Adobe

Adobe® Central Output Server

Version 5.7

Working with Central

for Microsoft® Windows®

Working with Central for Microsoft Windows
Edition 3.0, March 2009

As of April 12, 2002, Accelio Corporation (formerly JetForm Corporation) was purchased by Adobe Systems Incorporated. As of that date, any reference to JetForm or Accelio shall be deemed to refer to Adobe Systems Incorporated.

Technical information contained herein is subject to change without notice.

Adobe Systems Incorporated does not warrant that Adobe Central Output Server will function properly in every hardware/software environment.

Trademarks

Contents

1 Overview

Adobe® Central Output Server (Central) is a powerful server-based application that enables you to manage and administer your electronic templates in a networked environment.

This guide provides instructions on:

- Administering Central on a day-to-day basis.
- Setting up the Job Management Database.
- Controlling Central.
- Using features such as variable substitution, JFNOJOB, and custom Agents.

This guide is intended for:

- System administrators who are responsible for the ongoing administration of Central.
- Application developers who require an understanding of Central processing and the Job Management Database.

Central Processing

Central accepts transaction files and routes them to various Agents by following a series of processing steps defined in the Job Management Database.

The Job Management Database

Central uses a Job Management Database, jfserver.jmd, to define the steps required to process a transaction file. You use Central Control to maintain the Job Management Database. For completed details on creating table entries, see the guide, *Central Control*. To access this guide, from Central Control, click **Help** > **View Documentation**.

The Job Management Database consists of several tables including:

- Job Table - defines each step that Central must perform in processing the transaction. The job name from the **^job** command in the transaction file is the key into this table. The table includes the job name, logical output device and other printer specific information, template name, preamble file name, and the task identifier for the program that Central initiates to execute the step.
- Printer Table - maps a logical output device name to a physical output device. This table also contains command line options or variable substitution arguments that Central passes to Print Agent at print time.

- Task Table - maps a task identifier to a task name, which is an executable name for the Agent or a custom application, that Central executes to complete the step. This table also contains command line options or variable substitution arguments that Central passes to the task at execution time. When you install Central, this table includes several entries that define tasks such as:
  - The task to execute after Central is initiated.
  - The task to execute prior to Central shut down.
  - The task to execute when an error occurs.
  - The task to execute when there is no **^job** command in the transaction file.
  - Tasks associated with downloading and refreshing macros in the printers.
- Managed Memory Table - defines the amount of managed memory that Central controls for a specific printer. A Load Flag of “M” on the Job Table identifies those macros that Print Agent downloads into managed memory.

The ^job Command

Client applications write a data stream to the collector directory, creating a transaction file for Central. Generally, each transaction file must contain a **^job** command as the first line in the file, that identifies the name of the transaction to Central. The rest of the transaction file must be in a format acceptable to the tasks that process the transaction file. For example, the ASCII file given to Print Agent must conform to the processing modes described in the *Print Agent Reference* guide. The **^job** command in the transaction file is often referred to as a job card.

The **^job** command has the format:

```
^job jobname [-zprtname] [-axxtaskid] [-ajijobid] [jobtokens]
```

where:

`jobname` is a logical name for the transaction, as known to the application or user and as it appears in the Job Table of the Job Management Database. `jobname` can be up to 51 characters in length and cannot contain blanks. This value is case-insensitive.

`-z``prtname` (optional) identifies the logical printer. `prtname` is a logical printer name from the Printer Table of the Job Management Database, or an “*” (asterisk), as in **-z***. When the `prtname` does not exist on the Printer Table, Central considers it a valid network output device where output is written.

`-axx``taskid` (optional) identifies a task that Central executes exclusively. `taskid` appears in the Task Table of the Job Management Database. `taskid` can be up to nine characters in length and cannot contain blanks. This value is case-insensitive. If the **^job** command does not include this option, a task identifier must be present on the Job Table for the `jobname`.

`-aji``jobid` (optional) is a user assigned identifier for the job.

`jobtokens` (optional) are other options provided to the task by Central. `jobtokens` may be any command line options supported by the executable program that Central initiates, or the variable substitution arguments (see page 49) supported by Central.

Other characters may precede the **^job** command on the first line of the transaction file. Central ignores these characters. The characters may appear because the file transfer utility program that placed the transaction file in the collector directory inserted the characters at the beginning of the file. If a character other than the "^" (caret) precedes the word "job", then this is the character used as the prefix character for commands included in the transaction file.

^job Command Validation

By default Central accepts the first line of the transaction file as a valid **^job** command if it contains the three sequential letters "job". This allows for files that may have extraneous characters at the beginning of the first line. However, it can also cause Central to mistake a line of data for a **^job** command. You can impose stricter validation using the "JobValidation" setting in the [Options] section of the Central configuration file, jfserver.ini.

The value of JobValidation is a string containing lower-case letter codes (in any order) for each of the validations that you desire. The letter codes and their meanings are:

Letter Code	Description
`a`	"job" must occupy columns two through four.
`b`	"job" must be immediately preceded by "^" (caret).
`c`	"job" must be immediately followed by whitespace.
`d`	"job" must not be immediately preceded by whitespace, that is, the prefix character must not be whitespace.
`e`	"job" must not be immediately preceded by an English alphabetic or numeric character, that is, the prefix character must not be alphanumeric.
`f`	The **^job** command must have a job name, that is, following "job" there must be whitespace and then at least one printable character.
`g`	The job name must start with a letter of the English alphabet.
`h`	The job name must consist entirely of English alphabetic or numeric characters.
`i`	The job name must consist entirely of English alphabetic or numeric characters or the characters "_" (underscore) or "$" (dollar sign).
`j`	All of the job options (everything after the job name) must start with the character "-" (minus). Options may contain whitespace as long as it is properly enclosed within quotation marks. This is purely a syntax check - the options themselves are not interpreted at this stage.

Note: Validations `f` through `j`, inclusive, are never applied to a **^job** command that contains an **-axx** option. The letters **"axx"** can be in upper, lower, or mixed case.

For example, if jfserver.ini contains …

```
[Options]
JobValidation=abcfij
```

… Central interprets the first line of the file as follows:

First Line of the Data File	Valid/ Invalid	Reasoning
x^job abc	Invalid	Fails validation `a` - “job” must occupy columns two through four.
xjob abc	Invalid	Fails validation `b` - “job” must be immediately preceded by “^” (caret).
^jobabc	Invalid	Fails validation `c` - “job” must be immediately followed by whitespace.
^job	Invalid	Fails validation `f` - the **^job** command must have a job name.
^job /abc	Invalid	Fails validation `i` - the job name must consist entirely of English alphabetic or numeric characters or the characters “_” (underscore) or “$” (dollar sign).
^job abc def	Invalid	Fails validation `j` - there is no parameter starting with “-”.
^job abc -d“xxx	Invalid	Fails validation `j` - the option is improperly quoted.
^job /abc def -axxpppp	Valid	Although this **^job** command would normally fail validation `i`, it is exempt from it because of the presence of the **-axx** option.
^job abc -dxxx -e“yyy zzz”	Valid	Meets all JobValidation criteria.

The XML Job Card

Historically Central processed data files which were organized as sequences of lines. In keeping with this, the job card with the **^job** command, if it was present, was simply the first line in the file. This approach does not work for XML, as XML imposes strict syntax rules. In particular, something like:

```
^job myjob
<abc>
  <def>laksjf</def>
</abc>
```

is NOT valid XML. XML processors not only cannot handle a file like this, they are REQUIRED to refuse to even try.

Effective with the 5.6 release, Central has been enhanced to perform a separate parse for a job card embedded in an XML processing instruction. This test is done prior to the regular job card parsing. If an XML job card is found, it is used. If it is not found, Central re-parses at the start of the data file using the old method, that is looking for a first-line data card. Hence processing of data files that do not have an XML job card in them is not affected in any way by the change. Furthermore the syntax of an XML job card is tightly defined (in keeping with XML practice) so there is essentially no chance of a false positive (that is, of an XML job card being seen where none was intended).

An XML job card does not have to be the first thing in the data file. It can be anywhere within the first 4096 bytes of the file. However, it must be completely contained within that 4096 bytes - that is, it must not run over the end of the 4096 bytes. Central reads only the first 4096 bytes into its buffer.

The syntax of an XML job card is:

```
<?adobecentral [^]job[ parm]...[ ]?>
```

Square brackets signify an optional item. The components of the processing instruction are as follows:

`<?` is the opening tag for the processing instruction.

`adobecentral` is the application identifier. This is case-insensitive. This token CANNOT be preceded by whitespace and MUST be followed by whitespace.

`[^]` signifies an optional command prefix character. If not supplied this defaults to “^”, just as it does for a non-XML job card.

`job` identifies this particular processing instruction as a job card. This is required and it is case-insensitive. If this token has any other value, the whole processing instruction is silently ignored on the assumption that it is some future processing instruction which has not been implemented in the running release of Central; parsing continues, searching for an XML job card farther down in the buffer.

`[ parm]...` signifies zero or more job parameters. Each one must be preceded by whitespace, which as always may include newlines. Parameters are processed by the same rules as always, so that the first parameter that does not start with “-” (or “/” on Windows) is the form name.

`[ ]` signifies optional trailing whitespace before the closing tag. As always the whitespace may include newlines.

`?>` is the closing tag for the processing instruction.

For non-XML files, or XML files which do not contain an XML job card, Central searches for an XML job card in the buffer and fails to find it. When this happens it silently reverts to the old method, reprocessing the same buffer.

If, while searching for an XML job card, Central finds what looks like the start of one but the syntax is incorrect, it issues a warning message and continues searching forward in the buffer for a valid XML job card. The possible syntax errors are:

- no whitespace after `adobecentral`. This must be followed by whitespace.
- `job` is immediately followed (without whitespace) by something other than the closing “?>”. `job` must be separated from parameters by whitespace.
- no closing “?>”. A possible cause of this is that the processing instruction straddled the 4096-character limit.

Note that it is NOT a syntax error for `job` to be replaced by something else. As explained above, such a processing instruction is silently ignored.

When Central finds an XML job card in the buffer, it removes line breaks before passing the job card on for processing. This preserves the readability and parsability of log messages that quote the job card.

If no job card of any kind is found, or a job card is found but no parameters, Central passes the data file to JFNOJOB in the usual manner.

Central can optionally perform a series of validations on first-line (non-XML) job cards, as outlined in the section “^job Command Validation” on page 8. These validations are provided primarily to reduce the likelihood of false positives which interpret a line of user data as a job card. These validations are unnecessary for XML job cards, because XML inherently separates out user data from processing instructions. Therefore those validations are not performed for XML job cards, regardless of the option setting. On the other hand, the validations described above for XML job cards are always performed, regardless of any option setting.

When the job card is an XML processing instruction, Central sets the value of @SkipLines to “0”. This is because XML may not parse properly if the first line is excluded. In previous releases the value passed to Print Agent via -asl@SkipLines was ignored for XML files, and this is still the case, but Central nonetheless tries to respect the fact that it is dealing with XML. Note however that Central only knows the file is XML if it contains an XML job card. If it is XML but does not contain an XML job card, Central does not know that the data file contains XML so it still sets @SkipLines to 1.

Considerations when Processing XML Files

When processing XML files through Central, be aware of these considerations:

- If you are generating new XML data files for processing by Central, you should include the new XML job card somewhere within the first 4 Kbytes of the data file. You can then freely process the data file through XML processors confident that the job card will be preserved intact.
- The XML file must be coded as UTF-8 when it is submitted to Central. Note that ASCII is a subset of UTF-8, so an ASCII file is acceptable. The XML file cannot be encoded as UTF-16, UCS-2, or UCS-4. These coding requirements are unchanged from previous releases.
- When the job card is an XML processing instruction, Central sets the value of the new @XMLData substitution variable to “Yes”. This is incorporated into the JFMERGE task table entry as “-axml@XMLData” so that when an XML job card is found, Print Agent is instructed to process the transaction file as XML. For details about the @XMLData substitution variable, see “Variable Substitution” on page 49.

Backwards Compatibility

Some users of Central version 5.5 and earlier have been finessing a non-XML job card into an XML transaction file as follows:

```
<?xml version= "1.0 "?><?jetform ^job myjobname -myparm
?>
```

Note that the “?>” which completes the “jetform” processing instruction is preceded by an ASCII newline character. Using the old non-XML job card processing method, Central considers the newline to be the end of the job card. It also discards all of the text before the “^job” in accordance with the standard processing rules. This technique still works. Since the processing instruction target is “jetform” instead of “adobecentral”, the processing instruction is not recognized by the new XML job card detection, and Central falls back onto the old method. Hence the transaction is processed in exactly the same way as for releases prior to 5.6.

However, although this technique works, it is not recommended because it is fragile. If the XML transaction file is passed through a third-party XML processor, the third-party processor is likely to discard the line break before the last “?>”. XML processors are allowed to do this because the rules of XML state that a newline between “-myparm” and “?>” is not significant. For new projects, the new XML job card mechanism should be used instead.

Named Pipes and the Central Print Processor

Central uses named pipes to achieve its task of providing an electronic template “back-end” to any application that can write an ASCII data file to the named pipe. In addition, Central includes a Central Print Processor, that enables applications to “print” requests through the Central Print Processor to Central. This takes advantage of the ability of the Microsoft® Windows® Print Manager to queue multiple print requests to a single queue, from which the Central Print Processor can write one request at a time to Central, via the named pipe. For details about the named pipe capabilities of Central, see “Accessing Central via Named Pipes” on page 39.

When a client application writes data to a print queue monitored by the Central Print Processor, it is automatically sent to the machine on the network where Central is running, through a named pipe. When writing data to a named pipe, you must supply the name of the pipe that Central uses. This name has the form:

```
\\machinename\pipe\jetform\centralname
```

When writing data to the print queue handled by the Central Print Processor, you use the print queue name. This is usually:

```
\\machinename\queuename
```

where *machinename* is the name of the machine on which Central and the Central Print Processor are installed. *queuename* is the share name of the printer that uses the Central Print Processor. For easy identification, the share name should be the same as the Central name. The queue must be shared to be accessible on the network. For details on installing the Central Print Processor, see the *Getting Started* guide.

A Walkthrough of the Central Process

When you initiate Central, it performs a number of activities before scanning the control and collector directories.

Central checks the options in the startup command. If the options include the **-aii***configfile* option, Central opens the configuration file (.ini) specified by *configfile*. Otherwise, Central finds the jfserver.ini configuration file, either in the directory location of the Central executable or in the same directory where Central runs, that is, the current working directory. In any event, Central loads all the default values from the .ini file. It then overrides the values with any options provided from the command line.

Central then makes a working copy of the Job Management Database. This enables you to make changes to the tables in the Job Management Database, while Central continues processing transaction files. Central is unaware of any changes to the Job Management Database, and processes transactions according to the instructions contained in its working copy of the Job Management Database. When you shut down Central and then start it again, or you click **Control** > **Reload Job Management DB** in Central Control, Central refreshes its working copy of the Job Management Database.

Central starts the Pipe Manager. The Pipe Manager receives transaction files sent through named pipes, as well as transaction files from the Central Print Processor, and writes them to the collector directory.

The Task Table contains control entries installed with Central for startup and shut down. If there is a task to execute at startup for the control entry “JFSTARTUP” from the Task Table, Central executes this task.

When the startup command includes the **-l** option, Central deletes and recreates the Macro Status Table files for all macros on the Job Table with a Load Flag of “P” (permanent). A Macro Status Table file exists for each printer. Central uses these files to keep track of template images (macros) in printer memory. Each Job Table entry causes Central to execute the task “JFDNLOAD” from the Task Table. This task launches Print Agent to download the permanent macros into the printer and recreate the Macro Status Table file for that printer. If the option specifies **-l**[*printername*], Central loads the macros for the specified printer only.

Central then scans the control directory for control files that affect its operation. Files are listed in the control directory in the order:

- stopserv.msg
- jf*.msg

When a file named stopserv.msg appears in the control directory, Central processes it first. Central executes the task associated with the control entry “JFSHUTDN” on the Task Table, then deletes stopserv.msg and shuts down both itself and the Pipe Manager.

Files with names matching the pattern jf*.msg are submitted to Central from the **Control** menu in Central Control. They contain control commands sent by the Central Control process to direct the activity of Central. Central Control places the files in the control directory and awakens Central, if it is sleeping, overriding the **Scan time** setting. A file with the pattern jf*.msg placed in the control directory by other means, such as with the Copy command, is the next task processed by Central. That is, it is processed after the processing of the current task, or, if Central is sleeping, after the **Scan time** setting has elapsed.

When there are no files to be processed in the control directory, Central scans the collector directory for transaction files. The collector directory is the **Transaction files** path specified at configuration. Central scans for transaction files with a file extension that matches the **File specification** value set at configuration. Starting Central with the **-d** command line option overrides both the collector directory path and the file specification.

Central builds a list of all files present in the collector directory at the time of the scan, and processes them one-by-one. Files are processed by the time/date stamp down to the second. When one or more files have the same time/date stamp, the files are processed alphanumerically. The collector directory is not re-scanned after processing of a file, only after all files in the list are processed. However, after each file in the collector directory list is processed, the control directory is re-scanned. Should a file appear in the control directory, the file list in the collector directory is dropped and processing moves to the control directory. Once all files in the control directory are processed, the collector directory is re-scanned and Central builds a new list of files for processing.

When there are no files found in the control or collector directories, Central sleeps for the period specified by the **Scan time** in Central Control. When Central awakens, it scans the directories again, and the cycle repeats. When the command line includes the **-e** option and there are no more files found in the control or collector directories, Central shuts down.

Log, Status, and Error Files

While running, Central maintains a log file, a status file, and an error directory:

- The log file contains entries for all the activities of Central while it is running.
- The status file contains a single entry for the most recent activity that Central has processed.
- When Central cannot complete the processing for a transaction file, it creates an error file containing the relevant log file entries. Central creates the error file in the error directory specified in its configuration file. The error file uses the same name as the transaction file, with a file extension of .err. In addition, Central moves the transaction file set to the error directory.

You can access these files from the **Display** menu in Central Control.

Processing Transaction Files

When Central encounters a transaction file in the collector directory, it reads the first line in the file, expecting a **^job** command. Central uses the information on the **^job** command to create a record complex for the transaction file. This means that Central links together the tables in the Job Management Database to create a record containing all the information from the tables for one step. Since the Job Table enables you to create more than one step for processing a transaction, there may be many record complexes created for one transaction file. Each record complex causes Central to initiate a subordinate task. Central processes the transaction file according to the record complexes it creates from the Job Management Database.

The Job, Printer, and Task Tables contain control entries installed with Central. These entries are defaults used when the information provided on the **^job** command is insufficient for Central to process the transaction file. In these circumstances, Central uses these control entries from the tables to create a single record complex and executes the associated task.

For the Job Management Database tables and their control entries, see the guide, *Central Control*. To access this guide, from Central Control, click **Help** > **View Documentation**.

The default control entries for processing transaction files include:

Job Table

DEFAULT Used as the job name when a **^job** command does not appear at the start of the transaction file.

Printer Table

DEFAULT Specifies the default device where all output prints when Central cannot determine the printer. Central uses DEFAULT as the Printer Name when a printer name is specified as "*" (any printers) on the Job Table, and either the **-z** option does not appear on the **^job** command or appears as **-z***.

Task Table

DEFAULT Used as the task identifier when Central cannot associate a task identifier with a job step and the **^job** command does not include the **-axx** option.

JFNOJOB Used as the task identifier when Central cannot find a **^job** command at the start of the transaction file.

JFDEFPARM Uses the Program options for this task identifier when a task identifier specified by the **-axx** option on the **^job** command is not in the Task Table.

There are additional default control entries in the Job Management Database tables; however, they do not directly affect the processing of the transaction files.

Missing ^job Command in Transaction File

When Central cannot find a **^job** command at the top of the transaction file, it initiates the task "JFNOJOB", which is one of the control entries from the Task Table.

The task associated with "JFNOJOB" analyzes the transaction file to determine an appropriate **^job** command, based on entries in a definition file, default.def. The Central Administrator creates and maintains this definition file. JFNOJOB matches an entry in the definition file to the transaction file, and uses a **^job** command specified in the definition file as the **^job** command for the transaction file. It creates a new transaction file containing the **^job** command as the first line in the file, followed by the contents of the original transaction file. JFNOJOB assigns a new name to the new transaction file, but uses the same file extension as the original transaction file. It writes the new file to the Central collector directory.

For details about the default.def definition file, see "JFNOJOB Processing" on page 54.

Note: The default behavior for JFNOJOB processing is for JFNOJOB to write the new file to the collector directory. This is governed by the -d@CollectorDir option in the **Program options** section of the JFNOJOB Task Table entry. To have JFNOJOB write the new file to the control directory, update the JFNOJOB Task Table entry to change -d@CollectorDir to -d@ControlDir.

Usually a transaction file can be accompanied by other related files, linked together by their file names. However, when a transaction file has no **^job** command, it must not be accompanied by related files, because before being processed it is copied into a new file with a different name.

Task Included on ^job Command

When the **^job** command specifies a task using the **-axx** option, Central performs a lookup on the Task Table using the argument from the **-axx** option as the task identifier. For example, a **^job** command that includes a task to print a purchase order may look like:

```
^job P_ORDER -zACCTING -axxJFMERGE
```

When Central finds an entry in the Task Table, it initiates the executable program and passes it any Program Options associated with the task identifier. Central uses the job name on the **^job** command as the template name.

However, Central may not find an entry in the Task Table for the argument from the **-axx** option, such as:

```
^job P_ORDER -zACCTING -axxXXX.EXE
```

In this case, Central initiates the executable program specified by the **-axx** option using the Program Options associated with the control entry “JFDEFPARM” from the Task Table.

When the printer name option on the **^job** command specifies -z* or when there is no printer specified, Central uses the default printer associated with the control entry “DEFAULT” on the Printer Table. Otherwise, Central determines the output device from the argument provided with the **-z** option. Either Central finds the printer name on the Printer Table and uses its physical device, or if not found, Central uses the *prtname* as the output device.

When Central initiates the process, it passes any options associated with the process and replaces any variable substitution options from the Printer Table and Task Table with the value for that variable.

Note that when the **^job** command specifies a task using the **-axx** option, Central does not perform a lookup on the Job Table.

Task Not Included on ^job Command

When the **^job** command does not include the **-axx** option, Central scans the Job Table for entries that match the job name and the **-z** option on the **^job** command, if provided. The **-z** option specifies a logical printer name; its argument may be a logical output device, or an “*”

(asterisk). Central treats a printer name of “*” as a match to any printer name on the Job Table that also matches on job name. If the **-z** option does not appear on the **^job** command, the default is **-z***.

Central creates the record complexes for one or more matches to the Job Table, linking the Job Table with the Printer Table and the Task Table for each step. Central then initiates the executable program associated for the first step. This can be an Agent or a custom application. Central completes all activities for one step before it proceeds with the task for the next step.

When Central does not find the task identifier from the Job Table on the Task Table, then Central executes the program associated with the control entry “DEFAULT”.

When Central initiates the process, it passes any options associated with the process and replaces any variable substitution options from the Printer Table and Task Table with the value for that variable.

When a **^job** command contains a job name that does not exist in the Job Table and there is no **-axx** option specifying a task identifier, then Central uses the job name as the template name and initiates the “DEFAULT” task from the Task Table.

When Transaction Processing Completes

After all the steps complete successfully, Central either deletes the transaction file, or moves it to the backup directory. It performs one of these actions based on the **Save backup files** option set at configuration. It deletes or moves other related files based on the **Delete extension** value specified at configuration.

If the steps for the transaction file do not complete successfully, Central moves the transaction file set to the error directory specified at configuration. Central then creates an error log file with the file extension .err, and copies in relevant log file entries from the Central log file. It then deletes the transaction file from the collector directory.

Macro Control

One of the powerful features of Central is its ability to manage printer resources. Print Agent keeps a Macro Status Table file for each printer. The file name assigned to the Macro Status Table file for a printer is the Printer Id from the Printer Table, with the file extension .mst, for example, prntr1.mst. Each file contains current information about the templates downloaded into the printer. The Macro Status Table file contains information for Central use only. When Central starts with the **-l** option, it deletes the Macro Status Table files for all printers. Central then determines whether the macros for one or all printers require downloading. It scans the Job Table for entries with a Load Flag of “P” (permanent) and with a Task id for Print Agent. It maps the printer name from the Job Table to the Printer Table to determine the physical device. Each entry found causes Central to execute a control entry “JFDNLOAD” from the Task Table. This task initiates Print Agent to download the macro into the printer and recreate the Macro Status Table file for that printer. For a download to occur, the entry “JFDNLOAD” on the Task Table must specify a task of Print Agent.

Central also causes the release of all managed memory. As well, it sets all templates designated as Load-First-Time, that is, with a Load Flag of “F”, to “not loaded” in the Macro Status Table files.

If a printer is powered off, it loses all permanent templates stored in its memory. Since Central has no means to detect this printer problem, the Central Administrator must realize that the permanent templates for that printer are lost. You may download the templates into the printer by using the clicking **Control** > **Reload Printer** in Central Control, or by restarting Central with the **-l** [*printername*] option.

For the Load Flags on the Job Table of “F” (load-first-time), “T” (temporary), and “M” (managed memory), the macro for the associated template is loaded into the printer when the template is used. As well, the Macro Status Table file is updated at this time for Load Flags of “F” and “M”. The Macro Status Table file does not track temporary macros.

When Central receives a request to download a macro into managed memory, Print Agent determines whether the macro is already downloaded. If the macro is not already downloaded, then Print Agent loads it if there is sufficient space in managed memory. If there is insufficient space, Print Agent erases other macros in managed memory, removing the fewest and smallest possible while still freeing enough space for the new macro. Print Agent uses the Macro Status Table file for the printer to maintain the status of the macros erased from managed memory, and those loaded into managed memory.

Central treats the macros for Load Flags on the Job Table of “R” (store in nonvolatile printer memory) and “H” (store in the hard disk of the printer) as permanent; however, Central never downloads the macros for these load flags. Loading templates into nonvolatile printer memory or the hard disk in the printer is a Print Agent administration function.

If you set up several printers in a printer pool, ensure that you designate all templates with a Load Flag of “T” (temporary). Because Windows controls printer resources, there is no way of controlling to which printer macros and data are sent.

Related Documentation

Getting Started provides an overview of Central and its Agents and components. It includes instructions on installing Central and Agents, and on configuring Central.

Central Control provides task oriented help for working with the tables of the Job Management Database, and for monitoring, controlling, and configuring Central. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

The Central documentation map provides links to the guides available for the Central Agents and components. You can access this documentation map by clicking **Start** > **Programs** > **Adobe Central Output Server 5.7** > **Documentation**.

About This Guide

As well as the new features listed in the *What's New* document, there are additional changes throughout the guide. These symbols draw your attention to "new" and "updated" topics.

NEW

New - indicates a new topic in this guide.

UPDATED

Updated - indicates an updated topic in this guide.

In addition to the symbol, updated text within the topic is marked with change bars, like the one to the left of this text.

2 Defining Central Tasks

Central uses a Job Management Database called jfserver.jmd, to manage and control the processing of transaction files. By default, when you install Central, the Job Management Database resides in the same directory as Central.

The Job Management Database consists of four tables. Using the tables, Central can process multiple tasks for a single transaction, such as transforming data, printing, and executing a custom application.

Two of the tables enable you to define the job steps and their associated tasks for processing a transaction file. The other tables define the printers in your organization, and enable you to specify the amount of managed memory for each printer that is available to Central.

The four tables are:

- Job Table
- Printer Table
- Task Table
- Managed Memory Table

While running, Central uses a working copy of the Job Management Database file, not the master Job Management Database. When you add entries or make changes to the tables, Central is unaware of them until you stop and restart Central, or reload the Job Management Database.

Central uses the four tables in the Job Management Database and the **^job** command from the transaction file to assemble all the specific details it needs to process the tasks for a transaction.

Under normal circumstances, the first line of any transaction file processed by Central is a **^job** command. This command identifies the job name and other options necessary for Central to process the transaction file.

The format for the **^job** command:

```
^job jobname [-zprtname] [-axxtaskid] [-ajijobid] [jobtokens]
```

where:

jobname is the name of the job as it appears in the Job Table of the Job Management Database.

-z*prtname* is the logical print device.

-axx*taskid* is the name of the task that Central executes exclusively.

`-aji`*`jobid`* (optional) is a user assigned identifier for the job.

`jobtokens` include command line options defined by the client application that are passed to the task that Central executes.

Normally, the first step in defining the tasks for a transaction file is to create the job name in the Job Table. When Central encounters the **^job** command in a transaction file, it uses the *`jobname`* and *`prtname`* and performs a lookup on the Job Table.

The Job Table entries define one or more steps that Central executes to complete the transaction. Each job step may map to an output device and a task, defined on the Printer Table and Task Table, respectively. In addition, a job step may contain a preamble file name, a template name, and related macro information. A job step may execute a task that creates a work file. A work file output from one job step may be the input file for another job step. The Job Table enables you to specify input and output file names for temporary work files for a job step. This table also enables you to specify whether Central should continue processing subsequent job steps when it encounters an error in the current step.

The logical output device name from the Job Table must also have a corresponding entry in the Printer Table. Central looks to the Printer Table to find the details about which device to use. For each output device, you add an entry in the Printer Table containing its logical name, its printer id, the physical device name, any command line options that Print Agent requires, and any comments.

The task identifier on a job step must have a corresponding entry in the Task Table. Each task entry in the Task Table must contain the name of the executable file to launch, the program options and any comments. Program options are command line options or substitution variables necessary to run the task; for example, command line options for Print Agent.

Finally, for each printer defined in the Printer Table that supports managed memory, you can create an entry in the Managed Memory Table that specifies the amount of managed memory Central controls.

MAINTAINING THE JOB MANAGEMENT DATABASE

To set up the tables in the Job Management Database for processing a transaction, enter the details using Central Control. The **Job Management Database** cascading menu from the **File** menu provides access to a series of windows and dialog boxes for capturing the tables information.

When you do not enter a value in an optional text box, Central defaults the value to an "*" (asterisk), rather than leaving the text box blank. When you finish modifying a table in the Job Management Database, save your changes.

Note: Central Control provides a facility to easily maintain the Job Management Database tables. However, you can also edit the tables directly in the Job Management Database file, **jfserver.jmd**, with a text editor.

When editing the **jfserver.jmd** file, keep these considerations in mind:

- The first line of the **jfserver.jmd** file contains the comment:

  ```
  # JetForm Job Management Database (JMD) file
  ```

  This line must not be changed. Without this line, Central will not load the Job Management Database.
- Central requires a value for each option. Where you do not need a value for an option, enter an “*” (asterisk) as a placeholder. If the option contains spaces or non-alphanumeric characters, surround the option with ““ ”” (double quotation marks).

Job Table

The Job Table contains an entry for the job name defined for each transaction. This table is the central management point for defining and maintaining the job steps for a transaction. Job steps tell Central what tasks to perform for a transaction file. Central matches the job name and printer name from the **^job** command in the transaction file to the entries in the Job Table. For each match found, Central processes the task associated with the job step. Steps are processed in the order shown in the Job Table. When Central finds no more matches in the Job Table, processing of the transaction is complete.

For complete details about editing the Job Table, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

Task Table

The Task Table defines the tasks that Central processes. Each entry in the Task Table defines an executable task, identified by its task identifier. Central launches a task from a job step defined in the Job Table or from the **-axx** option on the **^job** command.

For complete details about editing the Task Table, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

Printer Table

The Printer Table defines each logical output device available to Central. This table gives you the flexibility of changing the details for a printer in only one place.

For complete details about editing the Printer Table, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

Managed Memory Table

The Managed Memory Table specifies the amount of memory to manage for a specific printer. Managed memory is a block of printer memory that Central manages for storing permanent macros.

A Load Flag of “M” on the Job Table specifies the template is downloaded into managed memory. When Central gets a request to download a template into managed memory, it determines, through Print Agent, whether the template is already loaded. If the template is not already downloaded into managed memory, and if there is sufficient room in managed memory to load the template, then the template is downloaded. If there is insufficient space in managed memory, then Print Agent erases other macros in managed memory, removing the fewest number and smallest amount of space while still freeing up enough space for the new macro.

Print Agent maintains information about erased macros, and those currently loaded into managed memory. This information, together with other information about templates downloaded into the printer, is maintained in the Macro Status Table file for the printer. The name of the file is *printername*.mst, where *printername* is the **Printer id** from the Printer Table.

For complete details about editing the Managed Memory Table, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

3 Central Utilities

The Central Utilities allow you to monitor the operation of one or more instances of Central.You can install up to twenty (20) instances of Central on the same computer.

The Central Utilities commands are case sensitive. Command options in square brackets are optional. If the options in square brackets are separated by vertical bars, such as:

```
[item1|item2|item3]
```

use only one of the options within the vertical bars at any one time.

Note: For all Central Utilities commands, the default Central name is “server”. You can change the default by setting the variable JFSERVERNAME in your environment, or by using either the **-s** or the **-P** parameter on the command line.

JFC (Control Central)

The **jfc** command enables you to:

- Stop, start, pause and resume Central.
- Reload the Job Management Database.
- Download a macro to one or more printers.
- Display a help message detailing the syntax of the command.

Syntax:

```
jfc [-sCentral] [stop|start|pause|resume|reload|about]
```

- or -

```
jfc [-sCentral] download [-fMDFName] [-pPrinterName…]
```

- or -

```
jfc [-h|-'?']
```

where:

`-s`*`Central`* identifies which Central to control. *`Central`* is the short name of the Central. The default is **server**.

`stop|start|pause|resume` are the parameters to control the running of Central. Note that the Central Daemon must be running in order to start Central with the **jfc** command.

`reload` causes the **jfc** command to reload the Job Management Database.

`about` displays registration information about the installed Central products. This includes the product name, version number, user name, and Product Authorization Code (PAC).

`-f`*`MDFName`* restricts the download to just the one file and to only those printers that need it, as determined by the Job Table.

`-p`*`PrinterName`* restricts the download to the specified printers. The default is all printers.

`-h` and `-'?'` cause the **jfc** command to display a help message detailing the syntax of the command.

If Central is sleeping, the **jfc** command awakens it. However, if Central is awake but busy processing a job, Central will not process the command until it finishes the current job.

JFJMD (DISPLAY THE JOB MANAGEMENT DATABASE)

The **jfjmd** command enables you display the Job Management Database file.

Syntax

```
jfjmd [-sCentral] [-f|-j] [-p] [-x|-t] [-m] [-a] [-n] [File]
```

- or -

```
jfjmd -h
```

- or -

```
jfjmd -'?'
```

where:

`-s`*`Central`* identifies the Central for which you want to display the Job Management Database. *`Central`* is the short name of the Central. The default is **server**.

`-f` and `-j` mean display the job table (!f). Either option provides the same result.

`-p` means display the printer table (!p).

`-x` and `-t` mean display the task table (!x). Either option provides the same result.

`-m` means display the managed memory table (!m).

`-a` means display all tables. This is the default.

`-n` means display in field-nominated format.

`File` is the name of the Job Management Database file. If *`File`* is not given, the default is **jfserver.jmd**.

`-h` and `-'?'` cause the **jfjmd** command to display a help message detailing the syntax of the command.

The **jfjmd** command opens the Job Management Database before displaying it. If the file contains a fatal error, such as a duplicate printer name in the Printer Table, the file will not be displayed.

JFKICK (KICK CENTRAL)

The **jfkick** command enables you to kick or awaken Central to processes a transaction file. The **jfkick** command overrides the **PauseTime** set at configuration for Central. This command is useful if you have a program that places transaction files in the Central collector directory.

Syntax

```
jfkick [-sCentral]
```

- or -

```
jfkick -h
```

- or -

```
jfkick -'?'
```

where:

`-sCentral` identifies which Central to kick. *`Central`* is the short name of the Central. The default is **server**.

`-h` and `-'?'` cause the **jfkick** command to display a help message detailing the syntax of the command.

To use **jfkick**, the instance of Central must be running under the control of the Central Daemon. **jfkick** merely determines what node the instance of Central runs on and sends a message to the Central Daemon on that node. (It may be the same node where **jfkick** is running.) All instances of Central controlled by the Central Daemon are its own child processes. This makes it possible for the Central Daemon to wake them up by signalling them.

JFLISTEN (LISTEN TO A NAMED PIPE)

The **jflisten** command listens to a named pipe and puts data in the Central collector directory.

Syntax

```
jflisten [-sCentral] [PipeName]
```

where:

`-s`*`Central`* identifies the Central to which data is to be queued. *`Central`* is the short name of the Central. The default is **server**.

`PipeName` identifies a named pipe to which the **jflisten** command is to listen. The default is **stdin**. If *`PipeName`* is supplied, the named pipe must already exist. You can create it manually using the appropriate operating system utility, such as “mkfifo”.

jflisten is used to capture the output of programs that write to a named file. If a program writes to standard output, you can more conveniently capture its output with **jfp**.

A line containing “^end” and nothing else (case-insensitive) denotes the end of a particular file. The file is also taken to end when the talking process closes the pipe, that is, when **jflisten** reads EOF. Zero-length files are silently thrown away.

The file number of the generated file is jfl*nnnnn*.dat where *nnnnn* is an automatically generated number.

jflisten must be the only listener on the named pipe. There must be only one talker at a time on the named pipe.

Sending SIGTERM to **jflisten** causes it to carry out an orderly shutdown. It finishes processing the current job but exits at ^end or when the talker closes the pipe.

Upon successful start-up, **jflisten** writes a message to **stdout**, forks a child of itself into background and from the parent task, exits with a status of 0. If start-up is unsuccessful, it writes an error message to **stdout** and exits with a non-zero status.

Note: On a Network File System (NFS), named pipes do not work across the network. It is necessary to run **jflisten** on each node where a talker may run.

JFP (PUT A JOB IN CENTRAL QUEUE)

The **jfp** command puts a job in the Central collector directory.

Syntax

```
jfp [-sCentral] [FileName…]
```

- or -

```
jfp -h
```

- or -

```
jfp -'?'
```

where:

`-s`*`Central`* identifies the Central. *`Central`* is the short name of the Central. The default is **server**.

`-h` and `-'?'` cause the **jfp** command to display a help message detailing the syntax of the command.

`FileName` is the name of the file or files to be put in the collector directory. If not supplied, the **jfp** command reads from **stdin** until it encounters EOF. If more than one file is supplied, they are queued separately.

The new data file is created in the appropriate collector directory with the file name jfp*nnnnn*.dat. *nnnnn* is an automatically generated number. The number (not the whole file name) is also written in %05ld format to **stdout** as the job number.

The output file is created in the collector directory, but with the extension **.dxx**. After the file is copied and closed, it is renamed to **.dat**.

If it is desired to enforce security, the collector directory must have neither read nor write permission for group or world, and **jfp** must be set to the owner of the collector directory or root.

JFQ (DISPLAY CENTRAL QUEUE)

The **jfq** command displays the files waiting for processing in the Central collector directory.

Syntax

```
  jfq [-n] [+[SleepTime]] [-sCentral] [JobNumber…|FileName…]
```

- or -

```
  jfq -h
```

- or -

```
  jfq -'?'
```

where:

`-n` means display in field-nominated format.

`+` means continue displaying the queue until it empties.

`SleepTime`, if supplied, is the time in seconds to sleep before redisplaying the queue. *`SleepTime`* can only be given after `+`.

`-s`*`Central`* identifies the Central whose queue is to be displayed. *`Central`* is the short name of the Central. The default is **server**.

`JobNumber` is the job number as displayed on screen when using the **jfp** command. *`FileName`* is the base name of a file in the Central collector directory. If the first character in the parameter is a digit, it is treated as a job number; otherwise, a file name.

`JobNumber`, if supplied, limits the display to the given jobs. Multiple job numbers may be supplied.

`FileName`, if supplied, limits the display to the given files. Multiple file names may be supplied.

`-h` and `-'?'` cause the **jfq** command to display a help message detailing the syntax of the command.

The first line displayed by the **jfq** command gives the status of Central as indicted by the **jfserver.sts** file. Subsequent lines represent files in the collector directory, sorted in processing order. The display shows both message files (jf***.msg**) and data files.

JFRM (REMOVE A JOB FROM CENTRAL QUEUE)

The **jfrm** command removes a transaction file from the Central collector directory.

Syntax

```
jfrm [-sCentral] [JobNumber…|FileName…]
```

where:

`-s`*`Central`* identifies which Central queue to delete from. *`Central`* is the short name of the Central. The default is **server**.

`JobNumber` is the job number as displayed on screen when using the **jfp** command. *`FileName`* is the base name of a file in the Central collector directory. If the first character in the parameter is a digit, it is taken to be a job number, otherwise a file name.

At least one *`JobNumber`* or one *`FileName`* must be supplied, but you can supply any number of each. *`JobNumber`*, if supplied, removes the given jobs. *`FileName`*, if supplied, removes the given files.

Unlike the UNIX **lprm** command, the **jfrm** command cannot delete a job if it is currently being printed. The **jfrm** command sends a message file (jf***.msg**) to Central. If Central is processing a transaction file when it receives the jf***.msg** file, it finishes processing the transaction file before it reads the jf***.msg** file. If Central is sleeping, the **jfrm** command awakens Central and the jf***.msg** file is processed before any transaction files.

The **jfrm** command just posts the request; it does not wait for it to be processed. Use the **jfq** command to watch the queue and/or wait for the queue to empty.

Note: When Central is paused, it still processes jf***.msg** files. Therefore, a **jfrm** command can be used to remove jobs from the collector directory, even when Central is paused.

JFSTAT (DISPLAY CENTRAL STATUS)

The **jfstat** command displays the status of Central.

Syntax

```
jfstat [-sCentral|-a] [-l[Lines]] [-b] [-e] [-n]
```

where:

`-s`*`Central`*`|a` identifies the instance or instances of Central for which you want to display the status.

`Central` is the short name of the Central. The default is **server**.

`-a` means display the status for all instances of Central.

`-l` means display the end of the log file, showing the most recent messages written to the file. *`Lines`* identifies the number of lines to display. The default is 22 lines.

`-b` means display the backup directory.

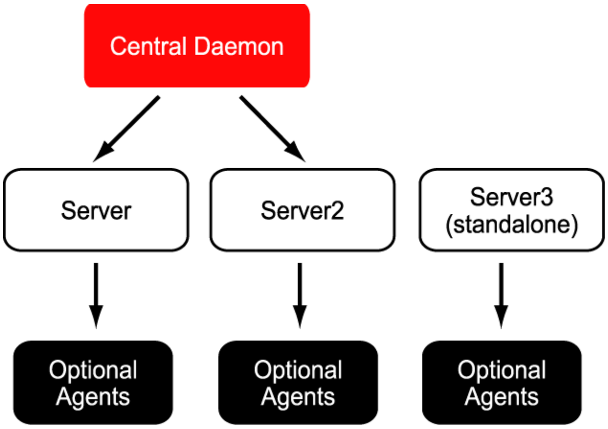
`-e` means display the error directory.

`-n` means display in field-nominated format.

4 Central Daemon

Central can be controlled across a network. The Central Utilities can submit messages (pause, resume, stop, download) and jobs to Central. This is true whether Central is on the same node or another node. In addition, when Central is running under the Central Daemon, the Central Utilities can also start or kick it across the network. Note that the Central Daemon must be running in order to start or kick instances of Central.

With UNIX, Central can be run either under the Central Daemon or without it, that is stand-alone. The same machine can even have a mix of instances of Central, some running under Central Daemon and some running stand-alone.



In the above example, Central **server** and Central **server2** can be kicked or started locally or across the network, but Central **server3** can only be started locally. As well, **server3** cannot be kicked, because this feature is not available for stand-alone installations. Whether a Central runs stand-alone or under the Central Daemon is determined by the **RemoteAccess** parameter in the **jfserver.ini** file. This parameter is set in accordance with your responses to questions during the installation.

The Central Daemon is intended to start-up when the computer boots and to stay running until the computer is shut down. By default, it listens to port 1706, officially assigned to it by the Internet Assigned Number Authority.

You are encouraged to add this entry to your **/etc/services** file:

```
jetform    1706/udp    # event notifications
jetform    1706/tcp    # data transfers
```

If you have already allocated port 1706 to some other service, you can override the number by changing the **/etc/services** entry shown above. However, if you do this, ensure you put the modified entry into the **/etc/services** file on every computer in your network that will be running the Central Daemon or the Central Utilities.

A Central configured to run under the Central Daemon may be set to start-up when the Central Daemon starts, or to wait until a start request is received by the Central Daemon. This is controlled by the **AutoStart** parameter in the **jfserver.ini** file, set in accordance with your response to a question during the installation process.

The Central Daemon ignores a request to start a Central that is already running, or to kick a Central that is not running, or to start or kick a Central that is not installed on its node. Central itself ignores any kicks that arrive while it is busy processing a job. There is no acknowledgment or response of any kind back from the Central Daemon to the process that made the request. It is strictly a one-way communication. Indeed, the process making the request does not have any way of knowing if the Central Daemon is running. If the computer, that is the one on which the Central Daemon is supposed to be running, is completely down or disconnected from the network, the process sending the request may hang. However, this is dependent both on the operating system and on the physical arrangement of the network (bridges, routers, etc.).

Messages logged by the Central Daemon are logged to the system event log.

Central Daemon Termination

Under UNIX, when the Central Daemon receives a termination signal (SIGTERM), it sends a SIGTERM to all instances of Central under its control. It then logs a message and exits. When an instance of Central receives the SIGTERM, it terminates the currently running Agents, if any, and does a fast shutdown. The transaction file currently being processed is left in the collector directory. Central reprocesses this transaction file, starting with the first job step, when it is restarted.

5 Running Central

Central runs in the background, scanning its control and collector directories and processing transaction files. Through Central Control, you can perform a number of control functions for Central and monitor its activity.

All of the commands that enable you to stop, start, pause, and resume Central appear on the **Control** menu. Central must be started in order to activate these commands. Once started, you may wish to stop Central only in the event of network maintenance or when you install an optional Agent. You may wish to pause Central due to problems with a printer or a network spooler. Pausing Central interrupts its scanning cycle; scanning for, and processing transaction files continues when you resume Central.

When you update the Job Management Database tables, you can refresh the working copy of the Job Management Database that Central is using by clicking the **Reload Job Management DB** command. In the event that a printer is switched off, you can refresh one or more printers with one or more macros by choosing the **Reload Printer** command.

While running, Central maintains log and status files. Errors can occur while processing transaction files, and Central captures information about the failed transactions. By analyzing the log and status files, you can resolve problems with transaction files created by client applications.

Status File

It is advisable to add a section to your startup routine to delete any Central status file (.sts) at boot time. If your system should crash and the .sts file remains behind, Central Control will not allow you to restart Central. Deleting the .sts file will resolve the problem.

As a further note, while checking for the presence of an .sts file, also check for the presence of a stopserv.msg file. If one exists, delete it as well. If not deleted and you restart Central, Central will process the stopserv.msg file and shut itself down again.

Starting Central

There are a number of ways to start Central:

- From the Services dialog box, as a Service
- From the command prompt
- From the **Start** menu

One of the activities Central performs when it starts is to execute the task associated with the control entry “JFSTARTUP” on the Task Table. You can create a custom application and associate it with this control entry if you have special startup requirements for Central.

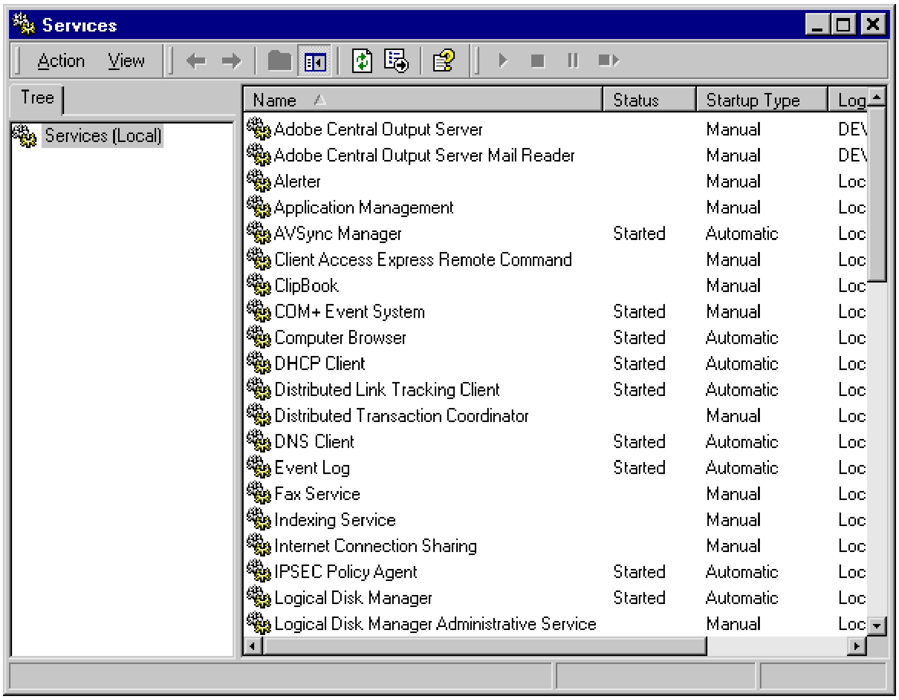
From the Services Dialog Box

UPDATED

Central is Service enabled, and can be started from the Services dialog box.

To start Central as a Service

1. On Windows 2000 and up, click **Start** > **Settings** > **Control Panel**, and click the **Administrative Tools** icon. Then click the **Services** icon.



Name	Status	Startup Type	Log
Adobe Central Output Server		Manual	DE\
Adobe Central Output Server Mail Reader		Manual	DE\
Alerter		Manual	Loc
Application Management		Manual	Loc
AVSync Manager	Started	Automatic	Loc
Client Access Express Remote Command		Manual	Loc
ClipBook		Manual	Loc
COM+ Event System	Started	Manual	Loc
Computer Browser	Started	Automatic	Loc
DHCP Client	Started	Automatic	Loc
Distributed Link Tracking Client	Started	Automatic	Loc
Distributed Transaction Coordinator		Manual	Loc
DNS Client	Started	Automatic	Loc
Event Log	Started	Automatic	Loc
Fax Service		Manual	Loc
Indexing Service		Manual	Loc
Internet Connection Sharing		Manual	Loc
IPSEC Policy Agent	Started	Automatic	Loc
Logical Disk Manager	Started	Automatic	Loc
Logical Disk Manager Administrative Service		Manual	Loc

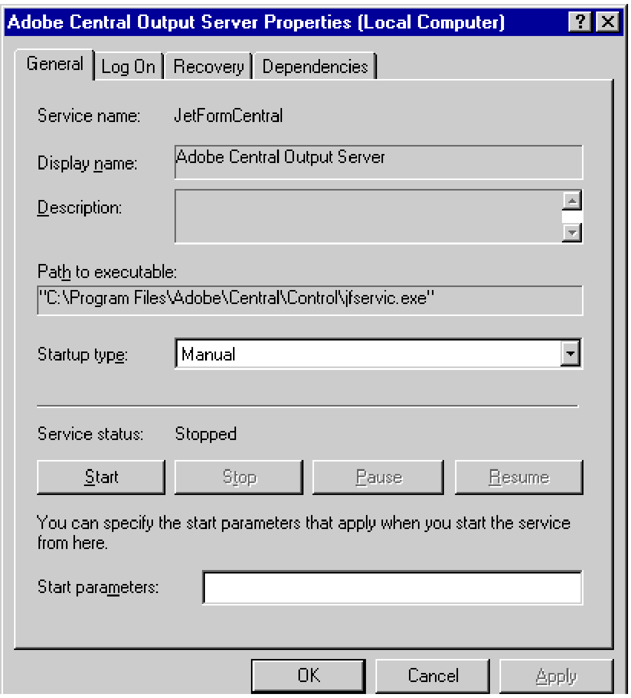
2. Click **Adobe Central Output Server**. Click **Action** > **Start**. Note that this action starts all of the currently configured instances of Central.

   Central instances started by the Adobe Central Output Server Service are always started with the **-aii** option specifying the default location for the configuration file. Assuming a default install location for the first instance, the instance will be started with the option -aii"C:\Program Files\Adobe\Central\Server\jfserver.ini".

   All instances of Central started by the Adobe Central Output Server Service will continue running, even if the user who started the Service logs off the system.

   The Adobe Central Output Server Service can be started or stopped from the Services dialog box. The **Pause** and **Continue** or **Resume** commands are grayed out. To pause and resume Central, do so from Central Control.

3. When you install Central, the default setting for startup is manual. To change this setting and other Service options, click **Action** > **Properties** in the Services dialog box.



Adobe Central Output Server Properties (Local Computer)

Service name: JetFormCentral

Display name: Adobe Central Output Server

Description:

Path to executable:
"C:\Program Files\Adobe\Central\Control\jfservic.exe"

Startup type: Manual

Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

Start parameters:

4. Here you can configure startup for a service:

   Automatic starts the service every time the system starts. However, the Service will not start automatically unless the machine has at least 12 MB of memory.

   Manual allows the service to be started by a user or by a dependent service.

   Disabled prevents the service from being started.

   You must specify the user account the service will use to log on. Go to the **Log On** tab. **System Account** specifies that the Service will log on using the system account. **This Account** specifies a user account the service will use to log on. If you select **This Account**, click the **Browse** button and the Select User dialog box on Windows. Type the password for the user account in both the **Password** and **Confirm Password** text boxes.

5. Click **OK** to accept the changes or **Cancel** to close the dialog box with making changes.

From the Command Prompt or Start Menu

You can start Central from a command prompt or from the **Start** menu. You use either of these methods if you wish to start Central with any of the command line options. Note, however, that a Central not started by the Adobe Central Output Server Service will terminate when you log off the system.

➤ **To start Central, either:**

| ● Click **Start** > **Programs** > **Accessories** > **Command Prompt** to open a command prompt and type:

- or -

● Click **Start** > **Run** and type:

the fully qualified executable program name for Central and any command line options. For example:

```
machinename\sharename\Adobe\Central\bin\jfserver.exe" -aiiconfigfile
[options]
```

`-aiiconfigfile` specifies the fully qualified file name of the .ini file to use when Central starts. It is a required value.

`options` refer to the command line options you can specify for Central, including:

`-all`*`logfile`* specifies the fully qualified path name of the .log file to use when Central starts. This option overrides the log file specified at configuration.

`-d`*`filespec`* or *`filespec`* option specifies the file specification that Central scans for in the collector directory. You may specify this option on the command line as **`-d`**`*.prn` or `*.prn`. The option overrides the **File specification** option set at Central configuration. This option may also specify a path for the Central collector directory, which overrides the **Transaction files** path set at configuration. Note that the file specification provided for this option must match the file specification provided for the **`-g`***`filespec`* option described below.

`-e` tells Central to shut down after it processes all messages and transaction files in the control and collector directories. Shut down occurs as soon as both directories are empty.

`-g`*`filespec`* specifies the file specification that Central searches for, to either delete or save transaction file sets. This option overrides the **Delete extension** option set at Central configuration. Note that the file specification provided for this option must match the file specification provided for the **`-d`***`filespec`* or *`filespec`* option.

`-h` prints a help message detailing the syntax of the startup command and describing the allowable command line options.

`-l``[`*`printername`*`]` requests the downloading of all templates in the Job Table with a load flag of “P” (permanent). *`printername`* is optional and identifies which logical device to reload. If the option does not include *`printername`*, all printers are reloaded. For example, **`-l`** reloads all printers, while **`-l`**`prntr1` reloads only the print device “prntr1”. Note, this is the letter “el”, not the number one.

`-p`*`nn`* specifies the pause time for Central as it scans its control and collector directories. You specify *`nn`* in seconds. The **`-p`***`nn`* option overrides the contents of the **Scan time** text box specified at configuration.

`-s` is the save transaction file option that causes Central to save the transaction file in the backup directory. By default, transaction files are deleted by a wildcard delete of the file name and first two characters of the file extension. For example, if the transaction file is *filename*.dat, then *filename*.da* is the file specification used in the delete command.

When you use the **`-s`** option, all files that match the file specification are moved to the backup directory after processing. Note that a transaction file in the backup directory is lost when another transaction file with the same name is moved into the directory. Specifying **`-s`** is the same as selecting the **Save backup files** check box at configuration in the Central Configuration dialog box.

`-v``nn` enables tracing of Central activity, which helps resolve problems. `nn` specifies the level of message detail provided:

Level	Description
-10	Trace - provides the maximum amount of detail available. Writes initialization parameters, task messages, progress messages, status messages and both warning and fatal error messages to the log file.
0	Complete information - writes all available information to the log file, except the initialization parameters and the progress messages. This is the default.
10	Information only - writes only the task messages and both warning and fatal error messages to the log file.
20	Errors - writes only the warning and fatal error messages to the log file.
30	Serious Errors - writes only the fatal error messages to the log file.
40	Banner Only - writes only the startup and shut down messages to the log file.

WRITING DATA STREAMS FROM USER APPLICATIONS

There are three methods an application may use to access the electronic template capability of Central:

- Writing a transaction file to a directory
- Writing a transaction file to a named pipe
- Writing a transaction file to a print queue configured to use the Central Print Processor

In all three scenarios, the transaction file is in the same format; namely it should begin with a **^job** command with a valid set of parameters. The format and options that may be included on the **^job** command are described in the next section, followed by sections describing the three methods that an application may use to access Central.

^job Command Format

Each transaction file sent to Central must contain a **^job** command as the first line in the file. The **^job** command has the format:

```
^job jobname [-zprtname] [-axxtaskid] [-ajijobid] [jobtokens]
```

where:

jobname is a logical name for the transaction, as known to the application or user and as it appears in the Job Table of the Job Management Database. *jobname* can be up to 51 characters in length and cannot contain blanks. This value is case-insensitive.

-z*prtname* (optional) identifies the logical printer. *prtname* is a logical printer name from the Printer Table of the Job Management Database, or an "*" (asterisk), as in **-z***. When the *prtname* does not exist on the Printer Table, Central assumes that *prtname* is a valid output location and attempts to write the output to that location.

-axx*taskid* (optional) identifies a task that Central executes exclusively. *taskid* appears in the Task Table of the Job Management Database. *taskid* can be up to nine characters in length and cannot contain blanks. This value is case-insensitive. If the **^job** command does not include this option, a task identifier must be present on the Job Table for the *jobname*.

-ajijobid (optional) is a user assigned identifier for the job.

jobtokens (optional) are other options provided to the task by Central. *jobtokens* may be any command line options supported by the executable program that Central launches, or the variable substitution arguments supported by Central.

Other characters may precede the **^job** command on the first line of the transaction file. Central ignores these characters. The characters may appear because the file transfer utility program that placed the transaction file in the collector directory inserted the characters at the beginning of the file. If a character other than the "^" (caret) precedes the word "job", then this is the character used as the prefix character for commands included in the transaction file.

Accessing Central via a Transaction File

UPDATED

Accessing Central via a transaction file is the standard method of accessing Central services. To use this method, the application creates a transaction file and writes it to the Central collector directory. It is the application's responsibility to ensure that it is not overwriting another transaction file by using a duplicate name. The file name can be any valid Windows file name, and it can contain spaces. The file name must have the same extension that Central is configured to scan for (.da? by default). However, we recommend that you create the transaction file in the Central collector directory with a file extension other than .dat. Once the file is written or copied to the collector directory, rename it with the .da? file extension.

Accessing Central via Named Pipes

Accessing Central via a named pipe is one of the more powerful capabilities provided by Central. It is the method used by the Central Print Processor. It is also available to any user or application. For installing the Central Print Processor, see the *Getting Started* guide.

To use the named pipe access method, you need to know:

- *machinename*, the name of the host machine on which Central is running.
- The pipe name, which is "jetform\centralname".

centralname is a unique logical name that identifies the instance of Central, assigned automatically as part of the installation process. The default is SERVER.

➤ To verify this name for an instance of Central

1. In Central Control, click **Configuration** > **Configure Central and Agents**.
2. Select **Central Control Process** from the **Available Processes** list box and click **Edit File**. The Central Control Configuration dialog box appears.
3. Select the appropriate Central and click **Edit**. The Edit Central Control Configuration dialog box appears. *centralname* is the name in the Central Name text box.

The pipe command is in the format:

```
\\machinename\pipe\jetform\centralname
```

or, if you are writing to a named pipe on your own machine:

```
\\.\pipe\jetform\centralname
```

where the "." (period) is a special notation indicating this machine.

To show how easy it is to use the named pipe access method, consider two examples. In the first, you want to copy a transaction file to Central, which is running on the same machine. To access Central with a `centralname` of SERVER, copy the transaction file to the named pipe:

```
copy xxx.yyy \\.\pipe\jetform\server
```

Note that the file name is not important, as the Pipe Manager, the part of Central that services the data pipes, generates a unique spool file name at the other end of the pipe. The file name it creates has the format:

```
SPnnnnnn.xxx
```

where `nnnnnn` is a 6-digit string that the Pipe Manager guarantees is unique and `xxx` is the file extension that Central is looking for.

As a second example, you want to copy a transaction file to Central, which is running on a different machine on the network. If the name of that machine is SERVER1, you provide:

```
copy xxx.yyy \\server1\pipe\jetform\server
```

The only difference in the second example is that the pipe name is prefixed with the name of the machine.

For user-written applications, all the application needs to do is to open a file for output with the appropriate name. As an example, to access Central where the *centralname* is SERVER on machine SERVER1, a C language program would do the following:

```
FILE *fpServer;
if (NULL == (fpServer = fopen("\\\\server1\\pipe\\jetform\\server",
"w")))
{
   // Error processing here - perhaps Pipe Manager
   // or Central not running
   :
}
else
{
   // Write data to fpServer
   :
}
```

In the C language the “\” (backslash) is an escape character. To get a backslash character, as required here, use “\\” (two backslashes).

Whenever the Pipe Manager places a file in the Central collector directory, it signals Central that there is a transaction waiting for processing. If Central is sleeping, it awakens immediately and processes all the transactions in the collector directory, in the standard priority order.

Accessing Central via Printer Queues

Another feature of Central is the ability to receive data via appropriately configured printer queues. To do this, the Central Print Processor must be installed and a queue configured. (For installing the Central Print Processor, see the *Getting Started* guide.) All transaction files (starting with **^job**) sent to this queue are written to:

```
\\.\pipe\jetform\queuename
```

queuename must be the same as the Central name. The default is SERVER.

As an example of receiving data via a printer queue, these conditions exist:

- SERVER is the Central name.
- The printer name is the same as the Central name, SERVER.
- The print queue is defined on the same machine on which Central is installed. It is shared, to be accessible on the network.

To access Central, copy (or print) a transaction file, beginning with a **^job** command, to the print queue, for example \\server1\server. The Print Manager invokes the Central Print Processor, which attempts to write the contents of the spool file to \\.\pipe\jetform\server. If a problem exists and the transaction file cannot be written, the Central Print Processor reports the error to the Event Log and displays a Retry/Cancel dialog on the console.

PAUSING CENTRAL

While Central is running, you can interrupt its operation, rather than shutting it down completely.

Ensure that the name of the Central you wish to pause shows in the Title Bar of the Central Control window. If it is not, click **File** > **Select Instance of Central** to select another instance of Central. Then click **Control** > **Pause Central**.

When you pause Central, Central Control creates and places a message file in the control directory. The pattern for the file name is jf*.msg. Central Control awakens Central, if it is sleeping. Central immediately scans the control directory and picks up the message file containing the control command. The control command pauses Central.

While Central is paused, it continues to scan the control directory, processing .msg files. Central ignores all transaction files in the collector directory while it is paused.

RESUMING CENTRAL

When you have paused Central, click **Control** > **Resume Central** to restart it.

Ensure that the name of the instance of Central you wish to resume shows in the Title Bar of the Central Control window. If it is not, click **File** > **Select Instance of Central** to select another instance of Central. Then click **Control** > **Resume Central**.

Resume Central causes Central to start scanning its control and collector directories for files to process.

STOPPING CENTRAL

There are a number of ways to stop Central:

- From the Services dialog box
- From Central Control
- From the command prompt

When you stop Central, a message file, stopserv.msg, is written to the Central control directory. If the file is placed there as a result of the Adobe Central Output Server Service being shut down, or by Central Control, Central immediately scans the control directory and encounters the stopserv.msg file. If the file is placed in the control directory using the copy command, or other means, it is the next file processed by Central. That is, it is processed after the processing of the current file, or, if Central is sleeping, after the **Scan time** setting has elapsed.

The stopserv.msg file carries a higher priority than other message files, and processes it first. Central recognizes the file name stopserv.msg as its signal to shut down. If there is an executable program associated with the “JFSHUTDN” from the Task Table, Central launches this task, and then shuts down.

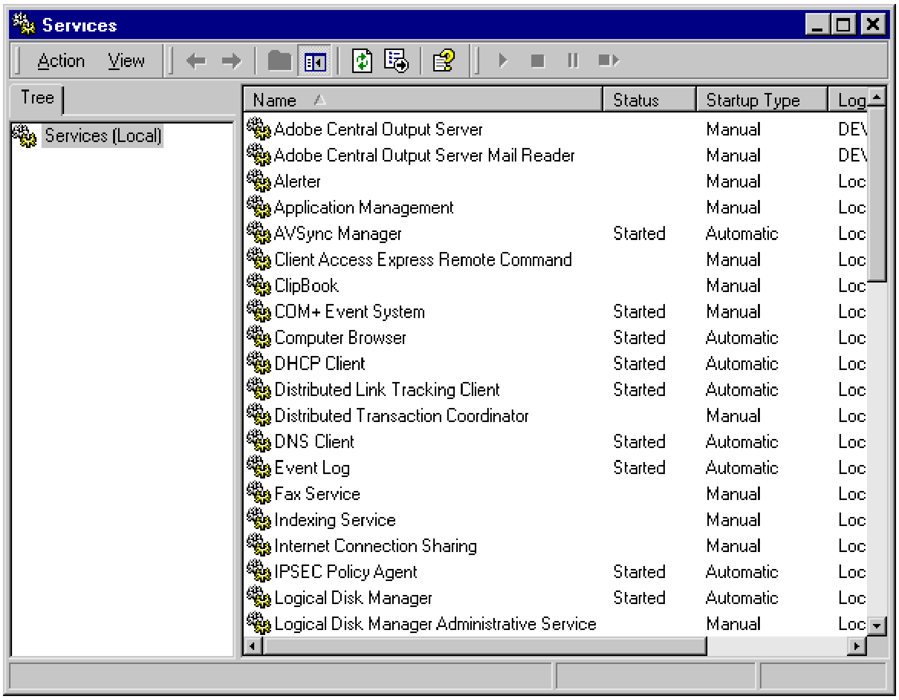
From the Services Dialog Box

UPDATED

Central is Service-enabled and can be stopped as a Service in the Services dialog box.

To stop Central from the Services dialog box

1. On Windows 2000 and up, click **Start** > **Settings** > **Control Panel**, and click the **Administrative Tools** icon. Then click the **Services** icon.



Name	Status	Startup Type	Log
Adobe Central Output Server		Manual	DEV
Adobe Central Output Server Mail Reader		Manual	DEV
Alerter		Manual	Loc
Application Management		Manual	Loc
AVSync Manager	Started	Automatic	Loc
Client Access Express Remote Command		Manual	Loc
ClipBook		Manual	Loc
COM+ Event System	Started	Manual	Loc
Computer Browser	Started	Automatic	Loc
DHCP Client	Started	Automatic	Loc
Distributed Link Tracking Client	Started	Automatic	Loc
Distributed Transaction Coordinator		Manual	Loc
DNS Client	Started	Automatic	Loc
Event Log	Started	Automatic	Loc
Fax Service		Manual	Loc
Indexing Service		Manual	Loc
Internet Connection Sharing		Manual	Loc
IPSEC Policy Agent	Started	Automatic	Loc
Logical Disk Manager	Started	Automatic	Loc
Logical Disk Manager Administrative Service		Manual	Loc

2. Click **Adobe Central Output Server**. Click **Action** > **Stop**. This action stops all the currently configured instances of Central.

From Central Control

To shut down Central from Central Control, ensure that the name of the Central you wish to shut down shows in the Title Bar of the Central Control window. If it is not, click **File** > **Select Instance of Central** to select another instance of Central. Then click **Control** > **Shut Down Central**.

From the Command Prompt

If you start Central from the command prompt, the preferred shut down method is to place a stopserv.msg file in the control directory. However, you can also use CTRL+C at the command prompt to shut down Central.

CENTRAL SECURITY

The [JetForm] section of the Central configuration file, jfserver.ini, contains a parameter called “SecurityLevel”. If SecurityLevel is zero (the default), the **-axx** option can specify any arbitrary command and the **-z** option can specify any arbitrary file name or pipe command. Setting SecurityLevel to any positive value restricts the **-axx** parameters to **Task ids** in the Task Table, and restricts the **-z** parameters to **Printer ids** in the Printer Table. If a submitted job does not conform to these restrictions, then Central rejects the job and logs a security message. If saving transaction files to a backup directory after processing is enabled (BackupFiles=Yes in jfserver.ini), then the transaction file for the rejected job is moved to the backup directory.

Note: Having the **-z** parameter match with or originate from the **Printer name** column of the Job Table is not sufficient. The **-z** parameters are restricted to **Printer ids** in the Printer Table.

RELOAD JOB MANAGEMENT DATABASE

Central runs with a working copy of the Job Management Database. When you update the Job Management Database tables, you do not affect the operation of Central. Central does not recognize the changes you make until you stop and restart Central, or until you click **Control** > **Reload Job Management DB**.

The **Reload Job Management DB** command creates a message file, with the pattern jf*.msg, that is written to the Central control directory. When Central scans the control directory, it picks up the file containing the control command. The **Reload Job Management DB** command causes Central to replace its working copy of the Job Management Database with the updated file.

RELOAD PRINTER MACROS

The **Reload Printer** command on the **Control** menu causes Central to launch the “JFREFRESH” task, which reloads the macros stored in a printer's memory.

Print Agent recognizes when a macro stored in printer memory does not match its template. Print Agent reloads the macro automatically the next time it uses it. You can use the **Reload Printer** command to update the printer immediately, if you wish. When a printer supports both

PostScript® and PCL printing, use the **Reload Printer** command to reload the macros when you switch the printer between modes. When you shut off a printer, you lose the macros stored in its memory.

When you reload the Job Management Database, Central Control creates a message file and places it in the Central control directory. The pattern for the file name is jf*.msg. When Central scans the control directory, it picks up the message file containing the control command. The control command reloads the specified printers. If Central is processing another file, the control command is processed after the current file completes.

For complete details about the **Reload Printer** command, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

MONITORING CENTRAL

Central operates in the background on a network server. Many of the operations that Central performs are transparent to you as a user. The activity of Central and its Agents is captured in log and status files. These files provide historical and current information about the activity of Central and the Agents. You can view the log and status files from the **Display** menu in Central Control.

Through Central Control, you can monitor all transaction files in the Central collector directory. In addition, you can view .err files containing the log entries for rejected transaction files.

Central Status and Queue

The **Central Status and Queue** command lists, for each instance of Central, its status and the files waiting for processing.

For complete details about the **Central Status and Queue** command, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

Central Log

The **Central Log** command enables you to view the history of the activity of Central. As Central processes transaction files, it records information, status, and error messages in the log file. The default configuration records all log messages in the jfserver.log file, including those of any installed Agents.

Central maintains the log file size based on a value set in the configuration file. Central keeps one generation of the log for historical purposes. When, at the end of a transaction, the log file has grown in size to meet or exceed the **Size** setting in the **Log file** section in the configuration file, Central deletes the older generation log file, renames the current log file to a file extension of .bak and starts a new log file.

For complete details about the **Central Log** command, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

Central Rejects

The **Central Rejects** command enables you to view the list of all files in the Central error directory.

Central collects all transaction files that it was not able to process in an error directory. These are the rejected transaction files. The rejection of a transaction file may occur because its **^job** command was incorrect, because Central could not find the job name in the Job Management Database, or because a job step returned an unsuccessful return code.

When a transaction file rejects, Central moves the transaction file set to the error directory specified in the configuration file, and creates a file with the same file name and the file extension .err. Central writes the relevant log file entries for this transaction from the log file into the .err file. Note that a transaction file in the error directory is lost when another transaction file with the same name is moved into the directory.

For complete details about the **Central Rejects** command, see the guide, *Central Control*. You can access this guide from Central Control by clicking **Help** > **View Documentation**.

Resolving Problems

There are many causes for a transaction file to fail, such as:

- Incorrect commands or invalid data in a transaction file from a client application.
- Incorrect or missing table entries in the Job Management Database.
- Unsuccessful return code from an Agent or custom application.
- Invalid path specifications.
- Missing files, such as templates.
- Running out of resources, such as memory.

When an error occurs while processing a job step for a transaction file, Central determines whether to continue processing subsequent job steps based on the **On error** option from the Job Table associated with the failed job step. With the flag set to “C”, Central continues processing; with the flag set to “S”, Central stops processing the transaction file. In either case, after the transaction file processing completes, Central moves the transaction file to the error directory and creates a file with a file extension **.err** containing the relevant log file entries. Central then invokes the control task “JFERROR”.

Your organization may decide on a routine for handling transaction files that fail to process successfully. When Central is installed, there is no executable program name associated with the control entry “JFERROR”. When you create a custom error handling application, update the

Program name and **Program options** for the control entry “JFERROR” in the Task Table. An example of this error handling could include a notification message sent to the creator of the transaction file that failed.

Central provides several tools you can use to resolve problems.

Log File

Your principal source of information in resolving any problems that occur when running Central is its log file. This file records each activity of Central and its Agents. By analyzing the contents of the log file, you can reconstruct the sequence of events performed before and after the problem occurred.

The log file entries have the format:

```
DATE TIME MODULE: MESSAGE
```

where `DATE` has the format YYYYMMDD, `TIME` has the format HH:MM:SS, `MODULE` is the name of the executable program that generated the message and `MESSAGE` is the text of the log message.

Note: The Central log file is constantly written to during Central operation. Locking the log file while Central is running causes entries to be lost.

You can set the level of message detail that is written to the log file in two ways:

- By setting the message display level value when configuring Central.
- By using the **-v*nn*** option with the command used to start Central from the **Start** menu. This option overrides the value set in the configuration file.

If desired, you can also change the format of the `DATE` and `TIME` settings. You do this in the jfserver.ini configuration file, in the [JetForm] section under the setting **MsgDateStampFormat**.

The possible values for the settings are any valid XPG format specifiers, but the following are the most useful:

Specifier	Description
%y	Year without century (two digit)
%Y	Year with century (4 digit)
%m	Month as a number
%d	Day of month as a number
%H	Hour in 24 hour format
%I	Hour in 12 hour format

Specifier	Description
%M	Minute as a number
%S	Second as a number
%p	AM/PM indicator (for 12 hour clock)

The default specifier, “%Y/%m/%d %H:%M:%S”, provides a date in the format, YYYY/MM/DD HH:MM:SS:

```
2000/01/31 15:47:23
```

For a two digit year, use a specifier of “%y/%m/%d %H:%M:%S”. This provides the date in the format YY/MM/DD HH:MM:SS:

```
00/01/31 15:47:23
```

The use of “/” (slash marks), “:” (colons) and spaces in the specifier are for ease of readability only. For example, using a specifier of “%Y%m%d%I%M%S%p” provides the date in the format YYYYMMDDHHMMSSXM:

```
20000131034723PM
```

whereas, using a specifier of “%Y/%m/%d/ %I:%M:%S:%p” provides the date in the format YYYY/MM/DD HH:MM:SS XM:

```
2000/01/31 03:47:23 PM
```

Note: The default configuration for Central and components (Print Agent, Transformation Agent, etc.) is to write to the same log file, jfserver.log. To ensure that they all write to the log file in the same manner, it is recommended that the **MsgDateStampFormat** setting in the configuration file for each component be the same.

Viewing Error Files

You can examine the log file entries or the transaction file for a failed transaction saved in the error directory. Click **Display** > **Central Rejects**.

This command lists all the files contained in the error directory. You can select a file from the list box, either the log file with the .err extension, or the transaction file itself. Click the **View File** button to open and view the file.

Examination of the log messages contained in the .err file tells you why the job failed. You can also view the transaction file to determine where to make a correction.

Saving Transaction Files

By selecting the **Save backup files** option when configuring Central, you can save a backup of your transaction files that have successfully completed. You can use these transaction files for comparison purposes with similar transactions that failed.

The files retain their original file extensions. The complete family of files is moved to the backup directory. Note that a transaction file in the backup directory is lost when another transaction file with the same name is moved into the directory.

When you do not select the **Save backup files** option, Central deletes the transaction file when processing finishes.

6 Variable Substitution

A variable is a value that changes from one transaction to the next, for example, job name, printer name or template name. In Central processing, a substitution variable acts as a placeholder for a variable. You can place a Central substitution variable on the command line passed to the executable program for a task. Before launching the program, Central replaces the substitution variable with the actual value of the variable.

An "@" (at) prefixes the Central substitution variables. You must terminate substitution variables with a "." (period) or spaces. The format for these variables is:

```
@VariableName.
```

where `VariableName` is a reserved name whose value changes as Central processes each task.

The entire construct must be enclosed in **" "** (quotation marks) if the value generated contains spaces within its parameters. For example, if:

```
-z@PhysicalDev.
```

generates a value of:

```
-zHP LaserJet 5Si
```

the program executing the task would process the string as three separate options. To identify the string as a single option, define the variable as:

```
-z"@PhysicalDev."
```

which Central passes to the executing program as:

```
-z"HP LaserJet 5Si"
```

You can use substitution variables:

- In the **Print Agent options** on the Printer Table. These are passed to Print Agent, and contain information specific to the printer.
- In the **Program options** on the Task Table. These are passed to the executable program that Central initiates for the task. If @OtherJobTokens is one of the substitution variables, then Central passes any *jobtokens* that appear on the **^job** command to the executable program.
- In the *jobtokens* option on the **^job** command. These are passed to the executable program Central initiates when the Program Options include @OtherJobTokens.

In processing a transaction file, Central uses the **^job** command and builds the task information from the entries in the Job Management Database. When Central finds an "@" (at), it retrieves the stored value for the substitution variable from memory and replaces the value for the *@VariableName* in the option list.

The Central substitution variables are:

Variable Name	Value Substituted
@ControlDir	Used with the **-d** option of JFNOJOB, as in -d@ControlDir. This option is used to determine the directory to which JFNOJOB will place a file during processing. ControlDir specifies the path name of the Central control directory.
@CollectorDir	Used with the **-d** option of JFNOJOB, as in -d@CollectorDir. This option is used to determine the directory to which JFNOJOB will place a file during processing. CollectorDir specifies the path name of the Central collector directory.
@DataFile	Fully qualified name of the transaction file.
@ErrjobName	Used with the JFERROR task. This option can be used to pass the name of the job that caused JFERROR to be invoked. To use this variable enter a custom name under **Program name** and supply the variable @ErrJobName under **Programs options**.
@GetProfile(“Name”[,”Section”,[“File”]])	A value from an .ini file. “Name” is the variable name from the .ini file whose value is retrieved and substituted. If “File” is specified, both “Section” and “File” must be specified. The default section is “User”. The default file is the Print Agent .ini file.
@InFile	Name of the file that is input to the current job step, from the Job Table. If the Job Table does not contain a file name, the default for @InFile is the transaction file name.
@InFileBase	The base of the current input file, to be used in constructing the name of the output file. For example, if the current input file is C:\Program Files\Adobe\Central\server\data\myfile.dat, the base is myfile.
@InFileExt	The extension of the current input file, to be used in constructing the name of the output file. For example, if the current input file is C:\Program Files\Adobe\Central\server\data\myfile.dat, the extension is .dat.
@InFilePath	The path of the current input file, to be used in constructing the name of the output file. For example, if the current input file is C:\Program Files\Adobe\Central\server\data\myfile.dat, the path is C:\Program Files\Adobe\Central\server\data.

NEW

The three @InFile components, @InFileBase, @InFileExt, and @InFilePath, allow the creation of an output file name containing elements from the input file name. For example, to generate an output file whose name contains the base of the input file (that is, it differs in extension and path), use this **Program option**:

```
-z"C:\Program Files\Adobe\Central\Server2\data\@InFileBase..xyz"
```

The @InFile components can be useful in cases where you have two or more instances of Central which scan for different file extensions. The components provide a mechanism for moving files between instances of Central, while still maintaining elements of the file name.

Variable Name	Value Substituted
@IniFileName	File name of the .ini file. The Central Agents require this value. It is included, where required, in the **Program options** on the Task Table, as part of the installation process.
@JobID	User assigned identifier for a job. When the **^job** command includes the option **-aji***jobid* and the **Program options** on the Task Table include the variable @JobID, Central passes the job identifier to the task.
@JobName	Job name specified on the **^job** command.
@LoadFlag	Type of macro for the current job step, from the Job Table.
@LogFileName	File name of the log file. The Central Agents require this value. It is included, where required, in the **Program options** on the Task Table, as part of the installation process.
@Macro#	Macro number for the job step, from the Job Table.
@ManagedMem	Amount of managed memory defined for the current printer, on the Managed Memory Table.
@MDFName	Either, the name of the template used by the current job step, from the Job Table, or if a template name is not supplied, the default template name is the job name with an .mdf file extension.

Note: When a **^form** command appears in the transaction file, the @MDFName option is not required. A **^form** command in the transaction file takes precedence over a template name from either the job step or the **^job** command.

Variable Name	Value Substituted
@MSTName	Macro Status Table file name, from the Printer Table.
@NumberPages	The number of surfaces printed for a job. Print Agent passes this value to Central. This makes it possible for a job step using Print Agent to pass the number of pages to a subsequent job step. Print Agent can then indicate the total page number, as well as the current page number, on each page of a printed document. For example, page 3 of 6. When Central starts a new job, it resets NumberPages to zero. For an example of using the NumberPages variable, see the sample “Field Overflow”, in the file *Print_Samples*. *Print_Samples* is available in either .pdf or .txt format and is located in the \Samples\exprint directory under the directory in which you installed Central.

Variable Name	Value Substituted
@OtherJobTokens	Values for *jobtokens* taken from the **^job** command after its substitution variables are satisfied.
@OutFile	Temporary file name of the file output from the current job step.
@OutputFileName	OutputFileName is set by any job step that invokes Print Agent to write to an output file. It is particularly useful when Print Agent has itself generated the file name in response to the `,u` argument of the **-z** option. When Central starts a new job, it resets OutputFileName to an empty string (“”).
@PhysicalDev	From the **-z** option of the **^job** command. Before substituting a value for @PhysicalDev, Central matches the job name and the printer name from the **-z** option on the **^job** command to the Job Table. For each match, Central substitutes the value found for **Physical Device** on the Printer Table for @PhysicalDev. When a match is not made, Central substitutes the value from the string following the **-z** option for @PhysicalDev.
@PreambleName	Preamble file name from the job step, on the Job Table.
@PrefixChar	Character preceding “job” on the **^job** command.
@PrintDirectorParms	Taken from the Printer Table, after its substitution variables are satisfied.
@PrinterID	From the Printer Table.
@PrinterName	Name of the printer from the Printer Table.
@ServerName	Used with the **-asn** option, as in -asn@ServerName. Central substitutes its name, from its .ini file, for @ServerName. The option is passed with the appropriate value, such as **-asnserver**.
@SkipLines	Used with the **-asl** option, as in, -asl@SkipLines. If the transaction file contains a **^job** command on its first line, Central substitutes a “1” for @SkipLines; otherwise it substitutes a “0”. The option is passed with the appropriate value, as **-asl1** or **-asl0**. The **-asl1** option instructs the Agent completing the task to ignore the first line of the transaction file. The **-asl0** option instructs the Agent completing the task to process the first line of the transaction file.
@TaskID	From the job step on the Job Table.
@TimeoutValue	Used with the **-adt** option, as in -adt@TimeoutValue. This is the time, in seconds, for the Agent to wait, if idle, before shutting itself down. The default is 60 seconds. Central substitutes the TimeoutValue from its .ini file for @TimeoutValue. The option is passed with the appropriate value, such as **-adt60**.

Variable Name	Value Substituted
@XMLData	Used with the **-axml** option, as in -axml@XMLData. If Central detects an XML job card in the transaction file, it replaces @XMLData with the value "yes". This causes Print Agent to process the file as XML. But if Central does not detect an XML job card in the transaction file, it replaces @XMLData with the null string (""). This tells Print Agent to make its own determination whether or not the file is XML, controlled by the ProcessAsXML option in its configuration, **jfmerge.in**i.

Caution: If you use an older task table entry to spawn Print Agent 5.6 under Central 5.6, it is possible for Central to recognize and process an XML job card but for Print Agent to treat the data file as field-nominated. You should add **-axml@XMLData** to your old Print Agent task table entries to ensure that this cannot happen. This parameter is ignored by older versions of Print Agent so you can use the same task table for older versions of Print Agent and for Print Agent 5.6.

7 JFNOJOB Processing

When Central processes a transaction file, it looks for a **^job** command on its first line. The job name from the **^job** command links to the tables in the Job Management Database, determining the processing of the transaction file. However, there may be cases where it is impossible for the user to include the **^job** command in the transaction file. In these cases, the user can include the **^job** command in an external file.

This external file, created by the Central administrator, is called a job definition file. It is a simple text file with a file name of default.def, that resides in the server\ directory. It can include any number of **^job** commands, one for each of the transaction file types that a user submits to Central without an embedded **^job** command. For example, a user may submit three different types of transactions to Central: invoices, balance sheets, and expense statements. The job definition file should include **^job** commands for use with each type of transaction.

The job definition file identifies one or more search strings and the range of record numbers where the search string may appear within the transaction file. Associated with each search string is the **^job** command that Central will use in processing the transaction file.

When Central receives a transaction file without a **^job** command, it initiates the job “DEFAULT” from the Job Table. The task id for DEFAULT is JFNOJOB. Central uses the **Program options** from the Task Table entry JFNOJOB to construct the command line to call JFNOJOB. The **Program options** are:

@InFile — The transaction file for which Central cannot determine a **^job** command. Central substitutes the transaction file name for @InFile.

*pathname*default.def — The fully qualified name of the default.def job definition file. For example ”C:\Program Files\Adobe\Central\Server\default.def”. The pathing information is prefixed to default.def at installation.

-d@CollectorDir — This option is included for compatibility with previous versions of Central. It identifies the output directory to which JFNOJOB writes new transaction files. The option can direct files to either the collector directory or the control directory. Using the collector directory causes the resultant file to be placed at the end of the processing queue. Using the control directory causes the resultant file to be processed next.

-d@CollectorDir (Continued)

When the option reads -d@CollectorDir., Central substitutes the location of the collector directory for @CollectorDir and passes this command line option to JFNOJOB:

```
-d"C:\Program Files\Adobe\Central\Server\Data"
```

When the option reads -d@ControlDir., Central substitutes the location of the control directory for @ControlDir and passes this command line option to JFNOJOB:

```
-d"C:\Program Files\Adobe\Central\Server\Control"
```

-x

Tells JFNOJOB to identify the **^job** command in its response file, rather than generating a new transaction file with the **^job** command as its first line.

-aii@IniFileName

The fully qualified name of the JFNOJOB configuration file, jfnojob.ini. Central substitutes the fully qualified name for @IniFileName and then passes this command line option to JFNOJOB:

```
-aii"C:\Program Files\Adobe\Central\Server\jfnojob.ini"
```

JFNOJOB analyzes the transaction file to determine an appropriate **^job** command, based on entries in the job definition file. Once determined, JFNOJOB writes the **^job** command to its response file. Central retrieves the **^job** command from the response file, then continues with normal processing by linking the job name from the **^job** command to the tables in the Job Management Database.

In the event that JFNOJOB can not find a match in the job definition file, it completes processing and writes an “unsuccessful” return code to its response file. Central performs normal error processing; it writes the transaction file to the error directory, and writes any relevant log file entries to a file with the same file name as the transaction file and a file extension of .err. It also writes this file to the error directory.

Creating the Job Definition File

The format of the default.def job definition file is a single line for each transaction type, that includes

Name	Format	Description
Starting Record Number	9(3)	Range of records in the transaction file where the Search String is valid.
Ending Record Number	9(3)	
Search String	x(nn)	Search string to scan for in the transaction file.
^job Command	x(nn)	**^job** command for the specified search string.

For example, in the job definition file ...

```
001 015 "Expense Statement" "^job EXP_STMT"
001 005 "Invoice" "^job INVOICE"
```

... the **^job** command determined for the transaction is either `^job EXP_STMT`, if the text string `Expense Statement` is encountered in the first fifteen lines of the transaction file, or `^job INVOICE`, if the text string `Invoice` is encountered in the first five lines of the transaction file.

When creating the job definition file, note that:

- The Starting and Ending Record Numbers define an inclusive range of records in the transaction file. That is, the range is up to and including the Ending Record Number. Record numbering in the transaction file starts at 1.
- Search Strings and **^job** commands that include spaces must be enclosed in “ ” (quotation marks).
- The values in the job definition file must be separated with blanks or tabs.
- Matching is case-insensitive.
- The sequencing of the records in the job definition file determines the order in which JFNOJOB searches for that string in the transaction file. When JFNOJOB finds the first match, the searching process is complete.

Use of Quotation Marks in the Job Definition File

As shown in the examples above, the **^job** command included in the job definition file is enclosed in “ ” (quotation marks). If you include command line options as part of your **^job** command and these command line options themselves require quoting, ensure the entire construct is quoted properly. For example, this **^job** command contains a **-z** command line option which refers to a file name containing spaces within its name and path:

```
"^job EXP_STMT -z""c:\My Data\Embedded Blanks.PDF"" "
```

Notice in this example the two sets of quotation marks after **-z**. These are necessary to define a quoted string within a quoted string. When Central reads the **^job** command, it expects the entire command to be enclosed in quotation marks. The second quotation mark encountered is considered to be the end of the **^job** command, UNLESS it is immediately followed by another quotation mark. The presence of two quotation marks together tells Central that it is dealing with a quoted string within a quoted string.

Example

When you create the default.def job definition file for your organization, include records for those transaction files that you know will not contain a **^job** command. For example, you may want to use Central to print an Accounting report on various printers throughout your organization. A client application creates the report output that prints in overlay mode on a simple preprinted template.

When Central receives the transaction file, it contains only the formatted report data, without a **^job** command on the first line of the transaction file. The report output in the transaction file appears as:

```
DATE: 03/31/2004                                            PAGE: 1
TIME: 1:05 PM            Any Company Inc.                   PERIOD: 03/2004

                         TRIAL BALANCE

Account Number/          ------------ Current Period ------------ Year-To-Date
Account Description      Debit              Credit              Balance

1234
Assets                   80,123.11          0.00                175,236.18
3567
Liabilities              0.00               51,255.75           (60,557.87)
5754
Accounts Receivable      60,899.90          0.00                380,276.89
68890
Accounts Payable         0.00               5,670.65            (6,880.22)
78563
Expenses                 0.00               9,100.25            (15,678.90)

GRAND TOTALS             141,023.01         66,026.65           472,396.08
```

The default.def file contains these records:

```
001 015 "Expense Statement" "^job EXP_STMT"
001 005 "Invoice" "^job BALANCE_SHEET"
001 015 "Trial Balance" "^job TRIAL_BALANCE"
003 006 "Posting Summary Report" "^job POST_SUMMARY"
```

When Central processes the transaction file containing the report data, it reads the first line of the transaction file looking for the **^job** command. When Central does not find the **^job** command, it initiates the job “DEFAULT” from the Job Table. The task id for DEFAULT is JFNOJOB. The executable program associated with JFNOJOB uses the contents of the default.def job definition file and searches through the transaction file attempting to match on one of the Search Strings within the range of records specified. The Search String, “Trial Balance”, matches on record 4 of the transaction file.

Note: Blank lines in report output count as records in the transaction file.

Once JFNOJOB identifies `^job TRIAL_BALANCE` as the **^job** command, it passes that information to Central, using its response file. Central continues with normal processing by linking the job name `TRIAL_BALANCE` to the tables in the Job Management Database.

JFNOJOB Considerations

JFNOJOB processing in Central versions prior to Version 5.4 required JFNOJOB to create a new transaction file consisting of the **^job** command and the contents of the original transaction file. In Central Version 5.4 and later, JFNOJOB uses its uses its response file to inform Central of the **^job** command. This means that, for subsequent processing, Central sets the skip lines substitution variable, @SkipLines, to zero, because the transaction file does not include a **^job** command. If you have tasks in your Job Management Database which hard code the skip lines value, these tasks may produce errors if processing involves the later JFNOJOB functionality.

Pre-5.4 JFNOJOB Processing

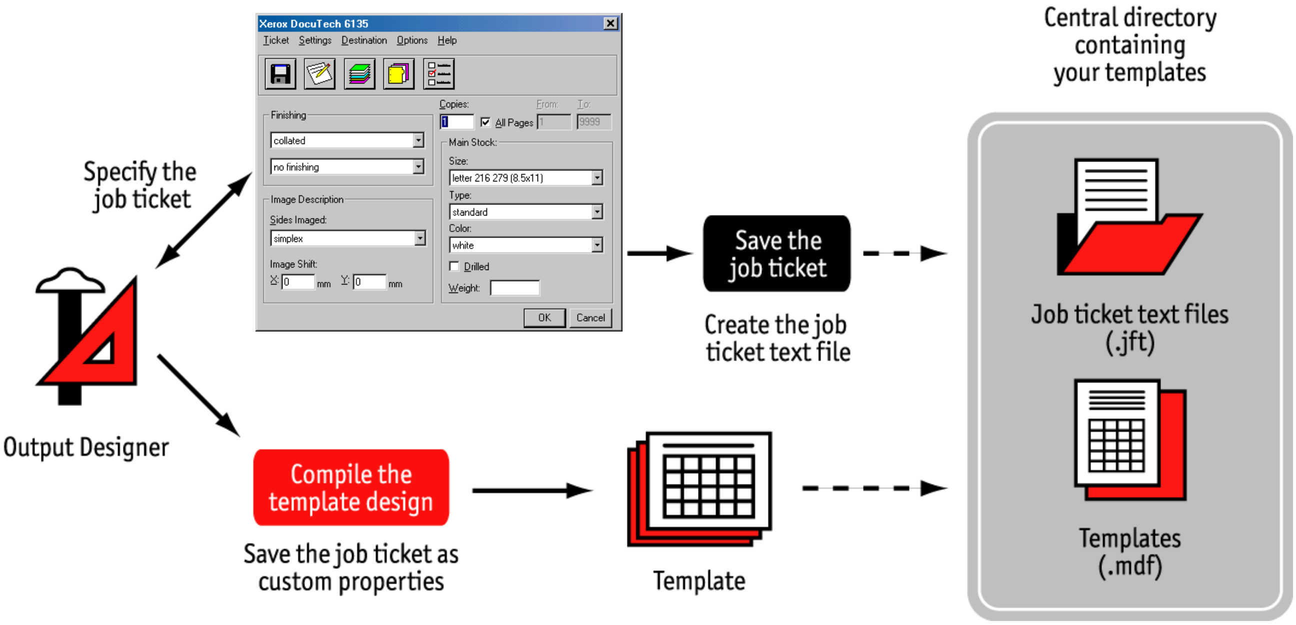
The Central Version 5.4 and later JFNOJOB functionality is enabled by the inclusion of the **-x** option in the JFNOJOB task in the Job Management Database. Removing the **-x** option will cause JFNOJOB processing to revert to that of the pre-5.4 versions.

8 Job Tickets

Job tickets are a feature of Xerox® DocuPrint and DocuTech printers that let users take advantage of a wide range of printing and finishing options. Requirements such as paper type, number of copies, and banner page information may be specified using job tickets. Adobe supports this feature by providing printer drivers for DocuPrint models 4850 NPS, 4890 NPS, 4090 NPS, and DocuTech models 135 and 6135.

Job tickets consist of a series of mandatory and optional records. Each record contains one or more attributes or instructions related to a specific print job requirement.

The following illustration shows the process for creating a job ticket for a form.



Xerox DocuTech 6135

Finishing: collated; no finishing

Image Description: Sides Imaged: simplex

Image Shift: X: 0 mm Y: 0 mm

Copies: 1 ☑ All Pages From: 1 To: 9999

Main Stock: Size: letter 216 279 (8.5x11)

Type: standard

Color: white

□ Drilled

Weight:

Ticket Settings Destination Options Help
OK
Cancel
Specify the job ticket
Output Designer
Save the job ticket
Create the job ticket text file
Compile the template design
Save the job ticket as custom properties
Template
Central directory containing your templates
Job ticket text files (.jft)
Templates (.mdf)

Using Output Designer, form designers can associate a job ticket with a form design. Information for the job ticket is captured through a series of dialog boxes. When the form designer saves the job ticket, Output Designer creates a text file for the job ticket. The text file, saved separately from the form, contains the job ticket records stored as command line options for the Print Agent.

At compile time, the form designer selects the option to create a form for use with Print Agent. Output Designer also saves the job ticket records in the custom properties of the form design. These are used to populate the job ticket dialog boxes with the appropriate values to allow for modifications.

Both the text file and the form are required at print time. Print Agent assembles the job ticket information and adds it to the beginning of the output stream. Command line options for Print Agent and printer operator intervention allow for overrides to job ticket records.

Printing a Transaction With a Job Ticket

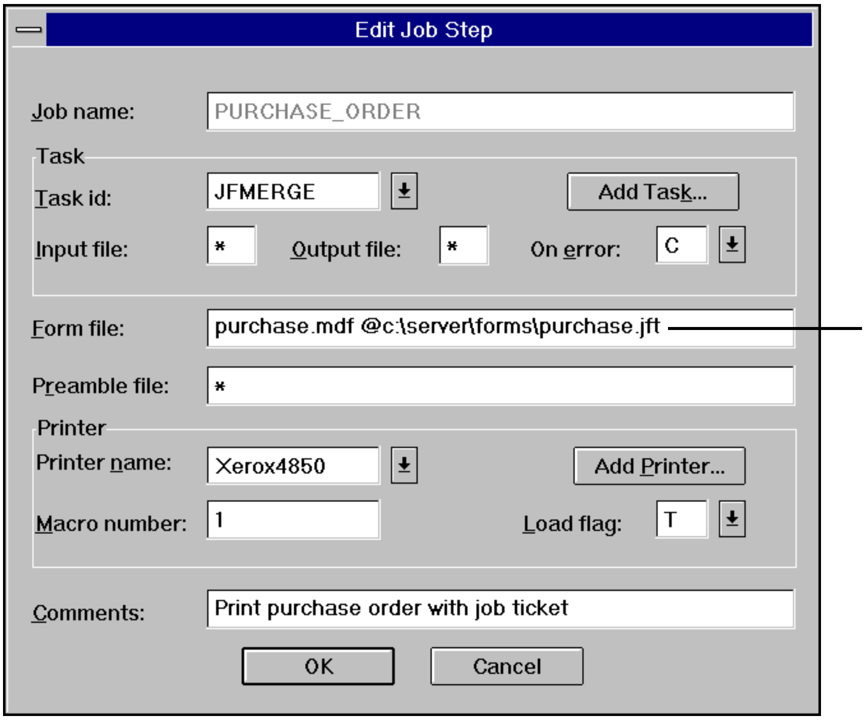
Print Agent processes job ticket information provided in the form and in the job ticket text file. It assembles the job ticket records into a format that is acceptable to the Xerox printer, and places this information at the top of the data stream.

Setting Up the Job Management Database

To provide job ticket information to the printer with the output, both the form (.MDF) and the text file (.JFT) must reside in the folder where Print Agent forms are stored.

In addition, you must configure the appropriate print job steps in the Job Management Database to ensure that Print Agent will use the job ticket text file. You specify the name of the job ticket text file as shown in the following illustration:

Specify Job Ticket Text File for Job Step



Edit Job Step

Job name: PURCHASE_ORDER

Task

Task id: JFMERGE

Input file: * Output file: * On error: C

Form file: purchase.mdf @c:\server\forms\purchase.jft

Preamble file: *

Printer

Printer name: Xerox4850

Macro number: 1 Load flag: T

Comments: Print purchase order with job ticket

Identify the form and the job ticket text file for the job.

When identifying the job ticket text file, prefix the file name with an “@” (at) and provide the full path.

Overriding Job Ticket Record Values

It is possible to override the job ticket records stored in the form and in the job ticket text file. This may be necessary in situations where the output is diverted to another device due to printer availability or maintenance schedules. You can use these methods to specify overrides for job ticket records:

- You can use the **-agv** command line option on the **^job** command of the transaction file. You can include as many of these options as necessary to specify the appropriate job ticket records.

  The Print Agent will process the **-agv** command line options provided that the program options in its Task Table entry includes the substitution variable, @OtherJobTokens. This substitution variable can appear anywhere in the list of program options, as long as it is after @MDFName. This placement ensures that the **-agv** options on the **^job** command override those in the job ticket text file.

  To specify the **-agv** command line option, use the syntax:

  ```
  -agv global=value
  ```

  where:

  global identifies a job ticket record using its corresponding global variable name. Use the name only; drop the “@” (at) prefix and the “.” (period) terminator. (For the global variables, see “Global Variables for Job Ticket Records” on page 62.)

  value is the value for the job ticket record. (For job ticket record values, see your printer documentation.)

  Enclose *global=value* in quotation marks when *value* contains spaces. For example, to identify the department that will be billed for the print job and to print on blue paper, include the following on the **^job** command:

  ```
  -agv"account=Finance Dept" -agvpcolor=blue
  ```

- The printer operator can add to or modify the job ticket records if the job is held before printing. Two ways to ensure that a job is held are:

  - Send the job to a queue that is set up as a “hold” queue. Jobs sent to a hold queue will require operator intervention.
  - Set the paper size to “INTERVENTION”. Since this is not a valid value, the print job will require operator intervention. Set this value on the **^job** command of the transaction file using the command line option: -agv”size=INTERVENTION”.

  Refer to the documentation that accompanies your printer for information on changing job ticket records at the printer operator level.

Global Variables for Job Ticket Records

The following table lists the job ticket records recognized by the supported Xerox DocuPrint and DocuTech printers. The table shows the corresponding global variable name, if any, used by Central.

For each **-agv** command line option, Central creates an entry in its Global dictionary. You can retrieve a stored value using the global variable name.

Note: For each printer, the supported job ticket records and their use (provided in the Description column in the table) may vary. Refer to your printer documentation for details.

Mapping Job Ticket Records to Central Global Variables

Job Ticket Record	Global Variable Name	Description
%XRXaccount:	@account.	The name of the account to charge for the printing costs associated with the job.
%XRXbegin:	n/a	Identifies the beginning of the job ticket. (Mandatory)
%XRXbinding:	@binding.	The edge on which to apply a length of pre-glued paper tape for binding.
%XRXbookletMaker:	@book.	Boolean value (True/False) indicating whether to apply a set of finishing options.
%XRXcopyCount:	@copies.	The number of copies to print.
%XRXdisposition:	n/a	The job disposition, such as print, or print and save. (Mandatory)
%XRXdocumentPaperColors:	@pcolor.	The color names of the media for the entire print job.
%XRXedgeStitching:	@stitch.	The type of stapling operation to apply to the document.
%XRXend.	n/a	Identifies the end of the job ticket. (Mandatory)
%XRXjobStartMessage:	@jsmsg.	Special instructions for the printer operator.
%XRXhighlightMappingAlgorithm:	@hmapalg.	A mapping algorithm used for all colors except the color designated by %XRXhighlightMappingColor:.
%XRXhighlightMappingColor:	@hmapcolor.	The highlight color with which to print the document.
%XRXhighlightPrintingColor:	@hprintcolor.	The color of the toner to use for highlight printing.

Job Ticket Record	Global Variable Name	Description
%XRXmessage:	@msg.	A banner page message.
%XRXmismatchedHighlightPrintingColor:	@hmismatch.	The action to take if the desired highlight color is not currently loaded in the printer.
%XRXpageExceptions:	@pe*n*. where *n* is 1, 2, ...50	Indicates that specified pages will print on media according to the values provided.
%XRXpageExceptions-medium:	@pe-m*n*. where *n* is 1, 2, ...50	Indicates that specified pages will print on media according to the values provided.
%XRXpagesToPrint:	Use both of: @from. and @to.	A range of pages to print.
%XRXpaperType-opacity:	@opacity.	The opaqueness of the media for the entire job.
%XRXpaperType-preFinish:	@pfinish. @holes. @tabs	The prefinished format of the media.
%XRXpaperType-size:	Use both of the following, which refer to width and height in millimeters: @xmm. @ymm.	The size of the media.
%XRXpaperType-weight:	@weight.	The weight of the media.
%XRXPDLformat:	n/a	The document type and version.
%XRXprinterName:	@printname.	A virtual printer with a particular set of characteristics and capabilities.
%XRXrecipientName:	@recipient.	Text strings that print on the banner page.
%XRXrequirements:	Use one or both of: @require. (collation option) @plex. (plex option)	Specifies some document imaging and output characteristics.
%XRXsenderName:	@sender.	Name associated with the job, usually the name of the submitter. (Mandatory)
%XRXtitle:	@title.	Name associated with the job, usually the name of the document. (Mandatory)
%XRXxImageShift:	@xshift.	The value to shift the decomposed page in the horizontal direction.
%XRXyImageShift:	@yshift.	The value to shift the decomposed page in the vertical direction.

9 Creating a Custom Agent

By creating your own custom Agents, you can further utilize the power and flexibility of Central. Because you write the application code to meet your requirements, these Agents provide unlimited capability for processing your tasks with Central. Custom Agents fit seamlessly into the Central domain. A custom Agent may be a stand-alone task, or it may be one of many job steps used in combination with any of the Agents available for Central.

Identifying a Custom Agent to Central and Central Control

You specify the custom Agent name and a description in the [Processes] section of the Central configuration file, jfserver.ini. For example:

```
CUSTAGT1 = Purchase Order Processing
CUSTAGT2 = Invoice Processing
```

Note: You can optionally identify your custom Agent to Central Control. This causes the Agent name to appear in the list boxes for the configuration and registration functions within Central Control.

For each Central Agent, a section appears in the Central configuration file, jfserver.ini, that identifies the Agent to Central. Likewise, you must add a section to the .ini file for each custom Agent you create, containing:

```
[CUSTAGT1]
ExePath = path of the executable program or script
IniFileName = fully qualified file name of the
 custom Agent .ini file
```

If the Agent does not use an .ini file, specify a dummy file name. The IniFileName parameter must not be blank. Central looks at this parameter regardless of whether you specify the substitution variable @IniFileName on the call to the Agent or not.

You can include a parameter for writing to a separate log file (see page 66) for the custom Agent:

```
LogFileName = fully qualified file name of the
 custom Agent log file
```

You can include a parameter that specifies the fully qualified name of a file editor for editing the custom Agent .ini file. This parameter is optional.

```
EditCfgRtn = notepad %s
```

Central Control invokes the editor when you select the custom Agent from its configuration list box. The name of the custom Agent .ini file is substituted for `%s`. Central Control can seamlessly integrate any editor, both third party or custom software, that uses a Microsoft Windows GUI.

Calling a Custom Agent

To call a custom Agent, you must create a Task Table entry for it. On the table entry, you specify the task identifier and the executable program name. In the Central .ini file, you specify the path for the custom Agent.

Specify any options you wish to pass to the custom Agent using the **^job** command or the **Program options** in the Task Table entry. When Central initiates the custom Agent, it resolves any substitution variables, and then launches the program with the appropriate command line. This is the standard processing for Central.

Microsoft Windows allows only 90 characters for the command line. If you require a longer command line for a call to a custom Agent, you can specify an *@parmfile* option on the command line, with the options for the custom Agent specified in an ASCII file. The custom Agent must be able to process the *@parmfile* as a valid file that contains a string of options, separated by blanks.

Returning Status and Parameters to Central

When the custom Agent completes its task, it must return a status to Central in one of two ways:

- As the exit value, with 0 indicating a successful completion, and any other value indicating an unsuccessful completion
- By creating a file called jetform.rsp in the current directory

When both methods of returning a status are used, the .rsp file takes precedence. The exit status within the .rsp file is used in place of the actual exit status of the Agent. When the custom Agent is a shell script, this may be the preferred approach.

When the custom Agent completes its task, it can also return parameters to Central. These parameters are available to substitute into the **Program options** during subsequent steps in the same job. Returned parameters are reset to their default values at the beginning of each job, so that one job cannot affect a later job.

The names of all return parameters and their default values are controlled by the [ReturnParms] section of jfserver.ini. If a parameter is not listed in this section, it can not be set by an Agent; any returned value for it is ignored. The [ReturnParms] section of jfserver.ini contains these values when distributed:

```
[ReturnParms]
NumberPages=0
OutputFileName=
User1=
User2=
User3=
User4=
User5=
User6=
User7=
User8=
User9=
User10=
```

The parameters `User1` through `User10` are available for use with custom Agents. You can add more such parameters, but to prevent clashes with Central parameters, ensure that all of your parameter names start with “User”. You are also free to set any default values for `User*`. `NumberPages=0` is reserved for use by Print Agent and must not be removed or altered.

To take advantage of this feature, custom Agents must use a field-nominated format in the response file (jetform.rsp). The first character in the file must be a “^” (caret), to indicate that the file is in field-nominated format. Each field in the file is delimited with **^field**, followed by the field name. The name of the field is “status” for the numerical exit code, “message” for the farewell message, and the name of the return parameter for return parameters. Field names are case-insensitive. Fields can occur in any order.

For example:

```
^field status
0
^field message
970311 14:01 Custom: This is a sample farewell message.
^field user3
This is the new value for User3.
```

All fields are optional, but the initial “^” is required to indicate that field-nominated format is being used. If the status field is absent, the code returned by the operating system is used.

The status value must be a single decimal integer. The farewell message may be multiline, but it must be less than 512 bytes long altogether. Returned parameters must be single line, because Central can not cope with parameter substitutions that result in multiline **Program options**.

Note: Using return parameters will cause Central logs with verbosity **-10** (see page 37) to get longer. This is because a trace message is logged every time the value of a return parameter is set, either to the default value (which happens frequently) or to the value returned by an Agent.

Writing to a Log File

There is no requirement to log messages from a custom Agent. However, your custom Agent may append to a log file that is:

- Accessible to Central, such as the Central log file, jfserver.log.

- Identified for the custom Agent in the Central .ini file by the parameter:

  ```
  LogFileName = fully qualified file name of the custom
  Agent log file
  ```

  The substitution variable @LogFileName is set to the Central log file when this parameter does not exist or its value is blank.

When you use the Central log file, appropriate error processing occurs. Central saves the relevant lines of the log file to a file in the error directory. When you use a separate log file for the custom Agent, Central is unable to do this.

To avoid conflicts in accessing the file, Central closes its log file before calling the custom Agent.

Follow these requirements for writing to a log file specifically created for the custom Agent:

- The log file is a flat ASCII file.
- Pass the log file name to the custom Agent using the substitution variable, @LogFileName.
- Open the log file for Append.
- Use the standard format for log messages including the date, time and the name of the Agent. For example:

  ```
  991027 14:05:32 CUSTAGT1: *Processing data file: 'filein.dat'
  ```

10 Examples

This chapter provides two examples of how to use Central. The examples demonstrate the ease with which you can create the table entries in the Job Management Database to use the powerful capability of Central and its Agents.

The first example, “Creating Print Tasks”, shows the entries required in the Job Management Database to handle the printing requirements of various users, and changes in circumstances or user needs. In addition, this example shows the **^job** command for each requirement and the use of different options with the **^job** command.

The second example, “Creating Many Tasks for a Transaction”, demonstrates the setup required to perform many tasks for a single transaction file, including data transformation, printing and calling a custom application.

Each example provides a number of scenarios that are accomplished using the **^job** command from the transaction file and/or the Job Management Database. The relationship between these two elements of Central is evident as you work through these examples, and each step is explained for the scenarios provided.

Creating Print Tasks

This example demonstrates how to create the appropriate entries in the tables of the Job Management Database for various print tasks, including:

- Handling different types of printers.
- Printing the parts of a multipart template on different printers.
- Printing all parts of a multipart template on one printer.
- Printing two templates from one transaction file.
- Printing either of two templates independently, from one transaction file.
- Printing in another location when a printer is temporarily unavailable.
- Printing a template at the most convenient location for the requester.

In addition, the following changes are illustrated:

- Changing printer equipment.
- Adding a device to the network.

For this example, a LAN services three departments in an organization. Each department has its own printer:

Department	Printer
Accounting	PCL 5 printer
Purchasing	PCL 6printer
Personnel	PostScript Level 2 printer

Template Management

When the templates are designed and compiled in Adobe Output Designer, you are preparing the template to print on a specific printer.

This organization uses four templates:

- Three-part Purchase Order
- Employee Salary History
- Employee Education Summary
- Employee Timesheet

This organization has some very specific requirements for handling these templates:

- For purchase orders raised by Accounting and Personnel, one part prints in Accounting and the other two parts print in Purchasing.
- For purchase orders raised by Purchasing, all three parts print in Purchasing.
- Personnel can choose to print its two personnel templates together or print either template independently.
- When the Purchasing department's printer is out of order, Purchasing uses the Accounting printer.
- For the Employee Timesheet, print the template at the most convenient location for the employee. This is a monthly timesheet template for capturing employee hours.

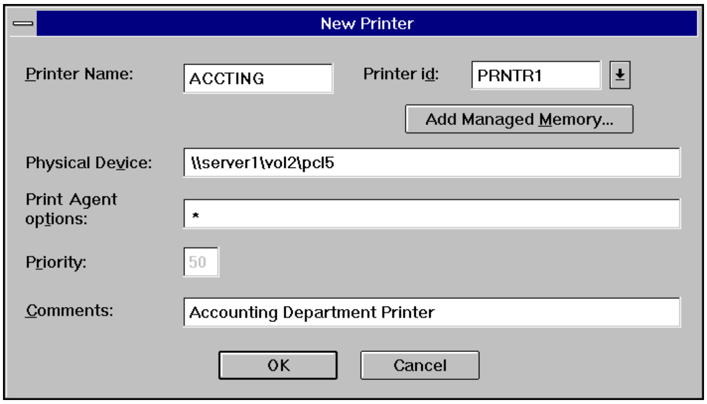
Template Name (.mdf)	Prints On	Description
po1pcl5	PCL 5	Single part purchase order that prints in Accounting.
po23pcl6	PCL 6	Two part purchase order that prints in Purchasing.
popcl6	PCL 6	Three part purchase order used internally by Purchasing.
salhist	PostScript Level 2	Employee Salary History template that prints independently or with the Employee Education Summary template.

Template Name (.mdf)	Prints On	Description
eduhist	PostScript Level 2	Employee Education Summary template that prints independently or with the Employee Salary History template.
popcl5	PCL 5	Three part purchase order used by Purchasing. It prints in Accounting when the Purchasing printer is unavailable.
timepcl5	PCL 5	Employee Timesheet template.
timepcl6	PCL 6	Employee Timesheet template.
timeps_2	PostScript Level 2	Employee Timesheet template.

Printer Table Setup

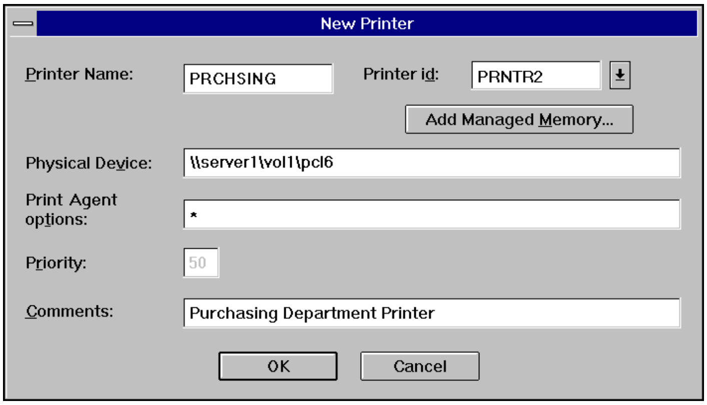
The printers for the organization are defined on the Printer Table in the Job Management Database. These entries are:

Accounting Printer



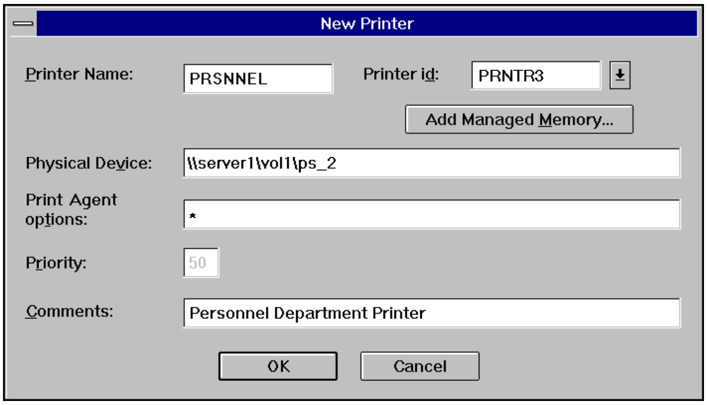
Printer Name: ACCTING
Printer id: PRNTR1
Physical Device: \\server1\vol2\pcl5
Print Agent options: *
Priority: 50
Comments: Accounting Department Printer

Purchasing Printer



Printer Name: PRCHSING
Printer id: PRNTR2
Physical Device: \\server1\vol1\pcl6
Print Agent options: *
Priority: 50
Comments: Purchasing Department Printer

Personnel Printer



Printer Name: PRSNNEL

Printer id: PRNTR3

Physical Device: \\server1\vol1\ps_2

Print Agent options: *

Priority: 50

Comments: Personnel Department Printer

Task Table Setup

All the tasks for these examples use Print Agent to print templates in various locations in the organization. When the Job Management Database is installed with Central, a control entry with a task identifier of “JFMERGE” is included on the Task Table. This task causes Central to launch Print Agent with the command line options and substitution variables specified in the Task Table.

For these examples, the task identifier “JFMERGE” is used either with the **-axx** option on the **^job** command or specified in the entries on the Job Table.

Job Table Setup

The steps for processing the transaction files submitted to Central are defined on the Job Table in the Job Management Database. The job name and optional printer name provided on the **^job** command determine the entries from the Job Table that Central uses to process the transaction file.

When creating the steps on the Job Table, you must determine the number of job steps required, their order and the task each performs, the printers or other devices you wish to use, whether an output file from one step becomes the input file for another, and so on.

To address each of the printing requirements of the organization, we will use the templates and the Printer and Task Table entries to build the entries for the Job Table in the Job Management Database.

For this example, some of the values for the Job Table entries are set to their default value. The transaction file is the input file to Print Agent, specified by the “*” (asterisk) in the **Input file** text box. **Output file** always contains an “*” as a placeholder in the table, as an output file is not required for any of the job steps. Unless otherwise noted, the **Macro number** is set to 1 and the **Load flag** is set to “T”, indicating that the template is downloaded to the printer as a temporary macro. The **On error** flag is set to “C” on each entry, specifying that processing continues in the event of an error. The **Comments** text box provides a description of the entry.

Printing Parts of a Multipart Template on Different Printers

In this organization, purchase orders raised by Accounting and Personnel print one part of the purchase order in Accounting and the other two parts in Purchasing.

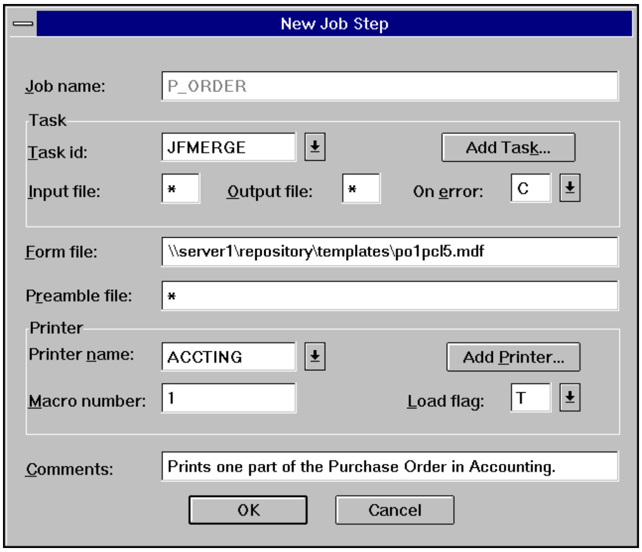
This requirement needs two entries on the Job Table. The entries use the same job name, but each specifies a printer where the parts of the purchase order print.

The **^job** command required in the transaction file is:

```
^job P_ORDER -z*
```

This command tells Central to process the job named P_ORDER from the Job Table. The **-z** option with the "*" (asterisk) specifies all characters; it tells Central to initiate all job steps for the job name P_ORDER and any printer name.

The Job Table contains these entries:



Job name: P_ORDER

Task id: JFMERGE

Input file: *

Output file: *

On error: C

Form file: \\server1\repository\templates\po1pcl5.mdf

Preamble file: *

Printer name: ACCTING

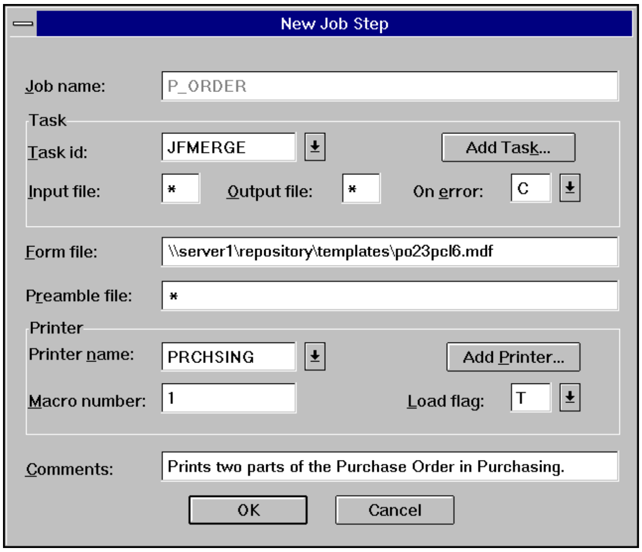
Macro number: 1

Load flag: T

Comments: Prints one part of the Purchase Order in Accounting.

Step 1

This entry specifies that for the job name P_ORDER, the one part purchase order template, po1pcl5.mdf, that prints in Accounting on the PCL 5 printer, performs the task JFMERGE from the Task Table.



Job name: P_ORDER
Task id: JFMERGE
Input file: *
Output file: *
On error: C
Form file: \\server1\repository\templates\po23pcl6.mdf
Preamble file: *
Printer name: PRCHSING
Macro number: 1
Load flag: T
Comments: Prints two parts of the Purchase Order in Purchasing.

Step 2

This entry specifies that for the job name P_ORDER, the two part purchase order template, po23pcl6.mdf that prints in Purchasing on the PCL 6 printer, performs the task JFMERGE from the Task Table.

Printing a Multipart Template

In this organization, a purchase order raised by Purchasing prints all three parts of the template in Purchasing.

There are two ways to meet this requirement:

- Using a **^job** command that specifies the appropriate template, and the desired printer and task:

  ```
  ^job POPCL6 -zPRCHSING -axxJFMERGE
  ```

  Central launches Print Agent to print the purchase order on the printer specified in the **^job** command. Central uses the job name as the template name for the call to Print Agent.

- Creating one entry on the Job Table that specifies the template, printer and task for printing all three parts of the purchase order in Purchasing.

  The **^job** command required in the transaction file is one of:

  ```
  ^job PURCHASE_ORDER -z*
  ```

  - or -

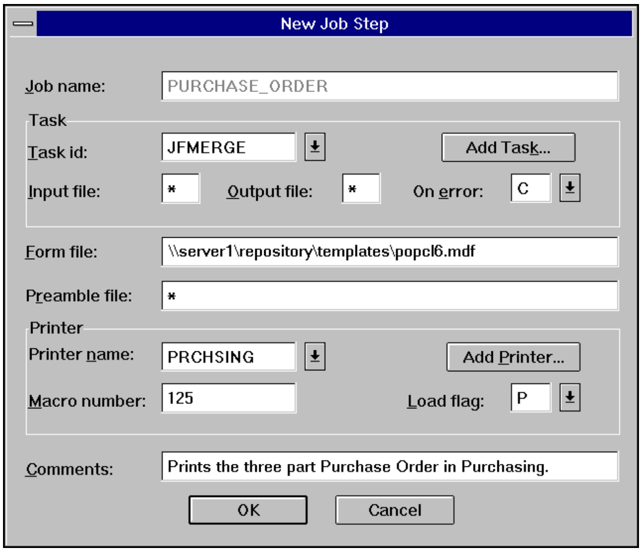
  ```
  ^job PURCHASE_ORDER
  ```

  - or -

  ```
  ^job PURCHASE_ORDER -zPRCHSING
  ```

  These commands tell Central to process the job named PURCHASE_ORDER from the Job Table. Using the first or second command, Central matches on all printers specified for that job name. Using the third command, Central specifically matches on the Purchasing printer. Since this requirement needs only one Job Table entry, Central matches on only one entry in the Job Table using any of these **^job** commands.

The Job Table contains this entry:



New Job Step

Job name: PURCHASE_ORDER

Task

Task id: JFMERGE

Input file: * Output file: * On error: C

Form file: \\server1\repository\templates\popcl6.mdf

Preamble file: *

Printer

Printer name: PRCHSING

Macro number: 125 Load flag: P

Comments: Prints the three part Purchase Order in Purchasing.

Step 1

This entry specifies that for the job name PURCHASE_ORDER, the three part purchase order template, popcl6.mdf, that prints in Purchasing on the PCL 6 printer, performs the task JFMERGE from the Task Table.

The template is loaded into the printer as a Permanent macro, with a macro number of 125.

Printing templates Independently or Together

Personnel can choose to print its two personnel templates, Employee Salary History and Employee Education Summary, together or print either template independently. These two templates can be requested separately at any time, but print together at salary review time.

The data for these two templates is contained in a personnel database. Whether the templates print separately or together, the same information is extracted from the database and written to the Central transaction file. This saves significant programming and analysis time.

This requirement contains three components:

- Print the Employee Salary History template on the Personnel printer. This requirement does not need any entries on the Job Table when you use this **^job** command in the transaction file:

  ```
  ^job SALHIST -zPRSNNEL -axxJFMERGE
  ```

  The **^job** command provides all the necessary information for Central to print the Employee Salary History template. The **-axx** option tells Central to launch Print Agent. Central does not perform a lookup on the Job Table when the **-axx** option is used. Instead, Central uses the job name SALHIST as the template name when it launches Print Agent. The **-z** option specifies the Personnel printer.

- Print the Employee Education Summary template on the Personnel printer. This requirement does not need any entries on the Job Table when you use this **^job** command in the transaction file:

  ```
  ^job EDUHIST -zPRSNNEL -axxJFMERGE
  ```

This command provides all the necessary information for Central to print the Employee Education Summary template. The **-axx** option tells Central to launch Print Agent. Central does not perform a lookup on the Job Table when the **-axx** option is used. Instead, Central uses the job name EDUHIST as the template name when it launches Print Agent. The **-z** option specifies the Personnel printer.

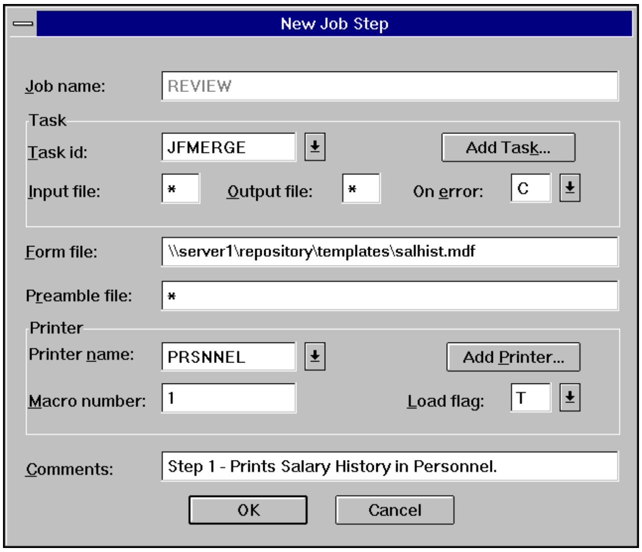
- Print both templates on the Personnel printer. This requirement uses the same transaction file as printing each of the Employee Salary History and the Employee Education Summary templates alone. Two entries are needed in the Job Table using the same job name and printer specifications, but with different template specifications.

  The **^job** command required in the transaction file is:

  ```
  ^job REVIEW
  ```

  This command tells Central to process the job named REVIEW from the Job Table. The **-z** option is not specified, so Central defaults this option to -z*. The "*" (asterisk) specifies all printers; it tells Central to match on job name to all entries on the Job Table.

  The Job Table contains these entries:



Job name: REVIEW

Task

Task id: JFMERGE

Input file: *
Output file: *
On error: C

Form file: \\server1\repository\templates\salhist.mdf

Preamble file: *

Printer

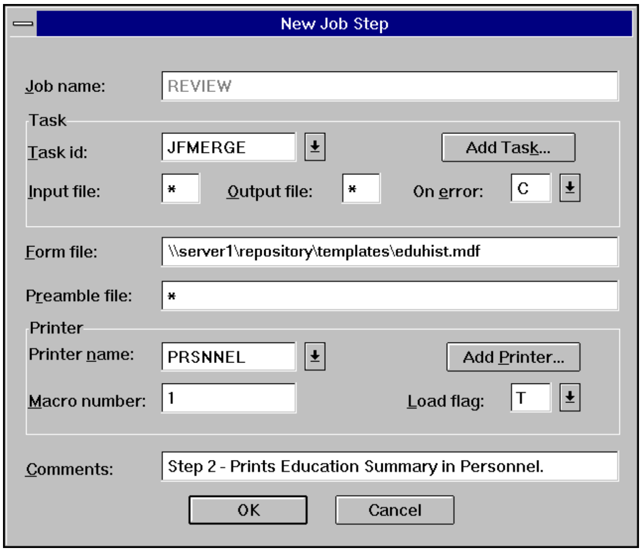
Printer name: PRSNNEL

Macro number: 1
Load flag: T

Comments: Step 1 - Prints Salary History in Personnel.

Step 1

This entry specifies that for the job name REVIEW, the Employee Salary History template, salhist.mdf, that prints in Personnel on the PostScript Level 2 printer, performs the task named JFMERGE from the Task Table.



Job name: REVIEW
Task id: JFMERGE
Input file: *
Output file: *
On error: C
Form file: \\server1\repository\templates\eduhist.mdf
Preamble file: *
Printer name: PRSNNEL
Macro number: 1
Load flag: T
Comments: Step 2 - Prints Education Summary in Personnel.

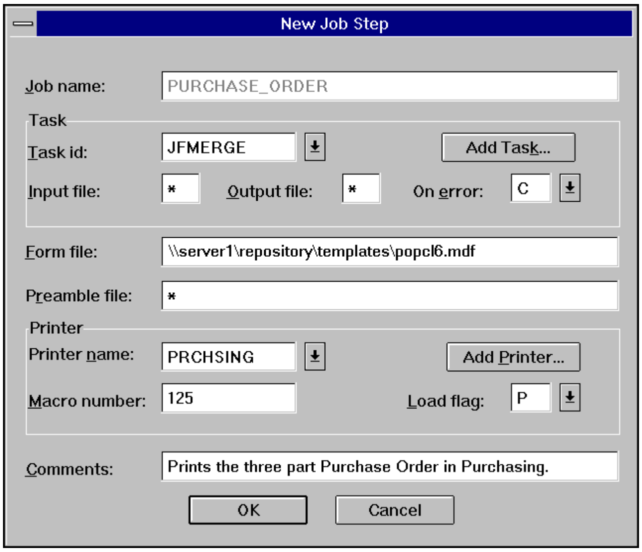
Step 2

This entry specifies that for the job name REVIEW, the Employee Education Summary template, eduhist.mdf, that prints in Personnel on the PostScript Level 2 printer, performs the task named JFMERGE from the Task Table.

Changing Printers

When the Purchasing department's printer is out of order, Purchasing uses the Accounting printer. Purchasing can continue to print purchase orders if their printer is not available by making the following change to the Job Table:

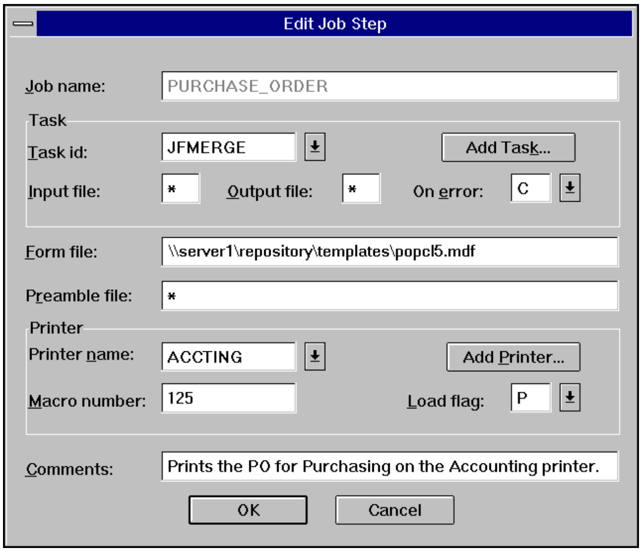
From:



Job name: PURCHASE_ORDER
Task id: JFMERGE
Input file: *
Output file: *
On error: C
Form file: \\server1\repository\templates\popcl6.mdf
Preamble file: *
Printer name: PRCHSING
Macro number: 125
Load flag: P
Comments: Prints the three part Purchase Order in Purchasing.

This entry in the Job Table specifies printing the job PURCHASE_ORDER using the template popcl6.mdf and the printer PRCHSING. The template is compiled for the PCL 6 printer.

To:



Job name: PURCHASE_ORDER

Task

Task id: JFMERGE

Input file: * Output file: * On error: C

Form file: \\server1\repository\templates\popcl5.mdf

Preamble file: *

Printer

Printer name: ACCTING

Macro number: 125 Load flag: P

Comments: Prints the PO for Purchasing on the Accounting printer.

By changing the template and printer specification in the Job Table, Purchasing can continue to print their purchase orders on the Accounting printer with a minimum of disruption. No change is required to the client application. The Accounting printer is a PCL 5 printer, so the template name changes to popcl5.mdf to accommodate the Accounting printer.

Note that if Purchasing and Accounting used the same make and model of printer, a change to the Printer Table to map the PRCHSING printer from PRNT2 to PRNT1 (Printer Id of the accting printer) provides the desired result. No change would be required to the Job Table, because in this case the same template could print on either printer.

Printing at a Conveniently Located Printer

The last requirement is for employees to print their monthly Employee Timesheet template at the most conveniently located printer relative to their work area.

With Different Printers

Within the organization, three Employee Timesheet templates are required to accommodate the three different printers, PCL 5, PCL 6, and PostScript Level 2. The requirement can be met in two ways, by:

- Using a **^job** command that specifies the appropriate template for the desired printer, and task:

  ```
  ^job TIMEPCL5 -zACCTING -axxJFMERGE
  ```

  - or -

  ```
  ^job TIMEPCL6 -zPRCHSING -axxJFMERGE
  ```

  - or -

  ```
  ^job TIMEPS_2 -zPRSNNEL -axxJFMERGE
  ```

  This solution requires no entries in the Job Table. Central launches Print Agent to print the Employee Timesheet template at the printer in the **^job** command. Central uses the job name as the template name for the call to Print Agent.

- Creating three separate entries on the Job Table, with the same job name, specifying a different printer. Note that using this solution requires an entry on the Job Table for each different printer in the organization.

  The **^job** command required in the transaction file is:

  For Accounting:

  ```
  ^job TIMESHEET -zACCTING
  ```

  For Purchasing:

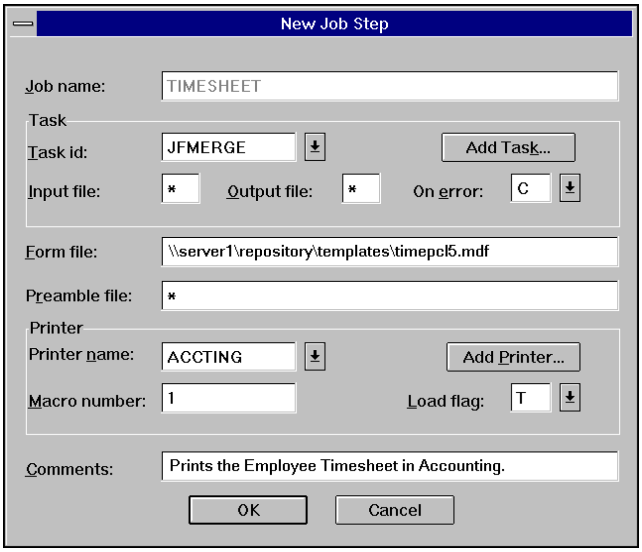
  ```
  ^job TIMESHEET -zPRCHSING
  ```

  For Personnel:

  ```
  ^job TIMESHEET -zPRSNNEL
  ```

  Each command tells Central to match the job name and printer name to entries in the Job Table. Only one entry is found on the Job Table for each job name/printer name. The appropriate printer must be specified on the **^job** command for the Employee Timesheet template to print at the desired location.

  The Job Table contains these entries:



New Job Step

Job name: TIMESHEET

Task

Task id: JFMERGE

Input file: * Output file: * On error: C

Form file: \\server1\repository\templates\timepcl5.mdf

Preamble file: *

Printer

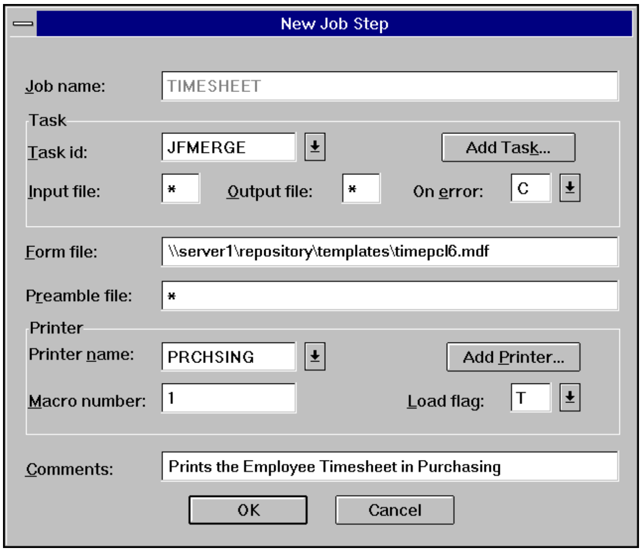
Printer name: ACCTING

Macro number: 1 Load flag: T

Comments: Prints the Employee Timesheet in Accounting.

Step 1

This entry specifies that for the job name TIMESHEET, the Employee Timesheet template timepcl5.mdf that prints in Accounting on the PCL 5 printer, performs the task JFMERGE from the Task Table.



Job name: TIMESHEET

Task id: JFMERGE

Input file: * Output file: * On error: C

Form file: \\server1\repository\templates\timepcl6.mdf

Preamble file: *

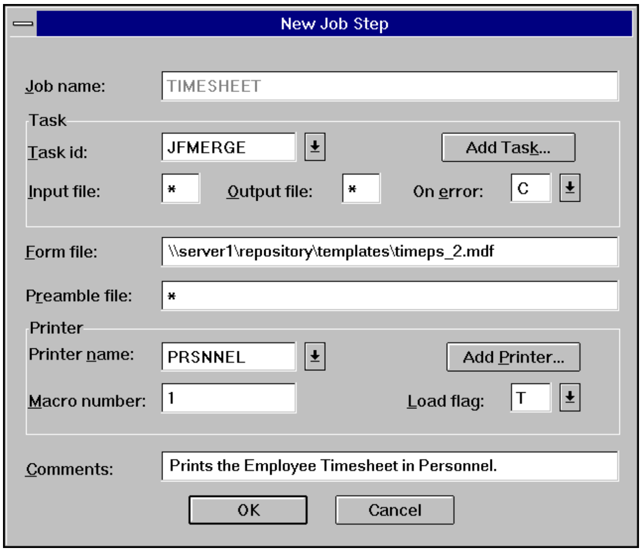
Printer name: PRCHSING

Macro number: 1 Load flag: T

Comments: Prints the Employee Timesheet in Purchasing

Step 1

This entry specifies that for the job name TIMESHEET, the Employee Timesheet template timepcl6.mdf that prints in Purchasing on the PCL 6 printer, performs the task JFMERGE from the Task Table.



Job name: TIMESHEET

Task id: JFMERGE

Input file: * Output file: * On error: C

Form file: \\server1\repository\templates\timeps_2.mdf

Preamble file: *

Printer name: PRSNNEL

Macro number: 1 Load flag: T

Comments: Prints the Employee Timesheet in Personnel.

Step 1

This entry specifies that for the job name TIMESHEET, the Employee Timesheet template timeps_2.mdf that prints in Personnel on the PostScript Level 2 printer, performs the task JFMERGE from the Task Table.

With the Same Printer

If the same make and model of printer is used throughout the organization, only one Employee Timesheet template and one entry in the Job Table is required. If the entire organization used the PCL 5 printer, a template is compiled for this printer only.

There are two ways of printing the Employee Timesheet template on the most conveniently located printer:

- Using a **^job** command that specifies the appropriate template, and the desired printer and task:

  ```
  ^job TIMEPCL5 -zPRCHSING -axxJFMERGE
  ```

  Central launches Print Agent to print the Employee Timesheet template on the printer specified by the **-z** option on the **^job** command. Central uses the job name as the template name for the call to Print Agent.

- Creating a single entry in the Job Table and using the **^job** command that specifies a job name and printer name:

  ```
  ^job TIMESHEET -zACCTING
  ```

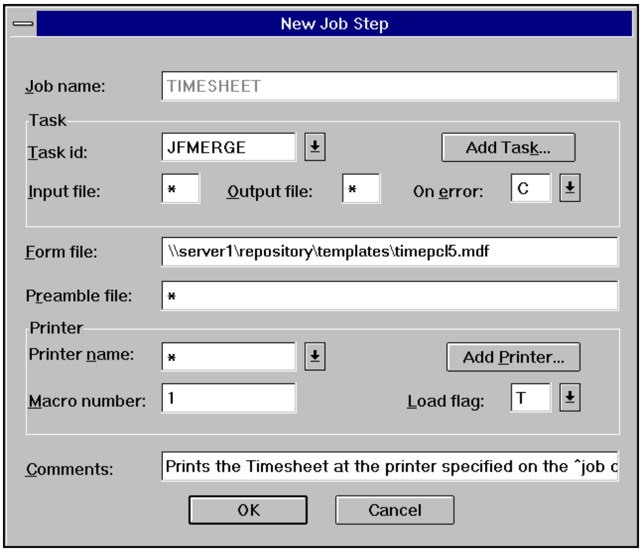
  - or -

  ```
  ^job TIMESHEET -zPURCHSING
  ```

  - or -

  ```
  ^job TIMESHEET -zPRSNNEL
  ```

  Note that the entry in the Job Table uses an “*” (asterisk) for the printer name. This tells Central to use the printer specified by the **-z** option on the **^job** command.



New Job Step

Job name: TIMESHEET

Task

Task id: JFMERGE

Input file: *

Output file: *

On error: C

Form file: \\server1\repository\templates\timepcl5.mdf

Preamble file: *

Printer

Printer name: *

Macro number: 1

Load flag: T

Comments: Prints the Timesheet at the printer specified on the ^job c

Step 1

This entry specifies that for the job name TIMESHEET, the Employee Timesheet template timepcl5.mdf prints on the printer specified by the **-z** option on the **^job** command, and performs the task JFMERGE from the Task Table.

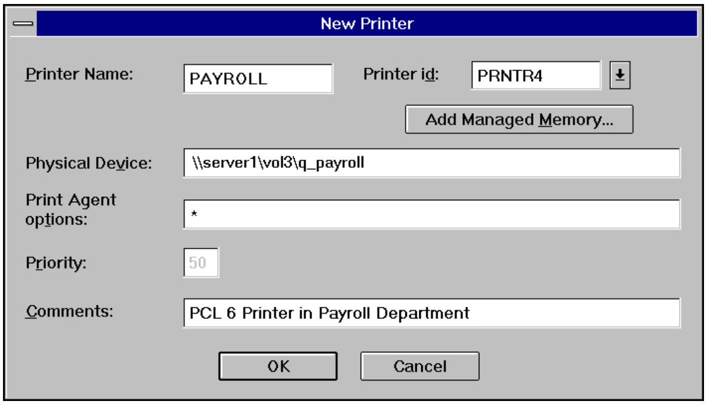
Handling Equipment Changes

If Personnel obtains a new PCL 6 printer to replace its PostScript printer, all that is necessary is to re-compile the templates (salhist.mdf and eduhist.mdf) in Output Designer for the new printer. No changes are required to the Job Management Database or client application as long as the same .mdf names are used.

Adding a New Device to the Network

The Payroll department is added to the network and they wish to print the Employee Salary History template on their PCL 6 printer.

To add the new device, the following Printer Table entry is needed:

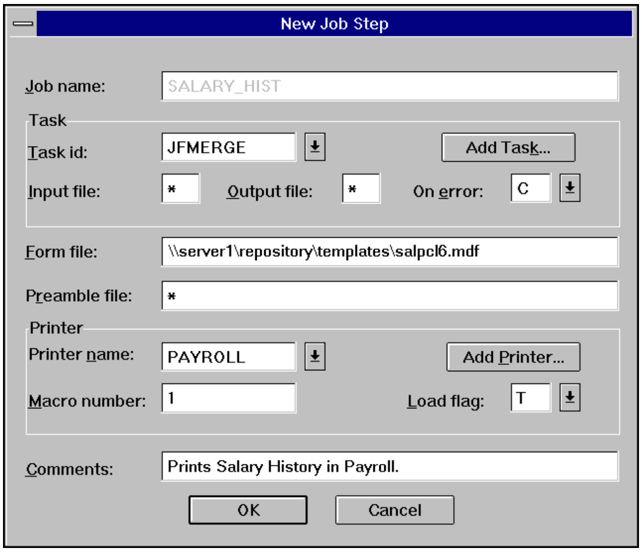


Printer Name: PAYROLL
Printer id: PRNTR4
Physical Device: \\server1\vol3\q_payroll
Print Agent options: *
Priority: 50
Comments: PCL 6 Printer in Payroll Department

The only template for the Employee Salary History is compiled for a PostScript printer. To print this template on a PCL 6 printer, it must be compiled for that printer. That new template is called salpcl6.mdf. To enable printing through Central, create a **^job** command for the transaction file that specifies the job name only:

```
^job SALARY_HIST
```

and create an entry in the Job Table:



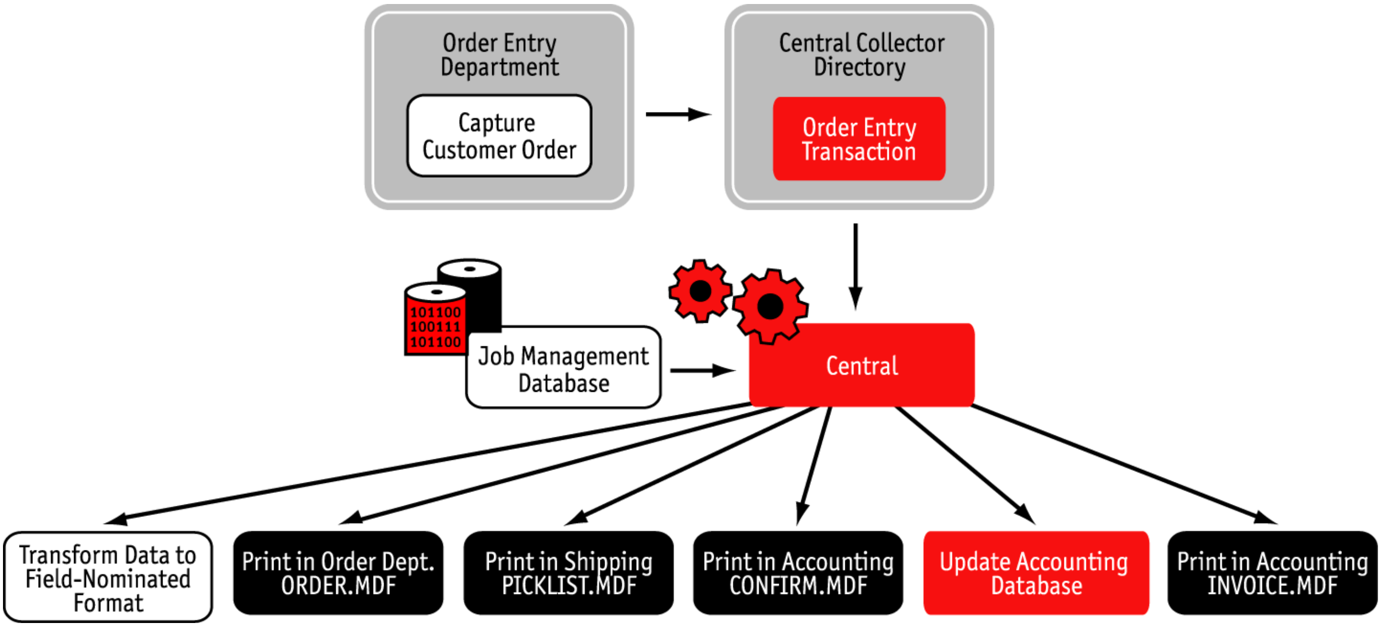
Job name: SALARY_HIST
Task id: JFMERGE
Input file: *
Output file: *
On error: C
Form file: \\server1\repository\templates\salpcl6.mdf
Preamble file: *
Printer name: PAYROLL
Macro number: 1
Load flag: T
Comments: Prints Salary History in Payroll.

Creating the appropriate table entries in the Job Management Database in conjunction with setting the **^job** command for a transaction enables you to use the powerful capability of Central to manage the printing requirements of an organization.

Creating Many Tasks for a Transaction

This example uses a customer order, captured in a legacy Order Entry system, to illustrate that many tasks may be executed from one transaction file. Central uses the **^job** command from the transaction file and Job Table in the Job Management Database to determine the steps for processing the transaction, which include:

- Launching Transformation Agent to convert the data from the format created by the legacy Order Entry system to Central field-nominated format.
- Printing different views of the customer order on different printers, using Print Agent.
- Launching a custom application to update an existing database and create an extract file for printing an invoice in Accounting.


Order Entry Department
Capture Customer Order
Central Collector Directory
Order Entry Transaction
101100
100111
101100
Job Management Database
Central
Transform Data to Field-Nominated Format
Print in Order Dept. ORDER.MDF
Print in Shipping PICKLIST.MDF
Print in Accounting CONFIRM.MDF
Update Accounting Database
Print in Accounting INVOICE.MDF

Template Management

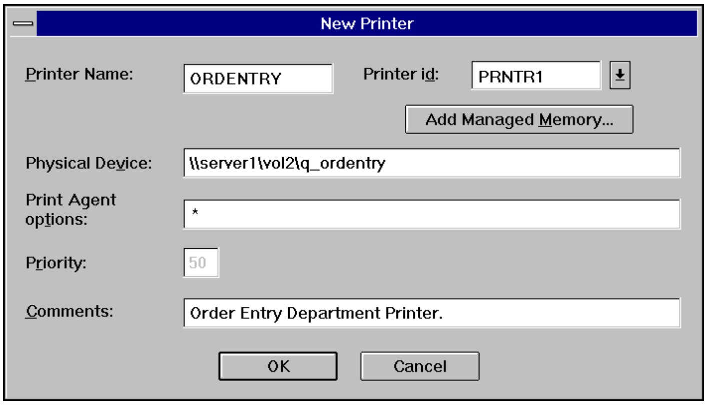
Each department receives a print of the order with the template view that is required for that department:

Template Name (.mdf)	Description
order	Single-part customer order template that prints in Order Entry.
picklist	Single-part picking list that prints in Shipping.
confirm	Single-part order confirmation template that prints in Accounting.
invoice	Multipart invoice template that prints in Accounting.

Setting Up the Printer Table

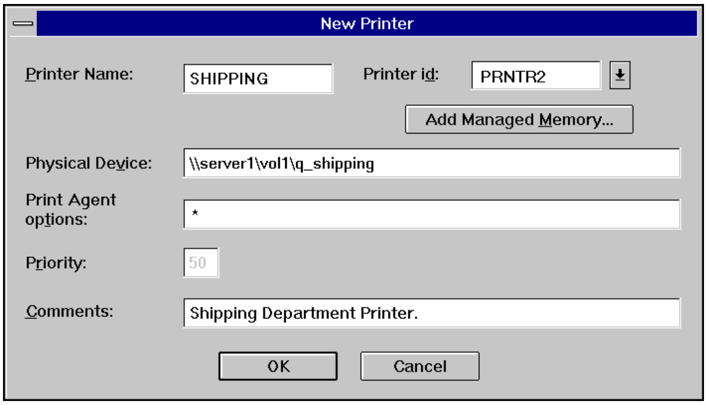
The printers for the organization are defined on the Printer Table in the Job Management Database. All departments use the same type of printer.

Order Entry Printer



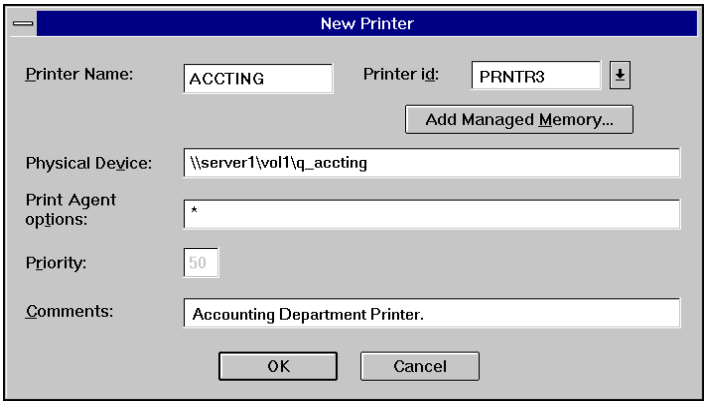
Printer Name: ORDENTRY
Printer id: PRNTR1
Physical Device: \\server1\vol2\q_ordentry
Print Agent options: *
Priority: 50
Comments: Order Entry Department Printer.

Shipping Printer



Printer Name: SHIPPING
Printer id: PRNTR2
Physical Device: \\server1\vol1\q_shipping
Print Agent options: *
Priority: 50
Comments: Shipping Department Printer.

Accounting Printer



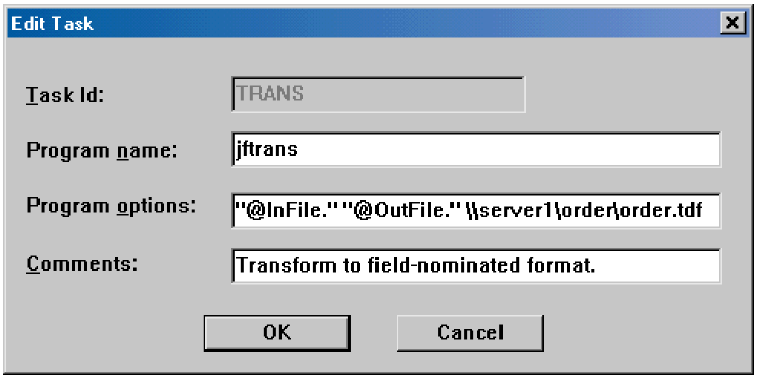
Printer Name: ACCTING
Printer id: PRNTR3
Physical Device: \\server1\vol1\q_accting
Print Agent options: *
Priority: 50
Comments: Accounting Department Printer.

Setting Up the Task Table

Some of the tasks for this example use Print Agent to print templates in various locations in the organization. When the Job Management Database is installed with Central, a control entry with a task identifier of “JFMERGE” is included on the Task Table. When this task identifier is used, Central launches Print Agent with the command line options and substitution variables specified in the Task Table.

One of the tasks uses Transformation Agent to convert the transaction file from a non-Central format to Central field-nominated format.

When Transformation Agent is installed, a control entry with a task identifier of “JFSAMPLE” is added to the Task Table. This is intended as a sample of the task identifier associated with Transformation Agent. Using the sample as a guide, the Task Table entry looks like:



Task Id: TRANS
Program name: jftrans
Program options: "@InFile." "@OutFile." \\server1\order\order.tdf
Comments: Transform to field-nominated format.

The **Program options** contain options necessary for the completion of the task. These are passed by Central to the **Program name**, which is Transformation Agent for this task. In this example:

@InFile The name of the temporary file that is input to the current job step. On the Job Table for this step, the Input file name is “*”. This means that the transaction file is the file for conversion.

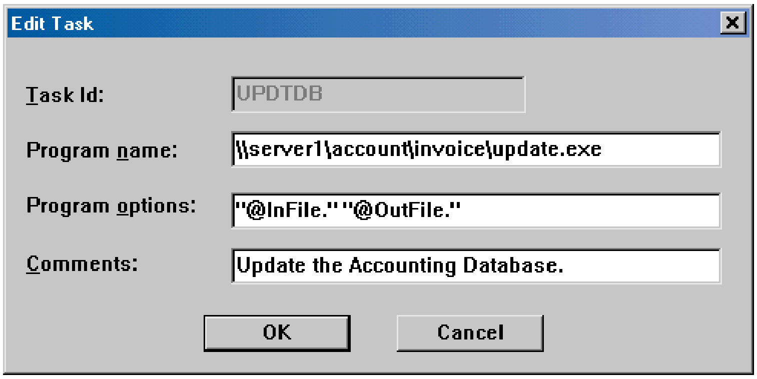
@OutFile The name of the temporary file that is output from the current job step. On the Job Table for this step, the Output file name is “A”. This means that Transformation Agent writes the transformed data to a temporary file named “A”, created by Central.

\\server1\order\order.tdf The name of the definition file required by Transformation Agent to complete its task.

-asl@SkipLines Tells Transformation Agent whether to skip the first line of the transaction file. If the transaction file contains a **^job** command on its first line, Central substitutes a “1” for @SkipLines; otherwise it substitutes a “0”. The option is passed with the appropriate value:

- **-asl1**, that tells Transformation Agent to ignore the first line of the transaction file, or
- **-asl0**, that tells Transformation Agent to process the first line of the transaction file.

An additional entry is required in the Task Table for the custom application that updates the Accounting database. The entry is:



Task Id: UPDTDB

Program name: \\server1\account\invoice\update.exe

Program options: "@InFile." "@OutFile."

Comments: Update the Accounting Database.

The **Program options** passed to the custom application by Central for this task are:

@InFile The name of the temporary file that is input to the current job step. On the Job Table for this step, the Input file name is “A”. This means that the file the custom application uses to update the Accounting database is a temporary file created by Central in a previous job step.

@OutFile The name of the temporary file that is output from the current job step. On the Job Table for this step, the Output file name is “B”. This means that the custom application creates an output file that it writes to a temporary file named “B”, created by Central.

Setting Up the Job Table

The steps for processing the transaction file associated with a customer order are defined on the Job Table in the Job Management Database. The information provided on the **^job** command determines the entries from the Job Table that Central uses to process the transaction file.

In creating the steps on the Job Table for processing a transaction file, you must determine the number of job steps required, their order and the task each performs, the printers or other devices you wish to use, whether an output file from one step becomes the input file for another, and so on.

For this example, the following tasks are required:

- Transform the data in the transaction file from the format created by the legacy Order Entry system to Central field-nominated format.
- Print the customer order in the Order Entry Department.
- Print the picking list in the Shipping Department.
- Print the customer confirmation in Accounting.
- Update the Accounting database using a custom application.
- Print the invoice in Accounting using the data file created by the custom application.

The tasks are required in the order presented above. When Central receives the transaction file, it must first convert the data to field-nominated format for processing by the other tasks. Once the data is in field-nominated format, a print is produced in the Order Entry, Shipping and Accounting Departments, each with its own required view of the data. The Order Entry view includes all the data captured for the order, including the customer information, and stock numbers, item description, and pricing information. The Shipping Department view requires only the stock numbers and item descriptions for the picking list. The Accounting view, which is mailed to the customer as a confirmation, requires only the customer information and item descriptions. A custom application then updates the Accounting database for customer billing purposes. Any additional information, such as payment terms, is included in the database update. Once the Accounting information is in place, the custom application creates a data file used for printing the invoice in Accounting.

The **^job** command required in the transaction file is:

```
^job CUST_ORDER -z*
```

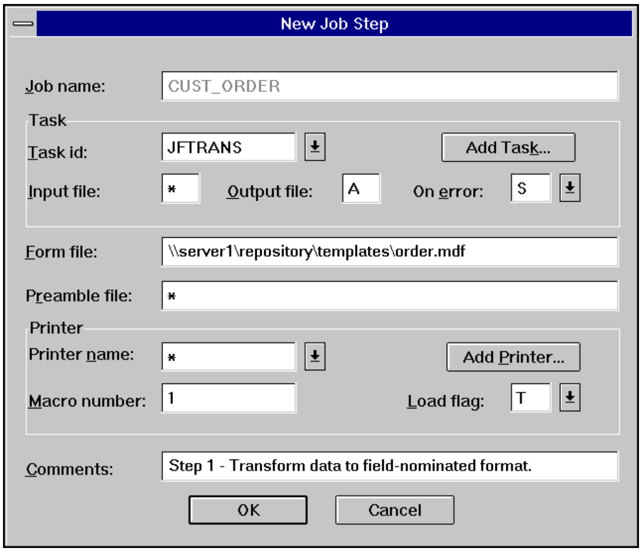
This command tells Central to process the job named CUST_ORDER from the Job Table. The **-z** option with the “*” (asterisk) specifies all printers; it tells Central to initiate all job steps for the job name CUST_ORDER and any printer name.

Step 1 - Transform the Customer Order Transaction File

The first step that Central performs is a conversion of the transaction file into field-nominated format. This is accomplished using Transformation Agent.

The step uses the data from the original transaction file as input to Transformation Agent. Transformation Agent creates an output file in field-nominated format, used as input for steps 2 through 5.

The Job Table entry for this step is:


New Job Step

Job name: CUST_ORDER

Task

Task id: JFTRANS

Input file: * Output file: A On error: S

Form file: \\server1\repository\templates\order.mdf

Preamble file: *

Printer

Printer name: *

Macro number: 1 Load flag: T

Comments: Step 1 - Transform data to field-nominated format.

Step 1

This entry specifies that for the job name CUST_ORDER, Central performs the task JFTRANS from the Task Table.

The original transaction file, specified by the “*” in **Input file** is the input file to Transformation Agent. Central creates a temporary output file for the data converted into field-nominated format, specified by the “A” in **Output file**.

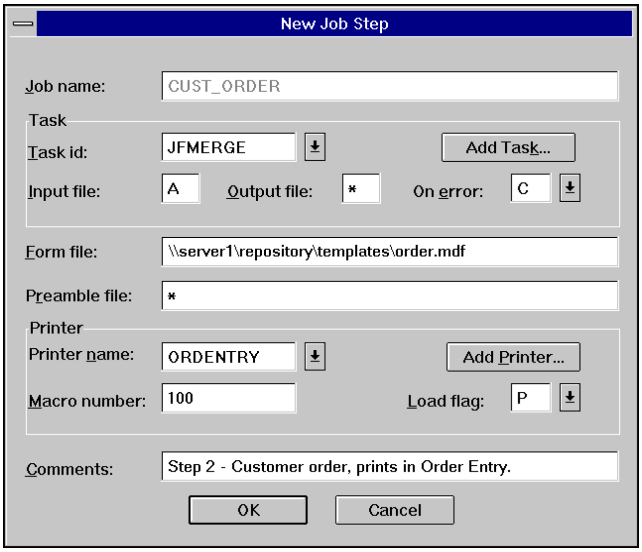
The “S” in **On error** tells Central to stop processing this transaction file if an error code is returned in executing this step.

Steps 2, 3, and 4 - Print Order Views

The next steps print a copy of the customer order in Order Entry, a picking list in Shipping and a customer confirmation in Accounting. Each of these prints use the data from the converted transaction file to print a different view of the order.

These printing requirements need three entries on the Job Table. The entries use the same job name, but each specifies a printer where a different view of the order prints.

The Job Table contains these entries:


New Job Step

Job name: CUST_ORDER

Task

Task id: JFMERGE

Input file: A Output file: * On error: C

Form file: \\server1\repository\templates\order.mdf

Preamble file: *

Printer

Printer name: ORDENTRY

Macro number: 100 Load flag: P

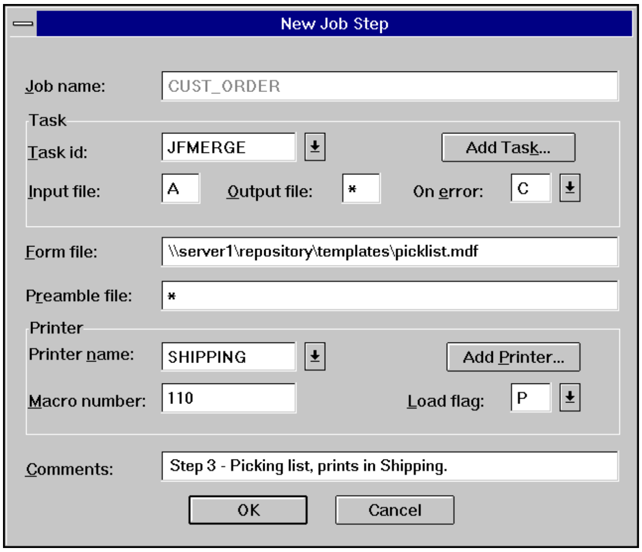
Comments: Step 2 - Customer order, prints in Order Entry.

Step 2

This entry specifies that for the job name CUST_ORDER, the template order.mdf that prints in Order Entry, performs the task JFMERGE from the Task Table.

The **Output file** “A” from Step 1 is the **Input file** to this job step.

Since orders print in this department all day, the template is loaded into the printer as a Permanent macro, with a macro number of 100.



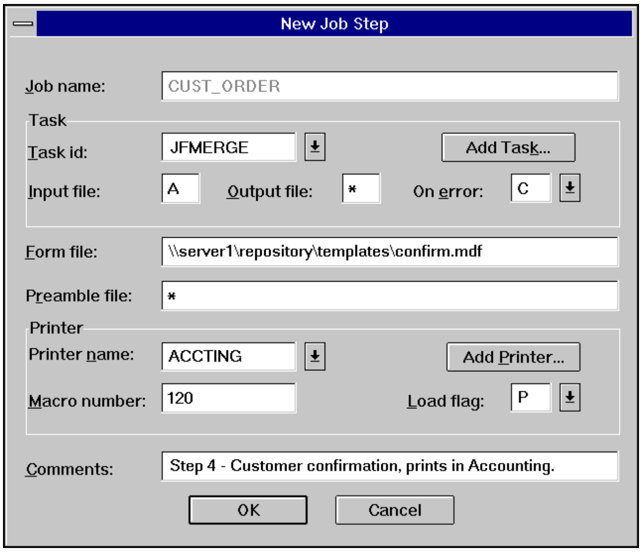
Job name: CUST_ORDER
Task id: JFMERGE
Input file: A
Output file: *
On error: C
Form file: \\server1\repository\templates\picklist.mdf
Preamble file: *
Printer name: SHIPPING
Macro number: 110
Load flag: P
Comments: Step 3 - Picking list, prints in Shipping.

Step 3

This entry specifies that for the job name CUST_ORDER, the picking list template, picklist.mdf, that prints in Shipping, performs the task JFMERGE from the Task Table.

The **Output file** “A” from Step 1 is the **Input file** to this job step.

The template is loaded into the printer as a Permanent macro, with a macro number of 110.



Job name: CUST_ORDER
Task id: JFMERGE
Input file: A
Output file: *
On error: C
Form file: \\server1\repository\templates\confirm.mdf
Preamble file: *
Printer name: ACCTING
Macro number: 120
Load flag: P
Comments: Step 4 - Customer confirmation, prints in Accounting.

Step 4

This entry specifies that for the job name CUST_ORDER, the confirmation template, confirm.mdf, that prints in Accounting, performs the task JFMERGE from the Task Table.

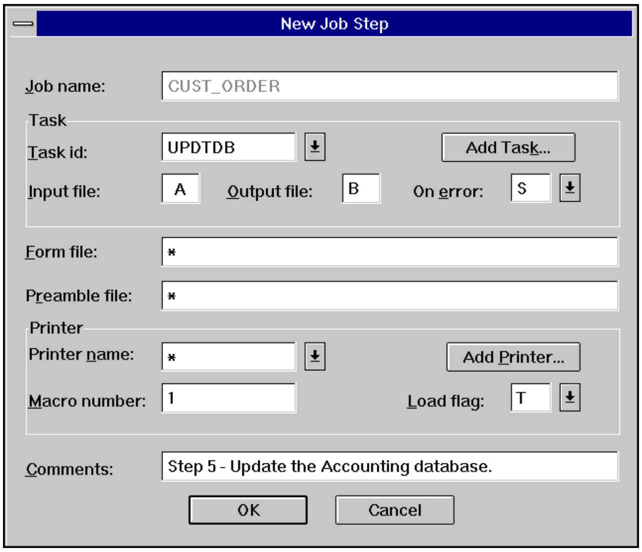
The **Output file** “A” from Step 1 is the **Input file** to this job step.

The template is loaded into the printer as a Permanent macro, with a macro number of 120.

Step 5 - Update the Accounting Database

This step specifies a task for a custom application that updates an Accounting database used for customer billing purposes. The step uses the **Output file** “A” from Step 1 as input to the custom application. The custom application creates an output file in Central field-nominated format, used as input for printing the invoice.

To meet this requirement, the Job Table contains this entry:



Job name: CUST_ORDER
Task id: UPDTDB
Input file: A
Output file: B
On error: S
Form file: *
Preamble file: *
Printer name: *
Macro number: 1
Load flag: T
Comments: Step 5 - Update the Accounting database.

Step 5

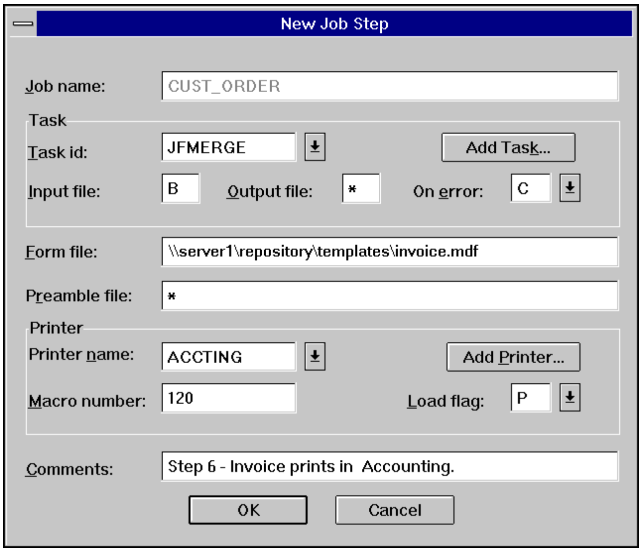
This entry specifies that for the job name CUST_ORDER, the custom application from the Task Table associated with the task identifier, UPDTDB is launched by Central.

The **Output file** “A” from Step 1 is the **Input file** to the custom application. The custom application updates the Accounting database and creates an **Output file**, “B”.

Step 6 - Print Invoice for Accounting

This step prints an invoice for the customer order in the Accounting Department. It uses the **Output file** “B” from Step 5 as input for printing the invoice.

To meet this requirement, the Job Table contains this entry:



Job name: CUST_ORDER
Task id: JFMERGE
Input file: B
Output file: *
On error: C
Form file: \\server1\repository\templates\invoice.mdf
Preamble file: *
Printer name: ACCTING
Macro number: 120
Load flag: P
Comments: Step 6 - Invoice prints in Accounting.

Step 6

This entry specifies that for the job name CUST_ORDER, the invoice template, invoice.mdf, that prints in Accounting, performs the task JFMERGE from the Task Table.

The **Output file** “B” from Step 5 is the **Input file** to this job step.

Index