAR;
       %LINE_COUNT = '22';
       %INCLUDE ROPESLGA;
                    .
                    .
```

For Assembler programs . . .

```
                  .
                  .
    ROPESLGA LINECNT=22       BUILD 22 LINES
```

Note that the number of line areas to be generated in the ROPES Line Group Area must be specified in the manner shown above. The name of this variable is:

Assembler:	LINECNT
COBOL:	LINE-COUNT
PL/I:	LINE_COUNT

Online application programs must define the ROPES Line Group Area in non-static, non-volatile storage to maintain CICS pseudo-reentrancy. For COBOL, the WORKING-STORAGE SECTION should be used. For PL/I programs, AUTOMATIC-class storage is dynamically obtained. For Assembler programs, the ROPES Line Group Area must be defined after statement "DFHEISTG DSECT" (use of the "DFHEISTG" reserved name is described in the Customer Information Control System Application Programmer's Reference Manual (Command Level) under "Command Language Translator").

Report Processing

This section describes the general use and execution sequence of the ROPES commands. The ROPES commands are described in detail in Chapter 6 of this section. The descriptions and examples given below represent a sequence of events which lead to the successful processing of a ROPES report. They are not intended to serve as a guide to coding applications. The two modes of processing a ROPES report (report generation and report retrieval) are described separately.

Report Generation

ROPES reports can be generated using batch (job step) or online (CICS transaction) application programs. Reports are generated by sending one report line at a time to ROPES using the Sendline command or by sending several lines at once to ROPES using the Sendlinegroup command. Once a report is generated successfully, it becomes eligible for retrieval by online and batch application programs and printing by ROPES on the destination printers.

The ROPES commands used to generate reports are:

	ONLINE	*BATCH*
Prepare	ROPEOPR	ROPEBPR
Sendline	ROPEOSL	ROPEBSL
Sendlinegroup	ROPEOSG	ROPEBSG
Endlines	ROPEOEL	ROPEBEL
Checkpoint	ROPEOCK	ROPEBCK
Terminate	ROPEOTR	ROPEBTR
Delete report	ROPEODR	ROPEBDR

In general, the following sequence of ROPES commands is used when generating a report:

Prepare (ROPEOPR, ROPEBPR)

Sendline (ROPEOSL, ROPEBSL)
-or-
Sendlinegroup (ROPEOSG, ROPEBSG)

Endlines (ROPEOEL, ROPEBEL)
-or-
Delete report (ROPEODR, ROPEBDR)
Checkpoint (ROPEOCK, ROPEBCK)

Terminate (ROPEOTR, ROPEBTR)

- The Prepare command must be executed before any other ROPES processing can occur. After the Prepare command is executed successfully, one or more reports may be generated by the same program execution.

- The Sendline command is executed once for each line in a report. Alternatively, the Sendlinegroup command may be used to send several lines to the report at once and, optionally, perform an Endlines command after the lines have been sent.

- The Endlines command must be executed after all the lines for a report have been generated. Once this command is executed a new report may be started by changing the report name and sending lines to the new report using the Sendline or Sendlinegroup command.

- The Checkpoint command is used to preserve the report generation status. The status of a ROPES report (e.g. the number of buffer records written for this report) is reflected in the ROPES system information tables. The Checkpoint command directs ROPES to save these tables so that program failure or system outages will not cause the loss of the completed reports. Since the generated report lines are "non-existent" until the Checkpoint command is issued for batch applications, its use is highly dependent on the application program requirements. The Checkpoint command must be issued at least once, usually immediately proceeding the Terminate command. There are two common cases that illustrate the use of the Checkpoint command:

- An application program is designed so it will generate all reports every time it is started. Under this condition, the Checkpoint command must be executed at the end of ROPES processing, before the Terminate command is executed;

- An application program, when restarted, only generates the reports not previously completed. Under this condition the Checkpoint command must be issued immediately after every Endlines command.

- The Terminate command is executed to indicate termination of ROPES processing by the application program.

- The Delete report command should be executed when an error condition occurs which prevents the successful completion of report generation. This insures proper "clean-up" processing. A more detailed description of the Delete report command usage is provided under the heading "Exceptional Conditions."

- ROPES expects you to begin each page with a Skip to Channel 1 carriage control character ('1') in the first position of the first line of the new page. If you do not do so, then logically, the page starts at the first line you generate and is of no particular length. If ROPES needs to position to the start of the current page to recover from an I/O error or restart after an interrupt command and no Skip to Channel 1 carriage control is found, ROPES position to the start of the report. Forward and backward spacing will not be possible, and error recovery will be adversely affected.

General Examples of Report Generation Flow

<u>Online COBOL - Line-at-a-time Generation</u>

```
           .
           .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
           .
           .
 PROCEDURE DIVISION.
           .
           .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE. COPY ROPEOPR.
           .
           .
* READ RECORD TO BE PROCESSED
 READ-CUSTOMER-FILE.
           .
           .
* BUILD ROPES REPORT LINE
           .
           .
* SEND THIS LINE TO ROPES NOW
 ROPES-SENDLINE. COPY ROPEOSL.
           .
           .
 IF MORE-DATA GO TO READ-CUSTOMER-FILE.
           .
* AT THIS POINT THERE IS NO MORE DATA
* TO PROCESS
* END THE REPORT GENERATION
       .
 ROPES-ENDLINES. COPY ROPEOEL.
 ROPES-TERMINAT. COPY ROPEOTR.
        .
        .
```

<u>Batch COBOL - Line-at-a-time Generation</u>

```
        .
        .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
        .
        .
 PROCEDURE DIVISION.
        .
        .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE. COPY ROPEBPR.
        .
```

```
        .
* READ RECORD TO BE PROCESSED
 READ-CUSTOMER-FILE.
         .
         .
* BUILD ROPES REPORT LINE
         .
         .
* SEND THIS LINE TO ROPES NOW
 ROPES-SENDLINE. COPY ROPEBSL.
         .
         .
 IF MORE-DATA GO TO READ-CUSTOMER-FILE.
         .
         .
* AT THIS POINT THERE IS NO MORE DATA
* TO PROCESS
* END THE REPORT GENERATION
         .
         .
 ROPES-ENDLINES. COPY ROPEBEL.
 ROPES-CHECKPNT. COPY ROPEBCK.
 ROPES-TERMINAT. COPY ROPEBTR.
```

Online PL/I - Line-at-a-time Generation

```
           .
           .
%INCLUDE ROPESPRM;
           .
           .
/* PREPARE FOR ROPES PROCESSING */
%INCLUDE ROPEOPR;
           .
           .
/* PROCESSING LOOP */
           .
           .
/* READ A RECORD FROM CUSTOMER FILE */
/* FORMAT REPORT LINE USING */
/* CUSTOMER INFORMATION */
           .
/* ISSUE ROPES SENDLINE COMMAND */
%INCLUDE ROPEOSL;
           .
/* DO PROCESSING LOOP UNTIL ALL CUSTOMER */
/* RECORDS HAVE BEEN PROCESSED */
           .
           .
/* LAST CUSTOMER RECORD PROCESSED */
/* END ROPES REPORT GENERATION */
%INCLUDE ROPEOEL;
/* TERMINATE ROPES PROCESSING */
%INCLUDE ROPEOTR;
           .
```

Batch PL/I - Line-at-a-time Generation

```
           .
           .
%INCLUDE ROPESPRM;
           .
           .
/* PREPARE FOR ROPES PROCESSING */
%INCLUDE ROPEBPR;
           .
           .
/* PROCESSING LOOP */
           .
           .
/* READ A RECORD FROM CUSTOMER FILE */
/* FORMAT REPORT LINE USING */
/* CUSTOMER INFORMATION */
           .
           .
/* ISSUE ROPES SENDLINE COMMAND */
%INCLUDE ROPEBSL;
           .
           .
/* DO PROCESSING LOOP UNTIL ALL CUSTOMER */
/* RECORDS HAVE BEEN PROCESSED */
           .
           .
/* LAST CUSTOMER RECORD PROCESSED */
/* END ROPES REPORT GENERATION */
%INCLUDE ROPEBEL;
/* CHECKPOINT THE REPORT */
%INCLUDE ROPEBCK;
/* TERMINATE ROPES PROCESSING */
%INCLUDE ROPEBTR;
           .
           .
```

Online Assembler - Line-at-a-time Generation

```
             .
             .
DFHEISTG DSECT
         ROPESPRM
             .
             .
MYPROGRM CSECT
             .
             .
         ROPEOPR       PREPARE
             .
             .
GENREPT  EQU  *        REPORT GENERATION LOOP
             .
             .
* READ CUSTOMER RECORD.
             .
             .
* FORMAT THE ROPES REPORT LINE.
             .
             .
           ROPEOSL      SEND LINE TO ROPES
             .
             .
* IF MORE RECORDS IN CUSTOMER FILE
* GO TO GENREPT.
             .
* NO MORE RECORDS, END REPORT GENERATION
           ROPEOEL      END REPORT GENERATION
           ROPEOTR      TERMINATE PROCESSING
             .
```

Batch Assembler - Line-at-a-time Generation

```
MYPROGRM CSECT
             .
             .
         ROPEBPR       PREPARE PROCESSING
             .
             .
GENREPT  EQU  *         REPORT GENERATION LOOP
             .
             .
* READ CUSTOMER RECORD.
             .
             .
* FORMAT THE ROPES REPORT LINE.
             .
             .
         ROPEBSL        SEND LINE TO ROPES
             .
             .
* IF MORE RECORDS IN CUSTOMER FILE
* GO TO GENREPT.
             .
             .
* NO MORE RECORDS, END REPORT GENERATION
         ROPEBEL        END REPORT GENERATION
         ROPEBCK        CHECKPOINT REPORT
         ROPEBTR        TERMINATE PROCESSING
             .
             .
         ROPESPRM       COMMUNICATION AREA
             .
```

Online COBOL - Line-group Generation

```
       .
       .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
 01 ROPESLGA  COPY ROPESLGA
                 REPLACING LINE-COUNT BY 20.
* WE WILL SEND UP TO 20 LINES AT ONCE
       .
 PROCEDURE DIVISION.
       .
     MOVE 1 TO INDEX
       .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE.  COPY ROPEOPR REPLACING
                     GROUP-AREA BY ROPESLGA.
* USE THE LINE-GROUP-AREA CALLED ROPESLGA
       .
* READ RECORD TO BE PROCESSED
 READ-CUSTOMER-FILE.
       .
* BUILD ROPES REPORT LINES (UP TO 20) IN
* THE LINE-GROUP-AREA
       .
     MOVE MY-DATA TO ROPESLGA-LINE (INDEX).
     ADD 1 TO INDEX.
       .
* WHEN THE INDEX = 20 SEND LINES TO ROPES
     MOVE 20 TO ROPESLGA-LINE-COUNT.
 ROPES-SENDLG.  COPY ROPEOSG.
       .
       .
     IF MORE-DATA GO TO READ-CUSTOMER-FILE.
       .
       .
* AT THIS POINT THERE IS NO MORE DATA
* TO PROCESS
* BUILD THE 5 TOTAL LINES IN THE LINE
* GROUP AREA
       .
       .
* SEND THE LAST 5 LINES (TOTAL LINES) AND
* END THE REPORT
     MOVE 5 TO ROPESLGA-LINE-COUNT.
     MOVE HIGH-VALUE TO ROPESLGA-END-LINES.
 ROPES-LAST-GROUP.  COPY ROPEOSG.
 ROPES-TERMINAT.  COPY ROPEOTR.
       .
       .
```

Batch COBOL - Line-group Generation

```
       .
       .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
 01 ROPESLGA COPY ROPESLGA
                 REPLACING LINE-COUNT BY 20.
* WE WILL SEND UP TO 20 LINES AT ONCE
       .
     MOVE 1 TO INDEX.
       .
 PROCEDURE DIVISION.
       .
       .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE. COPY ROPEBPR REPLACING
                     GROUP-AREA BY ROPESLGA.
* USE THE LINE-GROUP-AREA CALLED ROPESLGA
       .
       .
* READ RECORD TO BE PROCESSED
 READ-CUSTOMER-FILE.
       .
       .
* BUILD ROPES REPORT LINES (UP TO 20) IN
* THE LINE-GROUP-AREA
       .
     MOVE MY-DATA TO ROPESLGA-LINE (INDEX).
     ADD 1 TO INDEX.
       .
* WHEN THE INDEX = 20 SEND LINES TO ROPES
     MOVE 20 TO ROPESLGA-LINE-COUNT.
 ROPES-SENDLG. COPY ROPEBSG.
       .
     IF MORE-DATA GO TO READ-CUSTOMER-FILE.
       .
* AT THIS POINT THERE IS NO MORE DATA
* TO PROCESS
* BUILD THE 5 TOTAL LINES IN THE
* LINE-GROUP-AREA
       .
* SEND THE LAST 5 LINES AND END THE REPORT
     MOVE 5 TO ROPESLGA-LINE-COUNT.
     MOVE HIGH-VALUE TO ROPESLGA-END-LINES.
 ROPES-LAST-GROUP. COPY ROPEBSG.
 ROPES-CHECKPNT. COPY ROPEBCK.
 ROPES-TERMINAT. COPY ROPEBTR.
```

Online PL/I - Line-group Generation

```
       .
 %INCLUDE ROPESPRM;
 %DECLARE LINE_COUNT CHAR;
 %LINE_COUNT = '20'; /* WE CAN DO 20 NOW*/
 %INCLUDE ROPESLGA;
       .
 %DECLARE GROUP_AREA CHAR;
 %GROUP_AREA = 'ROPESLGA'
 /* USE THE LINE GROUP AREA */
 /* PREPARE FOR ROPES PROCESSING */
 %INCLUDE ROPEOPR;
       .
 /* PROCESSING LOOP */
       .
 LINE_INDEX = 1;
       .
 /* READ A RECORD FROM CUSTOMER FILE */
 /* FORMAT REPORT LINES USING CUSTOMER */
 /* INFORMATION*/
 /* WE CAN SEND UP TO 20 LINES AT ONCE */
       .
 ROPESLGA_LINE(LINE_INDEX) = MY_DATA;
 LINE_INDEX = LINE_INDEX + 1;
       .
 /* SEND 20 LINES THIS TIME */
 ROPESLGA_LINE_COUNT = 20;
 /* ISSUE ROPES SEND GROUP COMMAND */
 %INCLUDE ROPEOSG;
       .
       .
 /* DO PROCESSING LOOP UNTIL ALL CUSTOMER */
 /* RECORDS HAVE BEEN PROCESSED */
       .
       .
 /* LAST CUSTOMER RECORD PROCESSED */
 /* BUILD THE 5 TOTAL LINES IN LINE */
 /* GROUP AREA */
       .
       .
 /* SEND THE LAST 5 LINES AND END REPORT */
 ROPESLGA_LINE_COUNT = 5;
 ROPESLGA_END_LINES = B'11111111';
 %INCLUDE ROPEOSG;
 /* TERMINATE ROPES PROCESSING */
 %INCLUDE ROPEOTR;
```

Batch PL/I - Line-group Generation

```
       .
       .
 %INCLUDE ROPESPRM;
 %DECLARE LINE_COUNT CHAR;
 %LINE_COUNT = '20'; /* UP TO 20 LINES */
 %INCLUDE ROPESLGA;
 %DECLARE GROUP_AREA CHAR;
 %GROUP_AREA = 'ROPESLGA';
 /* USE THE LINE GROUP AREA */
 /* PREPARE FOR ROPES PROCESSING */
 %INCLUDE ROPEBPR;
       .
 LINE_INDEX = 1;
```

```
     .
/* PROCESSING LOOP */
     .
     .
/* READ A RECORD FROM CUSTOMER FILE */
/* FORMAT REPORT LINES USING CUSTOMER */
/* INFORMATION */
/* WE CAN SEND UP TO 20 LINES AT ONCE */
     .
ROPESLGA_LINE(LINE_INDEX) = MY_DATA;
LINE_INDEX = LINE_INDEX + 1;
     .
/* SEND 20 LINES THIS TIME */
ROPESLGA_LINE_COUNT = 20;
/* ISSUE ROPES SEND GROUP COMMAND */
%INCLUDE ROPEBSG;
     .
/* DO PROCESSING UNTIL ALL CUSTOMERS */
 /* RECORDS HAVE BEEN PROCESSED */
     .
/* LAST CUSTOMER RECORD PROCESSED */
/* BUILD THE 5 TOTAL LINES IN LINE */
/* GROUP AREA */
     .
/* SEND THE LAST 5 LINES AND END REPORT */
ROPESLGA_LINE_COUNT = 5;
ROPESLGA_END_LINES = B'11111111';
%INCLUDE ROPEBSG;
/* CHECKPOINT THE REPORT */
%INCLUDE ROPEBCK;
/* TERMINATE ROPES PROCESSING */
%INCLUDE ROPEBTR;
```

Online Assembler - Line-group Generation

```
       .
DFHEISTG DSECT
         ROPESPRM
         ROPESLGA LINECNT=20
       .
MYPROGRM CSECT
       .
       ROPEOPR GRPAREA=ROPESLGA  PREPARE
       .
GENREPT  EQU     *      GENERATION LOOP
       .
* READ CUSTOMER RECORD.
       .
* FORMAT THE 20 ROPES REPORT LINE.
       .
* COMPUTE OFFSET TO NEXT LINE AND STORE DATA.
       L     R1,INDEX         GET THE INDEX
       BCTR  R1,0             SET FOR OFFSET
       MH    R1,=H'255'       COMPUTE OFFSET
       LA    R1,RLGALINE(R1)  COMPUTE ADDRESS
       MVC   0(255,R1),MYLINE MOVE THE DATA
       L     R1,INDEX         GET INDEX AGAIN
       LA    R1,1(R1)         INCREMENT LINE
       ST    R1,INDEX         FOR NEXT CYCLE
       .
       MVC   RLGALCNT,=F'20'  SEND 20 LINES
       ROPEOSG                SEND GROUP
       .
* IF MORE RECORDS IN CUSTOMER FILE
* GO TO GENREPT.
       .
* NO MORE RECORDS
* BUILD THE 5 TOTAL LINES IN LINE GROUP AREA
       .
* SEND THE LAST 5 LINES AND END THE REPORT
        MVC  RLGALCNT,=F'5'    SEND 5 LINES
        MVI  RLGAENDL,X'FF'    AUTO-ENDLINES
        ROPEOSG                END NORMALLY
        ROPEOTR                TERMINATE
```

Batch Assembler - Line-group Generation

```
MYPROGRM CSECT
       .
         ROPEBPR GRPAREA=ROPESLGA   PREPARE
       .
GENREPT  EQU   *      REPORT GENERATION LOOP
       .
* READ CUSTOMER RECORD.
       .
* FORMAT THE ROPES REPORT LINE.
       .
* COMPUTE OFFSET TO NEXT LINE AND STORE DATA.
         L     R1,INDEX    GET THE INDEX
         BCTR  R1,0        SET FOR OFFSET
         MH    R1,=H'255'  COMP OFST FOR LINE
         LA    R1,RLGALINE(R1)  COMP LINE ADR
         MVC   0(255,R1),MYLINE MOVE IN DATA
         L     R1,INDEX    GET INDEX AGAIN
         LA    R1,1(R1)    INCREMENT FOR LINE
         ST    R1,INDEX    FOR NEXT CYCLE
       .
         MVC   RLGALCNT,=F'20'  SEND 20 LINES
         ROPEBSG               SEND GROUP TO ROPES
       .
* IF MORE RECORDS IN CUSTOMER FILE GO TO
* GENREPT.
       .
* NO MORE RECORDS
* BUILD THE 5 TOTAL LINES IN LINE GROUP AREA
       .
* SEND LAST 5 LINES (TOTAL LINE), END  REPORT
         MVC   RLGALCNT,=F'5'    SEND 5 LINES
         MVI   RLGAENDL,X'FF'    AUTO-ENDLINES
         ROPEBSG                 END NORMALLY
         ROPEBCK                 CHECKPOINT
         ROPEBTR                 TERMINATE
       .
         ROPESPRM          COMMUNICATION AREA
         ROPESLGA LINECNT=20
```

Report Retrieval

Successfully generated ROPES reports may be retrieved by an application program at any time, if the report was not deleted in the interim.

Report retrieval is accomplished by requesting one report line at a time from ROPES using the Readline command, by requesting several lines at a time from ROPES using the Readlinegroup command, or by positioning in the report using the Forwardspace and Backspace commands. Response code QRCERPT (in ROPESPRM) indicates that the last line of the report has been retrieved, analogous to an end-of-file situation when processing a sequential file.

Report retrieval is performed by ROPES whenever a report is selected by ROPES for printing on a ROPES controlled printer. Report retrieval by an application program does not affect the printing of the report by ROPES.

ROPES reports can be retrieved by online (CICS transaction) or batch (job step) application programs. Typical usages of report retrieval are outlined by the following requirements:

- A batch program needs to inspect print output from online transactions to product summary reports.

- An operator needs to browse a report prior to printing to be sure a job is executed correctly.

The ROPES commands used to retrieve a report are:

	ONLINE	*BATCH*
Prepare	ROPEOPR	ROPEBPR
Findreport	ROPEOFR	ROPEBFR
Readline	ROPEORL	ROPEBRL
Readlinegroup	ROPEORG	ROPEBRG
Forwardspace	ROPEOFS	ROPEBFS
Backspace	ROPEOBS	ROPEBBS
Endread	ROPEOER	ROPEBER
Terminate	ROPEOTR	ROPEBTR

In general the following sequence of ROPES commands is used when retrieving a report:

Prepare (ROPEOPR, ROPEBPR)

Findreport (ROPEOFR, ROPEBFR)

Readline (ROPEORL, ROPEBRL)
-or-
Readlinegroup (ROPEORG, ROPEBRG)

-in combination with-

Forwardspace (ROPEOFS, ROPEBFS)
-and-
Backspace (ROPEOBS, ROPEBBS)

Endread (ROPEOER, ROPEBER)

Terminate (ROPEOTR, ROPEBTR)

- The Prepare command must be executed before other ROPES processing can occur. After the Prepare command is executed successfully, one or more reports may be retrieved by the same program execution.

- The Findreport command verifies that ROPES is available, that the report in the report name field is known to ROPES, and that the specified report is available for retrieval. The command also returns the number of lines in the report.

- The Readline command may be executed as many times as the number of lines in the report. ROPES indicates “end of report” (i.e., no more lines are available) by moving the value of QRCERPT into field QUEUERC. After this condition is indicated the application program must not issue an Endread command for the report. A Readline command, issued after the “end of report” is indicated, returns the first line of the report. A new report may be retrieved after the “end of report” is indicated by changing the report name and using the Readline command to read the new report.

- The Readlinegroup command is used to retrieve several lines at a time from a report into the line group area (named ROPESLGA). ROPES indicates “end of report” by moving the value of QRCERPT into field QUEUERC. When “end of report” is detected, ROPES returns the count of lines in the line group area before the “end of report” to the application. After this condition is indicated, the application program must not issue an Endread command for the report. A Readlinegroup command issued after the “end of report” is indicated returns the first lines of the report. A new report may be retrieved after the “end of report” is indicated by changing the report name and using the Readlinegroup command to read the new report lines.

- The Forwardspace command is used to move forward through a report by a user-specified number of lines. The content of the last line spaced over is returned in the report line area which was specified (or defaulted) in the Prepare command. If the last line of the report is reached before the forward space operation is complete, ROPES indicates this “end of report” condition by moving the value of QRCERPT into field QUEUERC. ROPES also returns the number of lines which could not be spaced over. After the “end of report” is indicated, the application must not issue an Endread command for the report. A Forwardspace command issued after the “end of report” is indicated begins spacing from the first line of the report. Spacing may be performed over a new report after the “end of report” is indicated by changing the report name and using the Forwardspace command to pass the lines of the new report from first to last.

- The Backspace command is used to move backward through a report by a user-specified number of lines. The contents of the last line spaced over is returned in the report line area which was specified (or defaulted) in the Prepare command. If the first line of the report is reached before the backspace operation is complete, ROPES indicates this “end of report” condition by moving the value of QRCERPT into field QUEUERC. ROPES also returns the number of lines which could not be spaced over. After the “end of report” is indicated, the application must not issue an Endread command for the report. A Backspace command issued after the "end of report" is indicated begins spacing from the last line of the report. Spacing may be performed over a new report after the "end of report" is indicated by changing the report name and using the Backspace command to pass the lines of the new report from last to first.

- The Endread command must be executed if retrieval of report lines is to be interrupted before the "end of report" condition is indicated (response code QRCERPT in the QUEUERC field). A new report may be retrieved after the Endread command is executed by changing the report name and using the retrieval commands to read the new report lines.

- The Terminate command is executed to indicate termination of ROPES processing by the application program.

General Examples of Report Retrieval Flow

Online COBOL - Line-at-a-time Retrieval

```
               .
               .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
               .
               .
 PROCEDURE DIVISION.
               .
               .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE. COPY ROPEOPR.
               .
               .
* READ A REPORT LINE
 ROPES-READLINE. COPY ROPEORL.
               .
* PROCESS THE ROPES REPORT LINE
               .
* CONTINUE PROCESSING THE REPORT
     GO TO ROPES-READLINE.
               .
               .
 ERR-PARA.
     IF QUEUERC NOT = QRCERPT GO TO FATAL-
ERROR.
* HERE WE KNOW ALL THE REPORT LINES WERE
PROCESSED
               .
               .
 ROPES-TERMINATE. COPY ROPEOTR.
               .
               .
```

Batch COBOL - Line-at-a-time Retrieval

```
               .
               .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
               .
               .
 PROCEDURE DIVISION.
               .
               .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE. COPY ROPEBPR.
               .
               .
* READ A REPORT LINE
 ROPES-READLINE. COPY ROPEBRL.
               .
               .
* PROCESS THE ROPES REPORT LINE
               .
               .
* CONTINUE PROCESSING THE REPORT
     GO TO ROPES-READLINE.
               .
               .
 ERR-PARA.
     IF QUEUERC NOT = QRCERPT GO TO FATAL-
ERROR.
* HERE WE KNOW ALL THE REPORT LINES WERE
PROCESSED
               .
               .
 ROPES-TERMINATE. COPY ROPEBTR.
               .
               .
```

Online PL/I - Line-at-a-time Retrieval

```
               .
               .
 %INCLUDE ROPESPRM;
               .
               .
 %INCLUDE ROPEOPR;   /* PREPARE */
               .
               .
 /* PROCESSING LOOP */
               .
               .
 %INCLUDE ROPEORL;    /* READ REPORT LINE */
               .
               .
 /* PROCESS THE REPORT LINE */
               .
               .
 /* PERFORM LOOP UNTIL ROPES RETURN */
 /* CODE IS QRCERPT WHICH INDICATES END */
 /* OF LINES FOR REPORT */
               .
               .
 ERR_PARA:
     IF QUEUERC = QRCERPT THEN DO;
     % INCLUDE ROPEOTR;
     END;
     ELSE...       /* PROCESS FATAL ERROR */
               .
               .
```

Batch PL/I - Line-at-a-time Retrieval

```
               .
               .
 %INCLUDE ROPESPRM;
               .
               .
 %INCLUDE ROPEBPR;  /* PREPARE PROCESSING */
               .
               .
 /* PROCESSING LOOP */
               .
               .
 %INCLUDE ROPEBRL;     /* READ REPORT LINE */
               .
               .
 /* PROCESS THE REPORT LINE */
               .
               .
 /* PERFORM PROCESSING LOOP UNTIL ROPES */
 /* RETURN CODE IS QRCERPT INDICATING */
 /* END OF LINES FOR REPORT */
```

```
               .
               .
 ERR_PARA:
     IF QUEUERC = QRCERPT THEN DO;
     %INCLUDE ROPEBTR;
     END;
     ELSE...        /* PROCESS FATAL ERROR */
               .
               .
```

Online Assembler - Line-at-a-time Retrieval

```
               .
DFHEISTG DSECT
         ROPESPRM
               .
               .
MYPROGM  CSECT
               .
               .
         ROPEOPR       PREPARE FOR PROCESSING
               .
READREPT EQU   *        REPORT RETRIEVAL LOOP
               .
         ROPEORL        READ A REPORT LINE
               .
               .
* PROCESS THE REPORT LINE
               .
         B     READREPT    CONTINUE READING
               .
ERRPARA  EQU   *
* IF THE RETURN CODE INDICATES THAT ALL THE
* LINES IN THE REPORT ARE PROCESSED CLOSE THE
* REPORT, ELSE PROCESS THE FATAL ERROR
         CLC   QUEUERC,QRCERPT REPORT
PROCESSED OK?
         BNE   BADERR      NO, PROCESS ERROR
         ROPEOTR           END OF PROCESSING
               .
               .
```

Batch Assembler - Line-at-a-time Retrieval

```
MYPROGM  CSECT
                 .
                 .
         ROPEBPR        PREPARE PROCESSING
                 .
                 .
READREPT EQU      *     REPORT RETRIEVAL LOOP
                 .
                 .
         ROPEBRL        READ A REPORT LINE
                  .
* PROCESS THE REPORT LINE
                  .
         B     READREPT      CONTINUE READING
                  .
ERRPARA  EQU   *
* IF THE RETURN CODE INDICATES THAT ALL THE
* LINES IN THE REPORT ARE PROCESSED CLOSE THE
* REPORT, ELSE PROCESS THE FATAL ERROR
         CLC   QUEUERC,QRCERPT  PROCESSED OK?
         BNE   BADERR       NO, PROCESS ERROR
         ROPEBTR            END OF PROCESSING
                  .
         ROPESPRM       COMMUNICATION AREA
```

Online COBOL - Line-group Retrieval

```
                 .
                 .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
 01 ROPESLGA COPY ROPESLGA
                  REPLACING LINE-COUNT BY 22.
* WE RETRIEVE UP TO 22 LINES AT ONCE
                 .
                 .
 PROCEDURE DIVISION.
                 .
                 .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE. COPY ROPEOPR REPLACING
                    GROUP-AREA BY ROPESLGA.
* USE THE LINE GROUP AREA CALLED ROPESLGA
                 .
* READ REPORT LINES (UP TO 22 AT ONCE)
     MOVE 22 TO ROPESLGA-LINE-COUNT.
 ROPES-READGRP. COPY ROPEORG REPLACING
                    ERR-PARA BY RG-ERROR.
 RGEOR.
* RG-ERROR COMES HERE FOR END OF REPORT
* PROCESS THE ROPES REPORT LINES
* ROPESLGA-VALID-LINES HAS COUNT OF
* VALID LINES (MAY BE 0)
                 .
     MOVE ROPESLGA-LINE (INDEX) TO WORK-LINE.
                 .
* CONTINUE PROCESSING REPORT IF MORE LINES
     IF QUEUERC NOT = QRCERPT GO TO ROPES-
                      READGRP.
* HERE WE KNOW ALL LINES HAVE BEEN PROCESSED.
                 .
                 .
 ROPES-TERMINATE. COPY ROPEOTR.
                 .
                 .
 ERR-PARA.
* HERE IF SERIOUS ROPES ERROR
                 .
                 .
 RG-ERROR.
* HERE IF ERROR ON ROPEORG
* IF 'END OF REPORT' PROCESS LAST LINES
     IF QUEUERC = QRCERPT GO TO RGEOR.
* HERE IF SERIOUS ERROR ON ROPEORG.
                 .
```

Batch COBOL - Line-group Retrieval

```
                 .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
 01 ROPESLGA  COPY ROPESLGA
                REPLACING LINE-COUNT BY 50.
* WE RETRIEVE UP TO 50 LINES AT ONCE
                 .
 PROCEDURE DIVISION.
                 .
* PREPARE FOR ROPES PROCESSING
 ROPES-PREPARE. COPY ROPEBPR REPLACING
                    GROUP-AREA BY ROPESLGA.
* USE THE LINE GROUP AREA CALLED ROPESLGA
                 .
* READ REPORT LINES (UP TO 50 AT ONCE)
     MOVE 50 TO ROPESLGA-LINE-COUNT.
 ROPES-READGRP. COPY ROPEBRG REPLACING
                    ERR-PARA BY RG-ERROR.
 RGEOR.
* RG-ERROR COMES HERE FOR END OF REPORT
* PROCESS THE ROPES LINES
* ROPESLGA-VALID-LINES HAS COUNT OF VALID
*           LINES (MAY BE ZERO)
                 .
     MOVE ROPESLGA-LINE (INDEX) TO WORK-LINE.
                 .
* CONTINUE PROCESSING THE REPORT IF MORE
LINES
     IF QUEUERC NOT = QRCERPT GO TO ROPES-
                      READGRP.
* HERE WE KNOW ALL LINES HAVE BEEN PROCESSED
                 .
 ROPES-TERMINATE. COPY ROPEOTR.
                 .
 ERR-PARA.
* HERE IF SERIOUS ROPES ERROR
                 .
 RG-ERROR.
```

```
* HERE IF ERROR ON ROPEBRG
* IF 'END OF REPORT' PROCESS LAST LINES
     IF QUEUERC = QRCERPT GO TO RGEOR.
* HERE IF SERIOUS ERROR
```

Online PL/I - Line-group Retrieval

```
              .
 %INCLUDE ROPESPRM;
 %DECLARE LINE_COUNT CHAR;
 %LINE_COUNT = '50'; /* READ 50 LINES */
 %INCLUDE ROPESLGA;
              .
 %DECLARE GROUP_AREA CHAR;
 %GROUP_AREA = 'ROPESLGA';
 /* USE THE LINE GROUP AREA */
 %INCLUDE ROPEOPR;  /* PREPARE PROCESSING */
              .
 /* PROCESSING LOOP */
              .
 MOVE 50 TO ROPESLGA_LINE_COUNT;
 %DECLARE ERR_PARA CHAR;
 %ERR_PARA = 'RG_ERROR';
 READ_RG:
     %INCLUDE ROPEORG;      /* READ LINES */
 RGEOR: /* RG_ERROR HERE ON END OF REPORT */
              .
 /* PROCESS THE REPORT LINES */
 /* ROPESLGA_VALID_LINES HAS LINE COUNT */
              .
 WORK_LINE = ROPESLGA_LINE (LINE_INDEX);
 /* PERFORM PROCESSING LOOP UNTIL ROPES */
 /* RETURN CODE IS QRCERPT INDICATING */
 /* END OF LINES FOR REPORT */
 IF QUEUERC NOT = QRCERPT GO TO READ_RG;
              .
 %ERR_PARA = 'ERR_PARA';
 %INCLUDE ROPEOTR;
              .
 ERR_PARA:  /* HERE IF SERIOUS ROPES ERROR */
              .
 RG_ERROR:  /* HERE IF ERROR ON READGROUP */
 /* IF 'END OF REPORT' PROCESS LAST LINES */
 IF QUEUERC = QRCERPT GO TO RGEOR;
 /* HERE IF SERIOUS ERROR ON READGROUP
              .
```

Batch PL/I - Line-group Retrieval

```
              .
 %INCLUDE ROPESPRM;
 %DECLARE LINE_COUNT CHAR;
 %LINE-COUNT = '50'; /* WE READ 50 LINES */
 %INCLUDE ROPESLGA;
              .
 %DECLARE GROUP_AREA CHAR;
 %GROUP_AREA = 'ROPESLGA';
 /* USE THE LINE GROUP AREA */
 %INCLUDE ROPEBPR;   /* PREPARE PROCESSING */
              .
 /* PROCESSING LOOP */
              .
 MOVE 50 TO ROPESLGA_LINE_COUNT;
 %DECLARE ERR_PARA CHAR;
 %ERR_PARA = 'RG_ERROR';
 READ_RG:
 %INCLUDE ROPEBRG;      /* READ GROUP LINES */
 RGEOR: /* RG_ERROR HERE ON END OF REPORT */
              .
 /* PROCESS THE REPORT LINES */
 /* ROPESLGA_VALID_LINES HAS LINE COUNT */
              .
 WORK_LINE = ROPESLGA_LINE(INDEX);
              .
 /* PERFORM PROCESSING LOOP UNTIL ROPES */
 /* RETURN CODE IS QRCERPT INDICATING */
 /* END OF LINES FOR REPORT */
 IF QUEUERC NOT = QRCERPT GO TO READ_RG;
              .
 %ERR_PARA= 'ERR_PARA';
 %INCLUDE ROPEBTR;
              .
 ERR_PARA:  /* HERE IF SERIOUS ROPES ERROR */
              .
 RG_ERROR:   /* HERE IF ERROR ON READGROUP */
 /* IF 'END OF REPORT' PROCESS LAST LINES */
 IF QUEUERC = QRCERPT GO TO RGEOR;
 /* HERE IF SERIOUS ERROR ON READGROUP */
              .
```

Online Assembler - Line-group Retrieval

```
DFHEISTG DSECT
         ROPESPRM
         ROPESLGA LINECNT=50     50 LINES
               .
MYPROGM  CSECT
         ROPEOPR GRPAREA=ROPESLGA   PREPARE
* ROPES USING THE LINE GROUP AREA
               .
READREPT EQU   *         REPORT RETRIEVAL LOOP
               .
         MVC   RLGALCNT,=F'50'  READ 50 LINES
         ROPEORG ERRPARA=RGERROR  READ REPORT
RGEOR    DS    0H               HERE AT E.O.R.
* RLGAVLIN HAS COUNT OF VALID LINES
               .
* PROCESS REPORT LINES UNTIL END OF REPORT
               .
         CLI   QUEUERC,QRCERPT END OF REPORT?
         BNE   READREPT         NO, CONTINUE
               .
         L     R1,INDEX         GET LINE INDEX
         BCTR  R1,0             COMPUTE OFFSET
         MH    R1,H'255'        TO THE LINE
         MVC   WORKLINE,0(R1)   MOVE LINE
               .
TRMROPES DS    0H
         ROPEOTR          END OF PROCESSING
               .
ERRPARA  DS    0H
* HERE IF SERIOUS ROPES ERROR
               .
RGERROR  DS    0H
* HERE IF READGROUP ERROR
* IF 'END OF REPORT', PROCESS LAST LINES
          CLI  QUEUERC,QRCERPT    IS IT EOR?
          BE   RGEOR              YES, BRANCH
* HERE IF SERIOUS RG ERROR
```

Batch Assembler - Line-group Retrieval

```
MYPROGM  CSECT
                .
         ROPEBPR GRPAREA=ROPESLGA   PREPARE
* USING THE LINE GROUP AREA
                .
READREPT EQU   *         REPORT RETRIEVAL LOOP
                .
         MVC   RLGALCNT,=F'50'  READ 50 LINES
         ROPEBRG ERRPARA=RGERROR  READ REPORT
RGEOR    DS    0H               HERE AT E.O.R.
* RLGAVLIN HAS COUNT OF VALID LINES
                .
* PROCESS THE REPORT LINES UNTIL END OF
REPORT
         L     R1,INDEX         GET LINE INDEX
         BCTR  R1,0             COMPUTE OFFSET
         MH    R1,H'255'        TO THE LINE
         MVC   WORKLINE,0(R1)   MOVE LINE
                .
         CLI   QUEUERC,QRCERPT END OF REPORT?
         BNE   READREPT         NO, CONTINUE
                .
TRMROPES DS    0H
         ROPEBTR           END OF PROCESSING
                .
* HERE IF SERIOUS ROPES ERROR
                .
```

```
RGERROR  DS    0H
* HERE IF READGROUP ERROR
* IF 'END OF REPORT', PROCESS LAST LINES
         CLI   QUEUERC,QRCERPT    IS IT EOR?
         BE    RGEOR              YES, BRANCH
* HERE IF SERIOUS RG ERROR
                .
         ROPESPRM
         ROPESLGA LINECNT=50     50 LINES
```

Online COBOL - Forwardspace and Line-group Retrieval

```
                .
 WORKING-STORAGE SECTION.
 01 ROPESPRM  COPY ROPESPRM.
 01 ROPESLGA  COPY ROPESLGA
                  REPLACING LINE-COUNT BY 22.
* WE RETRIEVE UP TO 22 LINES AT ONCE
                .
 PROCEDURE DIVISION.
                .
 ROPES-PREPARE. COPY ROPEOPR REPLACING
                     GROUP-AREA BY ROPESLGA.
* WE WILL FORWARDSPACE 500 LINES
     MOVE 500 TO ROPESLGA-SPACE-COUNT.
 ROPES-FSPACE.  COPY ROPEOFS REPLACING
                     ERR-PARA BY FS-ERROR.
* IF IT WORKED, WE READ STARTING AT LINE 501.
* FORWARDSPACE USES LINE-AREA IN ROPESPRM
     MOVE 22 TO ROPESLGA-LINE-COUNT.
* READ REPORT LINES (UP TO 22 AT ONCE)
 ROPES-READGRP. COPY ROPEORG REPLACING
                     ERR-PARA BY RG-ERROR.
 RGEOR.
* RGERROR COMES HERE FOR END OF REPORT
* PROCESS THE ROPES REPORT LINES
* ROPESLGA-VALID-LINES HAS LINE COUNT
* (MAY BE 0)
                .
 ROPES-TERMINATE. COPY ROPEOTR.
                .
 ERR-PARA.
* HERE IF SERIOUS ROPES ERROR
                .
 RG-ERROR.
* IF 'END OF REPORT' PROCESS LAST LINES.
     IF QUEUERC = QRCERPT GO TO RGEOR.
* HERE IF SERIOUS ERROR ON ROPEORG.
                .
 FS-ERROR.
* IF QUEUERC=QRCERPT, THE REPORT HAS LESS
* THAN 500 * LINES
     IF QUEUERC NOT = QRCERPT, GO TO FS-ERRX.
* HERE IF EARLY 'END OF REPORT'.  NOTE THAT
* ROPESLGA-SPACE-REMAINDER CONTAINS THE COUNT
* OF LINES THAT COULD NOT BE SPACED OVER.
                .
 FS-ERRX.
* HERE IF SERIOUS ERROR ON ROPEOFS
```

Online Assembler - Backspace and Line-group Retrieval

```
DFHEISTG DSECT
         ROPESPRM
         ROPESLGA LINECNT=22 LINES AT ONCE
                .
MYPROGM  CSECT
                .
         ROPEOPR GRPAREA=ROPESLGA     PREPARE
* USING THE LINE GROUP AREA
                .
* BACKSPACE 22 LINES FROM END OF REPORT
         MVC   RLGASCNT,=F'22'  SHOW SPACE 22
         ROPEOBS ERRPARA=BSERROR BACKSPACE 22
BSEOR    DS    0H   BSERROR COMES HERE IF EOR
* WE ARE SET TO READ LAST LINES (UP TO 22)
* NOTE BACKSPACE USED LINAREA OF ROPESPRM
         MVC   RLGALCNT,=F'22' READ MAX OF 22
         ROPEORG ERRPARA=RGERROR TRY TO READ
RGEOR    DS    0H   RGERROR COMES HERE IF EOR
* RLGAVLIN HAS LINE COUNT IN LGA (MAY BE 0)
*        PROCESS THESE LINES
                .
         ROPEOTR       DISCONNECT FROM ROPES
                .
ERRPARA  DS    0H      HERE IF SERIOUS ERROR
                .
BSERROR  DS    0H      ERROR DURING BACKSPACE
* IF QUEUERC=QRCERPT, REPORT HAS LESS THAN 22
* LINES.
* RLGASREM HAS NUMBER OF LINES NOT BE SPACED.
         CLI   QUEUERC,QRCERPT    IS IT EOR?
         BE    BSEOR    YES, READ WHAT WE CAN
* HERE IF REAL ERROR
                .
RGERROR  DS    0H  HERE IF ERROR ON READGROUP
* IF END OF REPORT, WE HAVE LESS THAN 22
* LINES. USE THEM
         CLI   QUEUERC,QRCERPT  IS IT EOR?
         BE    RGEOR         YES,PROCESS LINES
* HERE IF REAL ERROR
                .
```

Programming Considerations

Concurrent Batch and Online Operations

Concurrent batch and online ROPES operations are supported through use of the ROPES Alternate Facility.

A batch report generation application which might be executed when the online ROPES system is active should direct its output to the Alternate Facility data sets. The ROPES Alternate Facility will automatically process the batch data to ensure its proper availability in the primary ROPES data sets.

A batch application which will never be executed when the online ROPES system is active may direct its output to either the Alternate Facility data sets or the primary ROPES data sets. If the output is directed to the Alternate Facility, it will be automatically transferred to the primary ROPES data sets when ROPES and the Alternate Facility become active online.

Concurrent Batch Operations

ROPES supports concurrent batch operations by multiple tasks through the use of the operating system ENQueue or LOCK facility. The ENQ or LOCK is used to serialize batch access to ROPES resources beginning with the Prepare command through the Terminate command or until job step termination. This serialization will usually be unnoticeable. If you plan to install multiple copies of ROPES (a production version and a test version) you must assign each version of ROPES its own data sets. The ENQ mechanism must also be given a unique identifier in each system. This is done through the ROPES

Maintenance Transaction. Please refer to the Maintenance section of the Administrator's Guide for a discussion of the parameters on the Miscellaneous Controls screen.

Data Sets for Batch

Batch report generation/retrieval job steps must have the following two data sets defined:

ROPEQBR Contains the print data.

ROPERIB Contains the report controls.

The Alternate Facility data sets should be specified for ROPEQBR and ROPERIB if a batch report generation application might be executed when the online ROPES system is active.

The following JCL allocates the Alternate Facility data sets for use by your batch report generation application (assuming that you use the data sets which are distributed with ROPES):

```
//ROPERIB   DD   DSN=ROPES.RIBALTF.BASE,DISP=SHR
//ROPEQBR   DD   DSN=ROPES.QBRALTF.BASE,DISP=SHR
```

Report Queue Reorganization

Each installation is expected to establish a schedule for reorganizing the ROPEQBR data set. Since this reorganization can clear the queue (depending on the options specified), the use of the ROPES queue by application programming systems should take into account the installation's maintenance schedules and procedures.

Batch Support Modules and Linkages

The command level facility requires ROPES modules which must be link-edited into the application load module that will use them. The modules must be included in a way that will ensure that they will not be overlaid. They pass control through the subroutine call linkage. Therefore, Assembler language programmers must be aware that registers 0, 1, 14 and 15 may be modified when a ROPES command is issued. Additionally, other ROPES modules must be available for loading during execution.

Buffer/Page Numbering

If a report is defined for buffer numbering on the report maintenance menu, the first one to ten characters of each line recognized by ROPES as a top of page will be used to display the buffer number of the start of that page. The size of the number to display is set on the same maintenance panel. Therefore, any user data starting in position two through the last digit of the number will be overlaid by ROPES when printed. If buffer numbering is not specified, these characters are not overlaid by ROPES.

MASSINSERT Logic

To improve performance when generating large reports in on-line transactions, transferring large reports through the Alternate Facility or in retrieving large data sets from JES, you may instruct ROPES to use the MASSINSERT option when writing records to the ROPES queue. This option is selected on the Report Options screen of the ROPES Maintenance transaction, discussed in section 12 on page 234. The MASSINSERT option takes advantage of the special VSAM processing performed when records with sequential keys are being inserted into a KSDS. CICS processing of these requests will cause serialization on the ROPES Queue data set, which may have adverse effects on response time if you generate many large reports concurrently. If you generate only small reports, no benefit will be gained using this option.

Threadsafe Considerations

ROPES V10.0 begins the process of making the entire ROPES product eligible for execution in the OTE. In this release we have insured that all ROPES CICS modules are reentrant. Furthermore, we have insured that the transaction interface module, ROPEOSRV and all of the Axios Products supplied exits, are Threadsafe. ***It is your responsibility to insure that the user exits coded by your site are also Threadsafe.***

You should be aware of the fact that the ROPES interface program uses CICS commands that have not yet been made Threadsafe by IBM. These include, primarily, File Control commands. As a consequence, we strongly recommend that client sites wishing to run DB2 transactions that also use ROPES review their application structure with the goal of eliminating, as much as possible, the interspersing of ROPES calls and SQL, to minimize the number of Change Mode events that occur during transaction processing.

It is our intention to continue to review the ROPES modules and we intend that all ROPES CICS modules be eligible for execution in the OTE when this is available to non-DB2 transactions, and that any ROPES module invoked by a client application is Threadsafe. To this end, you are invited to scan the ROPES Online Modules using the IBM supplied Load Module Scanner and the filter tables provided in each release of CICS to review the suitability of this use in your environment. Over time, we expect more of the CICS Basic API to be made Threadsafe by IBM, and we will not introduce any new commands or features to ROPES that are not Threadsafe without clearly documenting this for you.

In Version 10 you will notice that the object module linkage editing for the ROPES software has been broken up to allow the ROPES CICS modules to be linked with the RENT option. The CSDPPT member provided has been supplemented with a CSDPPTTS member in which programs that are Threadsafe are marked with the appropriate concurrency option. It will now be possible for you to prevent storage overlays in ROPES modules if you run with RENTPGM=YES. If you are not already doing so with your own applications, use caution when switching to this environment.

Exceptional Conditions

Exceptional conditions may occur during the execution of a ROPES command. This is indicated by a response code other than QUEUEOK in the QUEUERC field in the ROPES Communication Area (see Appendix A on page 88).

The only response code (other than QUEUEOK) considered normal is QRCERPT, which indicates “end of report” during report retrieval. All other response codes indicate exceptional conditions and generally mean that report processing can not continue. A list of all the response codes is available later in this section.

Application programs should preserve the contents of the QUEUERC field when an exceptional condition occurs. Most of the error conditions can be easily corrected by modifying the application program or the ROPES controls affected. Other errors indicate hardware malfunctions, etc.

The following outlines some of the actions to take when errors occur:

- When generating a report, use the Delete report command (ROPEBDR or ROPEODR) if the application program detects an error or other logical condition which requires that the current report generation be terminated. This command will free the ROPES resources in use and will return the control information for the current report to the state that existed before the report generation began. Your program should not issue an Endlines command for the current report.

- When retrieving a report, use the Endread command (ROPEOER or ROPEBER) if the application program detects an error or other logical condition which requires that the current report retrieval be ended before the end of the report. This frees the ROPES resources in use. The Endread command should not be issued before the first retrieval command or after the “end of report” condition is detected. The “end of report” condition will automatically release the ROPES resources associated with the report retrieval.

Introduction To ROPES Command Usage

This chapter describes the rules governing the use of the ROPES commands. ROPES commands are distributed as sets of instructions residing in the appropriate programming language library (books for DOS, members for OS). ROPES commands contain all the necessary instructions to perform their function. Each command returns a response code indicating the status of command processing. A generalized routine is given control if the response code from ROPES indicates an exceptional condition. The application programmer is responsible for detailed error analysis. A list of the response codes and their meanings is provided in later in this section.

The ROPES Commands

The ROPES commands appear below in alphabetical order. For each command, the following is given:

- a short description of the command function;
- the command identifier for online and batch application programs;
- a list of the options in each of the three supported programming languages.

Checkpoint - Preserve Report Generation Status

This command is used to preserve (i.e., checkpoint) the ROPES system status.

COMMAND: ROPEOCK (online)
ROPEBCK (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Delete Report - Delete The Report Now In Process

This command requests that ROPES delete the report lines which this transaction or job step added to the current report. This command may not be issued after the Endlines command is issued for the report.

COMMAND: ROPEODR (online)
ROPEBDR (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Endlines - End Generation Of Report Lines

This command indicates that generation of report lines is complete.

COMMAND: ROPEOEL (online)
ROPEBEL (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Endread - End Retrieval Of Report Lines

This command is used to end retrieval of lines from a report before the "end of report" response code (QRCERPT) has been received.

COMMAND: ROPEOER (online)
ROPEBER (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Findreport - Find A Report

This command is used to determine whether or not a report can be accessed and the number of lines in the report. This command uses the line group area (ROPESLGA).

COMMAND: ROPEOFR (online)
ROPEBFR (batch)

OPTIONS:
Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Total Lines[1]
Assembler: RLGAVLIN
COBOL: ROPESLGA-VALID-LINES
PL/I: ROPESLGA_VALID_LINES

[1] When ROPES is available and the report is defined and available (i.e., QUEUERC = QUEUEOK), the command returns the number of lines stored in the report in this field. The value returned may be zero which indicates that the report is empty.

Backspace - Move Backward Through A Report

This command is used to backspace through a report during retrieval. The last line spaced over is retrieved into the line area specified (or defaulted) in the Prepare command. This command also uses the line group area.

COMMAND: ROPEOBS (online)
ROPEBBS (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Lines To Space [1]
Assembler: RLGASCNT
COBOL: ROPESLGA-SPACE-COUNT
PL/I: ROPESLGA_SPACE_COUNT

Lines Remaining [2]
Assembler: RLGASREM
COBOL: ROPESLGA-SPACE-REMAINDER
PL/I: ROPESLGA_SPACE_REMAINDER

[1] This field must be set to the number of lines to be spaced over before the command is issued.

[2] When Backspace indicates "end of report," it returns the count of lines which could not be spaced over in this field.

Forwardspace - Move Forward Through A Report

This command is used to forward space through a report during retrieval. The last line spaced over is retrieved into the line area specified (or defaulted) in the Prepare command. This command also uses the line group area.

COMMAND: ROPEOFS (online)
ROPEBFS (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Lines To Space [1]
Assembler: RLGASCNT
COBOL: ROPESLGA-SPACE-COUNT
PL/I: ROPESLGA_SPACE_COUNT

Lines Remaining [2]
Assembler: RLGASREM
COBOL: ROPESLGA-SPACE-REMAINDER
PL/I: ROPESLGA_SPACE_REMAINDER

[1] This field must be set to the number of lines to be spaced over before the command is issued.

[2] When Forwardspace indicates "end of report," it returns the count of lines which could not be spaced over in this field.

Prepare - Prepare For ROPES Processing

This command is used to prepare for ROPES processing. It must be executed by the application program before other ROPES processing can occur.

COMMAND: ROPEOPR (online)
ROPEBPR (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Report Name
Assembler: RPTNAME
COBOL: REPORT-NAME
PL/I: REPORT_NAME

Line Area
Assembler: LINAREA
COBOL: LINE-AREA
PL/I: LINE_AREA

Line Group Area [1]
Assembler: GRPAREA
COBOL: GROUP-AREA
PL/I: GROUP_AREA

[1] Change this default name to "ROPESLGA" if you use the Findreport, Sendlinegroup, Readlinegroup, Forwardspace or Backspace commands. Be sure to include the ROPES Line Group Area ("ROPESLGA") in your application program (see ROPES Line Group Area, on page 69.)

Readline - Read A Line From A Report

This command is used to retrieve a single report line from a specified report. The line is retrieved into the line area specified (or defaulted) in the Prepare command.

COMMAND: ROPEORL (online)
ROPEBRL (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Readlinegroup - Read A Group Of Lines From A Report

This command is used to retrieve a group of lines from a specified report. The lines are retrieved into the line group area (ROPESLGA).

COMMAND: ROPEORG (online)
ROPEBRG (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Maximum Number Of Lines To Read [1]
Assembler: RLGALCNT
COBOL: ROPESLGA-LINE-COUNT
PL/I: ROPESLGA_LINE_COUNT

Count Of Lines Read [2]
Assembler: RLGAVLIN
COBOL: ROPESLGA-VALID-LINES
PL/I: ROPESLGA_VALID_LINES

[1] This field must be set to the maximum number of lines to be retrieved by a single Readlinegroup command before the command is issued. This number must not exceed the number of line entries specified for ROPESLGA.

[2] When Readlinegroup completes normally (QUEUERC = QUEUEOK) or indicates "end of report" (QUEUERC = QRCERPT), it returns the count of valid lines in this field. This count may be zero which indicates immediate "end of report".

Readonlyprepare - Prepare For ROPES Input Processing

This command is used to prepare for ROPES processing. It must be executed by the application program before other ROPES processing can occur. ROPES processing will be restricted to input functions only; no data storage or deletion will be permitted.

COMMAND: ROPEOOP (online)
ROPEBOP (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Report Name
Assembler: RPTNAME
COBOL: REPORT-NAME
PL/I: REPORT_NAME

Line Area
Assembler: LINAREA
COBOL: LINE-AREA
PL/I: LINE_AREA

Line Group Area [1]
Assembler: GRPAREA
COBOL: GROUP-AREA
PL/I: GROUP_AREA

[1] Change this default name to "ROPESLGA" if you use the Findreport, Sendlinegroup, Readlinegroup, Forwardspace or Backspace commands. Be sure to include the ROPES Line Group Area ("ROPESLGA") in your application program (see ROPES Line Group Area on page 69).

Resetreport - Clear All Data From A Report

This command is used to delete all data stored in a report. The data is purged from the report queue and is not archived.

COMMAND: ROPEORR (online)
ROPEBRR (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Sendline - Send A Report Line To ROPES

This command is used to add a single report line to a specified report.

COMMAND: ROPEOSL (online)
ROPEBSL (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Sendlinegroup - Send A Group Of Lines To A Report

This command is used to add a group of lines to a report. Optionally, it may also cause an Endlines command to be performed for the report. Before this command is issued, the lines to be added must be placed in the line group area (“ROPESLGA”) specified in the Prepare command.

COMMAND: ROPEOSG (online)
ROPEBSG (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

Endlines Request Flag [1]
Assembler: RLGAENDL
COBOL: ROPESLGA-END-LINES
PL/I: ROPESLGA_END_LINES

Line Count [2]
Assembler: RLGALCNT
COBOL: ROPESLGA-LINE-COUNT
PL/I: ROPESLGA_LINE_COUNT

[1] This field may be set to HIGH-VALUE (X’FF’) before the command is issued to indicate that these are the last lines to be sent and that an Endlines command is to be performed by ROPES after these lines have been added to the report. ROPES resets this field to LOW-VALUE (X’00’) to indicate successful Endlines processing.

[2] This field must be set to the number of lines to be added to the report by a particular Sendlinegroup command before the command is issued. This number must not exceed the number of line entries specified for ROPESLGA.

Terminate - Terminate ROPES Processing

This command is used to end ROPES processing by the application.

COMMAND: ROPEOTR (online),
ROPEBTR (batch)

OPTIONS:

Error Paragraph
Assembler: ERRPARA
COBOL: ERR-PARA
PL/I: ERR_PARA

ROPES Command Options

This chapter describes the options of the ROPES commands. The format and function of each option for each application programming language is given. The options are listed in alphabetical order.

COBOL: ERR-PARA
PL/I: ERR_PARA
Assembler: ERRPARA

This is the label of a routine which is to receive control whenever an exceptional condition is detected by ROPES during command processing. Note that this routine will receive control when a command results in an end-of-report-lines condition (response code QRCERPT).

COBOL: GROUP-AREA
PL/I: GROUP_AREA
Assembler: GRPAREA

This is the name of the ROPES line group area. It must be specified as “ROPESLGA” for the Prepare and Readonlyprepare command when the Findreport, Sendlinegroup, Readlinegroup, Forwardspace or Backspace commands will be used.

COBOL: LINE-AREA
PL/I: LINE_AREA
Assembler: LINAREA

This is the name of a 255 character area to be used to send or receive report lines (usually included in the ROPES Communication Area “ROPESPRM”).

COBOL: LINE-COUNT
PL/I: LINE_COUNT
Assembler: LINECNT

This is the number of lines to be processed in the largest single line group request. This parameter must be specified on the ROPESLGA data area definition.

COBOL: REPORT-NAME
PL/I: REPORT_NAME
Assembler: RPTNAME

This is the name of an 8 character area which will contain the report name (usually included in the ROPES Communication Area “ROPESPRM”).

ROPES Command Exceptional Conditions

The application program is expected to examine the ROPES response code after each command. The field QUEUERC, a field in the ROPES Communication Area (see Appendix A on page 88), is used by all applications.

The names listed are included in the ROPES Communication Area and are valid for COBOL, PL/I and Assembler programs.

The following list defines the ROPES response codes:

QUEUEOK (X’00’)
Request completed without errors.

QRCQCLS (C’0’)
The report was marked not available due to a prior unrecoverable error.

QRCNFND (C’1’)
The report name, as specified in the report name area, has not been defined to ROPES.

QRCMIDL (C’3’)
The application attempted to add a report line before issuing the PREPARE command.

QRCCOMP (C’4’)
The application attempted to add a report line to a report which was not empty and was not defined as an “appendable” report.

QRCNOTA (C’5’)
ROPES was not available (i.e. not active) when the request for services was made.

QRCQFUL (C’6’)
The ROPES Queue data set is full. No records can be added.

QRCIOPQ (C’7’)
An I/O error occurred while ROPES was accessing the Queue data set.

QRCIOCK (C’8’)
An I/O error occurred while ROPES was checkpointing system information.

QRCLOGR (C’9’)
An I/O error or format error was detected while processing a logging function (should be set by the user’s exit routine).

QRCINVF (C’A’)
An invalid command request code was detected.

QRCINVS (C’B’)
An invalid sequence of commands was detected, such as an Endlines command not proceeded by at least one Sendline command or a change in the report name used with Sendline without an Endlines for the prior report.

QRCNQBR (C’C’)
ROPES commands issued in illogical order.

QRCMQBR (C’D’)
ROPES commands issued in illogical order.

QRCINVL (C'E')
ROPES commands issued in illogical order.

QRCDELC (C'F')
In invalid Deletereport function request was made by the application program.

QRCRWIF (C'G')
An internal ROPES communication error was detected.

QRCRWIQ (C'H')
An internal ROPES input area identification error was detected.

QRCRWIW (C'I')
An internal ROPES output area identification error was detected.

QRCRWRA (C'J')
An internal ROPES record identification error was detected.

QRCRWCK (C'K')
An internal ROPES control area identification error was detected.

QRCCCNG (C'L')
An invalid carriage control character was found in a print line passed to ROPES.

QRCERPT (C'M')
The end of the report was reached during report retrieval.

QRCINVC (C'N')
The Readline command found a report buffer that can not be interpreted.

QRCTRNC (C'O')
The Readline command truncated a line found in an input QBR because the line was greater than 255 characters long (including the carriage control character).

QRCXTNT (C'P')
In the DOS version only, too many extents are defined for the ROPES QBR data set. This return code cannot occur in the MVS version.

QRCQREC (C'Q')
Unable to open the Queue data set during system initialization.

QRCRPCB (C'R')
Unable to open the Report control data set (ROPERIB) during initialization. This condition can occur for any of the following reasons:
1) the ACB for the ROPERIB data set could not be built;
2) the RPL for the ROPERIB data set could not be built;
3) the ACB for the ROPERIB data set could not be opened;
4) the Special Report Options record could not be read; or
5) the Miscellaneous Options record could not be read.

QRCRRNG (C'S')
In invalid buffer number was requested.

QRCTABL (C'T')
A ROPES system table could not be loaded.

QRCTWAU (C'U')
The TWA space required for ROPES macro level functions has not been defined in the PCT.

QRCRLGA (C'X')
For a Findreport, Readlinegroup, Sendlinegroup, Forwardspace, or Backspace command: the line group area was not properly specified in the Prepare command; RLGALCNT not > 0 on SG or RG; or RLGASCNT not > 0 on FS or BS.

Messages and Diagnostic Dumps

The messages and dumps associated with ROPES operations are described in a another section of this manual.

Appendix A. ROPES Communication Area

This appendix describes the fields in the ROPES Communication Area (ROPESPRM). An application program can access all of the fields in the ROPES Communication Area by name but must not change the contents of any of them except the LINE-AREA and REPORT-NAME fields. The contents and format of each field which is meaningful to the ROPES command level application programmer are given. Fields are listed in alphabetical order.

COBOL: LINE-AREA PIC X(255)
PL/I: LINE_AREA CHAR(255)
Assembler: LINAREA CL255

Contains the line used for report processing (report retrieval or report generation). The report line is composed of a carriage control character in the first position followed by 254 characters of text or blanks.

COBOL: LINE-CC PIC X
PL/I: LINE_CC CHAR(1)
Assembler: LINCC CL1

Contains the report line carriage control character. This field is part of LINE-AREA, described above. The ROPES carriage control characters are listed in Appendix C.

COBOL: LINE-DATAPIC X(254)
PL/I: LINE_DATA CHAR(254)
Assembler: LINDATA CL254

Contains the report line text or blanks. This field is part of LINE-AREA, described above.

COBOL: QUEUERC PIC X
PL/I: QUEUERC CHAR(1)
Assembler: QUEUERC CL1

Contains the response code set by ROPES when processing a command. A list of the response codes was given above.

COBOL: QUEUETR PIC X
PL/I: QUEUETR CHAR(1)
Assembler: QUEUETR CL1

Contains the ROPES command request code.

COBOL: REPORT-NAME PIC X(8)
PL/I: REPORT_NAME CHAR(8)
Assembler: RPTNAME CL8

Contains the ROPES report name.

COBOL: ROPESAIR PIC X(18)
PL/I: ROPESAIR CHAR(18)
Assembler: ROPESAIR CL18

Defines all valid ROPES command request codes.

COBOL: ROPESCC PIC X(26)
PL/I: ROPESCC CHAR(26)
Assembler: ROPESCC CL26

Defines the ROPES carriage control characters.

COBOL: ROPESRC PIC X(42)
PL/I: ROPESRC CHAR(42)
Assembler: ROPESRC CL42

Defines all ROPES command response codes. The meanings of individual response codes are explained above.

Appendix B. ROPES Line Group Area

This appendix describes the fields in the ROPES Line Group Area (ROPESLGA) which may be used by the application program.

COBOL: ROPESLGA-LINE-COUNT PIC S9(8) COMP
PL/I: ROPESLGA_LINE_COUNT FIXED BIN(31)
Assembler: RLGALCNTF

This full word must be set to the number of lines (>0) to be processed by a particular Sendlinegroup or Readlinegroup command before the command is issued. This number must not exceed the number of line entries specified in the ROPESLGA definition. ROPES does not change this field.

COBOL: ROPESLGA-SPACE-COUNT
PIC S9(8) COMP
PL/I: ROPESLGA_SPACE_COUNT
FIXED BIN(31)
Assembler: RLGASCNT F

This full word must be set to the number of lines (>0) to be spaced by a particular Forwardspace or Backspace command before the command is issued. ROPES does not change this field.

COBOL: ROPESLGA-SPACE-REMAINDER
PIC S9(8) COMP
PL/I: ROPESLGA_SPACE_REMAINDER
FIXED BIN(31)
Assembler: RLGASREMF

This full word is set by ROPES to the number of lines which could not be spaced during a Forwardspace or Backspace command.

COBOL: ROPESLGA-VALID-LINES
PIC S9(8) COMP
PL/I: ROPESLGA_VALID_LINES
FIXED BIN(31)
Assembler: RLGAVLIN F

This full word is set by ROPES to the number of valid lines in the line group area after a Readlinegroup command. It is set to the number of lines in the report after a successful Findreport command.

COBOL: ROPESLGA-END-LINES PIC X
PL/I: ROPESLGA_END_LINES
CHAR(1)
Assembler: RLGAENDLX

This one character field is used to cause ROPES to internally issue an Endlines command after all lines in a line group area have been sent to a report in response to the Sendlinegroup command. To activate this function, the user must set this field to HIGH-VALUE (X'FF') before issuing the Sendlinegroup command. ROPES resets this field to LOW-VALUE (X'00') to indicate successful internal Endlines processing.

COBOL: ROPESLGA-LINE-GROUP PIC nX(255)
PL/I: ROPESLGA_LINE_GROUP n CHAR (255)
Assembler: RLGALGRP nXL255

This is the name of the group of 255 character lines supported in this line group area. It consists of "n" line areas as defined below. "n" is determined by the line count specified on the ROPESLGA data area definition.

COBOL: ROPESLGA-LINE PIC X(255)
PL/I: ROPESLGA_LINE CHAR (255)
Assembler: RLGALINE CL255

This is the layout of an individual report line in the line group area. A report line is composed of a carriage control character in the first position followed by 254 characters of text or blanks.

COBOL: ROPESLGA-LINE-CC PIC X
PL/I: ROPESLGA_LINE_CCCHAR(1)
Assembler: RLGALNCC CL1

Contains the report line carriage control character. This field is part of ROPESLGA-LINE, described above. The ROPES carriage control characters are listed in Appendix C.

COBOL: ROPESLGA-LINE-DATA PIC X(254)
PL/I: ROPESLGA_LINE_DATA CHAR(254)
Assembler: RLGALNDA CL254

Contains the report line text or blanks. This field is part of ROPESLGA-LINE, described above.

Appendix C. Carriage Control Characters

The following table defines the ANSI carriage control characters accepted by ROPES and the symbols for each programming language. The carriage control characters are included in the ROPES Communication Area.

The proper initial alignment of the forms at the device is the user's responsibility. ROPES performs automatic pagination as required by the forms control block associated with each report.

The ROPES application programmer must coordinate with those responsible for maintaining the ROPES system so that the proper forms control blocks are used when printing the reports.

Carriage Control Table

COBOL Symbol	PL/I Symbol	Assembler Symbol	Control Character	Write After Advancing
SINGLE-SPACE	SINGLE_SPACE	SINGLE	(blank)	Single Space.
DOUBLE-SPACE	DOUBLE_SPACE	DOUBLE	0 (zero)	Double Space.
TRIPLE-SPACE	TRIPLE_SPACE	TRIPLE	- (hyphen)	Triple Space.
NO-SPACE	NO_SPACE	NOSPACE	+ (plus)	Suppress spacing.
CHANNEL-1	CHANNEL_1	CH1	1	Skip to channel 1 (top of form).
CHANNEL-2	CHANNEL_2	CH2	2	Skip to channel 2.
CHANNEL-3	CHANNEL_3	CH3	3	Skip to channel 3.
CHANNEL-4	CHANNEL_4	CH4	4	Skip to channel 4.
CHANNEL-5	CHANNEL_5	CH5	5	Skip to channel 5.
CHANNEL-6	CHANNEL_6	CH6	6	Skip to channel 6.
CHANNEL-7	CHANNEL_7	CH7	7	Skip to channel 7.
CHANNEL-8	CHANNEL_8	CH8	8	Skip to channel 8.
CHANNEL-9	CHANNEL_9	CH9	9	Skip to channel 9.
CHANNEL-10	CHANNEL_10	CH10	A	Skip to channel 10.
CHANNEL-11	CHANNEL_22	CH11	B	Skip to channel 11.
CHANNEL-12	CHANNEL_12	CH12	C	Skip to channel 12.
NO-TRANS	NO_TRANS	NOTRANS	X	Do not translate data on this line.
ACCT-CODE	ACCT_CODE	ACCTCD	Z	Use next 8 characters as the accounting code.

Appendix D. Sample Programs

This Appendix contains sample ROPES command level application programs written in ANS COBOL, PL/I and Assembler.

There are twelve sample programs:

Report Generation

- ANS COBOL online
- ANS COBOL batch
- PL/I online
- PL/I batch
- Assembler online
- Assembler batch

Report Retrieval

- ANS COBOL online
- ANS COBOL batch
- PL/I online
- PL/I batch
- Assembler online
- Assembler batch

Example of report generation by an online ANS COBOL program

```
IDENTIFICATION DIVISION.
PROGRAM-ID. COBSAMP.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77  C3            PIC X VALUE 'C'.
77  LEN-OKMSG     PIC S9(4) VALUE +25.
77  LEN-ERRMSG    PIC S9(4) VALUE +19.
01  ROPESPRM COPY ROPESPRM.
01  LINE1    PIC X(132) VALUE
             'TOP OF PAGE FROM SAMPLE PROGRAM'.
01  LINE2    PIC X(132) VALUE 'SINGLE SPACE'.
01  LINE3    PIC X(132) VALUE 'DOUBLE SPACE'.
01  LINE4    PIC X(132) VALUE 'DOUBLE SPACE'.
01  LINE5     PIC X(132) VALUE 'TRIPLE SPACE AND LAST
LINE'.
01  REPTOK   PIC X(25)  VALUE 'ROPES REPORT GENERATED
OK'.
01  ERRMSG.
    02 ERRCNST PIC X(18) VALUE 'ROPES ERROR CODE='.
    02 ERRCODE PIC X      VALUE ' '.
PROCEDURE DIVISION.
ROPES-PREPARE. COPY ROPEOPR.
    MOVE 'REPORTA' TO REPORT-NAME.
    MOVE CHANNEL-1 TO LINE-CC.
    MOVE LINE1 TO LINE-DATA.
SEND-FIRST-LINE. COPY ROPEOSL.
    MOVE SINGLE-SPACE TO LINE-CC.
    MOVE LINE2 TO LINE-DATA.
SEND-SECOND-LINE. COPY ROPEOSL.
    MOVE DOUBLE-SPACE TO LINE-CC.
    MOVE LINE3 TO LINE-DATA.
SEND-THIRD-LINE. COPY ROPEOSL.
    MOVE DOUBLE-SPACE TO LINE-CC.
    MOVE LINE4 TO LINE-DATA.
SEND-FOURTH-LINE. COPY ROPEOSL.
    MOVE TRIPLE-SPACE TO LINE-CC.
    MOVE LINE5 TO LINE-DATA.
SEND-FIFTH-LINE. COPY ROPEOSL.
END-REPORT-GENERATION. COPY ROPEOEL.
CHECKPOINT-THE-REPORT. COPY ROPEOCK.
TERMINATE-ROPES-PROCESSING. COPY ROPEOTR.
    EXEC CICS SEND FROM(REPTOK) LENGTH(LEN-OKMSG)
              ERASE CTLCHAR(C3) END-EXEC.
    GO TO PROGRAM-EXIT.
ERR-PARA.
    MOVE QUEUERC TO ERRCODE.
    EXEC CICS SEND FROM(ERRMSG) LENGTH(LEN-ERRMSG)
              ERASE CTLCHAR(C3) END-EXEC.
DELETE-BAD-REPORT. COPY ROPEODR REPLACING
                                ERR-PARA BY ROPES-
TERMINATE.
ROPES-TERMINATE.  COPY ROPEOTR  REPLACING
                               ERR-PARA BY PROGRAM-
EXIT.
PROGRAM-EXIT. EXEC CICS RETURN END-EXEC.
    STOP RUN.
```

Example of a report generation by a batch ANS COBOL program

```
IDENTIFICATION DIVISION.
PROGRAM-ID. COBSAMP.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  ROPESPRM COPY ROPESPRM.
01  LINE1     PIC X(132) VALUE
              'TOP OF PAGE FROM SAMPLE PROGRAM'.
01  LINE2     PIC X(132) VALUE 'SINGLE SPACE'.
01  LINE3     PIC X(132) VALUE 'DOUBLE SPACE'.
01  LINE4     PIC X(132) VALUE 'DOUBLE SPACE'.
01  LINE5      PIC X(132) VALUE 'TRIPLE SPACE AND LAST
LINE'.
01  REPTOK    PIC X(25) VALUE 'REPORT GENERATED OK'.
01  ERRMSG.
02  ERRCNST PIC X(18) VALUE 'ROPES ERROR CODE='.
02  ERRCODE PIC X     VALUE ' '.
PROCEDURE DIVISION.
ROPES-PREPARE. COPY ROPEBPR.
    MOVE 'REPORTA' TO REPORT-NAME.
    MOVE CHANNEL-1 TO LINE-CC.
    MOVE LINE1 TO LINE-DATA.
SEND-FIRST-LINE. COPY ROPEBSL.
    MOVE SINGLE-SPACE TO LINE-CC.
    MOVE LINE2 TO LINE-DATA.
SEND-SECOND-LINE. COPY ROPEBSL.
    MOVE DOUBLE-SPACE TO LINE-CC.
    MOVE LINE3 TO LINE-DATA.
SEND-THIRD-LINE. COPY ROPEBSL.
    MOVE DOUBLE-SPACE TO LINE-CC.
    MOVE LINE4 TO LINE-DATA.
SEND-FOURTH-LINE. COPY ROPEBSL.
    MOVE TRIPLE-SPACE TO LINE-CC.
    MOVE LINE5 TO LINE-DATA.
SEND-FIFTH-LINE. COPY ROPEBSL.
END-REPORT-GENERATION. COPY ROPEBEL.
CHECKPOINT-THE-REPORT. COPY ROPEBCK.
TERMINATE-ROPES-PROCESSING. COPY ROPEBTR.
    DISPLAY REPTOK.
    GO TO PROGRAM-EXIT.
ERR-PARA. MOVE QUEUERC TO ERRCODE.
          DISPLAY ERRMSG.
DELETE-BAD-REPORT. COPY ROPEBDR REPLACING
                   ERR-PARA BY ROPES-TERMINATE.
ROPES-TERMINATE. COPY ROPEBTR REPLACING
                      ERR-PARA BY PROGRAM-EXIT.
PROGRAM-EXIT.      STOP RUN.
```

Example of report generation by an online PL/I application program

```
PLISAMP:     PROC OPTIONS(MAIN);
    %INCLUDE ROPESPRM;
    DCL C3 CHAR(1) INIT('C');
      DCL  LINE1  CHAR(132)  INIT('TOP  OF  PAGE  FROM
SAMPLE PROGRAM');
    DCL LINE2 CHAR(132) INIT('SINGLE SPACE');
    DCL LINE3 CHAR(132) INIT('DOUBLE SPACE');
    DCL LINE4 CHAR(132) INIT('DOUBLE SPACE');
   DCL LINE5 CHAR(132) INIT('TRIPLE SPACE AND LAST
LINE');
        DCL  REPTOK  CHAR(25)  INIT('ROPES  REPORT
GENERATED OK');
    DCL 1 ERRMSG,
    2 ERRCNST CHAR(18) INIT('ROPES ERROR CODE='),
    2 ERRCODE CHAR(1) INIT(' ');
    DCL LEN_OKMSG BIN FIXED INIT(25);
    DCL LEN_ERRMSG BIN FIXED INIT(19);
START_PROCESS:
```

```
    %INCLUDE ROPEOPR; /*PREPARE ROPES PROCESSING */
    REPORT_NAME = 'REPORTA'; /* SET REPORT NAME */
    LINE_CC = CHANNEL-1; /* TOP OF PAGE CC */
    LINE_DATA = LINE1;     /* LINE DATA */
    %INCLUDE ROPEOSL;      /* SEND LINE TO ROPES */
    LINE_CC = SINGLE_SPACE; /* SINGLE SPACE CC */
    LINE_DATA = LINE2;     /* LINE DATA */
    %INCLUDE ROPEOSL;      /* SEND LINE TO ROPES */
    LINE_CC = DOUBLE_SPACE; /* DOUBLE SPACE CC */
    LINE_DATA = LINE3;      /* LINE DATA */
    %INCLUDE ROPEOSL;      /* SEND LINE TO ROPES */
    LINE_CC = DOUBLE_SPACE;   /* DOUBLE SPACE CC */
    LINE_DATA = LINE4;     /* LINE DATA */
    %INCLUDE ROPEOSL;      /* SEND LINE TO ROPES */
    LINE_CC = TRIPLE_SPACE;    /* TRIPLE SPACE */
    LINE_DATA = LINE5;             /* LINE DATA */
    %INCLUDE ROPEOSL;      /* SEND LINE TO ROPES */
    %INCLUDE ROPEOEL;   /* END REPORT GENERATION */
    %INCLUDE ROPEOCK;   /* CHECKPOINT THE REPORT */
    %INCLUDE ROPEOTR;          /* TERMINATE ROPES */
    EXEC CICS SEND FROM(REPTOK) LENGTH(LEN_OKMSG)
            ERASE CTLCHAR(C3);
    GO TO PROGRAM_EXIT;
ERR_PARA:  ERRCODE = QUEUERC;
    EXEC CICS SEND FROM(ERRMSG) LENGTH(LEN_ERRMSG)
           ERASE CTLCHAR(C3);
    %DECLARE ERR_PARA CHAR;
    %ERR_PARA ='TERM_ROPES'; /* AVOID ERROR LOOP */
    %INCLUDE ROPEODR;       /* DELETE BAD REPORT */
    %ERR_PARA='PROGRAM_EXIT'; /* AVOID LOOP */
TERM_ROPES: %INCLUDE ROPEOTR: /* TERMINATE ROPES */
PROGRAM_EXIT:
    EXEC CICS RETURN;
END PLISAMP;
```

Example of report generation by a batch PL/I program

```
PLISAMP:  PROC OPTIONS(MAIN);
    %INCLUDE ROPESPRM;
    DCL LINE1 CHAR(132)
        INIT('TOP OF PAGE FROM SAMPLE PROGRAM');
    DCL LINE2 CHAR(132) INIT('SINGLE SPACE');
    DCL LINE3 CHAR(132) INIT('DOUBLE SPACE');
    DCL LINE4 CHAR(132) INIT('DOUBLE SPACE');
    DCL LINE5 CHAR(132)
        INIT('TRIPLE SPACE AND LAST LINE');
    DCL REPTOK CHAR(25)
        INIT('ROPES REPORT GENERATED OK');
    DCL 1 ERRMSG,
          2 ERRCNST CHAR(18)
            INIT('ROPES ERROR CODE='),
          2 ERRCODE CHAR(1) INIT(' ');
START_PROCESS:
    %INCLUDE ROPEBPR;          /* PREPARE ROPES */
    REPORT_NAME = 'REPORTA';  /* SET REPORT NAME */
    LINE_CC = CHANNEL_1;       /* TOP OF PAGE CC */
    LINE_DATA = LINE1;         /* LINE DATA */
    %INCLUDE ROPEBSL;      /* SEND LINE TO ROPES */
    LINE_CC = SINGLE_SPACE;   /* SINGLE SPACE CC */
    LINE_DATA = LINE2;         /* LINE DATA */
    %INCLUDE ROPEBSL;      /* SEND LINE TO ROPES */
    LINE_CC = DOUBLE_SPACE;   /* DOUBLE SPACE CC */
    LINE_DATA = LINE3;         /* LINE DATA */
    %INCLUDE ROPEBSL;      /* SEND LINE TO ROPES */
    LINE_CC = DOUBLE_SPACE;   /* DOUBLE SPACE CC */
    LINE_DATA = LINE4;         /* LINE DATA */
    %INCLUDE ROPEBSL;      /* SEND LINE TO ROPES */
    LINE_CC = TRIPLE_SPACE;   /* TRIPLE SPACE */
    LINE_DATA = LINE5;         /* LINE DATA */
    %INCLUDE ROPEBSL;      /* SEND LINE TO ROPES */
    %INCLUDE ROPEBEL;    /* END REPORT GENERATION*/
    %INCLUDE ROPEBCK;   /* CHECKPOINT THE REPORT */
    %INCLUDE ROPEBTR;          /* TERMINATE ROPES */
    DISPLAY (REPTOK);          /* REPORT OK MSG */
    GO TO PROGRAM_EXIT      /* RETURN TO CALLER */
ERR_PARA:
    ERRCODE = QUEUERC;
    DISPLAY (ERRMSG);
    %DECLARE ERR_PARA CHAR;
    %ERR_PARA='TERM_ROPES'; /* AVOID ERROR LOOP */
    %INCLUDE ROPEBDR;       /* DELETE BAD REPORT */
    %ERR_PARA='PROGRAM_EXIT'; /* AVOID LOOP */
TERM_ROPES:
    %INCLUDE ROPEBTR;         /* TERMINATE ROPES */
PROGRAM_EXIT:
    END PLISAMP;
```

Example of report generation by an online Assembler program.

```
DFHEISTG  DSECT
          ROPESPRM
ERRMSG    DS  0CL19                 ERROR MESSAGE
ERRCNST   DS  CL18                 ERROR CONSTANT
ERCODE    DS  CL1             ROPES RESPONSE CODE
ASMSAMP   CSECT
          ROPEOPR                   PREPARE ROPES
          MVC RPTNAME,=CL8'REPORTA' SET REPORT NAME
          MVC LINC,CH1         TOP OF PAGE CC (CH1)
          MVC LINDATA,LINE1             SET LINE DATA
          ROPEOSL                SEND LINE TO ROPES
          MVC LINCC,SINGLE          SINGLE SPACE CC
          MVC LINDATA,LINE2              LINE DATA
          ROPEOSL                SEND LINE TO ROPES
          MVC LINCC,DOUBLE          DOUBLE SPACE CC
          MVC LINDATA,LINE3              LINE DATA
          ROPEOSL                SEND LINE TO ROPES
          MVC LINCC,DOUBLE          DOUBLE SPACE CC
          MVC LINDATA,LINE4             LINE DATA
          ROPEOSL                SEND LINE TO ROPES
          MVC LINCC,TRIPLE          TRIPLE SPACE CC
          MVC LINDATA,LINE5             LINE DATA
          ROPEOSL                SEND LINE TO ROPES
          ROPEOEL           ND OF REPORT GENERATION
          ROPEOCK              CHECKPOINT THE REPORT
          ROPEOTR            END OF ROPES PROCESSING
     EXEC CICS SEND FROM(REPTOK) ERASE CTLCHAR(C3)
          B  PRGEXIT
ERRPARA   MVC ERRCODE,QUEUERC          MOVE RESPONSE
          MVC ERRCNST,ERRCONST   MOVE ERROR MESSAGE
               EXEC  CICS  SEND  FROM(ERRMSG)  ERASE
CTLCHAR(C3)
          ROPEODR ERRPARA=TERMNAT      DELETE REPORT
TERMNAT   DS  0H
          EXEC CICS RETURN
LINE1        DC    CL132'TOP  OF  PAGE  FROM  SAMPLE
PROGRAM'
LINE2      DC  CL132'SINGLE SPACE'
LINE3      DC  CL132'DOUBLE SPACE'
LINE4      DC  CL132'DOUBLE SPACE'
LINE5      DC  CL132'TRIPLE SPACE AND LAST LINE'
REPTOK     DC  CL25'ROPES REPORT GENERATED OK'
ERRCONST   DC  CL18'ROPES ERROR CODE='
C3         DC  C'C'
           END
```

Example of report generation by a batch Assembler program.

```
ASMSAMP CSECT
        STM  14,12,12(13)            SAVE REGISTERS
        BALR 2,0                  SET REG 2 AS BASE
        USING *,2                    AND ASSIGN IT
        ST   13,REGSAVE+4          CALLER SAVE AREA
        LR   3,13                      SAVE POINTER
        LA   13,REGSAVE                MY SAVE AREA
        ST   13,8(3)               CHAIN SAVE AREAS
        ROPEBPR                   PREPARE FOR ROPES
        MVC  RPTNAME,=CL8'REPORTA'  SET REPORT NAME
        MVC  LINCC,CH1         TOP OF PAGE CC (CH1)
        MVC  LINDATA,LINE1              LINE DATA
        ROPEBSL                  SEND LINE TO ROPES
        MVC  LINCC,SINGLE           SINGLE SPACE CC
        MVC  LINDATA,LINE2              LINE DATA
        ROPEBSL                  SEND LINE TO ROPES
        MVC  LINCC,DOUBLE           DOUBLE SPACE CC
        MVC  LINDATA,LINE3              LINE DATA
        ROPEBSL                 SEND LINE TO ROPESS
        MVC  LINCC,TRIPLE           TRIPLE SPACE CC
        MVC  LINDATA,LINE4              LINE DATA
        ROPEBSL                  SEND LINE TO ROPES
        ROPEBEL               END REPORT GENERATION
        ROPEBCK                CHECKPOINT THE REORT
        ROPEBTR             END OF ROPES PROCESSING
*DISPLAY MESSAGE ON THE CONSOLE FROM REPTOK
        B    PRGEXIT           END RETURN TO CALLER
ERRPARA MVC  ERRCODE,QUEUERC    MOVE RESPONSE CODE
*DISPLAY MESSAGE ON THE CONSOLE FROM ERRMSSG
        ROPEBDR ERRPARA=TERMINAT      DELETE REPORT
TERMNAT DS     0H
        ROPEBTR ERRPARA=TERMINAT       END OF ROPES
PRGEXIT DS     0H
        L    13,4(13)            INT TO CALLER SAVE
        LM   14,12,12(13)         RESTORE REGISTERS
        BR   14                 AND RETURN TO CALLER
```

```
LINE1   DC   CL132'TOP OF PAGE FROM SAMPLE PROGRAM'
LINE2   DC    CL132'SINGLE SPACE'
LINE3   DC    CL132'DOUBLE SPACE'
LINE4   DC    CL132'TRIPLE SPACE AND LAST LINE'
REPTOK  DC    CL125'ROPES REPORT GENERATED OK'
ERRMSG  DS    0CL19
ERRCNST DC    CL18'ROPES ERROR CODE='
ERRCODE DC    CL1' '
REGSAVE DC    18F'0'
        ROPESPRM
        END
```

Example of report retrieval by an online ANS COBOL program

```
IDENTIFICATION DIVISION.
PROGRAM-ID COBSAMP.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77  C3          PIC X VALUE 'C'.
77  LEN-OKMSG   PIC S9(4) VALUE +25.
77  LEN-LINE    PIC S9(4) VALUE +133.
77  LEN-ERRMSG  PIC S9(4) VALUE +19.
01  ROPESPRM COPY ROPESPRM.
01   REPTOK          PIC X(25) VALUE 'ROPES REPORT
RETRIEVED OK'.
01  ERRMSG.
    02 ERRCNST PIC X(18) VALUE 'ROPES ERROR CODE='.
    02 ERRCODE PIC X     VALUE ' '.
PROCEDURE DIVISION.
ROPES-PREPARE. COPY ROPEOPR.
    MOVE 'REPORTA' TO REPORT-NAME.
ROPES-READLINE. COPY ROPEORL.
    EXEC CICS SEND FROM(LINE-AREA) LENGTH(LEN-LINE)
              CTLCHAR(C3) END-EXEC.
    GO TO ROPES-READLINE.
ERR-PARA.
    IF QUEUERC NOT = QRCERPT GO TO FATAL-ERROR.
ROPES-TERMINATE. COPY ROPEOTR REPLACING
                      ERR-PARA BY FATAL-ERROR.
    EXEC CICS SEND FROM(REPTOK) LENGTH(LEN-OKMSG)
              CTLCHAR(C3) END-EXEC.
    GO TO PROGRAM-EXIT.
FATAL-ERROR.
    MOVE QUEUERC TO ERRCODE.
    EXEC CICS SEND FROM(ERRMSG) LENGTH(LEN-ERRMSG)
              CTLCHAR(C3) END-EXEC.
ROPES-TERMINATE-ONERR. COPY ROPEOTR REPLACING
                                     ERR-PARA BY
PROGRAM-EXIT.
PROGRAM-EXIT.
    EXEC CICS RETURN END-EXEC.
    STOP RUN.
```

Example of report retrieval by batch ANS COBOL program

```
IDENTIFICATION DIVISION.
PROGRAM-ID. COBSAMP.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  ROPESPRM COPY ROPESPRM.
01   REPTOK          PIC X(25) VALUE 'ROPES REPORT
RETRIEVED OK'.
01  ERRMSG.
    02 ERRCNST PIC X(18) VALUE 'ROPES ERROR CODE='.
    02 ERRCODE PIC X VALUE ' '.
PROCEDURE DIVISION.
ROPES-PREPARE. COPY ROPEBPR.
    MOVE 'REPORTA' TO REPORT-NAME.
ROPES-READLINE. COPY ROPEBRL.
    DISPLAY LINE-AREA.
    GO TO ROPES-READLINE.
ERR-PARA.
    IF QUEUERC NOT = QRCERPT GO TO FATAL-ERROR.
ROPES-TERMINATE. COPY ROPEBTR REPLACING
                              ERR-PARA BY PROGRAM-
EXIT.
    DISPLAY REPTOK.
    GO TO PROGRAM-EXIT.
FATAL-ERROR.
    MOVE QUEUERC TO ERRCODE.
    DISPLAY ERRMSG.
ROPES-TERMINATE-ONERR.
    COPY ROPEBTR REPLACING
             ERR-PARA BY PROGRAM-EXIT.
PROGRAM-EXIT.
    STOP RUN.
```

Example of report retrieval by an online PL/I program

```
PLISAMP:
    PROC OPTIONS(MAIN);
    %INCLUDE ROPESPRM;
        DCL  REPTOK  CHAR(25)  INIT('ROPES  REPORT
RETRIEVED OK');
    DCL 1 ERRMSG,
          2 ERRCNST CHAR(18)
            INIT('ROPES ERROR CODE=');
          2 ERRCODE CHAR(1) INIT(' ');
    DCL LEN_OKMSG BIN FIXED INIT(25);
    DCL LEN_ERRMSG BIN FIXED INIT(19);
    DCL LEN_LINE BIN FIXED INIT(133);
    DCL C3 CHAR(1) INIT('C');
START_PROCESS:
   %INCLUDE ROPEOPR; /* PREPARE ROPES PROCESSING*/
   REPORT_NAME = 'REPORTA';  /* SET REPORT NAME */
READ_LINES:
    %INCLUDE ROPEORL;      /* READ A REPORT LINE */
   EXEC CICS SEND FROM(LINE_AREA) LENGTH(LEN_LINE)
              CTLCHAR(C3);
GO TO READ_LINES;      /* CONTINUE READING LINES */
ERR_PARA:
IF QUEUERC NOT EQUAL TO QRCERPT
    THEN GO TO FATAL_ERROR;
%DECLARE ERR_PARA CHAR;
   %ERR_PARA='FATAL_ERROR'; /* AVOID ERROR LOOP */
    %INCLUDE ROPEOTR; /* TERMINATE PROCESSING /*
EXEC CICS SEND FROM(REPTOK) LENGTH(LEN_OKMSG)
        CTLCHAR(C3);
GO TO PROGRAM_EXIT;
FATAL_ERROR:
ERRCODE = QUEUERC;
    EXEC CICS SEND FROM(ERRMSG) LENGTH(LEN_ERRMSG)
              CTLCHAR(C3);
%ERR_PARA = 'PROGRAM_EXIT';  /* AVOID ERROR LOOP */
%INCLUDE ROPEOTR;
PROGRAM_EXIT:
EXEC CICS RETURN;
END PLISAMP;
```

Example of report retrieval by a batch PL/I program

```
PLISAMP:
    PROC OPTIONS(MAIN);
    %INCLUDE ROPESPRM;
        DCL  REPTOK  CHAR(25)  INIT('ROPES  REPORT
RETRIEVED OK');
    DCL 1 ERRMSG,
          2 ERRCNST CHAR(18)
            INIT('ROPES ERROR CODE=');
          2 ERRCODE CHAR(1) INIT(' ');
START_PROCESS:
   %INCLUDE ROPEBPR; /* PREPARE ROPES PROCESSING*/
   REPORT_NAME = 'REPORTA';  /* SET REPORT NAME */
READ_LINES:
    %INCLUDE ROPEBRL;      /* READ A REPORT LINE */
    DISPLAY (LINE_AREA);    /* SHOW REPORT LINE */
    GO TO READ_LINES;       /* CONTINUE READING */
ERR_PARA:
    IF QUEUERC NOT EQUAL TO QRCERPT
    THEN GO TO FATAL_ERROR;
    %DECLARE ERR_PARA CHAR;
   %ERR_PARA='FATAL_ERROR'; /* AVOID ERROR LOOP */
    %INCLUDE ROPEOTR;         /* TERMINATE ROPES */
    DISPLAY (REPTOK);
    GO TO PROGRAM_EXIT;       /* RETURN */
FATAL_ERROR:
   ERRCODE = QUEUERC;    /* MOVE CODE TO MESSAGE */
   DISPLAY (ERRMSG);                /* SHOW ERROR */
   %ERR_PARA='PROGRAM_EXIT';      /* AVOID LOOP */
   %INCLUDE ROPEBTR;          /* TERMINATE ROPES */
PROGRAM_EXIT:
    END PLISAMP;
```

Example of report retrieval by an online Assembler

program

```
DFHEISTG DSECT
         ROPESPRM
ERRMSG   DS     0CL19               ERROR MESSAGE
ERRCNST  DS     CL18             MESSAGE CONSTANT
ERRCODE  DS     CL1           ROPES RESPONSE CODE
ASMSAMP  CSECT
         ROPEOPR     PREPARE FOR ROPES PROCESSING
         MVC    RPTNAME,=CL8'REPORTA' SET NAME
READLINE DS     0H
         ROPEORL                READ REPORT LINE
*SHOW REPORT LINE ON SCREEN
         EXEC CICS SEND FROM(LINAREA) CTLCHAR(C3)
         B      READLINE        CONTINUE READING
ERRPARA  DS     0H
*IF END OF REPORT DETECTED, TERMINATE PROCESSING
*OTHERWISE PROCESS THE ERROR
         CLC    QUEUERC,QRCERPT  END OF REPORT?
         BNE    BADERR           NO, FATAL ERROR
         ROPEOTR ERRPARA=BADERR  TERMINATE ROPES
         EXEC CICS SEND FROM(REPTOK) CTLCHAR(C3)
         B      PRGEXIT                AND RETURN
BADERR   DS     0H
         MVC    ERRCODE,QUEUERC   RESPONSE CODE
         MVC    ERCNST,ERRCONST   MESSAGE CONSTANT
         EXEC CICS SEND FROM(ERRMSSG) CTLCHAR(C3)
         ROPEOTR ERRPARA=PRGEXIT  AVOID ERR LOOP
PRGEXIT  DS     0H
         EXEC CICS RETURN
REPTOK   DC     CL25'ROPES REPORT RETRIEVED OK'
ERRCONST DC     CL18'ROPES ERROR CODE='
C3       DC     C'C'
         END
```

Example of report retrieval by a batch Assembler program

```
ASMSAMP  CSECT
         STM    14,12,12(13)     SAVE REGISTERS
         BALR   2,0              SET REG 2 AS BASE
         USING  *,2              AND ASSIGN IT
         ST     13,REGSAVE+4     CALLER SAVE AREA
         LR     3,13             SAVE POINTER
         LA     13,REGSAVE       MY SAVE AREA
         ST     13,8(3)          INTO CALLER'S
         ROPEBPR                 PREPARE ROPES
         MVC    RPTNAME,=CL8'REPORTA' SET NAME
READLINE DS     0H
         ROPEBRL                 READ REPORT LINE
*DISPLAY REPORT LINE ON THE CONSOLE FROM LINEAREA
         B      READLINE         CONTINUE READING
ERRPARA  DS     0H
         CLC    QUEUERC,QRCERPT  END OF REPORT?
         BNE    BADERR           NO,FATAL ERROR
               ROPEBTR  ERRPARA=BADERR   TERMINATE
PROCESSING
*DISPLAY MESSAGE ON THE CONSOLE FROM REPTOK
         B      PRGEXIT           AND RETURN
BADERR   DS     0H
           MVC     ERRCODE,QUEUERC    MOVE CODE TO
MESSAGE
*DISPLAY MESSAGE ON THE CONSOLE FROM ERRMSG.
         ROPEBTR ERRPARA=PRGEXIT  AVOID ERR LOOP
PRGEXIT  DS     0H
         L      13,4(13)   POINT TO CALLER'S SAVE
         LM     14,12,12(13)   RESTORE REGS
         BR     14             AND RETURN
REPTOK   DC     CL25'ROPES REPORT RETRIEVED OK'
ERRMSG   DS     0CL19                 ERR MESSAGE
         DC     CL18'ROPES ERROR CODE='
ERRCODE  DC     CL1' '
REGSAVE  DC     18F'0'
         ROPESPRM
         END
```

Appendix E. Invoking Commands From Programs

Your application programs may dynamically invoke ROPES commands whenever you wish. This facility will be helpful in integrating ROPES functions into your existing and new application systems.

To invoke a ROPES command from your program, follow the instructions in the chapter on Security, "Command Processor Front-end Programs."

Appendix F. Front-ending ROPES Programs

You can "Front End" most ROPES programs to integrate them into your application menus and systems. By "Front Ending" we mean that you can invoke a ROPES program from your own program by preparing the necessary terminal input and transferring control to the ROPES program by issuing a LINK or XCTL command. Alternatively, you can replace a ROPES program in your PCT with a module of your own which then LINKs or XCTLs to the ROPES module after your "Front End" program finishes its processing.

In some cases modification to a ROPES routine is needed before it can be integrated into your application. The modification involves the creation of a COMMAREA in which you pass data to the ROPES program. Several such modifications have already been prepared and are available to you. Please call or write to our technical support staff for assistance.

Appendix G. Data Areas

Some installations have found it desirable to integrate the maintenance of ROPES Control Information into their application systems. In other cases, specific end-user or application requirements made it necessary for their applications to directly access ROPES Control Information. The following information is provided to you to facilitate your use of ROPES in this way.

The ROPES Control Information is contained in two VSAM KSDS Clusters.

The ROPERIB data set contains the System Control Information and the Report Control Information. The cluster is keyed using the eight character Report Name. The System Control Record key is " SYSCNTL" (the first character is a blank).

The ROPES Printer Controls are contained in the ROPEPCR data set. The records are keyed using the four character printer identifier.

We have provided assembler DSECTS that describe each of the three control record types. You may use these DSECTS in your programs. All fields in these areas can be defined in COBOL and PL/1. If you need assistance in translating these DSECTS into COBOL or PL/1 record descriptions, please call our technical support staff.

The DSECTS and the areas they map are as follows:

ROPESYSA	maps the System Control Record
ROPEROP	maps the Report Options Record
ROPEPCO	maps the Printer Options Record
ROPEAUT	maps the Automatic Start-up Options Record
ROPEBRW	maps the Browse and Display Options Record
ROPEMCO	maps the Miscellaneous Options record
ROPERIB	maps the Report Information Block Record
ROPERPI	maps the Report To Printer Index Record
ROPEPCR	maps the Printer Control Record
ROPEQBR	maps the Queue Data Record

Please note that whenever ROPES is accessing these records for update it uses the CICS File Control facilities through the UPDATE parameter of the READ command. This convention must be followed by your application programs in order to insure the integrity of the ROPES data.

Appendix H. IPDS and PPDS Programming in ROPES

ROPES supports IPDS bar code output on LU1 (SCS) attached 4224 printers. The feature is a full IPDS implementation, using the IPDS Application in SCS mode as defined in the 4224 Product and Programming Description Manual. That is to say, the printer is defined as an SCS device, but is switched into IPDS mode for ROPES printing and then returned to SCS mode. To select IPDS processing, you must either specify DSTYPE=IPDS in the ROPES DCB which defines your printer or allow the DSTYPE to default to the DEVTYPE value.

ROPES places the printer into IPDS mode by sending a Function Management Header (FMH). Then the page set-up controls are sent based on the values in the Forms Control Block. Each page of the report is started with a Begin Page structured field, and ended with an End Page structured field. Every transmission of text is terminated by a trailer that requests acknowledgment from the printer.

ROPES ships all text to the printer up to the bar code data in the ROPES report. The next transmission to the printer contains the IPDS bar code structured fields as described below. The values in the bar code structured fields will be supplied, in part, from user data found in the ROPES bar code data fields.

ROPES detects bar code data by scanning each print line prior to output formatting. If the escape character sequence (which is "¬(") is detected, the first two characters following the "¬(" will be examined. If they are "01", then a bar code function will be identified. Any other value will cause all data until the next ")¬" or end of line to be ignored. In the future, other codes will be defined to implement other IPDS facilities.

The escape sequence data is passed to the bar code interpretation routine for processing. The bar code interpretation routine is responsible for all aspects of the processing, including the transmission of the IPDS data stream to the printer.

When the Bar Code interpretation routine returns control, printing continues with the remaining text on the data line.

ROPES also supports PPDS bar code output on any printer that supports PPDS bar code commands, such as the IBM 4247 or IBM 6400 family printers attached by IP and served using either the TCP/IP LPR or Direct Socket printing services. To select PPDS processing, you must specify DSTYPE=PPDS in the ROPES DCB which defines your printer. Your application program must construct the ROPES IPDS bar code data in the ROPES report. The ROPES PPDS interpretation routine will be invoked to translate from IPDS values to PPDS values as required.

The Bar Code Interpretation Routine

The bar code interpretation routine parses the bar code data, constructs the IPDS bar code data stream, and sends the data to the printer. The bar code data is defined in the table "IPDS Bar Code Control

Values” on page 96. All data values shown in the form C’…’ must be coded as characters. Values shown as digits must be coded as characters representing the numeric value of the parameter. Where values are specified as X’….’, you must code the value as a series of hexadecimal digit pairs. Each parameter is separated from the next by a comma. No embedded spaces are permitted. Omitted parameters may be specified by coding two consecutive commas (,,). Default values will be assumed. If the end of the string is encountered before all parameters have been scanned, the omitted parameters will assume their defaults.

IPDS Bar Code Control Values

Field Name	Purpose	Values	Default
Escape Sequence	Identifies the start of a special printer function.	C’¬(‘	None. Must be present.
Function Code	Identifies this as a bar code data function.	C’01’	None.
Text	Supplies the data characters to be expressed as a bar code.	C’…’ Any desired characters, enclosed between ‘ characters. Eg.: ‘00123003’	‘123456789’
Orientation	Supplies the bar code orientation (not all values are supported by all printers).	Three digits, 000 or 090 or 180 or 270.	000
Bar Code X Coordinate	Specifies the location, from the left edge of the page, to the left edge of the bar code.	4 decimal digits, expressed in units of 1/1440".	1440 units (1")
Bar Code Y Coordinate	Specifies the location, from the top edge of the page, to the top edge of the bar code.	4 decimal digits, expressed in units of 1/1440".	1440 units (1")
Type	Indicates the bar code type.	X’..’ From list of values supported by the printer. See your printer manual.	3 of 9 (X’01’)
Modifier	Indicates special functions associated with the bar code.	X’..’ From list of values supported by the printer. See your printer manual.	no check character (X’01’)
Human Readable Data (not all values are supported by all printers)	Indicates the need for HRI print.	C’.’ N = no A = yes, above B = yes, below	N

IPDS Bar Code Control Values			
Field Name	**Purpose**	**Values**	**Default**
Module Width	Indicates the size, in mils of the bar code elements.	Any value from X’00’ to X’FF’. See your printer manual.	X’0E’ (for 4224 printer)
Element Height	Indicates the height of the bar code elements.	Any value from X’0001’ to X’FFFF’. See your printer manual.	X’02D0’ (½")
Height Multiplier	Determines the number of times Element Height is repeated to display the bar code.	Any value from X’01’ to X’FF’. See your printer manual.	X’01’
Wide To Narrow Ratio	Bar code specific coding value.	Any value from X’0000’ = does not use X’0001’ - ‘X’FFFE’ X’FFFF’ - printer default. See your printer manual.	X’00FA’
Escape Terminator	Identifies the end of the special printer function.	C’)¬’	End of line

All other bar code values are set by the software. These include:

Position	Absolute X, Y, offset 0,0
Unit Base	10 inches
L-Units/Unit Base	14400
Font Local ID	X’FF’
Color	X’0008’ (black)

The bar code presentation space and bar code block are defined as the full page, an as such, overlaps any text printed. You should not select a combination of X,Y coordinates, bar code height and text length to cause destructive overlap. You must also take into account the need for a quiet zone surrounding the bar code when you place it on the page.

The following example shows the proper coding for a bar code located at X, Y = 4", 9", ½" in height, with a value of 123456879, in code 3 of 9 with no check digit and no Human Readable Interpretation.

¬(01,"12345689",,5760,12960,,,,,02D0)¬

Business Logic Services

Introduction

Beginning with Version 10.0 we started the process of separating the business logic of the some of the ROPES modules from the Presentation Logic. The Business Logic modules are Command Level, COMMAREA driven programs with a fully documented user interface that are being made available to all users who wish to provide fully integrated access to the ROPES information within their applications, or, through the various facilities of DPL, ECI, EXCI, the CICS Clients and/or the CICS Transaction Gateway, make this information available to users on non-z/OS platforms as required. You can now begin to integrate ROPES into your environment without having to give up control to ROPES modules that may allow unplanned user interactions. We want to make these changes so that you do not need a knowledge of the structure of the ROPES data files, which can, and will, change over time. The data provided in the COMMAREA will be interpreted and formatted by ROPES in a consistent and usable way, so that you do not need to know the internal format of the data, the means by which status values are determined, or how data records in the ROPES files are interrelated.

It is our intention to extend this support to as many modules as is practical, and new ROPEBLxx modules will be introduced, from time to time, in maintenance releases, enhancement releases and in future versions.

Printer Status Business Logic Service

The ROPO and ROPS transaction processor programs ROPEROPO and ROPEROPS, which can be linked to from user applications, present displays to your users and allow the user to respond in ways that might result in your users returning to native CICS and not to your application structure. Also, the Presentation Logic in those modules is organized for 3270 Model 2 terminals only, and may not meet your needs either for 3270 presentation or for use by non-MVS platforms. Our modules now contain only the Presentation Logic and both modules now obtain the data to display from a new module, ROPEBLPS (Business Logic – Printer Status). This Business Logic module is now available for your use as well.

Invocation

The ROPEBLPS module is invoked by an EXEC CICS LINK COMMAREA(PSCMAREA) LENGTH(32294) command. The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

Communication Area

The ROPEPSCM (ROPES Printer Status Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: PSCM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in Figure **Figure 15** on page 102. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area (including Working Storage). It is shown in Figure **Figure 16** on page 103.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

PSMCRPCR

The requested printer id. This is the printer to be displayed and is equivalent to the prtr field in following command: ROPO/prtr. If the field is blank, then ROPES will select the first printer which meets the other selection criteria, which are based on PSCMMODE and PSCMCTOVER.

PSCMRREPT

The requested report name. This is the first report to be displayed and is equivalent to the reptname field in the following command: ROPO/prtr//reptname. If the field is blank then ROPES will select the first report assigned to the printer specified or selected.

PSCMRCLAS
The requested class name. This is the report class to be used as a filter and is equivalent to the c field in the following command: ROPO/prtr/c/reptname. If the field is blank then ROPES will not filter based on report class.

PSCMNEXTP
The next eligible printer defined to ROPES is returned in this field. This is the same as the value shown on the ROPO or ROPS panel. It is determined by the cursor position in the ROPES Printer Control file at the end of the processing for the current call, the PSCMMODE and PSCMCTOVER options, and the PSCMRREPT and PSCMRCLAS values.

PSCMCURRP
The number of reports actually returned in the reports array of the COMMAREA. This is a two byte binary number.

PSCMMODE
This is the requested processing mode. You may specify “O” for processing in ROPO, or printer operator mode. You may specify “S” for processing in ROPS, or supervisor mode. In printer operator mode, only printers that can be controlled by the requesting terminal are eligible for processing. All other printers will be ignored. In supervisor mode, all printers are eligible. This is analogous to the ROPO and ROPS processing logic. The eligibility can be further determined by your processing the ROPEPRE1 optional user exit, which is invoked by ROPEBLPS on each call. ROPEPRE1 will be passed the value of the PSCMCTOVER field and this field can be used to identify the “requestor” if the origin of the request is not a CICS terminal.

PSCMDISPF
This flag allows you to override the ROPES system setting for Display Empty Reports. You can specify blank to use the system setting, “Y” to override the system setting with Y or “N” to override the system setting with N.

PSCMCTOVER
This field contains the “requesting” terminal id if the request is not coming from a CICS terminal or if you want to pass a pseudo terminal identifier to the ROPEPRE1 exit program. If the ROPEPRE1 exit is not present, this field will have no effect on the selection of eligible printers.

PSCMRC
This two byte binary field contains the return code set by the Business Logic routine. The return codes are defined as follows:

0 Processing completed without error
1 The requested printer is not eligible for processing in O mode for this requestor
2 The explicitly requested printer does not exist
3 There are no eligible printers for processing in O mode for this requestor
4 There was an Input error processing the ROPES data set. Further details can be determined from the PSCMEIBFN and PSCMEIBRCODE fields
5 The requested report was not found

PSCMEIBFN
This field contains the right-most byte from the EIBFN field which indicates the File Control requested function that failed if return code 4 was presented.

PSCMEIBRCODE
This field contains the left-most byte from the EIBRCODE field which indicates the File Control response value if return code 4 was presented.

PSCMTMID
This is the printer id of the printer selected by ROPEBLPS based on the selection criteria provided.

PSCMCTMID
This is the control terminal value found in the printer definition.

PSCMACLASS
This is the list of active classes found in the printer controls.

PSCMSTATUS

This is the interpreted value of the printer's current status. These values are documented in the ROPO and ROPS transaction descriptions in the *ROPES User's Guide*.

PSCMLASTMSG

This is the last message issued by ROPES regarding the printer's status. This is the value that normally appears on the second line of the ROPO or ROPS panel.

PSCMFORM

This is the current Forms Control Block Suffix that represents the form that ROPES considers the current form in or for this printer.

PSCMREPTS

This is an array of 525 report entries. The number of entries actually populated is indicated by the PSCMCURRP value. If the COMMAREA is reused by your application, the contents of any entries beyond that indicated by PSCMCURRP is unpredictable, as the unused array elements are not initialized by ROPEBLPS. Each array element consists of the following fields.

PSCMRPPNM

The name of the report.

PSCMQLINES

The number of lines on queue for the report. This number is right-justified in the field and edited to remove leading zeros.

PSCMQPAGES

The number of pages on queue for the report. This number is right-justified in the field and edited to remove leading zeros.

PSCMPLINES

The number of lines printed at this printer for the report. This number is right-justified in the field and edited to remove leading zeros.

PSCMPPAGES

The number of pages printed at this printer for the report. This number is right-justified in the field and edited to remove leading zeros.

PSCMRPCL

The report's class value.

PSCMRPPRI

The report's priority value at this printer. This field reflects the printer priority override value, if any. This number is right-justified in the field and edited to remove leading zeros.

PSCMRSTATUS

This is the interpreted value of the report's current status on this printer. These values are documented in the ROPO and ROPS transaction descriptions in the *User's Guide*.

PSCMRFORM

This is the assigned Forms Control Block suffix for this report.

User Exit Program

A user exit program is available for your use to further control the data made available to the end-user. The user exit is invoked as follows:

```
EXEC  CICS LINK PROGRAM('ROPEPRE1')
 COMMAREA(COMMAREA) LENGTH(=AL2(12)) NOHANDLE.
```

The COMMAREA is defined as follows:

```
COMMAREA DS    0D
PCRADDR  DS    A        ADDRESS OF THE PCR
RETCODE  DS    AL4      RETURN CODE FIELD
CTERMID  DS    CL4      CTL TERMINAL OVERRIDE
```

Use the data provided to determine if the data is to be returned to the user. Set the RETCODE field to zero if the data is to be used, or to any non-zero value if the data is to be suppressed.

```
         ROPEBLCM P=PSCM
***********************************************************************
*                     ---     R O P E S     ---                       *
*           Printer Status Business Logic Communication Area          *
***********************************************************************
PSCMAREA     DS 0D    *  Start of COMMAREA.
PSCMCNTL     DS 0C    *  Start of controls segment.
PSCMRPCR     DS CL4   *  Requested printer id.
PSCMRREPT    DS CL8   *  Requested report name.
PSCMRCLAS    DS CL1   *  Requested class.
             DS CL1   *  Reserved.
PSCMNEXTP    DS CL4   *  Next eligible printer for display.
PSCMCURRP    DS AL2   *  Current number of reports on the printer.
PSCMMODE     DS CL1   *  Request mode (rop)O or (rop)S.
PSCMDISPF    DS CL1   *  Display empty override flag (Y, N or blank).
PSCMCTOVER   DS CL4   *  Control terminal override value.
PSCMRC       DS AL2   *  Return code.
*                     *  0 - No error.
*                     *  1 - Control terminal error.
*                     *  2 - Printer not found.
*                     *  3 - No printers in system.
*                     *  4 - PCR file I/O error.
*                     *  5 - Requested report not found.
PSCMEIBFN    DS XL1   *  EIBFN+1 IF RC=4
PSCMEIBRCODE DS XL1   *  EIBRCODE IF RC=4
PSCMCFILL    DS XL19  *  Reserved.
PSCMHEADER   DS 0C    *  Start of header data segment.
PSCMTMID     DS CL4   *  Returned printer id.
PSCMCTMID    DS CL4   *  Control terminal for this printer.
PSCMACLASS   DS CL8   *  Active classes.
PSCMSTATUS   DS CL10  *  Printer status (interpreted).
PSCMLASTMSG  DS CL60  *  Last status message for this printer.
PSCMFORM     DS CL4   *  Last FCB used with this printer.
PSCMJIC2     DS XL22  *  RESERVED.
PSCMREPTS    DS 0C    *  Start of the report entries.
***********************************************************************
* The printer has CURRP report entries.  There are at most, 525       *
* report entries, each defined as follows:                            *
***********************************************************************
PSCMRPPNM    DS CL8   *  Report name.
PSCMQLINES   DS CL8   *  Queued lines.
PSCMQPAGES   DS CL8   *  Queued pages.
PSCMPLINES   DS CL8   *  Printed lines.
PSCMPPAGES   DS CL8   *  Printed Pages.
PSCMRPCL     DS CL1   *  Report class.
PSCMRPPRI    DS CL4   *  Report priority.
PSCMRSTATUS  DS CL12  *  Report status (interpreted).
PSCMRFORM    DS CL4   *  Report FCB suffix.
PSCMRFILL    DS XL1  *  Reserved
PSCMRPLL EQU *-PSCMRPPNM *  Length of one report entry.
             DS 524CL(PSCMRPLL) *  Balance of the entries.
PSCMCOMML EQU *-PSCMRPPNM                                              X
                         *  Length of the COMMAREA.
```

Figure 15 - Printer Status Commarea - Assembler

```
*****************************************************************
*               --- R O P E S         ---                       *
*     Printer Status Business Logic Communication Area          *
*****************************************************************
 01  PSCMAREA.
*  Start of controls segment.
     05  PSCMCNTL.
         10  PSCMRPCR     PIC X(4).
         10  PSCMRREPT    PIC X(8).
         10  PSCMRCLAS    PIC X(1).
         10  FILLER       PIC X(1).
         10  PSCMNEXTP    PIC X(4).
         10  PSCMCURRP    PIC 99 COMP.
         10  PSCMMODE     PIC X(1).
         10  PSCMDISPF    PIC X(1).
         10  PSCMCTOVER   PIC X(4).
         10  PSCMRC       PIC 99 COMP.
             88  PSCM_NO_ERROR                   VALUE 0.
             88  PSCM_CONTROL_TERMINAL_ERROR     VALUE 1.
             88  PSCM_PRINTER_NOT_FOUND          VALUE 2.
             88  PSCM_NO_ELIGIBLE_PRINTERS       VALUE 3.
             88  PSCM_PCR_FILE_IOERROR           VALUE 4.
             88  PSCM_REQUESTED_REPORT_NOT_FOUND VALUE 5.
         10  PSCMEIBFN    PIC X(1).
         10  PSCMEIBRCODE PIC X(1).
         10  PSCMCFILL    PIC X(19).
*  Start of header data segment.
     05  PSCMHEADER.
         10  PSCMTMID     PIC X(4).
         10  PSCMCTMID    PIC X(4).
         10  PSCMACLASS   PIC X(8).
         10  PSCMSTATUS   PIC X(10).
         10  PSCMLASTMSG  PIC X(60).
         10  PSCMFORM     PIC X(4).
         10  PSCMJIC2     PIC X(22).
     05  PSCMREPTS        OCCURS 525 TIMES.
*****************************************************************
* The printer has CURRP report entries.  There are at most 525  *
* report entries, each defined as follows:                      *
*****************************************************************
         10  PSCMRPPNM    PIC X(8).
         10  PSCMQLINES   PIC X(8).
         10  PSCMQPAGES   PIC X(8).
         10  PSCMPLINES   PIC X(8).
         10  PSCMPPAGES   PIC X(8).
         10  PSCMRPCL     PIC X(1).
         10  PSCMRPPRI    PIC X(4).
         10  PSCMRSTATUS  PIC X(12).
         10  PSCMRFORM    PIC X(4).
         10  PSCMRFILL    PIC X(1).
```

Figure 16 Printer Status Commarea - COBOL

Network Status Business Logic Service

The ROPN transaction processor program ROPEROPN, which can be linked to from user applications, presents a display to your users and allows the users to respond in ways that might result in your users returning to native CICS and not to your application structure. Also, the Presentation Logic in this module is organized for 3270 Model 2 terminals only, and may not meet your needs either for 3270 presentation or for use by non-MVS platforms. Our module now contains only the Presentation Logic and the module now obtains the data to display from a new module, ROPEBLNS (Business Logic – Network Status). This Business Logic module is now available for your use as well.

Invocation

The ROPEBLNS module is invoked by an EXEC CICS LINK COMMAREA(NSCMAREA) LENGTH(32072) command. The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

Communication Area

The ROPEBLCM (ROPES Printer Status Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: NSCM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in **Figure 17** on page 106. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area (including Working Storage). It is shown in **Figure 18** on page 107.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

NSCMRPCR
The requested printer id. This is the printer to be displayed and is equivalent to the prtr field in following command: ROPN/prtr. If the field is blank, then ROPES will select the first printer.

NSCMNEXTP
The next eligible printer defined to ROPES is returned in this field. This is the same as the value shown on the ROPN panel. It is determined by the cursor position in the ROPES Printer Control file at the end of the processing for the current call.

NSCMCNTRQ
The maximum number of printers to be returned in the COMMAREA. This is a two byte binary number.

NSCMCNTRT
The actual number of printers returned in the COMMAREA. This is a two byte binary number.

NSCMRC
This two byte binary field contains the return code set by the Business Logic routine. The return codes are defined as follows:

- 0 Processing completed without error
- 2 The explicitly requested printer does not exist
- 3 There are no eligible printers for processing in O mode for this requestor
- 4 There was an Input error processing the ROPES data set. Further details can be determined from the NSCMEIBFN and NSCMEIBRCODE fields

NSCMEIBFN
This field contains the right-most byte from the EIBFN field which indicates the File Control requested function that failed if return code 4 was presented.

NSCMEIBRCODE
This field contains the left-most byte from the EIBRCODE field which indicates the File Control response value if return code 4 was presented.

NSCMTMID
This is the printer id of the printer selected by

ROPEBLPS based on the selection criteria provided.

NSCMACLASS
This is the list of active classes found in the printer controls.

NSCMSTATUS
This is the interpreted value of the printer's current status. These values are documented in the ROPO or ROPS transaction descriptions in the *ROPES User's Guide*.

NSCMLASTMSG
This is the last message issued by ROPES regarding the printer's status. This is the value that normally appears on the second line of the ROPO or ROPS panel.

NSCMFORM
This is the current Forms Control Block Suffix that represents the form that ROPES considers the current form in or for this printer.

NSCMRPPNM
The name of the current report, if any, or blank.

NSCMQLINES
The number of lines on queue for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCMQPAGES
The number of pages on queue for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCMPLINES
The number of lines printed at this printer for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCMPPAGES
The number of pages printed at this printer for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCMRPCL
The current report's class value. If there is no current report, this field is blank.

NSCMRPPRI
The current report's priority value at this printer. This field reflects the printer priority override value, if any. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCMRSTATUS
This is the interpreted value of the current report's current status on this printer. These values are documented in the ROPO and ROPS transaction descriptions in the *User's Guide*. If there is no current report, this field is blank.

NSCMRFORM
This is the assigned Forms Control Block suffix for the current report. If there is no current report, this field is blank.

User Exit Program

A user exit program is available for your use to further control the data made available to the end-user. The user exit is invoked as follows:

```
EXEC  CICS LINK PROGRAM('ROPENSE1')
 COMMAREA(COMMAREA) LENGTH(=AL2(8)) NOHANDLE.
```

The COMMAREA is defined as follows:

```
COMMAREA DS    0D
PCRADDR  DS    A        ADDRESS OF THE PCR
RETCODE  DS    AL4      RETURN CODE FIELD
```

Use the data provided to determine if the data is to be returned to the user. Set the RETCODE field to zero if the data is to be used, or to any non-zero value if the data is to be suppressed.

```
         ROPEBLCM T=NSCM
***********************************************************************
*                     ---     R O P E S     ---                        *
*            Network Status Business Logic Communication Area          *
***********************************************************************
NSCMAREA     DS 0D    *  Start of COMMAREA.
NSCMCNTL     DS 0C    *  Start of controls segment.
NSCMRPCR     DS CL4   *  Requested printer id.
NSCMNEXTP    DS CL4   *  Next eligible printer for display.
NSCMCNTRQ    DS AL2   *  Number of entries requested.
NSCMCNTRT    DS AL2   *  Number of entries returned.
NSCMRC       DS AL2   *  Return code.
*                     *  0 - No error.
*                     *  2 - Printer not found.
*                     *  3 - No printers in system.
*                     *  4 - PCR file I/O error.
NSCMEIBFN    DS XL1   *  EIBFN+1 IF RC=4
NSCMEIBRCODE DS XL1   *  EIBRCODE IF RC=4
NSCMCFILL    DS XL16  *  Reserved.
***********************************************************************
* The network has an unknown number of printers.  There are, at most, *
* 200 printer entries in the COMMAREA.  The actual number present is  *
* set in the &P.CNTRT field. Each entry is defined as follows:        *
***********************************************************************
NSCMPRTR     DS 0C    *  Start of printers data segment.
NSCMTMID     DS CL4   *  Returned printer id.
NSCMACLASS   DS CL8   *  Active classes.
NSCMSTATUS   DS CL12  *  Printer status (interpreted).
NSCMLASTMSG  DS CL60  *  Last status message for this printer.
NSCMFORM     DS CL4   *  Last FCB used with this printer.
NSCMRPPNM    DS CL8   *  Report name.
NSCMQLINES   DS CL8   *  Queued lines.
NSCMQPAGES   DS CL8   *  Queued pages.
NSCMPLINES   DS CL8   *  Printed lines.
NSCMPPAGES   DS CL8   *  Printed Pages.
NSCMRPPRI    DS CL4   *  Report priority.
NSCMRSTATUS  DS CL12  *  Report status (interpreted).
NSCMRFORM    DS CL4   *  Report FCB suffix.
NSCMRPCL     DS CL1   *  Report class.
NSCMRFILL    DS XL11  *  Reserved
NSCMRPLL EQU *-NSCMPRTR  *  Length of one report entry.
             DS 200CL(NSCMRPLL) *  Balance of the entries
NSCMCOMML EQU *-NSCMRPPNM                                              X
                         *  Length of the COMMAREA.
```

Figure 17 - Network Status Commarea - Assembler

```
***************************************************************** 00010000
*               --- R O P E S         ---                       * 00020000
*     Network Status Business Logic Communication Area          * 00030001
***************************************************************** 00040000
 01  NSCMAREA.                                                      00050001
*  Start of controls segment.                                       00060000
     05  NSCMCNTL.                                                  00070001
*  Requested printer id.                                            00080000
         10  NSCMRPCR     PIC X(4).                                 00090001
*  Next eligible printer for display.                               00160000
         10  NSCMNEXTP    PIC X(4).                                 00170001
*  Requested number of printers.                                    00180001
         10  NSCMCNTRQ    PIC 9(4) COMP.                            00190001
*  Retrieved number of printers.                                    00240001
         10  NSCMCNTRT    PIC 9(4) COMP.                            00250001
*  Return code.                                                     00260000
         10  NSCMRC       PIC 9(4) COMP.                            00270001
             88  NSCM_NO_ERROR                  VALUE 0.            00280001
             88  NSCM_PRINTER_NOT_FOUND         VALUE 2.            00300001
             88  NSCM_NO_ELIGIBLE_PRINTERS      VALUE 3.            00310001
             88  NSCM_PCR_FILE_IOERROR          VALUE 4.            00320001
*  EIBFN+1 IF RC=4                                                  00340000
         10  NSCMEIBFN    PIC X(1).                                 00350001
*  EIBRCODE IF RC=4                                                 00360000
         10  NSCMEIBRCODE PIC X(1).                                 00370001
*  Reserved.                                                        00380000
         10  NSCMCFILL    PIC X(16).                                00390001
*  Start of header data segment.                                    00400000
     05  NSCMDETAIL.                                                00410001
*  Returned printer id.                                             00420000
         10  NSCMPRTR     PIC X(4).                                 00430001
*  Active classes.                                                  00460000
         10  NSCMACLASS   PIC X(8).                                 00470001
*  Printer status (interpreted).                                    00480000
         10  NSCMSTATUS   PIC X(12).                                00490001
*  Last status message for this printer.                            00500000
         10  NSCMLASTMSG  PIC X(60).                                00510001
*  Last FCB used with this printer.                                 00520000
         10  NSCMFORM     PIC X(4).                                 00530001
*  Report name.                                                     00620000
         10  NSCMRPPNM    PIC X(8).                                 00630001
*  Queued lines.                                                    00640000
         10  NSCMQLINES   PIC X(8).                                 00650001
*  Queued pages.                                                    00660000
         10  NSCMQPAGES   PIC X(8).                                 00670001
*  Printed lines.                                                   00680000
         10  NSCMPLINES   PIC X(8).                                 00690001
*  Printed Pages.                                                   00700000
         10  NSCMPPAGES   PIC X(8).                                 00710001
*  Report priority.                                                 00711001
         10  NSCMRPPRI    PIC X(4).                                 00712001
*  Report status (interpreted).                                     00760000
         10  NSCMRSTATUS  PIC X(12).                                00770001
*  Report FCB suffix.                                               00780000
         10  NSCMRFORM    PIC X(4).                                 00790001
*  Report class.                                                    00791001
         10  NSCMRPCL     PIC X(1).                                 00792001
*  Reserved                                                         00800000
         10  NSCMRFILL    PIC X(11).                                00810001
```

Figure 18 Network Status Commarea - COBOL

Report List Business Logic Service

The RORL and ROAL transaction processor programs ROPERORL and ROPEROAL, which can be linked to from user applications, presents a display to your users and allows the users to respond in ways that might result in your users returning to native CICS and not to your application structure. Also, the Presentation Logic in this module is organized for 3270 Model 2 terminals only, and may not meet your needs either for 3270 presentation or for use by non-MVS platforms. Our modules now contain only the Presentation Logic and the modules now obtain the data to display from a new module, ROPEBLRL (Business Logic – Report List). This Business Logic module is now available for your use as well.

Invocation

The ROPEBLRL module is invoked by an EXEC CICS LINK COMMAREA(RLCMAREA) LENGTH(12912) command. The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

Communication Area

The ROPEBLCM (ROPES Printer Status Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: RLCM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in Figure **Figure 19** on page 110. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area (including Working Storage). It is shown in Figure **Figure 20** on page 111.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

RLCMRRPT
The requested report name. This is the first report to be displayed and is equivalent to the reportid field in following commands: RORL/reportid or ROAL/reportid. If the field is blank, then ROPES will select the first report.

RLCMNEXTR
The next report defined to ROPES is returned in this field. This is the same as the value shown on the RORL and ROAL panels. It is determined by the cursor position in the ROPES Report Control file at the end of the processing for the current call.

RLCMFILE
The name of the file to be processed. This is the FCT name, and should be ROPERIB for the primary report file, and ROPEALR for the Alternate Facility report file.

RLCMCNTRQ
The maximum number of reports to be returned in the COMMAREA. This is a two byte binary number.

RLCMCNTRT
The actual number of reports returned in the COMMAREA. This is a two byte binary number.

RLCMRC
This two byte binary field contains the return code set by the Business Logic routine. The return codes are defined as follows:

- 0 Processing completed without error
- 2 File open/close error
- 3 End of file reached
- 4 There was an Input error processing the ROPES data set. Further details can be determined from the RLCMEIBFN and RLCMEIBRCODE fields

RLCMEMPTY
This field is a Y | N field used to control the return of empty reports. Set the field to “Y” if you want empty reports returned. Set the field to “N” if you want empty reports skipped.

RLCMSERIAL

This field is a Y | N field used to control the use of Dynamic ENQ/OPEN/CLOSE/DEQ logic on the input file. This should be set to "Y" when the file name is "ROPEALR" and should be set to "N" when the file name is "ROPERIB"

RLCMEIBFN

This field contains the right-most byte from the EIBFN field which indicates the File Control requested function that failed if return code 4 was presented.

RLCMEIBRCODE

This field contains the left-most byte from the EIBRCODE field which indicates the File Control response value if return code 4 was presented.

RLMRPPNM

The name of the report.

RLCMQLINES

The number of lines on queue for the report. This number is right-justified in the field and edited to remove leading zeros.

RLCMQPAGES

The number of pages on queue for the report. This number is right-justified in the field and edited to remove leading zeros.

RLCMRPPRI

The report's priority. This number is right-justified in the field and edited to remove leading zeros.

RLCMRSTATUS

This is the interpreted value of the report's current status on this printer. These values are documented in the ROPO and ROPS transaction descriptions in the *User's Guide*.

RLCMRFORM

This is the assigned Forms Control Block suffix for the report.

RLCMRPCL

The report's class value.

RLCMDISC

The report's discard value.

RLCMRETPD

The report's retention period.

User Exit Program

A user exit program is available for your use to further control the data made available to the end-user. The user exit is invoked as follows:

```
EXEC  CICS LINK PROGRAM('ROPERLE1')
 COMMAREA(COMMAREA) LENGTH(=AL2(12)) NOHANDLE.
```

The COMMAREA is defined as follows:

```
COMMAREA DS     0D
PCRADDR  DS     A        ADDRESS OF THE PCR
FILENAME DS     CL8      FILE TO BE PROCESSED
RETCODE  DS     AL4      RETURN CODE FIELD
```

Use the data provided to determine if the data is to be returned to the user. Set the RETCODE field to zero if the data is to be used, or to any non-zero value if the data is to be suppressed.

```
          ROPEBLCM T=RLCM
***********************************************************************
*                      ---     R O P E S     ---                      *
*              Report List Business Logic Communication Area          *
***********************************************************************
RLCMAREA     DS 0D    *  Start of COMMAREA.
RLCMCNTL     DS 0C    *  Start of controls segment.
RLCMRRPT     DS CL8   *  Requested report id.
RLCMNEXTR    DS CL8   *  Next eligible report for display.
RLCMFILE     DS CL8   *  Input file name to use.
RLCMCNTRQ    DS AL2   *  Number of entries requested.
RLCMCNTRT    DS AL2   *  Number of entries returned.
RLCMRC       DS AL2   *  Return code.
*                     *  0 - No error.
*                     *  2 - File Open/Close error.
*                     *  3 - End of file.
*                     *  4 - Input file I/O error.
RLCMEMPTY    DS CL1   *  Y/N flag to control use of empty reports
RLCMSERIAL   DS CL1   *  Dynamic ENQ/OPEN/CLOSE requested
RLCMEIBFN    DS XL1   *  EIBFN+1 IF RC=4
RLCMEIBRCODE DS XL1   *  EIBRCODE IF RC=4
RLCMCFILL    DS XL14  *  Reserved.
***********************************************************************
* The system has an unknown number of reports.  There are, at most,   *
* 200 report entries in the COMMAREA.  The actual number present is   *
* set in the &P.CNTRT field. Each entry is defined as follows:        *
***********************************************************************
RLCMREPT     DS 0C    *  Start of repoerts data segment.
RLCMRPPNM    DS CL8   *  Report name.
RLCMQLINES   DS CL8   *  Queued lines.
RLCMQPAGES   DS CL8   *  Queued pages.
RLCMRPPRI    DS CL4   *  Report priority.
RLCMRSTATUS  DS CL12  *  Report status (interpreted).
RLCMRFORM    DS CL4   *  Report FCB suffix.
RLCMRPCL     DS CL1   *  Report class.
RLCMDISC     DS CL1   *  Report discard value.
RLCMRETPD    DS CL4   *  Report retention period.
RLCMRFILL    DS CL14  *  Reserved
RLCMRPLL EQU *-RLCMREPT  *  Length of one report entry.
             DS 200CL(RLCMRPLL) *  Balance of the entries.
RLCMCOMML EQU *-RLCMAREA                                               X
                        *  Length of the COMMAREA.
```

Figure 19 - Report List Commarea - Assembler

```
****************************************************************  00010000
*               --- R O P E S         ---                      *  00020000
*     Report List Business Logic Communication Area            *  00030000
****************************************************************  00040000
 01  RLCMAREA.                                                    00050000
*  Start of controls segment.                                     00060000
     05  RLCMCNTL.                                                00070000
*  Requested report id.                                           00080000
         10  RLCMRRPT     PIC X(8).                               00090000
*  Next eligible report for display.                              00160000
         10  RLCMNEXTR    PIC X(8).                               00170000
*  File to be processed (ROPERIB or ROPEALR).                     00170100
         10  RLCMFILE     PIC X(8).                               00171000
*  Requested number of reports.                                   00180000
         10  RLCMCNTRQ    PIC 9(4) COMP.                          00190000
*  Retrieved number of reports.                                   00240000
         10  RLCMCNTRT    PIC 9(4) COMP.                          00250000
*  Return code.                                                   00260000
         10  RLCMRC       PIC 9(4) COMP.                          00270000
             88  RLCM_NO_ERROR                  VALUE 0.          00280000
             88  RLCM_REPORT_NOT_FOUND          VALUE 2.          00300000
             88  RLCM_NO_END_OF_FILE            VALUE 3.          00310000
             88  RLCM_INPUT_FILE_IOERROR        VALUE 4.          00320000
*  Y/N flag to control the use of empty reports.                  00330000
         10  RLCMEMPTY    PIC X(1).                               00331000
*  Y/N flag to control Dynamic ENQ/OPEN/CLOSE requested.          00332000
         10  RLSERIAL     PIC X(1).                               00333000
*  EIBFN+1 IF RC=4                                                00340000
         10  RLCMEIBFN    PIC X(1).                               00350000
*  EIBRCODE IF RC=4                                               00360000
         10  RLCMEIBRCODE PIC X(1).                               00370000
*  Reserved.                                                      00380000
         10  RLCMCFILL    PIC X(14).                              00390000
*  Start of data segment.                                         00400000
     05  RLCMDETAIL.                                              00410000
*  Returned report name.                                          00420000
         10  RLCMREPT     PIC X(8).                               00430000
*  Queued lines.                                                  00440000
         10  RLCMQLINES   PIC X(8).                               00450000
*  Queued pages.                                                  00451000
         10  RLCMQPAGES   PIC X(8).                               00452000
*  Report priority.                                               00453000
         10  RLCMRPPRI    PIC X(4).                               00454000
*  Report status (interpreted).                                   00455000
         10  RLCMRSTATUS  PIC X(12).                              00456000
*  Report FCB suffix.                                             00457000
         10  RLCMRFORM    PIC X(4).                               00458000
*  Report class.                                                  00459000
         10  RLCMRPCL     PIC X(1).                               00459100
*  Report discard option.                                         00460000
         10  RLCMDISC     PIC X(1).                               00470001
*  Report retention period.                                       00480000
         10  RLCMRETPD    PIC X(4).                               00490000
*  Reserved                                                       00800000
         10  RLCMRFILL    PIC X(14).                              00810000
```

Figure 20 Report List Commarea - COBOL

Report Status Business Logic Service

The RORS transaction processor program ROPERORS, which can be linked to from user applications, presents a display to your users and allows the users to respond in ways that might result in your users returning to native CICS and not to your application structure. Also, the Presentation Logic in this module is organized for 3270 Model 2 terminals only, and may not meet your needs either for 3270 presentation or for use by non-MVS platforms. Our module now contain only the Presentation Logic and the module now obtains the data to display from a new module, ROPEBLRS (Business Logic – Report Status). This Business Logic module is now available for your use as well.

Invocation

The ROPEBLRS module is invoked by an EXEC CICS LINK COMMAREA(RSCMAREA) LENGTH(4112) command. The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

Communication Area

The ROPEBLCM (ROPES Printer Status Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: RSCM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in Figure **Figure 21** on page 114. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area (including Working Storage). It is shown in Figure **Figure 22** on page 115.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

RSCMRRPT
The requested report name. This is the report to be displayed and is equivalent to the reportid field in the command RORS/reportid. If the field is blank, then ROPES will select the first report.

RSCMNEXTR
The next report defined to ROPES is returned in this field. This is the same as the value shown on the RORS panel. It is determined by the cursor position in the ROPES Report Control file at the end of the processing for the current call.

RSCMCNTRQ
The maximum number of printer names to be returned in the COMMAREA. This is a two byte binary number.

RSCMCNTRT
The actual number of printer names returned in the COMMAREA. This is a two byte binary number.

RSCMRC
This two byte binary field contains the return code set by the Business Logic routine. The return codes are defined as follows:

- 0 Processing completed without error
- 3 End of file reached
- 4 There was an Input error processing the ROPES data set. Further details can be determined from the RSCMEIBFN and RSCMEIBRCODE fields

RSCMEIBFN
This field contains the right-most byte from the EIBFN field which indicates the File Control requested function that failed if return code 4 was presented.

RSCMEIBRCODE
This field contains the left-most byte from the EIBRCODE field which indicates the File Control response value if return code 4 was presented.

RSCMRPPNM
The name of the report.

RSCMQLINES
The number of lines on queue for the report. This number is right-justified in the field and edited to remove leading zeros.

RSCMQPAGES
The number of pages on queue for the report. This number is right-justified in the field and edited to remove leading zeros.

RSCMRPPRI
The report's priority. This number is right-justified in the field and edited to remove leading zeros.

RSCMRSTATUS
This is the interpreted value of the report's current status on this printer. These values are documented in the ROPO and ROPS transaction descriptions in the *User's Guide*.

RSCMRFORM
This is the assigned Forms Control Block suffix for the report.

RSCMRPCL
The report's class value.

RSCMDISC
The report's discard value.

RSCMRETPD
The report's retention period.

RSCMAPRTR
An array of up to 1000 four character printer names. These are the printers to which this report has been assigned. The actual number of entries in this list is provided in the RSCMCNTRT field.

User Exit Program

A user exit program is available for your use to further control the data made available to the end-user. The user exit is invoked as follows:

```
EXEC  CICS LINK PROGRAM('ROPERSE1')
 COMMAREA(COMMAREA) LENGTH(=AL2(8)) NOHANDLE.
```

The COMMAREA is defined as follows:

```
COMMAREA DS    0D
RIBADDR  DS    A        ADDRESS OF THE RIB
RETCODE  DS    AL4      RETURN CODE FIELD
```

Use the data provided to determine if the data is to be returned to the user. Set the RETCODE field to zero if the data is to be used, or to any non-zero value if the data is to be suppressed.

```
         ROPEBLCM T=RSCM
***********************************************************************
*                      ---     R O P E S     ---                      *
*             Report Status Business Logic Communication Area         *
***********************************************************************
RSCMAREA     DS 0D    *  Start of COMMAREA.
RSCMCNTL     DS 0D    *  Start of controls segment.
RSCMRRPT     DS CL8   *  Requested report id.
RSCMNEXTR    DS CL8   *  Next eligible report for display.
RSCMCNTRQ    DS AL2   *  Number of entries requested.
RSCMCNTRT    DS AL2   *  Number of entries returned.
RSCMRC       DS AL2   *  Return code.
*                     *  0 - No error.
*                     *  2 - Printer not found.
*                     *  3 - No printers in system.
*                     *  4 - PCR file I/O error.
RSCMEIBFN    DS XL1   *  EIBFN+1 IF RC=4
RSCMEIBRCODE DS XL1   *  EIBRCODE IF RC=4
RSCMCFILL    DS XL24  *  Reserved.
***********************************************************************
* The system has an unknown number of printers. There are, at most,   *
* 1000 printer entries in the COMMAREA.  The actual number present is *
* set in the &P.CNTRT field. Each entry is defined as follows:        *
***********************************************************************
RSCMREPT     DS 0C    *  Start of reports data segment.
RSCMRPPNM    DS CL8   *  Report name.
RSCMQLINES   DS CL8   *  Queued lines.
RSCMQPAGES   DS CL8   *  Queued pages.
RSCMRPPRI    DS CL4   *  Report priority.
RSCMRSTATUS  DS CL12  *  Report status (interpreted).
RSCMRFORM    DS CL4   *  Report FCB suffix.
RSCMRPCL     DS CL1   *  Report class.
RSCMDISC     DS CL1   *  Report discard value.
RSCMRETPD    DS CL4   *  Report retention period.
RSCMRFILL    DS CL14  *  Reserved.
RSCMAPRTR    DS 1000CL4  *  Assigned printers.
RSCMCOMML EQU *-RSCMAREA                                               X
                        *  Length of the COMMAREA.
```

Figure 21 - Report Status Commarea - Assembler

```
****************************************************************** 00010000
*               --- R O P E S         ---                        * 00020000
*     Report Status Business Logic Communication Area            * 00030002
****************************************************************** 00040000
 01  RSCMAREA.                                                      00050002
*  Start of controls segment.                                       00060000
     05  RSCMCNTL.                                                  00070002
*  Requested report id.                                             00080001
         10  RSCMRRPT     PIC X(8).                                 00090002
*  Next eligible report for display.                                00160001
         10  RSCMNEXTR    PIC X(8).                                 00170002
*  File to be processed (ROPERIB or ROPEALR).                       00170101
         10  RSCMFILE     PIC X(8).                                 00171002
*  Requested number of printers.                                    00180002
         10  RSCMCNTRQ    PIC 9(4) COMP.                            00190002
*  Retrieved number of printers.                                    00240002
         10  RSCMCNTRT    PIC 9(4) COMP.                            00250002
*  Return code.                                                     00260000
         10  RSCMRC       PIC 9(4) COMP.                            00270002
             88  RSCM_NO_ERROR                  VALUE 0.            00280002
             88  RSCM_REPORT_NOT_FOUND          VALUE 2.            00300002
             88  RSCM_NO_END_OF_FILE            VALUE 3.            00310002
             88  RSCM_INPUT_FILE_IOERROR        VALUE 4.            00320002
*  EIBFN+1 IF RC=4                                                  00340000
         10  RSCMEIBFN    PIC X(1).                                 00350002
*  EIBRCODE IF RC=4                                                 00360000
         10  RSCMEIBRCODE PIC X(1).                                 00370002
*  Reserved.                                                        00380000
         10  RSCMCFILL    PIC X(24).                                00390002
*  Start of data segment.                                           00400001
     05  RSCMDETAIL.                                                00410002
*  Returned report name.                                            00420001
         10  RSCMRPPNM    PIC X(8).                                 00430002
*  Queued lines.                                                    00440001
         10  RSCMQLINES   PIC X(8).                                 00450002
*  Queued pages.                                                    00451001
         10  RSCMQPAGES   PIC X(8).                                 00452002
*  Report priority.                                                 00453001
         10  RSCMRPPRI    PIC X(4).                                 00454002
*  Report status (interpreted).                                     00455001
         10  RSCMRSTATUS  PIC X(12).                                00456002
*  Report FCB suffix.                                               00457001
         10  RSCMRFORM    PIC X(4).                                 00458002
*  Report class.                                                    00459001
         10  RSCMRPCL     PIC X(1).                                 00459102
*  Report discard option.                                           00460001
         10  RSCMDISC     PIC X(1).                                 00470002
*  Report retention period.                                         00480001
         10  RSCMRETPD    PIC X(4).                                 00490002
*  Reserved                                                         00800000
         10  RSCMRFILL    PIC X(14).                                00810002
*  Start of the printers array.                                     00820002
         10  RSCMAPRTR    PIC X(4)                                  00830002
                          OCCURS 0 TO 1000 TIMES                    00840002
                          DEPENDING ON RSCMCNTRT.                   00850002
```

Figure 22 Report Status Commarea - COBOL

System Status Business Logic Service

The ROSS transaction processor program R51SSTAT presents a display to your users. Also, the Presentation Logic in this module is organized for 3270 Model 2 terminals only, and may not meet your needs either for 3270 presentation or for use by non-MVS platforms. Our module now contain only the Presentation Logic and the module now obtains the data to display from a new module, ROPEBLSS (Business Logic – System Status). This Business Logic module is now available for your use as well.

Invocation

The ROPEBLSS module is invoked by an EXEC CICS LINK COMMAREA(SSCMAREA) LENGTH(372) command. The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

Communication Area

The ROPEBLCM (ROPES Printer Status Communication Area) is available in an assembler version. The assembler version is a macro that allows you to designate the field name prefix (default: SSCM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in **Figure 23** on page 118. The COBOL version is shown in **Figure 24** on page 119.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function.

SSCMRC

This two byte binary field contains the return code set by the Business Logic routine. The return codes are defined as follows:

- 0 Processing completed without error
- 1 RIB file I/O error. There was an Input error processing the ROPES data set. Further details can be determined from the SSCMEIBFN and SSCMEIBRCODE fields
- 2 PCR file I/O error. There was an Input error processing the ROPES data set. Further details can be determined from the SSCMEIBFN and SSCMEIBRCODE fields

SSCMEIBFN

This field contains the right-most byte from the EIBFN field which indicates the File Control requested function that failed if return code 4 was presented.

SSCMEIBRCODE

This field contains the left-most byte from the EIBRCODE field which indicates the File Control response value if return code 4 was presented.

SSCMROPESStat

The overall ROPES Status. The value will be either “STARTED” or “STOPPED”.

SSCMReptCount

The number of reports defined to ROPES. This number is right-justified in the field and edited to remove leading zeros.

SSCMPrtCount

The number of printers defined to ROPES. This number is right-justified in the field and edited to remove leading zeros.

SSCMLineCount

The total number of lines on queue. This number is right-justified in the field and edited to remove leading zeros.

SSCMPageCount

The total number of pages on queue. This number is right-justified in the field and edited to remove leading zeros.

SSCMReorgID

This is the identification value assigned to the scheduled on line Queue Reorganization request when it is made.

SSCMReorgTime

The time of day when the on line Queue

Reorganization task should be run.

SSCMAltStat
The status of the ROPES Alternate Facility.

SSCMAltReports
The number of reports defined to the ROPES Alternate Facility. This number is right-justified in the field and edited to remove leading zeros.

SSCMAltLines
The total number of lines on the Alternate Facility queue. This number is right-justified in the field and edited to remove leading zeros.

SSCMaltPages
The total number of pages on the Alternate Facility queue. This number is right-justified in the field and edited to remove leading zeros.

SSCMXferStat
The status of the ROPES Spool Transfer Utility.

SSCMStbyStat
The status of the ROPES Standby Facility.

SSCMExitName
The name of the Task Exit for screen capture.

SSCMVersion
The current ROPES Version number.

SSCMLevel
The current ROPES Level number.

SSCMExpire
The date on which the ROPES License expires. It may also say “never” for perpetual licenses.

SSCMLicCPUS
A list of up to 10 CPU id’s of those CPUs that are licensed to run ROPES. The leading digits of the CPU id may be “x” to indicate that the LPAR number and engine sequence number are not important.

SSCMThisModel
This field contains the current machine number (4 digits).

SSCMThisCPU
This field contains the CPU serial number of the current CPU.

SSCMCICSVer
This field contains the interpreted value of the CICS Version Number.

SSCMOpSysRel
This field contains the interpreted operating system release.

SSCMOpSysVer
This field contains the interpreted operating system version.

SSCMMessage
This field contains blanks or a diagnostic message from the service.

```
         ROPEBLCM T=SSCM
***********************************************************************
*                     ---     R O P E S     ---                       *
*           System Status Business Logic Communication Area           *
***********************************************************************
SSCMAREA     DS 0D    *  Start of COMMAREA.
SSCMCNTL     DS 0C    *  Start of controls segment.
SSCMRC       DS AL2   *  Return code.
*                     *  0 - No error.
*                     *  1 - RIB file I/O error.
*                     *  2 - PCR file I/O error.
*                     *  3 -
*                     *  4 -
*                     *  5 -
SSCMEIBRCODE DS XL1   *  EIBRCODE IF RC=4
SSCMCFILL    DS XL16  *  Reserved.
SSCMOPSDATA  DS 0C    *  Start of operations status data segment.
SSCMROPEStat DS CL8   *  ROPES overall status.
SSCMReptCount DS CL6  *  Count of defined reports.
SSCMPrtCount DS CL6   *  Count of defined printers.
SSCMLineCount DS CL11 *  Lines on queue.
SSCMPageCount DS CL7  *  Pages on queue.
SSCMReorgID  DS CL8   *  Queue reorganization id.
SSCMReorgTime DS CL5  *  Queue reorganization time.
SSCMAltStat  DS CL8   *  Alternate facility status.
SSCMAltRepts DS CL6   *  Alternate facility report count.
SSCMAltLines DS CL11  *  Alternate facility queued lines count.
SSCMAltPages DS CL7   *  Alternate facility queued pages count.
SSCMXferStat DS CL8   *  Spool Transfer status.
SSCMStbyStat DS CL8   *  Standby Facility status.
SSCMExitName DS CL8   *  Task exit for screen capture.
SSCMPRDDATA  DS 0C    *  Start of product status data segment.
SSCMVersion  DS CL4   *  ROPES Version.
SSCMLevel    DS CL2   *  ROPES Level.
SSCMExpire   DS CL13  *  ROPES License Expiration date.
SSCMLicCPUS  DS CL60  *  Licensed CPU numbers (10x6).
SSCMThisModel DS CL4  *  Current CPU model.
SSCMThisCPU  DS CL6   *  Current CPU serial number.
SSCMCICSVer  DS CL16  *  Indicated CICS Version.
SSCMCICSRel  DS CL5   *  Indicated CICS Release.
SSCMOpSysRel DS CL8   *  Indicated system version and release.
SSCMOpSysVer DS CL6   *  Indicated system version and release.
SSCMMessage  DS CL72  *  Diagnostic message.
SSCMDFILL    DS CL49  *  Reserved.
SSCMCOMML EQU *-SSCMAREA                                               X
                       *  Length of the COMMAREA.
```

Figure 23 System Status Commarea - Assembler

```
*****************************************************************
*                   ---     R O P E S     ---                   *
*         System Status Business Logic Communication Area       *
*****************************************************************
*  Start of COMMAREA.
 01  SSCMAREA.
*  Start of controls segment.
     05  SSCMCNTL.
*  Return code.
         10  SSCMRC           PIC S9(4) COMP.
*                                0 - No error.
*                                1 - RIB file I/O error.
*                                2 - PCR file I/O error.
*  EIBFN+1 IF RC=4
         10  SSCMEIBFN        PIC X.
*  EIBRCODE IF RC=4
         10  SSCMEIBRCODE     PIC X.
*  Reserved.
         10  SSCMCFILL        PIC X(16).
*  Start of Operations Data.
     05  SSCMOPSDATA.
*  ROPES overall status.
         10  SSCMROPEStat     PIC X(8).
*  Count of defined reports.
         10  SSCMReptCount    PIC X(6).
*  Count of defined printers.
         10  SSCMPrtCount     PIC X(6).
*  Lines on queue.
         10  SSCMLineCount    PIC X(11).
*  Pages on queue.
         10  SSCMPageCount    PIC X(7).
*  Queue reorganization id.
         10  SSCMReorgID      PIC X(8).
*  Queue reorganization time.
         10  SSCMReorgTime    PIC X(5).
*  Alternate facility status.
         10  SSCMAltStat      PIC X(8).
*  Alternate facility report count.
         10  SSCMAltRepts     PIC X(6).
*  Alternate facility queued lines count.
         10  SSCMAltLines     PIC X(11).
*  Alternate facility queued pages count.
         10  SSCMAltPages     PIC X(7).
*  Spool Transfer status.
         10  SSCMXferStat     PIC X(8).
*  Standby Facility status.
         10  SSCMStbyStat     PIC X(8).
*  Task exit for screen capture.
         10  SSCMExitName     PIC X(8).
*  Start of product status data segment.
   05  SSCMPRDDATA.
*  ROPES Version.
       10  SSCMVersion        PIC X(4).
*  ROPES Level.
       10  SSCMLevel          PIC X(2).
*  ROPES License Expiration date.
       10  SSCMExpire         PIC X(13).
*  Licensed CPU numbers (10x6).
       10  SSCMLicCPUS        PIC X(60).
*  Current CPU model.
       10  SSCMThisModel      PIC X(4).
*  Current CPU serial number.
       10  SSCMThisCPU        PIC X(6).
*  Indicated CICS Version.
       10  SSCMCICSVer        PIC X(16).
*  Indicated CICS Release.
       10  SSCMCICSRel        PIC X(5).
*  Indicated system version and release.
       10  SSCMOpSysRel       PIC X(8).
*  Indicated system version and release.
       10  SSCMOpSysVer       PIC X(6).
*  Diagnostic message.
       10  SSCMMessage        PIC X(72).
*  Reserved.
   05  SSCMDFILL              PIC X(49).
```

Figure 24 System Status COMMAREA - COBOL

Business Logic Services using Containers

Introduction

Beginning with Version 10.0 we started the process of separating the business logic of the some of the ROPES modules from the Presentation Logic. In Version 12.0 we are starting the process of extending this implementation to use the CICS Business Transaction Services facilities to overcome the limitations on the size of the COMMAREA. The Business Logic modules are Command Level, COMMAREA driven programs with a fully documented user interface that are being made available to all users who wish to provide fully integrated access to the ROPES information within their applications, or, through the various facilities of DPL, ECI, EXCI, the CICS Clients and/or the CICS Transaction Gateway, make this information available to users on non-z/OS platforms as required. You can now begin to integrate ROPES into your environment without having to give up control to ROPES modules that may allow unplanned user interactions. We want to make these changes so that you do not need a knowledge of the structure of the ROPES data files, which can, and will, change over time. The data provided in the COMMAREA will be used to control the processing, but the ROPES data will be returned in a CICS BTS Process Container, interpreted and formatted by ROPES in a consistent and usable way, so that you do not need to know the internal format of the data, the means by which status values are determined, or how data records in the ROPES files are interrelated. The Container used by the modules supplied can be defined using the BTSCNTL member of the ROPES.SOURCE library. You can use this as a model for other containers you may choose to define.

Our current implementation, which will function as a pilot for this development, consists of two service routines, ROPECNNS and ROPECNRL, which are documented here, and a demonstration program, ROPECLPN, which uses this service routine to produce a variation of the ROPES Network Status display, similar to the output of the ROPN transaction. All of these modules are provided in source form to illustrate the process and to help you implement this new feature in your systems. Our intention is to continue to provide new services in this way to support your enterprise's implementation of a Service Oriented Architecture.

It is our intention to extend this support to as many modules as is practical, and new ROPECLxx modules will be introduced, from time to time, in maintenance releases, enhancement releases and in future versions.

Network Status Business Logic Service using Containers

The ROCN transaction processor program ROPECLPN presents a display to the user using BMS SEND TEXT ACCUM logic. The Presentation Logic in this module is organized for 3270 Model 2 terminals only, and may not meet your needs either for 3270 presentation or for use by non-MVS platforms. The module now contains only the Presentation Logic and the module now obtains the data to display from a new module, ROPECNNS (Container Business Logic – Network Status). This Business Logic module is now available for your use as well.

Invocation

The ROPECNNS module is invoked by an EXEC CICS LINK COMMAREA(NSCLAREA) LENGTH(128) command. The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

Communication Area

The ROPECLCM (ROPES Container Logic Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: NSCM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in Figure **Figure 25** on page 124. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area (including Working Storage). It is shown in Figure **Figure 26** on page 124.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

NSCMRPCR

The requested printer id. This is the printer to be displayed and is equivalent to the prtr field in following command: ROPN/prtr. If the field is blank, then ROPES will select the first printer.

NSCMCNTRT

The actual number of printers returned in the COMMAREA. This is a two byte binary number.

NSCMPROCESS

This thirty-six byte character field contains the name of the BTS Process to be associated with the container that will be created and returned to the calling program. You may build the Process Name in any way that is suitable to your application's use.

NSCMRC

This two byte binary field contains the return code set by the Business Logic routine. The return codes are defined as follows:

0 Processing completed without error

3 There are no eligible printers for processing

NSCMErrMsg

This sixty byte character field will contain an error or informational message from the service module if an error has occurred. It is composed of the following two fields.

NSCMEText

This twenty character field contains the message prefix which will identify the message.

NSCMFileEMsg

This forty character field contains the message proper.

Container Area

The ROPECLCM (ROPES Container Logic Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: NSCO) and can map a dynamically acquired (GETMAINed) storage area. It is shown in Figure **Figure 27** on page 124. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area. It is shown in Figure **Figure 28** on page 125. The Container contains zero or more returned entries, based on the content of the NSCMCNTRT value. Each entry contains a full set of fields as described next.

Container Field Names and Their Uses

The Container fields for one entry are defined here. There will be zero or more of these entries in the container. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

NSCOContainerTMID

The name of the printer.

NSCOContainerACLASS

This is the list of active classes found in the printer controls.

NSCOContainerSTATUS

This is the interpreted value of the printer's current status. These values are documented in the ROPO or ROPS transaction descriptions in the *ROPES User's Guide*.

NSCOContainerLASTMSG

This is the last message issued by ROPES regarding the printer's status. This is the value that normally appears on the second line of the ROPO or ROPS panel.

NSCOContainerFORM

This is the current Forms Control Block Suffix that represents the form that ROPES considers the current form in or for this printer.

NSCOContainerRPPNM
The name of the current report, if any, or blank.

NSCOContainerQLINES
The number of lines on queue for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCOContainerQPAGES
The number of pages on queue for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCOContainerPLINES
The number of lines printed at this printer for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCOContainerPPAGES
The number of pages printed at this printer for the current report. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCOContainerRPPRI
The current report's priority value at this printer. This field reflects the printer priority override value, if any. This number is right-justified in the field and edited to remove leading zeros. If there is no current report, this field is blank.

NSCOContainerRSTATUS
This is the interpreted value of the current report's current status on this printer. These values are documented in the ROPO and ROPS transaction descriptions in the *User's Guide*. If there is no current report, this field is blank.

NSCOContainerRFORM
This is the assigned Forms Control Block suffix for the current report. If there is no current report, this field is blank.

NSCOContainerRPCL
The current report's class value. If there is no current report, this field is blank.

NSCOContainerFILL
A filler area for future use.

User Exit Program

A user exit program is available for your use to further control the data made available to the end-user. The user exit is invoked as follows:

```
EXEC  CICS LINK PROGRAM('ROPENSE1')
 COMMAREA(COMMAREA) LENGTH(=AL2(8)) NOHANDLE.
```

The COMMAREA is defined as follows:

```
COMMAREA DS     0D
PCRADDR  DS     A        ADDRESS OF THE PCR
RETCODE  DS     AL4      RETURN CODE FIELD
```

Use the data provided to determine if the data is to be returned to the user. Set the RETCODE field to zero if the data is to be used, or to any non-zero value if the data is to be suppressed.

```
NSCLCOMM DS    0D
         ROPECLCM T=NSCM
***********************************************************************
*                    ---     R O P E S     ---                        *
*            Network Status Container Logic Communication Area        *
***********************************************************************
NSCMAREA     DS 0D    *  Start of COMMAREA.
NSCMRPCR     DS CL4   *  Requested printer id.
NSCMCNTRT    DS AL2   *  Number of entries returned.
NSCMPROCESS  DS CL36  *  Process Name.
NSCMRC       DS AL2   *  Return code.
NSCMErrMsg   DS 0CL60 *  Error Message Area
NSCMEText    DS CL20  *     Error Prefix
NSCMFileEMsg DS CL40  *     Error Message
NSCMCFILL    DS XL24  *  Reserved.
NSCMCOMML EQU *-NSCMAREA                                              X
                        *  Length of the COMMAREA.
```

Figure 25 - Network Status Commarea - Assembler

```
      ***************************************************************
      *                ---     R O P E S     ---                    *
      *       Network Status Container Logic Communication Area     *
      ***************************************************************
       01  NSCMAREA.
      *  Requested printer id.
           05  NSCMRPCR             PIC X(4).
      *  Number of entries returned.
           05  NSCMCNTRT            PIC S9(4) COMP.
      *  Process Name.
           05  NSCMPROCESS          PIC X(36).
      *  Return code.
           05  NSCMRC               PIC S9(4) COMP.
      *  Error Message Area
           05  NSCMErrMsg.
      *  Error Prefix
               10  NSCMEText        PIC X(20).
      *  Error Message
               10  NSCMFileEMsg     PIC X(40).
      *  Reserved.
           05  NSCMCFILL            PIC X(24).
```

Figure 26 Network Status Commarea - COBOL

```
         ROPECLCM T=NSCO
***********************************************************************
*                    ---     R O P E S     ---                        *
*                 Network Status Container Entry Mapping              *
***********************************************************************
NSCOContainerData    DS 0D    *  Start of Container.
NSCOContainerTMID    DS CL4   *  Printer name.
NSCOContainerACLAS   DS CL8   *  Active classes.
NSCOContainerSTATUS DS CL12  *  Printer status.
NSCOContainerLASTMSG DS CL60 *  Last message.
NSCOContainerFORM    DS CL4   *  Last FCB used with this printer.
NSCOContainerRPPNM   DS  CL8  *  Active report, if any.
NSCOContainerQLINES DS  CL8  *  Queued lines.
NSCOContainerQPAGES DS  CL8  *  Queued pages.
NSCOContainerPLINES DS  CL8  *  Printed lines.
NSCOContainerPPAGES DS  CL8  *  Printed pages.
NSCOContainerRPPRI   DS  CL4  *  Report priority.
NSCOContainerRSTATUS DS CL12 *  Report status.
NSCOContainerRFORM   DS  CL4  *  Report FCB suffix.
NSCOContainerRPCL    DS  CL1  *  Report class.
NSCOContainerFILL    DS XL24  *  Reserved.
NSCOContainerLL      EQU *-NSCOContainerData *  Length of the entry.
```

Figure 27 - Network Status Container - Assembler

```
************************************************************
*                ---     R O P E S     ---                 *
*            Network Status Container Entry Mapping         *
************************************************************
*  Start of Container.
 01  NSCOContainerData.
*  Printer name.
     05  NSCOContainerTMID       PIC X(4).
*  Active classes.
     05  NSCOContainerACLAS      PIC X(8).
*  Printer status.
     05  NSCOContainerSTATUS     PIC X(12).
*  Last message.
     05  NSCOContainerLASTMSG    PIC X(60).
*  Last FCB used with this printer.
     05  NSCOContainerFORM       PIC X(4).
*  Active report, if any.
     05  NSCOContainerRPPNM      PIC X(8).
*  Queued lines.
     05  NSCOContainerQLINES     PIC X(8).
*  Queued pages.
     05  NSCOContainerQPAGES     PIC X(8).
*  Printed lines.
     05  NSCOContainerPLINES     PIC X(8).
*  Printed pages.
     05  NSCOContainerPPAGES     PIC X(8).
*  Report priority.
     05  NSCOContainerRPPRI      PIC X(4).
*  Report status.
     05  NSCOContainerRSTATUS    PIC X(12).
*  Report FCB suffix.
     05  NSCOContainerRFORM      PIC X(4).
*  Report class.
     05  NSCOContainerRPCL       PIC X(1).
*  Reserved.
     05  NSCOContainerFILL       PIC X(24).
```

Figure 28 Network Status Container - COBOL

Report Status Business Logic Service using Containers

Invocation

The ROPECNRL module is invoked by an EXEC CICS LINK COMMAREA(RLCMAREA) LENGTH(128) command. The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

Communication Area

The ROPECLCM (ROPES Container Logic Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: RLCM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in **Figure 29** on page 128. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area (including Working Storage). It is shown in **Figure 30** on page 128.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

RLCMRRPT
The requested report name. This is the report to be displayed first, and is equivalent to the reportid field in following command: RORL/reportid. If the field is blank, then ROPES will select the first report.

RLCMFILE
This is the name of the file to process. ROPERIB indicates the Primary Facility RIB file is to be processed. ROPEALR indicates that the Alternate Facility RIB file is to be processed. This field is not currently supported.

RLCMCNTRT
The actual number of reports returned in the COMMAREA. This is a two byte binary number.

RLCMPROCESS
This thirty-six byte character field contains the name of the BTS Process to be associated with the container that will be created and returned to the calling program. You may build the Process Name in any way that is suitable to your application's use.

RLCMRC
This two byte binary field contains the return code set by the Business Logic routine. The return codes are defined as follows:
- 0 Processing completed without error
- 2 File Open/Close error
- 3 End of file
- 4 Input File I/O error

RLCMEMPTY
This is a Y/N flag that controls whether or not empty reports are to be returned.

RLCMSERIAL
This flag indicates whether or not the input file should be processed dynamically with ENQ/OPEN/CLOSE logic. This field is not currently supported.

RLCMErrMsg
This sixty byte character field will contain an error or informational message from the service module if an error has occurred. It is composed of the following two fields.

RLCMEText
This twenty character field contains the message prefix which will identify the message.

RLCMFileEMsg
This forty character field contains the message proper.

RLCMCFILL
Reserved for future use.

Container Area

The ROPECLRS (ROPES Report Status Container Logic Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: RLCO) and can map a dynamically acquired (GETMAINed) storage area. It is shown in **Figure 31** on page 129. The COBOL version is shown in **Figure 32** on page 129. The Container contains zero or more returned entries, based on the content of the RLCMCNTRT value. Each entry contains a full set of fields as described next.

Container Field Names and Their Uses

The Container fields for one entry are defined here. There will be zero or more of these entries in the container. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

RLCOContainerRPTID
The name of the report.

RLCOContainerQLINES
This is the number of lines on queue for the report.

RLCOContainerQPAGES
This is the number of pages on queue for the report.

RLCOContainerRPPRI
The report's priority value.

RLCOContainerRForm
This is the FCB suffix value assigned to the report. It can be blank.

RLCOContainerRETPD
The assigned Retention Period, in hours, for the report. This is only meaningful if the Discard Option implies expiration processing.

RLCOContainerCLASS
This is the assigned class of the report.

RLCOContainerDISCARD
This is the assigned Discard Option for the report.

RLCOContainerFILL
A filler area for future use.

User Exit Program

A user exit program is available for your use to further control the data made available to the end-user. The user exit is invoked as follows:

```
EXEC  CICS LINK PROGRAM('ROPERLE1')
 COMMAREA(COMMAREA) LENGTH(=AL2(8)) NOHANDLE.
```

The COMMAREA is defined as follows:

```
COMMAREA DS     0D
PCRADDR  DS     A        ADDRESS OF THE PCR
RETCODE  DS     AL4      RETURN CODE FIELD
```

Use the data provided to determine if the data is to be returned to the user. Set the RETCODE field to zero if the data is to be used, or to any non-zero value if the data is to be suppressed.

```
         ROPECLCM T=RLCM
***********************************************************************
*                    ---     R O P E S     ---                        *
*             Report List Container Logic Communication Area          *
***********************************************************************
RLCMAREA     DS 0D    *  Start of COMMAREA.
RLCMRRPT     DS CL8   *  Requested report id.
RLCMFILE     DS CL8   *  Input file name to use.
RLCMCNTRT    DS AL2   *  Number of entries returned.
RLCMPROCESS  DS CL36  *  Process Name.
RLCMRC       DS AL2   *  Return code.
*                     *  0 - No error.
*                     *  2 - File Open/Close error.
*                     *  3 - End of file.
*                     *  4 - Input file I/O error.
RLCMEMPTY    DS CL1   *  Y/N flag to control use of empty reports
RLCMSERIAL   DS CL1   *  Dynamic ENQ/OPEN/CLOSE requested
RLCMErrMsg   DS 0CL60 *  Error Message Area
RLCMEText    DS CL20  *     Error Prefix
RLCMFileEMsg DS CL40  *     Error Message
RLCMCFILL    DS XL4   *  Reserved.
RLCMCOMML EQU *-RLCMAREA                                               X
                        *  Length of the COMMAREA.
```

Figure 29 Report List Commarea - Assembler

```
     ***************************************************************
     *                 ---     R O P E S     ---                   *
     *       Report Status Container Logic Communication Area      *
     ***************************************************************
      01  RLCMAREA.
     *  Requested report id.
          05  RLCMRRPT       PIC X(8).
     *  File to process.
          05  RLCMFILE       PIC X(8).
     *  Number of entries returned.
          05  RLCMCNTRT      PIC S9(4) COMP.
     *  Process Name.
          05  RLCMPROCESS    PIC X(36).
     *  Return code.
          05  RLCMRC         PIC S9(4) COMP.
     *  Empty Report Flag.
          05  RLCMEMPTY      PIC X.
     *  File Serialization Flag.
          05  RLCMSERIAL     PIC X.
     *  Error Message Area
          05  RLCMErrMsg.
     *  Error Prefix
              10  RLCMEText  PIC X(20).
     *  Error Message
              10  RLCMFileEMsg PIC X(40).
     *  Reserved.
          05  RLCMCFILL      PIC X(24).
```

Figure 30 Report List Commarea - COBOL

```
         ROPECLCM T=RLCO
***********************************************************************
*                     ---     R O P E S     ---                       *
*                 Report List Container Entry Mapping                 *
***********************************************************************
RLCOContainerData    DS 0D    *  Start of Container.
RLCOContainerRPTID   DS CL8   *  Report name.
RLCOContainerQLINES  DS CL8   *  Queued lines.
RLCOContainerQPAGES  DS CL8   *  Queued pages.
RLCOContainerRPPRI   DS CL4   *  Priority.
RLCOContainerRFORM   DS CL4   *  FCB suffix.
RLCOContainerRETPD   DS CL4   *  Retention Period.
RLCOContainerCLASS   DS CL1   *  Class.
RLCOContainerDISCARD DS CL1   *  Discard option.
RLCOContainerFILL    DS XL10  *  Reserved.
RLCOContainerLL      EQU *-RLCOContainerData *  Length of the entry.
```

Figure 31 Report List Container - Assembler

```
     ***************************************************************
     *                 ---     R O P E S     ---                   *
     *            Network Status Container Entry Mapping           *
     ***************************************************************
     *  Start of Container.
       01  RLCOContainerData.
     *  Printer name.
           05  RLCOContainerRPTID      PIC X(8).
     *  Queued lines.
           05  RLCOContainerQLINES     PIC X(8).
     *  Queued pages.
           05  RLCOContainerQPAGES     PIC X(8).
     *  Report priority.
           05  RLCOContainerRPPRI      PIC X(4).
     *  Report FCB suffix.
           05  RLCOContainerRFORM      PIC X(4).
     *  Retention period.
           05  RLCOContainerRETPD      PIC X(4).
     *  Report class.
           05  RLCOContainerCLASS      PIC X(1).
     *  Discard option.
           05  RLCOContainerDISCARD    PIC X(1).
     *  Reserved.
           05  RLCOContainerFILL       PIC X(10).
```

Figure 32 Report List Container - COBOL

Communicating With The ROPES External Interface Client

The Interface For The ROPES External Interface Client

Prior to ROPES Version 11, the interface for the ROPES External Interface Client program was undocumented. Communication with the External Interface Client program (ROPEXPLR) was accomplished strictly through the use of the ROPES Utility Driver Program. Two utility functions which "LIST PRINTER" or "LIST REPORT" property details could be executed through the ROPES Utility Driver module (ROPEUTIL). Please see the ROPES Utilities Guide for more information about these utility functions.

A new feature of ROPES Version 11, called the File Server Program, required enhancement to the ROPES External Interface Client. The enhancement allows the ROPES External Interface to be used to access ROPES files under CICS using the File Server Program. By documenting the interface with the ROPES External Interface Client it becomes possible to allow batch application programs to call the External Interface Client to obtain access to ROPES files (allocated and open to CICS) using the File Server Program.

The File Server Program Enhancement

The ROPES File Server program provides a common interface for accessing ROPES files. It eliminates alot of the repetitive coding required to access ROPES files by providing a flexible interface that does much of the "overhead" work required to access ROPES files through CICS. Most of the standard CICS File Control options are provided by the new File Server interface program. For example, the new File Server program supports the use of Record Level Sharing options, tokens, and request ids. It is also possible to obtain comprehensive information about the file that is being processed. Details on the use and installation of the File Server Program can be found after this chapter in the Programmer's Guide.

The ROPES External Interface enhancements provide support for calling the File Server Program from a batch application program. This means CICS files may be accessed by batch application programs designed to communicate with the File Server Program through the ROPES External Interface feature. A sample assembler batch program has been provided to demonstrate and verify the installation of this new feature of ROPES. The sample program is called ROPEXFLE and can be found in the ROPES Distribution Source data set. Sample JCL is also provided to run the sample program. The execution JCL for program ROPEXFLE can be found in member RUNXFLE in the ROPES Distribution Source Dataset.

The ROPES External Interface feature was originally implemented with the ROPES Utility Driver program ROPEUTIL. At that time several utility functions were provided to LIST PRINTERS and LIST REPORTS. Using the External Interface Feature these utility functions were able to access CICS files from batch and produce detailed information about ROPES reports and printers.

The new sample program ROPEXFLE interfaces with the ROPES External Interface differently. It does not use the ROPES Utility Driver program (ROPEUTIL), but rather calls the External Client module (ROPEXPLR) directly using a communication area. The communication area is defined by an assembler macro called ROPEXPCA. ROPEXPCA can be found in the ROPES Distribution Source data set.

When the File Server Program functions are used with the ROPES External Interface, the communication area for the File Server Program (ROPEFSCA) is merged with the External Interface communication area (ROPEXPCA). This enables file oriented information to be returned to the calling batch application in fields of the File Server Program communication area. The merge occurs at label "XPCAPRCA" which marks the start of a variable size area that normally contains record data returned by the ROPES External Interface Server program (ROPEXPLS). The size of the variable area is determined by the overall size of the External Interface communication area (ROPEXCPA). Generally, the ROPES External Interface communication area must be set at approximately 28K in size. This would provide enough storage to contain one large Printer Control Record. Due to

CICS restrictions, the communication area may not exceed 32K. In most cases, ROPES records will be considerably smaller allowing multiple records to be passed in the communication area in one call. Details of this process are described in the chapter on the ROPES File Server Program in this Guide.

ROPES External Interface Communication Area

The ROPES External Interface Client Program (ROPEXPLR) uses the ROPEXPCA communication area to interface with calling programs. In the first version of the ROPES External Interface, the utility driver program (ROPEUTIL) called ROPEXPLR directly and used the ROPEXPCA communication area to support that communication. Until now, no communication between ROPEXPLR and an end user application was required. With the new ROPES File Server Program enhancement, it becomes necessary for the end user application to communicate directly with the ROPES External Interface Client program (ROPEXPLR) if ROPES files are to be accessed while CICS is active.

The sample batch end user program, ROPEXFLE, passes requests/options for the ROPES File Server Program through the External Interface Client's communication area (ROPEXPCA). A subset of the fields in the ROPEXPCA communication area duplicate fields/options which can be found in the File Server Program's communication area (ROPEFSCA). The batch External Interface Client program (ROPEXPLR) moves these options forward to the File Server Program's communication area (ROPEFSCA). In this manner, the File Server Program receives the request from the ROPES External Interface and then passes back the appropriate fields, responses, and data.

Please find an example of the ROPEXPCA communication area in Figure **1** on page 138.

The sample File Server application (ROPEXFLE) obtains a storage area to accommodate the ROPEXPCA communication area. The amount of storage can be specified by the JCL execution ("STEP") statement PARM option COMWSLEN. Typically, the value of the COMWSLEN option is at least 26K and may be higher but cannot exceed 32K. Note that the value for COMWSLEN may be specified in bytes as well as "K". Please examine JCL member RUNXFLE, which is located in the ROPES Distribution Source data set, for a sample of the COMWSLEN PARM value.

In the ROPEXPCA communication area shown above, the prefix for every communication area field is specified by the "P=XPC" option on the MACRO statement. Thus, it is possible to have more than one copy of the communication area in a given application program. However, a separate MRO connection would have to be used for each ROPEXPCA communication area, and a separate mirror transaction would have to be established to receive the requests in CICS. The sample program ROPEXFLE has not been designed to make use of multiple MRO links.

ROPES External Interface Program Communication Specific Fields

ROPEXPCA communication area fields between field XPCCAREA and XPCAPCOM are strictly for ROPES External Interface options. To establish the External Interface Link it will be necessary to call program ROPEXPLR with an "OPENLINK" function. The fields that must be set to accomplish this are; XPCSRVPG, XPCSVTRN, XPCCLNTP, XPCTRGSY, and XPCFNCTN. The XPCUSRID field may be set but is optional.

The External Interface Link is terminated by issuing a call to ROPEXPLR with the "CLOSELINK" function set in field XPCFNCTN. All requests to the File Server Program occur in between these calls to ROPEXPLR, and are accomplished using the "SENDREQ" function which is specified in the XPCFNCTN field. All calls for the File Server program will have the value "SENDREQ" set in the XPCFNCTN field.

Please find below a detailed explanation of the communication area fields required for communicating with the External Interface Client Program (ROPEXPLR).

XPCSRVPG

Specifies the name of the ROPES External Interface Server Program. By default, the program is named ROPEXPLS. The value "ROPEXPLS" must be specified for this field unless the CICS program definition for this program has been modified. Modification of the name is not recommended but can be accomplished with assistance from Technical Support.

XPCSVTRN

Specifies the name of the mirror transaction id that will be associated with the ROPES External Interface Server program (ROPEXPLS). Note, that the program executed by the transaction is DFHMIRS. The CICS mirror program calls the External Interface Server program and passes it the communication area received from the External Interface Client program (ROPEXPLR). The default name for the ROPES server mirror transaction is "ROXP". Unless it is necessary to change the name, please specify the value "ROXP" in this communication area field. Changing the transaction name may require some assistance from Technical Support.

XPCCLNTP

Specifies the name of the ROPES External Interface Client program. For ROPES the name of this program is ROPEXPLR. Please specify the value "ROPEXPLR" for this communication area field unless it is necessary to change the name of the client program.

XPCTRGSY

Specifies the APPLID of the target CICS region. This would specify the CICS region where ROPES is running and perhaps more specifically the CICS region that owns the ROPES files. The default value for this field is "DBDCCICS". The default value will be supplied automatically if XPCTRGSY is found to be blank. Otherwise the specified APPLID is used. The appropriate APPLID can be supplied by the CICS systems programmer.

XPCFNCTN

Specifies the function that must be performed by the ROPES External Interface Client Program. There are three possible values; OPENLINK, CLOSELINK, and SENDREQ. The values "OPENLINK" and "CLOSELINK" must be used to establish and terminate the CICS External Interface link (using MRO) with CICS. The "SENDREQ" value is used to send data or send a request over the External Interface Link. To communicate with the ROPES File Server Program, it will be necessary to have one or more calls to the ROPEXPLR program using the "SENDREQ" value. Note, that the XPCFNCTN field is 10 bytes in length and must be padded with blanks.

XPCUSRID

Specifies the name of the RACF userid to be associated with the External Interface Link session. In most cases, this field is left blank and the CICS EXCI Interface adopts the userid associated with the batch job that executes the External Interface client program (ROPEXPLR in this case). This depends on whether the MRO connection specifies "ATTACHSEC(IDENTIFY)" or "ATTACHSEC(LOCAL)". Please consult the CICS RACF Security Guide for information about link security for MRO connections. Guidelines can also be found in the CICS Internet and External Interfaces Guide.

File Server Program And Application Specific Communication Area Fields

For all "SENDREQ" functions, the ROPES External Interface communication area fields starting at XPCAPCOM and below are needed. However, not all of the "application" commarea fields are necessary for communication with the File Server Program. It will be useful to remember that the External Interface is also driven by the ROPES Utility Driver which does make use of other fields in this communication area. Request types associated with the Utility Driver program are "LIST PRINTERS" and "LIST REPORTS". These utility functions can be found in the ROPES Utilities Guide under “The ROPES External Interface Utility Functions.”

XPCAPRQT

Specifies the request type that is to be sent to the External Interface Server program. The utility driver program sends "LIST PRINTERS" and "LIST REPORTS" in this commarea field when these utility functions are invoked using the utility driver. Request types that may be used with the File Server Program and their meaning are as follows:

GET FILE INFO	Obtain property information about the requested file name.
START BROWSE	Start a browse operation on the requested file name.
BROWSE NEXT	During browse, read the next record in the requested file.
BROWSE PREV	During browse, read the previous record in the requested file.
RESET BROWSE	During browse, reset the browse to start reading at

	a new record.
END BROWSE	End browse operations for the requested file.
READ	Read a record from the requested file.
READ UPDATE	Read a record from the requested file with intent to update.
WRITE	Write a new record to the requested file.
REWRITE	Rewrite a record that was obtained for update.
DELETE	Delete one or more records from the requested file.
UNLOCK	Unlock the requested file from any record locks.
CLEAR DATA	Clear the File Server Program's communication area data and option fields.

XPCRECI1

Specifies the record id to be used in the file operation specified by communication area field XPCAPRQT or the request type. The record id specified by this field is associated with the file name specified by communication area field XPCAPFL1. A value of 1 to a maximum of 60 bytes may be used for the record id. The value supplied should be consistent with the type of record being operated on. For example, if the file were the ROPERIB file, the value supplied should specify the name of a report id.

XPCRECI2

Specifies the record id to be used in the file operation specified by communication area field XPCAPRQT or the request type. The record id specified by this field is associated with the file name specified by communication area field XPCAPFL2. A value of 1 to a maximum of 60 bytes may be used for the record id. The value supplied should be consistent with the type of record being operated on. For example, if the file were the ROPEPCR file, the value supplied should specify the name of a ROPES printer id.

XPCRECI3

Specifies the record id to be used in the file operation specified by communication area field XPCAPRQT or the request type. The record id specified by this field is associated with the file name specified by communication area field XPCAPFL3. A value of 1 to a maximum of 60 bytes may be used for the record id. The value supplied should be consistent with the type of record being operated on. For example, if the file were the ROPEQBR file, the value supplied should specify the appropriate key value for a queue buffer record.

XPCAPFL1

Specifies the name of the file to be used for this request. The record id is supplied by communication area field XPCRECI1. The file name should specify the name of a ROPES file. The value entered must not exceed 8 bytes and should be padded with blanks if smaller.

XPCAPFL2

Specifies the name of the file to be used for this request. The record id is supplied by communication area field XPCRECI2. The file name should specify the name of a ROPES file. The value entered must not exceed 8 bytes and should be padded with blanks if smaller.

XPCAPFL3

Specifies the name of the file to be used for this request. The record id is supplied by communication area field XPCRECI3. The file name should specify the name of a ROPES file. The value entered must not exceed 8 bytes and should be padded with blanks if smaller.

XPCAPRMS

Specifies the APPLID of a remote CICS region which owns the file in question. If available, the file operation will most likely result in a 'Function Shipping' operation with the remote CICS region. If the CICS region that owns the file is local to the requesting application, then this field must be left blank.

XPCAPBRW

This field is set by the External Interface Server program (ROPEXPLS) to indicate if the file is in browse mode or not. When using the File Server Program, this flag is set based on browse operations executed by the File Server Program. If the requested file is in browse mode a value of 'Y' will be set in this field. If the file is not in browse mode there will either be a value of 'N' or possibly binary zero (initial state).

XPCAPEOF

This field is set by the External Interface Server program (ROPEXPLS) to indicate if the end of the file has been reached during browse operations. When using the File Server Program, this flag is set based on browse operations executed by the File Server Program. If the requested file has reach the end of file, then a value of 'Y' will be set in this field. If end of file has not been reached for the requested file, then the value in this field will be 'N' or the initial value of binary zero.

XPCAPMSI

This field must be supplied by the end user application. This field specifies whether the File Server Program should use "mass insert" mode when adding records to the requested file. Two possible values may be set for this field. A value of 'Y' indicates mass insert mode should be used during write or record add operations. A value of 'N' or binary zero means mass insert must not be used. CICS rules regarding the use of mass insert must be followed. When insert activity is complete a request type of 'UNLOCK' must be issued to release the file from mass insert mode.

XPCAPSRT

This field must be supplied by the end user application. This field specifies the type of search that is to be used when reading a record from the requested file. Possible values are given by equated fields XPCAPSEQ and XPCAPSGT. The value associated with equate XPCAPSEQ is "E" which means locate the record that is EQUAL to the requested record id. The value associated with equate XPCAPSGT is "G" which means locate the record that is GREATER THAN or EQUAL to the requested record id.

XPCAPTOK

This field must be supplied by the end user application. This field specifies whether a CICS token should be used with the request specified by field XPCAPRQT. Possible values for this field are 'Y' or 'N' (initial value may also be binary zero). If a token is requested the File Server Program will return the token value from CICS in the File Server Program communication area field FSPTOKEN. Tokens may only be used with update file operations and not browse file operations.

XPCAPMAX

This field must be supplied by the end user application. This field specifies the maximum number of records that should be returned for a BROWSE NEXT or BROWSE PREV request type. The default value for this field is binary zero which results in 1 record being returned by default. A value other than 1 may be specified as a binary fullword positive value great than 1 but should not exceed a number that would exhaust the pre-allocated storage available for the placement of data records. When using the File Server Program through the ROPES External Interface, data records are automatically stored in the File Server Program's communication area. After a request is processed by the File Server Program through the External Interface, the File Server's communication area will reside at field **XPCAPRCA**. The record data will be located at field **FSPRECAR** in the File Server Program's communication area.

XPCAPRET

Supplies the return code from the ROPES External Interface Client program (ROPEXPLR). The return code value is specific to the ROPEXPLR module and may be interpreted from return codes received from the External Interface Server program and File Server Program. Message text also provided in the ROPEXPCA communication area will usually display error messages from the External Interface Server program and/or File Server Program so that the nature of the problem is clear. Other return codes including the original EIBRESP value may also be supplied in these messages. The message commarea fields are described below.

XPCAPRCL

This field may be supplied by the end user application, or it may be supplied by the File Server Program after executing a 'GET FILE INFO' request. It is also possible that the File Server Program may perform the file properties inquiry automatically if this field and the XPCAPFKL fields are null. This may occur the first time access to the file is made when the XPCAPRCL and XPCAPFKL fields are binary zero. This field supplies the record length, and must specify a value that is a halfword or 2 bytes in length. The value must be in binary format. A record length must be supplied that corresponds with the actual record length for the file being processed. In the case of variable length records, the maximum record length must be supplied.

XPCAPFKL

This field may be supplied by the end user application, or it may be supplied by the File Server Program after executing a 'GET FILE INFO' request. It is also possible that the File Server Program may perform the file properties inquiry automatically if this field and the XPCAPRCL fields are null. This may occur the first time access to the file is made when the XPCAPFKL and XPCAPRCL fields are binary zero. This field supplies the full key length, and must specify a value that is a halfword or 2 bytes in length. The value must be in binary format. A full key length must be supplied that corresponds with the actual key length for the file being processed.

XPCAPGKL

This field must be supplied by the end user application but is optional. This field specifies the generic key length to be used when accessing the file. Specifying a value other than binary zero in this field will result in the GENERIC option being used in the file request by the File Server Program (ROPEFSCP). The value specified must be a halfword or 2 bytes in length, and must be in binary format. The value specified must be less than the maximum key length for the file being processed.

XPCAPM1L

This field is supplied by the External Interface Client program. It specifies a one byte binary value containing the length of the message text located in field XPCAPMS1. If the value is binary zero, it indicates that no message text will be found in message field XPCAPMS1.

XPCAPMS1

This field is supplied by the External Interface Client program. This field will contain message text from the External Interface Client program (ROPEXPLR). The length of the message text is supplied by field XPCAPM1L.

XPCAPM2L

This field is supplied by the External Interface Client program. It specifies a one byte binary value containing the length of the message text located in field XPCAPMS2. If the value is binary zero, it indicates that no message text will be found in message field XPCAPMS2.

XPCAPMS2

This field is supplied by the External Interface Client program. This field will contain message text from the External Interface Client program (ROPEXPLR). The length of the message text is supplied by field XPCAPM2L.

XPCAPM3L

This field is supplied by the External Interface Client program. It specifies a one byte binary value containing the length of the message text located in field XPCAPMS3. If the value is binary zero, it indicates that no message text will be found in message field XPCAPMS3.

XPCAPMS3

This field is supplied by the External Interface Client program. This field will contain message text from the External Interface Client program (ROPEXPLR). The length of the message text is supplied by field XPCAPM3L.

XPCAPRCN

This field is supplied by the External Interface Client program. It contains the number of data records returned by the External Interface Server program, or the File Server Program. In the case of records returned by the External Interface Server program (when using the utility driver functions), record data is returned at communication field XPCAPRCA. In the case of records returned by the File Server Program (ROPEFSCP), record data is returned at File Server Program communication area field FSPRECAR. It is important to remember when the File Server Program is used through the ROPES External Interface, the File Server Program's communication area is placed at field XPCAPRCA. This area is a variable length area intended to contain record data, message text data, or other data (in this case the communication area for the File Server Program).

To observe how these fields are set when issuing requests that are handled by the File Server Program, please examine the sample program ROPEXFLE. Note that all request types are issued and processed by the ROPEXFLE program. However, not all possible options associated with the request types are employed. *In some cases, it may be necessary to modify the options fields of the File Server Communication area afer a request has been issued* *through the ROPES External Interface. One example of this would arise if you wanted to use a CICS token during the update of a record. Since no fields occur in the External Interface Client communication area (ROPEXPCA) for request ids, file tokens, or Record Level Sharing options these options would have to be specified directly in the File Server Communication area (ROPEFSCA).*

Linking To The External Interface Client From Your Application

The ROPES External Interface Client program (ROPEXPLR) can be called from your application using the following information for setting up the parameter list. There are six addresses in the parameter list passed to ROPEXPLR. Normally, the ROPEUTIL program uses the first 4 addresses to pass pointers which relate to utility statement syntax. These first 4 address pointers must be set to null when calling ROPEXPLR from your application. The last 2 pointers contain a pointer to the completion code field, and a pointer to the address of the External Interface Communication Area (ROPEXPCA).

A sample of the code fragment in the batch sample program ROPEXFLE is shown in Figure **3** on page 140. Note that register 1 is first cleared and then stored in each of the first 4 address pointers.

It is also important to note that the completion code field will be primed with the completion code last returned for the request type just processed. The code fragment above stores the highest completion code in a field called HICCDE (fullword 4 byte field) so that the highest return code can be set in register 15 when the batch application program terminates.

Preparing the communication area fields to invoke the File Server Program through the ROPES External Interface is not complicated. Most of the time it requires setting just a small number of communication area fields. Please examine the code fragment in Figure **4** on page 141 from the sample batch program ROPEXFLE to see what is required to prepare the communication area fields for a delete record request.

The basic steps required to prepare for a session with the ROPES External Interface Client Program are as follows:

1. Obtain storage for the ROPES External Interface communication area (normally about 26K of storage).
2. Set ROPEXPCA communication area field XPCCALEN to the total length of the communication area.
3. Perform an initial call to ROPEXPLR with the XPCFNCTN field set to "OPENLINK". The fields XPCSRVPG, XPCSVTRN, XPCCLNTP, and XPCTRGSY are required fields for the OPENLINK call.
4. Perform one or more calls to ROPEXPLR with the XPCFNCTN field set to "SENDREQ". These calls will send requests to the File Server Program. The request type is sent in field XPCAPRQT. Additional fields normally used for File Server communication are; XPCRECI1-3, XPCAPFL1-3, XPCAPFKL, XPCAPGKL, and XPCAPMAX.
5. When all File Server Program requests are complete, a final call to ROPEXPLR must be issued with the XPCFNCTN field set to "CLOSELINK". This terminates the ROPES External Interface session with the CICS region.

All of the code required to implement these five steps can be found in the sample program ROPEXFLE.

Similarly to the ROPEXFLE sample application, your application can be assembled and linked using AMODE(31) and RMODE(24). No special INCLUDE statements are required in the link edit controls. Once your application is assembled and linked, it can be used to call the ROPES External Interface Client (ROPEXPLR).

```
         MACRO
         ROPEXPCA &P=XPC
*+-----------------------------------------------------------------+
*|       Commarea Working Storage from ROPEUTIL.                   |
*+-----------------------------------------------------------------+
&P.DSECT DSECT                         COMMAREA WORKING STOR
&P.CAREA DS    0F                      START OF WS AREA
&P.CALEN DS    F                       LENGTH OF COMWS AREA
&P.VERSN DS    F                       DPL CALL VERSION
&P.CALRA DS    CL20                    DPL CALL RETURN AREA
&P.UTOKN DS    F                       USER TOKEN ID
&P.CALTY DS    F                       DPL CALL TYPE
&P.PTOKN DS    F                       PIPE TOKEN ID
&P.SRVPG DS    CL8                     APPLICATION SERVER PROGRAM
&P.APCML DS    F                       APPL. COMMAREA LENGTH
&P.DATLN DS    F                       APPL. DATA LENGTH
&P.SVTRN DS    CL4                     MIRROR TRANSID
&P.UOWID DS    F                       UNIT OF WORKID
&P.USRID DS    CL8                     REQUESTING USER ID
&P.DPLRA DS    CL12                    DPL RETURN AREA
&P.DPLOP DS    CL1                     DPL OPTIONS AREA
&P.CLNTP DS    CL8                     NAME OF CLIENT PGM (ROPEXPLR)
&P.TRGSY DS    CL8                     NAME OF TARGET SYS (APPLID)
&P.FNCTN DS    CL10                    FUNCTION              (RMC10012)
&P.RSRV1 DS    CL3                     RESERVED
&P.APCOM DS    0F                      START OF APPLICATION COMMAREA
&P.APRQT DS    CL20                      REQUEST TYPE
&P.APPR1 DS    CL4                       PRINTER ID 1
&P.APPR2 DS    CL4                       PRINTER ID 2
&P.APPR3 DS    CL4                       PRINTER ID 3
&P.APCPR DS    CL4                       CURRENT PRINTER ID
&P.APRP1 DS    CL8                       REPORT ID 1
&P.APRP2 DS    CL8                       REPORT ID 2
&P.APRP3 DS    CL8                       REPORT ID 3
&P.APCRP DS    CL8                       CURRENT REPORT ID
&P.RECI1 DS    CL60                      RECID 1
&P.RECI2 DS    CL60                      RECID 2
&P.RECI3 DS    CL60                      RECID 3
&P.CURCI DS    CL60                      CURRENT RECID
&P.FLDPL DS    CL240                     FIELD NAME POOL
&P.RSRV2 DS    CL20                      RESERVED AREA
&P.DATPL DS    10CL80                    FIELD DATA POOL
&P.RSRV3 DS    CL20                      RESERVED AREA
&P.APFL1 DS    CL8                       FILE ID 1 (USER)
&P.APFL2 DS    CL8                       FILE ID 2 (USER)
&P.APFL3 DS    CL8                       FILE ID 3 (USER)
&P.APFL4 DS    CL8                       FILE ID 4 (INTERNAL)
&P.APFL5 DS    CL8                       FILE ID 5 (INTERNAL)
&P.APRMS DS    CL4                       REMOTE SYSTEM ID    (RMC10012)
***&P.APSTA DS    CL1                     APPLICATION STATUS (RMC10012)
***&P.APBRW EQU   X'01'                   BROWSE IN PROGRESS (RMC10012)
***&P.APEOF EQU   X'02'                   END OF FILE REACHED(RMC10012)
&P.APBRW DS    CL1                       BROWSE STATUS (Y/N) (RMC10012)
&P.APEOF DS    CL1                       EOF STATUS (Y/N)    (RMC10012)
&P.APMSI DS    CL1                       MASS INSERT (Y/N)   (RMC10012)
&P.APSRT DS    CL1                       SEARCH TYPE         (RMC10012)
&P.APSEQ EQU   C'E'                        SEARCH EQUAL      (RMC10012)
&P.APSGT EQU   C'G'                        SEARCH GT/EQUAL   (RMC10012)
&P.APENQ DS    CL1                       ENQ ON RECORD (Y/N) (RMC10012)
&P.APTOK DS    CL1                       USE CICS TOKEN(Y/N) (RMC10012)
&P.APTKN DS    F                         CICS TOKEN VALUE    (RMC10012)
&P.APMAX DS    F                         MAX RECS TO RETURN  (RMC10012)
&P.APRET DS    F                         APPLIC. RETURN CODE
```

1 The ROPEXPCA communication area (part 1)

```
&P.APRCL DS    H                        RECORD LENGTH       (RMC10012)
&P.APFKL DS    H                        FULL KEY LENGTH     (RMC10012)
&P.APGKL DS    H                        GENERIC KEY LENGTH
&P.APRQI DS    H                        REQID               (RMC10012)
&P.APM1L DS    AL1                      MESSAGE 1 LENGTH
&P.APMS1 DS    CL80                     MESSAGE 1
&P.APM2L DS    AL1                      MESSAGE 2 LENGTH
&P.APMS2 DS    CL80                     MESSAGE 2
&P.APM3L DS    AL1                      MESSAGE 3 LENGTH
&P.APMS3 DS    CL80                     MESSAGE 3
&P.APRCN DS    AL2                      NUMBER OF RETURN RECORDS
&P.APCAL EQU   *-&P.APCOM             APPL. COMMAREA HDR LENGTH
         DS    0F                       ALIGN ON FULLWORD BNDARY
&P.CMWSL EQU   *-&P.CAREA             TOTAL COMMAREA FIXED WS SPACE
** THE RECORD AREA IS VARIABLE AND MAY CONTAIN AS MANY APPLICATION
** RECORDS AS WILL FIT IN THE COMMUNICATION AREA WORK SPACE. EACH
** RECORD SEGMENT MUST CONTAIN A HALFWORD DESCRIPTOR THAT CONTAINS
** THE LENGTH OF THE RECORD. THE LENGTH OF THE RECORD IS THEN FOLLOWED
** IMMEDIATELY BY THE RECORD CONTENTS. EACH RECORD SEGMENT MUST FOLLOW
** THE PREVIOUS SEGMENT IMMEDIATELY WITHOUT ANY SPACES OR SLACK BYTES.
** THE FINAL RECORD LENGTH DESCRIPTOR MUST CONTAIN A HALFWORD VALUE OF
** ZERO. THIS MARKS THE END OF THE RECORDS.
&P.APRCA DS    0F                     START OF RECORD AREA
         SPACE 2
         MEND
```

2 The ROPEXPCA communication area (part 2)

```
*+------------------------------------------------------------------+
*|        LINK TO EXTERNAL INTERFACE CLIENT (ROPEXPLR).               |
*+------------------------------------------------------------------+
LINKXPLR DS    0H
         ST    RBAL,LINKBAL        SAVE BAL REGISTER
         SPACE 1
         SLR   R1,R1               CLEAR R1
         ST    R1,ADDR1            SET 1ST FULLWORD NULL POINTER
         ST    R1,ADDR2            SET 2ND FULLWORD NULL POINTER
         ST    R1,ADDR3            SET 3RD FULLWORD NULL POINTER
         ST    R1,ADDR4            SET 4TH FULLWORD NULL POINTER
         XC    CMPCDE,CMPCDE       CLEAR COMPLETION CODE
         LA    R1,CMPCDE           POINT TO COMPLETION CODE
         ST    R1,ADDR5            STORE ADDRESS OF IT
         LA    R1,COMWSAD          GET COM WRK STOR ADDR
         ST    R1,ADDR6            SAVE IT IN ADDR6
         MVC   PROGNAME,=CL8'ROPEXPLR'  SET LINK PROGRAM NAME
         SPACE 1
         LINK  EPLOC=PROGNAME,PARAM=(ADDR1,ADDR2,ADDR3,ADDR4,          X
               ADDR5,ADDR6),VL=1
         SPACE 1
         L     R1,CMPCDE           GET LAST COMPLETION CODE
         C     R1,HICCDE           IS IT GREATER THAN HIGHEST?
         BNH   LXPRHICD            NO, BYPASS SET NEW HI
         MVC   HICCDE,CMPCDE       SET HIGHEST SO FAR
LXPRHICD DS    0H
         LTR   R1,R1               WAS RETURN NORMAL?
         BZ    EXITLXPR            YES, JUST RETURN
         MVC   SVECDE,CMPCDE       SAVE THE COMPLETION CODE
         SPACE 1
EXITLXPR DS    0H
         L     RBAL,LINKBAL        RESTORE BAL REGISTER
```

3 Code Fragment 1

```
*+--------------------------------------------------------------------+
*|        PROCESS TO DELETE RECORD.                                   |
*+--------------------------------------------------------------------+
DELTEREC DS    0H
         ST    RBAL,DLTEBAL         SAVE BAL REGISTER
         SPACE 1
         MVC   XPCFNCTN,=CL10'SENDREQ   ' FUNCTION TO BE INVOKED
         MVC   XPCAPRQT,=CL20'DELETE        ' SET APP REQUEST TYPE
         MVC   XPCRECI1,BLANKS      CLEAR RECORD ID
         MVC   XPCRECI1(8),=CL8'XPLRTEST'  SET REPORT ID
         MVC   XPCAPFL1,=CL8'ROPERIB ' SET FILE NAME
         MVC   XPCAPMAX,=F'0'       CLEAR MAX RECORDS TO RETURN
         SPACE 1
         BAL   RBAL,LINKXPLR        NOW CALL ROPEXPLR
         CLC   CMPCDE,=F'0'         WAS THE COMPLETION CODE ZERO?
         BNE   DLTEBDRC             NO, ISSUE ERROR
         B     EXITDLTE             EXIT NOW
         SPACE 1
DLTEBDRC DS    0H
         MVI   ERRCODE,ERR55303     INDICATE ERROR ON XPLR LINK
         B     EXITDLTE             EXIT NOW
         SPACE 1
EXITDLTE DS    0H
         L     RBAL,DLTEBAL         RESTORE BAL REGISTER
         BR    RBAL                 RETURN TO CALLER
```

4 Code Fragment 2

Using The ROPES File Server Program

Introduction

The ROPES File Server Program is a new facility in ROPES Version 11. The ROPES Files Server program, called ROPEFSCP, provides file access services to ROPES files for CICS applications and ROPES internal programs. The File Server Program provides a common and standard interface to ROPES files and eliminates much of the work required to access ROPES files directly through CICS. By selecting simple options using a communication area, access to a ROPES file can be accomplished. In many cases, filling in the basic function (i.e. READ), the file name, and the record id is sufficient to retrieve the desired record data. Provided there is enough room in the communication area, a single call can be made to the File Server program to browse a file and return a multiple number of records. For example, a browse can be started on a ROPES file, and then the next 20 records can be read into the communication on a single call to the ROPES File Server Program.

CICS options most frequently used when accessing ROPES files are supported by the File Server Program. These options include but are not limited to Record Level Sharing, tokens, and request ids. The File Server Program can also obtain detailed information about the file to be processed using the "GET FILE INFO" function. If you begin access to a ROPES file and do not provide the record length or key length, the GET FILE INFO function is performed automatically as part of your first call. The file information returned provides; key length, key position, record length, access method, record format, object type (base/path), read integrity type, recovery status, RLS access mode, file type (ESDS, KSDS, etc.), file access mode (add, delete, browse, etc.), base data set name, data set name, and file status.

Multiple files can be accessed at the same time by using multiple communication areas. The communication area for the ROPES File Server Program is described by the assembler macro ROPEFSCA. The fields of the communication area can be easily prefixed to change the names of the fields by using the "P=" option on the macro. The default is set to "P=FSP". So, by default, all communication area field names begin with "FSP". To accommodate additional copies of the communication area simply involves changing the prefix to "FS1", "FS2", etc..

Most forms of file access provided by CICS are also provided by the ROPES Files Server Program. For example, if you want to browse a file you would set the communication request type field to "START BROWSE". The "BROWSE NEXT" request type would then read the next record. The File Server functions work essentially the same way native CICS command level functions work. However, on the "BROWSE NEXT" and "BROWSE PREV" functions you can also specify how many records you want read and returned to your application. It is also possible to specify where the record data is to be returned. The record data can be returned in the File Server Communication Area, in a storage location provided by your program, or in storage obtained dynamically by the File Server Program. The ROPES File Server Program also permits the use of CICS Enqueue/Dequeue to serialize access to records. Support for generic keys, mass insert, and function shipping is also provided.

Using The File Server Program

An end user application may link directly to the File Server Program (ROPEFSCP), or it can be invoked indirectly through the request processes available via the ROPES External Interface. In the case of a direct link to the File Server Program, the end user application would use a CICS LINK command to ROPES module ROPEFSCP. The File Server Program's communication area (ROPEFSCA) would be used to convey processing options to ROPEFSCP. The end user application would have to obtain a communication area large enough to accommodate the record data that will be returned by the ROPEFSCP program. When calling ROPEFSCP directly from an end user CICS application, it is possible to specify where the data records are to be placed. Record data can be returned in the communication area, in a storage area obtained by the end user application, or an area of storage obtained dynamically by the File Server Program. Control of this is provided through the communication area field FSPPUTRW.

The File Server Program's communication area is divided into two sections. The first section contains all of the option fields, file property information fields, and message and return code fields. This section is fixed in length and ends at communication area symbol FSPHDRLN. The second section is variable in length depending on the size of the total commarea. The second section is used by the File Server Program to store returned data records. This area starts at communication area symbol FSPRECAR. The communication area should be large enough to accommodate the maximum number of records that can be returned by the File Server Program. The size of those records is also a factor in determining the size of the communication area. If there is not enough storage to return records in the communication area, the File Server Program will stop reading/storing data in the communication area to prevent a storage violation.

The name of the File Server Program's communication area is ROPEFSCA. The assembler macro ROPEFSCA defines the File Server Program's communication area for the application program. Not all fields in the communication area need to be modified by the application program. Many are present to provide information about the file being processed. Please read the section on the File Server Program's Communication Area for more information.

File Server Program Access Through The ROPES External Interface

When the File Server Program is invoked through the ROPES External Interface the application is a batch program, and it does not link to the File Server Program directly. The batch application must in this case link to the ROPES External Interface Client module which is named ROPEXPLR. The File Server Program request is made using the External Interface Client program's (ROPEXPLR) communication area. The ROPES External Interface Client program's communication area is described by assembler macro ROPEXPCA. A copy of this macro is provided in the ROPES distribution source data set in member ROPEXPCA. The batch application must include the ROPEXPCA communication area as well as the File Server communication area (ROPEFSCA). When the first request to the File Server Program is made, the File Server Communication area fields are initialized at symbol XPCAPRCA within the External Interface Client program's communication area (ROPEXPCA). In this manner, the File Server communication area is contained in the ROPES External Interface Client program's communication area starting at the variable data area located at field XPCAPRCA.

When an External Interface request is made, the External Interface Client and Server programs move results and data provided by the File Server Program between the fields of the File Server communication area to the External Interface Client program's communication area. This makes the results of most File Server requests available to the batch application via the ROPEXPCA communication area. In the case where file properties are required (from GET FILE INFO), the File Server communication area must be accessed directly. For example, if the batch application needed to know the file's status (open/closed), it would have to reference field FSPFLEST to make this determination. Otherwise, in most cases it will not be necessary to access the File Server Communication area directly except to obtain record data. It is important to remember that record data can only be returned in the File Server Program's communication area when being invoked through the ROPES External Interface.

When record data is returned by the File Server Program it is always stored with a halfword binary length field preceding the record data. Multiple records may be returned and each record will have a record length descriptor or RDW preceding it. The last record in the sequence will have an RDW halfword value of binary zero. This marks the end of the sequence of records in the variable data area of the communication area. Remember, the variable data area starts at symbol FSPRECAR. Records are always returned in this area when the File Server Program is invoked through the ROPES External Interface!

The File Server Program Communication Area

The File Server Program communication area is defined by the ROPEFSCA macro. The ROPEFSCA macro can be found in the ROPES distribution source data set. A COBOL version of the communication area can be found in the ROPES COBOL library. It is also named ROPEFSCA.

A sample of the ROPEFSCA macro appears in figure 5 on page 153.

File Server Communication Area Fields

Please find below the description for all useable File Server Program communication area fields. These fields are included in the assembler macro named ROPEFSCA which can be found in the ROPES Distribution Source data set. Fields marked with "(Info)" are those fields which are returned by the File Server Program when a 'GET FILE INFO' requested is issued, or when the File Server Program does a file inquiry automatically. Those fields that indicate "Supplied" are normally provided or set by the File Server Program. Those fields that indicate "Specifies" are normally set by the calling application program. There are a couple of cases where a field may be both supplied and reset by the calling application. The field FSPRECCA is one of these fields.

FSPCAREA

Marks the start of the File Server Communication Area.

FSPVERSN

Specifies the version of ROPES this Communication Area belongs to. This field is set by the ROPEXPLR program.

FSPLEVEL

Specifies the ROPES level this Communication Area belongs to. This field is set by the ROPEXPLR program.

FSPCALEN

Specifies the entire length of the File Server Communication Area. This field is expressed as a fullword (4 bytes) binary length. It must accurately contain the length of the File Server Communication Area, and the length should be at least several kilobytes larger than the minimum length specified by the **FSPHDRLN** equated value. It is recommended that the entire length of the File Server Communication Area be at least 24-26 kilobytes in size. This would normally accommodate the size of the largest Printer Control Record when processing the ROPEPCR file. There are other size and pointer fields which are set based on the size specified in the FSPCALEN field so it is very important that it be set accurately!

FSPREQST

Specifies the type of request being made of the File Server Program (ROPEFSCP). The possible requests are:

GET FILE INFO	Get file properties for the file being requested.
START BROWSE	Start browse operation on the file being requested.
BROWSE NEXT	Read the next record for the file requested.
BROWSE PREV	Read the previous record for the file requested.
RESET BROWSE	Reset browse for the file requested.
END BROWSE	End browse operation for the file requested.
READ	Read a record from the file requested.
READ UPDATE	Read a record with intent to update for the file requested.
WRITE	Write a record to the file requested.
REWRITE	Rewrite a record to the file requested.
DELETE	Delete one or more records from the file requested.
UNLOCK	Unlock any locked records for the file requested.
CLEAR DATA	Clear all communication area fields to start processing a new file.

The FSPREQST field is 16 bytes in length and all request values must be padded with blanks.

FSPFLENM

Specifies the name of the file to be processed. This field must specify the name of a ROPES file. The field must contain a value 1 to 8 bytes in length and must be padded with blanks.

FSPKEYLN

Specifies the key length of the file being requested by field FSPFLENM. If the key length is required (for the type of operation being performed), and it is not known, then it is possible to also leave the FSPRECLN field null to force the File Server Program to retrieve this information automatically on the first request. Otherwise, it is also possible to perform a 'GET FILE INFO' request for the desired file to retrieve the key length. Note that other file property details will also be filled in. If the key length is known by the application and is specified, it must be set to the correct value for the file in question and it must be specified as a halfword (2 bytes aligned on a halfword boundary) in binary format.

FSPGENKL

Specifies the generic key length for generic file operations. Specifying a value for this field is optional. If a value is specified, it must be less than the full key length for the file being processed (FSPFLENM). Otherwise, this field must contain a value of binary zero. The value must be in the format of a halfword (2 bytes aligned on a halfword boundary) and be in binary format.

FSPRECID

Specifies the record id for the record to be processed. The record id may contain any alpha-numeric or binary value up to 80 bytes in length. The record id value should be appropriate for the ROPES file being processed. For example, when accessing the ROPERIB file, the record id would be an 8 character report id. The value should also be padded with blanks or nulls.

FSPLRECI

Identifies the last record id processed. This field is supplied by the File Server Program and is for information purposes only.

FSPRECLN

Specifies the record length for the file being processed. If the file contains variable length records, this field should contain the maximum record length. The value may be supplied by the calling application program, or it can be supplied by the File Server Program if both FSPRECLN and FSPKEYLN are left null on the first call. Otherwise, the value for FSPRECLN can also be filled in by issuing a 'GET FILE INFO' request. The FSPRECLN value must be supplied as a halfword (2 bytes aligned on a halfword boundary) and must be in binary format. The record length supplied should be correct for the file being processed.

It is important to note that while the record length is received from FSPRECLN for read functions, it is NOT used as the record length for WRITE operations! This includes the WRITE and REWRITE operations. The record length for write operations is received from the record descriptor word (RDW) that precedes the record in storage. Since the record length in the FSPRECLN field is normally the maximum record length, the RDW record length can be smaller. The FSPRECLN field should always contain the maximum record length for a file, and this is why the RDW is used to supply the record length on write operations. The application is responsible for making sure the correct record length is supplied in the RDW preceding the record data to be written.

FSPKYPOS

This field is for informational purposes only and is supplied by the File Server Program when a 'GET FILE INFO' request is made. This field is expressed as a halfword (2 byte binary) field.

FSPRECAA

This field contains an address pointer to the first area used to store a retrieved data record. The field is expressed as a fullword (4 bytes aligned on a fullword boundary) in binary format. This field must be set by the calling application if the storage area used for returned records is provided by the calling CICS application. In other words, the value of communication area field **FSPPUTRW** is set to "FSPUSERA" or "2" which means the calling application is supplying the storage for the returned records. ***When using the File Server Program through the ROPES External Interface this field is always set by the External Interface Server program (ROPEXPLS) to FSPSTORC or "1" which means records must be returned to the communication area starting at FSPRECAR. The value of FSPPUTRW should never be changed when invoking the File Server facilities through the ROPES External Interface.*** Otherwise, when calling the File Server Program directly, records may be stored in an area specified by the calling program and this is the communication area field that would specify the starting location of that area. If the record data area is supplied by the calling application, then communication area field ***FSPRECAL*** must also be set by the calling application, and must specify the total area length (in a fullword) for the storage area provided.

FSPRECCA
Specifies the address pointer for the current record location. The calling application can set this address pointer if desired, and in most cases would probably reset the pointer to the first record location possible. If records were stored in the communication area, then the first possible location would be at symbol FSPRECAR. FSPRECAA should always point to the RDW for the record (if there is one in storage). This field is expressed as a fullword (4 bytes aligned on a fullword boundary) in binary format. The address value for this field must be great than the address stored in FSPRECAA, but less than the address stored in communication area field FSPEORCA.

FSPRECAL
Specifies the total length of the record data area portion of the File Server communication area, or the total length of a storage area provided by the calling application or File Server program (depending on the setting of FSPPUTRW). This field is expressed as a fullword address pointer (4 bytes aligned on a fullword boundary) in binary format.

FSPRECRL
This field is normally supplied by the File Server Program and indicates the remaining area left for storing records. It is constantly updated based on the records currently stored in the record data area. This field is expressed as a fullword length (4 bytes aligned on a fullword boundary) in binary format.

FSPLUSRA
Specifies the address pointer of the last record stored in the record data area. This field is normally maintained by the File Server Program. It is expressed as a fullword address pointer (4 bytes aligned on a fullword boundary) in binary format.

FSPEORCA
Normally supplied by the File Server Program, this field specifies the address pointer for the end of the record data area used to store records. This pointer may point to the end of the File Server communication area, or to the end of an area of storage provided by the calling application or ROPEFSCP. This field is initialized on the first call to the File Server program even if the storage area is provided by the calling application. This field's initial value should always be binary zero even when the calling program supplies the storage area for record data. This field is expressed as a fullword address pointer (4 bytes aligned on a fullword boundary) in binary format.

FSPMAXRR
Specifies the maximum number of records to be returned during a browse next or browse previous request. The value must supplied as a fullword binary field (4 bytes on a fullword boundary). The number of records requested should not exceed the maximum amount of storage available for record data in the communication area or in the storage provided by the calling program. When storage is obtained for record data by the File Server it uses the FSPMAXRR field to determine the size of the storage to obtain. This is accomplished by multiplying the max record length, plus the size of the RDW by the number of records requested in FSPMAXRR.

FSPRECNO
This field is returned by the File Server Program to indicate the number of data records returned for the browse next or browse previous request. The field is expressed as a fullword binary field (4 bytes on a fullword boundary).

FSPTOKEN
This field is returned by the File Server program from CICS when the use of a token is requested. A token may be requested by the flag setting in communication area field FSPUTOKN. This field is expressed as a fullword binary value (4 bytes on a fullword boundary). When doing a read for update, CICS fills this field with a unique binary identifier for subsequent use in a rewrite or delete request. The token uniquely identifies the originating read for update request so that it can be properly associated with the rewrite, delete, or unlock request.

FSPFSERC
Supplies the File Server Error code which indicates the normal or error status of processing performed by the File Server Program. The File Server error code is expressed as a fullword binary value (4 bytes on a fullword boundary). Messages returned in fields FSPMSG1T through FSPMSG3T reflect the nature of the error.

FSPFSRET
Supplies the File Server Return code. The return code value is either binary zero (normal), or 8 for

all error conditions. This provides a quick means for checking the return status. This field is expressed as a fullword binary value.

FSPFSRSP
Supplies the last EIBRESP code returned after a CICS failed request. The EIBRESP field is returned in a fullword binary field.

FSPREQID
Specifies the request id to be issued with the CICS browse operation. The request id is specified as a halfword binary value. It is used to control multiple browse requests against the same file. A unique binary identifier is placed in this field to identify the browse request. By default the value is binary zero which tells CICS not to use the request id. However, the request id is specified on all CICS browse commands issued by the File Server program. If you don't want the request id used, then you must specify a value of binary zero for the FSPREQID field.

FSPENQNL
Specified with the FSPENQRC and FSPENQNM fields. This field specifies the length of the enqueue name given by communication area field FSPENQNM. The use of the enqueue is controlled by field FSPENQRC which must be set to "Y" to turn on enqueueing for the current request. The FSPENQNL field is specified as a halfword binary value (2 bytes on a halfword boundary). The enqueue name is comprised of the file name plus the record id. Therefor, the length specified in FSPENQNL must be 8 (for the file name) plus the key length. In the case of the ROPERIB file, the length specified for FSPENQNL would be 16 because the key length is 8 bytes (report name).

FSPENQRC
Specifies whether a CICS enqueue should be issued for this File Server request. Note that enqueue/dequeue is issued for READ UPDATE and REWRITE requests only. The enqueue option is not valid when the File Server is invoked through the ROPES External Interface Feature. The primary reason is there is no task continuity when using the File Server through the ROPES External Interface Feature. The ROPES External Interface Server program compensates for this on all other requests but cannot maintain the enqueue option when used with the External Interface. To request the enqueue feature for directly File Server update/rewrite requests you must set a value of 'Y' in the FSPENQRC field. A value of 'N' or binary zero (initial value) will turn of enqueue for the current request.

FSPENQNM
The File Server Program uses this field to build the name for the enqueue request. The name is made up of the file name plus the record id. This makes a unique enqueue name for any record obtained from a ROPES file. The FSPENQNM field is 88 bytes long which can accommodate the 8 byte file name and an 80 character record id. This field is ignored if FSPENQRC is set to 'N' or binary zero.

FSPRMSYS
Specifies the remote system id to be used if the request should be function shipped to a remote CICS region. This field may contain blanks or the name of the system id of a remote CICS region. The system id value must be a value 1 - 4 characters in length and must identify a valid CICS system id.

FSPACMTH
(Info) Supplies the access method for the requested file. Possible values are; B=BDAM, R=REMOTE, and V=VSAM. The field is supplied by the File Server Program after a 'GET FILE INFO' request has been made, or when the File Server Program automatically performs a file inquiry to obtain the file properties. This field is 1 byte in length and will contain "B", "R", "V", or a blank.

FSPRMTNM
(Info) Supplies the remote name for the file if one is applicable. This field is returned by the File Server Program after a 'GET FILE INFO' request or when ROPEFSCP performs an automatic file inquiry to determine the properties of the file. This field is 8 bytes in length and will contain either the remote name or blanks.

FSPRCFMT
(Info) Supplies the record format for the file requested. This field returns a 1 byte value which may contain one of the following values; F=Fixed, V=Variable, U=Undefined, or blank.

This field is populated after a file inquiry by the File Server Program.

FSPFOBJT
(Info) Supplies the file object. This field is a 1 byte character field and will contain either; B=Base, P=Path, or blank. This field is supplied after the File Server Program performs a file inquiry to determine the properties of the file.

FSPRDINT
(Info) Supplies the Read Integrity mode for the file requested. This field is 1 byte in length. Possible values returned are; C=Consistent, N=Notapplicable, R=Repeatable, U=Uncommitted, or blank. If the value is blank, Record Level Sharing is not in force for this file. This field is supplied after a file inquiry by the File Server Program.

FSPRECOV
(Info) Supplies the the recover status for the requested file. This field is 1 byte in length. Possible values returned are; R=Recoverable, N=Notrecoverable, or blank. If the value is blank, there is no recovery status set for the file definition. This field is supplied after a file inquiry by the File Server Program.

FSPRLSAC
(Info) Supplies the Record Level Sharing (RLS) access mode for the requested file. This field is 1 byte in length. Possible values returned are; A=Notapplicable, R=RLS, N=NotRLS, or blank. This field is supplied after a file inquiry by the File Server Program.

FSPFLTYP
(Info) Supplies the file type for the requested file. This field is 9 bytes in length. Possible values returned for his field are; ESDS, KEYED, KSDS, NOTKEYED, RRDS, VRRDS, NOTAPPLIC, or blanks. This field is supplied after a file inquiry has been performed by the File Server Program.

FSPFACCS
(Info) Supplies the file access modes for the requested file. This field is 1 byte in length. Possible values returned are; A=Add, B=Browse, D=Delete, R=Read, U=Update, or blank. These are the access modes permitted for the file. This field is supplied after a file inquiry by the File Server Program.

FSPBSEDS
(Info) Supplies the base data set name for the requested file. This field is 44 bytes in length. Possible values for this field are; the base data set name, or blanks. This field is supplied after a file inquiry is performed by the File Server Program.

FSPDSME
(Info) Supplies the data set name for the requested file. This field is 44 bytes in length. Possible values for this field are; the data set name, or blanks. This field is supplied after a file inquiry by the File Server Program.

FSPFLEST
(Info) Supplies the file status for the requested file. This field is 1 byte in length. Possible values for this field are; O=Open, C=Close, D=Disabled, E=Enabled, N=Not Available, U=Unenabled/Unenabling, or blank. This field is supplied after a file inquiry by the File Server Program.

FSPMASIN
Specifies that mass insert mode should be turned on for this file write request. This field is 1 byte in length. Possible values for this option field are; Y=Yes use mass insert, or N=No don't use mass insert.

FSPEOFIN
(Info) Supplies the end of file status for the requested file. If one or more read requests results in the end of file condition, the File Server Program will set this flag to a value of 'Y'. This field is 1 byte in length. Possible values for this field are; Y=Yes end of file is on, N=No end of file is not on.

FSPUTOKN
Specifies that the File Server Program should use the CICS token value supplied by the FSPTOKEN field to save the token value for a read update request, or use the token value for a rewrite, delete, or unlock request. This field is 1 byte in length. Possible values for this field are; Y=Yes use a token, N=No don't use the token.

FSPPUTRW
Specifies that record data is to be stored in one of three possible storage locations. This field is 1 byte in length. Possible values for this field are;

1=Store records in the commarea at field FSPRECAR, 2=Use the address in FSPRECAA to store records, 3=File Server Program obtains the storage area and sets FSPRECAA to point to it. Three equate values have been establish for the possible flag settings. They are; 1=FSPSTORC, 2=FSPUSERA, 3=FSPGSTOR. When invoking the File Server Program through the ROPES External Interface, the External Interface Server automatically sets the value of this flag to '1', or store records in the COMMAREA. This is required in order to enable the passing of record storage from the File Server Program back to the requesting batch application program and yet maintain an address that will be permitted by CICS for update purposes. When doing direct calls to the File Server Program this is not a consideration and therefor any flag setting may be used.

FSPBRWSM

Supplies the browse status for the requested file. If a start browse has been issued for the requested file, then the File Server Program will set the value of this flag to 'Y' to indicate browse mode is on. If an end browse has been requested for the file, then the File Server Program will set the value of the flag to 'N' to indicate the browse mode is now off. If the value is binary zero, then it is the initial value which would indicate that the file has never been in browse mode. This field is 1 byte in length and may contain 'Y', 'N', or null.

FSPSRCHT

Specifies the type of search requested for the file. When reading or browsing the file, the search type may be set to match records when the key is equal, or match records when the key is greater than or equal. This field is 1 byte in length. Possible values for this field are; 'E'=Equal key is a match, 'G'=Greater than or equal key is a match. This field may also contain an initial value of binary zero if it has not been set, or if the 'CLEAR DATA' function has been used to clear communication area fields.

FSPMSG1L-3L

Supplies the length of the corresponding message field (FSPMSG1T through FSPMSG3T). The length field is 1 byte long and must contain a binary value specifying the length of message text in the corresponding message text field (FSPMSG1T thru FSPMSG3T).

FSPMSG1T-3T

Supplies the error message text returned by the File Server Program in the event errors have occurred processing the request. Message text fields may be up to 80 bytes in length and will contain alpha-numeric data.

FSPRECAR

This symbol marks the start of a variable length area where record data may be returned by the File Server Program. The length of this area is set in field FSPRECAL, and is determined by subtracting the header length (FSPHDRLN) of the communication area from the overall length set in field FSPCALEN. This area should be large enough to contain the maximum number of records to be returned by the File Server Program including an RDW prefix for each record and a final RDW containing binary zero to mark the end of the record sequence. When invoking the File Server Program through the ROPES External Interface Feature, records are always stored at the location marked by this label.

Task Continuity When Invoking FSP Through The External Interface

There is a loss of task continuity when invoking the File Server Program from the ROPES External Interface. This occurs because the task (ROXP) which executes the ROPES External Interface Server and File Server Program terminates when it returns to the calling batch application. This means when a read for update is executed by the File Server Program on one request, task continuity is lost by the time the rewrite is executed. This problem is rectified by a form of emulation carried out by the ROPES External Interface Server program (ROPEXPLS). In the case of Browse operations the problem is compensated for in a simple manner. ROPEXPLS simply issues another START BROWSE request before it issues the BROWSE NEXT request. In fact, any BROWSE NEXT or BROWSE PREV request is always preceded by a START BROWSE request to the File Server Program so that it can be accomplished under the same task. Since task termination ends the browse, ROPEXPLS does not actually issue an END BROWSE request. However, when the batch application finishes browsing a file, it must issue the END BROWSE request to shut off the browse mode flag. In this manner, the continuity of all browse operations is maintained. It should be noted however, options which involve tokens or request ids may not work correctly when invoking the File Server through the External Interface.

In the case of read for update and rewrite the problem is a bit more complex. However, ROPEXPLS compensates for the lack of task continuity using another means. When the first read for update is issued for a record, ROPEXPLS saves the before image of the record to temporary storage. When the rewrite is executed, ROPEXPLS does another read for update for the same record and compares the before image (from temporary storage) of the previous read for update with the before image of the second read for update. If the before images match, then no intervening update has taken place and the rewrite for the after image is allowed to take place. If the before images do not match, then an error is returned to the calling application which indicates an intervening update has taken place. Error message ROPES55217 will result, or an External Interface Server return code of 18 (decimal) or X'12' (hexadecimal) will be returned to the calling application. This condition is similar to a "record busy" condition when an attempt is made to update a record using record level sharing. The calling batch application will have to check for this response and either attempt to update the record again after waiting a short while or terminate the process.

When invoking the File Server Program through the ROPES External Interface, all request type sequences must be issued in exactly the same manner/order as they would if the File Server Program were being called directly. This is to say that 'START BROWSE' must be issued before 'BROWSE NEXT' or 'BROWSE PREV', and 'END BROWSE' must be issued after all 'BROWSE NEXT/PREV' request types have been issued. A 'READ UPDATE' must always precede a 'REWRITE' request type (XPCAPRQT is the request type field). The ROPEXPLS module compensates for the lack of task continuity.

When ROPEXPLS writes the temporary storage records to save the before image of a record during read for update processing, the key of that storage is comprised of the last 4 characters of the file name followed by a 4 byte checksum value that represents the key of the record. Using this method, the key for the temporary storage record should always be unique. The External Interface Server program always deletes this temporary storage after the REWRITE is done, or if any error should occur during the READ UPDATE or REWRITE process. If another type of failure should leave the temporary storage record in the queue, then another read for update request on the same record should result in the temporary storage record being deleted.

How To Use Concurrent Multiple File Access

When invoking the File Server Program through the ROPES External Interface Feature, multiple File Server Program communication areas are not available. However, the calling batch application can maintain several File Server Communication Areas at the same time by saving one to another area of storage, and then moving the new copy of the File Server Communication Area over the location where the old one resided (at label XPCAPRCA). Care must be taken to make sure the copies of the File Server Communication area are EXACTLY the same size or a storage overrun will occur at the end of the ROPES External Interface Communication Area (ROPEXPCA). Using this method, copies of the File Server Communication area (ROPEFSCA) can be "pushed" to a new storage save area, and "popped" back into the ROPEXPCA communication area when they are needed.

It is also possible to terminate access with one file and switch access to a new file from the same batch application. This is accomplished by finishing up access to the first file, then doing a 'CLEAR DATA' request to clear the File Server communication area to make it ready to process a new file. Using this method, any number of files may be accessed in one batch application process.

When invoking the File Server Program from a CICS application (directly), all requests are made through the File Server communication area (ROPEFSCA). This scenario permits multiple copies of the File Server communication area to be used at the same time. Each communication area would request processes for a different file concurrently. The only limitation using this scenario might be the amount of storage required to maintain multiple copies of the File Server communication area. Each copy of the communication area can also be cleared using the 'CLEAR DATA' request type to enable a file name switch and start processing a new file.

File Server Error Messages

The File Server Program's error messages start with the number ROPES6000 and end with ROPES6099. When trouble shooting errors these messages will contain the description of the original File Server error. In nearly all cases, the request type and file name are mentioned in the error and where appropriate the record id or key of the record is also displayed.

Handling Errors In The Batch Application

When handling errors in the batch application it may be necessary to examine the error code returned in the FSPFSERC field of the File Server Communication Area. This code will give the exact nature of the error and each error code is an ascending sequential number that corresponds with the message numbers for the File Server Program. For example, FSPFSERC error code 01 results in ROPES message ROPES6001. All File Server Error Messages will also be accompanied by the ROPES6000 message which identifies the request type and the file name the request was made for. So, File Server error code (FSPFSERC) 28 will produce error message ROPES6028, and File Server error code 17 will produce error message ROPES6017. Knowing this relationship between the error code and the ROPES600n messages will make it possible to handle errors appropriately in the calling batch application.

Execution JCL

Please find below a sample of the execution JCL for the sample batch application program ROPEXFLE. The JCL member is located in the ROPES distribution source data set under member RUNXFLE. The sample JCL appears in figure 7 on page 155.

Note that the CICS.SDFHEXCI data set has been included in the STEPLIB concatenation for the program. This is required to properly run the ROPES External Interface programs. The DD statements for the ROPELST and ROPERPT SYSOUT files must also be included to receive all produced messages. Program ROPEXFLE does not open DD statement ROPERPT, but it will be opened by the ROPEXPLR program which called by ROPEXFLE. Therefore, when designing your batch application to work with the ROPES External Interface and File Server Program, these files must be included in the JCL.

The ROPERPT and ROPELST message files are 133 byte fixed length output files that are generally allocated to SYSOUT.

The SYSMDUMP and SYSUDUMP data sets are optional, but may be required in the even of any abends. It is recommended they be included in the JCL.

Before running the RUNXFLE job stream, please make sure you have ROPES report ids REPORT01 through REPORT05 defined to your system. The content of the report definitions is not important. These report ids are simply samples that are being used to demonstrate the functions of the ROPES External Interface batch sample program. After the sample program has been run successfully, the REPORT01 through REPORT05 definitions can be deleted from your system. Since we provide these reports as samples it is possible that they may already exist on your system. When checking the output from the RUNXFLE job, records obtained from the CICS/ROPES ROPERIB file are shown in dump format. On browse requests, a maximum of two records has been requested when doing the browse next/previous functions. If more records had been requested, they would have been displayed in the output in dump format. The number of records selected was strictly arbitrary. Browse operations in the job show that browse next and browse previous operations work correctly. During browse operations the last record read is always the first record read on the next browse operation. This is due to the fact that task continuity is not available when making file requests through the ROPES External Interface. The requesting application must compensate for this if necessary.

The "File Server Log" shows exactly what File Server requests were made and for what file and record ids. A sample of the Log is available in figure **Figure 33** on page 156 For this test, the file is always ROPERIB. The test sequence demonstrates that all External Interface requests work correctly. In one case, there is a DELETE request for record id "XPLRTEST" that fails (the record does not yet exist at this point in the job). The program ROPEXFLE overrides the fact that this request failed, and continues processing anyway. This demonstrates that failed return codes can be ignored and processing can continue if required. Later on, the "XPLRTEST" record is deleted successfully.

At the end of the RUNXFLE job, the highest return code indicated should be zero.

```
         MACRO
         ROPEFSCA &P=FSP
*+-------------------------------------------------------------------+
*|       ROPES File Server Communication Area.                      |
*+-------------------------------------------------------------------+
&P.DSECT DSECT                         File Server COMMAREA
&P.CAREA DS    0F                      Start of COMMAREA
&P.VERSN DC    CL6'V10.00'             ROPES Version Number
&P.LEVEL DC    CL2'00'                 ROPES Modification Level
&P.CALEN DS    F                       Length of COMMAREA Storage
         SPACE 2
&P.REQST DS    CL16                    REQUEST TYPE FOR KEY AND FILE
&P.FLENM DS    CL8                       File Name
&P.KEYLN DS    H                         Key Length
&P.GENKL DS    H                         Generic Key Length
&P.RECID DS    CL80                      RECID or Key Value
&P.LRECI DS    CL80                      Last RECID key value
&P.RECLN DS    H                         Record Length
&P.KYPOS DS    H                         Key Position
&P.RECAA DS    F                         Address Of Record Area
&P.RECCA DS    F                         Current Record Address Pointer
&P.RECAL DS    F                         Record Area Length provided
&P.RECRL DS    F                         Record Area Remaining Length
&P.LUSRA DS    F                         ADDR. OF LAST USED RECAREA
&P.EORCA DS    F                         End Of Record Area Address
&P.MAXRR DS    F                         Max Records To Read
&P.RECNO DS    F                         Records Returned/Deleted
&P.TOKEN DS    F                         CICS Token
&P.FSERC DS    F                         FILE SERVER ERROR CODE
&P.FSRET DS    F                         FILE SERVER RETURN CODE
&P.FSRSP DS    F                         FILE SERVER EIB RESPCDE
&P.REQID DS    H                         Request ID
         SPACE 1
&P.ENQNL DS    H                         Enqueue Name Length
&P.ENQRC DS    CL1                       Enqueue On Record Flag(Y/N)
&P.ENQNM DS    CL88                      Enqueue Name
&P.RMSYS DS    CL4                       Remote System ID
&P.ACMTH DS    CL1                       Access Method
*                                          B=BDAM/R=REMOTE/V=VSAM
&P.RMTNM DS    CL8                       Remote Name
&P.RCFMT DS    CL1                       Record Format
*                                          F=FIXED/V=VAR/U=UNDEFINED
&P.FOBJT DS    CL1                       File Object
*                                          B=BASE/P=PATH
&P.RDINT DS    CL1                       Read Integrity
*                                          C=CONSISTENT/N=NOTAPPLIC
*                                          R=REPEATABLE/U=UNCOMMITTED
&P.RECOV DS    CL1                       Recovery Status
*                                          R=RECOVERABLE/N=NOTRECOVABLE
&P.RLSAC DS    CL1                       RLS Access Mode
*                                          A=NOTAPPLIC/R=RLS/N=NOTRLS
&P.FLTYP DS    CL9                       File Type
*                                          ESDS/KEYED/KSDS/NOTKEYED/
*                                          RRDS/VRRDS/NOTAPPLIC
         SPACE 1
&P.FACCS DS    0CL5                      File Access modes flag
&P.FAADD DS    CL1                         Add Mode Flag
&P.FABRW DS    CL1                         Browse Mode Flag
&P.FADEL DS    CL1                         Delete Mode Flag
&P.FARDF DS    CL1                         Read Mode Flag
```

5 File Server Communication Area (part 1)

```
&P.BSEDS DS    CL44                        Base Dataset Name
&P.DSNME DS    CL44                        Dataset Name
         SPACE 1
&P.FLEST DS    CL1                         File Status Flag (O/C/D/E/N/U)
&P.FLOPN EQU   C'O'                          File is Open
&P.FLCLO EQU   C'C'                          File is Closed
&P.FLDIS EQU   C'D'                          File is Disabled
&P.FLENA EQU   C'E'                          File is Enabled
&P.FLUNA EQU   C'N'                          File is Not Available
&P.FLUNE EQU   C'U'                          File is Unenabled/Unenabling
         SPACE 1
&P.MASIN DS    CL1                         Mass Insert Flag (Y/N)
         SPACE 1
&P.EOFIN DS    CL1                         End Of File Indicator (Y/N)
         SPACE 1
&P.UTOKN DS    CL1                         Use CICS Token (Y/N)
         SPACE 1
&P.PUTRW DS    CL1                         Put Records Where Flag
&P.STORC EQU   C'1'                          Store record(s) in COMMAREA
&P.USERA EQU   C'2'                          Use Address in FSPRECAD
&P.GSTOR EQU   C'3'                          Get Storage for record(s)
         SPACE 1
&P.BRWSM DS    CL1                         Browse Mode Flag
&P.BRWOF EQU   C'N'                          Browse Off
&P.BRWON EQU   C'Y'                          Browse On
         SPACE 1
&P.SRCHT DS    CL1                         Search Type
&P.SRCEQ EQU   C'E'                          Search Equal
&P.SRCGE EQU   C'G'                          Search GTEQ
         SPACE 1
&P.MSG1L DS    AL1                         MESSAGE 1 LENGTH
&P.MSG1T DS    CL80                        MESSAGE 1
&P.MSG2L DS    AL1                         MESSAGE 2 LENGTH
&P.MSG2T DS    CL80                        MESSAGE 2
&P.MSG3L DS    AL1                         MESSAGE 3 LENGTH
&P.MSG3T DS    CL80                        MESSAGE 3
         SPACE 1
&P.RSRV1 DS    F                           RESERVED AREA
&P.RSRV2 DS    F                           RESERVED AREA
&P.RSRV3 DS    F                           RESERVED AREA
&P.RSRV4 DS    CL1                         RESERVED AREA
&P.RSRV5 DS    CL1                         RESERVED AREA
&P.RSRV6 DS    CL16                        RESERVED AREA
&P.RSRV7 DS    CL20                        RESERVED AREA
         SPACE 1
&P.HDRLN EQU   *-&P.CAREA                COMMAREA HEADER LENGTH
** THE RECORD AREA IS VARIABLE AND MAY CONTAIN AS MANY ROPES FILE
** RECORDS AS WILL FIT IN THE COMMUNICATION AREA WORK SPACE. EACH
** RECORD SEGMENT MUST CONTAIN A HALFWORD DESCRIPTOR THAT CONTAINS
** THE LENGTH OF THE RECORD. THE LENGTH OF THE RECORD IS THEN FOLLOWED
** IMMEDIATELY BY THE RECORD CONTENTS. EACH RECORD SEGMENT MUST FOLLOW
** THE PREVIOUS SEGMENT IMMEDIATELY WITHOUT ANY SPACES OR SLACK BYTES.
** THE FINAL RECORD LENGTH DESCRIPTOR MUST CONTAIN A HALFWORD VALUE OF
** ZERO. THIS MARKS THE END OF THE RECORDS.
&P.RECAR DS    0C                        START OF RECORD AREA
         SPACE 2
         MEND
```

6 File Server Communication Area (part 2)

SAMPLE RUNXFLE JCL MEMBER

```
//RUNXFLE JOB (ANYNAME),'123-123-1234',MSGLEVEL=1,CLASS=A,
//         MSGCLASS=X,NOTIFY=ANYNAME
//*+-----------------------------------------------------------------+
//*| EXECUTE ROPES EXTERNAL FILE SERVER APPLICATION THRU THE        |
//*| ROPES EXTERNAL CICS INTERFACE.                                  |
//*+-----------------------------------------------------------------+
//ROPEXCI  EXEC  PGM=ROPEXFLE,PARM='COMWSLEN=24K'
//STEPLIB  DD DSN=ROPES.LOADLIB,DISP=SHR
//         DD DSN=CICS.SDFHEXCI,DISP=SHR
//ROPELST  DD SYSOUT=*
//ROPERPT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSMDUMP DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//
```

7 RUNXFLE JCL Member

[illegible]

Figure 33 Sample File Server Log

Jun [illegible]

PDF and HTML Conversion Services

Introduction

In Version 10.0 we added the ability to send e-mail attachments formatted as Adobe® Acrobat® PDF documents. Along with that capability, we provided a batch utility program that allows you to convert files or ROPES reports to PDFs, and an online transaction to allow you to convert ROPES Reports to PDFs.

In Version 11.0 we added the ability to send e-mail attachments formatted as HTML documents. Along with that capability, we provided a batch utility program that allows you to convert files or ROPES reports to HTML, and an online transaction to allow you to convert ROPES Reports to HTML.

The conversion programs we use can be invoked by any user application. This chapter describes the interface to the batch and online versions of these conversion programs.

PDF and HTML Conversion Services

The conversion programs provided are available for batch applications and for online applications. The conversion process uses controls provided in the ROPES Forms Control Blocks (RO$Fxxxx) modules, and vary depending on the type of file being created. Please refer to the Forms Control Block tailoring chapter in the Adminstrator's Guide.

ROPES generated PDFs contain unadorned text only. No provision has been made, yet, to provide for more elaborate text formatting or for the inclusion of graphical elements.

ROPES generated HTML permit more formatting of text, the selection of foreground (text) and background colors (one each). You can request the generation of a document header, which may be repeated on each page. The font size of the body text and the header (Banner) text are independently selectable. You can also provide link information for a graphic that will appear in the Banner.

Invocation

Batch

The batch conversion program is invoked using the CALL macro or verb, depending on your programming language. The parameters required by the conversion programs are provided in the calling arguments. The conversion programs are written in assembler and follow the standard linkage conventions.

To invoke the batch conversion program, you must first place the data to be converted into a sequential data set. The data set format must be RECFM=VB, LRECL=300, and any desired BLKSIZE value may be selected. The conversion program will return the converted output to another sequential file with the same attributes, which you must allocate before invoking the converter.

The conversion program is invoked by a CALL statement.

```
Assembler:
   CALL xxxxxxxx USING COMMAREA
COBOL:
   CALL xxxxxxxx,(COMMAREA),VL
```

The COMMAREA is described below. The same data area format is used for batch and online conversions.

The program name (xxxxxxxx) is specified as follows:

```
HTML Conversions: ROPECHTM
PDF Conversions:  ROPECPDF
```

Upon return from the conversion program, the converted data will be in the specified output file. It is your responsibility to insure that the data is placed in its final location.

Online

The online conversion program is invoked using the EXEC CICS LINK command. The parameters required by the conversion programs are provided in the COMMAREA.

Before you invoke the conversion program you must clear the source and target Temporary Storage queues to be used for the conversion input and output, and you must place the data to be converted into the input Temporary Storage queue, one data line per TS Queue record.

The conversion modules are invoked by an EXEC CICS LINK PROGRAM('xxxxxxxx') COMMAREA(CPRMAREA) LENGTH(=H'94') command.

The COMMAREA, defined in detail below, is primed by you to specify the Request Information, and then populated by ROPES to contain the Response Information.

The program name (xxxxxxxx) is specified as follows:
HTML Conversions: ROPECHTO
PDF Conversions: ROPECPDO

Communication Area

The ROPECPRM (ROPES Converter Parameter Communication Area) is available in assembler and COBOL versions. The assembler version is a macro that allows you to designate the field name prefix (default: CPRM) and can map a dynamically acquired (GETMAINed) storage area (including DFHEISTG). It is shown in Figure **Figure 39** on page 166. The COBOL version is an 01 level record description that can map a dynamically acquired (GETMAINed) storage area (including Working Storage). It is shown in Figure **Figure 40** on page 166.

Communication Area Field Names and Their Uses

The COMMAREA fields are defined here. There are also brief comments in the code to remind you of their function (omitted here in the COBOL version for clarity).

CPRMAREAS
The beginning of the COMMAREA.

CPRMINTS
The name of the input Temporary Storage Queue. This queue must contain one record for each input line of the file or report to be converted.

CPRMOUTTS
The name of the output Temporary Storage Queue. This queue will receive one record for each output line of the generated document.

CPRMFCBPTR
The address of the ROPES FCB to be used for this conversion. You must bring the desired FCB into main storage prior to invoking the conversion program. If you want the conversion program to use the default values for all FCB supplied options, set this address to zero.

CPRMLINESOUT
The number of lines generated by the conversion program will be returned to you in this field. The field is 4 bytes long and is in Packed Decimal (COMP-3) format. The count includes all control and data lines.

CPRMRC
This is the conversion program return code. The possible values for this field and their meanings are:
0 - the conversion was successful
8 - the conversion failed. The error message field contains the reason.

CPRMERRM
This field contains the text of the error message that produced the return code of 8. The possible values are:
1 Unable to open the input work file.
2 Unable to open the output xxx file.
3 Unable to get necessary storage.
4 Object table size has been reached.
5 Page table size has been reached.
6 Output TS Queue cannot be cleared.
7 Error reading the input TS Queue.
8 Error writing the output TS Queue.
9 No space left in the output TS Queue.

Messages 4 and 5 are specific to the PDF conversion utility. In message 2, xxx will be PDF or HTM, depending upon the conversion program invoked.

CPRMMEDIA
This field contains a code indicating the type of Temporary Storage to be used. Specify M if TS

Main is to be used. Specify A if TS Auxiliary is to be used. Main will produce better performance for small reports, but Auxiliary is a better choice for large reports.

```
         MACRO
         ROPECPRM &P=CPRM
***********************************************************************
*                      ---     R O P E S     ---                      *
*         PDF and HTML Conversion Services Communication Area         *
***********************************************************************
&P.AREA      DS 0D         *  Start of COMMAREA.
&P.INTS      DC CL8' '     *  Input TS Queue Name.
&P.OUTTS     DC CL8' '     *  Output TS Queue Name.
&P.FCBPTR    DC CL1' '     *  Address of the FCB module to use.
&P.LINESOUT  DC PL4'0'     *  Count of output lines.
&P.RC        DC AL4(0)     *  Return code.
&P.ERRM      DC CL65' '    *  Error message text.
&P.MEDIA     DC CL1'A'     *  Converter TS Media (Aux or Main).
         MEND
```

Figure 39 - Conversion Program Commarea - Assembler

```
      ***************************************************************** 00010000
      *               --- R O P E S         ---                       * 00020000
      *     PDF and HTML Conversion Services Communication Area       * 00030000
      ***************************************************************** 00040000
       01  CPRMAREA.                                                    00050000
      *  Input TS Queue name.                                           00060000
           05  CPRMINTS     PIC X(8).                                   00090000
      *  Output TS Queue name.                                          00100000
           05  CPRMOUTTS    PIC X(8).                                   00110000
      *  Address of the FCB module.                                     00120000
           05  CPRMFCBPTR   PIC S9(8) COMP.                             00130000
      *  Lines Output.                                                  00140000
           05  CPRMLINESOUT PIC S9(8) COMP-3.                           00150000
      *  Return Code.                                                   00160000
           05  CPRMRC       PIC S9(8) COMP.                             00170000
      *  Error Message text.                                            00180000
           05  CPRMERRM     PIC X(65).                                  00190000
      *  TS Queue Media (A=Aux or M=Main).                              00200000
           05  CPRMMEDIA    PIC X(1).                                   00210000
```

Figure 40 Conversion Program Commarea - COBOL

Sample Programs

Introduction

The ROPES distribution tape contains three source modules which, when compiled/assembled and link-edited, serve as sample application programs interfacing with the ROPES system. Both batch modules, ROPEDEMO and ROPESPOL, are written in OS/VS COBOL, as is the online module, ROPEDUPL. The operation of each of these is described below.

The ROPEDUPL Sample Program

ROPEDUPL is a CICS transaction which will copy ROPES report to any other ROPES report. The transaction operates in conversational mode, attempting each requested function, displaying the ROPES response code, and inviting another request. It may be initiated from any 3270-type terminal (unless prohibited by CICS security).

To use this transaction, enter the appropriate transaction identification code from a cleared screen. The transaction will respond with a prompting map. Then, for as many times as necessary, enter the source and target ROPES report names. The transaction will respond each time with a ROPES response code and another prompting message. The response code is “0” for normal completion of the requested functions, or a one-character error code whose meaning is defined in the programming guide sections of this manual. To terminate the transaction, press CLEAR.

The ROPEDEMO Sample Program

The ROPEDEMO program is provided to enable a user to store text into a specified report. All calls to ROPES are determined and performed by ROPEDEMO; the input need not describe the sequence or nature of such calls. Input control and data card images are read from SYS005. All input is reproduced on SYSLST (DOS) or SYSOUT (OS) together with any diagnostic messages and illegal response codes. Any error will cause the program to terminate with a return code of 8.

The ROPEDEMO program is executed with the following JCL:

```
*-------------------------------------------------*
*  OS only:                                       *
*                                                 *
//TESTDEMO EXEC     PGM=ROPEDEMO
//STEPLIB     DD DSNAME=ROPES.LOADLIB,DISP=SHR
//ROPEQBR     DD DSNAME=ROPES.QBRFILE.BASE,DISP=SHR
//ROPERIB     DD DSNAME=ROPES.RIBFILE.BASE,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//SYSOUT   DD SYSOUT=A
//SYS005   DD *
     (input as described below)
/*
*                                                 *
*-------------------------------------------------*
*  DOS only:                                      *
*                                                 *
// DLBL   ROPEQBR,'ROPES.QBRFILE.BASE',,VSAM
// DLBL   ROPERIB,'ROPES.RIBFILE.BASE',,VSAM
// ASSGN  SYSLST,X'cuu'
// ASSGN  SYS005,SYSIPT
// EXEC   ROPEDEMO
     (input as described below)
/*
/&

*                                                 *
*-------------------------------------------------*
```

The input consists of packets of cards. Each packet begins with a “report name card,” containing a “$” in column 1, and the name of the destination report in columns 2-9. (The remainder of the report name card is ignored.) The text to be stored in the named report is punched in subsequent cards, with ANSI carriage control characters punched in column 1, until the end-of-file is encountered on SYSIPT or a “$” is found in column 1, indicating the start of a new packet.

Your input stream may include one or more of the following command cards to perform the functions indicated. The command cards will indicate the end of a test packet. The command must be punched in columns 1 - 9 of the input card.

$CHKPOINT Command

This command will cause the Report Information Block to be checkpointed. The checkpointing will occur immediately. All report data added to the queue prior to the $CHKPOINT command will be preserved.

$DELREPT Command

This command causes all data entered in the current test packet to be discarded. If the data has previously been checkpointed, this command will cause an error to be indicated.

Your test packets should specify report names which have been defined at your installation.

The ROPESPOL Sample Program

The ROPESPOL program is provided to enable a user to store text into a specified report from a print-image data set. All calls to ROPES are determined and performed by ROPESPOL.

The ROPESPOL program is executed with the following JCL:

```
*-------------------------------------------------*
*  OS only:                                       *
*                                                 *
//RUNSPOL  EXEC  PGM=ROPESPOL,PARM=reportid
//STEPLIB  DD    DSN=ROPES.LOADLIB,DISP=SHR
//ROPEQBR  DD    DSN=ROPES.QBRFILE.BASE,DISP=SHR
//ROPERIB  DD    DSN=ROPES.RIBFILE.BASE,DISP=SHR
//REPORT   DD    DSN=user.data.set,DISP=SHR
*                                                 *
*-------------------------------------------------*
```

The user data set, REPORT, is a blocked or unblocked sequential data set of fixed length, 133 character records with ROPES carriage control characters in position 1 of each line. The EXEC statement PARM field contains the name of the ROPES report to receive the data.

Index