Output Management GUI Client

Concepts

Version 3.4.3

November 2016

This document applies to Output Management GUI Client Version 3.4.3.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: NGC-ONOMCONCEPTS-343-20161111

Table of Contents

1 Concepts and Facilities

This document describes what Entire Output Management can do and how it works. It covers the following topics:

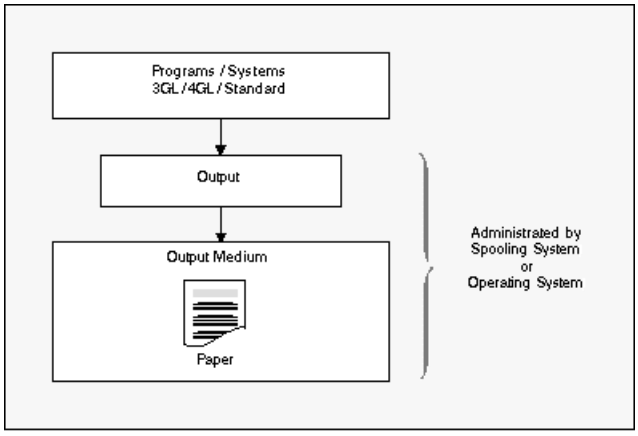
What Can Entire Output Management Do?

Entire Output Management is a tool that can process every kind of print data in heterogeneous client/server environments, rule-based and automatically in a user-friendly format, without changing the applications or programs that created the data.

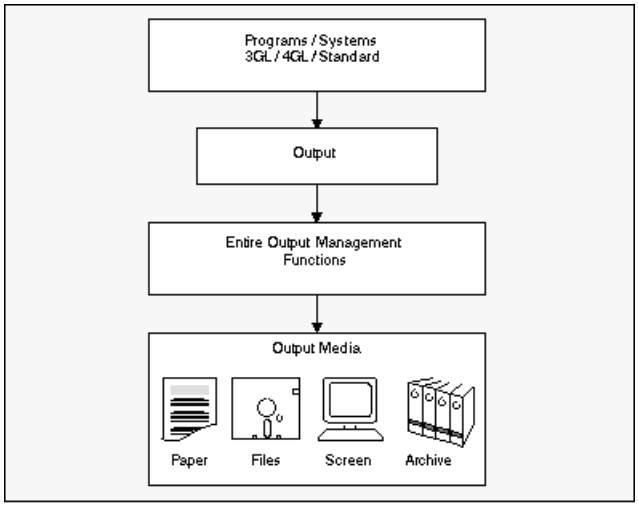
For this reason, a data center can use Entire Output Management to automate the distribution of print data. Individual departments can also use Entire Output Management for presenting their data on screen, so they can be used immediately for decision-making, without having to resort to hard copy. One of the main goals of Entire Output Management is to save on printing costs.

Entire Output Management functions like a logical administration system for data: it has clearly defined interfaces to existing spooling systems and the functionally equivalent parts of an operating system.

Administration of Program Output without Entire Output Management



Administration of Program Output with Entire Output Management



This means that the programs producing the data remain unchanged: output is produced in the usual way and is passed to the spooling system for administration. Instead of printing these data automatically, Entire Output Management reads them and uses predetermined rules on them.

Among its numerous processing options, Entire Output Management, then allows you to return all or part of the data to the spooling system, which in turn controls the physical printing process.

How Does Entire Output Management Work?

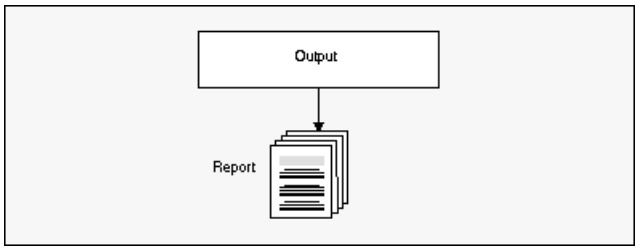
- Archiving/Reviving

General

Entire Output Management allows you to adapt your present data distribution system step by step. You can also specify characteristics to identify the processes or programs whose output Entire Output Management will handle.

In addition, you can determine which parts of the output from the identified programs is to be further processed by Entire Output Management. This is called *separation* and the result is a *report*.

Reports

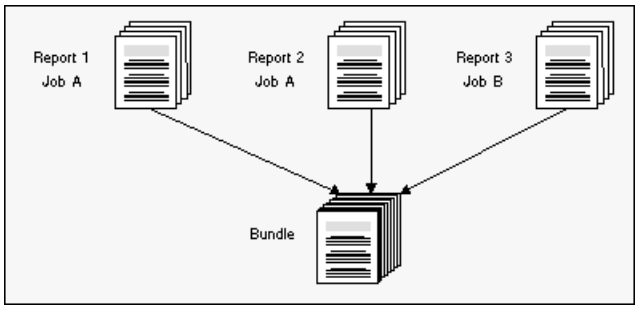


A report contains those parts of the output that are important for the recipient. Entire Output Management's rules make it easy for you to determine what is important.

Entire Output Management by itself allows you to create and process reports.

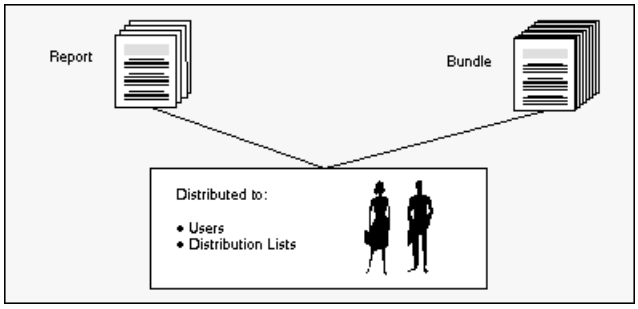
Bundles

Reports can be combined to form larger packets, which are called *bundles*. This bundling is possible even if the reports come from different data sources (for example, jobs).



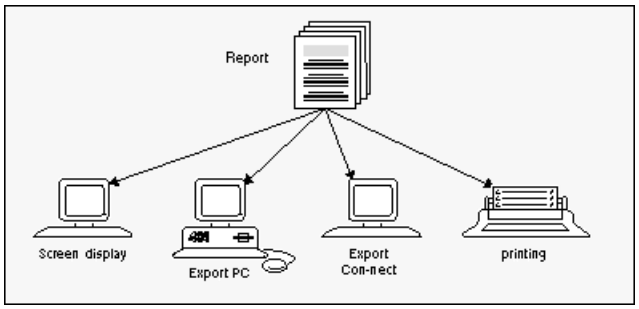
Distribution Lists

Distribution lists, which contain the names of individual users or the names of other distribution lists or both, allow you to distribute reports and bundles to selected users.



Only the members of the distribution list of a report or bundle have access to the report or bundle contents (multi-client facility) and they can display these data on the screen.

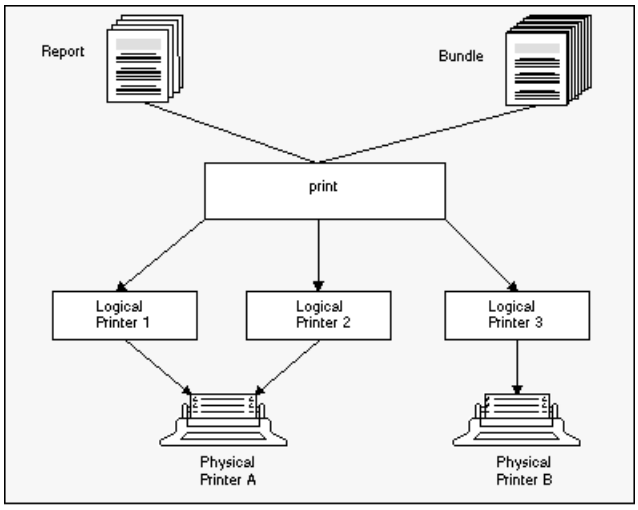
Data Distribution



Up to now, the user had to view data on paper after an involved manual distribution process and in a form determined by the application that produced the data. But now Entire Output Management enables the user to display the important data directly at his or her terminal. And all or part of the data can be exported for example to Con-nect, Software AG's office communication system, or to a PC for further local processing.

You can, of course, also print reports and bundles. This can be done either automatically with a rule-based process or manually, when requested by a user viewing the data at his or her terminal. Printing in Entire Output Management means the selected data (bundles, reports, subsections of reports) can be formatted for a particular printer. They are then passed to a logical printer.

Printing

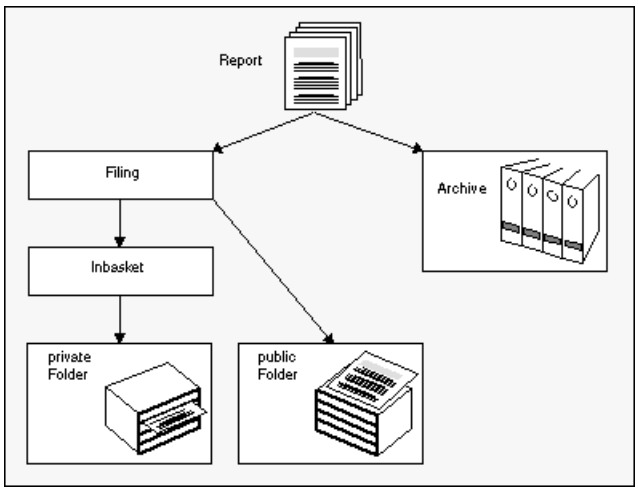


A *logical printer* is a specific set of attributes linked to a *physical printer*, which determine the printing format for the physical printer.

Reports and bundles have an individual lifetime, during which they are available online. Each member of a distribution list can authorize other users to access those reports for which he or she is authorized. Such a report appears in the inbasket of the recipient, from which he or she can file it in individually defined folders.

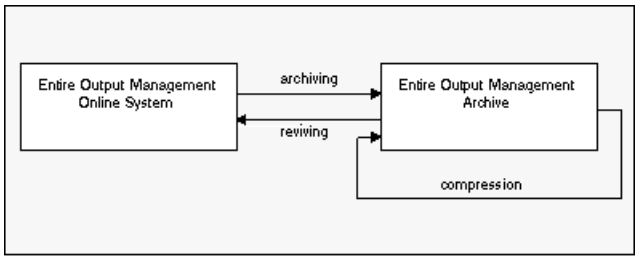
Folders

There are also public folders, to which each Entire Output Management user has access. The procedure described above is, of course, not necessary for the public folders.



Archiving/Reviving

When the lifetime of a report has expired, the report can be archived on storage media.



Entire Output Management keeps an archive account and allows you to revive archived reports if you need them online again. Archive storage media which are no longer needed are freed during compression.

Condensing (Compression)

"Condense" jobs read archives and delete all data whose retention period has been exceeded and which are no longer needed. This helps to reduce space for archive data (which are sequential files) to a minimum. With the function Automatic Archiving Defaults you can define a threshold for the number of reports in an archive data set that will trigger its being marked for condensation.

Cleanup

This section provides an overview of the various cleanup functions provided by Entire Output Management. It covers the following topics:

Cleanup Types

There are three types of cleanup:

- daily cleanup,
- source cleanup,
- report cleanup.

The primary cleanup type is daily cleanup. Entire Output Management performs its daily cleanup once per day on its next activation after the time specified in the system defaults (option 8.1.1). If no time is specified there, the daily cleanup will be done on the first activation after midnight.

Source cleanup and report cleanup are optional and can be activated in option 8.1.7. Each can be activated individually and a schedule defined for it. They may be scheduled to run whenever you wish, using a calendar to specify on which days of the week or month they are to run, whether they should run before or after non-working days, the time window during which they are to run and how often. For example, you could define that source/report cleanup is to run every 2 hours between 08:00 and 20:00 on Mondays, Wednesdays and Fridays.

Daily Cleanup

The daily cleanup processes all expired entities:

- Active reports that have expired are purged or marked for archiving, as appropriate.
- Active bundles, printouts and log records that have expired are purged.
- Revived reports are “unrevived”, that is, the contents are deleted if the report was revived to the database and the active report is reset to its archived state.
- Archived active reports are deleted.
- Active report sources (spool files or container file entries) are checked and, if no longer needed, deleted.

Source Cleanup

Source cleanup checks Entire Output Management sources (for example, JES or Power spool files in Entire Output Management’s temporary classes or entries in the container files) to see if they are still needed - and deletes them if they are not needed. They are no longer needed when there is no active report with location "S" referring to them. Container file entries which are no longer needed are deleted even if Spool Cleanup is set to "N".

Report Cleanup

Report cleanup checks all active reports with location "S" to establish if the source file is still available. If it is not, the active report is deleted.

Daily Cleanup – Online or in Batch Mode?

The daily cleanup process may entail a lot of work (typically it can run between 30 and 60 minutes in a production environment), and during this time the monitor can do nothing else; this means that no reports, bundles or printouts can be processed until the daily cleanup has finished. This is because the monitor is a single task.

To avoid this problem, the daily cleanup can run in an asynchronous batch job. To do this, you execute the program NOMCLEAN in a standard batch-mode Natural (with the necessary parameters set, such as LFILE 206). This batch job can then be activated by a job scheduler to run a couple of hours before the Entire Output Management monitor is scheduled to perform the daily cleanup. When the monitor is about to perform the daily cleanup, it can ascertain that NOMCLEAN has ran and so it will not have to do anything.

Communication

NOMCLEAN and the monitor keep each other informed of the current status by using two control records in the Entire Output Management system file. These control records are accessed using the view NRM-ARCHIVE-TASK and the descriptor M-MONITOR-ID = 'CLEANUP' or M-MONITOR-ID = SYSNOM (actually the library Entire Output Management is running from).

The communication paths are as follows:

- NOMCLEAN at its start sets SYSNOM's CLEANUP-IDENTIFIER to CLEAN, CLEANUP-ACTIVE-NOW to TRUE and CLEANUP-RESCHEDULE to FALSE to show that it is active.
- RMONITOR (monitor main loop) only deletes no longer needed sources if SYSNOM's CLEANUP-IDENTIFIER is not CLEAN or CLEANUP-ACTIVE-NOW is not TRUE (that is, only if NOMCLEAN is not currently active). In addition, to avoid concurrency problems while creating active reports, RMONITOR sets CLEANUP's REPORT-PROCESSING-NOW to TRUE while creating active reports, and to FALSE when it has finished. This is necessary to prevent NOMCLEAN from deleting reports or sources that are being worked on by the monitor.
- RMARCSCH (monitor scheduled functions) only performs daily cleanup if SYSNOM's CLEANUP-RESCHEDULE and CLEANUP-ACTIVE-NOW are both FALSE (that is, only if NOMCLEAN is not currently active and has not already run on that day). If CLEANUP-RESCHEDULE is TRUE, this means that NOMCLEAN has already run so RMARCSCH resets CLEANUP-RESCHEDULE to FALSE and updates the next scheduled daily cleanup to the same time on the next day. RMARCSCH only performs source/report cleanup if SYSNOM's CLEANUP-ACTIVE-NOW is FALSE. This is to avoid collision with NOMCLEAN which may also be cleaning up sources.
- RMSRCK (source/report cleanup) may be invoked by the monitor or NOMCLEAN. When invoked by NOMCLEAN, it needs to avoid the concurrency problem described above. It does this by checking REPORT-PROCESSING-NOW and, if it is TRUE, uses the NPR EVENTING view to make itself dormant (repeatedly, for 5 seconds at a time) until REPORT-PROCESSING-NOW is FALSE, which is when RMSRCK resumes processing where it left off.

If multi-tasking is active (more than one monitor task is defined), the monitor cleanup functions will be performed by the main task, task 1. This will allow cleanup to be separated from other monitor functions like active report and printout creation.

TCP/IP Direct Printing

Entire Output Management provides methods for accessing printers: It is able to directly print to TCP/IP printers. This print method is intended to access printers that have either got a dedicated IP address or can be reached using printer queue names of printer servers.

This print method is intended for direct printing of shorter documents directly from online applications. However, it is executed in a Natural subtask and no online space (such as CICS memory) is affected. No spooling system is required, and no extra address space will be used, not even an ESY server. This makes this print method independent from batch jobs, Broker, CA-SPOOL, ESY, JES, and Natural Advanced Facilities.

Performance enhancements are achieved, and 4 MB of space in an Entire Output Management printer subtask per 1000 pages of text output are needed as soon as the printer (server) acknowledges the reception of the printout.

At print time, the printout will be initiated online or by the monitor and executed in one of the defined print tasks of Entire Output Management. At execution time, the printout will be kept in virtual memory and then sent to TCP/IP directly (via socket programming) using the LPR/LPD protocol (RFC1179).

Processing of Binary Data

Entire Output Management also allows you to process binary data. This means that any kind of file or printout can be kept in Entire Output Management, archived, printed, distributed, and passed to a destination system.

This is possible with UNIX and Windows file systems, but not with mainframe spool systems.

Note: If binary data are read from a mainframe source (spool or dataset), subsequent processing may cause Natural abends like `IW060` or `0C7`. On mainframes, only text data can be processed. For performance reasons, however, no check is performed to ensure that the input data do not contain binary data.

Basically there are two ways to transfer binary data to Entire Output Management:

- You can use the direct-input interface together with the Open Print Option (OPO). A Windows client passes output of the Windows systems to Entire Output Management. This can be the output of a Windows printer driver or any program that can forward data to OPO using the

pipe mechanism of Windows. The output is redirected to Entire Output Management using the Remote Procedure Call (RPC) facility of EntireX.

- You can define a report that receives binary data from a UNIX or Windows directory. In the report definition, the parameter "binary reading" must be set to "B" to indicate that the file is to be processed in binary format.

The report is passed through the Entire Output Management trigger queue, similar to the NOMPUT interface. Therefore the trigger queue must be activated (NOM API and User-Exit Defaults). The opened active report receives the type "binary" (a special CC type).

A binary document cannot be browsed with the mainframe character interface. However, the Entire Output Management GUI Client is able to receive the data, store them on the PC (please note that the whole file has to be transferred) and browse (view) them with the Windows default application of the specific file type (such as ".doc" or ".pdf").

Binary active reports can be printed with UNIXLP printers or on BS2000/OSD with a SYSPRBS2 printer type. Under UNIX, a printer type NATUNIX is available which can pass the data to the "lp" or "lpr" utility (in which case the option "-l" has to be used to ensure that the data are forwarded in a transparent mode) or to a program or script, or to a file in a UNIX directory.

Please note that a binary report that is an output of a Windows printer driver will be bound to the hardware it was created for. The decision where to print the data Entire Output Management may have received a long time ago is made at creation time. Consider to use file formats like PDF if you want to keep binary data for a long time.

Entire Output Management converts binary data to the BASE64 format. This makes it possible to pass the data across platforms and code pages, because BASE64 consists of printable characters only. However, this also means that the data are larger than the original binary data stream. When active reports are output to destinations, the BASE64 data stream is decoded to binary format again.

The output of printer drivers often result in large data streams, because a printer driver creates print pages with all graphical statements of the appropriate printer language.

In theory, separation exits can be used. However, binary data cannot be separated with standard exits, as they are not readable.

Any report Entire Output Management receives via the OPO can be enriched with user-defined meta data. The meta data can be viewed via PF2 on the spooling attributes screen of an active report, or in any user exit that uses the field `SPOOL-ATTRIBUTES-EXTENDED`. Meta-data values that exceed the length of the terminal line size of the Entire Output Management character interface will be truncated to one line and marked with the continuation marker "...". Meta data are passed to Entire Output Management via an XML file that can be defined in an OPO printer definition.

For further details and examples, see the section *Setting Up Environments for Binary Documents* in the *System Administration* documentation.

Transferring Output to Entire Output Management under UNIX

Entire Output Management can handle text output from mainframe operating systems and from UNIX or Windows systems. This output has to be stored as ASCII or EBCDIC files, on mainframes with or without control characters.

Basically, Entire Output Management on UNIX handles the same output formats as on mainframes unless binary data are to be processed. Binary data have to be converted into the ASCII data format before being processed by Entire Output Management.

To get output data from UNIX or Windows directories, use the UNIX node definitions as described under *UNIX Defaults* in the *System Administration* documentation.

UNIX applications can print into a Entire Output Management file directory (ASCII files) or access spoolers like CUPS to print output to the queue of a virtual printer. In this case the printer queue of CUPS has to be connected to a CUPS backend like "pipe" (default: `/usr/lib/cups/backend`) which stores the data as ASCII files in the Entire Output Management directory defined in *UNIX Defaults*.

However, some interfaces do not exist under UNIX and can therefore not be used in an Entire Output Management UNIX environment:

- CA Spool
- NOMPUT (unless NOMPUT is executed on a mainframe with an Adabas UNIX Entire Output Management container file)
- SAP

Printing under UNIX

Under UNIX, Entire Output Management is designed to only print on the printer types NATUNIX and DISKUNIX. These are described under *Special Printer Types* in the *System Administration* documentation.

Converting the Report Format

A report can be converted into common multimedia file formats either when it is loaded or when it is printed.

Converting When Loading

Note: The functionality described in this section is not yet available. It will be made available with the next version.

A report can be converted into common multimedia file formats when it is loaded. The source format must be text, PDF or PostScript (mainframe formats like ASA on IBM operating systems or machine code on Siemens OSD are treated as text) . The ASCII text format will also be processed.

The conversion destination format will be the format which is used to store the data in Entire Output Management.

The formatting parameters are passed to the utilities Enscript and Ghostscript to convert the input data stream into the desired format which is to be stored in Entire Output Management as a binary report.

If the original format is PDF or PostScript, all Enscript formatting parameters will be ignored because the report is already formatted (rendered).

To achieve this, an Entire System Server UNIX node must be defined which runs on a machine where the utilities Ghostscript and Enscript are installed. For systems which run Entire Output Management on UNIX, this can be the local Entire System Server node. If a conversion is to be processed on mainframe systems, a remote Entire System Server UNIX node must be specified which will perform the conversion and send the files back to Entire Output Management. If no separate UNIX machine is available for mainframe systems, you may consider using a z/Linux system for the conversion.

The conversion is triggered by specifying the field **Report Format** in the report definition (see *Formatting Attributes* in the *User's Guide*).

Converting When Printing

It is also possible to convert the output to common multimedia file formats when it is printed. The printer type DISKUNIX is available to convert a file after it has been written to disk. As a prerequisite the report to be converted must be stored in Entire Output Management in the format "text" (ASCII, ASA, machine code), PDF or PostScript (or mixed in bundles). The DISKUNIX printer will convert text reports to the desired format according to the format definitions. Reports that are already in PDF or PostScript format will not be formatted because they are already rendered. PDF and PostScript formatted reports will only be converted to the desired output format without applying the Enscript parameters (see *Formatting Attributes for File-Format Conversion* under *DISKUNIX Printers* in the *System Administration* documentation). If separator pages are defined, however, they will follow the format definitions as they are text portions of the report. If bundles are to be printed DISKUNIX will collect all reports, regardless if they are in text, PDF, or PostScript format and put them into a single file of the desired output format. All text portions (text reports, separator pages) will be converted as defined in the format parameters. DISKUNIX uses the utilities Ghostscript and Enscript to perform the conversion; these haves to be installed on the destination UNIX system to which DISKUNIX writes.

IF UTF-8 is used as the file code page for reading and printing, you can use the UNIX utility Uniprint as the renderer instead of Enscript. Uniprint is part of the UNIX package "yudit" (Unicode editor), which also has to be installed on the node on which the conversion is done.

The conversion is triggered by specifying the attribute field **Output Format**.

PDF formatted reports can use a mask file for every page. This mask file is a PDF file with transparent background. This file is treated as a stamp on each page: Only the parts of the mask file which are transparent will show the original report. This means that logos or forms can be integrated into a PDF report. If the mask file contains more than one page, the corresponding pages of the original report will be overlaid.

To use the overlay function in Entire Output Management, the package "pdftk" (PDF Toolkit) has to be installed on the converting node. It is available for Windows and Linux machines. If the field **Command** is filled, this command line can be used to further process the resulting file after the conversion For example, the utilities "ps2afp" or "pdf2afp" (contained in the IBM product Infoprint) can be used to finally create AFP-formatted files for further processing.

The above-mentioned UNIX utilities are third-party products which are *not* part of Entire Output Management; Software AG neither delivers them nor provides support for them.

Conversion of Special Characters

Entire Output Management transfers data from and to Entire System Server on UNIX and the Open Print Option (OPO). If the transfer is performed remotely, the code pages of the source and destination machines will differ in most cases. This means that you have to implement code-page translation using one or more middleware products. As a result, you may end up with complex environments in which special characters are still not always translated to the desired characters on the destination machine.

To avoid this problem, Entire Output Management, Entire System Server and OPO send named characters instead of the characters themselves. Entire Output Management uses the trigraph feature of Entire System Server and HTML named characters with OPO. This means that only the ASCII 7-bit and basis EBCDIC tables are used for the conversion of characters across platforms.

Adabas Files

Entire Output Management uses the following Adabas files:

- definition-data file (logical file number 206)
- active-data file (logical file number 91)
- container files and special trigger container file.

You can use the same Adabas file for both the definition-data file and the active-data file (however, the logical file specifications for both 206 and 91 are necessary).

Container Files and Active-Data File

Entire Output Management can copy report contents from their original location (for example, JES spool) into a container file or into the Entire Output Management active-data file, or both.

Copying into the active-data file is independent of any container file usage and will only be done for reports that are defined with the option **Copy report content to NOM database** set to "Yes".

On z/OS systems, reports should only be copied into the active-data file if absolutely necessary (for example, to avoid accidental loss through spool deletion), because it is a big overhead to store large reports in the database.

A special container file is the one in which trigger data are stored. It is necessary for Entire Output Management on UNIX, the Open Print Option and Natural Advanced Facilities.

Usage of Container Files

Entire Output Management copies report sources into a container file under one of the following circumstances:

- If Entire Output Management on UNIX, the Open Print Option, or the Output Management GUI Client is used.
- If the report is from CA Spool, Natural Advanced Facilities, SAP, or the 3GL interface (including the VTAM virtual-printer application `NOMVPRNT` with the parameter `STORE=DB`).
- In BS2000/OSD, if the option **Copy files** is set to "Y" in the **Monitor Defaults** and at least one destination is defined, as described under *Defining Container Files*.
- In JES2, JES3 and POWER, if a spool file is processed with a `DEST` specification that matches one of the destinations defined in the **Monitor Defaults** (see *Defining Container Files*).

Defining Container Files

Container files for mainframe spooling and job-entry systems, together with the spool destination that will be copied into the associated container file, are defined in **Monitor Defaults** as described under *Defining Container Files*.

Container files for CA Spool, Natural Advanced Facilities, SAP Spool and the 3GL Interface are defined in *CA Spool Defaults*, *Natural Advanced Facilities Defaults*, *SAP-Spool Defaults* and *3GL Interface Maintenance* respectively.

The special container file for trigger data is defined in the *NOM API and User Exit Defaults*.

Compression and Blocking

Entire Output Management stores output in arrays of 16 KB. Adabas compresses null values and trailing blanks in alphanumeric fields. Text data of output often contain blocks of blanks to present the data in the form of lists or tables. These blocks of blanks are compressed by Entire Output Management itself. This results in a more efficient use of Adabas database space and fewer databases accesses, thus improving performance.

Direct Access

Separating, browsing and printing in parts usually results in the need for direct access to record ranges of the output. This, however, is not given in mainframe spooling systems and disk data sets with a variable record length. Access to the Entire Output Management container file is extremely fast.

Entire Output Management does this storage into the container file before processing any report definitions. The original output is copied as a whole, all following separation processing, browsing and printing in parts will be very fast.

Entire Output Management keeps the whole original output stored in the database as long as there is at least one active report with location "S" (= source) pointing to it. This could mean that the container file is filled with very large output, even though only a fraction is actually needed.

It makes sense to store in a container file whenever intensive separation processing, browsing or printing in parts is necessary.

If you have intensive separation processing, but the resulting reports out of the whole original is only a fraction, set the option **Copy report content to NOM database** in the report definition(s). The resulting reports will be copied from the original output residing in the container file into the Entire Output Management system file, and the location of the report will become "D". When the next cleanup is done and there are no reports with location "S" pointing to the original source, it will be deleted from the container file.

Be aware though, that for the lifetime of the original output in the container file, reports created with **Copy report content to NOM database** in the report definition(s) mean that those parts of the output are indeed stored again in the active-data file, whereas with location "S" Entire Output Management would keep only pointers to the container file.

In summary:

- Use container files for heavy processing, for example, when separation exits are used.
- Use the special container file for trigger data with Entire Output Management on UNIX, Open Print Option and Natural Advanced Facilities.
- In addition, use **Copy report content to NOM database** (active-data file) if only small parts of the original are needed or if the resulting reports have very different expiration dates.
- Remember that the report with the highest expiration date will determine the lifetime of the whole original output in the container file.

Overview of Entire Output Management's Functions