Documaker

Documaker Workstation Supervisor Guide

version 11.2

Skywire Software, L.L.C.
3000 Internet Boulevard
Suite 200
Frisco, Texas 75034
www.skywiresoftware.com

Phone:	**(U. S.)**	**972.377.1110**
	(EMEA)	**+44 (0) 1372 366 200**
FAX:	**(U. S.)**	**972.377.1109**
	(EMEA)	**+44 (0) 1372 366 201**
Support:	**(U. S.)**	**866.4SKYWIRE**
	(EMEA)	**+44 (0) 1372 366 222**
		support@skywiresoftware.com

PUBLICATION COPYRIGHT NOTICE

Copyright © 2008 Skywire Software, L.L.C. All rights reserved.
Printed in the United States of America.
This publication contains proprietary information which is the property of Skywire Software or its subsidiaries. This publication may also be protected under the copyright and trade secret laws of other countries.

TRADEMARKS

Skywire® is a registered trademark of Skywire Software, L.L.C.

Docucorp®, its products (Docucreate™, Documaker™, Docupresentment™, Docusave®, Documanage™, Poweroffice®, Docutoolbox™, and Transall™), and its logo are trademarks or registered trademarks of Skywire Software or its subsidiaries.

The Docucorp product modules (Commcommander™, Docuflex®, Documerge®, Docugraph™, Docusolve®, Docuword™, Dynacomp®, DWSD™, DBL™, Freeform®, Grafxcommander™, Imagecreate™, I.R.I.S. ™, MARS/NT™, Powermapping™, Printcommander®, Rulecommander™, Shuttle™, VLAM®, Virtual Library Access Method™, Template Technology™, and X/HP™ are trademarks of Skywire Software or its subsidiaries.

Skywire Software (or its subsidiaries) and Mynd Corporation are joint owners of the DAP™ and Document Automation Platform™ product trademarks.

Docuflex is based in part on the work of Jean-loup Gailly and Mark Adler.

Docuflex is based in part on the work of Sam Leffler and Silicon Graphic, Inc.
Copyright © 1988-1997 Sam Leffler.
Copyright © 1991-1997 Silicon Graphics, Inc.

Docuflex is based in part on the work of the Independent JPEG Group.
The Graphic Interchange Format© is the Copyright property of CompuServe Incorporated. GIF℠ is a Service Mark property of CompuServe Incorporated.

Docuflex is based in part on the work of Graphics Server Technologies, L.P.
Copyright © 1988-2002 Graphics Server Technologies, L.P.

All other trademarks, registered trademarks, and service marks mentioned within this publication or its associated software are property of their respective owners.

SOFTWARE COPYRIGHT NOTICE AND COPY LIMITATIONS

Your license agreement with Skywire Software or its subsidiaries, authorizes the number of copies that can be made, if any, and the computer systems on which the software may be used. Any duplication or use of any Skywire Software (or its subsidiaries) software in whole or in part, other than as authorized in the license agreement, must be authorized in writing by an officer of Skywire Software or its subsidiaries.

PUBLICATION COPY LIMITATIONS

Licensed users of the Skywire Software (or its subsidiaries) software described in this publication are authorized to make additional hard copies of this publication, for internal use only, as long as the total number of copies does not exceed the total number of seats or licenses of the software purchased, and the licensee or customer complies with the terms and conditions of the License Agreement in effect for the software. Otherwise, no part of this publication may be copied, distributed, transmitted, transcribed, stored in a retrieval system, or translated into any human or computer language, in any form or by any means, electronic, mechanical, manual, or otherwise, without permission in writing by an officer of Skywire Software or its subsidiaries.

DISCLAIMER

The contents of this publication and the computer software it represents are subject to change without notice. Publication of this manual is not a commitment by Skywire Software or its subsidiaries to provide the features described. Neither Skywire Software nor it subsidiaries assume responsibility or liability for errors that may appear herein. Skywire Software and its subsidiaries reserve the right to revise this publication and to make changes in it from time to time without obligation of Skywire Software or its subsidiaries to notify any person or organization of such revision or changes.

The screens and other illustrations in this publication are meant to be representative, not exact duplicates, of those that appear on your monitor or printer.

Contents

Chapter 1, Setting Up the System

Chapter 2, Using Resource Libraries

Chapter 3, Setting Up Printers

Chapter 4, Customizing Your System

Chapter 5, Importing and Exporting Information

Chapter 6, Maintaining Your System

CHAPTER 1

Setting Up the System

This manual serves as a reference tool for system supervisors. It contains information on how to maintain and customize Documaker Workstation.

This chapter provides an overview of the system and contains information to help get you started configuring your operating environment, installing the software, and creating user IDs.

You will find information on...

System Overview

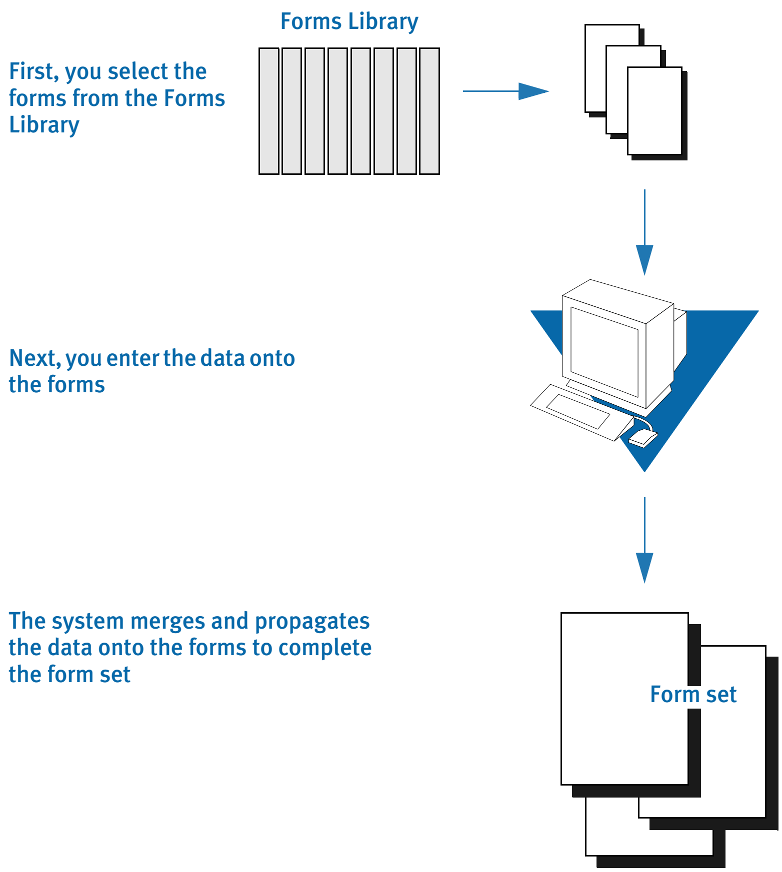
Documaker Workstation is the workstation-based form set entry and publishing piece of the Documaker system. Documaker (the Rules Publishing system) is a total form set automation system which enables forms-intensive industries such as insurance, finance, utilities, and government to automate enterprise-wide forms and forms processing.

You collect the information you need for your form sets from various sources including manually entered data, system default data, archived data, and data extracted from external application systems. Documaker Workstation lets you enter that information and print complete, collated form sets on laser printers.

The system's unique data import and export feature enhances the data entry process. The system lets you import data files that automatically fill or propagate the data onto a form's fields. Exporting lets you extract data from Documaker Workstation for use in other applications or for import back into the system.

The system's user interface makes data entry and forms processing easy for non-technical users. You enter basic information and Documaker Workstation displays a list of applicable forms. You then select specific form sets in which to enter data.

This illustration shows the process:

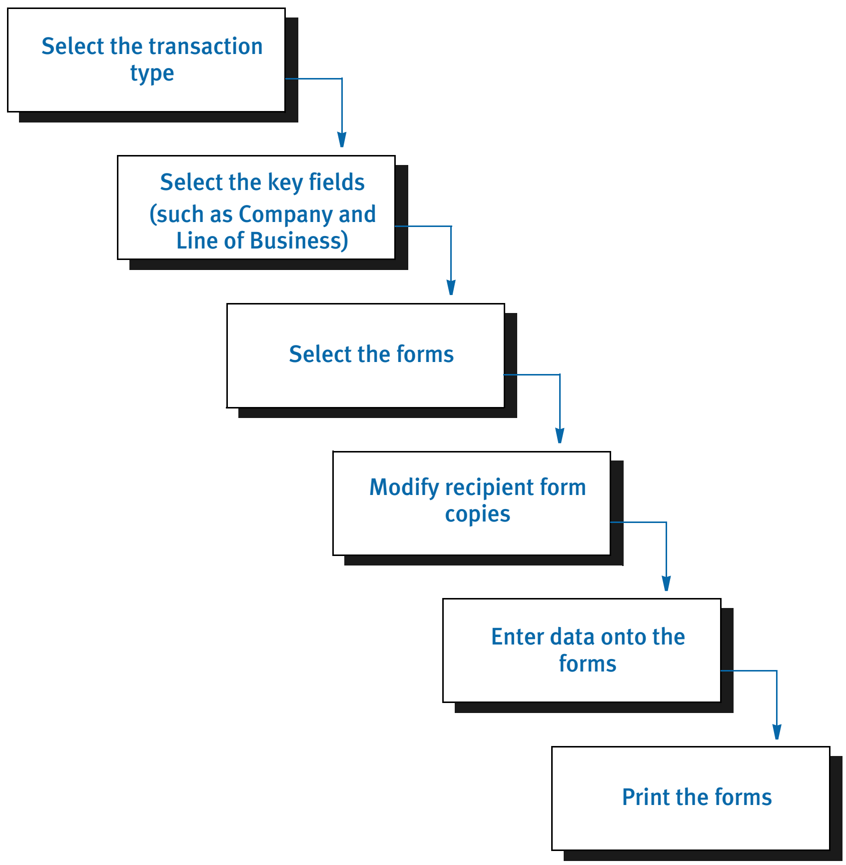


The system lets you import data from an internal or an external source and apply the data to specific forms. You can also export system data to use in other application systems or to import back into the system.

Task Flow

From the system's main window, you choose File, New and enter the key field information, such as selecting a company and corresponding lines of business, for a particular form set. The system displays the list of forms applicable for that key field data.

You can then change the number of recipient copies in the forms list. Next, you enter the required variable data into each form. If multiple forms require the same field data, the system automatically propagates the data into subsequent forms.

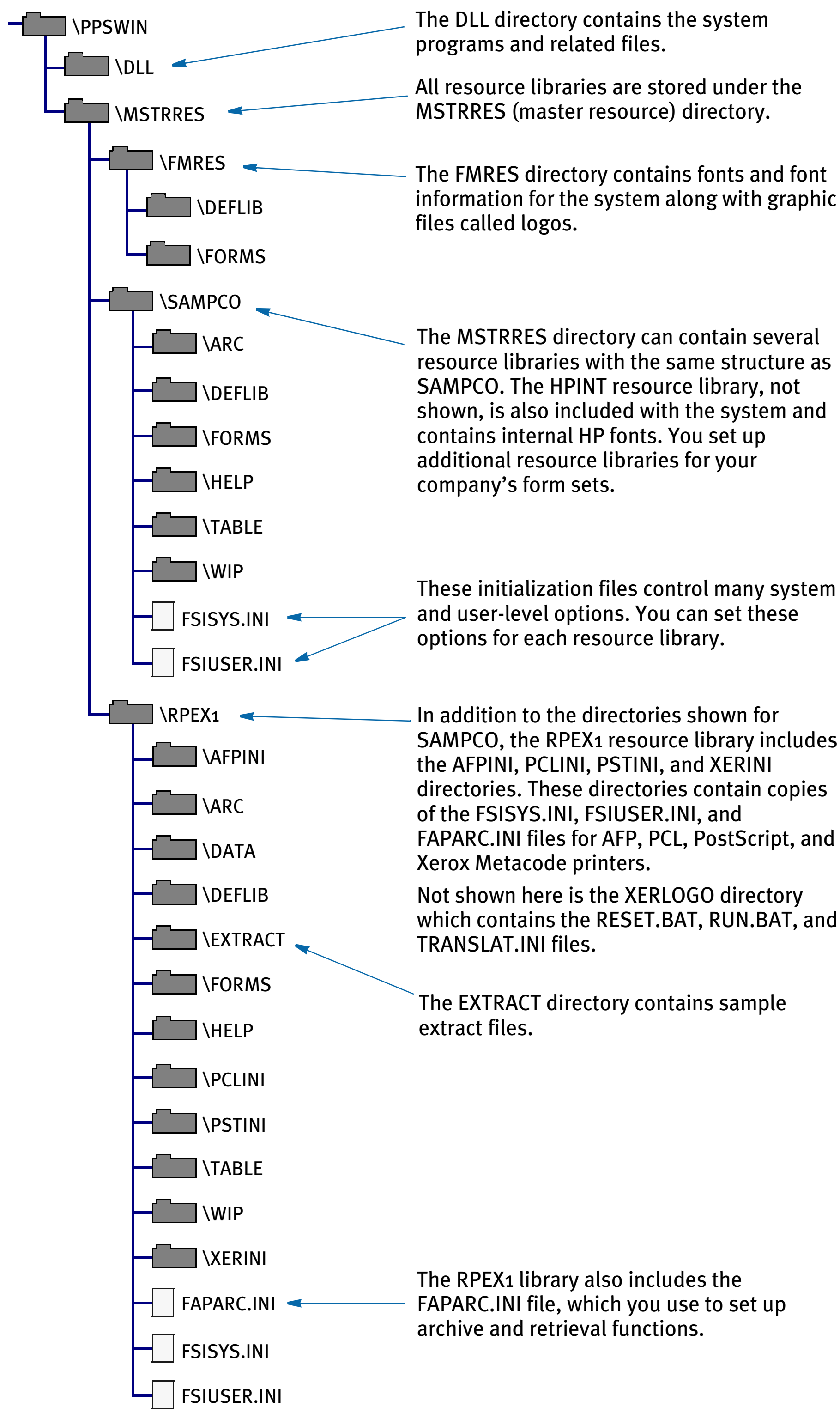


You can print a form set once you complete data entry, or send a form set to a batch queue for printing later. When you print form sets, the system merges the data with the appropriate form *template* and prints the form set. The system then archives the data and the form set template separately.

Note: For detailed information on using the system's many features, see the Documaker Workstation User Guide.

System Directory Structure

Because the system lets you use external resources such as forms, you can modify menus, form and image organization, and data archiving and printing functions. The example below illustrates the system's directory and subdirectory structure:



When you set up your system and build your resource library, you work primarily with the MSTRRES directory structure. You modify the files and subdirectories within each resource library to implement external resources and customize your system.

MSTRRES, the resource library directory, contains various subdirectories and files that define and store system resources. Primary components of the MSTRRES directory include the FSISYS.INI and FSIUSER.INI files, the Form Set Definition file (FORM.DAT) and the forms files. You manipulate many of the subdirectories and files within this directory when you create new resource libraries.

PRINTER SUPPORT

The system supports the GDI (Graphical Device Interface) print driver (GDIW32.DLL). When you select the GDI printer type, the driver calls the print drivers installed and set up by a system supervisor. You can then select a specific printer for printing your form sets. With the GDI printer drivers, printers such as the HP5500C, HP1200C, and Xerox 4700 color printers are supported.

The system also supports the following print devices and print languages:

- PCL. Print Control Language (PCL), designed and developed by Hewlett Packard, incorporates commands in compact escape sequence codes embedded in the print data stream. The embedded command approach minimizes data transmission and command decoding overhead. PCL is used by numerous printing devices.
- AFP. Advanced Function Printing (AFP), developed by IBM for its Print Services Facility (PSF), is a print server language that generates data streams of objects. The data streams merge with print controls and system commands to generate IPDS (Intelligent Printer Data Stream). The IPDS is then sent to an IBM PSF printer.
- Metacode. Metacode, developed by Xerox, is the native language of its Centralized Printing Systems. Metacode contains code that defines printing functions to the hardware. Metacode lets you position data via page addressing and specify multiple record fonts and data orientations.

PostScript. PostScript developed by Adobe Systems, is a page description language that describes a document from a computer composition system to a raster output printing system. PostScript describes pages at a high level such as a series of abstract graphic objects.

Configuring the Operating Environment

Set up considerations differ between LAN and single user workstations. Here are some general considerations that apply to both environments:

- The system stores a resource library's forms separate from the applicable data.
- The system stores all transaction data, including print-ready data and archived data (stored in compressed format).

Configuring a Single User Workstation

Independent or single user workstations do not involve communication with external peripherals other than a printer. Consequently, system configuration simply requires setting the system path (*c:\ppswin\dll*) in your AUTOEXEC.BAT file. The installation routine completes this procedure automatically.

You can optionally configure Carbon Copy® or PC Anywhere® and your modem so your system will be accessible to external technical support personnel.

Configuring a Networked System

When configuring the system for use on a LAN, keep the following considerations in mind:

- Documaker Workstation assumes that the software, the resource libraries, and form set data reside on the same storage device (hard drive).
- If you plan to use large form set libraries, select a server powerful enough to accommodate heavy input/output processing and high speed data transfer.

Perform the following tasks on your network server to configure your network system to run Documaker Workstation:

1. If you use Novell NetWare access the CAPTURE command and add NT (no tab expansion) to the command line. By adding this command you reduce the potential for problems when Documaker Workstation sends binary files to a printer. Refer to your Novell Utilities Reference Guide for more specific information.
2. Configure Carbon Copy® or PC Anywhere® and your modem on the network to make your system operations accessible to external technical support personnel.

Installing the Software

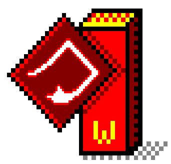
If any questions arise during the installation process and you need assistance, contact Support Services.

Once you make sure your computer has the correct hardware and software, you can install the system.

Simply insert the CD into your CD-ROM drive and use a text editor to view the *readme* file on the CD for detailed installation instructions.

The system icon

By default, the system creates an icon for the SAMPCO resource library. If you use multiple resource libraries, such as if you process insurance policies for multiple companies, you may want to create icons for each company's resource library. The graphic below shows the system icon.



Re-installing the Software

If at any time in the future, you need to re-install the system, keep the following points in mind:

- If you re-install the software to the same drive and sub-directory, the PATH statement in your AUTOEXEC.BAT file will not be affected.
- If you re-install the software to a different drive or sub-directory, the new path statements will append to any previous settings in the AUTOEXEC.BAT file.

  This makes the system use path statements from the previous installation, and ignore the new path statements. If you want the system to use the new path statements, you must delete the old path statements in the AUTOEXEC.BAT file.
- System and user INI files (FSISYS.INI and FSIUSER.INI) are located in the PPSWIN\DLL directory and the SAMPCO directory. The settings of the INI files located in the PPSWIN\DLL directory will not be affected; however, the INI files in the SAMPCO directory will be overwritten.
- The INI files in the PPSWIN\DLL directory:
  - Specify default values and other defined parameters
  - Provide system flexibility
  - Store system default settings
  - Allow a user ID to be reset
- The INI files in the SAMPCO and RPEX1 directories:
  - Allow users to specify Documaker Workstation functions
  - Determine how Documaker Workstation uses the library resources
- FSISYS.INI - Controls information related to the entire system, such as systems settings and program function calls.
- FSIUSER.INI - Controls settings which vary between resource libraries, such as sorting options, archival mode, and import and export ability.

Setting Up Multiple Users

Setting up multiple users includes these tasks:

- Creating user IDs
- Testing the user IDs
- Creating user icons

The system provides you with two default user IDs: *DOCUCORP* and *USER1*. *DOCUCORP* is the system supervisor's ID and password. It lets you access all system options, including system maintenance, user ID set up, INI files, and so on. Do not disclose the DOCUCORP ID to those who are not authorized system supervisors or support technicians. *USER1* is the generic user ID and password for data entry users.

Note: You can also customize the FSIUSER.INI file so logging on happens automatically. This may be more convenient in some situations, but keep in mind it also circumvents the security features. See Bypassing the Logon Window on page 14 for more information.

User ID Guidelines

The system supervisor sets up, modifies, and deletes user IDs. You can set up user IDs either before or after you install the resource library. Consider these guidelines when you set up user IDs:

- IDs can consist of up to 64 alphanumeric characters. Each ID must be unique.
- The system segregates data by user ID, so users only have access to their own data. You can set up IDs so that supervisory personnel have access to subordinate user data. System supervisors (DOCUCORP ID) have access to all data.
- The system reserves user access level 0 (access rights) for the system supervisor functions, and levels 1 - 9 for user functions (operator access). The base system makes no distinction between access level 1 through 9.

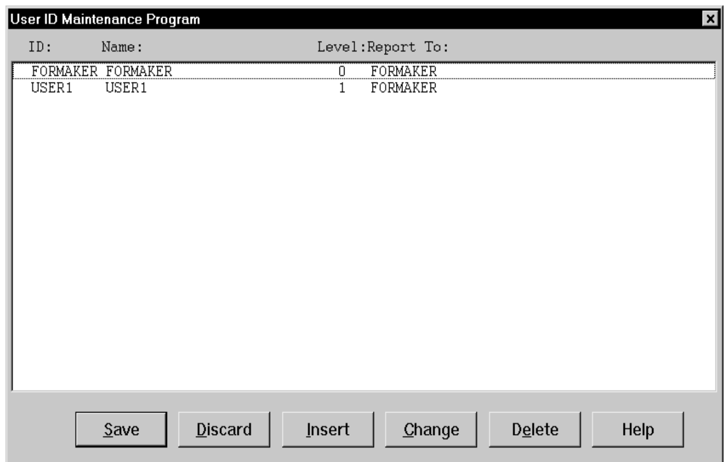
  For more information about assigning access rights, see Assigning Access Rights on page 13.

CREATING USER IDS

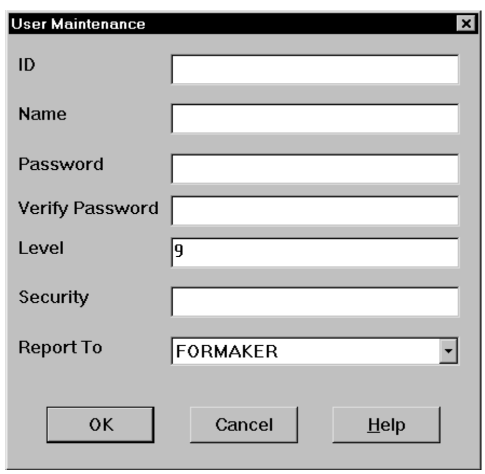
You create user IDs for all users of the system, and use the Tool, User ID Maintenance option to add, change, or delete user IDs as necessary. Log on using the *DOCUCORP* ID when you use this option, then follow these steps:

1 Choose Tools, User ID Maintenance. The User ID Maintenance Program window appears.

The system lists the users you have already set up.



2 Click Insert to add a new user. The User Maintenance window appears.



3 Type the new user's ID in the ID field then type the new user's full name in the Name field.

4 Type the user's password in the Password field. Asterisks (*) appear when you type the password. Then, type the password again in the Verify Password field. Again, asterisks appear when you type the password.

5 Type a number between 1 and 9 for the user access level in the Level field, and press TAB twice to skip the Security field (future use). Level zero (0) is reserved for the system supervisor.

6 Click the scroll arrow next to the Report To field to display a list of users. Select the user's supervisor from the list.

NOTE: Use only the *DOCUCORP* ID as the supervisor ID. When performing a batch print function, users can print only their own form sets and the form sets of users reporting to them. Having all users report to *DOCUCORP* lets the system supervisor print all users' form sets sent to the batch print queue.

7 Click Ok to add the new user information. The system returns you to the User ID Maintenance Program window.

8 Click Save to record the new user ID information. The Save Confirmation message appears.

NOTE: To revert your most recent unsaved settings to their prior status, click Discard in the User ID Maintenance Program window. The Discard button lets you make changes to a user ID, then quickly revert to the prior settings if you change your mind.

9 Click Yes to save the new user ID information.

SETTING UP SUPERUSERS

SuperUsers have all possible access rights plus the ability to...

- Search through the user IDs by name or ID
- Import new users from another database or a text file
- Reset locked user IDs

To designate a SuperUser, make sure the user has his or her access rights set to zero (0) and has the SupportSuperUser option in the UserInfo control group in either the FSIUSER.INI or FSISYS.INI file set to Yes, as shown below:

```
< UserInfo >
   SupportSuperUser = Yes
```

You should also have the Do_Logon option set as shown here:

```
< Enviroment >
   Do_Logon = Yes
```

With these settings in place, the system displays an enhanced User ID Maintenance Program window with easy access to the features listed above. This window works like the standard User ID Maintenance window except there is no Save button and there is an Import button.

You can import from a...

- Text file. With only one user per line, the text file should have the following format:

```
UserID,Name,Password,ReportTo,Level
```

- Database file.

To use the Import button, you must add the following options in your INI file:

```
< UserImportFunctions >
    01=;Text file;USRMAINT->USRImportText;
    02=;Another UserInfo database;USRMAINT->USRImportDBF;
```

The format of these lines is:

```
XX=;Description;DLLNAME->FunctionName;
```

where *XX* is the number, *Description* defines the type of file to import, *DLLNAME* is the name of the DLL that contains the function, which is named in *FunctionName*.

The import functions shown above do not have to be in the order shown, nor have the same titles, such as *Text File*.

For instructions on using the features in the SuperUser User ID Maintenance Program window, see Using SuperUser Access on page 317.

Assigning Access Rights

You can assign access rights to different system menus and functions by editing the MEN.RES file in system's DLL directory. You can edit this file using any ASCII text editor. Before you edit the file, make a backup copy and assign it a name such as MEN.OLD.

Here is a excerpt from the MEN.RES file distributed with the system:

```
POPUP "Form&set" 170 "Formset"
 BEGIN
  MENUITEM "&Close" 1069 "Close document" "NULL"
  SEPARATOR
    .
    .
    .
POPUP "&Tools" 255 "Utility Programs"
 BEGIN
  MENUITEM "&User ID Maintenance..." 501 "USRMAINT->UMUserMaint"
"User ID Maintenance" 0
  SEPARATOR
```

The first line sets the Formset option on the menu bar. The third line sets the Close option on the Formset menu. There is no access level set for this option so the system will make it available to all users.

The second Popup line sets the Tools option on the menu bar. The first menu option is User ID Maintenance. The zero (0) at the end of this line sets the access level to zero, which typically denotes system supervisors.

If you set the access level for the User ID Maintenance option to three (3), users with access levels 3, 2, 1, and 0 (zero) will see this option on the menu. Users with access level 4-9 will not see this option. You can add or change access levels as necessary to meet your company's needs.

Allowing users to modify user information

To allow users who have access rights from 1-9 to change their user information or the information for users who report to them, change this line in those users' MEN.RES file:

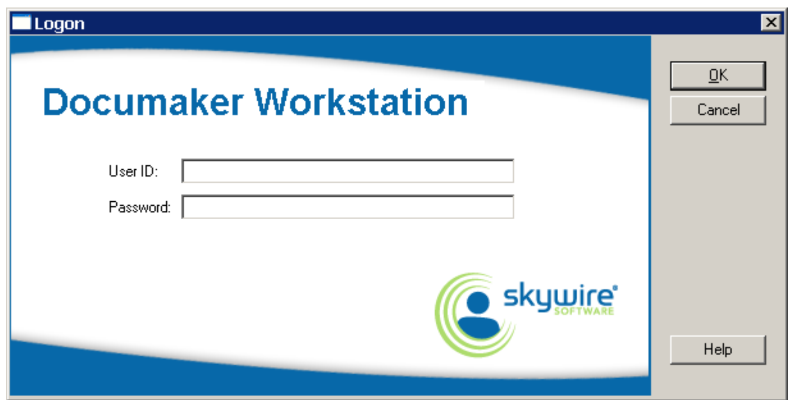
```
MENUITEM "&User ID Maintenance..." 501 "USRMAINT->UMUserMaint" "User
ID Maintenance Program" 0
```

Change the zero (0) at the end of the line to 9, or whatever level lets the User ID Maintenance option be enabled for them. For example, if managers have an access level of seven and you want managers to have access to the User ID Maintenance option, enter seven. If you leave it at zero (0), only users with that level of access can use the User ID Maintenance option.

TESTING A USER ID

Follow these steps to test a new user ID:

1 Double click the system icon. The Logon window appears.



2 Type the user ID in the User ID field. Type the user password in the Password field. Asterisks (*) appear when you type the password.

3 Click Ok or press ENTER. The main menu appears. If you cannot start the system, repeat the steps for creating user IDs to make sure you performed them correctly.

BYPASSING THE LOGON WINDOW

If you do not want to have to log on each time you start the system, you can bypass the Logon window by setting this INI option:

```
< SignOn >
    UserID  = DOCUCORP
```

The UserID option lets you record a user's ID so the user does not have to enter it each time he or she starts the system. Place this control group and option in the FSIUSER.INI file for the appropriate resource library.

NOTE: Keep in mind that while this option makes it easier to log onto the system, it also circumvents the system's security features.

Using a Windows User ID to Log On

If you are using Windows security and you want to use your Windows user ID to automatically log onto the system, do so by adding this function in the FSIUSER.INI file:

```
< AFEProcedures >
    Security = CSTW32->CSTUserFromNT
```

Microsoft does require you use the security features in Windows NT/2000. Keep in mind...

- Documaker Workstation user IDs can consist of up to 64 characters.
- Use the UserInfo database to assign Report To chains so you are not limited to your own WIP. Otherwise, you might not be able to view the WIP of subordinates.

Note: If you have overridden the login hook with custom code, you may be able to provide the Report To chain via some other mechanism. If, however, you want Documaker Workstation to handle it, use the UserInfo database.

Setting the Startup Mode

You can use command line options to have individual users start up in entry, WIP, or archive/retrieval mode. You can also use this option to set the startup mode:

```
< Control >
    EntryMode = WIP (or RETRIEVE)
```

The default is Entry. If you use the EntryMode option and omit the command line options, you will not have to reset user icons when you install an upgrade.

Note: The command line options (/mode=wip or /mode=retrieve) override this option.

Creating User Icons

After you create a user ID for each user, you can create a desktop icon for each user. The steps below assume that you installed the system on a network and no icons exist on the end user's desktop.

If you installed the system on a single user workstation, instead of a network, a Documaker Workstation program group should already exist. In this case, begin with Step 2 below, to create separate icons for each user.

Follow these instructions to create a user icon:

1 Create an icon for the system on the user's desktop. Refer to your operating system documentation for specific information.

2 Open the properties window for each system icon you create. Use the information in the table below to complete the fields in the Properties window.

System	Program name	Target directory
Windows 32-bit	ppswin\dll\afemnw32.exe	ppswin\mstrres\(resource library name)

Repeat these steps for each user.

Caching FAP Files

If your users are networked together using shared resources, you can increase workstation performance by adding the following INI option:

```
< Control >
   CacheFAPFiles = 100
```

This example tells the system to cache up to 100 frequently-used FAP files. By caching these files, users can retrieve the forms from memory instead of from across the network. You will see the greatest performance gain in situations where users are re-using the same forms with the same images. Caching is disabled by default.

Note: The number of images you cache affects available memory. If you set the cache too high, you reduce the amount of memory available for processing. Experiment with different settings to see which setting works best for you.

Chapter 2

Using Resource Libraries

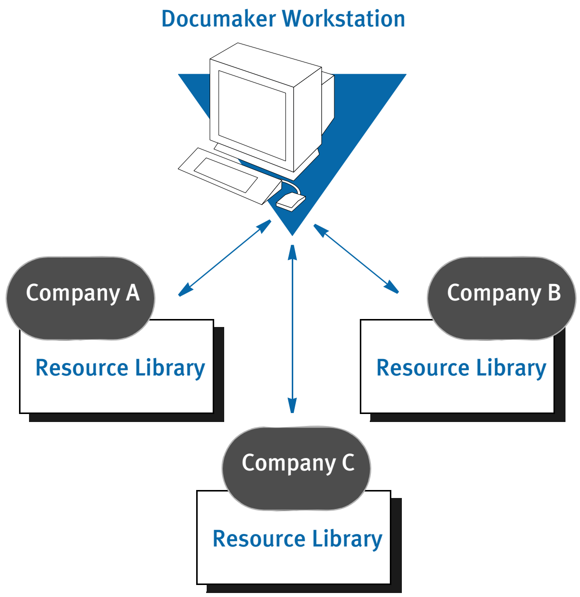
A resource library is a set of customized resources used to create and process form sets. You can use multiple resource libraries and you can create your own resource libraries using an existing library, such as SAMPCO, as a model.

This chapter discusses:

Overview

When you set up a master resource library, you tell the system where to look for necessary files and directories. You can also use these instructions to access different resource libraries or resource library components. This is advantageous if your library contains a large amount of resources.

This illustration shows a typical setup where the system accesses several resource libraries. For example, many insurance agents often use one resource library for each company for which they underwrite policies. Resource libraries let you better organize, store, and track resources.

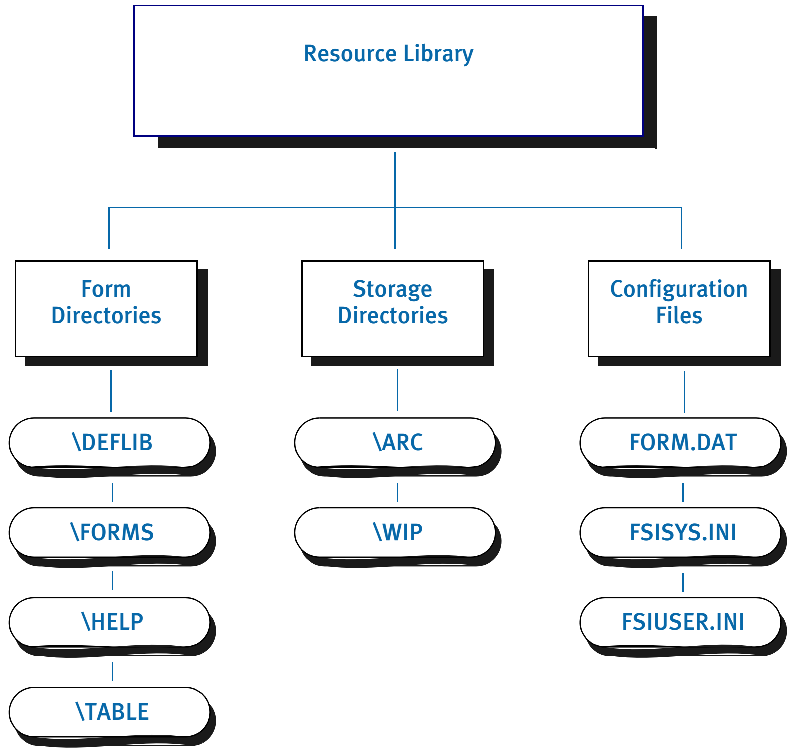


Resource Library Components

The system's directory structure contains directories and files which store and control resources. The basic building blocks of the system are form directories, storage directories, and configuration files.

Form directories store the resources the system uses to produce on-line and printable forms. Form directories include FORMS, DEFLIB, HELP, and TABLE. Storage directories are databases which contain temporary or permanently saved data files. Storage directories include WIP for temporary storage before form completion, and ARC for permanently archived data.

The configuration files, FORM.DAT, FSISYS.INI, and FSIUSER.INI, tell the system how to use resources. Here is an illustration of a resource library's major components:



Form Directories

The main component of any resource library is forms. Forms contain viewable and printable images and logos. Forms and their components are contained in these directories: DEFLIB, FORMS, HELP, and TABLE.

- DEFLIB - contains the form set definition (FORM.DAT), and all the font files used during forms creation, form set production, and form set printing.
- FORMS - stores FAP and LOG files. FAP (image) files define page layout, static fields, and variable fields. LOG files contain graphics used on forms. You create FAP files using Documaker Studio or Image Editor, or get them from an approved forms provider.
- HELP and TABLE - contain databases of help and table information. When creating forms, the forms developer attaches help and table information to specific variable fields to assist the user during form set production.

When you receive the image and related files, copy them into the appropriate resource library directory so they become part of the resource library. See Implementing Images on page 24 for more information.

Storage Directories

After users enter form data into form sets, the system provides these storage directories for the data: WIP (work-in-process) and ARC (archive).

- WIP - contains temporary form data. There are times when a user is not able to complete a form in one sitting. In these instances, the user can save form data to the WIP directory, then reopen and complete the form at a later time.
- ARC - stores data from completed and printed form sets. Users can retrieve and view archived form data, but they cannot edit it.

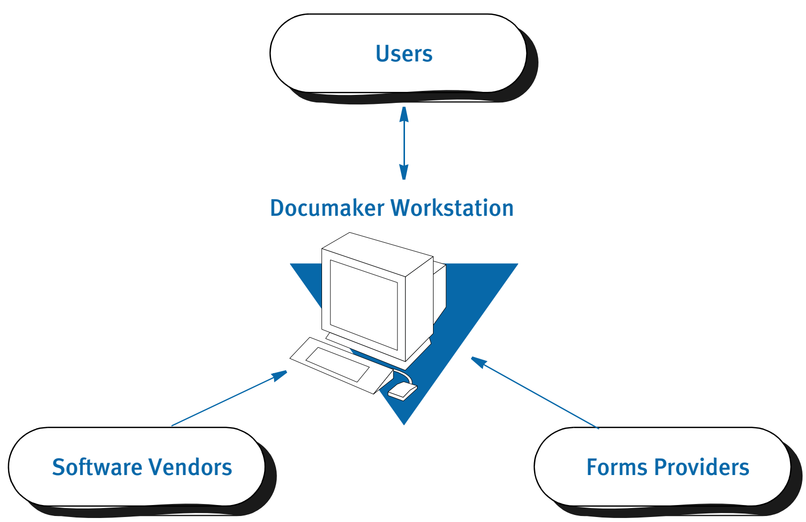
Configuration Files

You can easily configure the system to meet your company's needs using these configuration files:

- FORM.DAT (*form set definition table*) - defines the images that comprise the forms and the forms that comprise the form sets. The FORM.DAT file identifies the library to which images belong, the image groupings within a form, and the form and image options (types). The forms designer usually creates the FORM.DAT file.
- FSISYS.INI and FSIUSER.INI - lets you customize the system for your company or individual users. You can configure the INI files in each resource library to use different system options with different libraries. For example, you may want to use the data importing and exporting functions with one library, but not with others.

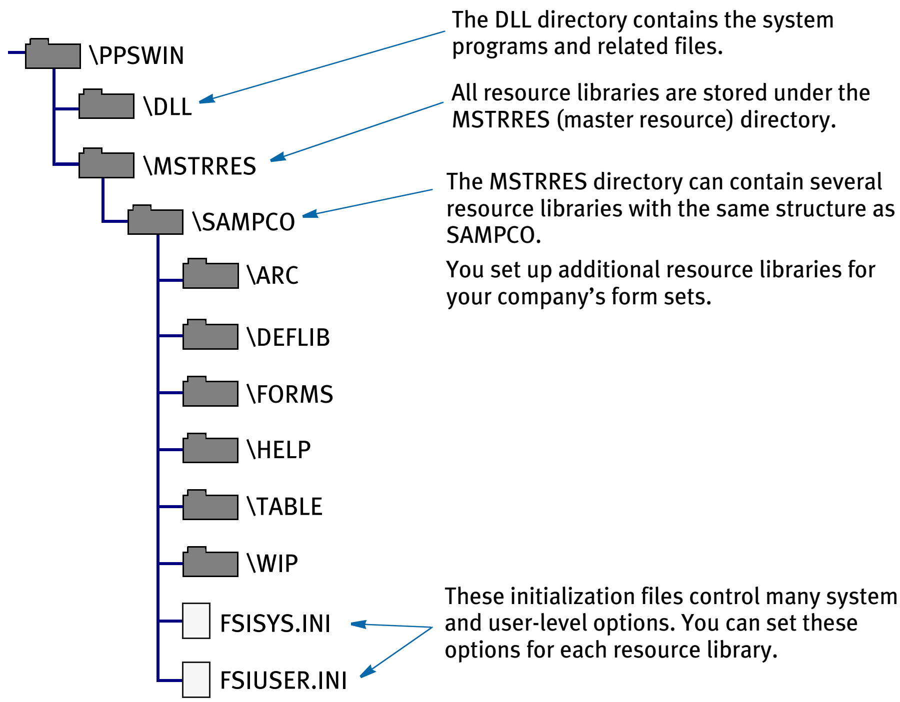
Receiving Libraries from Other Sources

If your company gets forms and libraries from outside sources, you may include their representatives in your implementation team. The implementation team shares the responsibility for providing resource files and preparing the operating environment. Usually forms providers and software vendors build the library information files necessary for the system. Users perform setup activities to prepare their environment for the libraries provided.



Building a Resource Library Directory Structure

This example illustrates the structure of the MSTRRES directory, the master resource library directory. Understanding the resource library directory structure in relation to the system should make building a resource library easier to follow.



MSTRRES Directory

MSTRRES, the *master resource library* directory, contains your custom resource libraries. You perform most of the library implementation activities in the MSTRRES directory. When you install the system, the MSTRRES directory contains the SAMPCO directory. Your MSTRRES directory may also contain other predefined resource libraries created specifically for your company.

SAMPCO Directory

The SAMPCO directory is a sample resource library. You use the files and subdirectories in SAMPCO as a model for additional form or resource libraries. A description of selected SAMPCO subdirectories and files follows:

- ARC - This directory contains an empty archive database definition file the system uses to correctly archive information users enter into the system. The directory contains the files APPIDX.DFD, CATALOG.DAT, CATALOG.MDX, and ARCHIVE.CAR. These files define all data fields in the archive database. *Do not change these files*.

- DEFLIB - This directory contains all the font files used during the forms creation process and form production. DEFLIB contains the actual font files, font cross-reference files, and a downloadable font spool file that the system sends to the printer. The system uses the font files to ensure that forms and form sets appear correctly on-line and when printed. DEFLIB also contains a special file, FORM.DAT, which specifies how individual forms are grouped into form sets.

- FORMS - This directory contains a sample forms library. Each file located in this directory stores information about a form used by the system. The FORMS directory stores the actual form and logo images that appear on screen during form production and form set printing. Forms can be created using Documaker Studio or Image Editor, or can be produced by an approved forms provider.
- HELP - This directory contains help files specific to the forms you create and store in your resource library. Help directory files consist of help database files (DBF) and help index files (MDX). These files store and track the help information attached to specific form fields during forms creation.
- TABLE - This directory contains table files specific to the forms you create and store in your resource library. Table directory files consist of table database files (DBF) and table index files (MDX). These files store and track the table information attached to specific form fields during forms creation.
- WIP - This directory stores form data a user designates as WIP. When users save forms to WIP for the first time, the system generates two database files, WIP.DBF and WIP.MDX. The system uses these files to store and track form sets saved to WIP.

Here is a description of some of the important files in a master resource directory

- FORM.DAT - This file is the *form set definition* file. FORM.DAT defines the organization of images within forms, and the organization of forms within form sets. The FORM.DAT file identifies the library to which images belong, the image groupings within a form, and the form and image options. The FORM.DAT file also specifies the recipients of each image, and the number of recipient copies. This file resides in the DEFLIB directory.
- FSISYS.INI - This file is the system configuration file for each resource library. You modify this file to enable or disable import and export functions. See Customizing Your System on page 127, for more information.
- FSIUSER.INI - This file is the company library configuration file for each resource library. You modify this file to specify how users perform certain operations such as archiving or printing within a company resource library.

Creating a Resource Library

As the system supervisor, you may be responsible for implementing resource libraries for your organization. In most cases, you use a separate resource library for each logical grouping of form sets. For instance, in the insurance market, you would probably create a resource library for each company for which you underwrite policies.

When you install the system, the MSTRRES directory contains the SAMPCO subdirectory. As defined previously, SAMPCO is a sample resource library, which you can use as a model for the additional resource libraries you create. To create a new resource library, simply create the appropriate directory structure for the new library and place the various forms and other files into the appropriate directories. You can do this using operating system commands, such as **md**, or you can use a file utility program, such as Windows Explorer.

NOTE: The examples below assume that you are familiar with basic operating system commands for creating directories and copying files. If you need additional instruction, please refer to your operating system manual.

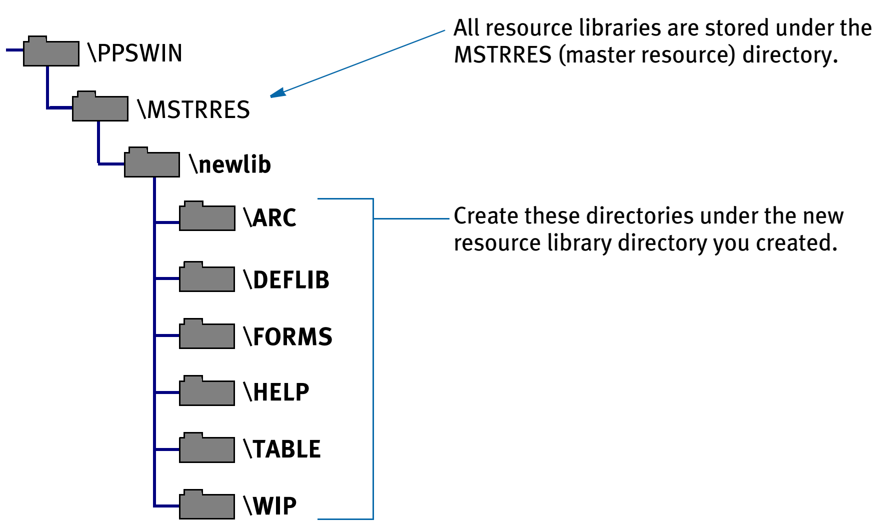
Follow these instructions to create a new resource library:

1 From the command line, go to the MSTRRES directory. For example, you could enter:

cd\ppswin\mstrres

2 From the MSTRRES directory, create a new directory for your new resource library. For example, you could enter **md newlib** to create a new resource library called *newlib*.

3 Go to your new directory and create the same subdirectories you see under the SAMPCO directory. For example, if you named your new resource library *newlib*, your new directory structure would look like this:



4 Copy the APPIDX.DFD file from the ARC subdirectory of SAMPCO to the ARC subdirectory in your new resource library. For example, you could enter:

copy c:\ppswin\mstrres\sampco\arc\ appidx.dfd c:\ppswin\mstrres\newlib\arc

5 If you received your form library from a forms provider, make sure you also received the FSISYS.INI and FSIUSER.INI files. If you did not receive the files, copy the FSISYS.INI and FSIUSER.INI files from the SAMPCO directory into your new resource library. For example, you could enter these commands:

copy c:\ppswin\mstrres\sampco\fsisys.ini c:\ppswin\mstrres\newlib
copy c:\ppswin\mstrres\sampco\fsiuser.ini c:\ppswin\mstrres\newlib

Renaming INI File Library References

If you copied the FSIUSER.INI file from the SAMPCO resource library, all references in the file point to the SAMPCO subdirectory. You must change these references so the system will know where to get the information it needs.

You can make these changes several ways:

- Using the Library Setup option
- Manually changing the references in a text editor

For information on using the Library Setup option, see Adding or Changing Libraries on page 27.

To manually change the references, use the search and replace option available in most text editors to rename the subdirectory references. You must modify all FSIUSER.INI files of each resource library you create so the system correctly accesses the resources within your resource library.

Follow these steps to manually rename INI file library references:

1 Open the FSIUSER.INI file in your new resource library using a text editor.

2 Use the search and replace feature of your text editor to change all SAMPCO references to your resource library name. For example, if you named the new resource library *newlib*, you would search for *SAMPCO* and replace with *newlib*.

This...	Changes to...
[MasterResource]	[MasterResource]
FormDef = [CONFIG:SAMPCO] FormDef =	FormDef = [CONFIG:*newlib*] FormDef =

IMPLEMENTING IMAGES

The next step in creating your resource library is to copy your resource files (images, graphics, fonts, tables, help files) to the appropriate directory. Make sure you place each resource in the appropriate directory so that the system can find them. If you are unsure where to place certain files, contact your forms provider. This table lists the resource library directories, their component file types, and a description.

Directory	File Types	Description
FORMS	.FAP files .LOG files	Files which comprise the images and graphics which appear on screen during data entry.

Directory	File Types	Description
DEFLIB	FORM.DAT	Defines the organization of images within forms, and the organization of forms within form sets.
	Font files	Font files used during image creation and printing. These can be single fonts (FNT files) or a downloadable spool file used when printing.
	.FXR files	Font cross-reference files.
HELP	.DBF files	Help database files associated with specific forms.
	.MDX files	Index file used for tracking and using help files.
TABLES	.DBF files	Table database files associated with specific forms.
	.MDX files	Index file used for tracking and using table files.

Setting Up User Icons

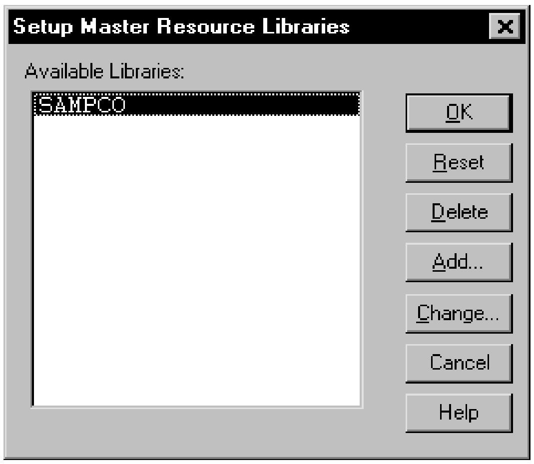
The final step in creating your resource library is to set up icons for each user. You must create a separate icon for each resource library created. See Creating User Icons on page 16, for more information.

Selecting a Resource Library

Selecting a master resource library lets you choose the resources you want to use. When you select a resource library, you configure your system setting to use the resources within the library. The default master resource library is the SAMPCO library included with the system.

NOTE: Access to this menu option depends on your user security level as specified during user ID setup.

To select a master resource library, choose File, Library Setup. The system displays the Setup Master Resource Libraries window.



Click the library you want to work with; then, click Ok. The system configures the INI files to use resources in the company library you select.

NOTE: You can use a built-in INI function (~*MRL*) which lets you set up INI options for specific master resource libraries (MRLs). This is useful if you have multiple master resource libraries and you need to customize the way the system works for each MRL. For more information, see Setting INI Options for Specific Master Resource Libraries on page 128.

Using the Library Setup Option

If you modified the FSIUSER.INI file when you built your resource library, you do not need to use the File, Library Setup option at this time. However, you can use this option if you later choose to change directory paths or file names.

By using the File, Library Setup option, you can tell the system how to locate and use the resources contained in your resource library. This option lets you modify the FSIUSER.INI file.

With the Library Setup option, you can add or remove a library, or change the directory paths and file settings for a library. Access to this option depends on the security level assigned to individual users during user ID setup.

NOTE: The Library Setup option is available only when all images and forms are closed.

Adding or Changing Libraries

Using the library setup options you can add a customized resource library or change the existing library path and file settings.

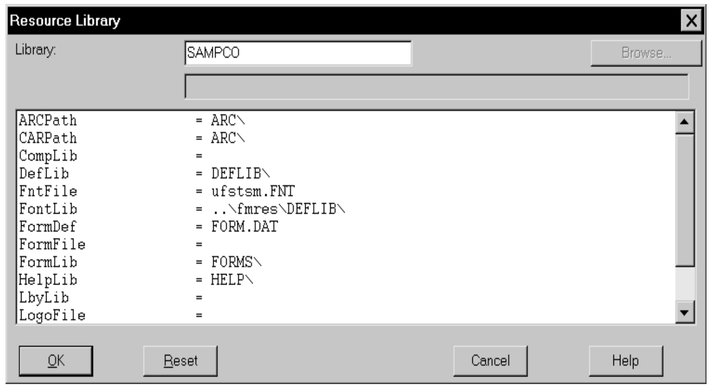
To add or change a library, follow these steps:

1 Choose File, Library Setup. The Setup Master Resource Libraries window appears.

2 Click the library you want to work with; then, do one of the following:

To	Do This
Create a library	Click Add.
Change a library	Click the library you want to change; then click Change.

The Resource Library window appears. Make sure the directories correspond with your selected library. If not, repeat the task of selecting a library.



NOTE: If you click Reset after you change or add library files and subdirectories, your settings revert to their prior status and you lose all changes.

The Resource Library window contains the following fields:

Field	Description
Library	Name of the current library
ARCPath	Directory path of the archived files
CARPath	Directory path of the compressed archived files
CompLib	Directory path of the compiled FAP files
DefLib	Directory path of the library fonts. Depending on your system customization, your fonts may be alternately located in the FontLib directory.
FntFile	Name of the downloadable font spool file
FontLib	Directory path of the specific font set. Depending on your system customization, your fonts may be located in the DefLib directory.
FormDef	Name of the FORM.DAT *(form set definition table*) file
FormFile	Name of the Library Manager (.LBY) file
FormLib	Directory path of the library forms
HelpLib	Directory path of the form related help information
LbyLib	Directory path of the Library Manager files (*.LBY)
LogoFile	Name of the logo Library Manager file
TableLib	Directory path of the form related table information
WIPPath	Directory path of the Work-in-Process files

Field	Description
XrfFile	Name of the font cross-reference file

3 Type the name of your new library, or change the library name in the Library field.

NOTE: If you know the various directories and file names you want to change, you can click the applicable field and type the new path or file name in the entry field. *If you rename files or directories within the window, you must also rename the files or directories in your operating system.*

4 Click the library path or file you want to add or change; then, click the Browse button. The Select Location window or the Select File window appears, depending on your field selection.

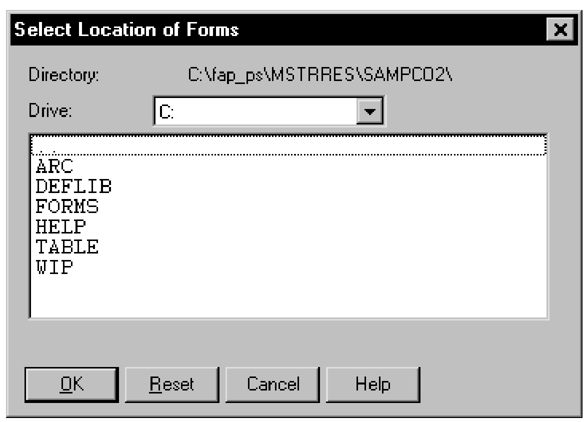
To add or change directory information, follow the instructions below. If adding or changing file information, see Adding or Changing File Information on page 30.

Adding or Changing Directory Information

Follow these steps to add or change directory information:

1 Select the drive where the system stores the files you want to use or change.

Double click these dots to display the previous subdirectory in this path.



2 Double click the dots in the window to display the previous subdirectory within the current directory path. Each double click moves you one step back in the directory hierarchy.

3 Click the appropriate directory in the list; then, click Ok to accept your changes to the directory path. The system returns you to the Resource Library window.

NOTE: If you need to change your most recent unsaved settings to their prior status, click Reset in any resource library window. Reset lets you quickly revert your changes on a particular window to their prior settings.

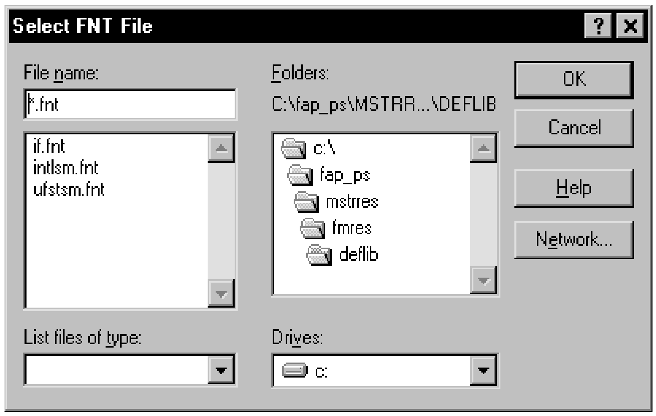
4 Click Ok in the Resource Library window to file your master resource library changes. The system returns you to the Setup Master Resource Libraries window.

5 Click Ok in the Setup Master Resource Libraries window to save your additions or changes.

Adding or Changing File Information

Follow these steps to add or change file information:

1 Select the drive where the system stores the files you want to use or change.



2 Select a new directory in the Folders field if you want to store your master resource library in a different directory.

3 Enter the name of the file you want to include in your master resource library in the File Name field or select it from the File list.

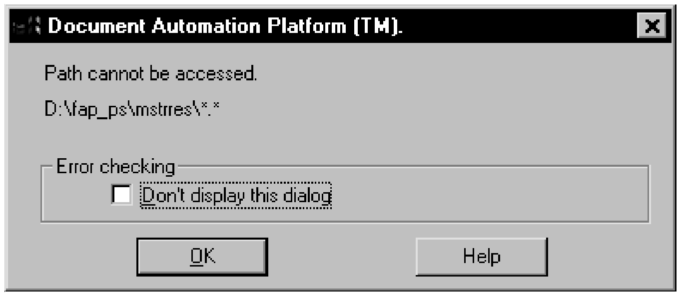
4 Click Ok to save file additions or changes to your master resource library. The system returns you to the Resource Library window.

5 When you complete making changes or additions to your master resource library files, click Ok in the Resource Library window. The system returns you to the Setup Master Resource Libraries window.

6 Click Ok to save your changes and close the Setup Master Resource Libraries window.

Control Path Error Checking

This option tells the system to recognize new paths when you set up resource libraries and alert you when you have defined a non-existent path. The system displays an error message with an option to turn off the error message display.



Click Ok to continue displaying the error message when you add an undefined path to your master resource library. Click the field; then, click Ok to disable the message and close the window. The system lets you define new paths during library setup without displaying the message.

You can make the directories for the new paths after you have set up the master resource library.

NOTE: The error window becomes active again after you exit the Resource Library window.

Deleting a Resource Library

Deleting a library lets you delete a resource library from your available libraries list. By deleting a library, you do not delete the library resources from the system; you merely remove those grouped resources from the option list.

1 To delete a library, choose File, Library Setup. The system displays the Setup Master Resource Libraries window.

2 Click the resource library you want to delete, then click Delete. The system displays the Delete Confirmation window.

3 Click *Yes* to delete the resource library or *No* to cancel the action. The system returns you to the Setup Master Resource Libraries window.

4 Click the resource library you want to continue working with; then, click Ok. You can no longer use the removed resource library.

NOTE: If you need to add a library that's been deleted from your available libraries list, see Adding or Changing Libraries on page 27.

Chapter 3

Setting Up Printers

The system supports printing on a variety of printers ranging from network laser printers to high volume production printers. This chapter describes how to set up the system to print on this wide array of printers.

In this chapter you will find information on the following topics:

For each type of printer, this chapter discusses set up issues, printer resources, special features, performance considerations, troubleshooting, and more.

AFP Printers

IBM created the Advanced Function Printing (AFP) language. The data streams produced by Skywire Software applications for AFP printers are called *Mixed Object Document Content Architecture* (MO:DCA) data streams. MO:DCA data streams are sometimes referred to as AFP data streams (AFPDS).

You must have a program such as IBM's Print Services Facility (PSF) to convert AFP data stream into the printer's native language. PSF is the umbrella software that brings the AFP resources (created by AFP or system utilities) together in one print job and sends it to the printer.

NOTE: All system print drivers support 24-bit color graphics. If you printer does not support color, the print driver will automatically convert the color graphics into monochrome graphics. Keep in mind that for the best performance you should avoid color graphics.

AFP INI Options

You define the necessary printer options for the system to produce AFP data streams. These options specify how the system creates AFP output. Most of the AFP-related options are found in a PrtType:XXX control group, where *XXX* indicates the different printer types. *PrtType:AFP* is a common control group name used to contain AFP settings. The most common AFP printer options are shown below (default values are bold):

Option	Values	Description
Device	Any file or device name	The name of the file or device (LPT1) where the AFP data stream should be written. This setting is ignored by the GenPrint program but is used by Documaker Studio and other system programs.
Module	AFPPRT	The name of the program module which contains the system's AFP print driver. See also the discussion of the Class option. See also Using defaults for the Module and PrintFunc options on page 38.
PrintFunc	AFPPrint	The name of the program function that is the main entry point into the system's AFP print driver. See also Using defaults for the Module and PrintFunc options on page 38.
Resolution	240/300	The dots per inch (dpi) resolution of the printer which receives the AFP data stream
SendOverlays	Yes/No	Set to Yes if you created AFP overlays for each FAP file

Option	Values	Description
ChartResolution	120/150/ 240/300	Used when printing charts as inline bitmap graphics on an AFP printer that does not have graphics (GOCA) support. Defaults to one-half of the Resolution option setting.
LandscapeSupport	Yes/**No**	Although not required for printing, you can set this option to *Yes* if your printer supports landscape medium maps. Generally, AFP printers using cut-sheet paper *do not* support landscape medium maps.
SplitText	Yes/**No**	Used to minimize the print differences between 240 and 300 dpi printing.
SplitPercent	0 to 100 (**50**)	Percentage of the width of the space character used to determine when the rounding error between 240 and 300 dpi printing has caused a significant difference and the text string should be split into smaller strings.
FudgeWidth	any number (**0**)	Can be used when building page overlays for sections smaller than a page.
GraphicSupport	0, 1, 2, **3**	0 = no graphics (GOCA) support 1 = inline bitmap graphics support 2 = GOCA charts support 3 = inline bitmap graphics and GOCA charts support
PageNumbers	Yes/**No**	Set to Yes to turn on form or form set page numbering
PrintViewOnly	Yes/**No**	If set to Yes, the view only sections will print. This does not apply to entry only sections, which are never printed. Entry only sections are usually worksheets. If the section is marked as hidden and view only, it will not print.
PrePrintedPaper	Yes,Disabled	Determines if the check box which lets you print or not print pre-printed objects appears on the Print window. Also determines the default for this check box—checked or unchecked. You must add this option to the INI file if you want the check box to appear on the Print window. The default for this option includes the checkbox on the Print window and leaves it unchecked. All objects except fields can be designated as pre-printed on the object's Properties window.

Option	Values	Description
Class	(*first three characters of the Module option*)	Specifies the printer classification, such as AFP, PCL, XER, PST, or GDI. If you omit this option, the system defaults to the first three letters from the Module option. Some internal functions expect a certain type of printer. For instance, all 2-up functions require an AFP printer. The internal functions check the Class option to make sure the correct printer is available before continuing.
OnDemandScript		Use this option to add comments to the print stream. This lets you handle archiving using OnDemand. Enter the name of the DAL script you want the system to run. This DAL script creates the On Demand records and adds them as comments. The AddComment function is also used in DAL scripts to add OnDemand command records. For more information about this and other functions, see the DAL Reference.
TLEScript		Enter the name of the DAL script to execute to add Tagged Language Element (TLE) records to the print stream. See Adding TLE Records on page 123 for more information.
TLESeparator		Enter the character you want to use to separate the key and value portions of the TLE comment string.
TLEEveryPage	Yes/**No**	Optional. If you enter Yes, the TLE DAL script will be executed at the start of every page. If you enter No, the TLE DAL script is executed at the start of every form set. The default is No.
PaperSize	**0**, 1, 2, 3, 98	Use this option to set a default paper size when converting AFP print streams using the Internet Document Server or the MRG2FAP utility. Enter zero (0) for letter size (default) Enter1 for legal size Enter2 for A4 size Enter3 for executive size Enter 98 for a custom size

Option	Values	Description
DocusaveScript		Use this option to add comments to the print stream. This lets you handle archiving using Docusave. Enter the name of the DAL script you want the system to run. This DAL script creates the Docusave records and adds them as comments.
SendColor	Yes/**No**	Enter Yes to send color information to the printer. AFP highlight color printing on printers from Xerox and Oce is supported. Make sure the objects you want to print in color (text, lines, shades, and so on) are set to print in color. The Print in Color option is on the Color Selection window. You can display this window by clicking the Color button on the object's Properties window.
NamedColors		Use this option to tell the system to use only specific AFP named colors. For example, if you wanted all highlight (non-black) colors mapped to blue, you would set the NamedColors option to *blue*. To allow the mapping of the colors you assigned to the objects in the FAP file to multiple colors, separate each color with a semicolon (;). For example, to use red, blue, and magenta, set the NamedColors option as shown here: `NamedColors = red;blue;magenta` The order you list the colors does not matter.
SkipChartColorChange	Yes/**No**	Enter Yes to suppress color changes normally done to enhance 3D bar charts.
SuppressLogoUnload	Yes/**No**	Enter Yes to suppress the unloading of logo (LOG) files during a conversion of AFP files to FAP (or PDF) format.. The default is No.
DisplayCodedFont	**Yes**/No	Enter No to include the character set/code page combinations in the AFP font list, instead of the coded fonts. The defaults is Yes, which tells the system to include the coded fonts. See Outputting character set and code page information on page 39 for more information.

There are some additional options you can use to print inline graphics (LOG files). Be aware that not all AFP printers support these settings. You'll find these options in the AFP control group.

AFP Options	Values	Description
OutputHalfRes	Yes/**No**	Scales the bitmap loaded from the graphic to half resolution in memory before writing the output.
DoubleOutputRes	Yes/**No**	Does not change the bitmap loaded from the graphic, but would tell the printer to double its resolution when printed. This lets the system load graphics that are half resolution already.
SuppressZeroData	Yes/**No**	Suppresses data containing a series of zeros (white space in the bitmap).
TrimWhiteSpace	Yes/**No**	Suppresses data containing zeros (white space) at the right edge of the bitmap.
MultiLinesPerCommand	Yes/**No**	Tries to combine AFP commands into fewer records when printing the bitmap. You cannot use this option with the SuppressZeroData option.

Using defaults for the Module and PrintFunc options

Default values for the Module and PrintFunc options in the PrtType:xxx control group are provided when you use a standard print type name or print class, such as AFP, PCL, PDF, PST, VPP, XER, XMP, or GDI.

These defaults keep you from having to enter the Module and PrintFunc names in your INI file. For example, if you want to generate AFP print files, you can specify these INI options:

```
< Printer >
    PrtType     = MYAFP
< PrtType:MYAFP >
    Class       = AFP
```

And the system will default these options for you:

```
< PrtType:MYAFP >
    Module      = AFPPRT
    PrintFunc   = AFPPrint
```

Printing highlight colors

The system supports AFP highlight color printing on printers from Xerox and Oce. Like other color printer support, the SendColor option must be set to Yes and the objects, such as text, lines, and shades must be set to *Print In Color*.

The RGB (red,green,blue) color setting for each FAP object is mapped to the closest AFP named color. The names of the available colors are as follows: blue, red, magenta, green, cyan, yellow, dark_blue, orange, purple, dark_green, dark_cyan, mustard, gray, and brown.

You use the NamedColors option in the AFP printer group to specify certain AFP named colors. For example, if you wanted all FAP (non-black) colors to be mapped to brown, you would use this INI option:

```
NamedColors = brown
```

To let the system map FAP colors to multiple colors, separate each color with a semicolon (;). For example, to use all of the default AFP named colors except brown, you would use this INI option:

```
NamedColors = Red;Blue;Magenta;Green;Cyan;Yellow
```

NOTE: The order in which you name the colors does not matter. In addition, the LOG2PSEG and FAP2OVL utilities include a /C=color parameter, where color is the one of the named AFP colors.

Character set and code page font information

When loading AFP, the system uses the information in the Character Set and Code Page Font fields in the FXR file instead of using the font information contained in the IBMXREF.TBL.

The AFP loader expects the AFP file's Map Coded Font (MCF) structured fields to contain references to AFP coded fonts. However, MCF structured fields can contain character set and code page information instead of the coded font information the FXR file requires.

Before version 11.2, for MCF structured fields that contained character set and code page information instead of coded fonts, you had to manually set up the IBMXREF.TBL file to resolve the character set/code page information to coded fonts in the FXR file.

Since the system includes character set and code page information in the FXR file, the AFP loader first checks the FXR file for this information and, if it exists, uses it. If the information does not exist, the AFP loader loads the information from the IBMXREF.TBL file.

Outputting character set and code page information

You can output the AFP character set and code page combination instead of the coded font in the font list when you generate normalized AFP files. If you want the character set/code page combinations to be output in the AFP font list, instead of the coded fonts, you must add the DisplayCodedFont option, as shown here:

```
< PrtType:AFP >
    DisplayCodedFont = No
```

Keep in mind the FXR file must contain the character set and code page entries in the AFP font record for this option to work. If you set the INI option to No and the character set and code page entries are not in the FXR file, the font list in the AFP file will contain only the coded fonts.

NOTE: The AFP output record can only contain *either* coded fonts *or* character set/code page entries — it cannot contain a combination. It will default to coded fonts for all if the font for one or more objects does not contain character set/code page entries.

AFP Printer Resources

FormDef

The system uses copy groups from its own FormDef named *F1FMMST.DAT*. Each copy group in a FormDef contains information about paper size, duplex, tray selection, jog, orientation, and so on. The FormDef must be available to PSF to print AFP data streams. You can use the AFPFMDEF utility to create or modify the FormDef.

Fonts

AFP fonts are designed solely for AFP printers. For more information about fonts, see Working with Fonts on page 157. In IBM AFP terminology, a font is described by three components:

Coded Font. A coded font file contains references to specific character set and specific code page. Coded font files always begin with the letter *X*, such as *X0DATIN8*.

Code Page. In IBM AFP terminology, a code page file maps code points to an AFP character name in a character set file. Code page files always begin with the letter *T*, such as *T1DOC037*.

Character Set. A character set file contains the bitmap graphic of each character in the character set. Character set files always begins with the letter *C* (such as *C0FATIN8.240* or *C0FATIN8.300*). The character set file name extension (*240* or *300*) indicates whether the bitmap graphics are drawn at 240 or 300 dots per inch.

Monotype fonts

Skywire Software has licensed for use and distribution with its systems, fonts from Monotype Imaging, Inc. The system includes both 240 and 300 dpi AFP fonts.

Overlays

Use the FAP2OVL utility to create AFP overlays from FAP files. The OVLCOMP utility also lets you create AFP overlays from FAP files. These overlays must be available to PSF to print AFP data streams when the SendOverlays option is set to Yes.

Page segments

Use the LOG2PSEG utility to create AFP page segments from graphics (LOG files). These page segments must be available to PSF to print AFP data streams.

Note: For information on system utilities, see the Docutoolbox Reference.

AFP 2-up support

The system include rules you can use to generate and merge print streams for AFP printing for printers that support 2-up printing. See Handling 2-up Printing on page 60 in the Documaker Server System Reference for more information.

AFP Troubleshooting

Floating section limitations

The system lets you compose a page from several sections. The system also lets you create overlays for these sections. There is one limitation when you print these sections on a landscape page. Overlays on a landscape page can only be placed vertically on the page. Overlays on a landscape page *cannot* be placed horizontally on the page.

This means, in your SetOrigin rule, you *cannot* specify any non-zero, positive number for the X-relative displacement. Create your FAP files accordingly, but keep in mind that they can be moved down but not across. This limitation exists only for AFP overlays, and only in landscape mode.

Objects extending beyond the edges

Another type of error can occur if the overlay for a custom-sized section is too small for the objects (text, lines, graphics, and so on) contained within it. If the AFP overlay's page size is too small, objects may be clipped to the page size, printed as solid black rectangles, or trigger error messages.

Documaker Studio and Image Editor offer an Auto-size option which you can use to make sure the custom-sized section is large enough to contain all objects placed within it. Use this feature to prevent most custom page size problems.

Be careful placing text at the extreme left edge of the section because it may cause errors that the Auto-size option cannot detect. For instance, suppose you have this text label positioned on the left edge of the FAP file (left offset = 0):

Beneficiary

When printed, black rectangles or an error message may appear instead of the text.

This can occur because some of the characters in the italic font (Times New Roman) have a *negative left offset*. This means that the characters print to the left of where they would normally start. A negative left offset may be easier to understand by looking at these characters:

ef

Notice how the bottom of the *f* goes under the *e*. This is an example of a negative left offset. Because it is positioned to the left of where it would normally start, the character is now positioned off the left edge of the overlay.

This kind of detailed character information is not stored in the FXR file so Documaker Studio and Image Editor have no way to know there may be a problem. You can, however, move the text labels in the FAP file to correct the problem.

Conflicts between page and form orientation

If you create a custom-sized page, be aware of any conflict between page orientation and the form orientation. If the form orientation is not the same as the page orientation, the page will not print according to the page orientation, but will follow the form's orientation.

NOTE: This happens only in case of custom size pages. Standard size pages obey the page orientation.

Multi-page FAP limitation

There is a problem when a landscape, multi-page FAP has different page sizes on each page. All pages of a multi-page FAP file should be the same size. As a workaround, use Documaker Studio or Image Editor to correct the page sizes. After saving the FAP file, you can then generate proper AFP overlays.

Printing rotated variable fields

Here is a list of field options you can specify in the NAFILE.DAT file:

Option	Description
E	Error
M	Manual
P	Protected
G	Global scope (entire form set)
F	Form scope
H	Hidden field – a dummy field, not displayed or printed
N	Nonprintable field (displayed, not printed)
C	Send-copy-to field (receives current recipient name at print time)
9	Rotated 90 degrees
8	Rotated 180 degrees
7	Rotated 270 degrees

Some of these options require the FAP field attributes to be available at runtime, since the DDT file does not include the necessary information. Use the CheckImageLoaded rule to make sure this information is available.

AFP 240 dpi print problems

Due to differences in resolution on 240 and 300 dpi printers, a text string may print with slightly different lengths. One example where this may be noticeable is when the text is printed inside of a boxed region. Another example where this may be noticeable is when a text area contains an embedded variable field.

To minimize the print differences between 240 and 300 dpi printing, use the SplitText option. Make sure these options are in your printer PrtType:xxx control group:

```
< PrtType:AFP >
   SplitText    = Yes/No (default is No)
   SplitPercent = ###    (% of space-width as max rounding error)
   Resolution   = ###    (default is 300)
```

If you set the SplitText option to Yes, each text string is checked to see if it needs to be split into sections for printing. The SplitPercent value helps determine when a text string must be split into sections for printing.

The SplitPercent option sets the percentage of the width of the space character to use as the maximum amount of rounding error that can accumulate in a string before it is broken into sections.

The SplitPercent value is from zero (0) to 100. Do not enter a value greater than 100. For example, if you set the SplitPercent option to *75*, the string is broken into sections if the accumulated rounding error is greater than 75% of the width of the space character. This value is set to 50 by default.

NOTE: Using *50* as the SplitPercent value is a good trade-off between the appearance and the performance impact on the GenPrint program and print spool size. Setting the SplitPercent option to a smaller value gives you a more accurate printout but slows the GenPrint program, increases the size of the print spool, and increases the amount of time it takes to print.

The Resolution option determines the rounding error. Most FXRs are built using 300 dpi fonts. This causes rounding errors when the FXR is used for printing to a 240 dpi printer. If you omit the Resolution option, the system uses the default setting of 300.

You need to know whether the FXR you are using was built by importing 300 dpi fonts or 240 dpi fonts. The standard FXRs are built using 300 dpi fonts. When an FXR is built using 300 dpi fonts, there are rounding errors when printing to a 240 dpi printer.

Here are some examples of options to use in different situations:

- If your font cross-reference (FXR) file was built from 300 dpi fonts and your printer resolution is 240 dpi, set the options as shown here:

```
< PrtType:AFP >
   SplitText    = Yes
   SplitPercent = 50
   Resolution   = 240
```

- If your font cross-reference file was built from 240 dpi fonts and your printer resolution is 300 dpi, set the options as shown here:

```
< PrtType:AFP >
   SplitText    = Yes
   SplitPercent = 50
   Resolution   = 300
```

- If your font cross-reference file was built from 300 dpi fonts and your printer resolution is 300 dpi, you do not need to set the SplitText option.
- If your font cross-reference file was built from 240 dpi fonts and your printer resolution is 240 dpi, you do not need to set the SplitText option.

Including Documerge Form-level Comment Records

You can include Documerge form-level comments in AFP print streams produced by Documaker. You may want to include form-level comments if you have a reprint utility program that needs information about a form before it can reprint it.

To include form-level comment records, add the FormNameCR option in your AFP printer control group and set it to Yes, as shown here:

```
< PrtType:AFP >
    FormNameCR      = Yes
    Module          = AFPPRT
    PrintFunc       = AFPPrint
    SendOverlays    = Yes,Enabled
    ...
```

Here is an example of the AFP records in an AFP print stream which includes the Documerge form level comment (NOP) records:

```
000,Begin,Document,29,
001, Data,NOP,84,%%%DMGFORMBEG%%% DEC PAGE                    00001
AFP  Docucorp 000001
002, Map,Medium Map,16,PLUD
...
033, End,Page,16,
034, Data,NOP,84,%%%DMGFORMEND%%% DEC PAGE                    00001
AFP  Docucorp 000001
035, Data,NOP,84,%%%DMGFORMBEG%%% LETTER                      00001
AFP  Docucorp 000002
036, Begin,Page,16,
...
053, End,Page,16,
054, Data,NOP,84,%%%DMGFORMEND%%% LETTER                      00001
AFP  Docucorp 000002
173,End,Document,16,
000,Begin,Document,29,
001, Data,NOP,84,%%%DMGFORMBEG%%% OP714                       00001
AFP  Docucorp 000001
002, Map,Medium Map,16,PLUO
...
```

Metacode Printers

The Metacode language is the native mode language for Xerox 4000 and 9000 series printers. This language is superior to printing using line data with Xerox Laser Printing Systems (LPS). The advantages of using Metacode over line data printing include support for portrait and landscape text on the same page, support for different fonts on the same line, precise text positioning, and text justification. In addition, Metacode lets you merge multiple forms onto a single page.

Note: All system print drivers support 24-bit color graphics. If you printer does not support color, the print driver will automatically convert the color graphics into monochrome graphics. Keep in mind that for the best performance you should avoid color graphics.

Required JSL INI Options

The system does not require you to use a special JSL on your printer to print its Metacode output. The Xerox Metacode printer driver is configurable based on options to produce Metacode which match your existing JSL settings. Here is an example of the PrtType:XER control group which contains these options:

```
< PrtType:XER >
    DJDEIden        = A'@@@DJDE'
    DJDEOffset      = 0
    DJDESkip        = 8
    OutMode         = BARR
    ImageOpt        = No
    CompressMode    = LIN
    JDEName         = META
    JDLCode         = NONE
    JDLData         = 0,255
    JDLHost         = IBMONL
    JDLName         = CBA
    PaperSize       = 0
    Device          = dummy.txt
    RelativeScan    = Yes
```

Several of these options are based on the comparable parameter values in the settings of the printer's JSL. A JSL may contain many JDLs from which to choose, or there may be multiple JSLs compiled into multiple JDLs.

A portion of a JDL may look like the following:

```
CBA:     JDL;
T1:      TABLE        CONSTANT=X'121212121212121212';
T2:      TABLE        CONSTANT=X'131313131313131313';
T3:      TABLE        CONSTANT=X'FFFF26FFFF';
C1:      CRITERIA     CONSTANT=(0,9,EQ,T1);
C2:      CRITERIA     CONSTANT=(0,10,EQ,T2);
C3:      CRITERIA     CONSTANT=(1,5,EQ,T3);
VOLUME                HOST=IBMONL;
LINE                  DATA=(0,255);
IDEN                  PRE=A'@@@DJDE',
                      OFF=0,
                      SKIP=8;
ROFFSET               TEST=C1;
RSTACK                TEST=C2,DELIMITER=YES,PRINT=NONE;
RPAGE   TEST=C3,SIDE=NUFRONT,WHEN=NOW;

/* 8.5 x 11 job */
USA1: JDE;            /* JOB can be used in place of JDE  */
OUTPUT                PAPERSIZE=USLETTER;

/* 8.5 x 14 job */
META: JOB;
VOLUME                CODE=NONE

/* Default job */
DFLT: JDE;
VOLUME                CODE=EBCDIC
END;
```

Here are the required options which are based on settings in the printer's JSL file.

JDLName Represents the name of the JDL to use. The following table shows the relevant JSL statement for the earlier example and the proper option to use based on the JSL example.

JSL statement	CBA: JDL;
INI option	JDLName = CBA

JDEName Represents the name of the job to use. A JDL may contain many jobs (JDEs) from which to choose. This JDE must contain a *VOLUME CODE=NONE* statement. The following table shows the relevant JSL statements for the earlier example and the proper option to use based on the JSL example.

JSL statements	META: JOB; VOLUME CODE=NONE
INI option	JDEName = META

DJDEIden, DJDEOffset, and DJDESkip

Represent the IDEN statement of the JDL. The value of the DJDEIden setting is a string constant. The types of string constants supported are ASCII (A'string'), EBCDIC (E'string'), Character ('string'), and Hex (X'string'). Octal, H2, and H6 strings *are not* supported.

Strings containing repeat counts, embedded hex values, and upper/lower case toggles *are not* supported. The following table shows the relevant JSL statements for the earlier example and the options to use based on the JSL example.

JSL statements	IDEN PRE=A'@@@DJDE', OFF=0, SKIP=8;
INI options	DJDEIden = A'@@@DJDE' DJDEOffset = 0 DJDESkip = 8

JDLCode

Represents the type of input format expected by the Xerox printer. Character translation occurs as necessary. Currently, the supported code types are *EBCDIC*, *ASCII*, *NONE* (same as ASCII), *BCD*, *H2BCD*, *H6BCD*, *IBMBCD*, and *PEBCDIC*. User-defined code translations are not supported.

Referring to the sample JSL, if the printer is normally started with STA DLFT,CBA then the JDLCode option must be set to *CODE = EBCDIC*. The system's option must contain the value of the CODE statement for the printer's normal operation. This table shows the relevant JSL statements for the earlier example and the proper option to use based on the JSL example.

JSL statements	DFLT: JDE; VOLUME CODE=EBCDIC
INI option	JDLCode = EBCDIC

JDLData

Represents the starting position and length of the print line data within an input data record. The LINE statement contains a DATA entry that holds these values. This table shows the relevant JSL statement for the earlier example and the proper option to use based on the JSL example.

JSL statement	LINE DATA=(0,255);
INI option	JDLData = 0,255

JDLHost

Represents whether the printer is normally in an on-line or off-line state. Currently, the only values we accept for this option are IBMONL (on-line) and IBMOS (off-line). The following table shows the relevant JSL statement for the earlier example and the proper option to use based on the JSL example.

JSL statement	VOLUME HOST=IBMONL;
INI option	JDLHost = IBMONL

Additional Required INI Options

Below are the additional required options not based on the printer's JSL file.

OutMode

The OutMode option indicates the output format for the Metacode data stream generated by Skywire Software applications.

Use *BARR*, if the Metacode output is to be transmitted to the Xerox printer via BARR SPOOL hardware and software. When using the BARR setting, a length byte is placed at the start and end of each Metacode record.

Use *BARRWORD*, if the Metacode output is to be transmitted to the Xerox printer via BARR SPOOL hardware and software. BARRWORD should be used *only* if the Xerox printer can handle records longer than the 255 characters.

Use *PCO*, if the output is transmitted to the Xerox printer via PCO hardware and software (from Prism). When using the PCO setting, a 4-byte length field is placed at the start of each Metacode record.

NOTE: The PCO interface has not been tested, but should work.

Use *JES2*, if the Environment option is set to MVS.

Use *MRG4*, if you will transmit the Metacode output to the mainframe using Commcommander or if you will archive it in Docusave (see Creating Print Streams for Docusave on page 119 for more information).

Use *LAN4235*, if the output is generated for a Xerox 4235 printer attached to a network.

Here is an example:

```
OutMode = BARR
```

ImageOpt

The ImageOpt option specifies if the graphics are being saved on Xerox printer as *IMG* files or as *FNT* files. To use IMG files, the printer needs a special GVG or GHO hardware installed. Also, in the JSL you have to specify GRAPHICS = YES. If you are using IMG files, set the ImageOpt option to *Yes*; otherwise set it to *No*. Also, if you want to print charts, you must set the ImageOpt option to *Yes*. Metacode printers have a limit of sixteen (16) images on a page. Here is an example:

```
ImageOpt = No
```

CompressMode

The CompressMode option compresses bitmaps output as inline graphics, such as charts and graphics with the inline graphics flag set. There are four compression modes available, which you can specify using the CompressMode option in the PrtType:XER control group:

- CompressMode = UNC
- CompressMode = ENC
- CompressMode = HTN
- CompressMode = LIN

UNC is the uncompressed or raw bitmap mode. If none is specified, the system defaults to *HTN* mode.

To demonstrate the effects of Metacode graphics compression, the following chart shows the GenPrint program run times and file sizes with the different compression options for a test environment containing in-line images.

Test	GenPrint time	File size
No charts (ImageOpt=No)	182 seconds (3:02)	697,599
UNC – uncompressed	309 seconds (5:09)	9,011,058
LIN compression	290 seconds (4:50)	1,589,226
ENC compression	301 seconds (5:01)	2,248,302
HTN compression	296 seconds (4:56)	1,831,050

Which compression method yields the smallest file size or the quickest compression time depends on the graphic bitmaps you are printing. In general, HTN or LIN compression provides the best results. HTN generally does best with graphics which contain more filled-in or shaded areas, while LIN performs better with graphics which contain more line art. Experiment with your sections to determine the best compression method.

The results of compression can be dramatic, as the table shows. The uncompressed print-ready file is over nine megabytes in size, while the compressed file size ranges from 18% to 25% of the uncompressed file. However, keep in mind that while the reduced file sizes save disk space and reduce transmission times, these files must be decompressed by the printer at print time, which is done automatically by the print controller.

CompileInStream

The CompileInStream option determines whether the FAP files have been loaded. If set to *Yes*, the print driver compiles the print stream using FAP files. Make sure the DownloadFAP option in the RunMode control group is set to *Yes*. If set to *No*, pre-compiled MET files are used.

The print driver creates the print stream using pre-compiled Metacode files. Use the FAP2MET utility to create pre-compiled Metacode files. The GenPrint program loads pre-compiled Metacode members from the PMETLIB PDS under MVS. On other platforms, the PMetLib option specifies the directory which contains the pre-compiled MET files. If you do not set this option, the system uses the setting for the FormLib option in the MasterResource control group.

NOTE: To use FRM files in your Metacode print stream, set the CompileInStream INI option to No in the Xerox printer control group. Using FRM files enhances performance in high volume situations that use a repeated background form on every page.

Device

This is the name of the file or device, such as LPT1, where the Metacode print stream should be written. This option is ignored by the GenPrint program but should not be left blank or omitted. For instance, you could enter *dummy.txt*.

RelativeScan

When set to Yes, the RelativeScan option tells the system to consolidate all records in the print stream. When set to No, this option tells the system to omit Relative Scan records when consolidating records. If you are using GenPrint version 9.0 or higher you will probably want to leave this option at its default setting (Yes) for maximum optimization.

Specifying Installable Functions

For the Xerox print driver, you must specify the following set of installable functions in the PrtType:XER control group:

```
OutputFunc     = XEROutput
OutMetFunc     = XEROutMet
InitFunc       = XERInit
TermFunc       = XERTerm
Module         = XERW32
PrintFunc      = XERPrint
```

Using defaults for the Module and PrintFunc options

Default values for the Module and PrintFunc options in the PrtType:xxx control group are provided when you use a standard print type name or print class, such as AFP, PCL, PDF, PST, VPP, XER, XMP, or GDI.

These defaults keep you from having to enter the Module and PrintFunc names in your INI file. For example, if you want to generate XER print files, you can specify these INI options:

```
< Printer >
    PrtType     = MYXER
< PrtType:MYAFP >
    Class       = XER
```

And the system will default these options for you:

```
< PrtType:MYAFP >
    Module      = XERPRT
    PrintFunc   = XERPrint
```

Optional INI Options

Setting the end of the report

Use the JDLRStack option to set the criteria which signals an *end of report* condition to the printer. In the JDL sample listed earlier, the RSTACK statement performed a criteria test named C2. The C2 test checks a specific part of each input line against the string named T2. If the string T2 matches an input data record at position zero (0) for a length of 10 bytes, an *end of report* condition is signaled. Only CONSTANT criteria using an EQ operator are supported.

Setting the JDLRStack option is optional. If your printer is used for both Metacode and text file print jobs, you *must* set this option. Using the JDL sample listed earlier, the option should be:

JSL statements	T2: TABLE CONSTANT=X'13131313131313131313'; C2: CRITERIA CONSTANT=(0,10,EQ,T2); RSTACK TEST=C2,DELIMITER=YES,PRINT=NONE;
INI option	JDLRStack = 0,10,EQ,X'13131313131313131313'

Starting new pages

Use the JDLRPage option to set the criteria which signals a *jump to front side of a new sheet* to the printer. In the JDL sample listed earlier, the RPAGE statement performed a criteria test named C3. The C3 test checks a specific part of each input line against the string named T3. If the string T3 matches an input data record at position zero (0) for length of 5 bytes, a *jump to new sheet* condition is signaled because of the *SIDE=NUFRONT* statement. Only CONSTANT criteria using an EQ operator are supported. For the JDLRPage option to work properly, the *SIDE=NUFRONT* and *WHEN=NOW* statements must be used as a part of the RPAGE settings in the JSL file.

Setting the JDLRPage option is optional. If the print job contains duplex pages alternating with simplex (one-sided) pages, this option provides a way to leave blank the backsides of certain pages. Using the JDL sample listed earlier, the option should be:

JSL statements	T3: TABLE CONSTANT=X'FFFF26FFFF'; C3: CRITERIA CONSTANT=(1,5,EQ,T3); RPAGE TEST=C3,SIDE=NUFRONT,WHEN=NOW;
INI option	JDLRPage = 1,5,EQ,X'FFFF26FFFF'

The Metacode print driver automatically places the SIDE=NUFRONT statement on all front pages when operating in duplex mode. This lets the system support print stream sorting facilities such as Mobius InfoPak. Also, the SIDE=NUBACK statement is now added to blank back pages when in duplex mode.

These statements eliminate the need for the ADDPAGES utility which some systems used with Mobius InfoPak support. Without this functionality the first page of an output may print on the back of a previous output.

You will need to add the SIDE=NUFRONT statement on all front pages printed, not only those pages that specify a tray change. This is necessary to handle the end of job condition where the last page prints on the front and is moved by InfoPak.

Also, the system will now add a SIDE=NUBACK statement for pages that start on the back side of the page, leaving the front side blank.

NOTE: You cannot configure these statements. The system automatically enters them into the print stream. You do not need to add SIDE=NUFRONT and SIDE=NUBACK statements to your Xerox printer control group (PrtType:XER).

Jogging pages

Use the JDLROffset option to set the criterion that tells the printer to initiate a page offset in the output bin. This option has not been fully implemented.

In the JDL sample, the ROFFSET statement performed a criteria test named *C1*. The C1 test checks a specific part of each input line against the string named *T1*. If the string T1 matches an input data record at position zero (0) for length of 9 bytes, a page offset is initiated. Only CONSTANT criteria using an EQ operator are supported.

Setting the JDLROffset option is optional. Using the JDL sample listed earlier, the option should be:

JSL statements	T1: TABLE CONSTANT=X'121212121212121212'; C1: CRITERIA CONSTANT=(0,9,EQ,T1); ROFFSET TEST=C1;
INI option	JDLROffset = 0,9,EQ,X'121212121212121212'

You can also jog form sets by transaction instead of by batch. In some situations, this can make manual assembly easier. To do this, set the OffsetLevel option to Formset, as shown here:

```
< PrtType:XER >
    OffsetLevel = Formset
```

This adds an additional 'OFFSET' parameter to the SIDE=NUFRONT command, which tells the printer to jog after each transaction.

Specifying spot color

Use the PrinterInk option to specify the color of ink loaded on a Xerox highlight color printer. You can set this option to one of the following colors:

Blue	Red	Green	Ruby	Violet	Brown
Gray	Cardinal	Royal	Cyan	Magenta	

Blue is the default if you omit this option. This option is used with the SendColor option. If you set the SendColor option to *Yes*, be sure to also set the PrinterInk option. Here is how you would specify cyan as the color of the ink stored on the printer:

```
PrinterInk    = cyan
```

Chart performance and print quality

By default, charts are rendered at 150 dpi (dots per inch) in a Metacode print stream. This setting typically provides for a smaller print stream and optimal performance from the GenPrint program.

Charts are scaled by the printer to their proper size and are printed as 300 dpi bitmaps. Because fewer dots are used at these lower dpi settings, you may notice some loss of detail in the printed output and effects such as:

- The circle which makes up the pie chart is less precise
- The lines used in a chart are thicker

Test charts printed to see if the loss of detail is acceptable. In general, horizontal and vertical lines scale with little or no loss of precision. Arcs and diagonal lines may lose some detail.

To disable rendering charts at 150 dpi, add the following option to the Xerox printer control group, usually named *PrtType:XER*:

```
ChartResolution = 300
```

The only other acceptable value for this option is 150. This option does not affect graphics printed as inline graphics.

Optimizing Metacode print streams

The GenPrint program lets you produce optimized Metacode print streams. You may want to consider using optimization if your Metacode output causes the printer to cycle down (wait) while printing.

This condition can occur when Metacode records cannot be transferred fast enough to the printer. Optimization helps remedy this situation by combining Metacode print records into larger and fewer records. Reducing the number of records that must be transmitted reduces the amount of time needed to spool the Metacode print stream to the printer. The cost is decreased GenPrint performance. You can also use the METOPT utility to optimize normal (non-optimized) Metacode output. For more information on this utility, see the Docutoolbox Reference.

To have the GenPrint program produce optimized Metacode output streams, add this FSISYS.INI option to have the GenPrint program sort and consolidate records to create more efficient print streams:

```
< PrtType:XER >
    Optimize = Yes
```

The Optimize option defaults to No, which tells the GenPrint program to run without sorting and consolidating records.

You can enable some extra error checking during optimization. If optimization encounters critical errors, such as the inability to find or open a file, it will notify you and stop immediately. It can report actual or potential non-critical problems it encounters while it runs. For instance, if optimization finds Metacode records that may prevent the file from printing, it can warn you.

To have optimization notify you if it spots potential problems, add the following option to your PrtType:XER control group:

```
< PrtType:XER >
    ValidLevel = 0 (default)
```

Enter zero (0) to tell the utility not to report non-critical problems. Enter one (1) to tell the utility to report warnings for non-critical problems, but continue optimizing. Enter two (2) to tell the utility to report warnings for non-critical problems and attempt to fix the problems. Enter three (3) to tell the utility to report warnings for non-critical problems and exit immediately.

Regardless of the option you choose, if you receive any warnings, be sure to closely check both the original and, if applicable, the optimized file.

Using a common font list

The METOPT utility and the Metacode print driver let you use common font lists at the beginning of a Metacode print stream. A common font list names all of the Xerox fonts that will be used by the print job.

By knowing all of the fonts up front, the Metacode driver can issue a single DJDE FONTS command once at the beginning of the job and avoid issuing DJDE FONTS commands on subsequent pages. This helps some Metacode printers print jobs at their highest rated speed.

In the CommonFonts control group, you will see a list of options similar to these:

```
< CommonFonts >
    Names = 28
    Name1 = FORMSX
    Name2 = FXUNBD
    Name3 = FXUNN6
    Name4 = FXCON6
    Name5 = FXUNN8
    Name6 = FXUNN0
    Name7 = FXUNBH
```

```
...
Name28 = FXUNI0
```

The first option, Names, defines the number of font name entries that follow. The following options specify the Xerox fonts which will be used in the print job.

NOTE: The format used for the CommonFonts control group is the same as that used by Documerge. Therefore, if you used this in Documerge, you can copy that INI control group into your Documaker INI file.

To use common font lists, you must use the METOPT utility or use the Metacode print driver and have the following INI options in the Xerox print group:

```
< PrtType:XER >
    Optimize = Yes
    MaxFonts =
```

Option	Description
Optimize	To use common font lists, set this option to Yes.
MaxFonts	Set this option to the maximum number of fonts your printer can handle in a single DJDE command. This number will vary based upon the printer's memory and configuration. The maximum value is 99 and the default is 20.

If the number of fonts in your common font list exceeds the MaxFonts value, the system outputs the MaxFonts number of fonts in the DJDE FONTS command. The DJDE FONTS command will contain the names of the fonts used on that page plus additional fonts from the common fonts list until the MaxFonts number of fonts is reached.

If the system encounters a page that uses a font not specified in the common fonts list (or the prior DJDE FONTS command to be more precise), it issues a new DJDE FONTS command which appends to the common font list the new fonts for that page.

Setting a default paper size

Use the PaperSize option to set a default paper size when converting Metacode print streams using the Internet Document Server or the MRG2FAP utility.

```
< PrtType:XER >
    PaperSize = 0
```

Enter	Description
zero (0)	for letter size (default)
1	for legal size
2	for A4 size
3	for executive size
98	for a custom size

Inline graphic performance and print quality

Graphics at 75, 100, or 150 dpi, printed using inline graphics, are scaled by the printer to their proper size and printed as 300 dpi bitmaps. Because fewer dots are used at these lower dpi settings, you may notice some loss of detail in the printed output and effects such as:

- Arcs and circles are less precise
- The lines used in a graphic are thicker

Test LOG files printed as inline graphics to see if the loss of detail is acceptable. In general, horizontal and vertical lines scale with little or no loss of precision. Arcs and diagonal lines may lose some detail.

To avoid scaling inline graphic LOG files, use Documaker Studio or Logo Manager to scale your graphics to 300 dpi. Most graphics are normally 300 dpi and most graphics are not printed as inline graphics.

Adding color to charts

Use the ColorCharts option to print the graphic portion of the chart in color.

```
ColorCharts = Yes
```

This option is used with the SendColor and PrinterInk options.

Using named paper trays

By default, Metacode output specifies the main tray for pages that use Tray 1. The AUX tray is specified for all other trays. If you have named trays in your JSL, specify these named trays in your options. An example of this option is shown here:

```
Tray1 = ONE1
Tray2 = TWO2
Tray3 = THREE3
Tray4 = FOUR4
```

Specifying the printer model

Use the PrinterModel option to specify the particular printer model you are using. There may be subtle differences between printer models that can affect the output sent to the printer. Currently, only the 3700 printer requires this setting. An example of this option is shown here:

```
PrinterModel = 3700
```

Specifying the resolution

Use the Resolution option to specify the printer's dots per inch resolution. Currently, only 300 dpi is supported, which is also the default.

```
Resolution = 300
```

Displaying console messages

Use the OTextString option to display a message on the printer console. The text you specify is sent before the print job starts. For example, this lets you display the message, *Put BLUE paper in tray 1* before a print job starts. Here is an example:

```
OTextString = "Put BLUE paper in tray 1"
```

The system also supports multiple OTEXT messages in the Metacode print driver at a print batch level. Additionally, the system lets OTEXT messaging generate multiple messages per print batch. To turn on multiple OTEXT messaging, add this option to the FSISYS.INI file

```
< PrtType:XER >
   MultipleOText = Yes
```

The default is No.

This tells the system to ignore the OTextString value in the PrtType control group and instead use the ones found in the appropriate print batch group.

For example, if you have three print batches, called BATCH1, BATCH2, and BATCH3, under each separate batch group, put required number of sequential messages for that batch:

```
< BATCH1 >
    ...
    OTextString1 = "Batch 1 OText String1"
    OTextString2 = "Batch 1 OText String2"
    OTextString3 = "Batch 1 OText String3"
< BATCH2 >
    ...
    OTextString1 = "Batch 2 OText String1"
    OTextString2 = "Batch 2 OText String2"
    OTextString3 = "Batch 2 OText String3"
< BATCH3 >
    ...
    OTextString1 = "Batch 3 OText String1"
    OTextString2 = "Batch 3 OText String2"
    OTextString3 = "Batch 3 OText String3"
    ***
```

Keep in mind that the index tags OTextStringX (where X is a number) must start with one (1) and be sequential. The system stops writing OTEXT records to the batch when it finds a tag that is out of sequence. Here is an example:

```
OTextString1 = "Batch 3 Otext String 1"
OTextString3 = "Batch 3 Otext String 3"
```

In this example, only the first one would display on the screen, because OTextString2 is not encountered next.

Stapling forms

Some Metacode printers include a stapling feature. The system supports this feature, but it has not been tested and is not warranted.

Using this feature, forms printed on certain Metacode printers can be stapled if you specify a StapleJDEName option in the PrtType control group. This causes a new JDE to be specified on forms that need to be stapled.

It is assumed that the Staple JDE option has the same settings as the normal JDE specified except for the additional STAPLE command. You specify which forms should be stapled using the Form Set Manager, which is part of Docucreate or via Documaker Studio.

This option only affects implementations which print to Metacode printers with the optional stapling feature. For more information on this feature, see the Docucreate User Guide. An example of this option is shown here:

```
StapleJDEName = JDESTP
```

Duplex switching

In earlier versions of the system, a Metacode print stream began and continued as a simplex job until the system encountered a page that needed to be duplex. At that point, the duplexing option was turned on. From that point forward, the print stream remained in duplex mode. For performance reasons, the system did not switch out of duplex mode. Research showed that for most cases, this was the most efficient way to drive the printer.

If, however, you are directing the printer output stream to a *value-added* process, you may want to include the actual duplex selection information with each form set. Without the commands to specify the duplex state, some value-added processes may not work properly. By setting the DJDELevel option to **Formset**, each form set will include a duplex command which specifies either simplex or duplex mode (DJDE DUPLEX=YES or NO always appears at the beginning of every new form set). A value other than *Formset* causes the duplex commands to be output as before. Here is an example:

```
DJDELevel =  Formset
```

Using VSAM to store resources

The system lets you store DDT files, precompiled Metacode resources, NA and POL files, and transaction trigger files in VSAM KSDS (Virtual Storage Access Method/Key Sequence Data Set) data sets. If you use this feature, you must set the following options in the VSAM control group in the FSISYS.INI file:

```
< VSAM >
    DDTVSAM = DD:DDTVSAM        DDT files
    METVSAM = DD:PMETVSAM       PreCompiled Metacode files
    VSAMRCPTB = DD:SETRCPVS     Transaction Trigger file
    VSAMNA = DD:NAFILE          NA and POL files
```

For more information on implementing VSAM support under MVS, see Optimizing Performance in the Installation Guide.

PrintViewOnly

If set to Yes, this option tells the system to print the view only sections. The default is No. This does not apply to entry only sections, which are never printed. Entry only sections are usually worksheets. If the section is marked as hidden and view only, it will not print.

Caching files to improve performance

The following options let you minimize the opening and closing of frequently used PDS members by retaining, or caching, file handles and file data. In many cases the default values are sufficient, but for specific cases in which you use many different sections, you may need to increase these caching values to improve performance.

Here are the options you can customize:

```
< Control >
    CacheFAPFiles   =
    RuleFilePool    =
    LogCaching      =
    CacheMethod     =
```

Option	Description
CacheFAPFiles	Specifies the number of FAP files to keep available for re-use without re-loading them from disk. The default is 100.
RuleFilePool	Specifies the number of DDT files to keep available for re-use without re-loading them from disk. The default is 100.
LogCaching	Enter No if you do not want the system to log caching statistics. The default is Yes.
CacheMethod	Use to set the type of caching method. You can choose from *LFU* (least frequently used), *LRU* (least recently used), or *LFUO* (least frequently used optimized). LFUO is the default.

MET files contain pre-compiled Metacode information produced by the FAP2MET utility. The GenPrint program loads MET members from the PMETLIB PDS under MVS. On other platforms, the PMetLib option specifies the directory containing the pre-compiled MET files.

If not set, the system uses the setting for the FormLib option in the MasterResource control group. The CacheFiles option keeps frequently used MET members available for re-use. This option is placed in the PrtType:XER control group in the FSISYS.INI file, as shown here:

```
< PrtType:XER >
    CacheFiles = 100   (default is 100)
    InitFunc   = XERInit
    TermFunc   = XERTerm
```

Caching statistics for FAP files, DDT files and Xerox resources such as pre-compiled Metacode files (PMETs) and forms (FRMs) are collected and can be placed in the LOGFILE.DAT file. These statistics show the following information:

Item	Description
Method	The caching method you are using (LFUO, LFU, or LRU).
Size	The size of the caches. The default is 100.
Hits	The number of times the system tried to load a resource from the cache and found it there.
Misses	The number of times the system tried to load a resource from the cache and did not find it there.
Total	The combined hits and misses. This represents the number of times the system tried to load a resource from the cache.
Purges	The number of times the system had to remove a resource from the cache to put another resource into the cache. The system decides which resource to remove based on the method. If you are using LFUO or LFU, the least frequently used resource is removed. If you are using LRU, the least recently used resource is removed.

Using the loader

The system lets you load print-ready Metacode files. For this feature to work, the print-ready Metacode file must have the same extension as the Ext option in the Loader:MET control group in the FAPCOMP.INI file. Here is an example:

```
< Loader:MET >
    Desc        = Xerox Metacode (*.MET)
    Ext         = .MET
    LoadFunc    = XERLoadMet
    Module      = XERW32
< Loaders >
    Loader      = MET
```

Along with the Metacode loader feature, another INI option is required in the PrtType:XER control group. The DefaultFont option defines the default font to use to indicate the names of any graphics in the print-ready Metacode file.

The graphics do not display in Image Editor when the print-ready Metacode file is opened. Instead the name of the graphic appears, in the default font, and the space taken by the graphic is indicated. In addition, the default font is also used for displaying any text that references a font not present in the font cross-reference file.

To set the default font, enter the name of a Xerox font file contained in the font cross-reference file as shown here:

```
< PrtType:XER >
    DefaultFont = FXTIN8
```

If there are any graphics in the MET file, the system requires a LOGO.DAT file in the FormLib directory so it can display graphics properly for all rotations. The LOGO.DAT file, which is a semicolon-delimited file, should look similar to this:

```
[file name for 0° rotation];[file name for 90° rotation];[file name
for 180° rotation];[file name for 270° rotation];
```

Here are a few points to keep in mind when using this feature:

- The PrtType settings must match the setting used to produce the print-ready Metacode file.
- Rotated text will not display properly.
- Blank pages are created for simplex forms printed in duplex mode.
- This feature slows the printing of large print-ready files (more than 100 pages).
- If there is a reference to a FRM file in the MET file, the system cannot display the MET file.
- The system cannot display charts and graphics.

Using the Class option

You can use the following INI option to specify the printer classification, such as AFP, PCL, XER, PST, or GDI. If you omit this option, the system defaults to the first three letters from the Module option.

```
< PrtType:XER >
    Class = XER
```

Some internal functions expect a certain type of printer. For instance, all 2-up functions require an AFP printer. The internal functions check the Class option to make sure the correct printer is available before continuing.

Adding user-defined DJDE statements

You can place the AdditionalDJDE option anywhere in the PrtType:XER control group. Each AdditionalDJDE value represents a distinct and separate DJDE statement, given verbatim. You can include as many AdditionalDJDE statements as needed. All of the located AdditionalDJDE statements are inserted into the print stream. You can also specify the batch in which to output the DJDE statement. Here is an example:

```
< PrtType:XER >
    AdditionalDJDE = "BATCH1";FEED=COVER,;
    InitFunc       = XERInit
    ...
    AdditionalDJDE = "BATCH1";STOCKS=BLUE,;
    ...
    AdditionalDJDE = JDL=DPLJDL,JDE=STRTON,;
```

The first two occurrences only apply to the BATCH1 batch. The third occurrence has no batch specified, so this DJDE statement is written to all print batches.

Keep in mind that these user-defined DJDE statements are placed after the BEGIN DJDE record and before the other DJDEs that are always inserted, such as FONTS. Make sure the DJDE syntax is correct and that the new DJDE records do not interfere with the ones automatically inserted into the print stream by the system.

Also, it is very important that you follow the correct syntax when coding the INI line. If you enter an invalid batch name, no corresponding batch will be found and the DJDE line will be ignored or not output in any batch. And, if the DJDE syntax is incorrect, the printer will issue error messages or unpredictable print results may occur.

Using third-party software to read Metacode files

If you use third-party software to read Skywire Software Metacode files and that software needs the DJDE, RSTACK, and RPAGE commands to begin with a carriage control value other than the default value of *0x01*, you can use the DJDECarrControl option to handle this. You simply enter a value in the form of a string constant. These string constants are supported:

- ASCII (A'string')
- EBCDIC (E'string')
- character ('string')
- hex (X'string')

NOTE: The character string produces an EBCDIC string, same as E'string'.

The default value is 1 (X'01'). Here is an example:

```
< PrtType:XER >
    DJDECarrControl = X'09'
```

Keep in mind that any carriage control value will be accepted and no attempt is made to make sure a valid carriage control is used.

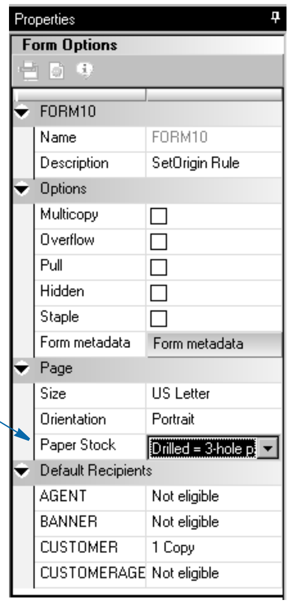
Specifying the paper stock

Using Documaker Studio you can specify what paper stock the form should print on. This will help users who have more than nine types of paper stocks. Here is an example of the INI options you could set up:

```
< PaperStockIDs >
    PaperStockID    = Drilled
    PaperStockID    = 20lb
    ...(and so on)
< PaperStockID:Drilled >
    Description     = 3-hole paper
< PaperStockID:20lbW >
    Description     = 20lb White Paper
    DJDE            = DJDE name
```

Once you have set up the appropriate PaperStockID options, you will see those options available via Studio's Form manager. Just open a form and select the appropriate paper stock in the Paper Stock field on the Properties tab, as shown here:

Click here to choose the paper stock.



Your selection is reflected in the POL file produced by the GenData program. In this example, the form called DEC PAGE has a paper stock ID of *Drilled*.

```
;SAMPCO;LB1;DEC
PAGE;||FORMPAPERSTOCK=Drilled||;R;;QPRUNA|DL(3360,18600)<AGENT,COMP
ANY,INSURED>
```

In the Metacode printer control group, you must set the TrayUsePaperStockID option to Yes, as shown here:

```
< PrtType:XER >
    TrayUsePaperStockID = Yes
```

If the TrayUsePaperStockID option is set to Yes, the Metacode print driver takes the form's PaperStockID and tries to find the DJDE INI option for it in the INI file when it emits the tray command.

Keep in mind...

- The paper stock selection applies to the entire form
- Only the Metacode print driver uses the paper stock selection
- Only Documaker Studio lets you select the paper stock

Metacode Printer Resources

A number of resources are used in the printing process. These resources generally reside on the printer's disk drive.

Fonts

Xerox fonts are ASCII fonts. Xerox fonts are not scalable and do not rotate. There is one font file for each rotation and different files are required for different sizes. The file extension is *FNT* and file names are up to six characters long. Skywire Software has licensed for use and distribution with its systems, fonts from Monotype Imaging, Inc. Xerox fonts for all four rotations are included.

Forms

Xerox forms are precompiled electronic files containing static text, boxes, graphics, and so on, ready to be merged with variable data. Forms always have the extension *FRM*. Like fonts, the maximum file name is six characters. You use the FAP2FRM utility to create Xerox forms from FAP files.

Images

Xerox images are large bitmaps or raster patterns that are stored in a special file format. These images are merged onto the forms which are then merged with the variable data. The file extension is *IMG* and the maximum file name is six characters.

NOTE: You must install a GVG hardware card on the printer to print IMG files. You can use the LOG2IMG utility to create Xerox images from LOG files.

Logos

Logos are small bitmaps stored in a different format than IMG files. The extension is *LGO* and the file name is six characters long. You can only use Xerox logos inside a FRM file. You cannot invoke them directly in the data stream.

NOTE: These *LGO* files are quite different than *LOG* files used in Documaker Studio and Logo Manager. Documaker software does not use Xerox LGO files.

Metacode Limitations

Xerox images

The maximum number of images and inline graphics per page is 16.

HMI support

HMI (horizontal motion index) is supported for zero (0) and 270 degree rotated text on portrait forms only. HMI combines separate text labels which are positioned on the same line and which use the same font into a single Metacode record. FAP files with justified paragraphs can benefit from this feature. Use the FAP2MET utility to implement HMI into pre-compiled MET files.

Changing the paper size on the 4235 printer

You can not easily change paper sizes in one job. Each job is controlled by a JDE. If you need to pull paper from bins of different sizes, you have to call a different JDE each time you change from one paper size to another. This is similar to staple support. There is no code to invoke different JDEs for change of paper size.

Xerox forms

If a Xerox form (FRM file) contains more than 48 blocks (each block is 512 bytes), your printer may not have enough memory to print it.

The CD (Character Dispatcher) memory is divided into three regions. The first region loads all fonts used on a page. The second region is used for TL/DLs which contain inline Metacode (may only be variable data if you use an FRM). The third region loads the TL/DLs from an FRM file, if one is being used for the page.

If you have version 2 of the printer software, your printer supports eight TL/DL buffers of 3K each (same as 48 blocks of 512 bytes each) for inline Metacode. With version 3.5 of the printer software, the limit was increased to 16 buffers of 3K each.

NOTE: Skywire Software's testing shows that with version 3.5, TL/DLs from FRMs (the third region of CD memory) are still limited to 8 TL/DL buffers of 3K each (same as 48 blocks of 512 bytes each).

Typically, Xerox 9700 and 9790 printers still have the older release installed. If so, you may want to upgrade to version 3.5. The Xerox 4000 series printers (4050, 4850, and so on) always come with version 3.5 or higher.

When you are not using FRM files in a print stream, the system does not use the CD memory reserved for FRM files.

Metacode Troubleshooting

Unexpected color output

Even though you set the SendColor option set to *No*, you still get color output when printing. This occurs when:

- You specified *Print in color* for some elements of the FAP file
- You precompiled the FAP files with the */C* option on FAP2MET
- A *SUB INK* command was issued on the printer

If ink substitution occurred because of an operator command, such as SUB INK BLUE (or RED or GREEN), the colored components of the precompiled MET file will be brought in with color attribute turned on and printed with color. This happens regardless of how you set the SendColor option. To print in black and white, either re-run the FAP2MET utility with no */C* flag, or use the *END* command to cancel ink substitution on the printer.

Unexpected black and white output

Even though you set the SendColor option to Yes, you still get black and white output when printing. Use this checklist to make sure you have done everything to print in color:

- Make sure you specified *Print in color* for the color elements in your FAP file, such as text, shaded areas, lines, and so on.
- Make sure you precompiled the FAP files with the */C* option on FAP2MET;
- If you are using precompiled FAP files, make sure you compiled those FAPs using the FAP2CFA utility.
- Make sure you run the GenPrint program with the SendColor option set to Yes.

Highlight color should match the PrinterInk option

The PrinterInk option causes a DJDE ILIST command which specifies the highlight color to use. If a different highlight color is installed on the printer, the printer follows the procedure specified in the ABNORMAL statement in the JDL and JDE loaded. The ABNORMAL procedure specifies whether the job should continue, abort, or stop. If no ABNORMAL procedure is declared, the default is for the printer to stop so a new ink cartridge can be loaded. Besides the ABNORMAL statement, the printer operator can override the ink setting using the SUB command (for example, SUB INK BLUE or SUB INK CURRENT).

LOG file orientation

To print a portrait section which contains a graphic on a landscape form using pre-compiled MET files, set the LoadFAPBitmap option to *Yes*. This is necessary because the graphic name must change from the portrait (zero degrees) name to the landscape (270 degrees) name.

Output catching up with the input

If your printer cycles down and displays a message stating that the output caught up with the input, it indicates the average number of records per physical page is greater than the maximum number of records that can be transferred across the channel in the time allowed for a page.

This situation causes the printer to cycle down so it can buffer more pages before it continues. This table shows the maximum average number of records that can be transferred across the channel in time to support the printer running at rated speed:

Printer	Maximum Records Per Page
4050	285
4090	155
DP96	149
41/4635	105
DP180	78

To resolve this problem, you need to optimize the Metacode print stream. For more information, see Optimizing Metacode print streams on page 52.

Printing rotated variables

Here is a list of field options you can specify in the NAFILE.DAT file:

Option	Description
E	Error
M	Manual
P	Protected
G	Global scope (entire form set)
F	Form scope
H	Hidden field (such as a dummy field, neither displayed nor printed)

Option	Description
N	Nonprintable field (displayed, not printed)
C	Send-copy-to field (receives current recipient name at print time)
9	Rotated 90 degrees
8	Rotated 180 degrees
7	Rotated 270 degrees

Some of these options require the FAP field attributes to be available at runtime, since the DDT file does not include the necessary information. You can use the CheckImageLoaded rule to make sure this information is available.

Multi-page sections

When you use multi-page FAP files and pre-compiled MET files, you must use the EjectPage rule. This rule enables the printing of multi-page sections. Here are the steps to apply the rule in Image Editor:

1. Open the FAP file in Image Editor.
2. Select Format, Image Properties and then click the Load DDT button.

Image Editor detects that the image contains multiple pages and inserts into the DDT file as many EjectPage rules as there are pages.

NOTE: You must have a variable field on each page. The variable field can be a dummy field that is hidden.

When you implement multi-page FAP files and pre-compiled MET files, keep these requirements in mind:

- Only multi-page FAP files are applicable.
- Multi-page FAP files cannot be mixed with single page FAP files on the same form. The system cannot easily determine the page number in this case.
- The multi-page FAP file came from Documaker Studio or Image Editor and therefore there is only one section per page, hence, each page on the form has an section list that contains one and only one section.
- The index of the page on which that section resides within that form is the number of the page.
- Multi-page sections can be duplexed by setting the form to either *Front* (long edge binding) or *Short* bind (short edge binding). Internally created sections will be set to *Rolling* for the remaining pages.

NOTE: If a form begins with a rolling duplex option, the print drivers begin printing on the blank back page of the previous form. Any form that starts with rolling and begins a form set is treated as the front page of a rolling set.

Operator command, FEED, causes duplex problems

If you enter an operator command to specify an input tray—because for instance, one paper tray is empty and while you refill it you want the printing to continue using another tray—you can no longer select trays from DJDEs in the job stream. Instead, you will get messages which tell you tray selection was suspended by an operator override.

All paper feed from that point forward, will be from the tray specified in the operator command. This can cause duplex jobs to print incorrectly if you have completed printing on a front page and the next page should print from a different tray.

To correct this situation, enter a *FEED=MAIN* command. This command tells the printer to switch to tray 1 and enables tray selection through DJDE commands so the next paper selection command is obeyed.

Line density errors

As the speed of the printer increases, there is less and less time available to the character dispatcher to form the scan line and send it to the image generator. Here is some information on how this affects the various Xerox printers:

- Since the 4135 printer is the fastest of the Xerox printers using the older CD/IG, the chances of running out of time and causing a *line density error* are greatest with this model.
- The Xerox 4050 and 4850 printers are too slow for this to be a problem. These printers allow more fetches from the font memory per scan line.
- The Xerox 4635 printer's image generation module has been revamped to such an extent that Xerox almost guarantees there will never be a line density exceeded error on a 4635 printer.
- The 4235 printer is slow and works quite differently than the centralized printers.
- If a job works fine on a 9790 printer but fails on a 4135, the number of character fetches is likely on the borderline of failure.

If you experience line density problems, check your FAP files for the following:

- Text superimposed on shaded areas.
- Large number of text lines with small fonts.
- Large number of horizontal lines whose thickness is measured in an odd number of dots. If you change the thickness of a horizontal line from three dots to two or four dots (0.01" to .006666" or 0.013333"—24 FAP units to 16 or 32 FAP units), it reduces the character count from two to one.

  The Xerox line drawing font has three horizontal line drawing characters which specify lines with thicknesses of two, four, and eight dots (.006666", .013333" and .0 26666" or 16, 32, and 64 FAP units). Odd thicknesses require the printer to overlay or overlap multiple lines.
- Large number of small boxes, many of which have common boundaries. On paper it looks like one line. Actually, there may be two or more character fetches for the same black dots. Create these kinds of boxes by drawing lines rather than boxes.

Output data length validation

Metacode printer JSL specifies the length of data that can be received. This data length must match the value output into the Metacode print stream. You specify the data length in the JSL as shown here:

```
LINE DATA = (0,213)
```

You specify the data length in the PrtType:XER control group in the FSISYS.INI file, as shown here:

```
< PrtType:XER >
    JDLData = 0,213
```

In this example, the JSL specifies a maximum data length of *213*, so the INI option has a matching value. The maximum length value is also used in the Metacode print driver to make sure no more than the specified amount of data is output in any Metacode record. If the amount of data to be emitted in the record exceeds this amount, an error message such as the following appears:

```
Record Length 214 is too long - maximum length is 213.
```

NOTE: Under MVS, Metacode output files are VB datasets. The JCL specifies a maximum length of a record (LRECL). If an attempt is made to write a record longer than the LRECL value, the write will fail and an error message appears.

Be advised that under MVS, with VB datasets, the LRECL size includes a 4-byte record length, known as the *RDW*. The RDW is implicitly added to the front of each variable length record. Therefore, you should set the LRECL value for the Metacode output dataset to a number equal to the JSL maximum length plus four to account for the RDW bytes at the front of the record. For the above example, set the LRECL of the Metacode output file to *217*.

Using Xerox Forms (FRMs)

The system lets you use Xerox form (FRM) resources when you print to Xerox Metacode printers. FRMs are printer resident resources that contain static full-page images. The system can use FRMs during the print process.

You can convert frequently used static full-page images into FRMs using the FAP2FRM utility. To indicate an image is resident on the printer as a FRM file, use the Form Set Manager. The Printer Resident field indicates the image is a pre-compiled resource resident on the printer—as opposed to a pre-compiled resource that needs to be downloaded to the printer. For more information on the Form Set Manager and the Printer Resident field, see the Docucreate User Guide.

Here are some guidelines for using Xerox forms (FRMs):

- Create one FAP file per page. If there is a text area, do not put variable data within the text area.
- The image size must be one of the standard paper sizes, such as US Letter, Legal, A4, or Executive.

- Because Xerox printers can only accept file names up to six characters in length, the image name can be up to six character long. If, however, the image is a multi-page FAP file, the name can consist of no more than four characters to accommodate the two-character number added by the FAP2FRM utility. Here are some examples: *TEST01.FRM* for the first page, *TEST02.FRM* for the second, and so on.
- Use the FAP2FRM utility to convert FAP files into FRM files. For multi-page FAP files, create multiple FRM files. The names are appended with two-digit numeric suffixes.
- On workstations, store the FRM files in the same directory as the FAP files. On MVS, keep them in a PDS attached to the PFRMLIB DD name. On AS400 systems, use the FRMFile option in the Data control group to specify where the FRM files are stored.
- Use the Printer Resident field in the Form Set Manager to mark individual forms as *printer resident*. After you do this, the FORM.DAT file contains the *V* image option which indicates the image is resident on the printer. When you run the GenPrint program, a DJDE *FORMS=fname* command is inserted for the corresponding images. The remainder of the images are printed by inline Metacode, possibly using precompiled MET files.
- Install the FRM files on the Xerox printer using the XERDNLD utility. Copy the resulting **.DAT* files to the printer. To make sure the forms are installed on the printer, use the SAMPLE console command to print the form files.

BARRWRAP

The BARRWRAP utility converts Metacode output from JES2 format into BARR format.

The BARR interface attachment for Metacode printers requires that the Metacode print stream files contain BARR specific information. The BARRWRAP utility adds this information to an existing Metacode print stream file, which lets you print the output file via the BARR interface.

After you run the utility on a Metacode file, *76 1A FF 00* is added at the beginning of the file. This tells BARR the file is a Metacode file. A byte denoting the record length is also added at the beginning and end of each record in the file.

Use this utility when you test the GenPrint program on MVS. If the MVS system is not directly channel-attached to the Xerox printer, you must download the print streams to an OS\2 system—use no ASCII translation, but do use CRLF. Then, using BARRWRAP, the print stream is packaged to successfully pass through BARR/SPOOL.

NOTE: Occasionally, the binary data contained in a Metacode file has a sequence of hex bytes (*x' 0D0A')* which could be misinterpreted as a carriage return/line feed. This is true particularly for charts and other inline graphics. Convert such data streams using the BARRWRAP utility on the MVS platform before you download them with the no ASCII and no CRLF (binary) options.

Transferring Files from Xerox Format Floppies

Resources saved on a 5 1/4-inch floppy, using *FLOPPY SAVE file.ext*, are saved in a special Xerox format. For use in the system, or for transferring to a 4235 printer, you must convert these resources into DOS format. You can use the following software packages to perform this required conversion:

- FloppyCopy by Lytrod Software – Inexpensive, easy to use
- Elixir – More expensive, but includes additional features.
- LaserLinx – No longer marketed.

PCL PRINTERS

Hewlett-Packard created the Printer Control Language (PCL) to provide a way for application programs to control a range of printer features across a wide array of printing devices.

The PCL language has evolved over time. For the most part, system-produced PCL output will run on any printer that supports PCL 5 or PCL 6. There are separate drivers for these two versions of the PCL language.

To support color printing, the printer must support PCL 5c, which contains color extensions. To support more than two paper trays, the printer must support PCL 5e.

NOTE: All system print drivers support 24-bit color graphics. The PXL (PCL 6) driver supports monocolor, 8-bit color (256 color), and 24-bit color graphics.

If your printer does not support color, the print driver will automatically convert the color graphics into monochrome graphics. Keep in mind that for the best performance you should avoid color graphics.

PCL INI OPTIONS

You must define the necessary printer options for the GenPrint program to produce PCL output. These options specify PCL output and are located in a PrtType:xxx control group, such as *PrtType:PCL* for PCL 5 or *PrtType:PXL* for PCL 6. Common PCL printer options are shown below, with default values in bold:

Option	Values	Description
Device	any file or device name	The name of the file or device (LPT1) where the PCL print stream should be written. This setting is ignored by the GenPrint program but is used by Studio, Image Editor, and other Documaker system programs.
Module	PCLW32	The name of the program module which contains the PCL print driver. See also the Class option. For PCL6, enter **PXLW32**. See also Using defaults for the Module and PrintFunc options on page 50.
PrintFunc	PCLPrint	The name of the program function that is the main entry point into the PCL print driver. For PCL6, enter **PXLPrint**. See also Using defaults for the Module and PrintFunc options on page 50.
Resolution	**300**	The dots per inch resolution of the printer which will receive the PCL data stream.
SendOverlays	Yes/No	Set to *Yes* if you created PCL overlays for each FAP file. This option is not supported for PCL 6.

Option	Values	Description

OverlayPath	any directory	Set to the directory containing the PCL overlays for each FAP file. The default is the FormLib option of the MasterResource control group.
		Here is an example:
		`< MasterResource >`
		`FormLib = <CONFIG:Batch Processing>`
		`FormLib = <CONFIG:Batch Processing>`
		`FormLib = ./forms/`
		This option is not supported for PCL 6.

OverlayExt	any file extension (**OVL**)	The file extension of the PCL overlays. This option is not supported for PCL 6.
PageNumbers	Yes/**No**	Set to *Yes* to enable form or form set page numbering.
SendColor	Yes/**No** Enabled/ Disabled/ Hidden	Set to *Yes* to enable color printing. Enabled = Option appears in the Print window and is active (available to be checked). Disabled = Option appears in the Print window but is grayed out (not available to be checked). Hidden = Option does not appear in the Print window.
HighlightColor	Yes/**No**	Set this option and the SendColor option to Yes to use simple color mode. See Using Simple Color Mode on page 77 for more information. This option is not supported for PCL 6.
DownloadFonts	**Yes**/No	Set to *Yes* to enable downloading of PCL fonts. For PCL6, you must enter Yes because internal font selection is not supported.
TemplateFields	Yes/**No**	Set to Yes to test print *X*s in variable fields.
FitToWidth	Yes/**No**	Not supported by either PCL print driver.
AdjLeftMargin	**Yes**/No	Automatically adjusts the left margin to compensate for the 1/4-inch left margin added by PCL printers. Yes = Automatically adjust the left margin. Forms print exactly as they appear on screen (default). No = Do not adjust the left margin. Forms may not print correctly on PCL printers after performing a retrieve function. This option is not supported for PCL 6.

Option	Values	Description
SelectRecipients	**Yes**/No **Enabled**/ Disabled/ Hidden	Set to *No* to disable the ability to select recipients. Enabled = Appears in the Print window and is active (available to be checked). Disabled = Appears in the Print window but is grayed out (not available to be checked). Hidden = Does not appear in the Print window.
PrintViewOnly	Yes/*No*	If set to Yes, the view only sections will print. This does not apply to entry only sections, which are never printed. Entry only sections are usually worksheets. If the section is marked as hidden and view only, it will not print.
PrePrintedPaper	Yes,Disabled	Determines if the check box which lets you print or not print pre-printed objects appears on the Print window. Also determines the default for this field—checked or unchecked. You must add this option to the INI file if you want the field to appear on the Print window. The default includes the field on the Print window and leaves it unchecked. All objects except fields can be marked pre-printed on the object's Properties window.
Class	(*first three characters of the Module option*)	Specifies the printer classification, such as AFP, PCL, XER, PST, or GDI. If you omit this option, the system defaults to the first three letters from the Module option. Some internal functions expect a certain type of printer. For instance, all 2-up functions require an AFP printer. The internal functions check the Class option to make sure the correct printer is available before continuing.
StapleBin		Set this option to the PCL printer escape sequence that selects the bin that contains the staple attachment. Use a tilde character (~) in place of the binary escape character. This option is not supported for PCL 6.
PJLCommentScript		To add PJL comments to a PCL print stream, enter the name of the DAL script you want the system to run. This DAL script creates the control strings and adds them as ASCII comments. This option is not supported for PCL 6.
PJLCommentOn	**batch**/formset	Use this option to add PJL comment records to the beginning of every form set or batch. This option is not supported for PCL 6.

Option	Values	Description
OutputBin		Enter the printer escape sequence to select the normal output bin (for non-stapled forms) if non-stapled forms are being sent to the wrong bin. This option is not supported for PCL 6.

NOTE: The default FAPCOMP.INI file should include the PrtType:GDI control group and options in addition to the PrtType:PCL or PrtType:PXL control group.

Using defaults for the Module and PrintFunc options

Default values for the Module and PrintFunc options in the PrtType:xxx control group are provided when you use a standard print type name or print class, such as AFP, PCL, PDF, PST, VPP, XER, XMP, or GDI.

These defaults keep you from having to enter the Module and PrintFunc names in your INI file. For example, if you want to generate PCL print files, you can specify these INI options:

```
< Printer >
    PrtType     = MYPCL
< PrtType:MYAFP >
    Class       = PCL
```

And the system will default these options for you:

```
< PrtType:MYAFP >
    Module      = PCLPRT
    PrintFunc   = PCLPrint
```

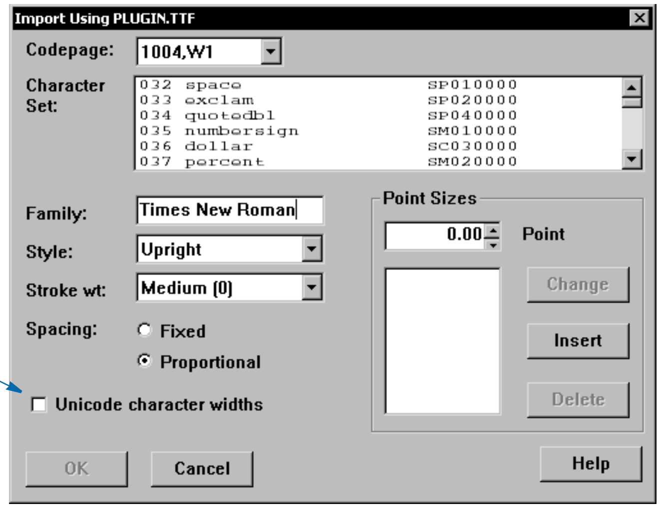
Using PCL 6

PCL 6 is a stack-based protocol (similar to PostScript) composed of attributes and operators that let you define paths, clip paths, pens, brushes, fonts, raster patterns, and so on. PCL6 also supports 16-bit character codes which makes it a better choice for supporting Unicode than PCL 5.

The PCL 6 driver lets you download both PCL bitmap fonts and TrueType fonts. You must specify the TrueType font file name in the Font File entry of the PCL printer section in the font cross-reference (FXR) file.

To turn on Unicode support, check the Unicode Character Widths field when you insert a TrueType font into the FXR file. Unicode support lets you use additional characters and languages supported by the TrueType font.

Click here to turn on Unicode support.



Keep in mind...

- The PCL 6 driver supports PCL bitmap fonts so you can use master resource libraries (MRLs) designed for PCL 5. Just remember to make the appropriate changes to your INI options.
- When printing using a TrueType font, only the characters used on the form are downloaded into the print stream. This reduces the size of print stream files, particularly if the TrueType font includes support for Asian languages.

In comparison to the PCL 5 printer driver, the PCL 6 driver has these limitations:

- No overlay support
- No support for a separate downloadable font file which contains multiple PCL fonts
- No internal printer font support
- Less paper tray support, no INI options to specify which PCL commands to use
- No INI options to specify PCL commands to output bin or staple bin
- No highlight color support
- No comment script support

Printing Under Windows

Windows XP/2000 does not recognize printer ports such as LPT1. If you are using Windows XP/2000, you must change the PrtType control group in the FSIUSER.INI file to reflect the print server name and print device. Here is an example:

```
< PrtType:PCL >
    Device = \\FSISRV03\\OPTRA1
```

Using High-Capacity Trays 3 and 4 on HP 5SI Printers

The system defines document attributes in a device-independent fashion. In prior versions, PCL support was based on options available to PCL 5 and similarly configured printers. The newer HP 5SI printer offers additional capabilities which depend upon (at least somewhat) commands that exist in PCL 5e. To add to the confusion, HP is not always consistent with its own terminology. Here is how the system treated PCL in prior versions:

NOTE: The ability to define trays or use the Tray# option is not supported for PCL 6.

System term	PCL command	PCL term	HP 4 term	HP 4si term	HP 5si term
Tray 1 (Main)	~&l1H	Tray 2 (upper)	PC Tray	Upper tray	Tray 2 (upper drawer)
Tray 2 (Aux)	~&l4H	Tray 3 (lower)	MP	Lower tray	Tray 3 (lower drawer)
Tray 3 (Man)	~&l2H	Manual feed	Tray 1	Manual feed	Tray 1 (manual side feed)
Tray 4 (Env)	~&l3H	Envelope feed	Tray 1	Manual feed	Tray 1 (manual side feed)
n/a	~&l5H	HCI, first tray	LC Tray	n/a	First tray of HCI
n/a	~l20H	HCI, second tray	n/a	n/a	Second tray of HCI

The terms for the current version are shown below, with changes highlighted:

System term	PCL command	PCL term	HP 4 term	HP 4si term	HP 5si term
Tray 1 (Main)	~&l1H	Tray 2 (upper)	PC Tray	Upper tray	Tray 2 (upper drawer)
Tray 2 (Aux)	~&l4H	Tray 3 (lower)	MP	Lower tray	Tray 3 (lower drawer)
Tray 3	~&l5H	HCI, first tray	LC Tray	n/a	First tray of HCI
Tray 4	~l20H	HCI, second tray	n/a	n/a	Second tray of HCI

The command *~&l5H* (first high-capacity tray) is supported by PCL 5, but the hardware is not typically found on HP printers. The command *~&l20H* requires PCL 5e.

You can use these INI options:

```
< PrtType:PCL >
    Tray1 = pcl command sequence (default is ~&l1H)
    Tray2 = pcl command sequence (default is ~&l4H)
    Tray3 = pcl command sequence (default is ~&l5H)
    Tray4 = pcl command sequence (default is ~&l20H)
```

Keep in mind the paper size overrides the tray selection.

NOTE: See also for Handling Multiple Paper Trays on page 124 more information.

If you depend on the prior sequence, you can return to the original operation by specifying:

```
< PrtType:PCL >
    Tray3 = ~&l2H
    Tray4 = ~&l3H
```

NOTE: The tilde (~) represents the escape character and is translated internally. The third character in each sequence shown is a lowercase *L*.

Using a staple attachment

In your PCL printer group, usually PrtType:PCL, add the StapleBin option to use a staple attachment on your PCL printer.

Set the StapleBin option to the PCL printer escape sequence that selects the output bin which contains the staple attachment. Use a tilde (~) in place of the binary escape character.

Here is an example:

~&l2G `(tilde, ampersand, lower case l, 2, upper case G)`

This example shows the escape sequence used to select an optional lower (rear) output bin that may have a staple attachment. Check with your printer manual for the escape sequence you should use.

The OutputBin option should contain the printer escape sequence needed to select the normal output bin (for non-stapled forms). Using the OutputBin option is not necessary unless you notice the non-stapled forms are being sent to the wrong output bin. This INI option is only necessary when you have both stapled and non-stapled forms in the same print batch.

Using Simple Color Mode

The PCL print driver supports PCL simple color mode in addition to full RGB color support. PCL simple color mode uses a 3-plane CMY palette. The 3-plane CMY palette contains these indexed colors:

0 - White

1 - Cyan

2 - Magenta

3 - Blue

4 - Yellow

5 - Green

6 - Red

7 - Black

To specify highlight color printing for PCL, include these INI options:

```
< PrtType:PCL >
    SendColor = Yes
    HighlightColor = Yes
```

Marking objects to print in color

For any object, such as lines, boxes, or text, select the Print in Color option on the Color Selection window if you want the object to print in a color other than black. Keep in mind...

- If the object is black and is not marked as Print in color, the system prints the object using a black color index.
- If the object has a color other than blank and is marked as Print in color, the system prints it using a highlight color index.
- Charts print in black, although you can print chart labels in the highlight color.

Specifying the highlight color to use

You can use these INI options to specify the PCL color commands to use for printing the black and highlight colors. The default values are shown here:

```
< PrtType:PCL >
    HighlightColorCmd = ~*v3S
    HighlightBlackCmd = ~*v7S
```

Note that the tilde (~) is used in place of the PCL escape character (hex 1B) and that the number used in the command corresponds to the color indexes specified earlier, such as 3=Blue and 7=Black.

To use a different highlight color, include the HighlightColorCmd option. To use a different black color, specify the HighlightBlackCmd option.

Printing on different types of printers

Printing black and white, highlight color, and full color print streams on black and white, highlight color, and full color PCL printers will produce varying results.

You can usually send a color PCL print stream to a black and white PCL printer without any problem—everything comes out black and white. PCL printers usually ignore any commands they do not understand.

If, however, you send a highlight color PCL print stream to a full color PCL printer, the results may be slightly different than if you had sent the print stream to a highlight color printer.

Bitmap graphics in a highlight color print stream may print as cyan on a full color printer. Bitmaps are a sequence of binary data—zeros (0) and ones (1)—so the zeros may print as white, while the ones may print as cyan. On a highlight color printer, the bitmap is printed as expected using the black or highlight color.

If you send a full color PCL print stream to a highlight color printer, your results may vary based on the printer model and printer settings.

Creating Compressed PCL Files

You can create compressed PCL files using Documaker. This capability is typically used with IDS because Windows does not let you print files that are a mixture of simplex and duplex pages from Acrobat. The whole document has to be printed the same way. IDS, however, lets you print a file to a local PCL printer which preserves the file's duplex information.

To create compressed PCL files, set these INI options. These options call the PRTZCompressOutPutFunc function to compress an output file, such as a PCL print batch file:

```
< PrtType:PCL >
    OutputMod = PRTW32
    OutputFunc = PRTZCompressOutputFunc
```

NOTE: The output is compressed, regardless of the file's extension. You must decompress the file before you can print it.

Bitmap compression

The PCL print driver also supports bitmap compression. To disable bitmap compression, add the following INI option to the PCL printer control group:

```
< PrtType:XXX >
    Compression = No
```

Adding Printer Job Level Comments

Printer Job Language (PCL) comments are supported by some PCL printers (not PCL 6). One type of command lets you add a comment to your PCL print stream. The PJL comment does not affect printing but can pass information to other products that look for specific information in PJL comment records, such as an imaging system.

NOTE: Imaging products can be used to archive PCL print streams. These products often require a control record at the beginning of the PCL print stream. These options and DAL functions let you create that control record.

To add PJL comments, add the following INI option to the PCL print group:

```
< PrtType:PCL >
    PJLCommentScript = imaging.DAL
```

The PJLCommentScript option specifies the DAL script you want to run. This DAL script creates a control string and adds it as an ASCII comment. Here is an example of the DAL script:

```
* Add imaging comment - use default APPIDX record.
Comment = AppIdxRec( )
AddComment(Comment,1)
Return('Finished!')
```

Notice the use of the second parameter to the AddComment DAL function. The *1* indicates the string should be an ASCII string. If you omit this parameter, the system converts the string into an EBCDIC string.

You can also use the PJLComment option to tell the system to add PJL comments to the beginning of every form set or print batch. Here is an example:

```
< PrtType:PCL >
    PJLCommentScript   = imaging.DAL
    PJLCommentOn       = formset
```

Adding Data for Imaging Systems

The PCL print driver can add free form text or data at the beginning of a batch or each form set within the batch. This can help you interface with imaging systems such as RightFax.

Use the TEXTScript INI option to specify the DAL script you want to run. This DAL script creates a free form data or text buffer and adds it to the print stream.

Here is an example of the DAL script:

```
* Populate the PCL stream comment with these values from RCBDFD
faxnum = trim(GVM('FaxNumber'))
faxname = trim(GVM('FaxName'))

AddComment('<TOFAXNUM:' & faxnum & '>',1)
AddComment('<TONAME:' & faxname & '>',1)

Return
```

Notice the use of the second parameter to the AddComment DAL function. The 1 indicates the string should be an ASCII string. If you omit this parameter, the system converts the string into an EBCDIC string. You can also use the TEXTCommentOn option to tell the system to add free form text or data to the beginning of every form set or print batch. Here is an example:

```
< PrtType:PCL >
   TEXTScript      = imaging.DAL
   TEXTCommentOn   = formset
```

PCL Printer Resources

A number of resources are used in the printing process. These resources reside in directories specified in the MasterResource control group.

Fonts

The system supports PCL bitmap fonts. These fonts reside in the directory specified in the FontLib option in the MasterResource control group when you set the DownloadFonts option to *Yes*. The system includes utilities for creating PCL fonts from PostScript, TrueType, Xerox, or AFP fonts.

Overlays

Use the OVLCOMP utility to create PCL overlays from FAP files. These overlays must reside in the directory specified in the OverlayPath option in the MasterResource control group when you set the SendOverlays option to *Yes*.

Note: Because the PCL 6 driver supports PCL bitmap fonts, you can use master resource libraries (MRLs) designed for PCL 5. Just remember to make the appropriate changes to your INI options.

PostScript Printers

Adobe Systems created the PostScript language. It is an interpretive programming language with powerful graphics capabilities. For the most part, system-produced PostScript output will run on any printer that supports PostScript Level 2.

NOTE: The PostScript print driver supports monocolor, 4-bit, 8-bit, and 24-bit color bitmaps. If you printer does not support color, the print driver will automatically convert the color graphics into monochrome graphics. Keep in mind that for the best performance you should avoid color graphics.

PostScript INI Options

You must define the necessary printer related options for the GenPrint program to produce PostScript output. These options specify PostScript output and are located in a PrtType:xxx control group, such as *PrtType:PST*. Common PostScript printer options are shown below, with default values in bold:

Option	Values	Description
Device	any file or device name	The name of the file or device (LPT1) where the PCL print stream should be written. This setting is ignored by the GenPrint program but is used by Documaker Studio, Image Editor, and other system programs.
Module	PSTW32	The name of the program module which contains the PostScript print driver. See also the Class option. See also Using defaults for the Module and PrintFunc options on page 84.
PrintFunc	PSTPrint	The name of the program function that is the main entry point into the PostScript print driver. See also Using defaults for the Module and PrintFunc options on page 84.
Resolution	**300**	The dots per inch resolution of the printer which will receive the PostScript data stream.
SendOverlays	Yes/**No**	Set to Yes if you have created PostScript overlays for each FAP file. See also Creating Smaller PostScript Output on page 84.
DSCHeaderComment		Use to specify PostScript Document Structure Convention (DSC) comments you want added to the header portion of the generated PostScript print stream. You can include as many DSCHeaderComment options as are necessary. See Adding DSC Comments on page 85 for more information.

Option	Values	Description

OverlayPath	any directory	Set to the directory which contains the PostScript overlays for each FAP file. The default is the FormLib option of the MasterResource control group.
		Instead of using the above control groups and options, you could use the following options:
		`< MasterResource >`
		`OverlayPath = <CONFIG:Batch Processing> OverlayPath =`
		`< CONFIG:Batch Processing >`
		`OverlayPath = .\PstOvl\`
		The default is the FormLib directory pointed to by the FormLib option in the MasterResource control group., as shown here:
		`< MasterResource >`
		`FormLib  = <CONFIG:Batch Processing> FormLib  =`
		`< CONFIG:Batch Processing >`
		`FormLib = ./forms/`

OverlayExt	any file extension (**OVL**)	The file extension of the PostScript overlays.
PageNumbers	Yes/**No**	Set to *Yes* to enable form or form set page numbering.
SendColor	Yes/**No** Enabled/ Disabled/ Hidden	Set to *Yes* to enable color printing. Enabled = Option appears in the Print window and is active (available to be checked). Disabled = Option appears in the Print window but is grayed out (not available to be checked). Hidden = Option does not appear in the Print window
DownloadFonts	**Yes**/No	Set to *Yes* to enable downloading of PostScript fonts. See also Creating Smaller PostScript Output on page 84.
PrinterModel	file name (omit extension)	Contains the name of the PostScript Printer Definition (*PPD*) file. This file contains information about printer-specific features. This file must be in the directory specified by the DefLib option of the FMRES control group.
TemplateFields	Yes/**No**	Set to Yes to test print *X*s in variable fields
FitToWidth	Yes/**No**	Not supported by the PostScript print driver

Option	Values	Description
PrintViewOnly	Yes/*No*	If set to Yes, the view only sections will print. This does not apply to entry only sections, which are never printed. Entry only sections are usually worksheets. If the section is marked as hidden and view only, it will not print.
PrePrintedPaper	Yes,Disabled	Determines if the check box which lets you print or not print pre-printed objects appears on the Print window. Also determines the default for this check box—checked or unchecked. You must add this option to the INI file if you want the check box to appear on the Print window. The default for this option includes the checkbox on the Print window and leaves it unchecked. All objects except fields can be designated as pre-printed on the object's Properties window.
Class	*(first three characters of the Module option)*	Specifies the printer classification, such as AFP, PCL, XER, PST, or GDI. If you omit this option, the system defaults to the first three letters from the Module option. Some internal functions expect a certain type of printer. For instance, all 2-up functions require an AFP printer. The internal functions check the Class option to make sure the correct printer is available before continuing.
LanguageLevel	Level1 **Level2**	Level2 is the default setting and is required for complex printing tasks, such as duplexing, tray selection, and so on. Only use Level1 if your printer only supports PostScript Level 1 language features.
StapleOn StapleOff	see description	These options work in a similar fashion to the Tray# options which let you specify PostScript commands directly as a quoted string or to look up the PostScript commands to use in your printer's PPD file. For detailed information, see Stapling Forms on page 87.
SelectRecipients	**Yes**/No **Enabled**/ Disabled/ Hidden	Enabled = Option appears in the Print window and is active (available to be checked). Disabled = Option appears in the Print window but is grayed out (not available to be checked). Hidden = Option does not appear in the Print window.

Using defaults for the Module and PrintFunc options

Default values for the Module and PrintFunc options in the PrtType:xxx control group are provided when you use a standard print type name or print class, such as AFP, PCL, PDF, PST, VPP, XER, XMP, or GDI.

These defaults keep you from having to enter the Module and PrintFunc names in your INI file. For example, if you want to generate PST print files, you can specify these INI options:

```
< Printer >
    PrtType     = MYPST
< PrtType:MYAFP >
    Class       = PST
```

And the system will default these options for you:

```
< PrtType:MYAFP >
    Module      = PSTPRT
    PrintFunc   = PSTPrint
```

Printing under Windows

Windows XP/2000 does not recognize printer ports such as LPT1. Change the PrtType control group in the FSIUSER.INI file to reflect the print server name and print device. Here is an example:

```
< PrtType:PST >
    Device = \\FSISRV03\\OPTRA1
```

Generating PostScript Files on OS/390

You can generate PostScript output files on OS/390 (MVS) systems with an updated (version 11.0 or later) PSTLIB. Be sure to include these settings in your FSISYS.INI file to print PostScript on OS/390:

```
< Printer >
    PrtType = PST
< PrtType:PST >
    Module = PSTW32
    Printfunc = PSTPrint
    SendOverlays = (Yes or No)
    SendColor = (Yes or No)
    DownloadFonts = (Yes or No)
```

Creating Smaller PostScript Output

The PostScript print driver automatically downloads (embeds) only the fonts that are needed. This results in smaller output files.

NOTE: To produce a PostScript print stream that only downloads (embeds) the minimum set of fonts required by the PostScript print stream, you *cannot* use overlays.

All PostScript fonts referenced in the FXR file are downloaded if the SendOverlays option is set to Yes because the system does not know which fonts are used by the overlays.

You must set these PostScript INI options as shown to tell the PostScript print driver to download the minimum set of fonts required by a print stream:

```
< PrtType:PST >
    DownloadFonts = Yes
    SendOverlays = No
```

If you are running the GenPrint program, you will need to tell GenPrint to load the FAP files (instead of overlays) by using the DownloadFAP option:

```
< RunMode >
    DownloadFAP = Yes
```

Bitmap compression

The PostScript print driver supports bitmap compression. Compression is enabled by default. To disable compression, add this option to the PostScript printer control group:

```
< PrtType:XXX >
    Compression = No
```

Color bitmaps are compressed in JPEG format.

Monocolor bitmaps are compressed using Run Length Encoding (RLE) compression. If compression or color is disabled, 4-bit and 8-bit color bitmaps are printed as monocolor bitmaps. For compatibility with previous releases, 24-bit color bitmaps are printed in color when compression is disabled and color is enabled.

PostScript print streams with bitmap compression are often smaller and may be produced faster than PostScript print streams without bitmap compression. PostScript print streams with compressed multicolor bitmaps will see the greatest reduction in terms of file size and time to produce.

The 4-bit and 8-bit color bitmaps printed in color with compression will likely produce larger print streams than 4-bit and 8-bit color bitmaps which have been converted to monocolor (black and white) bitmaps.

Keep in mind:

- For any bitmap to print in color, you must make sure the bitmap (LOG) is marked as *Print in Color* in the FAP file. Also make sure you set the SendColor option to Yes in the PCL or PostScript printer control group before printing.
- When using Forms Integrity Manager (FIM) to compare a version 11.2 or later PostScript print stream with bitmap compression against an older PostScript print stream without bitmap compression, FIM will report that some bitmaps are not identical. Older PostScript print streams without bitmap compression generated the bitmap data in multiple streams while the newer compressed bitmaps are always generated within a single stream. In this case, FIM will report the older print streams contains multiple *Overlay Images* entries while the new print streams contain a single *Overlay Images* entry. Also, FIM may report differences in some attributes (height, width, raster size, and so on) of *Overlay Images* and *Variable Images* due to differences in how bitmaps are emitted.

Adding DSC Comments

Use the DSCHeaderComment option to specify the PostScript Document Structure Convention (DSC) comments you want added to the header portion of the generated print stream. You can include as many DSCHeaderComment options as are necessary.

This example shows how, in addition to specifying PostScript commands in the Tray# options, you can also include DSC comments you want added to the header portion of the generated PostScript print stream:

```
< PrtType:PST >
    Device = test.ps
    DownloadFonts = Yes,Enabled
    DSCHeaderComment = %%DocumentMedia:Media1 612 792 75 (White)
(Tray1)
    DSCHeaderComment = %%+ Media2 612 792 75 (White) (Tray2)
    DSCHeaderComment = %%+ Media3 612 792 75 (White) (Tray3)
    DSCHeaderComment = %%+ Media4 612 792 75 (White) (Tray4)
    LanguageLevel = Level2
    Module = PSTW32
    PageNumbers = Yes
    PrinterModel = XDP92C2
    PrintFunc = PSTPrint
    Resolution = 300
    SendColor = No,Enabled
    Tray1 = "<< /MediaType (Tray1)/MediaColor(White) /MediaWeight
75>>
           setpagedevice"
    Tray2 = "<< /MediaType (Tray2)/MediaColor(White) /MediaWeight
75>>
           setpagedevice"
    Tray3 = "<< /MediaType (Tray3)/MediaColor(White) /MediaWeight
75>>
           setpagedevice"
    Tray4 = "<< /MediaType (Tray4)/MediaColor(White) /MediaWeight
75>>
           setpagedevice"
    SendOverlays = Yes,Enabled
```

The DSC header comments are added at the beginning of the generated PostScript print stream as shown here:

```
%!PS-Adobe-3.0
%%Title: INSUREDS COPY
%%Creator: FormMaker PostScript Driver
%%CreationDate: Thu Apr 04 17:50:57 2002
%For: INSURED
%%Pages: (atend)
%%DocumentData: Clean7Bit
%%DocumentSuppliedResources: font (atend)
%%DocumentMedia:Media1 612 792 75 (White) (Tray1)
%%+ Media2 612 792 75 (White) (Tray2)
%%+ Media3 612 792 75 (White) (Tray3)
%%+ Media4 612 792 75 (White) (Tray4)
%%EndComments
```

Stapling Forms

Use the StapleOn and StapleOff INI options in the PostScript printer control group to control staple support. These options work in a similar fashion to the Tray# INI options which let you specify PostScript commands directly as a quoted string or to look up the PostScript commands to use in your printer's PPD file.

By default, the PostScript print driver will use these commands:

```
< PrtType:PST >
...
StapleOn = "<</Staple 3 >> setpagedevice"
StapleOff = "<</Staple 0 >> setpagedevice"
```

You can override PostScript staple commands by providing an alternate PostScript command to use via the StapleOn and StapleOff options in your PostScript printer control group.

You can issue PostScript staple commands in these forms:

- A quoted string containing the PostScript commands. The quoted string should contain the appropriate PostScript commands for turning stapling on or off. Here is an example:

  ```
  StapleOn = "1 dict dup /Staple 0 put setpagedevice"
  ```

- A UI keyword from a PPD file. UI keywords represent features that commonly appear in a user interface (UI). They provide the code to invoke a user-selectable feature within the context of a print job, such as the selection of an input tray or manual feed. The entries of UI keywords are surrounded by these structure keywords:

  ```
  *OpenUI/*CloseUI or *JCLOpenUI/*JCLCloseUI
  ```

  Here is an example of an OpenUI structure for XRXFinishing:

  ```
  *OpenUI *XRXFinishing/Finishing: PickOne
  *OrderDependency: 60.0 AnySetup *XRXFinishing
  *DefaultXRXFinishing: None

  *XRXFinishing None/None: "
  1 dict dup /Staple 0 put setpagedevice"
  *End

  *XRXFinishing Single_Portrait_Staple/Single Portrait Staple: "
  2 dict dup /Staple 3 put
    dup /StapleDetails 2 dict dup /Type 1 put dup /StapleLocation
  (SinglePortrait) put
   put setpagedevice"
  *End

  *XRXFinishing Single_Landscape_Staple/Single Landscape Staple: "
  2 dict dup /Staple 3 put
    dup /StapleDetails 2 dict dup /Type 1 put dup /StapleLocation
  (SingleLandscape) put
   put setpagedevice"
  *End

  *XRXFinishing Dual_Portrait_Staple/Dual Portrait Staple: "
  2 dict dup /Staple 3 put
  ```

```
  dup /StapleDetails 2 dict dup /Type 1 put dup /StapleLocation
(DualPortrait) put
 put setpagedevice"
*End

*XRXFinishing Dual_Staple/Dual Landscape Staple: "
2 dict dup /Staple 3 put
  dup /StapleDetails 2 dict dup /Type 1 put dup /StapleLocation
(DualLandscape) put
 put setpagedevice"
*End

*?XRXFinishing: "(Unknown) = flush"

*CloseUI: *XRXFinishing
```

A PostScript Printer Definition (PPD) file is supplied with a PostScript printer. This file contains information about printer-specific features. You specify the PPD file you want to use in the PrinterModel option in your PostScript printer control group (just the file name, no drive, path, or file extension). If the PrinterModel option contains the name of a PPD file, this file must be in the directory specified in the DefLib option in the FMRes control group.

This example shows a PostScript printer group that uses a PPD file for a DocuPrint 65 printer (XRD60651.PPD) and specifies StapleOn and StapleOff options using keyword settings from the PPD file:

```
< PrtType:PST >
    ...
    PrinterModel = XRD60651
    StapleOn = *XRXFinishing Single_Portrait_Staple/Single Portrait
Staple:
    StapleOff = *XRXFinishing None/None:
```

PostScript Printer Resources

A number of resources participate in the total printing process. They reside in directories specified in the MasterResource control group.

Fonts

The system supports PostScript Type 1 fonts. These fonts must reside in the directory specified in the FontLib option in the MasterResource control group when the DownloadFonts option is set to *Yes*.

Overlays

Use the OVLCOMP utility to create PostScript overlays from FAP files. These overlays must reside in the directory specified in the OverlayPath option in the MasterResource control group when the SendOverlays option is set to *Yes*.

PostScript Printer Definition (PPD) Files

A PostScript Printer Definition (PPD) file is supplied with a PostScript printer. This file contains information about printer-specific features. If the PrinterModel option contains the name of a PPD file, this file must be in the directory specified in the DefLib option in the FMRES control group.

Using the GDI Print Driver

Skywire Software developed a Graphics Device Interface (GDI) print driver because it provides many opportunities for Windows platform users. For example, by using a GDI driver, you can now fax, since fax drivers can be installed into Windows as a GDI Windows printer driver.

Also, printing using GDI lets you print to printers that do not support any of the printer languages the system supports, such as inkjet printers. To make this driver even more useful, it includes the ability to scale output, which lets you shrink the printed output to the size of the paper.

The advantages of using the Graphics Device Interface (GDI) include:

- Ability to print to any printer attached via a Windows print driver
- Ability to print to any fax machine attached via a Windows print driver
- Ability to scale edge to edge forms to print within the printable area defined by the Windows print driver.

The disadvantages of using the Graphics Device Interface (GDI) include:

- Print quality is often poorer
- Inability to print a mixture of portrait and landscape forms
- Inability to print a mixture of simplex and duplex forms
- Inability to address the same printable area available when using our native print drivers.

Note: If you do not specify the option for sending color to a GDI printer, the system converts color (4-, 8-, or 24-bit) graphics into monochrome before sending them to the printer driver. Depending on the bitmap, this conversion from color to monochrome may not yield acceptable results. Be sure to consider your printers capabilities when you are creating graphics.

If you elect to send color, including color graphics, to a GDI printer that does not support color, the printer driver determines what to do. Some ignore the color commands (printing in black), and some apply a gray-scale adjustment to the output to simulate the color changes. Some GDI printer drivers cannot accept color commands at all. If printing to your Windows-attached printer causes a program fault, or print failure, try turning off the Send Color option via Skywire Software's Print window and sending the output again.

How it works

Most Windows applications print using the Windows GDI application programming interface. Essentially, the application uses commands similar to display commands to send print commands to the operating system. Windows, in turn, sends the commands to the currently installed Windows printer driver.

Note: Printer manufacturers provide Windows printer drivers for their printers. These come on install disks from the manufacturer, or sometimes ship with Windows itself. Other types of drivers (such as fax drivers) can be installed as Windows printer drivers.

When a Windows program talks to the operating system using GDI, printer commands are not emitted in the native language of the printer by the program. The program *prints* to Windows, and Windows then *prints* to the installed printer driver.

The printer driver then produces the native printer language commands, including the bitmap font definitions. If the printer driver belongs to a PCL printer, the print driver issues PCL commands, including fonts. In contrast, our PCL printer modules produce the PCL commands and fonts.

When you use Skywire Software's GDI driver, a Windows print driver will use the Windows screen fonts to print the document with its goal being to make the document look like it does on your screen.

Understanding the System

In Documaker implementations, users typically decide what fonts they want to use and then install those fonts on the production printer. Skywire Software applications try to make the screen look like the printed output, not the other way around. Information from the production printer fonts is loaded into the font cross-reference file. The system uses this information to try to represent the printer fonts on screen. The system can also convert production printer fonts into PCL bitmap fonts. The PCL fonts the system produces look like the fonts used on your production printer.

GDI print quality, by definition, is based on the fonts used for display. The attributes which describe fonts in the font cross-reference file determine which screen fonts are used. The screen fonts used determine what you see on the screen and how GDI printed output will look.

So, the key to improving GDI print is to improve the fonts used in the display system. Some of this can be improved by making sure the font's character widths and family name is correct. There are INI options for improving the screen font substitutions, if names cannot be matched up.

For the best results, you should use exact matching screen fonts. The system comes with a set of TrueType fonts that match the printer fonts included with the system. Install and use these fonts for best results.

NOTE: If you are instead working backward from existing production fonts, as is often the case, either an approximation must take place, or you have to find screen fonts built from the printer fonts.

GDI Printer Driver INI Options

You define the necessary printer options to print using the GDI printer driver. These options specify GDI output and are located in a PrtType:xxx control group, such as *PrtType:GDI*. Common GDI options are shown below, with default values in bold:

Option	Values	Description
Device	any file or device name	Not used by the GDI print driver.
Module	GDIW32	The name of the program module which contains the system's GDI print driver. See also the Class option. See also Using defaults for the Module and PrintFunc options on page 94.
PrintFunc	GDIPrint	The name of the program function that is the main entry point into the system's GDI print driver. See also Using defaults for the Module and PrintFunc options on page 94.
Resolution	**300**	Not used by the GDI print driver.
SendOverlays	Yes/**No**	Not used by the GDI print driver.
OverlayPath	any directory	Not used by the GDI print driver.
OverlayExt	any file extension (**OVL**)	Not used by the GDI print driver.
PageNumbers	Yes/**No**	Set to *Yes* to enable form or form set page numbering.
SendColor	Yes/**No** Enabled/ Disabled/Hidden	Set to *Yes* to enable color printing. Enabled = Option appears in the Print window and is active (available to be checked). Disabled = Option appears in the Print window but is grayed out (not available to be checked). Hidden = Option does not appear in the Print window
DownloadFonts	**Yes**/No	Not used by the GDI print driver.
FitToWidth	Yes/**No**	Scale pages to fit on the paper. This option will, if necessary, reduce the size of the page. It will not increase it.
TemplateFields	Yes/**No**	Set to Yes to test print *X*s in variable fields.

Option	Values	Description
SelectRecipients	**Yes**/No **Enabled**/ Disabled/Hidden	Enabled = Option appears in the Print window and is active (available to be checked). Disabled = Option appears in the Print window but is grayed out (not available to be checked). Hidden = Option does not appear in the Print window.
PrintViewOnly	Yes/*No*	If set to Yes, the view only sections will print. This does not apply to entry only sections, which are never printed. Entry only sections are usually worksheets. If the section is marked as hidden and view only, it will not print.
PrePrintedPaper	Yes,Disabled	Determines if the check box which lets you print or not print pre-printed objects appears on the Print window. Also determines the default for this check box—checked or unchecked. You must add this option to the INI file if you want the check box to appear on the Print window. The default for this option includes the checkbox on the Print window and leaves it unchecked. All objects except fields can be designated as pre-printed on the object's Properties window.
Class	(*first three characters of the Module option*)	Specifies the printer classification, such as AFP, PCL, XER, PST, or GDI. If you omit this option, the system defaults to the first three letters from the Module option. Some internal functions expect a certain type of printer. For instance, all 2-up functions require an AFP printer. The internal functions check the Class option to make sure the correct printer is available before continuing.
SuppressDlg	Yes/**No**	Set to *Yes* to suppress the Windows Print window.
GDIDevice		Specifies the Windows printer name. Click Start, Settings, Control Panel, Printers to see a list of the printers you can choose from. If you set the SuppressDlg option to *Yes* and leave this option blank, the system suppresses the Print window and automatically prints to the default printer.

Include these options in your FSISYS.INI file (for Documaker Workstation) and FAPCOMP.INI files (for Docucreate).

In addition, you can add the following INI setting to automatically select landscape mode when printing any of the specified sections:

```
< VBPrtOptions >
   Landscape = (list of landscape sections)
```

Beside the Landscape option, list the sections you want printed landscape. Separate each section with a comma.

Users can override this option at print time.

Understanding the System

If you do not set the SuppressDlg option to *Yes*, the Windows Print window appears when you use the print device to spool the job. If you omit the SuppressDlg option or set it to *No*, the user can select which Windows print device to spool the output through. By setting this option to *Yes*, the Windows Print window (not the system's Printer window which normally appears first), will be automatically completed for the user.

If you set the SuppressDlg option to *Yes*, the default Windows printer is used unless the GDIDevice option specifies a printer. You can use the GDIDevice option to name a specific Windows print device for spooling the raw output. The name you specify must match one of the installed printers. You can see these printer names by going to the Control Panel and clicking the Printers icon.

If you misspell the printer name or specify one not installed for the GDIDevice option, the system will send the output to the default printer device or you will get an error and printing will stop. On Windows, an incorrect setting sends the raw output to spool to the default printer device.

Don't confuse the *SuppressDlg* option with the *SuppressDialog* option in the Printer control group in the FSISYS.INI file. The *SuppressDialog* option suppresses the system's internal Printer Selection window—the one that names which PrtType:XXX group from the INI file you wish to use. The *SuppressDlg* option suppresses the operating system's (Windows 32-bit) Printer Selection window.

Using defaults for the Module and PrintFunc options

Default values for the Module and PrintFunc options in the PrtType:xxx control group are provided when you use a standard print type name or print class, such as AFP, PCL, PDF, PST, VPP, XER, XMP, or GDI.

These defaults keep you from having to enter the Module and PrintFunc names in your INI file. For example, if you want to generate GDI print files, you can specify these INI options:

```
< Printer >
    PrtType     = MYGDI
< PrtType:MYAFP >
    Class       = GDI
```

And the system will default these options for you:

```
< PrtType:MYAFP >
    Module      = GDIPRT
    PrintFunc   = GDIPrint
```

Avoiding Problems with FAX Drivers

Use the FullSupport option to prevent problems with FAX drivers which can occur when you are printing from Documaker Workstation or PPS.

The GDI driver first looks for this INI option in the control group whose name reflects the Windows print driver, such as *HP LaserJet 4050 Series PS*.

If the FullSupport option is set to Yes, the GDI driver assumes the Windows print driver contains full print support and can handle form sets with mixed simplex and duplex forms (some FAX drivers crash when presented these kinds of forms).

Here is an example:

```
< HP LaserJet 4050 Series PS >
    FullSupport = Yes
```

If not found there, the GDI driver looks for the FullSupport option in the control group for the printer type, such as PrtType:GDI. If you place the FullSupport option in the PrtType:GDI control group, it serves as a default for all GDI printers. Putting the option in for specific devices overrides this default.

Batch Printing to Files

You can use the GDI print driver to print to a file by adding the PrintToFile option in your GDI printer control group. This lets you direct output to the path and file you specify — equivalent to checking the Print to File field on the Print window.

```
< PtrType:GDI >
    PrintToFile = Yes
```

Option	Description
PrintToFile	Enter Yes to have the GDI print driver use the Port options as the output printfile names for each batch when running GenPrint. The default is No.

In the GenPrint program, output printfile names for each batch are specified using the Port INI option. When you use the GenPrint program with most Documaker print drivers, the Port option determines the name of the print stream created for each batch.

Normally, the GDI print driver prints directly to a Windows print driver and does not create files written to disk. By setting the PrintToFile option to Yes in your GDI printer control group, the GDI print driver creates a print stream for each batch based on the names specified in the Port options — just like the other Documaker print drivers.

Because the Documaker GDI print driver is not designed for batch print, these additional GDI print options are recommended when you set the PrintToFile to Yes:

```
< PtrType:GDI >
...
    SuppressDialog = Yes
    GDIDevice = (Windows printer name)
    FullSupport = Yes
```

Option	Description
SuppressDialog	Enter Yes to suppress the Windows Print window from appearing.
GDIDevice	Enter the name of the Windows print driver you want to use.
FullSupport	Enter Yes to tell the Windows driver to fully support duplexing, tray selection, and so on.

This feature is limited to using the GDI driver with GenPrint (multi-step batch print) to produce output print files and is limited to simple GenPrint (batch print) environments.

Keep in mind that all normal GDI print limitations (fidelity, tray selection, duplexing, and so on) apply, plus the following:

- Banner page processing may not work.
- Cannot use the SetDeviceName and BreakBatch DAL functions.
- Callback functions may not work.
- Single step processing does not work correctly (all transactions are printed to a single file).
- Multiple driver routers may not work.
- Printing from the Image Editor may work but the Device setting will be used to create the file. Printing from Documaker Workstation may not work.
- Printing to fax drivers, email drivers, and so on may not work and other types of print or print features not previously discussed may not work.

In other words, trying to use PrintToFile option with anything except GenPrint running in a simple batch mode using a normal Windows print driver is not supported.

Using Pass-through Printing

There are some problems which occur when you print to LPT1 on Windows platforms. One problem occurs if you run Netware Client 32 for Windows 95. Although you can open LPT1 from the system, you may receive errors when printing large amounts of data. Downloading PCL fonts usually causes this.

Another problem occurs when a print queue adds additional printer commands to system-created print jobs. This causes invalid output to be sent to the printer. The HP 5si print driver can cause this problem. Another problem affects other software which redirects printers and expects all print output to use the Windows GDI mechanism.

Skywire Software provides a GDI print driver that uses Windows-native calls for printing, which is how most applications print under Windows. However, the Windows system print drivers have problems handling some system printing requirements, such as enhanced font selection, the ability to combine duplexing with landscape forms, and so on.

To solve these problems, Skywire Software print drivers can produce the commands for controlling the printer while still using an installed Windows printer device. To use this feature, leave the Device field blank on the Print window, where you select the printer driver you want to use.

Leave this field blank

Normally, the Device field contains the name of the device (LPT1) or the name of the file (D:\OUTPUT.PCL) the system should print to. When you leave this field blank, you tell the print driver you want to print through an installed printer device. After you click Ok, the Windows Print window appears so you can select which printer device to send the print job.

This printer device must be associated with a printer supported by the system's print driver. If you have a printer device available that is associated with a printer not supported by the system's print driver, the results are unpredictable. For example, if you select PCL as the system printer type (print driver), but choose a printer device associated with an AFP printer, the AFP printer will not understand the PCL output and will print garbage.

Unlike the GDI driver, Skywire Software print drivers control the printed output. The Windows Print window is the standard print window provided by Windows. Skywire Software applications cannot control or change this window. In addition, since Skywire Software's print driver is controlling the printer, most of the options on the Windows Print window will be ignored. The only options you can use are:

- Select a printer device.
- Select the Cancel button and the print process is canceled.
- Check the Print to File field and the system will print the document to the file you specify.

NOTE: Not all Windows print drivers support pass-through printing. If you receive an error while printing in this manner, you are probably using a Windows print driver that does not support pass-through printing.

Creating PDF Files

Adobe Systems created the Portable Document Format (PDF). It is the native file format of the Adobe Acrobat family of products. The original PDF file format was version 1.0.

The system produces PDF files which adhere to PDF file format version 1.3 (or version 1.4 if 128-bit encryption is used). This version supports compression and page-at-a-time downloading. With page-at-a-time downloading (byte-serving), a web server sends only the requested page of information to the user, not the entire PDF document.

Note: When you use Acrobat Reader to view a PDF document, you do not have to do anything to make it download a page at a time. Acrobat Reader and the web server handle this for you.

If you want the entire PDF document to continue downloading in the background while you view the first page in Acrobat Reader, choose File, General Preferences and select the Allow Background Download of Entire File option.

For additional information about creating PDF files with Skywire Software applications, please refer to the following documents:

For	See
Docupresentment	Please see the Internet Document Server User Guide for more information on PDF support.
Documaker Server (z/OS)	Please see the additional configuration steps in the Documaker Server Installation Guide.
For all other products and for general PDF information	Please see Using the PDF Print Driver.

Creating RTF Files

The RTF print driver lets you create a medium-fidelity export of the contents of a form set in a format you view or edit with most popular word processors. The email print driver uses this capability to email form sets. See Emailing a Print File on page 103 for more information.

To use the RTF print driver, you need these INI settings:

```
< Printers >
    PrtType     = RTF
< PrtType:RTF >
    Module      = RTFW32
    PrintFunc   = RTFPrint
```

You will also need to specify an output device name on the Print window.

Note: The RTF print driver does not support graphics (bitmaps), charts, or bar codes.

Generating separate files

You can generate separate files for each transaction when you choose RTF (or PDF) from WIP or batch print.

The name of the files will have a rolling number appended to the end of the name that starts the process and is filled in on the Print window. This is automatically handled and you do not have to set INI options to get the WIP or batch print to work as long as your PrtType name is PrtType:RTF.

There are several INI options you can use to override the naming process and also name other print drivers that require this unique handling.

```
< BatchPrint >
    NoBatchSupport = RTF
    PreLoadRequired= RTF
```

These are the default settings and cannot be overridden. However, you can specify other PrtType print driver definitions you want to fall into these same categories.

Option	Description
NoBatchSupport	Indicates that the named PrtType items, separated by semicolon, do not really support batch transactions and require special handling.
PreLoadRequired	Lets you specify all the PrtType items, separated by semicolon, that should be forced to load the form set prior to the starting print. Most print drivers don't require this special requirement, but some, such as PDF do.

Also, you can name PrtType specific items under the BatchPrint control group to override the normal Device naming option. Here is an example:

```
< BatchPrint >
    PDF = ~HEXTIME .PDF
    RTF = ~HEXTIME --~KeyID .RTF
```

Any batch print sent to PrtType:PDF (picking PDF on the Print window) will override the name and store the current hexadecimal date and time, such as BCF09CA4.PDF, which is an eight-character name, as the name of each transaction's output.

Also, you can combine INI built-in calls as shown in the RTF example. Here any WIP or batch print sent to RTF will name the files using the HEXTIME and the KeyID from the WIP transaction. This will result in names similar to this: BCF099A4-123456.RTF

Note that you must leave a space after the built-in INI function name for it to work properly. That space will not appear in the resulting output name.

Adding or removing frames

By default, the RTF print driver uses frames to replicate the look of a document. If you do not want the frames, which print as boxes around the various document objects, to appear, set the WriteFrames option to No.

```
< PrtType:RTF >
    WriteFrames =
```

For instance, you can use the RTF driver to print form sets to an RTF file. Once the RTF file is created, you can then open it in a word processor. To avoid having frames in the file, you would set this option to No.

Creating form fields

You can use the RTF print driver to convert variable fields into RTF form fields. For example, a variable address field is converted into an RTF form field. The format of the field is retained. If, for example, the address field contained all uppercase characters, this would be reflected in the corresponding RTF form field.

To print form fields, include this INI option:

```
< PrtType:RTF >
    AllowInput = Yes
```

NOTE: This works with print types RTF and RTF_NoFrame.

You may also need to include the WordTimeFormats and WordDateFormats control groups. You can use these control groups in case you are using a time or date format that has no equivalent in Word. The following groups and options let you map a Documaker format to a Word format.

```
< WordTimeFormats >
    hh:mm XM =
< WordDateFormats >
    bD/bM/YY =
```

To the left of the equals sign, you list the Documaker format used on the section. To the right, you list the Word format you want to use.

Setting margins

The RTF print driver produces margins by calculating what is required and putting the result in the RTF output. You can, however, set minimum required margins using the RTF print type control group.

You must set the minimum required margins in FAP units (2400 dots per inch). Here are the default settings:

```
< PrtType:RTF >
   MinTopMargin    = 400
   MinLeftMargin   = 600
   MinRightMargin  = 600
   MinBottomMargin= 400
```

Margin values specified in the INI file override those set in the FAP file if the page margins in the FAP file are smaller.

NOTE: The changes in the margins are noticeable when you open the document in an application such as Microsoft Word. You will see the left and right margins shifting based on what you specified in the INI file. The top and bottom margins (seen on the left side of the page) will also vary based on what you specified in the INI file.

Removing the contents of headers and footers

Use these options to remove the contents, including graphics and text, from headers and footers when creating RTF files:

```
< PrtType:RTF >
   EmptyFooters = Yes
   EmptyHeaders = Yes
```

Option	Description
EmptyHeaders	Enter Yes to remove the contents from any headers in the file. This includes both text and graphics. The default is No.
EmptyFooters	Enter Yes to remove the contents from any footers in the file. This includes both text and graphics. The default is No.

Emailing a Print File

The system lets you set up an RTF (Rich Text Format) print driver which lets you create a print-ready file that you can email to another user. The recipients can immediately print the file.

NOTE: If you have the Internet Document Server, you can also use the included PDF print driver to create print-ready files you can email.

You install the email print driver (EPTLIB) by setting up INI options so the system will know how to use the driver. Since EPTLIB is essentially a *wrapper* for a real print driver, the INI options must also include a reference to the actual print driver the system will use to create the print-ready file, such as the PDFLIB or PCLLIB. There are also INI options for the email processing, in addition to the regular email INI options.

Creating EPTLIB print files for Documaker Workstation

The INI options for EPTLIB are as follows:

```
< Printers >
    PrtType = EPT
```

This option lets the system know that EPTLIB is a print driver so it will include it on the Print window when you print from Documaker Workstation.

You can use the PrtType:EPT control group to further customize the email print driver. For instance, you can add subject and message information and use the email address book when printing from Documaker Workstation using the EPT print driver. This lets you select print, choose form set (form or page), then select the EPT print type.

The system would then display the email address book. You select the recipients and a window appears into which you can enter the subject and message text. You then choose to send or cancel the message.

Here is an example of the INI options you would set up:

```
< PrtType:EPT >
    Device      =
    Filename    = EPTFILE.RTF
    InitFunc    = EPTInit
    KeepFile    = No
    Message     = Please respond ASAP
    Module      = EPTW32
    PrintFunc   = EPTPrint
    PrtType     = RTF
    RecipFunc   = CSTSetMailRecipgvm
    RecipMod    = CSTW32
    Recipient   =
    Subject     = New Application
    TermFunc    = EPTTerm
    KeepFile    = No
```

Creating EPTLIB print files for Documaker Server

Set up your INI options as shown here:

```
< Printer >
   PrtType = EPT
< PrtType:EPT >
   Module = EPTW32
   PrintFunc = EPTPrint
   InitFunc = EPTInit
   TermFunc = EPTTerm
```

These options tell the system which functions to call to execute the printing process.

```
PrtType = RTF
```

This tells the EPTLIB print driver which real print driver to use to create the print-ready file. If omitted, it defaults to the RTF print driver (Rich Text Format).

```
FileName = EPTFILE.RTF
```

This option gives the name of the output file to create. This is only used if the Device Name field is empty in the GUI print window (the batch file name is used for GenPrint). If the device name is empty and the FileName option is omitted, a temporary file name is used. Use a file name with an extension that matches the print driver type, such as *RTF*. For GenPrint, the file name is the name of the print batch.

```
KeepFile = No
```

The KeepFile option tells EPTLIB whether or not to keep the output file after it has been emailed. The default is No.

```
< Print >
   CallbackFunc = MultiFilePrint
   MultiFileLog = data\rtflog.dat
```

These options tell the system to divide large RTF files into smaller RTF files. If you omit these options, you will be able to view the first transaction, but not the following ones.

The RTFLOG.DAT file stores the information that defines which RTF file contains which transaction for which batch.

```
Recipient = Email Recipient
Subject = File from Documaker User
Message = PDF file attached
```

Use these INI options to set mail settings for EPTLIB. The Subject and Message options specify the Subject line and Message text for the email message. For the Recipient option, you can either include the actual email recipient or you can specify a field name where the system can go to look up the recipient. Here are some examples:

```
Recipient = Stephen Petersen; send to internal email recipient
Recipient = spetersen@skywire.com; send to Internet email address
Recipient = Fieldname:ADDRESS2; use text in ADDRESS2 field
```

If the email system cannot resolve recipients, or if you leave the Recipient option blank, an email address window appears so you can select an email address from the address book. The field lookup is a feature of the default recipient function in EPTLIB, which you can replace using these INI options:

```
RecipMod = CSTW32
RecipFunc = CSTSetMailRecip
```

These options tell the system which module and function to use to determine the recipient. Omit these options and the system uses EPTLIB's default recipient function.

The CSTSetMailRecip function displays a window which shows the subject and message text and lets you edit this text. This window also lets you provide the email recipient for Documaker Workstation. Documaker Server lets you use these functions to set up recipients:

```
RecipMod = CUSW32
RecipFunc = CUSSetMailRecip
```

or

```
RecipFunc = CUSSetMailRecipGVM
```

Function	Description

CUSSetMailRecip	This function finds the print recipient and looks up the recipient in the RECIP_MAIL control group to get the email address of the recipient. Here is an example:
	`< RECIP_MAIL >`
	`AGENT = myagent@sampco.com`
	`COMPANY = support@sampco.com`

CUSSetMailRecipGVM	This function finds the recipient in a global variable, the name of which is defined in this INI option:
	`< PrtType:EPT >`
	`Recipient = EAddress`
	Instead of using EAddress as the recipient name, the system uses it as the variable name to look up to find the recipient name. This global variable can have any name.

The recipient functions have the following syntax:

```
DWORD _VMMAPI EPTDefSetRecipient(VMMHANDLE objectH,
                                char FAR * recip,
                                size_t len);
```

Parameter	Description
objectH	The object being printed (form set, form, or page)
recip	The recipient buffer
len	Length of the buffer, currently 80 characters

The return value should be SUCCESS or FAILURE. If FAILURE, then the message is not sent and FAILURE is returned from EPTPrint. To set the recipient function without INI options, use the EPTSetRecipFunc function:

```
EPTRECIPFUNC _VMMAPI EPTSetRecipFunc(EPTRECIPFUNC newfunc);
```

Call it with the address of the recipient function:

```
EPTSetRecipFunc(func);
```

The EPTSetRecipFunc function returns the previous installed function, which can be used to set it back.

Creating PDF print files

If you are creating PDF files, use these INI options:

```
< Printers >
    PrtType     = PDF
< PrtType:PDF >
    Module      = PDFW32
    PrintFunc   = PDFPrint
```

Keep in mind that when the PDF driver is called from the EPT driver, the current printer control group remains PrtType:EPT, not PrtType:PDF. Therefore, unless you add PDF-specific options, the system uses the INI settings it finds for PrtType:EPT.

Many print options, such as the DownloadFonts option, are set before the system calls EPT, which then redirects the print to another driver. So, to have the system use the correct PDF options, set your PrtType:EPT control group to look like this:

```
< PrtType:EPT >
    PrtType         = PDF
    DownloadFonts   = [PrtType:PDF] DownloadFonts =
    SendColor       = [PrtType:PDF] SendColor =
```

This way, if you change the options in the PrtType:PDF control group, those changes are automatically picked up in the PrtType:EPT control group.

Overriding attached files

Keep in mind that the EPT (email print) driver can use the FSRSetFileAttachment API. This lets you create custom hooks to override the attached file and handle situations where you need to remove the attached file but still send the message.

Using email aliases

Multiple recipient addresses are not supported with the EPT PrtType. If you need to send an email to, for instance, all agents, use an Email Application Server, such as Microsoft Exchange (MailType = MSM) or ccMail (MailType = CCM). With these products you can define an alias to represent a group of email addresses. You cannot set the MailType option to SMTP unless your SMTP server understands aliases.

Email Application Servers usually run on top of an SMTP service and let you manage email messaging more efficiently. When using an application such as Exchange, you can create a group (such as *TestGroup*) and you can specify the group name when you specify the Recipient option.

For example, if you set the MailType option to MSM in the Mail control group and you have this defined for the Recipient option:

```
< PrtType:EPT >
    Recipient = TestGroup
```

This option is sent to the Exchange server which converts the alias (*TestGroup*) into its SMTP equivalent value, such as a list of email address similar to this:

```
hbean@skywire.com;jgaramond@skywire.com;tbottle@skywire.com...
```

The result is a message sent to the entire group represented by *TestGroup*.

NOTE: To use this feature, you must set up email-related INI options. These options are discussed in the Documaker Workstation Supervisor Guide.

Choosing the Paper Size

The system supports a variety of paper sizes including US and international sizes. The following tables show the paper sizes you can choose from:

- US Standard Sizes on page 108
- ISO Sizes on page 109
- Japanese Standard Sizes on page 112

You can also find the following related information in this topic:

- Printer Support for Paper Sizes on page 113
- Paper Sizes for AFP Printers on page 117

Note: Please note that the NA file stores the actual section height and width for custom sized sections. This information is stored in the SIZE entry in the NAFILE.DAT file. Here is an example:

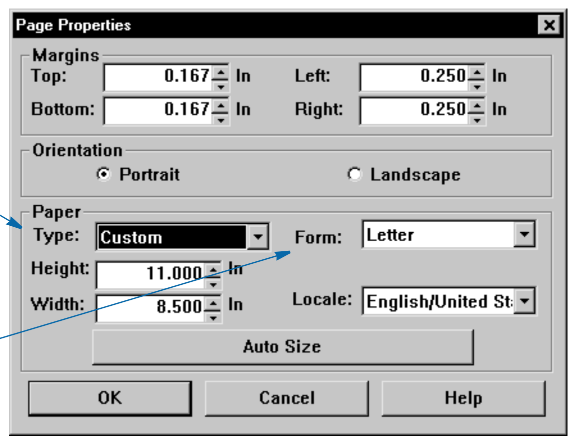
```
\NA=q1snam,LN=1,DUP=LB,SIZE=3360x18600,TRAY=U,X=600,Y=600...
```

The height and width are in FAP units (2400 per inch).

The Page Properties window in Image Editor (choose Format, Page Properties) includes a field called Form. This field is available when you set the paper type to Custom.

If you choose Custom here

the Form field is available



The Form field helps identify what type of paper to use when you print the custom section. For example, if the dimensions of the custom section are 12x14, the system needs to know what size paper to use when you do a test print of the section.

The system defaults to the size of paper that will contain the custom section, but you must tell it what paper is installed on your printer. For sections small enough to fit on letter size paper, the system defaults to letter.

Note: This affects section printing from Documaker Studio and Image Editor but has no effect on Form Set Manager or Form (FOR) definitions.

US Standard Sizes

These paper sizes are commonly used in the United States and Canada. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

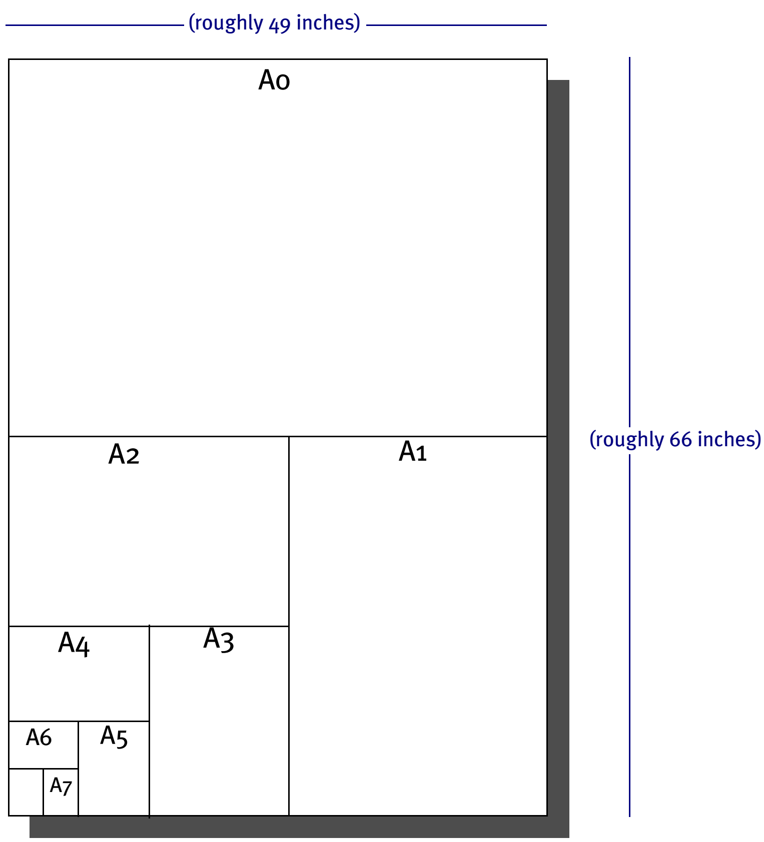
		Width x Height		
Name	**Code**	**FAP units**	**Millimeters**	**Inches (approximate)**
US letter	0	20400 x 26400	216 × 279	8½ x 11
US legal	1	20400 x 33600	216 × 356	8½ x 14
US executive	3	17400 x 25200	190 × 254	7¼ 10½
US ledger	4	40800 x 26400	432 x 279	17 x 11
US tabloid	5	26400 x 40800	279 × 432	11 x 17
US statement	6	13200 x 20400	140 x 216	5½ x 8½
US folio	7	20400 x 31200	216 x 330	8½ x 13
US fanfold	8	35700 x 26400	378 x 279	14⅞ x 11
Custom	98	any x any	any x any	any x any

ISO Sizes

The International Organization for Standardization (ISO) paper sizes, which are based on the earlier Deutsche Industrie Norm (DIN) sizes, are used throughout the world except in Canada, the United States, and Japan. There are three main series of paper sizes: A, B, and C.

ISO A sizes

The A series of sizes are typically used for correspondence, books, brochures, and other printed materials. This diagram shows most of the various A sizes. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.



		Width x Height		
Name	**Code**	**FAP units**	**Millimeters**	**Inches (approximate)**
ISO A0	20	79464 x 112345	841 x 1189	33⅛ x 46¼
ISO A1	21	56125 x 79464	594 x 841	23⅜ x 33⅛
ISO A2	22	39685 x 56125	420 x 594	16½ x 23⅜
ISO A3	23	28063 x 39685	297 x 420	11¾ x 16½

		Width x Height		
Name	Code	FAP units	Millimeters	Inches (approximate)
ISO A4	2	19842 x 28063	210 x 297	8¼ x 11¾
ISO A5	25	13984 x 19842	148 x 210	5⅞ x 8¼
ISO A6	26	9921 x 13984	105 x 148	4⅛ x 5⅞
ISO A7	27	6992 x 9921	74 x 105	2⅞ x 4⅛
ISO A8	28	4913 x 6992	52 x 74	2 x 2⅞
ISO A9	29	3496 x 4913	37 x 52	1½ x 2
ISO A10	30	2457 x 3496	26 x 37	1 x 1½
ISO 2A	32	112345 x 158927	1189 x 1682	46¾ x 66¼
ISO 4A	34	158927 x 224690	1682 x 2378	66¼ x 93⅝

ISO B sizes

The B series of sizes are designed primarily for posters, wall charts, and similar items where the difference between each A size represents too large a jump. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

		Width x Height		
Name	Code	FAP units	Millimeters	Inches (approximate)
ISO B0	40	94487 x 133605	1000 x 1414	39⅛ x 55⅛
ISO B1	41	66802 x 94487	707 x 1000	27⅞ x 39⅛
ISO B2	42	47244 x 66802	500 x 707	19⅝ x 27⅞
ISO B3	43	33354 x 47244	353 x 500	13⅞ x 19⅝
ISO B4	44	23622 x 33354	250 x 353	9⅞ x 13⅞
ISO B5	45	16630 x 23622	176 x 250	7 x 9⅞
ISO B6	46	11811 x 16630	125 x 176	5 x 7
ISO B7	47	8315 x 11811	88 x 125	3½ x 5
ISO B8	48	5858 x 8315	62 x 88	2½ x 3½
ISO B9	49	4157 x 5858	44 x 62	1¾ x 2½
ISO B10	50	2929 x 4157	31 x 44	1¼ x 1¾

		Width x Height		
Name	**Code**	**FAP units**	**Millimeters**	**Inches (approximate)**
ISO 2B	52	133605 x 188974	1414 x 2000	55¾ x 78¾
ISO 4B	54	188974 x 267209	2000 x 2828	78¾ x 111¼

ISO C sizes

The C series of sizes are designed for making envelopes and folders to take the A series of sizes. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

		Width x Height		
Name	**Code**	**FAP units**	**Millimeters**	**Inches (approximate)**
ISO C0	60	86645 x 122550	917 x 1297	36⅛ x 51
ISO C1	61	61228 x 86645	648 x 917	25½ x 36
ISO C2	62	43275 x 61228	458 x 648	18 x 25½
ISO C3	63	30614 x 43275	324 x 458	12¾ x 18
ISO C4	64	21638 x 30614	229 x 324	9 x 12¾
ISO C5	65	15307 x 21638	162 x 229	6⅜ x 9
ISO C6	66	10772 x 15307	114 x 162	4½ x 6⅜
ISO C7	67	7653 x 10772	81 x 114	3¼ x 4½
ISO C8	68	5386 x 7653	57 x 81	2¼ x 3¼
ISO C9	69	3779 x 5386	40 x 57	1⅝ x 2¼
ISO C10	70	2646 x 3779	28 x 40	1⅛ x 1⅝
ISO DL	71	10394 x 20787	110 × 220	4⅓ x 8⅔

The DL size is for a sheet 1/3 of the A4 size. This is the most common size of envelope.

Japanese Standard Sizes

Japan has its own standard paper sizes, called the Japan Industrial Standard (JIS). The JIS A series is identical in size to the ISO A series. The JIS B series, however, does not match the ISO B series. There is no equivalent to the ISO C series. This table shows the JIS paper sizes. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

		Width x Height		
Name	Code	FAP units	Millimeters	Inches (approximate)
JIS B0	80	97322 x 137573	1030 x 1456	40½ x 57¼
JIS B1	81	68787 x 97322	728 x 1030	28¾ x 40½
JIS B2	82	48661 x 68787	515 x 728	20¼ x 28¾
JIS B3	83	34393 x 48661	364 x 515	14¼ x 20¼
JIS B4	84	24283 x 34393	257 x 364	10⅛ x 14¼
JIS B5	85	17197 x 24283	182 x 257	7¼ x 10⅛
JIS B6	86	12094 x 17197	128 x 182	5 x 7¼
JIS B7	87	8598 x 12094	91 x 128	3½ x 5
JIS B8	88	6047 x 8598	64 x 91	2½ x 3½
JIS B	89	4252 x 6047	45 x 64	1¾ x 2½
JIS B10	90	3024 x 4252	32 x 45	1¼ x 1¾

Printer Support for Paper Sizes

This table outlines the various paper sizes supported by the different print drivers. The table includes information for the PDF, RTF, HTML, Metacode, PCL 5, PCL 6, GDI, PostScript, and AFP print drivers. The PDF, RTF, HTML, and Metacode print drivers support all paper sizes.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
US letter	X	X	X	X	X	X
US Legal	X	X	X	X	X	X
US executive	X	X	X	X	X	X
US ledger	X	X	X	X	X	X
US tabloid	X	Y	US letter	X	X	X
US statement	X	JIS B5	US executive	X	X	X
US folio	X	US legal	US legal	X	X	X
US fanfold	X	US ledger	US ledger	X	X	X
ISO 4A	X	Y	US letter	US letter	US letter	C
ISO 2A	X	Y	US letter	US letter	US letter	C
ISO A0	X	Y	US letter	US letter	X	C
ISO A1	X	Y	US letter	US letter	X	C
ISO A2	X	Y	US letter	US letter	X	C
ISO A3	X	X	X	X	X	X

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 117 for more information.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
ISO A4	X	X	X	X	X	X
ISO A5	X	X	X	X	X	X
ISO A6	X	X	X	X	X	X
ISO A7	X	ISO A6	ISO C5	ISO A6	X	C
ISO A8	X	ISO A6	ISO C5	ISO A6	X	C
ISO A9	X	ISO A6	ISO C5	ISO A6	X	C
ISO A10	X	ISO A6	ISO C5	ISO A6	X	C
ISO 4B	X	Y	US letter	US letter	US letter	C
ISO 2B	X	Y	US letter	US letter	US letter	C
ISO B0	X	Y	US letter	US letter	X	C
ISO B1	X	Y	US letter	US letter	X	C
ISO B2	X	Y	US letter	US letter	X	C
ISO B3	X	Y	US letter	US letter	X	C
ISO B4	X	JIS B4	US ledger	X	X	X
ISO B5	X	JIS B5	X	X	X	X
ISO B6	X	JIS B6	ISO C5	X	X	X
ISO B7	X	ISO A6	ISO C5	ISO A6	X	C
ISO B8	X	ISO A6	ISO C5	ISO A6	X	C

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 117 for more information.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
ISO B9	X	ISO A6	ISO C5	ISO A6	X	C
ISO B10	X	ISO A6	ISO C5	ISO A6	X	C
ISO C0	X	Y	US letter	US letter	X	C
ISO C1	X	Y	US letter	US letter	X	C
ISO C2	X	Y	US letter	US letter	X	C
ISO C3	X	Y	US letter	X	X	C
ISO C4	X	JIS B4	US ledger	X	X	C
ISO C5	X	X	X	X	X	C
ISO C6	X	JIS B6	ISO C5	X	X	C
ISO C7	X	ISO A6	ISO C5	ISO A6	X	C
ISO C8	X	ISO A6	ISO C5	ISO A6	US letter	C
ISO C9	X	ISO A6	ISO C5	ISO A6	US letter	C
ISO C10	X	ISO A6	ISO C5	ISO A6	US letter	C
ISO DL	X	X	X	X	X	X
JIS B0	X	Y	US letter	US letter	X	C
JIS B1	X	Y	US letter	US letter	X	C
JIS B2	X	Y	US letter	US letter	X	C
JIS B3	X	Y	US letter	US letter	X	C

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 117 for more information.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
JIS B4	X	X	X	US fanfold	X	X
JIS B5	X	X	X	X	X	X
JIS B6	X	X	X	X	X	X
JIS B7	X	ISO A6	ISO C5	ISO A6	X	C
JIS B8	X	ISO A6	ISO C5	ISO A6	X	C
JIS B9	X	ISO A6	ISO C5	ISO A6	X	C
JIS B10	X	ISO A6	ISO C5	ISO A6	X	C

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 117 for more information.

Paper Sizes for AFP Printers

The AFP formdef source file (F1FMMST.DAT) contains support for the following paper sizes, but since this file contains support for so many paper sizes, its size could affect printer performance. To limit the effect, some of the paper sizes are commented out, as shown in this table:

Size	Commented out?
Letter	No
Legal	No
Executive	No
Ledger	Yes
Tabloid	Yes
Statement	Yes
Folio	Yes
Fanfold	Yes
ISO A3	Yes
ISO A4	No
ISO A5	Yes
ISO A6	Yes
ISO B4	Yes
ISO B5	Yes
ISO B6	Yes
ISO DL	Yes
JIS B4	Yes
JIS B5	Yes
JIS B6	Yes

Note: The F1FMMST.DAT and F1FMMST.FDF files can be found in the FMRES master resource library (MRL).

The commented source line begins with an asterisk (*). To add support for another paper size, you open the F1FMMST.DAT file and delete the asterisk at the beginning of each line that references the paper size you want to add.

Because the AFP formdef is composed on medium map names that specify page orientation, paper size, tray selection, and duplex settings, there are 31 groups of medium map settings. Each of these groups contains the 57 possible paper sizes. So, for each paper size you add, there are 31 sources lines you must *uncomment* to fully support a paper size for all orientations, trays, and duplex settings.

After you uncomment the lines that reference the paper size you want to add, run the AFPFMDEF utility to rebuild your AFP formdef file with the new information. For more information on this utility, see the Docutoolbox Reference.

Creating Print Streams for Docusave

Docusave can archive AFP, Metacode, and PCL print streams that are in a Docusave-compatible format and contain special records used to index the archive.

For AFP and Metacode, you use the OutMode option in the PrtType:AFP or XER control group to tell the GenPrint program to create a Metacode or AFP print stream in a Docusave-compatible record format. You can choose between these Docusave-compatible formats: JES2 and MRG4.

For PCL, the process is similar but there is not OutMode option to set. You include comment records in the print streams to index the archive. You can use a DAL script to add those comment records.

For details, see...

- Archiving AFP Print Streams on page 119
- Archiving Metacode Print Streams on page 120
- Archiving PCL Print Streams on page 121

Archiving AFP Print Streams

Set the OutMode option to MRG4 to produce a print stream for Docusave from non-MVS platforms.

Here is an example:

```
< PrtType:AFP >
    OutMode = MRG4
```

When you set the OutMode option to MRG4, the GenPrint program creates print stream records with a 4-byte sequence that precedes them. This sequence defines the record lengths. Records are grouped into blocks with one or more records in each block. Both records and blocks have a 4-byte sequence that precedes them, defining their length.

These length indicators are formed by taking the high-order byte of length followed by the low-order byte of length followed by two bytes of zeros.

The maximum number that can be displayed is a 16-bit quantity. The value in each includes the length of the structure itself. A one-byte data record in its own block would have five for the record length and nine for the block length. This table shows what a 3-byte record would look like:

Byte offset	Value (Hex)	Meaning
0	00	Block length high-order
1	0B	Block length low-order
2	00	Always 0
3	00	Always 0
4	00	Record length high-order
5	07	Record length low-order
6	00	Always 0

Byte offset	Value (Hex)	Meaning
7	00	Always 0
8	31	‘1’
9	32	‘2’
10	33	‘3’

In addition to using the OutMode option, you must include comment records in the print streams to index the archive. You can use a DAL script to add comment records into the print stream. Use the DocusaveScript option in the PrtType:AFP control group to have the system execute a DAL script at the times when Docusave comments can be added to the print streams.

To add Docusave comments to an AFP print stream, you must add the DocusaveScript option and the name of a DAL script to execute. The DAL script should call the AddDocusaveComment function to add a string as a Docusave comment record. Here is an example:

```
< PrtType:AFP >
    DocusaveScript = Docusave.DAL
    OutMode = MRG4
```

Archiving Metacode Print Streams

Set the OutMode option to JES2 to produce print streams under MVS or OS/390. Here is an example:

```
< PrtType:XER >
    OutMode = JES2
```

When you set the OutMode option to JES2, the GenPrint program creates print stream records that are native to a mainframe environment.

Also include comment records in the print streams to index the archive. You can use a DAL script to add comment records into the print stream. Use the DocusaveScript option in the PrtType:XER control group to have the system execute a DAL script at the times when Docusave comments can be added to the print streams.

To add Docusave comments to a Metacode AFP print stream, add the DocusaveScript option and the name of a DAL script to execute. The DAL script should call the AddDocusaveComment function to add a string as a Docusave comment record. Here is an example:

```
< PrtType:XER >
    DocusaveScript = Docusave.DAL
    OutMode = JES2
```

Archiving PCL Print Streams

Note: Docusave is adding support for archiving PCL 5 print streams. In anticipation of Docusave's PCL archive capability, Documaker version 10.2 and later can produce PCL 5 print streams with the necessary Docusave comment information.

You must include comment records in the print streams to index the archive. You can use a DAL script to add comment records into the print stream. Use the DocusaveScript option in the PrtType:PCL control group to have the system execute a DAL script when Docusave comments can be added to the print stream.

To add Docusave comments to an PCL print stream, add the DocusaveScript option and the name of a DAL script to execute. The DAL script should call the AddDocusaveComment function to add a string as a Docusave comment record.

Here is an example:

```
< PrtType:PCL >
   DocusaveScript = DOCUSAVE.DAL
```

Here is an example of what the DOCUSAVE.DAL file might look like:

```
* Add DocuSave Comment - use default: APPIDX record!
COMMENT = AppIdxRec()
PRINT_IT(COMMENT)
ADDDOCUSAVECOMMENT(COMMENT)
RETURN('FINISHED!')
```

Note: PCL 6 print streams cannot be archived into Docusave.

Using DAL Functions

For all types of print streams, you can use these DAL functions to create archive keys to use with Docusave.

Function	Description
AddDocusaveComment	Adds a Docusave comment string to the print stream
AddComent	Adds a comment string to the print stream
AppIdxRec	Gets an archive record based on APPIDX.DFD and Trigger2Archive INI settings
HaveGVM	Verifies if a GVM variable exists
SetGVM	Updates the contents of a GVM variable
GVM	Gets the contents of a GVM variable
MajorVersion	Gets the system’s major version number

Function	Description
MinorVersion	Gets the system's minor version number
PrinterClass	Gets the type of print being produced
PrinterGroup	Gets the name of the print group being used
Print_It	Debug tool to print a string to the console

For more information on these functions, see the DAL Reference.

Adding TLE Records

You can add TLE (Tag Logical Element) records into AFP print streams which can be used by some 3rd-party archive systems to archive AFP print streams in a manner similar to archiving AFP or Metacode print streams in Docusave.

You must include comment records in the print streams to index the archive. You can use a DAL script to add comment records into the print stream. Use the TLEScript option in the PrtType:AFP control group to name the DAL script to execute when TLE records can be added into the print stream. The DAL script should call the AddComment function to add a string as a TLE comment record.

The TLE comment string must include a key and a value. Separate these components with a special character. This character can be any printable character as long as it is a unique character not found in the key or value portion of the comment string.

For example, you might build a comment string using a colon (:) as a separator as in the following example:

```
PolicyNum:7SAMPCO
```

The key portion of the string is *PolicyNum*, the value portion of the string is *7SAMPCO*, and the separator character is a colon (:).

Here is an example of what TLE DAL script might look like:

```
cidlabel = 'PolicyNum'
clientid = GVM("PolicyNum")
colon = ':'
AddComment (cidlabel & colon & clientid);
RETURN('FINISHED!')
```

Notice that the key portion remains constant (PolicyNum) while the value portion changes based on the contents of the GVM variable, PolicyNum.

Add these options to the PrtType:AFP control group to enable TLE record support:

```
< PrtType:AFP >
   TLEScript    = TLE.DAL
   TLEEveryPage= No
   TLESeparator= :
```

Option	Description
TLEScript	Enter the name of the DAL script to execute.
TLESeparator	Enter the character you want to use to separate the key and value portions of the TLE comment string.
TLEEveryPage	Optional. If you enter Yes, the TLE DAL script will be executed at the start of every page. If you enter No, the TLE DAL script will be executed at the start of every form set. The default is No.

Handling Multiple Paper Trays

You can set up PCL, PostScript, GDI, AFP, and Metacode print drivers to support up to nine paper trays. Setting up nine tray printer support for the various types of printers is outlined below.

NOTE: You can also use the Form Set Manager to specify tray settings. See the Docucreate User Guide for more information.

For PCL printers

You can override PCL tray commands by providing an alternate PCL command to use. Here are the default PCL INI settings:

```
< PrtType:PCL >
    Tray1 = ~&l1H
    Tray2 = ~&l4H
    Tray3 = ~&l5H
    Tray4 = ~&l20H
    Tray5 = ~&l21H
    Tray6 = ~&l22H
    Tray7 = ~&l23H
    Tray8 = ~&l24H
    Tray9 = ~&l25H
```

When writing PCL commands as an INI setting, the tilde (~) is used as a substitute for the PCL escape character (x1B).

For PostScript printers

You can override PostScript tray commands by providing an alternate PostScript command to use. You issue PostScript tray commands in these forms:

- A quoted string containing the PostScript commands. The quoted string should contain the appropriate PostScript commands for selecting a paper tray. Here is an example:

  ```
  Tray1 = "statusdict /lettertray get exec"
  ```

- A tray number from 1 to 9. You can use tray numbers to map non-existent trays. For example, Tray5=1 maps output for tray 5 to tray 1. The system checks the INI setting for overriding Tray1 before it checks the setting for Tray2 and so on. Because of this, do not specify a tray number *less than* the tray you are overriding. For example, you should not use a setting of Tray5=6.

- A UI keyword from a PPD file. UI keywords represent features that commonly appear in a user interface (UI). They provide the code to invoke a user-selectable feature within the context of a print job, such as the selection of an input tray or manual feed. The entries of UI keywords are surrounded by these structure keywords:

  ```
  *OpenUI/*CloseUI or *JCLOpenUI/*JCLCloseUI
  ```

  Here is an example of an OpenUI structure for MediaColor:

  ```
  *OpenUI *MediaColor:  PickOne
  *OrderDependency: 30 AnySetup *MediaColor
  *DefaultMediaColor: white
  *MediaColor white: "1 dict dup /MediaColor (white) put setpagedevice"
  *MediaColor clear: "1 dict dup /MediaColor (clear) put setpagedevice"
  *MediaColor blue: "1 dict dup /MediaColor (blue) put setpagedevice"
  *MediaColor buff: "1 dict dup /MediaColor (buff) put setpagedevice"
  *MediaColor green: "1 dict dup /MediaColor (green) put setpagedevice"
  ```

```
*MediaColor goldenrod: "1 dict dup /MediaColor (goldenrod) put
setpagedevice"
*MediaColor pink: "1 dict dup /MediaColor (pink) put setpagedevice"
*MediaColor yellow: "1 dict dup /MediaColor (yellow) put
setpagedevice"
*?MediaColor: "
save
    currentpagedevice /MediaColor
        {get} stopped
            {pop pop (white)} {dup null eq {pop (white)} if} ifelse
        = flush
    restore
"
*End
*CloseUI: *MediaColor
```

Input media (paper trays) are often selected on PostScript printers by specifying PageSize, MediaColor, MediaWeight, and MediaType. In the above example, media (paper) colors were defined for white, clear, blue, and so on. If you wanted to specify that the paper assigned to tray 5 uses blue paper, you could use one of these INI settings:

```
Tray5 = *MediaColor blue:
```

or

```
Tray5 = "1 dict dup /MediaColor (blue) put setpagedevice"
```

The first uses the UI keyword in the PPD file while the second uses the actual PostScript commands in a quoted string. When you use the UI keyword in an INI setting, always include the beginning asterisk (*) and the terminating colon (:).

Here are the default PostScript INI settings:

```
< PrtType:PST >
; UI keyword is used if PPD is specified and keyword is found.
; Otherwise, quoted string is used.
Tray1="0 statusdict /setpapertray get exec"  or  Tray1=*InputSlot
Upper:
Tray2="1 statusdict /setpapertray get exec"  or  Tray2=*InputSlot
Lower:
Tray3="2 statusdict /setpapertray get exec"  or  Tray3=*InputSlot
Manual:
Tray4="3 statusdict /setpapertray get exec"  or  Tray4=*InputSlot
Envelope:
; Make trays 5 through 9 use the PostScript commands for tray 1
Tray5=1
Tray6=1
Tray7=1
Tray8=1
Tray9=1
```

For GDI printers

You can override the GDI tray commands by specifying an alternate paper tray to use. Here are the default GDI INI settings:

```
< PrtType:GDI >
    Tray1 = 1
    Tray2 = 2
    Tray3 = 3
    Tray4 = 4
```

```
    Tray5 = 1
    Tray6 = 1
    Tray7 = 1
    Tray8 = 1
    Tray9 = 1
```

For AFP printers

You can override the AFP tray commands by specifying an alternate paper tray to use. Here are the default AFP INI settings:

```
< PrtType:AFP >
    Tray1 = 1
    Tray2 = 2
    Tray3 = 3
    Tray4 = 4
    Tray5 = 1
    Tray6 = 1
    Tray7 = 1
    Tray8 = 1
    Tray9 = 1
```

For Metacode printers

You can override the Metacode tray commands by specifying an alternate tray name to use. Here are the default Metacode INI settings:

```
< PrtType:XER >
    Tray1 = MAIN
    Tray2 = AUX
    Tray3 = AUX
    Tray4 = AUX
    Tray5 = AUX
    Tray6 = AUX
    Tray7 = AUX
    Tray8 = AUX
    Tray9 = AUX
```

Including Tray Selections in a Print Stream Batch

To include the header with the tray selection in a print stream batch, the first section written or triggered to the batch must have a tray, such as Tray 1 or Tray 2, listed in its FORM.DAT file. Otherwise, the information is not written to that batch print stream. Here is an example of header information from a PostScript print stream that had these INI options:

```
< PrtType:PST >
    Tray1 =*InputSlot Upper:
    Tray2 =*InputSlot Lower:
```

Here is the example header:

```
GenericDict begin
%%BeginSetup
%%BeginFeature: *Duplex
false statusdict /setduplexmode get exec false statusdict /settumble
get exec
%%EndFeature
%%BeginFeature: *InputSlot Upper
0 statusdict /setpapertray get exec
%%EndFeature
```

Chapter 4

Customizing Your System

The system includes many features you can add to meet specific needs. The most commonly-used features are discussed in this chapter, including...

Skywire Software also has technical documentation which you can request from Support Services.

Configuring INI Files

The system is very flexible. Virtually all functions can be configured to meet your company's specific needs through the use of two *INI*, or *initialization files*. INI files are used to specify default values, and other user-defined parameters.

An INI file is simply a text file consisting of *control groups* and *options*. A control group defines the file setting and is denoted by braces (< >). Control group options appear below each control group. Option lines define control option parameters which appear after the equals sign (=).

The INI file control group syntax appears below:

```
< Control_Group >
    Option1    = Parameters
    Option2    = Parameters
    ...
    OptionN    = Parameters
```

Understanding the System

For all binary INI options which require a Yes/No or True/False value, the system looks for Y, y, T, or t. Any of these values is interpreted as Yes or True. If the value entered begins with the letter “t” the system will read the value as true. Any other value, including blank, is interpreted as No or False.

Configuring the INI files lets you specify how you want the system to function, and how it should use your library resources. Each resource library has its own INI files and INI file settings control options related to specific resource libraries.

Each resource library uses two INI files: FSISYS.INI and FSIUSER.INI.

- FSISYS.INI - controls information related to the entire system, such as system settings and program function calls.
- FSIUSER.INI - controls settings which can vary between resource libraries, such as sorting options, archival mode, and import/export ability, as well as individual user options.

Many of the options are discussed on the following pages, as they relate to specific aspects of the system you can customize. Comprehensive INI file documentation is available on the support web site. To logon, use this web address:

www.docucorp.com/doss

The support web site provides registered users with immediate access to frequently asked questions (FAQs), technical guides, product documentation, supported software, and on-line activity tracking.

Setting INI Options for Specific Master Resource Libraries

You can use a built-in INI function (~*MRL*) which lets you set up INI options for specific master resource libraries (MRLs). This is useful if you have multiple master resource libraries and you need to customize the way the system works for each resource library.

For example, assume you have these master resource libraries:

- SAMPCO

- RPEX1
- RPEX2

You can set up options such as these:

```
< Control >
    FixedPosFields = [~MRL]FixedPosFields=
< SAMPCO >
    FixedPosFields = Yes
< RPEX1 >
    FixedPosFields = No
```

If you are using the SAMPCO library while editing a form set, the Fixed Edit option will be set to *Yes*. If you switch to the RPEX1 library, the edits will float because the option is set to *No*. If you switch to RPEX2, the option defaults to whatever value the system normally defaults to, which would be *No* in this case.

With the *~MRL* built-in INI function, you can create different setups using a single set of INI files.

Adding Personal Forms Lists

If your company has a large number of forms, you may want to use the Personal Forms List feature to help data entry users quickly find the forms they need. This feature lets you define a subset of all the forms for each user. Typically, you would use this feature to create personal forms lists for each data entry user which contain the forms those users work with most often. Data entry users can still select from any available form, they are not limited to those forms listed on their personal forms list.

Setting Up Personal Forms Lists

The Personal Forms List feature is available as soon as you or a data entry user assigns one or more forms to the list. You assign forms to or remove forms from a personal list using the Personal Formset Selection window.

To add the Personal Formset Selection window to your system, you must first add the following line to the MEN.RES file, which defines the system menus:

```
MENUITEM  "&Personal..." 262 "AFEW32->AFEPersonal" "Personal Forms
List" 9
```

You can add this line, or a line similar to it, under any menu group you like. You can also change the text of the option.

Note: The ampersand (&) indicates that the next character is the accelerator for this menu option. You can omit the ampersand if you like.

Also, make sure the menu ID (262) is not already in use by another menu option—if it is, choose another ID which is unique to the menu. The 9 at the end of the menu item represents the lowest user security level which should have access to this option. Security level values range between 0 (supervisor) and 9 (anybody).

Adding Buttons to the Forms Selection Window

You can also add a button to the Forms Selection window which the data entry user can use to display the Personal Formset Selection window. Since the Personal Forms List is related to form set selection, this is a logical place to provide access.

The system includes three buttons you can use for custom (and internal) functionality. To enable a button to display the Personal Formset Selection window, add this option:

```
< AFEProcedures >
    Button1 = AFEW32->AFEPersonalEdit
```

If Button1 is already in use, substitute Button2 or Button3. After you set this up, the button appears on the Forms Selection window, just above the forms list. By default, the text on the button reads, *Personal*. You can change the text using this option:

```
< DLGTitles >
    Personal = ~text
```

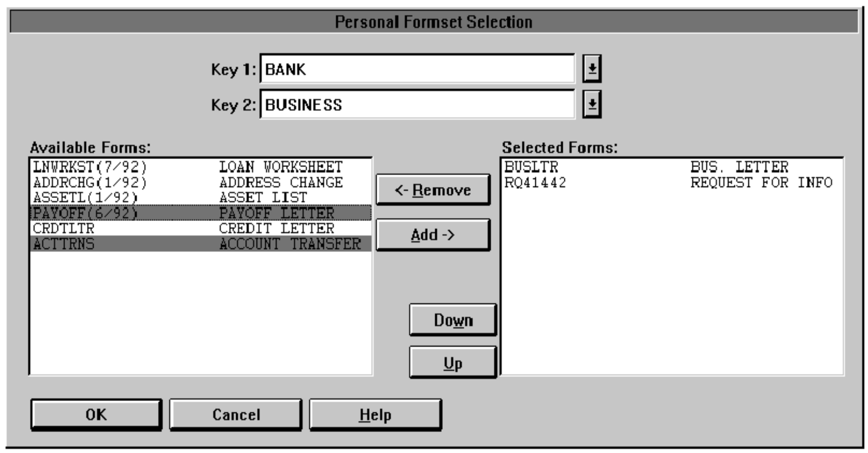
You can enter anything, but the size of the button is set so long descriptions may not fit.

Note: Use the tilde character (~) to indicate that the next character is the accelerator for the button.

Using the Personal Formset Selection Window

Customers who use this feature typically have a large number of available forms. To make it easier to select from a large number of forms, the Personal Formset Selection window contains fields for the Key1 and Key2 field combinations. You use these fields to filter the list of available forms.

After the data entry user chooses Key1 and Key2 categories, a list of forms which meet that criteria appears in the Available Forms column. Any forms in this column which are already in your personal list appear in the Selected Forms column.



To move forms into the Selected Forms column, highlight the forms in the Available Forms column and click Add. To remove forms from the Selected Forms column, highlight the forms you want to remove and click Remove.

Use the Up and Down buttons to change the order of the forms in the Selected Forms column.

You can change you entries in the Key1 and Key2 fields to display additional forms. Once you have all of the forms you want on your personal list displaying in the Selected Forms column, click OK to save that list.

You can customize the fields and column headings which appear on the Personal Formset Selection window using the following INI options. These options are shown below with their default values.

```
< DLGTitles >
    PersonalDlgTitle   = Personal Form Set Selection
    FormListTitle      = Available Forms:
    PersonalListTitle  = Selected Forms:
    UpBtnTitle         = ~Up
    DownBtnTitle       = Do~wn
    AddBtnTitle        = ~Add ->
    RemoveBtnTitle     = <- ~Remove
    Key1Title          = Key 1:
    Key2Title          = Key 2:
```

Several of these options (especially, Key1Title, Key2Title, and FormListTitle) affect other windows in the system. Keep this in mind if you customize this text.

Using the Personal Forms List

If you work in a network environment with shared resources, each user's selections must be maintained separately from those shared resources. For this reason, the Personal Forms List is stored in the FSIUSER.INI file (or whatever INI file was specified on the command line when you started the system).

Once defined, a new Key1 and Key2 combination will become the default entries on the Forms Selection window—as though it was loaded from the FORM.DAT file. The data entry user can then select these forms just as he or she would select any others.

By default, the individual forms shown on the personal list will not be checked, since they may come from any number of different form groups.

NOTE: Since the personal list is simply treated as a separate key combination, data entry users can open any form not shown on their personal list by changing the Key1/Key2 selection on the Forms Selection window.

You can also customize the names of the Key1 and Key2 groups using the following INI options. These options are shown with their default values.

```
< Personal >
    Group1 = Personal
    Group2 = Personal
```

The first option, Group1, appears as a choice in the Key1 list. Group2 appears as the selection in Key2 field.

Viewing all Lines of Business (Key2)

The Personal Forms List shows your forms for each Key2 field option. For example, in insurance implementations, the Key2 field typically indicates the line of business. By adding the following option, the system will show you all of your forms for all lines of business:

```
< AFEProcedures >
    Button1 = AFEW32->AFENewPersonalDlg
```

For example, if you have

Field	Entries
Company (Key1)	Formmaker Package
LOB (Key2)	General Liability, Property, and Inland Marine

The Button1 option lets you select forms from the General Liability, Property, and Inland Marine lines of business instead of only seeing one line of business at a time.

Assigning Form Sets to Users

The system provides several ways to customize how form sets are assigned to users. For instance, you can have the system automatically assign a form set to a user, add a menu option which makes the assignment, or customize the list of possible users.

All of these options are turned on by adding INI options. For more information, see the following topics.

Assigning Form Sets Using a Menu Option

You can add a menu option to automatically save and assign a form set to another user. This option lets you select from a list or automatically assign a form set to a specific user. The choices that appear are based on your user ID.

The Assign for Edit works similar to any assignment except a MEN.RES option is available that lets the user do the assignment while the form set is open.

Automatically Assigning Form Sets to Users

You can map users to other users so form sets can be automatically assigned by the system. All the user has to do is select the Assign option while the form set is open and the system takes care of making the assignment. To set up this feature, you use these INI options in the AutoAssign control group:

```
< AutoAssign >
    User1 = User2
    User2 = User3
```

This example tells the system that form sets from *User1* are always assigned to *User2*. Form sets from *User2* are always assigned to *User3*. The *From* user is listed on the left and the *To* user is listed on the right.

When you use this feature, the system does not display the Choose a User ID window during the assignment.

Creating an Assign-To List

You can set up the system to present only a subset of all users for selection during assignment.

The base system shows you all users when you assign a form set. By turning on this feature, you can define the users you want to appear on the Choose a User ID window, limiting the choices to only the appropriate ones.

To use this feature, you set up the AssignUserList control group similar to that shown here:

```
< AssignUserList >
    UserID = ;TOM;Thomas;
    UserID = ;ROB;Robert;
    UserID = ;JOHN;Jonathan;
    UserID = ;SMITH;Samuel;
    UserID = ;DOCUCORP;DOCUCORP;
```

Start each entry with *UserID =*. The system lists the users based on the order in which they appear in the INI file, so place the most likely choices first.

To the right of the equals sign, specify a user you want to appear on the list. The syntax for specifying the users is shown here:

```
;UserID;Name;Password;Rights;InUse;ReportsTo;Security;Message;
```

You do not have to include all of the parameters. Use semicolons at the beginning and end of the statement. Semicolons also separate parameters, so be sure to include them if you skip a parameter, as shown in this example:

```
;USER1;John Doe;;;;DOCUCORP;
```

This example defines the *UserID*, *Name*, and *ReportsTo* parameters, but leaves the *Password*, *Rights*, and *InUse* parameters blank.

Also, the *Security* and *Message* parameters were omitted after the *ReportsTo* field. Because these parameters were omitted, rather than skipped, you can also omit the semicolons that separate them.

Assigning Form Sets Using a DAL Script

You can use a MEN.RES function to automatically scan WIP and invoke a DAL script for each WIP record for that user. This lets you automatically save and then assign form sets assigned to one user to another user.

For instance, assume you set up the GenWip program to assign certain form sets to a specific user. Using this feature and a DAL script, you can then have the system automatically evaluate the form set data and then...

- Complete the form set
- Delete the form set
- Archive the form set
- Assign the form set to another user

See the DAL Reference for more information about DAL functions and DAL scripts.

Enabling the DAL Debugger

You can enable the DAL Debugger by editing the MEN.RES file used by the master resource library. You can edit this file using any ASCII text editor. Before you edit the file, make a backup copy of the MEN.RES file. Here is an example of what you need to add to the MEN.RES file:

```
POPUP        "&Tools" 255 "Utility Programs"
    BEGIN
    MENUITEM "Enab&le Debugger..."  502 "DBGW32->DBGEnableDebugger"
"Enable DAL debugger." 0
    SEPARATOR
```

REQUIRING FORMS

Use the R_Option INI option to require that any form marked as required in the FORM.DAT file be accepted by the user. Only the transaction type can override this option. Here is an example of how to set up this option:

```
< Control >
    R_Option = Required
```

There is no default for this option, so you have to define the option if you want to require the user to accept these forms.

Adding Form Description Lines

You can include a form (or several forms) which contain descriptions of the other forms included in the form set. The Form Description Line feature uses a method similar to Form Line fields on DEC pages with a few distinct differences.

- Form Description Line variable fields must have names that begin with *FORM DESC LINE*. You can include multiple lines of these fields on a form simply by varying the field's name, such as FORM DESC LINE #002, FORM DESC LINE #003, and so on.
- For each form in a form set (and, optionally, for each Key2 grouping), a Form Description Line field will be assigned a text description of that form. Unlike FORM LINEs which append all the form names together, only one text description is assigned to each Form Description Line field. You must make sure there are enough Form Description Line fields to include the maximum number of selected forms. Otherwise, some of the descriptive information may be lost.
- In addition, unlike Form Line fields, Form Description Lines do not wrap the text description to succeeding lines. If a text description is longer than the field's representation, the text can extend beyond the page boundaries or into undesirable areas. Make sure the Form Description Line fields are long enough to contain the longest description. Choosing a small font will allow the most characters on a given line.

An example of description lines generated from a sample package policy might look like this:

```
DEC PAGE              Common Policy Declarations
FIL 1010 04 92        Supplemental Declarations
CG DEC                General Liability Declarations
```

These default descriptions contain the form's name and the description provided in the FORM.DAT. Notice that this is the same information you see on the Form Selection window. If you want the information to line up appropriately, you should use a fixed pitch or non-proportional font, like Courier, to these fields.

Setting Up Form Description Lines

Unlike the Form Line fields, this feature is not limited to DEC pages. Any form can contain the Form Description Line fields. You can also place the form description lines on separate forms. You can place forms with these fields among other fields, logos, and so on.

The placement of the form in the FORM.DAT is important. Only those forms placed after the first form which contains a FORM DESC LINE field will be included in the listed forms. Please note that the first form which contains the form description lines is not included in the list. For example, assume a form set contains these forms:

Form Description Form (includes the FORM DESC LINE fields listing these forms:)

- User Letter
- Declaration Page
- Endorsement Page
- Supplementary Forms

The Form Description Form contains the form description lines which tell the system to include the descriptions of all listed forms which follow that form. If you place the Form Description Form first, the system includes the following four forms. If you place the Form Description Form after the User Letter, only the Declaration Page, Endorsement Page, and Supplementary Forms will be included.

NOTE: Since this first form containing a form description line is not included in the form list, placing the Form Description Form at the end of a form set has no affect.

Configuring INI File Settings

This feature is enabled (or disabled) by the following INI option:

```
< Control >
    DoFormDescLines = Yes
```

This option is set to *Yes* by default, so you do not have to modify the INI files to enable the option. The only reason to disable this option is if you intend to have fields with names which begin with *FORM DESC LINE* but you do not want those fields treated as form description lines. If you need to disable this option, set it to *No*. You must define this option under the Control INI control group.

Customizing Form Description Lines

There are two INI options you can use to customize form description lines.

Form description lines are formatted in a columnar format with the form name on left padded to the largest form name length used. The ColumnFormat option formats the form lines in a non-columnar format.

When using an associated table to look up form descriptions, if the form is not defined in the table, you will find no description of the form. The UseDefaultIfMissing option tells the system to use the form description from the FORM.DAT file.

Here is an example of these options:

```
< FormDescTable >
    ColumnFormat = No
    UseDefaultIfMissing = Yes
```

Option	Description
ColumnFormat	Set this option to No to have the system append the form description to the end of the form name, separated by two spaces. The default is Yes, which formats the form lines in a columnar fashion.
UseDefaultIfMissing	Enter Yes to tell the system to use the form description from the FORM.DAT file, if the entry is not found in the table. The default is No, meaning that the form name must be in the table for a description to appear.

Including Form Group Descriptions

By default, the form description lines only contain descriptions of the forms. Optionally, you can include descriptions for form groups, such as lines of business. These form groups are called *Key2*s.

These options tell the system how to represent form group lines on form description lines.

```
< FormDescTable >
    IncludeKey2     = No
    BoldKey2        = No
    Key2Prefix      =
    Key2PreInc      = 0
    Key2PostInc     = 0
    IncludeDuplicateForms = No
```

Option	Description
IncludeKey2	Use this option to enable or disable Key2 descriptions. To enable Key2 descriptions, set this option to *Yes*. The default is *No*.
BoldKey2	Use this option to present Key2 descriptions in a bold font. The system determines which font to use by querying the font defined on the field and selecting its bold equivalent. The fonts of normal Form Description Lines fields (not assigned a Key2 name) will be changed to their non-bold counterparts. If you enable this feature, the system will query the font associated with each Form Description Line field and request either the bold or non-bold equivalent from the same font family and size. If the requested font is not available, the system does not change the field's font. To choose the bold and non-bold equivalent of a font, the system uses your FXR file, which must be defined properly. Each font listed in the FXR file has a stroke weight assigned to it. The stroke weight indicates the boldness of the font.

Key2Prefix	Use this option to specify a text string which will appear before each Key2 description line. The system automatically appends a single space after the text string. By default, this option is blank and does not affect the description lines. Here is an example of how you can use this option:
	`Key2Prefix = Forms Applicable -`
	By setting the option as shown above, the system prefixes all Key2 descriptions with the specified text. For instance, the output might look like this:
	`Forms Applicable - General Liability Coverage`

Option	Description

Key2PreInc
Key2PostInc
	Use these options to add blank lines between the Key2 descriptions and the form descriptions. If you set both of these options to one (1), your output might look like this:
	`Forms Applicable - COMMON POLICY`
	`DEC PAGE          Common Policy Declarations`
	`FIL 1010 04 92    Supplemental Declarations`
	`Forms Applicable  GENERAL LIABILITY`
	`CG DEC            General Liability Declarations`
	If you include Key2 descriptions, the first text will always represent the first form grouping. This first Key2 description will not use the Key2PreInc option to include blank lines before the text. Subsequent groups, however, will have the specified number of blank lines before their text descriptions.

IncludeDuplicate Forms	When you use the Formset, Duplicate Form option to duplicate a form, the system excludes the duplicate forms from the form description lines. If you want the system to include the duplicate forms, change this option to Yes.

Excluding Form Group Descriptions

You can exclude a form or group from the Form Description lines without having to move it within the form set. This does not affect the printing of the form.

To exclude forms, use the ExcludedForm option. To exclude a group of forms (as defined in the Key2 field), use the ExcludedGroup option, as shown in this example:

```
< FormDescTable >
   ExcludedForm  = FORMDESC
   ExcludedForm  = CDEC
   ExcludedGroup = PROPERTY FORMS
```

Using Table Descriptions

One customized Form Description Line function is included in the base product. This function tells the system to look up each form name in a specified table and return the form name and description found in that table. When you use this table description method, the text items printed come from the matching table entries. Use these INI options to set up this feature.

```
< AFEProcedures >
   FormDescProc   = TRNW32->TRNFormDescTable
< FormDescTable >
   File           = tablefilename
   Table          = tablename
   IncludeFormName= Yes
   IncludeKey2Name= No
```

The AFEProcedures control group is defined in the following topic. See Customizing Form Descriptions on page 141 for more information. Here is a discussion of the options in the FormDescTable control group:

Option	Description
File	Defines the name of a system table file (located in the MasterResource TableLib directory). If you omit the file name or enter an invalid name, the function fails.
Table	This option defines the name of a specific table in the file defined by the previous option. If you omit the table name or enter an invalid table name, the function will fail. Keep in mind you do not have to attach the table to the Form Description Line fields. The INI options that enable this feature provide all the information the system requires. Therefore, you do not have to make any FAP file changes to begin using a table with existing Form Description Line fields.
IncludeFormName	When you include form description lines on a form, the system typically returns both the form name and description. The IncludeFormName option lets you omit the form name and only print the description for each form.The default is Yes. This tells the system to include the form name and description, as shown in this example: `DEC PAGE    Policy declarations for all coverage` The form's name (*DEC PAGE*) is included in the first portion of the string. The system pads the name with extra spaces equivalent to the maximum length of the form name to make sure the form description lines appear as columns of data if you use fixed pitch or non-proportional font (like Courier) to define the fields. To omit form names on the description line, set this option to No. Although this option is in the FormDescTable control group, you can use this option even when you do not use an associated table file to provide longer descriptions than those provided in the FORM.DAT.
IncludeKey2Name	Like the previous option, you can use this option to tell the system whether or not to include the Key2 group name obtained from the table in the returned description. The default is No. When you use the table feature, the system ignores the Key2Prefix option mentioned previously. The text for Key2 must appear in the table as you want it to appear on the form.If you include the Key2 name in the description text, the system pads the name with spaces up to the maximum length of a form name. This helps make sure the Form Description Lines appear as columns of data if you use a fixed pitch or non-proportional font (like Courier) to define the fields. If you do not want Key2 names included on the description line, set this option to Yes.

Creating the Table

Creating and maintaining this table must be handled by you or by the Professional Services Group. Since the table is not attached to the Form Description Line fields, use the Table Editor to maintain the description table. There are a few other rules you must follow to create and maintain a table used to store form and group names and descriptions.

Form names can consist of up to 40 characters. Therefore, the key length you assign for the table should be 40 (this is the value defaulted by the Table Editor).

Table keys are case sensitive. This means that the form and group names should be taken from the FORM.DAT exactly as they appear. If the case of the form name in the FORM.DAT does not match the key value of the corresponding entry in the table, the system will not return any text. This has the same effect as misspelling the name.

Customizing Form Descriptions

You can customize the descriptions placed in the Form Description Line fields if you do not want to use the default name and description identified in the FORM.DAT file. The Table Description method, described earlier, uses this hook to enable that feature.

The following INI setting shows you the information the system needs to identify a customization method:

```
< AFEProcedures >
    FormDescProc = module->funcname
```

Where *module* equates to a DLL name, such as TRNW32, and *funcname* is the name of the exported function in that DLL. If you enter an incorrect DLL or function name, the system displays an error message and uses the information provided in the FORM.DAT file, which is the default method.

The function called must conform to the FAPUSERPROC prototype defined in the FAPUSER.H header file. An example of this prototype is shown here:

```
FAPDW _VMMAPI TRNFormDescTable(FAPDW dwMessage,
                          FAPDW dwFAPHab,
                          FAPDW dwFAPHwnd,
               FAPDW dwObjectIdentifier,
               FAPDW dwObjectType,
               FAPDW dwInputFlag1,
               FAPDW dwInputFlag2,
               FAPDW dwInputFlag3,
               char FAR * lpszObjectName,
               char FAR * lpszFormatType,
               char FAR * lpszFormat,
               char FAR * lpszEditData,
               char FAR * lpszInputBuffer,
               char FAR * lpszOutputBuffer,
               FAPDW dwOutputBufferMaxSize,
               FAPDW FAR * lpdwOutputFlag1,
               FAPDW FAR * lpdwOutputFlag2,
               FAPDW FAR * lpdwOutputFlag3)
```

- *dwObjectIdentifier* will be a VMMHANDLE of the requested form or Key2 group.
- *dwObjectType* will be the appropriate FAP object type indicated for the item passed -- either FAP_OBJGROUP or FAP_OBJFORM.
- *lpszObjectName* will be a pointer to the name of the item passed -- either the form name or the name associated with the Key2 group.
- *lpszOutputBuffer* is a pointer to an area which will receive the output from this function.
- *dwOutputBufferMaxSize* is the maximum size of the output buffer.

All other parameters should be ignored and will be NULL.

Suppressing Duplicate Form Descriptions

You can suppress duplicate form descriptions. For instance, suppose you have several versions of what is basically the same form, but each instance of the form has a different name. For your purposes they are the same form and have the same description, even if the names are not identical. And when building the form table using the Form Description Lines feature, you would like to suppress the duplicate form descriptions. You can use the ExcludeDuplicateDescriptions option to do this:

```
< FormDescTable >
   ExcludeDuplicateDescriptions = No
```

Option	Description
ExcludeDuplicateDescriptions	Enter Yes if you want the system to look for and exclude duplicate form descriptions. The default is No.

NOTE: By default, the Form Description Line feature will eliminate duplicate forms.

Automatically Assigning Form Numbers

You can have the system automatically fill the KeyID field which is typically used to contain the form set or form number on the Form Selection window. For instance, if the KeyID contains the policy or account number, you can have the system assign these for you.

You define the set of numbers the system uses. You can also define multiple sets of numbers. This feature can be turned on or off and customized via INI file options.

Here is a summary of the available features:

- Each KeyID you define can only be used once.
- The system retrieves KeyIDs in alphanumeric order.
- You can add KeyIDs at any time.
- The system displays warnings if the number of available KeyIDs falls below an INI defined value. The warnings can be bypassed if you set this option to *-1*.
- KeyIDs can be up to 20 characters in length.
- Optionally, you can enable a feature which will ignore a KeyID you do not want the system to use and find the next available key.

You can use the number generation window to define the KeyIDs (such as policy numbers) you want the system to use. To display this window, you must first add an entry to the MEN.RES file, which defines system menus.

The system stores the series of KeyIDs in a database table. You can add additional series of KeyIDs to the table at any time. The system pulls KeyIDs from the table in alphanumeric order, not in the order in which they were created.

The system stores all defined KeyIDs in a database table named POLNUMBR. This table is stored in the directory specified by the TableLib option in the MasterResource control group. The system creates a record for each form set KeyID with these initial values:

```
Entry_name=policy number
Descrip=blank
```

When a data entry user assigns a system-generated KeyID (policy number) to a form, the system changes the description to show the USERID to which it gave the KeyID. Once the system changes the description to document which user used the KeyID, only two things can occur:

- If the form is archived, the system deletes the entry from the database table to prevent that KeyID from being assigned to another form.
- If the form is canceled before it is saved, or if the WIP is deleted before it is completed, the system releases the KeyID back to the table where it can be used again, on another form.

To the data entry user, the only change is that the system defaults a KeyID into the field, which is typically called the Policy Number field in the base system. Data entry users can override the system-generated KeyID by simply typing in a different one. If the data entry user does this, the system returns the unused KeyID it generated to the database for future use.

SETTING UP THE INI FILE

To use this feature, you must add information in the INI file. The following topics explain what you must do in these control groups:

- Control
- Window titles
- Entry procedures

Control control group

```
< Control >
   CaseSensitiveKeys = NO
```

The number generation window accepts both upper- and lowercase characters. You should, however, define the beginning and ending KeyID masks using uppercase characters. This way, the CaseSensitiveKeys option can be set either way without affecting this feature.

If you set the CaseSensitiveKeys option to No and you use lowercase letters in the form set mask, the system cannot track the numbers correctly. If you set this option to Yes, the system can use both upper- and lowercase as it generates the numbers.

```
< Control >
   AutoKeyIDWarnCount = #
```

Use the AutoKeyIDWarnCount option to set a value at which the system will display a message which tells the user the available KeyIDs, such as form set numbers, are running low. The system counts the remaining numbers in the database table each time it assigns a number.

If you omit this option, the system warns the user when he or she assigns the last number in the database table. If you do not generate a new set of KeyIDs, the system ignores any subsequent attempt to retrieve a KeyID from the database table. The user will not see another message.

If you do not want to receive any warning messages about the lack of KeyIDs, set this option to **-1**.

Window titles control group

```
< DLGTitles >
   AutoKeyIDBeginTitle = Begin
   AutoKeyIDEndTitle = End
```

Use these two options to define the text which appears on the Policy Number Generation window. The default values are *Begin* and *End*.

Entry procedures control group

```
< AFEProcedures >
    AutoKeyID = TRNW32->TRNAutoKeyIDUsrFunc
```

TRNAutoKeyIDUsrFunc is a base-supported installable procedure which handles the operations required for the automatic policy number feature. You can install a custom procedure by defining the custom DLL and exported function you want the system to call, in the same manner shown above.

NOTE: You can use multiple AutoKeyIDs. To set this up, modify your INI files as shown here:

```
< AFEProcedures >
    AutoKeyID = TRNW32->TRNMultipleAutoKeyID [MultiAutoKeyID]
    AutoKeyID = TRNW32->TRNAutoKeyIDUsrFunc

    AutoKeyID = TRNW32->TRNVerifyKeyID
```

SETTING UP THE MEN.RES FILE

To add the number generation window to the system, you must add the following line in the MEN.RES (*menu resource*) file. You can modify the name of the window, as necessary. This example uses *Policy Number Generation*.

```
SEPARATOR
MENUITEM "Policy Number &Generation..." 287 "TRNW32->TRNKeyIDMaster"
"Policy Number Generation" 0
```

The example shown above should appear on two lines: one for SEPARATOR and one for MENUITEM and the rest of the text.

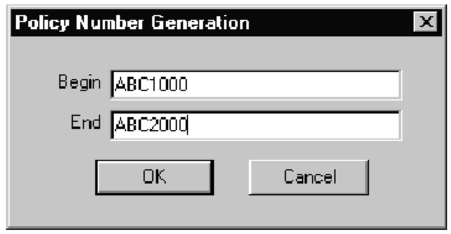
You can place the Policy Number Generation option on any menu and you can call the option anything you like. Make sure the menu ID (287) is not already in use by another menu option—if it is, choose another ID which is unique to the menu. Also, the zero (0) at the end of the menu item represents the lowest user security level which should have access to this option. Security level values range between 0 (supervisor) and 9 (typical user).

NOTE: To explain how to set up form set numbering, this manual will refer to the window as the Policy Number Generation window. If you chose another name, substitute that name as you read this topic.

GENERATING KEYIDS

If you have made the changes to the MEN.RES file as described earlier in this section, you can display the Policy Number Generation window by selecting the Policy Number Generation option from the appropriate menu.

Use the following window to create one or more series of KeyIDs.



Use the two fields in this window to specify the beginning and ending KeyID masks. All KeyIDs between the lower start value and the higher ending value will be created and stored in the database table for use.

For example, if you enter these beginning and ending values...

Begin	ABC000
End	ABC999

...the system creates a series of one thousand KeyIDs from ABC000, ABC001, ABC002, and so on, to ABC999.

Before it generates the numbers, the system examines the masks from left to right, one character at a time until a mismatch occurs. Everything to the left of that mismatch is considered the key and will be used as the first characters for every KeyID generated, in this case ABC.

All characters to the right, and including the first mismatch of characters, should consist of numbers. The system calculates the difference and uses it to create the incremental KeyIDs in the series.

Changing KeyIDs

If you need to make the system release a KeyID and assign to a form another KeyID from the database table (POLNUMBR), you can do this via the INI files. You may need to do this if either of these situations occurs:

- The data entry user cannot click Ok on the Form Selection window because the system displayed a “not unique” message as it tried to create the WIP.
- If WIP was created with a manual KeyID and now the data entry user wants to retrieve a KeyID from the automatic table.

To handle either case, you must enable the Next button on the Form Selection window by adding this INI file option:

```
< AFEProcedures >
    BUTTON1 = TRNW32->TRNAutoNextKey
```

This INI entry turns on a button normally hidden on the Form Selection window. If you have already activated BUTTON1, substitute BUTTON2 or BUTTON3.

Once activated, when a user presses Next, the system determines if the KeyID currently in use came from the database table. If it did not come from that table, the system returns a KeyID from the database table for the user to use.

If the current KeyID was assigned to the data entry user from the database table, the system displays a message which asks if it should remove the KeyID from the database table. If the user clicks *Yes*, the system deletes the current KeyID and retrieves a new KeyID from the database table. If the user clicks No, the system leaves the current KeyID in the field.

MAINTAINING THE POLNUMBR TABLE

Normally, the data entry user does not have to maintain the POLNUMBR table. If installed properly, the system handles the creation of this database table and the assignment and release of table entries.

On occasion, you may need to maintain the POLNUMBR table. To help you perform these tasks, Docucreate includes a table editor which you can also add to the system menu.

To add the Table Editor to your system menu, add these lines to your MEN.RES file:

```
SEPARATOR
MENUITEM "Table Maintenance..." 288  "TBSW32->TBSEdit" "Table
Maintenance" 0
```

You can place the Table Maintenance option on any menu you like. Make sure the menu ID (288) is not already in use by any another menu option—if it is, choose another ID which is unique to the menu.

Also, the zero (0) at the end of the menu item represents the lowest user security level which should have access to this option. Security level values range between zero (0) (supervisor) and 9 (typical user).

Follow these steps to maintain the tables:

NOTE: Make sure no other users are performing tasks which use the POLNUMBR table you are maintaining.

1 Select the Table Maintenance option. The system shows you a list of the files it found in the Master Resource table directory.

2 Select the file named POLNUMBR.DBF. There is only one table defined in this file and it too is named POLNUMBR. Choose this table and click the Edit Entries button.

3 The system displays a window which lists all entries found in this table. The first column shows you the KeyIDs which the system automatically generated. The second column shows any user IDs to which a KeyID has been assigned.

4 Add, change, or delete entries as necessary using the buttons on the window.

When you finish, be sure to close all Table Editor windows before you use the POLNUMBR table again.

Running Documaker Server

You can set up Documaker Workstation to run the Documaker Server programs (GenTrn, GenData, GenPrint, GenWIP, and GenArc) of Documaker in single-step mode.

All you have to do is add the following line to the MEN.RES file:

```
MENUITEM "&Run Rules.." 10200 "CSTW32-
>CSTRunRulesProcessor" "Run Rules Processor" 0
```

And, in the Documaker's AFGJOB.JDT file, replace

```
;JobInit1;1;;
```

with

```
;AFEJobInit;1;;
```

Keep in mind...

- Multi- and two-step mode are not supported.
- You cannot include run-time parameters.
- Documaker Workstation must be set to use the same master resources Documaker Server will use.

For more information, see the Documaker Server System Guide.

Configuring the Complete Option

When you complete a form set, the system optionally lets you perform several tasks automatically. These tasks include:

- Printing the form set (immediate, batch, or not at all)
- Exporting the form set
- Archiving the form set
- Deleting the form set

Documaker Workstation lets you specify what happens when a form set is completed by using INI options. These options are turned on or off in the FSISYS.INI or FSIUSER.INI files. If you want to hide or disable an option, include the comma before the appropriate option.

Printing the Form Set

```
< Complete >
    Print = Yes
```

This option defaults to Yes. If you set it to *No*, all print controls on the Complete window are hidden. In addition, you can configure the print options (Immediate or Batch), hiding or protecting these options so entry users cannot change them.

```
< Control >
    PrintOnComplete = Yes[No], Hidden, Disabled
    BatchOnComplete = No[Yes], Hidden, Disabled
    ImmediateOnComplete = Yes[No], Hidden, Disabled
```

The PrintOnComplete option determines whether the Print field is initially checked. If Yes, the available options are enabled. If No, the options are disabled.

NOTE: If you set this option to No and you also hide it, the result is the same as turning off the printing option using the Print option in the Complete control group.

The BatchOnComplete option affects the batch printing option. It can be hidden or disabled from change and can be initially checked or left blank. The system determines the default setting for this option by examining the DefaultPrint option in the CONTROL control group. If this option is set to BATCH, the batch printing option is enabled by default. The BatchOnComplete option overrides the DefaultPrint option.

The ImmediateOnComplete option affects the Immediate print option. It, too, can be hidden or disabled from change and can be initially checked or not checked. Like the BatchOnComplete option, the system determines the default for this setting based on The DefaultPrint option in the Control INI control group. The ImmediateOnComplete option overrides the DefaultPrint option.

If both the BatchOnComplete and ImmediateOnComplete options are checked on or off, the ImmediateOnComplete option serves as the default. If, however, one option is turned off and the other is turned on, the enabled option becomes the default, regardless of the INI setting.

NOTE: When you select batch print, the WIP transaction is not archived or deleted until that process occurs.

Exporting the Form Set

```
< Complete >
    Export = Yes
```

This option defaults to Yes. When set to No, the Export field and export list are hidden. Use this option to specify whether the Export field is initially checked, disabled, or hidden.

```
< Complete >
    ExportOnComplete = No[Yes], Hidden, Disabled
```

If the Export field is not checked but is turned off or hidden, the export list will also be hidden since it is not possible for a user to make such a selection.

Specifying recovery time from corrupted UNIQUE.* files

You may want to specify a recovery time for corrupted UNIQUE.* files if the complete process uses the form set ID from the Unique database to generate unique export file names. The LockAttempts option:

```
< DBHandler:CB5 >
LockAttempts =
```

lets you specify the length of time the system will try to recover from corrupted UNIQUE.* files. The default is 600 attempts. Since the system retries every second, the default is equivalent to five minutes of wait time.

If an open call fails, the system removes the UNIQUE.DBF and UNIQUE.MDX files and recreates them. The open call can fail because of a file lock error which forces the removal of the UNIQUE.* files.

NOTE: This INI option is not specifically limited to the UNIQUE.DBF file and would apply to any dBase IV type file you might attempt to lock. Careful consideration should be given before overriding the default setting in the INI file.

Archiving the Form Set

```
< Complete >
    Archive = Yes[No]
```

This option is on by default. When turned off, the Archive field is hidden. In addition, if turned off and the WIP is not in route, the Mode indicator is blank.

NOTE: This control has never been accessible by the user, and no INI option is provided to hide the option, other than to turn off Archive entirely.

Retaining WIP Files When Archiving

```
< Complete >
    Delete = No
```

The Delete option is turned on (Yes) by default. When you archive a form set using the Archive module, the WIP record and the associated files in the WIP directory, NAFILE.DAT and POLFILE.DAT, are deleted once the form set is archived. If the Archive module is disabled, the WIP record and the associated files are deleted when the form set is completed. Set the Delete option to No to retain the WIP files in the WIP directory. If you set this option to No, you must periodically purge the WIP either manually or through a timed or automatic process.

NOTE: This option only applies *if you are not using* the Archive module. There are no windows affected by this option.

When you use the following options in the Complete control group:

```
< Complete >
    Delete  = No
    Archive = No
```

the WIP records are updated with the status code specified in the Complete option of the Status_CD control group:

```
< Status_CD >
    Complete = CP
```

There is no default for this status code because normally a record is deleted when WIP is completed. The status code CP is shown as an example. You can define any one or two-character status code not in use by one of the other modes defined in the same INI control group.

Restricting Who Can Seed the UNIQUE file

The UNIQUE.* files contain the last number for the WIP file that was created. Whenever a WIP file is created, a number is generated to uniquely identify it to make sure no WIP file is overwritten. The GenWIP and GenArc programs use this information to create separate data and form information files for the incomplete transactions received from the GenData program and for the individual forms stored in archive.

If these files are corrupted, the next user to work with WIP is automatically given an opportunity to re-seed these files. The UniqueSeed function lets you specify who has rights to do this. To use this function, add the following line to your INI files:

```
< AFEProcedures >
    UniqueSeed = AFEW32->AFERestrictedUniqueSeed
```

The system will create a UNIQUE.DAT file if the UNIQUE.DBF and UNIQUE.MDX files are missing. If you want to always use the UNIQUE.DBF file, change the INI option which names the file to include the DBF extension, as shown here:

```
< WIPData >
    UNIQUE = UNIQUE.DBF
```

Normally this option is not present in the INI file. The name UNIQUE, without an extension, is used when the option is omitted.

If you omit the UniqueSeed function, the standard method of examining the AFELOG file is used and the file is seeded with the largest FORMSETID found in the log. If the AFELOG file is not present, the file is seeded with one (1) as the first form set.

The AFERestrictedUniqueSeed function prevents users from automatically seeding the file unless they meet specified criteria. The system displays a window stating that seeding of the UNIQUE file failed and that the user does not have authorization to perform this operation.

Adding a menu option for re-seeding the file

You can add the following MEN.RES line to set up a menu option for re-seeding the file:

```
MENUITEM "&Manual WIP Seed" 207 "AFEW32->AFEManualUniqueSeed"
"ReSeed the Unique file" 0
```

Make sure the ID of the menu option (207) does not conflict with other options. The security level you attach is up to you. Zero (0) indicates a supervisor.

With these settings, if the file is already seeded correctly and someone tries to re-seed it, the system informs the user that the UNIQUE data file is already seeded correctly.

If the file needs to be seeded, a window asks the user to enter the seed value. Enter an integer, not a hexidecimal value. The system converts the number to the proper format, if it is accepted, and displays a message stating that the operation was successful.

Customizing the Form Selection Window

You can customize the Form Selection window using INI options. These options let you control whether or not fields, columns, prompts, buttons, and titles appear and, in some cases, accept entry. Keep in mind that although you can hide these items, the data is still applied and active — the end user just can't see or manipulate it. Removing these items from view merely prevents users from changing the fields, it does not change the fields default values.

You use options in these control groups to customize the Form Selection window:

- FormSelection
- DlgTitles
- Control
- AFEProcedures
- AllowTransEdits

FormSelection options

Here is a list of the FormSelection control group options you can use to customize this window. Place this control group in either your FSISYS.INI or FSIUSER.INI file.

Option	Description
SuppressDialog	Use this option to prevent the Form Selection window from appearing when editing existing WIP.
HideTransaction	Controls whether or not the Transaction field and prompt appear. Regardless of what you choose, the default selection for the Transaction list is still active.
ShowEffectiveDate	Controls the Effective Date field on the Forms Selection window. If left blank or set to Yes, this option shows the Effective Date field and lets users create postdated form sets. If you set this option to No, the system hides the Effective Date field and uses the current date as the effective date when it creates a new form set.
HideKey1	Controls whether or not the Key1 field and prompt appear. The first item in the Key1 list will be selected.
HideKey2	Controls whether or not the Key2 field and prompt appear. The first item in the Key2 list will be selected. When Master Dec groups are used, the Key2 group that contains the Master Dec appears as the title and is automatically selected.
HideKeyID	Controls whether or not the KeyID field and prompt appear. Unless you limit users to editing existing WIP or use some type of automatic KeyID generation, you cannot leave this field blank, even when it is not visible.
HideDescription	Controls whether or not the Description field and prompt appear. When creating a form set, you must use a hook or some other method to assign a description if you want this field hidden from the user. WIP form sets use the description read from the WIP record.

Option	Description
HideImport	Removes the Import button.
HideRetrieve	Removes the Retrieval button.
HideDuplicate	Removes the Duplicate button.
HideCheckBox	Controls whether or not the column for selecting and deselecting forms in the available list appears. If you set this option to Yes, the user cannot change the forms selected.
HideMode	Controls whether or not the Mode column that identifies the entry and pull form setting appears.
HideFormName	Controls whether or not the Form Name column appears.
HideFormDescription	Controls whether or not the Form Description column appears.

HideRecip***	Use these options to hide specific recipient columns. To use, append the name of the recipient to the option, as shown here:
	`HideRecipInsured=Yes`
	`HideRecipAgent=Yes`
	`HideRecipHome Office=Yes`
	Note there is no space between *HideRecip* and the first character of the recipient name. If, however, the recipient name includes spaces, as in *Home Office*, include the spaces in the name.

ProtectRecip***	Use these options to identify recipient columns that should appear, but not accept entry (display-only). This prevents users from changing copy counts for that recipient. To use this feature, append the name of the recipient to the option, as shown here
	`ProtectRecipInsured=Yes`
	`ProtectRecipAgent=Yes`
	`ProtectRecipHome Office=Yes`
	Note there is no space between *ProtectRecip* and the first character of the recipient name. If, however, the recipient name includes spaces, as in *Home Office*, include the spaces in the name.

OnlyEditZeroRecip	This option only lets users change the recipient copy count if the copy count was set to zero (0) in the FORM.DAT file. If you are working with WIP and the form specifies a recipient not included in the FORM.DAT, or the form is not included in the FORM.DAT file, the system will let you change those recipient copy counts. When you set this option to Yes, only the editable copy counts appear, the rest are hidden.

Option	Description
FullRecipWidth	Use this option to size the recipient columns so you can read the entire recipient name on the column header. Variable space fonts may prevent you from seeing certain recipient names in their entirety, even if you turn on this option. By default, the system allocates minimal space to the recipient columns. This lets you see as many as possible and lessens the need to scroll to see additional recipients.
DescriptionRequired	Set this option to Yes if you want to require users to enter data in the Description field.
SelectByRowClick	Set this option to Yes if you want to be able to select a form on the Form Selection window by simply clicking on the form line.
CreateStartField	Use this option to select the field you want the system to assign focus when the window appears in *create* WIP mode. You can choose from: Transaction, Key1, Key2, KeyID, or Description.
UpdateStartField	Use this option to select the field you want the system to assign focus when the window appears in *update* WIP mode. You can choose from: Transaction, Key1, Key2, KeyID, or Description.

DlgTitles options

In the DlgTitles control group, you can use these options:

Option	Description
Key1Title	Changes the default prompt of a Key1 field on *all* windows.
Key2Title	Changes the default prompt of a Key2 field on *all* windows.
KeyIDTitle	Changes the default prompt of a KeyID field on *all* windows.
TransTitle	Changes the prompt of the Transaction field on the Form Selection window.
Key1TitleFS	Changes the prompt of a Key1 field on the Form Selection window.
Key2TitleFS	Changes the prompt of a Key2 field on the Form Selection window.
KeyIDTitleFS	Changes the prompt of a KeyID field on the Form Selection window.
InfoTitle	Changes the prompt of a Description field on the Form Selection window.
FormListTitle	Changes the Available Forms prompt on the Form Selection window.
CheckArchiveTitle	Changes the name of the Retrieve button on the Form Selection window.
CheckImportTitle	Changes the name of the Import button on the Form Selection window.

Option	Description
DuplicateTitle	Changes the name of the Duplicate button on the Form Selection window.
FormDlgTitle	Changes the name of the Form Selection window.

CONTROL options

In this control group, you can use these options:

Option	Description
HiddenGroupPrefix	Lets you hide certain form groups. For example, some worksheet-style form set implementations are set up to block from view certain forms that must be in the FORM.DAT file To specify, enter the prefix to use for a hidden group. Leaving the option blank overrides the default prefix of ".H." and makes all forms visible. Any group (Key1 = Company, or Key2 = LOB) in the FORM.DAT file that begins with a special prefix (by default ".H.") will not appear as a selection in the Key1 (Company) or Key2 (LOB) fields on the Form Selection window.
DoFormSelection	Use this option to allow the Form Selection window to appear when using Mode=WIP. The default behavior is not to show the Form Selection window when using Mode=WIP.
NoKey2Changes	Use this option to prevent the user from change Key2 selections when Retrieve or Import is used to create the form set, and when editing WIP.

Understanding the System

To hide an existing company or line of business, you must mark every line which has the company name or line of business with the prefix specified in the HiddenGroupPrefix option. You cannot simply mark the first line in the FORM.DAT file. The system does not extend the hidden flag to subsequent lines.

You can view images included in a company or line of business that have the hidden flag using the AddImage DAL function. Unless you use this function, those images will not display or print. For more information, see the DAL Reference.

AFEProcedures options

In the AFEProcedures control group, you can use these options:

Option	Description
Button1	Custom button hook
Button2	Custom button hook
Button3	Custom button hook
DupForm	This function handles the form duplication process. This function displays messages which indicate the success or failure of the duplication operation.

Option	Description
CheckDupForm	The CheckDupForm hook returns True or False to indicate if you can duplicate the form.

The DupForm and CheckDupForm options provide hooks which you can use to register functions for the Duplicate button.

The Form Selection window calls AFECallCheckDupFormHook at the appropriate places to determine whether the button should be active. It also calls AFECallDupFormHook to do the actual duplication.

AllowTransEdits options

In the AllowTransEdits control group, you can use these options:

Option	Description
Transaction	Normally during WIP Edit, the Transaction field cannot be edited. Use this option to identify which Transaction types can allow the field to be edited. The setting must specify the transaction code from the INI file. Separate additional codes with semicolons.
Key2	Normally during WIP Edit, the Key2 field can be edited. This can be disabled by setting the NoKey2Changes option in the Control control group to Yes. Use this option to identify which Transaction types can allow the field to be edited. Each type must specify the transaction code from the INI file. Separate additional codes with semicolons.
KeyID	Normally during WIP Edit, the KeyID field cannot be edited. Use this option to identify which Transaction types can allow the field to be edited. Each type must specify the transaction code from the INI file. Separate additional codes with semicolons.

Validating KeyID Entries

In addition to the following restrictions on KeyID values, you can make sure that data entered conforms to a specific alpha and numeric format. For instance, KeyIDs can be:

- limited by the use of the AutoKeyID table (only accepts KeyIDs listed in the table)
- limited as to whether there can be duplicates in WIP and/or Archive
- converted to uppercase (if the CaseSensitiveKeys option is set to No)
- limited to the length defined in the database. (A standard WIP file allows 20 characters for the KeyID.)

Note: KeyIDs are typically used as the policy, document, or form set number.

Beginning with version 10.2, you can use the VerifyKeyID hook to call a DAL script. Within the DAL script, the verification can be constant, or provide exceptions based on the Key1 (Company), Key2 (Line of Business), or the transaction code currently selected.

All the relevant WIP record information taken from the Form Selection window is available to the DAL script for examination. Simply use the available DAL functions like WIPKeyID, WIPKey1, or WIPFld.

NOTE: The script can retrieve WIP values, but not change them.

You must handle any error messages using the MSG function.

To install the KeyID validation hook, include these INI options.

```
< AFEProcedures >
    AutoKeyID = TRNW32->TRNVerifyKeyID
< VerifyKeyID >
    Script = KeyID.DAL
    OnCreate = Yes
    OnUpdate = No
```

Option	Description
AFEProcedures control group	
AutoKeyID	Enter **TRNW32**->**TRNVerifyKeyID** as shown above to install the KeyID validation hook.
VerifyKeyID control group	
Script	Enter the name of the script you want the system to use. Store this script in the DefLib directory specified for your master resource library (MRL). If you omit this option, a message appears on the Form Selection window. You will have to exit and correct the INI file by either defining the script or removing the hook declaration.
OnCreate	This option defaults to Yes to indicate you want to call the script when creating a new form set via the Form Selection window. To exclude newly-created form sets, set this option to No.
OnUpdate	This option defaults to No to indicate you do not want to call the script to verify the KeyID on transactions that have already been saved to WIP. To verify WIP transactions as well, set this option to Yes.

The script can do whatever evaluation is necessary for validation purposes. Here is an example DAL script that validates a KeyID using a format token string.

```
-------------------------------------------------------------------
* Define the format requirement in the fmt variable below.
* 9 - means numeric
* A - means alphabetic
* X - means alphanumeric
* * - means any character - not limited to alphabetic or numeric
* For example, if you need 4 numeric, followed by 2 alpha, followed
* by 2 numeric, followed by 2 alphanum, you would define:
* fmt = "9999AA99XA"
* The length of the overall format string is assumed to also define
* the required length of the key value.
```

```
* Note DAL does not support case sensitive string comparisons.
* Therefore, it assumes either case is sufficient and that if the
* key is required to be in uppercase, you have set the
* CaseSensitiveKeys option to No.

fmt="9999AA99XA"

* This next statement is used to get the KeyID prompt
name = GETINISTRING(,"DlgTitles", "KeyIDTitle", "Policy #");

val = WIPKeyID();
if (val = "")
* This is returned successfully because a blank key is going to
* be handled by the Form Selection window anyway.
    return("Yes");
End
#l = len(fmt);
if (#l != len(val))
    msg(name, "Length must be " & #l & '.');
    return("No");
End
* Now example each character from right to left because we
* already have the length from the earlier check.
top:
if (#l = 0)
    goto done:
end
f = sub(fmt,#l,1);
g = sub(val,#l,1);
if (f = '9')
    if (NUMERIC(g) = 0)
        msg(name, "Position "& #l & " must be numeric.");
        return("No");
    end
elseif (f = 'A')
    if (g < 'A' OR g > 'Z')
        msg(name, "Position "& #l & " must be alphabetic.");
        return("No");
    end
elseif (f = 'X')
    if (NUMERIC(g) = 0)
        if (g < 'A' OR g > 'Z')
            msg(name, "Position "& #l & " must be alphanumeric.");
            return("No");
        end
    end
elseif (f != '*')
    msg("Invalid format found at position " & #l & ".");
    return("No");
end
#l -= 1;
goto top:
done:
return("Yes");
-------------------------------------------------------------------
```

Configuring the Routing Slip Directory

Routing slips let you use an on-line email directory for sending documents. You can group recipients by department, or organize them in any way. When you send documents using routing slips, the system sends a document to each individual in the order they appear on the routing slip. The system then routes the document back to you.

You can use the following INI option to specify the location of routing slips:

```
< Mail >
    SlipTable = (path\table name)
```

You can use this option to specify:

- An alternate path for the routing slip table. In this scenario, the system uses the default slip table name (RTESLIPS), but accesses the table via another path. If you specify only the path, be sure to include a backslash at the end of the path.

  Here is an example:

  ```
  SlipTable = ..\MyPath
  ```

- An alternate file name for the routing slip table. In this scenario, the system uses the default table directory to locate the routing slip table you specify. The default table directory is specified in your master resource setup.

  In this scenario, you omit the path and enter only the table name. The extension is optional, but you may want to include it for purposes of clarity.

  Here is an example:

  ```
  SlipTable = MyTable.dbf
  ```

- Both a new path and table name for the routing slip table. The extension is optional, but you may want to include it for purposes of clarity.

  Here is an example:

  ```
  SlipTable = ..\MyPath\MyTable.dbf
  ```

Setting Up Timed Service Functions

You can use TMRLIB features to set up timed service functions. The system executes these functions based on criteria you specify. For instance, you can use these functions for a variety of purposes, some of which are listed below:

- Automatically Saving Form Sets on page 165
- Sending Imported Information to Archive on page 165
- Starting the Entry System Automatically on page 166
- Counting the WIP Records in the User's Queue on page 166
- Using Automatic WIP Import on page 174

You must set up timed service functions in your INI files as shown below:

```
< TimerFuncs >
    REF = ;STATE;URGENCY;SECONDS;DLLNAME->FuncName;\DATA
```

The semicolons (;) are required. The DATA option is optional. If you include it, be sure to precede it with a backslash (\).

REF is simply a placeholder to distinguish each entry in the TimerFuncs control group. Each listed function should have a different REF value. The actual value is not used by TMRLIB, so in most cases you can use simple ASCII numbering such as 01=, 02=, 03=, and so on, to distinguish each timer service function line.

NOTE: Although the REF value is not used by TMRLIB, keep in mind that INI files are sorted when loaded. If the sequence of the service functions is important, set up the REF values so the sorting does not change the sequence.

STATE flag

The first flag, STATE, is a mode or program status flag. The STATE, combined with URGENCY, tells the system at what point it can call the service function. This indicator only applies to service function calls which are triggered by the timer. Initialization and termination affects all service functions, regardless of the setting you enter on the registration line. This values for this flag are:

	Means...
0	The desktop is closed, no form set is currently loaded or in view.
1	The desktop is open, a form set is currently loaded or in view.
2	You can call the service function any time (*use with caution*).

NOTE: The desktop is considered opened when any form contained in the current form set (retrieved by FAPFormset) has a FAPWINDOW associated with it.

If you want the system to call a service function when the desktop is closed, enter zero (0) as the STATE. Note, that even if the desktop is closed, the system will not call a service function registered at STATE 0 (zero) if the URGENCY requirement is not also satisfied. You can set STATE to zero (0) to implement features like automatic import, or automatic WIP edit.

Enter **1** if you want the system to call a service function while the desktop is opened. Note, however, this means that a FAPWINDOW is associated with a form in the current form set. The system will not call a service function registered at STATE 1 if the URGENCY requirement is not satisfied. You can use STATE 1 to implement a feature like automatic save to WIP.

Only enter **2** with caution. The system will call any timed service function with STATE set to two (2) regardless of whether the desktop is opened or closed as long as the URGENCY requirement is satisfied. Service functions with STATE 2 should not attempt to alter current forms or change the form set management because this may cause the system to stop functioning. You should only use STATE 2 to implement features which do not hinder the data entry user and which do not affect form set management.

URGENCY flag

Once the system evaluates the STATE flag, it then evaluates the URGENCY flag. URGENCY tells the system how important it is to enforce the call to the service function. Depending on your entry, this setting tells the system to skip or delay the call if the data entry user is working with an open window or menu.

This indicator only applies to service function calls triggered by the timer. The initialization and termination processes call all registered service functions regardless of this setting. The values for this flag are:

	Means...
0	not urgent (okay to bypass if a window or menu is open)
1	rush (call as soon as possible after a window or menu is not open)
2	urgent (call even if a window or menu is open— *use with caution*)

NOTE: TMRLIB subclasses the main window of the application and looks for messages that indicate the menu is active. In addition, all child windows associated with the application window are scanned to determine if any are windows.

These tasks determine when the system should enforce the URGENCY flag.

Enter **0** to tell the system to skip the call to the service function if a system window or menu is open. The system will not call the service function again until the next time interval occurs.

Enter **1** to tell the system to call the service function if no system window or menu is active. This setting delays the call, instead of skipping it, if a window or menu is open.

The system evaluates delayed service functions approximately every second (based upon the system timer) to determine when it can safely make the call. Delaying the call lets the system make the call at the next available point in time, instead of making it wait for the next standard interval registered with the function.

NOTE: The system does not accumulate time intervals. If the delay causes the function to miss two or three time intervals, those intervals are ignored. The system will only call the function once when the URGENCY flag is satisfied.

Enter **2** with caution. With this setting, the system calls the service function regardless of whether a window or menu is active. Timed service functions with this setting should not attempt to alter current forms or change management of the form set. Doing so may cause the system to stop functioning.

Since the data entry user may be working with a window, this setting should only be used to implement features which do not affect data entry user operations. Also, avoid using these functions to open windows. This avoids a situation where as data entry user works on one task, the system suddenly switches to another task or window.

SECONDS flag

The SECONDS flag tells the system how often, in seconds, it should call the service function. You can set this time-out value to any value from 1 to 32767 (32767 seconds exceeds nine hours). If you enter zero (0), the system skips the line.

NOTE: Although you enter the time-out value in seconds, the actual time is approximated. Internally, the system maintains two time values. One value is used to test when STATE 2 functions should be called. STATE 2 functions are called whether the desktop is opened or closed. STATEs 0 and 1 use a second time value maintained by TMRLIB.

Only those functions satisfied by the current STATE flag (desktop opened or closed) will be called when the proper time interval has elapsed. When the desktop is opened or closed, the second time value is reset to zero. This guarantees the time interval associated with a function must elapse before being called when the desktop state changes. Calling a timed service function too frequently can slow the system.

DLLNAME->FuncName flag

The DLLNAME->FuncName flag lets you identify a DLL to load and an exported function to call. The DLLNAME must be a valid DLL name and FuncName much match a name which can be queried from that DLL.

Only functions named in a DLL's export list can be referenced by TMRLIB. The function you choose must conform to the FAPHANDLER function prototype.

\DATA flag

This flag is optional. Only use this flag if the timed service function requires it. There are no format requirements except it *must* begin with a backslash (\).

The system attaches function-specific data, as a string of ASCII characters, to the FSITIMERREC structure associated with the registration line. It is the service functions responsibility to verify the existence or validity of the line.

NOTE: The leading backslash does not appear in the data member of the structure.

Automatically Saving Form Sets

This timed service function lets you automatically save the current form set. To use this function, add an INI option similar to the one shown here:

```
< TimerFuncs >
   01    = ;1;1;60;AFEW32->AFEAutoSave;
```

Where the values (*;1;1;60;AFEW32->AFEAutoSave;*) represent:

;*State*;*Urgency*;*Seconds/Time-of-day*;*Function*

Value	Description
State	Always set to one (1)
Urgency	Always set to one (1). Zero (0) means the call can be skipped if the user is engaged in an operation, while 1 means to call the feature as soon as the user has finished the current operation.
Seconds or Time-of-day	Enter the number of seconds to call the function at a specific interval. The default is *60* which means the system will automatically save the form set every 60 seconds while the desktop is open and the user is not selecting a window or menu option. Or, enter the specific time of the day at which you want the system to call this function. The system uses a 24-hour clock, so enter **13:00** to indicate 1 pm.
Function	*AFEW32->AFEAutoSave* indicates the timed service function. Enter this value exactly as shown. Without this value, this feature will not work. Case is important when entering the timed service function name.

Sending Imported Information to Archive

The system also includes a timed service function which lets you automatically send imported information which meets certain criteria to manual archive. For more information on this feature, see Importing Information Directly into Archive on page 269.

Starting the Entry System Automatically

You can use a timer service function to automatically start the Entry module when a record is received or assigned to a user. You use the following timed service function and INI option to set up this feature:

```
< TimerFuncs >
    07=;0;0;100;AFEW32->AFEWipAutoCheck;
```

Where the values (*;0;0;100;AFEW32->AFEWipAutoCheck;*) represent:

;*State*;*Urgency*;*Seconds/Time-of-day*;*Function*

Value	Description
State	Always set to zero (0)
Urgency	Zero (0) means the call can be skipped if the user is engaged in an operation, while one (1) means to call the feature as soon as the user has finished the current operation.
Seconds or Time-of-day	Enter the number of seconds to call the function at a specific interval. The default is *60* which means the system will automatically check to see if there are any new form sets every 60 seconds while the desktop is open and the user is not selecting a window or menu option. Or, enter the specific time of the day at which you want the system to call this function. The system uses a 24-hour clock, so enter **13:00** to indicate 1 pm.
Function	*AFEW32->AFEWipAutoCheck* indicates the timed service function. Enter this value exactly as shown. Without this value, this feature will not work. Case *is* important when entering the timed service function name.

You can suppress the WIP List - Edit window which appears when the Entry module starts using this INI option:

```
< AFEWIPAutoCheck >
    SuppressDialog=Yes (No)
```

Counting the WIP Records in the User's Queue

The system now includes a WIP count timer function to report the number of WIP records in the user's queue.

Set up an INI option similar to the following to use this feature:

```
< TimerFuncs >
    01=;1;1;60;AFEW32->AFEWipAutoCountRec;
```

Where the values (*;1;1;60;AFEW32->AFEWipAutoCountRec;*) represent:

;*State*;*Urgency*;*Seconds/Time-of-day*;*Function*

Value	Description
State	always set to one (1)

Value	Description
Urgency	Zero (0) means the call can be skipped if the user is engaged in an operation, while one (1) means to call the feature as soon as the user has finished the current operation.
Seconds or Time-of-day	Enter the number of seconds to call the function at a specific interval. The default is *60* which means the system will automatically count the number of WIP records in the user's queue every 60 seconds while the desktop is open and the user is not selecting a window or menu option. Or, enter the specific time of the day at which you want the system to call this function. The system uses a 24-hour clock, so enter **13:00** to indicate 1 pm.
Function	*AFEW32->AFEWipAutoCountRec* indicates the timer function. Enter this value exactly as shown. Without this value, this feature will not work.

These parameters show the default for this function in the TimerFuncs control group. Using these settings, the system calls the AFEWipAutoCountRec function approximately every 60 seconds while the desktop is open and user is not engaged in a window or menu selection. If a window or menu is open, the function waits until the window or menu closes.

TROUBLESHOOTING

If you have problems using the automatic WIP import feature, check your INI settings. The system loads the FSIUSER.INI file from the current working directory—unless you specified a different INI file on the command line when you started the system. If you are starting the system by clicking an icon, check the properties for that icon to determine which INI file is being used.

Once you have determined which FSIUSER.INI file is being used, open that file in a text editor and find this control group and option:

```
< Environment >
    FSISYSINI  =   ..\FSISYS.INI
```

Your settings may vary, but this option always tells the system where to find the FSISYS.INI file. This example indicates that the file is loaded from the current working directory. If your setting has a relative path like the one shown above, you'll need to determine where the FSISYS.INI file is being read based upon your current working directory.

Determine the Location of the Option

FSIUSER.INI options take precedence over duplicate options defined in the FSISYS.INI file. For instance, suppose you have the following settings defined in your INI files.

In the FSIUSER.INI file

```
< MyGroup >
   Value = Yes
```

In the FSISYS.INI file

```
< MyGroup >
   Value = No
   File = x.tmp
```

If the system attempts to locate the current value for *Value* under *MyGroup*, the response will be *Yes*. If, however, the system attempts to locate the File option, the one defined in FSISYS.INI will be returned (x.tmp).

This means you need to examine the FSIUSER.INI file first and then the FSISYS.INI file to check which options are being used by the system. *Always* examine both files. This step shows the relationship between the FSIUSER.INI and FSISYS.INI files. At a higher level, the distinction between these two files is represented by their names—*user* options vs. *system* options.

System options are usually placed in the FSISYS.INI file. The type of options include any that must or should be the same for all users. By placing common options in the same file, you can usually be assured that everyone's system operates similarly.

The FSIUSER.INI file is where the user specific options are defined. User specific options are those which may vary between individuals. Usually this file contains the master resource directives which are used to locate the files and data required by the system. If users do not share printers, email, and other similar options, then these too may be defined in the FSIUSER.INI file.

When attempting to determine in which file to place options, ask if this applies to everyone or just certain individuals.

Check the Timer Setup

```
< TimerFuncs >
   01=;0;0;300;TRNW32->TRNAutoImport;
```

The definition above (or a variation of it) is required to activate the auto import feature. Descriptively, the line reads like this:

> Every five minutes (300 seconds), if no window or menu item is active, and a form set is not currently being edited, run the TRNAutoImport function located in the TRN??? DLL. If one of these assumptions fail, skip this call and check again at the next interval.

Valid values for the first variable after the equal sign are 0, 1, or 2. Where zero (0) means a form set is not loaded; 1 means only when a form set is loaded; and 2 means it does not matter.

For the automatic WIP import feature, this option would be zero (0) if you use the Edit=Yes feature. If the import information is complete (in and of itself), this field may be zero (0) or 1. Neither of these options allow the action to occur if a window or menu option is active when the time interval occurs. Entering the 2 is not appropriate for this feature.

The second field may also be 0, 1, or 2. Again, 2 is not a good choice. A zero (0) tells the system to skip the action if the form set is in the incorrect state or a window or menu option is active. A 1 tells the system to delay the call until the proper state is achieved.

For instance, in the example shown above, if you were to open a window at four minutes 59 seconds, the service function will be skipped and not checked again for five minutes. If you change the option to 1, the system checks the state each second after the initial attempt to see if the action can be taken. When all values are satisfactory, the system starts the automatic WIP import feature.

Check the Timer Service Functions

If you have not received any messages from the system to indicate where a problem may be, and it appears that nothing happens to your import files (like being renamed to *.BAD), test your timers.

To do this change the TimerFunc (or simply add another timerfunc) to do the following.

```
< TimerFuncs >
    01=;0;0;5;TMRW32->TMRTimerTest1;
```

The function referenced by this call will display a window each time the function is called. You should receive a window indicating the system is initializing when it first starts.

In this example, every five seconds—if a form set is not open and no window or menu option is active—another window should appear. When you exit the system, you should get a termination notice from the function.

If this test works, your timers are operational. Change the settings back to those appropriate for the Automatic WIP Import option.

Check Your AutoImport Options

There must be an AutoImport control group defined in the INI file or the feature cannot operate. The options supported by this feature are shown below:

```
< AutoImport >
    Path = c:\data\
    File =
    Match = *
    Ext = .DAT
    DeleteOnSuccess = Yes
    DeleteOnFail = No
    TransactionCode = NB
    Description = Imported
    StatusCode = WIP
    RecordType = NEW
    Edit = No
```

You must define a value for Path or File or the initialization will fail. You can omit all other options if you want to use the defaults shown above. If you specify a File setting, it must be a valid 8.3 file name. The other values allowed for these options are defined earlier in the Automatic WIP Import section.

If the File Imports But Some of the Data is Missing

This is generally a result of improper field scope. Field scopes are set in the Image Editor on the Field Property window. All fields defined in the import file after the Header Lines but before the first Form/Image Line should be defined with a “global” field scope.

All fields defined after a Form/Image line but before the next Form/Image line should have their scope set to either *form* or *image*. The deciding factor is whether an image was named by the “NA=” portion of the definition. If an image is named, the field must have an *image* scope. If an image was not named, which means only a form was specified, then any fields must have a *form* scope.

Another possibility is that the field name is not defined on the form and/or image specified. Remember importing does not create fields, it merely assigns data to existing fields which were defined with the proper scope.

If the System Does Not Import the File and does not Rename It

First, check to see if the FILE option is being used and if the file has the proper extension defined or defaulted by the AutoImport control group.

Also, check the user’s rights to the directory where the file is located. Due to potential multi-user clashes over the same file, each workstation attempts to rename the file as soon as it locates it. If the rename operation fails, the system does not attempt to import the file. Failure of the rename operation can occur if the import file has the read-only attribute or, on some networks, if the user does not have the right to rename files.

If the System Renames the Import File to *.BAD

Typically, in this situation the system displays a message to indicate what failed during the import process—unless you have the Edit option set to Yes. If the Edit option is set to Yes, the system renames the import file to *.BAD if the Form Selection window is canceled. This means the system did not accept the import.

Error messages and their meanings are discussed next. If you do not receive an error message, check the import file format. Check to make sure the header information is set up as required. Also, the header should define a valid Key1/Key2 (Company/Line of Business) which is listed in the FORM.DAT file currently in use. Likewise, any form definition lines which appear in the import file must identify the proper Key1/Key2 and form name found in the FORM.DAT file.

This error can also occur if insufficient memory is available during the WIP save operations.

Automatic WIP Import Error Messages

Here is a list of the common error messages you may receive when using the automatic WIP import feature.

Cannot load module TRNW32

This means that the DLLs associated with the function cannot be located. The actual name of the DLL being loaded will depend upon your operating system, but it will start with *TRN*.

Cannot query address of: func

Here, *func* will be the name of the function read from the INI file. Normally this error indicates you are using a version of the system which does not include the function you want to query; or the name does not match the proper case or letters required. For AutoImport, the function must be declared exactly as *TRNAutoImport*.

Error in [TimerFuncs] value in INI file Line

This message means a service function registration line under the TimerFuncs control group is not in the proper syntax. The line in question should appear as part of the error message.

Memory error in loading [TimerFuncs] values

There is not enough memory to load the service function list.

Timer function xxx failed to initialize

The named function (represented here by *xxx*) failed to initialize properly. Usually this means that one or more required options are set improperly.

Cannot register timer

This message indicates the system cannot get a timer from the operating system. Usually this indicates there are too many timers in use by other applications. Try closing other applications and starting this process again.

Function [xxx] has already been installed

Normally *xxx* will be *AutoImport*, but it may differ in some situations. This message tells you the system attempted to initialize an AutoImport control group twice. Remove the duplicate option or check the documentation on how to set up multiple imports properly. See Setting Up Multiple Import Sessions on page 258 for more information.

Invalid file name [xxx] FILE = file name

Normally *xxx* will be *AutoImport*, but it may differ in some situations. *File name* will be the file name as defined by the PATH and/or FILE options.

INI option [xxx] Path= or INI option [xxx] File= must be defined

Normally *xxx* will be *AutoImport*, but it may differ in some situations. This message tells you to specify a valid path or file name in the PATH or FILE options in the proper AutoImport control group.

Unable to create import list

Generally, this means the system is out of memory, or an internal error has occurred.

Unable to add to import list

Generally, this means the system is out of memory, or an internal error has occurred.

Function [xxx] has been requested but has not been initialized

Normally *xxx* will be *AutoImport*, but it may differ in certain situations. This message should seldom occur. Receiving this message means the service function is being asked to run before it is initialized.

Cannot allocate import structure

This message is associated with the Edit=Yes option. It indicates you do not have enough memory to perform the task or that an internal error occurred.

[xxx] StatusCode=code, WIP status code is not coded in INI

Normally *xxx* will be *AutoImport*, but it may differ in some situations. This message tells you an attempt was made to use an invalid status code. This attempt may be associated with the StatusCode option, or the improper code may exist in the import header record. Valid status codes are defined under the StatusCode control group.

[xxx] RecordType=type, WIP record type is not coded in INI

Normally *xxx* will be *AutoImport*, but it may differ in some situations. This message tells you an attempt was made to use an invalid WIP record type. This attempt may be associated with the RecordType option, or the invalid record type may exist in the import header record. Valid WIP record types are defined under the Record_Type control group.

AFEDATA WIP record not defined

This message indicates the form set failed to create a WIP record during import. If this message was preceded by another message, refer to its description of the problem. Otherwise, this may mean you do not have enough memory to perform the import. This message can also indicate an internal error occurred.

Transaction Code is missing in WIP

This message indicates that no valid transaction code was specified in either the INI file or in the import header record.

Field is missing in WIP record

This message indicates that a variable the system expected was not provided. This message may appear for the fields Key1 (Company), Key2 (Line of Business), or KeyID (Policy Number). These must be declared in the header records of the import file.

KeyID is not unique

This indicates that a form set with the same Key1+Key2+KeyID already exists and the transaction code requires unique keys (usually *New Business*). If the CheckInArc option is defined as Yes, the clashing record may have been located in the archive file.

Error loading archive file name

The archive file could not be located or opened. Check your master resource setup to determine where these files should be located and that a proper APPIDX.DFD file is available.

Error loading archive DFD file APPIDX.DFD

The archive DFD file could not be located or opened. Check your master resource setup to determine where this file should be located and that a proper APPIDX.DFD file is available.

WIPAppend failed

This is a general message which tells you the system could not successfully create or save the WIP record. This message does not indicate a specific error situation.

If you get this error, check the master resource setup to make sure you specified a WIP directory. Also, make sure that you have sufficient disk space and access rights to open and add records to this file.

CUSTOMIZING WIP

There are several ways you can customize the work-in-progress features of the system. These include:

- Using Automatic WIP Import on page 174
- Customizing WIP Windows and Options on page 183
- Limiting WIP and Batch Print to Specific Recipients on page 190
- Customizing WIP Transaction Codes on page 192
- Using ODBC WIP Indexes on page 195
- Using Compressed WIP on page 195
- Storing WIP on an SQL Server on page 196
- Deleting WIP on page 197
- Using GenData WIP Transaction Processing on page 199
- Using the WIPField Built-in function on page 200

USING AUTOMATIC WIP IMPORT

This feature lets you automatically identify and import data for user form selection and automatically import data files directly into the WIP database. This feature is designed as a *timed service function* which means you can tell the system to look for files to import at specified times during the day. If the system finds a file to import, it will then automatically import the file for you. You must set up this feature via an INI option and call it through TMRLIB.

NOTE: If you need additional customization of this feature to meet your business needs, call the Professional Services Group.

After you add the required INI file settings, the system scans the directory you specify to locate import data files. Additionally, you can tell the system to look for a specific file name. The system performs this scan on a timed interval, which you also specify in the INI file.

Once it locates an import file, the system renames the file so other workstations, also using this feature, do not attempt to import the file. After it renames the file, the system imports the renamed file using the standard import function included in the base system. This process usually takes only a few seconds.

In the (default) *automated mode*, this feature does not use the Forms Selection window. The imported form set is, however, created using logic which is similar to how the Form Selection window works. In automated mode, the user does not have to do anything — which is typically desired.

If the system cannot find the selected forms identified in the import file, it uses the default required forms specified by the FORM.DAT. The data entry user can remove any unused forms or add other forms once he or she retrieves the form set from WIP.

You can also set up this feature in *non-automated mode*, so that once the system reads the import file, it displays the Form Selection window. The data entry user then has control over which forms are included into the form set, as well as the other information provided on that window.

You can use other INI options to set up default WIP information not provided by the imported file. The system uses the current user's ID as the *creator* of the work-in-process and, in the automated mode, the system places the information into the WIP database specified by the user's current MasterResource control group options when the user chooses the WIP, Save option.

If the system cannot save the file, it renames the data file with the extension *BAD*. This will normally prevent any attempts to load this file again. You must periodically check these files to determine why the import failed and correct or delete the files with this extension. The system will alert you if this occurs.

NOTE: There is an optional INI setting you can use to automatically delete files the system cannot successfully import.

Setting Up the INI File

You must set up the Automatic WIP Import feature in your INI files before you can use it. Here is an example of how the settings look in an INI file.

```
< TimerFuncs >
    01=;0;0;300;TRNW32->TRNAutoImport;
```

Where the values (*;0;0;300;*) represent:

;*State;Urgency;Seconds/Time-of-day*;*Function*

Value	Description
State	Always set to zero (0)
Urgency	Zero (0) means the call can be skipped if the user is engaged in an operation, while one (1) means to call the feature as soon as the user has finished the current operation.
Seconds or Time-of-day	Enter the number of seconds to call the function at a specific interval. The default is *300* which means the system will automatically save the form set every five minutes while the desktop is open and the user is not selecting a window or menu option. Or, enter the specific time of the day at which you want the system to call this function. The system uses a 24-hour clock, so enter **13:00** to indicate 1 pm.
Function	**TRNW32**–>*TRNAutoImport* indicates the timed service function. Enter this value exactly as shown. Without this value, this feature will not work. Case *is* important when entering the timed service function name.

There may be occasions when users need to import data from more than one directory. For example, you may have one directory set up for form sets with a status code of *WIP* and another directory for form sets with a status code of *Batch Print*.

For information on handling this requirement, see Setting Up Multiple Import Sessions on page 258. See also Setting Up Timed Service Functions on page 162 for more information.

Alternate registrations are possible if more than one type of auto-import will be active at once. See the following topic for more information.

AutoImport Options

By default, the automatic import function (TRNAutoImport) expects to locate the AutoImport control group. In some situations, the control group can have a different name, however, the options under the control group remain the same.

This list defines the options you can specify, along with the default values, if any. You must define the Path or File options or the system will not initialize this function.

```
< AutoImport >
    Path = c:\data\
    File =
    Match = *
    Ext = .DAT
    DeleteOnSuccess = Yes
    DeleteOnFail = No
    TransactionCode = NB
    Description = Imported
    StatusCode = WIP
    RecordType = NEW
    Edit = No
```

Here is an explanation of the options:

Path option

Use the Path option to tell the system the directory path it should use to check for import data files. The directory path you enter must be valid and the data entry user must have access to it. For example:

```
PATH = c:\data\
```

If you do not specify a file, the system scans the specified directory at the timed interval for a file which matches the query you defined in the Match and Ext options. The default Ext (extension) is *DAT*. If you specify a file without a path, the system uses the Path option to identify the path for the file you specified.

NOTE: The system imports only one file at a time. If there are additional import files in the directory, the system will import the first one, and then import the next file at the next timed interval.

File option

Use the File option to tell the system to import a specific file. Enter the name of the file beside the option. If you omit this option, the system will import files which match the criteria created by the Match and Ext options.

If you enter a file name without a path, the system uses your entry for the Path option to locate the file. If you omit the extension, the system uses the extension you entered in the Ext option. If you do not enter an extension in the Ext option, the systems uses the default extension, which is *DAT*.

Match option

The system uses the Match option, combined with the Ext option, as search criteria for locating import files when the File option is not specified. The system performs the search in the directory you specify using the Path option.

This option defaults to the DOS wildcard asterisk (*), which means that any file name with the appropriate extension is a match. You can enter any valid DOS file name search value.

Ext option

Use this option to specify the file extension the import files will use. If you do not use the File option, or if the File value does not include an extension, the system looks to this option for the import file extension.

This option defaults to *DAT*. It is not recommended that you use the extension *BAD* because this can conflict with import files that fail to import properly. (Because non-successful import files are renamed to *BAD*.) Likewise, an extension consisting of a single period (dot) should not be used because this can conflict with the multi-user capability in a networked environment.

DeleteOnSuccess option

Before actually importing a file, the system renames the file to match the WIP NA and POL file it created for the form set. This option tells the system to delete files which are successfully imported. The option defaults to *Yes*.

If you want to keep the import files, enter No. The system does not rename the import file back to its original name — this prevents it from importing the files a second time. If you set this option to No, it is up to you to periodically move or delete the imported files.

DeleteOnFail option

Files the system cannot successfully import are renamed with the extension *BAD* unless you enter Yes for this option. If you enter Yes, the system deletes any files it cannot import. The default for this option is No.

NOTE: If you delete bad import files, there is no physical evidence an import was attempted other than any error message which may have appeared.

TransactionCode option

If the imported file does not contain a WIP transaction code, the system sets it to the value you specify here. The system checks the transaction code with those listed in the INI file. The default transaction code is NB (New Business).

Description option

If the imported file does not contain a WIP description, the system will set it to the value you specify with this option. The default WIP description is Imported.

StatusCode option

If the imported file does not contain a WIP StatusCode, the system will set it to the value you specify with this option. The system checks the status code with those listed in the INI file. The default status code is WIP.

By setting the WIP status code, the data entry user can control whether the form set is identified as work-in-process — which must be completed — or is marked with some other status, such as *Batch Print*.

RecordType option

If the imported file does not contain a WIP RecordType, the system sets it to the value you specify with this option. The system checks the record type with those listed in the INI file. The default value for this option is NEW.

Edit option

By default this option is set to No. This means the system saves import files directly to WIP without any data entry user interaction. If you enter Yes, the data entry user must complete the Forms Selection window before the import is considered successful.

When set to Yes, the system only recognizes this option when no form set is currently loaded (the desktop is closed) and the data entry user is not working with a menu or window. See the following discussion of TMRLIB's STATE and URGENCY flags for more information.

The data entry user can override much of the imported information on the Forms Selection window. If however, the data entry user changes the Key1 value on this window, the imported data will be lost.

When the user accepts the window (after any changes), the system will assume the import was successful. Aborting the Forms Selection window will cause the import to fail. The system will rename the import file by adding the extension *BAD* (or deleted if the DeleteOnFail option is activated).

NOTE: If the system is minimized when the system locates an import file and the Edit option is active, the system will be restored automatically before the Forms Selection window appears.

Accessing WIP Fields

You can access most standard WIP fields using the following built-in INI functions. For instance, if you want to create an export file and a PDF file and have the names for these files be identical except for the extension, you could use these functions to create a unique name for a file that does not depend on the current time, but rather on a time that does not change, such as the create or modify time.

Function	Returns the
~Key1	WIP Key1 field
~Key2	WIP Key2 field
~KeyID	WIP KeyID field
~ORIGUSER	Original WIP User ID field (the ID used to create the WIP)
~CREATETIME	WIP Create Time field. You can format this option.
~MODIFYTIME	WIP Modify Time field. You can format this option.
~ORIGFSID	Original WIP form set ID. Keep in mind when routing messages, the original form set ID is not necessarily the same as the current form set ID.
~TRANCODE	WIP Transaction Code field.
~DESC	WIP Description field.
~DATE	The current date value.
~USERID	Currently logged in user ID.
~FIELD	A field value from the form set.

Note: You can access all of the WIP fields via DAL using the WIPFld function. And, since DAL can be accessed via the ~DALRUN function, you have another method you can use to get those fields.

The system retrieves the Modify Time and Create Time from the WIP record. You can use the ~DATE function to get the current date value. You can also include a parameter to tell the system to format the date.

Keep in mind that if you are trying to use the value as part of a file name, you should only include characters that are valid in file names.

Here is an example of how to specify a date format:

```
~MODIFYTIME ;%m-%d-%Y;
```

Semicolons (:) begin and end the string that defines the date format. If you omit a semicolon, you get the hexadecimal value of the date for ~MODIFYTIME and ~CREATETIME. For the ~DATE function, you get the format specified by the DateFormat option in the Formats control group. This option defaults to:

```
%m/%d/%y
```

If you include the semicolon, but omit the format information after the semicolon, for ~MODIFYTIME and ~CREATETIME you get the format specified by the DateFormat option in the Formats control group. This option defaults to:

```
%m/%d/%y.
```

Formatting arguments

Format arguments consist of one or more codes. Begin each code with a percent sign (%). Characters that do not begin with a percent sign are copied unchanged to the output buffer.

Any character following a percent sign that is not recognized as a format code is copied to the destination—so you can enter *%%* to include a percent sign in the resulting output string.

You can choose from these format codes:

Code	Description
%d	Day of month as decimal number (01 - 31)
%H	Hour in 24-hour format (00 - 23)
%I	Hour in 12-hour format (01 - 12)
%m	Month as decimal number (01 - 12)
%M	Minute as decimal number (00 - 59)
%p	Current locale's AM/PM indicator for 12-hour clock
%S	Second as decimal number (00 - 59)
%y	Year without century, as decimal number (00 - 99)
%Y	Year with century, as decimal number
%A	Weekday name, such as Tuesday
%b	Abbreviated month name, such as Mar
%B	Full month name, such as March
%j	Day of year as decimal number (001 - 366)
%w	Weekday as decimal number (0-6, with Sunday as 0)
%@xxx	Specify language locale (where xxx is a 3-letter code that identifies one of the supported languages. For example. A format of *%@CAD%A* might produce *mardi*, the French word for Tuesday.

Here are some examples:

This format	Will result in
%m-%d-%Y	01-01-1999

This format	Will result in
The year is %Y.	The year is 1999.
Born %m/%d/%y at %I:%M %p	Born 01/01/99 at 11:57 PM

Here are some additional format attributes for certain codes:

Code	Description
#	Tells the system to suppress leading zeros for the following format codes. This flag only affects these format codes: %#d, %#H, %#I, %#j, %#m, %#M, %#S, %#w For example, if *%d* outputs *01*, using *%#d* will produce *1*. Subsequent codes are not affected unless they also have this flag.
>	Tells the system to uppercase the resulting text. This flag only affects these format codes: %>p, %>A, %>b, %>B For example, if *%A* results in *Tuesday*, using *%>A* will produce *TUESDAY*. Subsequent codes are not affected unless they also have this flag.
<	Tells the system to lowercase the resulting text. This flag affects only these codes: %<p, %<A, %<b, %<B For example, if *%b* results in *Mar*, using *%<b* will produce *mar*. Subsequent codes are not affected unless they also have this flag.
<>	Tells the system to capitalize the first letter of the resulting text. This flag affects only these codes: %<>p, %<>A, %<>b, %<>B For example, if *%p* results in *AM*, using *%<>p* will produce *Am*. Subsequent codes are not affected unless they also have this flag.

Specifying locales

When you use *%@xxx* in the format string, the *xxx* represents a 3-letter code that identifies one of the supported language locales.

Until a locale format code is encountered in the format string, the default locale (typically USD which is US English) is used. Once a locale format code is found, the locale specified remains in effect until another locale code is encountered.

For example, suppose the input date is 03-01-1999. This table shows the output from various formats:

This format	Will result in
“ %A, %B %d”	“Monday, March 01”.
“%@CAD%A %@CAD%A, %B %d”	“lundi, mars 01”
“%A, %@CAD%B %d”	“Monday, mars 01”

This format	Will result in
“%@CAD%A, %@USD%B %d”	“lundi, March 01”

Using the ~Field function

The ~Field function lets you use a quoted parameter string to name the specific field to locate within the form set. The definition of the field can name a specific image, form, and group (Key2 or Line of Business), separated by semicolons, that contains the field requested. This lets you make sure you are retrieving a specific field occurrence within the document.

Because object names, like fields, images, forms, and groups, can sometimes contain spaces or other special characters, you should enclose the entire definition in quotation marks (“). You cannot quote individual elements of the search.

Here are some examples:

This is a valid definition for the ~Field function:

```
option = ~FIELD "Field;Image;Form;Group"
```

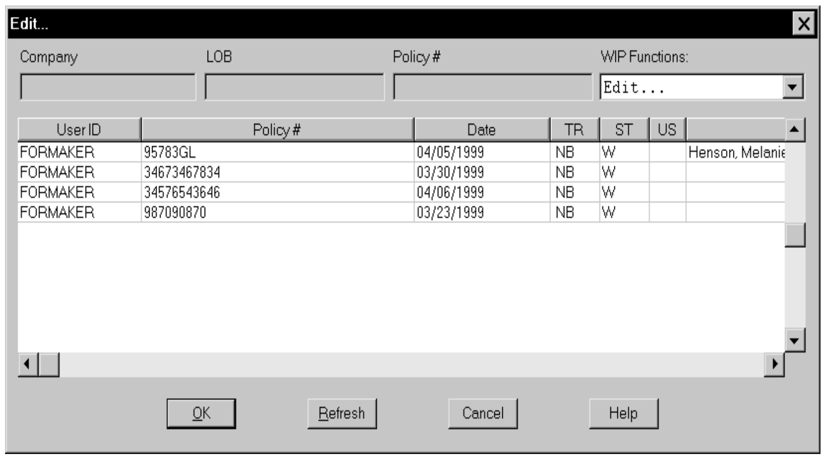
This is *not* a valid definition for the ~Field function:

```
option = ~FIELD "Field";"Image";"Form";"Group"
```

Customizing WIP Windows and Options

The WIP List option on the WIP menu provides one window from which you can select various WIP tasks. Earlier versions of the system did not include this option. Instead, menu options for each WIP task appeared on the WIP menu.

The WIP List option lets you switch back and forth between different WIP functions without having to close one window to open another. When you choose the WIP List option, the Edit window appears, as shown here:



The WIP List option does not affect the functionality of the old windows. In fact, you can use the older WIP windows and the WIP List option if you like, but we recommend using one or the other.

If you prefer to use the WIP menu option approach, see Using the WIP Menu Options on page 184 for more information.

If you are using the WIP List and the Automatic Return feature, see the following topic for more information.

NOTE: A range of MEN.RES menu IDs (400-499) are active only when in retrieval mode, such as when you view a form set. You can use these IDs to customize the system and set up options only available during retrieval.

Setting Column Widths on the WIP List Window

You can set the column widths on the WIP List window to a specific size using INI settings. Every field in this INI control group could have the optional setting:

```
< AFEWIPDisplay >
    Field = KeyID,%-20.20s,KeyID;55
    Field = CreateTime,DX%m/%d/%Y,Create Time;35
    Field = StatusCode,%-1.2s;ST
    Field = EffDate,%-10.10s,Eff Date;50
```

Where *% -10.10s* tells the system to set aside 10 column spaces, providing the header description does not exceed 10 spaces. The *50* tells the system that you want to override the previous column setting, *% -10.10s*.

Also notice StatusCode does not have the optional setting. Here the system uses *%-1.25*. You can also specify the INI options as shown here:

```
< AFEWIPDisplay >
    Field = KeyID,,KeyID;55
    Field = CreateTime,DX%m/%d/%Y,Create Time;35
    Field = StatusCode,,ST;5
    Field = EffDate,,Eff Date;50
```

Using the WIP List and the Automatic Return Feature

If you use the WIP List option with the Automatic Return feature (see Returning to the Prior Window on page 212 for more information), the system returns you to the previous window after you view or edit a form set. The window appears again just as it was before the system displayed the WIP List window.

If you use the WIP List option without using the Automatic Return feature, you return to the main system window if you chose the Edit option.

If you chose any other option, such as Status, Send, and so on, you return to the old WIP window.

Using the WIP Menu Options

To turn off the WIP List and use the WIP menu options, you must change your MEN.RES file, stored in the PPSWIN\DLL directory. In the MEN.RES file, place a semicolon (;) in front of the following line:

```
MENUITEM “&WIP List…” 294 “AFEW32->AFECombineWIPDlgs””WIP Functions”
```

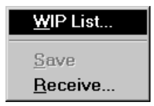
Then, remove the semicolons in front of the lines for the menu options. Your MEN.RES file should look as shown below when you finish:

```
POPUP     "W&ip" 257 "Wip menu"
BEGIN
; MENUITEM "&WIP List..." 294 "AFEW32->AFECombineWipDlgs""WIP
Functions"
MENUITEM "&Edit..." 270  "AFEW32->AFEUpdate""Edit existing document"
MENUITEM "&Batch Print..." 271  "AFEW32->AFEPrint""Batch Print
existing document"
MENUITEM "&View Batch Queue..." 272 "AFEW32->AFEViewBatch""View
existing document in Batch Queue"
MENUITEM "&Assign..." 273 "AFEW32->AFEAssign""Assign existing
document"
MENUITEM "&Manual Archive..." 274 "AFEW32->AFEArchive""Archive
existing document"
MENUITEM "&Delete..." 275 "AFEW32->AFEPurge""Delete existing
document"
MENUITEM "S&tatus..." 276 "AFEW32->AFEStatus""Change Status for
existing document"
SEPARATOR
MENUITEM "&Save" 103 "AFEW32->AFESaveClose"“Save document to wip and
close desktop"
SEPARATOR
MENUITEM "Se&nd..." 281 "AFEW32->AFEPackage"“Route documents via
mail"
MENUITEM "&Receive..." 282 "AFEW32->AFEReceive""Receive documents
via mail"
SEPARATOR
```

```
MENUITEM "Change In &Use..." 201 "AFEW32->AFEInUse""Edit In Use" 0
END
```

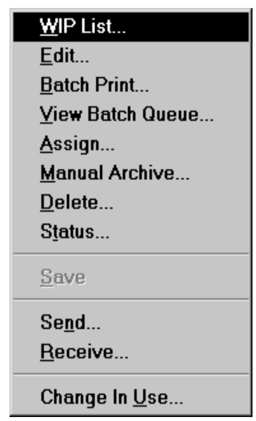
If you want to use both the older style WIP menu options and the WIP List option, include all of the lines shown above in the MEN.RES file, with no semicolons to comment out any of the lines.

Here are examples of how the WIP menu options can appear:



This is the default menu. You choose the WIP List option and then choose from the various tasks listed in the Task field.

This menu lets you choose the various WIP tasks directly from the menu.



This menu lets you use the WIP List or choose the various WIP tasks directly from the menu.

Using the Sort button

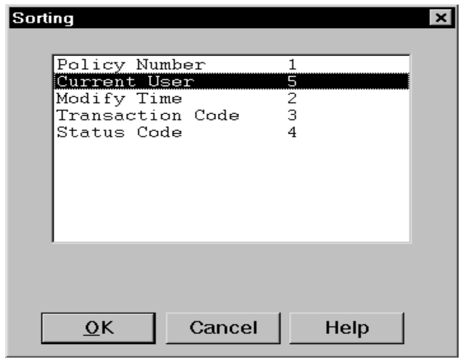
If you turn off the WIP List and instead use the WIP menu options, your system screens will include a Sort button which lets you sort the form sets in WIP. The following instructions explain how you can use the Sort button.

NOTE: If you have a large number of WIP records, you may want to use enhanced WIP selection to improve the speed with which the system displays WIP information. This is especially helpful if your users log in with the same ID. See Using WIP menu options and enhanced WIP selection on page 187 for more information.

Sorting the WIP list lets you display the stored files in any order, according to selected criteria. Common sort criteria include Current User, Modify Time, Transaction Code, and Status Code. The sort criteria is defined in the FSIUSER.INI file, which you can customize to meet your needs.

Follow these steps to sort the form sets:

1. Choose WIP, Edit or File, Edit. An edit window appears.
2. Click Sort to display the Sorting window.



3 Select a sort option and click Ok. The system displays the form sets in the new order.

Depending on your INI configuration, you may have a variety of WIP sorting options. The table below outlines available sort criteria.

Sort Criteria	Description	Examples
Company	Name of the company that uses the form set.	Insurance Underwriters, Mortgage Companies
LOB	Line of business associated with the form set.	Automobile Insurance, Homeowner's Insurance
Policy Number	Alphanumeric characters identifying the form set.	ABC12345
Record Type	Any type of company-specific code or number which further defines the form set.	N = New Policy R = Renewal 01 = Partial Policy
Create Time	The time of day the form set was created.	The create time is the time when the user saves entered form set data.
Original User	The user who originally processed the form set, if the set was reassigned to another user. Form set assigning and routing options are configurable in the INI file.	The original Insurance Agent who administers the form set. Also can identify a supervisor responsible for managing the form set.
Current User	The user who currently administers the form set.	Insurance Agents, Loan Officers
Modify Time	The time of day the form set data was modified.	Displays 24-hour time: 14:09 = 2:09 PM
Form Set ID	The system's name for the form set, based on the date and time.	Any combination of 8 alphabetic characters.

Sort Criteria	Description	Examples
Transaction Code	A one or two character code associated with the form set type. Codes are configurable in the INI file.	NB = New Business RN = Renewal Policy EN = Endorsement
Status Code	A one or two character code associated with the current status of the form set. Codes are configurable in the INI file.	W = Work in Process (WIP) A = Assign (to another user) Q = Quote U = Underwritten T = Transmit (via network) B = Batch print (group of form sets waiting for print processing) AR = Archived P = Printed CO = Combine (split or routed and assigned forms) DU = Duplicate * = User Defined
From User	The last user assigned to the form set.	Insurance Agent's supervisor
From Time/ To Time	The time of day the form set was reassigned from/to a user (when the new user received it)	Displays 24-hour time.
To User	The user to whom the form set was assigned.	Insurance Agent
Description	The description of the form set.	Insured - John E. Doe
In Use	The form sets that are currently in use.	Sorts in alphabetical order.
Record Number	Unsorted archive list.	Sorts in order of policy creation.

Using WIP menu options and enhanced WIP selection

If you turn off the WIP List and instead use the WIP menu options, your system screens will include a Sort button which lets you sort the form sets in WIP. If you have a large number of WIP records, you may want to make the following changes to improve the speed with which the system displays WIP information. This is especially helpful if your users log in with the same ID.

To make these changes, you must alter the MEN.RES file. These changes tell the system to display a full screen of WIP records as soon as possible, without first reading all of the records. Once the system fills the window with a list of WIP records, it stops searching until the user pages forward.

To apply these changes, you must change the following MEN.RES file values:

Change...	To...
AFEUpdate	AFEUpdateList
AFEPrint	AFEPrintList
AFEViewBatch	AFEViewBatchList
AFEAssign	AFEAssignList
AFEArchive	AFEArchiveList
AFEPurge	AFEPurgeList
AFEStatus	AFEStatusList
AFEPackage	AFEPackageList

Also, large WIP files are time-consuming to search if you do not give the system specific search criteria. After you make these changes, users can search for text in the columns displayed for the list.

Just double click the heading of the column you want to search. A window appears which lets you enter the text you want to search for. You can choose to do a partial match rather than matching the entire field. Use the buttons to scroll through the list.

NOTE: If you make these changes to the MEN.RES file, the system will omit the Sort button on each window you change.

Bypassing the Form Selection Window

To bypass the Form Selection window when you choose Edit from the WIP List, add the following control group and option to your FSISYS.INI or FSIUSER.INI file.

```
< FormSelection >
    SuppressDialog = Yes
```

The default is No, which tells the system to display the Form Selection window.

Filtering the Forms Viewed

You can filter, based on recipients, the forms you want to view when editing a policy. Here is an example of the INI options you would use:

```
< WIPFilter >
    DefaultRecipView = AGENT,client;
    DisplayRecipDialog = Yes
```

The DefaultRecipView option lets you specify the recipient forms you want to see.

The DisplayRecipDialog option tells the system whether or not it should display a window which shows all recipient names. The default is No.

If you include the DefaultRecipView option, the system ignores the DisplayRecipDialog option.

Automatically saving WIP

Each time you close a form set in WIP, the system asks if you want to save the data you entered. If you want the system to automatically save the information without asking, add the following control group and option to your FSISYS.INI or FSIUSER.INI file.

```
< EntryOptions >
    SaveWIPResponse = Yes
```

For example, if you set this option to Yes, as shown above, the system automatically saves the form set before it closes the form set. If you set this option to No, the system does not save the form set before it closes it.

There is no default for this option. If you omit the option or do not specify Yes or No, the system displays the confirmation prompt.

Limiting WIP and Batch Print to Specific Recipients

You can limit WIP and batch print to include specific recipients. If, for instance, you need to print specific recipients and those recipients are present, you can print that recipient's copy of the document. Regardless of whether anything actually prints, the document is then treated as though it did print and you continue to the next step of the completion process. The next step is usually automatic archiving or staged for manual archival, depending on your system's configuration.

To do this, add this INI option to the BatchPrint control group:

```
< BatchPrint >
    Select_Recipient = value1,value2,value3, ...
```

You can specify any number of recipients. Just separate the recipients using commas or semicolons. Space characters are not a valid separators because recipient names can contain spaces. The system removes leading and trailing spaces from recipient names.

During the batch print process, the system loads each selected transaction to see if it includes any of the specified recipients. If so, those recipients will print. If the document does not include any of the requested recipients, the system skips the print process for that transaction and continues through the workflow as though it had printed.

The Print Selection window shows the list of recipients you specified in the Select_Recipient option. Check the All Recipients option to indicate you want to have the recipients in the INI file printed. Keep in mind the All Recipients option *does not* mean the system will print all of the legitimate recipients in the documents.

You can turn off the All Recipients option on the Print Selection window and further limit the recipients.

AutoPrint

In case you are using the AutoPrint feature, the system includes a way to override the Select_Recipient option. This lets you set up the automatic print process to select one set of recipients, while having the normal batch print process select a different set of recipients.

To override the Select_Recipient option in the BatchPrint control group, you add another Select_Recipient option in the AutoPrint control group, as shown here:

```
< TimerFunc >
    0;0;10:10;AFEW32->AFEAutoPrint;
< BatchPrint >
    Select_Recipient = INSURED, AGENT
< AutoPrint >
    Select_Recipient = HOME OFFICE
```

Based on this example, BatchPrint will print one set of recipients while AutoPrint only prints a single recipient.

To have AutoPrint print all recipients, include the Select_Recipient option in the AutoPrint control group, but leave it blank. You would do this, for instance, if you want AutoPrint to print all recipients, while BatchPrint is limited to certain recipients. Here is an example:

```
< AutoPrint >
    Select_Recipient =
```

If you omit the Select_Recipient option from the AutoPrint control group, AutoPrint uses the setting provided by the BatchPrint control group — so you do not have to define the same option values in both control groups.

If you want BatchPrint to include all recipients and AutoPrint to limit recipients, only include the Select_Recipient option in the AutoPrint control group.

If, however, you want AutoPrint to omit all recipients, include the Select_Recipient option but define a value which is not a valid recipient name. Here is an example:

```
< AutoPrint >
    Select_Recipient = (NONE)
```

(NONE) will never match a recipient name because recipient names cannot include parentheses. If no recipients named in the form sets match those defined in the INI file, nothing prints. However, the WIP transaction still moves to the next completion stage.

Customizing WIP Transaction Codes

Individual units of *work* are usually called transactions. In the system, a user specifies a unit of work by selecting the key components which the system uses to distinguish this unit of work from all others.

In addition to the Key1, Key2, and KeyID (Company, Line of Business, and Policy Number) fields, all WIP transactions have a transaction code associated with them. This transaction code is a value you can define. You can also customize its affect on the transaction.

The WIP transaction code specifies certain criteria about the form set selection process. Several different transactions are defined in the default INI files included in the base system including...

- New Business
- Renewal
- Quote
- Endorsement

Defining Transaction Codes

You can define any number of transaction codes in the INI file. The following example shows the base system transaction codes recorded in the FSISYS.INI file used by the system.

```
< Transactions >
    01  = ;NB;New Business;TRNW32->TRNNew;
    02  = ;EN;Endorsement;TRNW32->TRNEndorse;
    03  = ;RN;Renewal;TRNW32->TRNRenew;
    04  = ;QU;Quote;TRNW32->TRNNew;
```

The numbers to the left distinguish each item and also specify the order the codes appear on the list the system presents to the data entry user. When the system loads the INI options, it sorts the list alphanumerically. In this example, New Business (NB) will be the default transaction presented to the user since it will sort as the first item in the list.

To the right of the equals sign (=) is the transaction code, its description, the DLL file name, and the function associated with the code. Notice that semicolons separate each of the components.

The first value is a two-character transaction code abbreviation. This is the value the system assigns to the WIP transaction record. The next value is the transaction description. The system displays this value to the user in the selection list for transaction type. The description should not exceed 19 characters.

The last value is the standard definition for DLL to load and an exported function name to call. Functions are associated with transactions to help data entry users select form sets or to affect how the system handles the transaction.

Each two-character transaction code should be unique. If it is not unique, it may not be possible for the system to match the transaction code to the description the data entry user selects when he or she creates the form set.

On the other hand, there may be times when you want to define a code more than once. For instance, suppose you want an endorsement based upon an archived form set to behave slightly different than one created by imported data. In each of these cases, you may want to assign the EN transaction code. To accomplish this during form selection, you would need different custom functions associated with separate transaction lines, as shown here:

```
02  = ;EN;Endorsement;TRNW32->TRNEndorse;
05  = ;EN;Arc Endorsement;TRNW32->TRNNew;
```

This example shows how one transaction code can be defined with two behaviors. If the user selects either of these transactions, the WIP record is still assigned EN as the transaction code. However, notice that each transaction invokes a different function and this affects how the user completes the form selection process.

This example works because the creation of the form set is usually the only time that a data entry user can specify the transaction code to use. Once WIP is created, the transaction code usually does not change.

New business = NB

Associating NB with a WIP transaction tells the system this is a first time entry or that the transaction stands on its own. The function invoked when selecting this transaction is usually TRNNew. The Form Selection window displays the available forms with the required forms already selected (checked). Also, if the INI settings require unique WIP keys, the default functionality enables this check on this transaction type.

Endorsement = EN

Endorsement is a term taken from the insurance industry. It refers to an amended transaction which is required as part of an existing business transaction. The function invoked by selecting Endorsement, normally TRNEndorse, will not preselect any of the forms for the user. The system assumes the user will know which forms to select in this situation. Since an endorsement usually implies that the transaction is associated with another, the check for unique WIP keys is relaxed.

Renewal = RN

Renewal is also a term taken from the insurance industry, but it applies to many others. There are many industries where a customer must be *renewed* after some period of time has elapsed. In addition to insurance, such a concept might apply for certain types of contracts, lines of credit, and licenses. The default functionality of TRNRenew is to preselect the required forms but to not require unique WIP keys.

Quotes = QU

A quote is usually considered a temporary transaction. The default behavior is assumed to be similar to New Business and uses the function, TRNNew. In fact, many times a quote transaction is later converted into New Business. This is the only transaction type users can change after the WIP is created.

NOTE: In addition to simply associating WIP transactions with the functions provided in a base system, you can also customize the base system. If you have a special requirement, contact Skywire Software's Sales department. Also, the technical document, AFE External Procedure Support, covers some of the details required to create a custom transaction function.

Modifying Transaction Codes

Normally, the system protects the WIP transaction code values for the Key1, Key2, and KeyID fields once the transaction is created. You can, however, modify the Key2 field, which is typically called the Line of Business (LOB) field. This can be important since the system lets you have multiple lines of business for a form set.

If, however, you do not want to let users modify Key2 fields on existing WIP, you can enter the following INI value:

```
< Control >
    NoKey2Changes = TRUE ; Default is FALSE to allow Key2 changes
```

You can also let data entry users edit other WIP fields on existing records. In addition, you can set up the system so the ability to edit transaction codes varies among the transaction codes. To do this, you must set up an INI control group to identify what transactions and what fields can be edited, as shown below:

```
< AllowTransEdits >
    KeyID       = QU
    Transaction = QU
    Key2        = NB;EN;RN;QU
```

As shown in this example, there are three fields you can edit as you work with WIP records, as long as those record have the specified transaction codes.

To the left of the equals sign (=) you see the name of these fields: KeyID (Policy number), Transaction (TransCode), and Key2 (LOB). To the right of the equals sign you see the transaction codes data entry users can edit.

This means, for instance, that while working with WIP transactions, you can change the transaction code if the code is QU (quote). A typical use of this feature is to let data entry users change a quote (QU) to new business (NB) and then assign a new KeyID. Notice too, that the Key2 example shows that multiple transaction codes are separated by semicolons.

NOTE: In the base system, there is no need to include the example Key2 line in any INI file since this definition represents the default behavior of the Key2 field. The behavior of the Key2 field is only changed if the NoKey2Changes option, mentioned earlier, is set to True.

Using ODBC WIP Indexes

WIP search windows can support ODBC WIP indexes. To use ODBC indexes, you must set up your INI files to set the database handler as shown in Chapter 8, Archiving and Retrieving Information in the Rules Processor System Guide.

Here are your options:

```
< DBHandler:ODBC >
   CreateIndex     = No
   CreateTable     = Yes
   Debug           = No
   Install         = SQW32->SQInstallHandler
   Qualifier       = e:\doculab\doculabs.mdb
   Server          = MS Access Database
```

Where *Server* is the name of the data source from your ODBC control panel setup and *Qualifier* is the name of the database (for MS Access).

Each database table you want to be an ODBC table will need entries in the INI file defining it as ODBC. For example, to define the WIP table, add these INI options:

```
< DBTable:WIP >
    DBHandler = ODBC
```

Using Compressed WIP

The system provides two ways to store WIP: *standard mode* and *compressed mode*. Storing WIP in compressed mode saves disk space.

Using compressed mode, you can work with compressed and non-compressed form sets. In standard mode, you can only work with non-compressed form sets.

Compressed mode uses the same storage format as the email attachment. It saves each WIP form set as a single compressed file rather than the two or more current WIP files, such as *.dat; *.pol, *.pkg.

To use WIP compression, you must add these INI options:

```
< WIPData >
    CompressWIP = Yes
< CompressWIPFiles >
    Extension = CWP
```

Since the default extension is *CWP*, the extension is not required in the example above. You must add the extension if you want to use an extension other than CWP.

Note: The CompressWIP option was added in version 11.1, patch 25. For earlier versions, you must specify the function as shown here:

```
< AFEProcedures >
    AFECompressWipFunc    = AFEW32->AFEPackWipFilesFunc
```

You should use the latest version, including all applicable patches.

The benefit is that only one smaller file per form set will be kept in the network WIP directory, which conserves disk space.

NOTE: Changing to the WIP compressed mode does not delete existing WIP files from the WIP directory.

STORING WIP ON AN SQL SERVER

Follow these steps to set up your INI files so you can store WIP information on an SQL server.

1 Add these INI options:

```
< DBTable:WIP >
    DBHandler   = ODBC
    UniqueTag   = DOCTAG
```

Option	Description
DBHandler	Enter ODBC to tell the system that WIP is a table in a ODBC-compliant database.
UniqueTag	Specify a unique key defined in the WIP.DFD file. For instance, if you are using the default WIP.DFD file, DOCTAG is a good choice since it is a unique key.

Here is a WIP.DFD example which uses DOCTAG:

```
< Key:DOCTAG >
    Expression  = KEY1+KEY2+KEYID+RECTYPE
    FieldList   = KEY1,KEY2,KEYID,RECTYPE
```

2 Add the DBHandler:ODBC control group and these options:

```
< DBHandler:ODBC >
    CreateTable= Yes
    CreateIndex= No
    Server      = MS SQL Server
    Qualifier   = master
    UserID      = sa
    Passwd      =
```

Use the Server option to define a data source name. Use the Qualifier option to define the database name if it differs from the default database name in the ODBC configuration.

3 Since Desc is a reserved word for SQL Server, convert it to a different name. Here is an example:

```
< ODBC_FieldConvert >
    Desc = DESC2
```

4 Use the ODBC_FileConvert control group to convert a short WIP file name into a long table name. Here is an example:

```
< ODBC_FileConvert >
    WIP = rpex1sqlwip
```

Deleting WIP

The system lets you delete WIP records by date, user ID, status code, or key ID. A counter indicates progress as the system deletes the records.

To do this, you must add the Delete WIP Records option to your Tools menu. Make the following changes to the MEN.RES file:

1. Search for the line that reads *Popup “&Tools” 255 “Utility Programs”*
2. On the line below the word *Begin*, remove the semicolon from this line:

```
; MENUITEM "&Delete WIP Records…" 2831 "AFEW32->AFEDeleteWIPRec"
"Delete WIP DBF records by …" 0
```

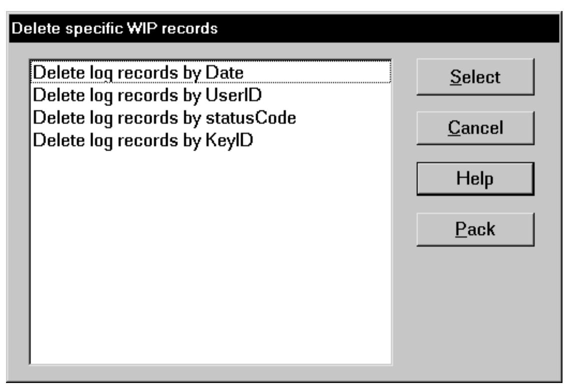
3. Add these options to your INI file:

```
< AFEWIPDBFFunc >
    01=;HT;Delete WIP records by Date;AFEW32->AFEDelWipDBFByDate"
    02=;UI;Delete WIP records by UserID;AFEW32-
>AFEDelWipDBFByUserID"
    03=;SC;Delete WIP records by StatusCode;AFEW32>AFEDelWipDBFBySt"
    04=;KI;Delete WIP records by KeyID;AFEW32->AFEDelWipDBFByKeyID"
```

Once the option is added, you can use it to delete WIP records by:

- Date
- User ID
- Status code
- Key ID

When you choose the Tools, Database Maintenance, Delete WIP Records option, the Delete Specific WIP Records window appears.



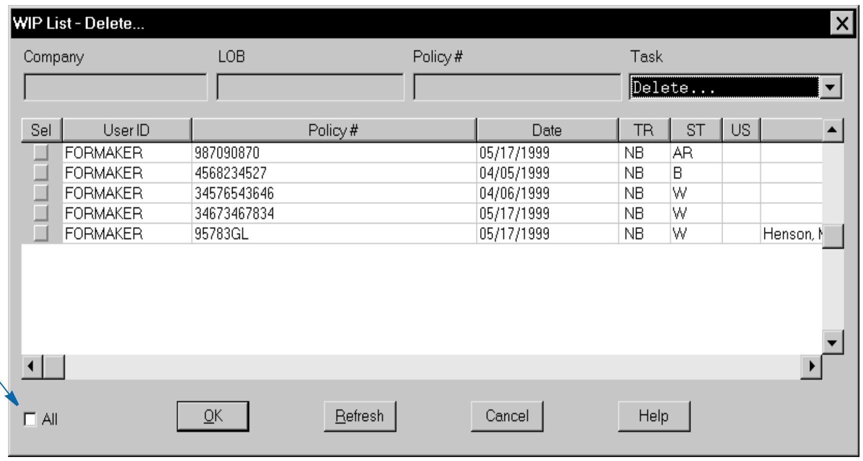
Note: You should *always* make a complete backup before you use this feature.

Choose the criteria you want to use to delete WIP records and click Select. Depending on the criteria you choose, the system displays a window to let you select a range of dates, a specific user ID, choose a status code, or enter a Key ID.

Removing the All field from the WIP List

When you select the WIP List option from the WIP menu and then select Delete in the Task field, the following window appears:

You can remove this field to prevent users from inadvertently deleting all work-in-process.



To remove the All field, change the AllSelect option from *Yes* to *No*, as shown here:

```
< AFEWIPDisplay >
   AllSelect = No
```

Keep in mind that if you change the AllSelect option to No, the All field will not appear on other WIP List windows, including the following:

- WIP List - Batch Print
- WIP List - Assign
- WIP List - View Batch Queue
- WIP List - Manual Archive
- WIP List - Status
- WIP List - Change in Use
- WIP List - Send

Using GenData WIP Transaction Processing

GenData WIP Transaction Processing lets you process WIP transactions based on their status code. The transactions are created by one of these processes:

Note: You must have a license to both Documaker and Documaker Workstation to use GenData WIP Transaction Processing.

- Executing the GenWIP program after the GenData program to process the transactions in the manual batch. Then using Documaker Workstation to:
  - Manually view a transaction and update any required data. Then use the WIP, Save option to save the transaction with a status code such as: Approved or Accepted.
  - Manually view a transaction and then use the WIP, Save option to save the transaction with a status code of Rejected.
  - Manually view a transaction, update any required data, and save the transaction using the File, Complete, Batch Print option. This assigns a Batch Print status code to the transaction.
- Creating a new transaction using Documaker Workstation and then using the WIP, Save or File, Complete, Batch Print option to save it with a status code such as Approved, Accepted, or Rejected.

You can then process these transactions as:

- New transactions
- Transactions appended to an existing MRL recipient batch, NewTrn, NA, and POL files created by a prior run of the GenData program.

GenData WIP Transaction Processing creates new recipient batch, NewTrn, NA, and POL files which you can print, archive, or both using the GenPrint and/or GenArc programs.

To process the transactions in Documaker Workstation, you need to use two MEN.RES functions. One lets you change the status of the transaction being edited and close the form set. The other deletes the transaction being edited.

You can define the change status functionality multiple times in the MEN.RES file, once for each status code you have. Typically, users only set up two status codes—*Approved* and *Rejected*—but you can define as many as you want.

Use the delete function when you do not want to batch process transactions that have been rejected and instead want to delete those transactions immediately.

The MEN.RES functions are:

- AFESetStatus. This function sets the status of a WIP transaction and closes it. You can use this function if you have to set the status of WIP transactions over and over again. Set up the codes you want to use in the Status_CD control group and then include those codes in the MEN.RES file.

This example sets up two status codes, *Approved* and *Rejected,* in the INI file, two Documaker Workstation menu functions to set the status of the WIP transaction to either *Approved* or *Rejected*, and two buttons on the toolbar to make it easier to approve or reject transactions.

Here is a example of the Status_CD control group:

```
< Status_CD >
    APPROVED = AP
    REJECTED = RJ
```

Here is an excerpt from a MEN.RES file:

```
MENUITEM    "&Approve Form" 150 "AFEW32->AFESetStatus" "Approved"
MENUITEM    "&Reject Form" 151 "AFEW32->AFESetStatus" "Rejected"
:
:
TOOL 71 150 DISABLED BUTTON
TOOL 70 151 DISABLED BUTTON
```

- AFECloseAndDelete. This function deletes a transaction from the WIP list and then closes the form set. Here is an excerpt from the MEN.RES file:

```
MENUITEM    "&Close and Delete" 152 "AFEW32->AFECloseAndDelete"
"Close and delete the current form set"
```

USING THE WIPFIELD BUILT-IN FUNCTION

You can use the WIPField built-in INI function to tell the system to substitute a value in the INI file with a value from the WIP record. This works with either Documaker Workstation (AFEMAIN) or the WIP Edit plug-in.

For example, if you want the UserDict value to equal the value for ORIGUSER in the current WIP record, you would set up the following option:

```
< Spell >
    UserDict = ~WIPFIELD ORIGUSER
```

Customizing Archive

There are several options you can use to customize how information is transferred to and from archive. These options are explained below.

Transferring Key Information to Archive

The File, New and File, Retrieve options transfer key field (Key1, Key2, and KeyID) information from the Form Selection window to preselect records on the Retrieve window.

If you have customized your implementation and your settings differ from the base system's default settings, this transfer of information may result in a message stating that no records were found. Users must respond to this message before they can continue the retrieval process.

You can use the following INI options to specify whether the system should continue to transfer this key field information:

```
< ArcRet >
    TransferKey1 = Yes/No
    TransferKey2 = Yes/No
    TransferKeyID = Yes/No
```

Set the options to *Yes* to transfer the information. Set the options to *No* if your system has been customized and you are receiving messages indicating that no records were found.

Mapping Form Set Field Values

You can assign form set field values to the archive index record. This is controlled by INI options in the AFEFormset2ArchiveRecord control group.

In the base system, the standard archive process uses the AFEWIP2ArchiveRecord control group to map fields from the WIP record to the archive index record. Once this takes place, you can use the AFEFormset2ArchiveRecord control group to match certain form set fields with archive index field destinations, as shown here:

```
< AFEFormset2ArchiveRecord >
    NAME = INSURED NAME
    CITY = CITY
```

The name on the left is a DFD field name defined in the archive index record. The name on the right is the name of the field located in the form set.

In this example, two fields from the form set are being mapped into the index record. The form set field, *INSURED NAME*, is mapped to the DFD field, *NAME*. In the second line, the DFD field name and form set field names are both named *CITY*.

When you use this feature, keep these items in mind:

- During archival, if the form set is not currently loaded (as is often the case when you are doing a manual archive), the form set is temporarily loaded while the mapping occurs.
- If a named form set field occurs multiple times within the form set, the value transferred will be taken from first occurrence the system locates. The scope of the field defined is not considered.
- If a form set field is named but cannot be found within the form set the archive process continues and no error message appears.

- If a DFD field is named but does not exist, an error appears and the process continues—which is consistent with the standard WIP to Archive transfer.
- If the DFD field length is defined shorter than the field data from the form set, it will be truncated to the DFD field length. No additional formatting occurs. No error message appears.

Preserving the CreateTime Value

Use the KeepCreateTime option to tell the system you want to preserve the CreateTime field value when it creates a new form set based on a document retrieved from archive.

Typically, when you create a new WIP Entry it has a create time that reflects when the form set was actually created. If, however, the form set was created from an archived document and you want to preserve the CreateTime from the archived document, add this option:

```
< ArcRet >
    KeepCreateTime = Yes
```

The default is No which tells the system to give the new form set a new create time. Enter Yes to tell the system to instead use the entry in the CreateTime field from the archive record.

Converting Archive Index Fields into WIP

You can convert all archive index fields into WIP index fields, not just the key fields.

You use the AFEWIP2ArchiveRecord control group to map back and forth between these two windows, instead of forcing the Key1 (Company), Key2 (LOB), and KeyID (Policy Number) fields to match on both windows.

```
< AFEWIP2ArchiveRecord >
    NAME = INSURED NAME
    CITY = CITY
```

The name identified on the left is a DFD field name defined in the archive index record. The name on the right is the name of the field in the form set in WIP.

If you do a retrieval from the Form Selection window the Retrieval window is populated with the fields from the Form Selection window, and when you pick a form set to retrieve that information is populated back onto the Form Selection window.

In other words, the default behavior should appear the same even though the underlying code has changed. However, this makes it possible for the two windows to have different fields and still transfer data from archive back to the Form Selection window.

Clearing Version Information

You can use the RetrieveVersionInfo option to clear the library version and revision information stored in an archived form set:

```
< FormSelection >
    RetrieveVersionInfo = No
```

For instance, you would use this option if you want to use newer versions of the forms than those in the retrieved form set. Set this option to No to clear that information when you are using an archived form set to create a new form set.

Specifying the Sort Order

You can specify the order, ascending or descending, during an archive search. To do this, you must make the following changes in your APPIDX.DFD file and FSIUSER.INI files.

NOTE: You can only use one column to change the order. The ODBC, Oracle, and DB2 drivers are supported.

Assuming this is in your APPIDX.DFD and you want to sort on Company in descending order, here is an example of the changes you must make (some fields were omitted for clarity):

```
< Fields >
    FieldName   = Company
< Field:Company >
    INT_Type    = CHAR_ARRAY
    INT_Length  = 6
    EXT_Type    = CHAR_ARRAY
    EXT_Length  = 6
```

You will also need to add these options:

```
< Keys >
    KeyName     = Company
< Key:YourTagName >
    Expression  = Company
    FieldList   = Company
    Descending  = 1
```

Your FSIUSER.INI file should include information similar to this:

```
< DBTable:APPIDX >
    DBHandler   = ODBC
    DefaultTag  = YourTagName
```

The string *YourTagName* can be anything, including the word Company. The SQL statement logged in the trace file should include *order by Company desc* at the end.

Specifying the File Size when Splitting Archive Files

You can limit the size of the CAR files created when you split an archive file to size no less than 100KB and no more than 1.4GB. The system does not accept values outside that range.

NOTE: You can also limit the size of CAR files created when you split an archive using the ARCSPLIT utility. See the Docutoolbox Reference for more information on the ARCSPLIT utility.

You enter the file size limit on the ArcSplit window. Use these options to enable this feature for Documaker Workstation:

```
< ArcSplitConfig:Example >
    CARFileSize        = 1
    EnableCARFileSize  = No
```

Option	Description
CARFileSize	Enter a number between one (1) to 14,000 to define the size for the CARFile. If you enter one (1), the system interprets that as 100,000 bytes (100 KB). If you enter 14,000, the system interprets that as 1,400,000,000 bytes (1,4000,000 KB).
EnableCARFileSize	This option turns on and off the related radio button field on the ArcSplit window in Documaker Workstation and also tells both Documaker Workstation and the ARCSPLIT utility whether to use the CARFileSize option. The default is No.

Customizing the Interface

The system provides numerous options you can use to customize the interface the end user sees. These options are discussed below.

Highlighting the Active Field

Use this INI option to control whether or not fields are highlighted on forms when using the View, Fixed Edits option.

```
< Control >
   ActiveFieldHighlight = Yes/No
```

The default is *Yes*.

Making Sure Required Fields have Data

You can now include the RACCheckRequiredFields function in your MEN.RES or WIPEDIT.RES file to have the system make sure required fields have data.

This table describes what the RACCheckRequiredFields function does:

If...	The function displays
The required fields have data	*All required fields have data.*
One or more of the required fields is missing data	*Data required for [name of field].* The system makes the first required field with missing data the current field.

Setting up a required fields check with WIPEdit

There is a RACLIB function that should be placed in the WIPEDIT.RES file. This function may send a response back to WIPCTL or it may display a message, depending on whether it is executed from WIPEdit or AFEMAIN (Documaker Workstation). It is set up like any MEN.RES function and must be identified by a command number not used by any other function in the WIPEDIT.RES file. The setup in the WIPEDIT.RES file must occur regardless of the way you start the required field check.

Here is an example line from the WIPEDIT.RES file:

```
MENUITEM "&CheckRequiredFields" 263 "RACOS2->RACCheckRequiredFields"
"Check Required"
```

You do not need to make changes to IDS.

There are two ways to start the required field check:

- From a menu
- From an ASP or JSP page

Starting from a menu

To let users start a required field check from the menu add the following line to the WIPEDIT.RES file. The number 263 may vary but it should not be used for another function in the WIPEDIT.RES file. This line tells the system to include a Check Required Fields option on the menu that appears when you right click the mouse.

```
MENUITEM "&Check Required Fields" 263 "RACOS2-
>RACCheckRequiredFields" "Check Required"
```

Starting from an ASP page

Follow these steps to start a required field check from an ASP page.

1 Register the WIPCTL.DLL file.

2 Set up the WIPEDIT.RES file to execute the required field check from the menu.

3 The web page must contain a script like the following. When this script executes, the required field check will occur. The variable rspmsg will contain either "All required fields have data" or an error indicating the field that needs data. The RACCheckRequiredFields function will be executed in this case as well but its response is returned to the script, not displayed in a message box.

```
< Script Language="VBScript" >
    Function SendToWipedit
        Dim rspmsg
        set aspobj = CreateObject("Wipctl.WipEd.1")
        aspobj.cmdWithMessage 263,rspmsg
        set aspobj = Nothing
        MsgBox(rspmsg)
    End Function
</Script>
```

Showing and Hiding Field Placeholders

You can control the display of field placeholders using this INI option:

```
< Control >
    ShowPlaceHolders=Yes/No
```

The default is *Yes*.

Display Notes as Form Help

Sticky notes can provide information about a field on a form if added when the form is created. In Entry, when you position your cursor over the sticky note icon, the system then displays the contents of the note in a tooltip-style window. When you move the cursor again, the note disappears.

To enable this option in the Entry module, add the ShowNotes INI option to your FSISYS.INI file:

```
< Control >
    ShowNotes = Yes
```

The default is No.

You can also add a menu option to turn on or off the display of notes. To add this option to your menu, add the Show Notes line to your MEN.RES file:

```
...
MENUITEM "Show &Notes" 1086 "NULL" "NULL"
...
```

1086 is the reserved menu ID for the Show Notes option.

NOTE: You cannot embed note objects within text area objects for use with paragraph assembly. TERSUB only imports information stored in the first text area in the image. Objects defined outside the text area are ignored.

Enabling Indexes and Tables of Contents

An image’s creator can add tables of contents, lists of figures, or indexes to form sets to make it easier for users to navigate through the various forms. To use this feature, all images must be loaded *before* the print operation executes. Otherwise, the system will not have all the content available and will not be able to create a complete table of contents, list of figures, or index.

Since some print drivers do not force the loading of all images until necessary, you need to include this INI option:

```
< Control >
   LoadPrintOnly = Yes
```

Fixed Prompt Placement

Use the FixedPrompt option to control the placement of the fixed prompt line used to enter data onto forms.

```
FixedPrompt=Yes/No
```

The default is No.

This option works with the Fixed Edits option. Once you select Fixed Edits, the system activates the Fixed Prompt option and puts a check mark beside both on the menu.

The following line in the MEN.RES file lets users toggle Fixed Prompt mode on and off:

```
MENUITEM “Fixed &Promp ”1084 “NULL” “NULL”
```

The code number 1084 is automatically keyed to fixed prompts so *“NULL”, “NULL”* is all that is required to activate this option.

Customizing the Titles of Windows

To change the title of the Complete window, use the following INI option. If you do not want the window to have a title, leave the right side (*TitleToDisplay*) blank.

```
< DLGTitles >
   CompleteDlgTitle = TitleToDisplay
```

The system defaults to the default title included in the MEN.RES file.

To change the title of the Retrieve window, use the following INI option. If you do not want the window to have a title, leave the right side (*TitleToDisplay*) blank.

```
< DLGTitles >
   RetrieveDlgTitle = TitleToDisplay
```

The system defaults to the default title included in the MEN.RES file.

Customizing the Text Editor Menus and Toolbar

To customize the Text Editor's menus and toolbar, you must add this INI option in the Menu control group:

```
< Menu >
    TXMMENU=TXM.RES
```

This option tells the system to find the definition of the Text Editor's menus and toolbar in the file named *TXM.RES*.

NOTE: Documaker Workstation uses a resource file named MEN.RES to define its menus. The TXMMENU option lets you do a similar thing with the Text Editor. Keep in mind, however, that the MEN.RES file can call custom functions and DLLs while the file specified with the TXMMENU option cannot.

Here is a sample resource file for defining the Text Editor menus and toolbars. This file specifies the default Text Editor menus and toolbar.

Sample resource file for the Text Editor

```
MENU        "Edit"
BEGIN
    POPUP "&File" 5850 "System menu"
    BEGIN
       MENUITEM "&Save\tCtrl+S"             5851 "NULL" "Description"
       MENUITEM "&Properties..."            5852 "NULL" "Description"
       MENUITEM "E&xit\tF3"                 5853 "NULL" "Description"
    END
    POPUP "&Edit" 5854
    BEGIN
        MENUITEM "&Undo\tCtrl+Z" 5856 "NULL" "Description"
        SEPARATOR
        MENUITEM "&Copy\tCtrl+C" 5857 "NULL" "Description"
        MENUITEM "Cu&t\tCtrl+X" 5858 "NULL" "Description"
        POPUP "&Paste"
        BEGIN
            MENUITEM "Formatted text\tCtrl+V" 5859 "NULL"
"Description"
            MENUITEM "Unformatted text\tCtrl+Alt+V" 5838 "NULL"
"Description"
        END
        MENUITEM "&Delete\tDel" 5844 "NULL" "Description"
        SEPARATOR
        POPUP "Se&lect"
        BEGIN
            MENUITEM "&Character" 5874 "NULL" "Description"
            MENUITEM "&Word\tCtrl+Sp" 5875 "NULL" "Description"
            MENUITEM "&Line" 5876 "NULL" "Description"
            MENUITEM "&Paragraph" 5877 "NULL" "Description"
            MENUITEM "C&olumn" 5846 "NULL" "Description"
            MENUITEM "&All\tCtrl+/" 5879 "NULL" "Description"
            MENUITEM "&Deselect\tCtrl+\\" 5880 "NULL" "Description"
        END
        SEPARATOR
        MENUITEM "&Bold\tCtrl+B" 5860 "NULL" "Description"
        MENUITEM "&Italic\tCtrl+I" 5861 "NULL" "Description"
```

```
        MENUITEM "U&nderline\tCtrl+U" 5862 "NULL" "Description"
        MENUITEM "&Strike-out" 5863 "NULL" "Description"
        MENUITEM "Do not &hyphen" 5885 "NULL" "Description"
        MENUITEM "&Do not break" 5895 "NULL" "Description"
        MENUITEM "C&olor..." 5864 "NULL" "Description"
        POPUP "C&hange Case..."
        BEGIN
            MENUITEM "&Upper Case\tAlt+U"  5803 "NULL" "Description"
            MENUITEM "&Lower Case\tAlt+L"   5804 "NULL" "Description"
            MENUITEM "&Sentence Case\tAlt+S"  5805 "NULL"
"Description"
            MENUITEM "&Title Case\tAlt+T"   5806 "NULL" "Description"
        END
    END    POPUP "&View" 5865
    BEGIN
       MENUITEM "&Zoom..."                   5866 "NULL" "Description"
        SEPARATOR
       MENUITEM "&Status Line"               5869 "NULL" "Description"
      MENUITEM "&Nonprinting Tokens"         5870 "NULL" "Description"
       MENUITEM "R&ulers"                    5830 "NULL" "Description"
       MENUITEM "&Title bar"                 5831 "NULL" "Description"
       MENUITEM "Tool&bar"                   5832 "NULL" "Description"
        SEPARATOR
       MENUITEM "&Refresh\tF5"               1009 "NULL" "Description"
       MENUITEM "Re&format\tF6"              5872 "NULL" "Description"
       MENUITEM "&Level Columns"             5898 "NULL" "Description"
    END
    POPUP "&Insert" 5881
    BEGIN
       MENUITEM "&Break..."                  5882 "NULL" "Description"
       MENUITEM "&File..."                   5883 "NULL" "Description"
       MENUITEM "&Logo..."                   5884 "NULL" "Description"
       MENUITEM "Fiel&d..."                  5878 "NULL" "Description"
       MENUITEM "B&ox..."                    5847 "NULL" "Description"
    END
    POPUP "F&ormat" 5885
    BEGIN
       MENUITEM "&Font...\tCtrl+F"           5886 "NULL" "Description"
        MENUITEM "&Paragraph...\tCtrl+P"       5887 "NULL"
"Description"
      MENUITEM "&Tab Stops...\tCtrl+T"       5888 "NULL" "Description"
       MENUITEM "&Border..."                 5889 "NULL" "Description"
       MENUITEM "&Shade..."                  5890 "NULL" "Description"
       MENUITEM "&Columns..."                5891 "NULL" "Description"
    END
    POPUP "&Tools" 5892
    BEGIN
      MENUITEM "&Spell Check...\tF7"         5893 "NULL" "Description"
      MENUITEM "Spelling &Options..."        5816 "NULL" "Description"
      MENUITEM "&Find/Replace..."            5894 "NULL" "Description"
    END
    POPUP "&Help" 5895
    BEGIN
      MENUITEM "&Contents\tF11"              9904 "NULL" "Description"
      MENUITEM "&How to...\tCtrl+F2"         9902 "NULL" "Description"
      MENUITEM "&Shortcuts\tShift+F2"        9903 "NULL" "Description"
```

```
        MENUITEM "&Glossary...\tCtrl+F11"      9907 "NULL"
"Description"
        MENUITEM "&Using Help\tShift+F11"      9901 "NULL"
"Description"
        SEPARATOR
        MENUITEM "&Product Information..."     9906 "NULL"
"Description"
    END
END
TOOL 2    5851  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
TOOL 21   5858  ENABLED BUTTON
TOOL 22   5857  ENABLED BUTTON
TOOL 23   5859  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
TOOL 52   5860  ENABLED BUTTON
TOOL 53   5861  ENABLED BUTTON
TOOL 54   5862  ENABLED BUTTON
TOOL 68   5885  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
TOOL 60   5870  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
TOOL 29   5884  ENABLED BUTTON
TOOL 34   5878  ENABLED BUTTON
TOOL 30   5847  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
TOOL 63   5891  ENABLED BUTTON
TOOL 64   5889  ENABLED BUTTON
TOOL 31   5890  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
TOOL 41   5856  ENABLED BUTTON
TOOL 62   5893  ENABLED BUTTON
TOOL 55   5886  ENABLED BUTTON
TOOL 8    5866  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
TOOL 13   5899  ENABLED BUTTON
TOOL 0    0     NULL    SEPARATOR
```

Adding Information to the Assign Window

The Assign window shows a list of users after you select the WIP, Assign option. The system also includes columns for the user ID and name.

By changing INI options, you can display more user information and control the placement of the information. These options are stored in the AFEAssignDisplay control group. Here is an example of how you can customize this window:

```
< AFEAssignDisplay >
    ; These are the defaults if you have no options defined
    FIELD = ID,%-9.8s,User ID
    FIELD = NAME,%-26.25s,User Name
    ; These are additional fields you can display
    FIELD = RIGHTS,%-2.1s,RT
    FIELD = INUSE,%-2.1s,US
    FIELD = REPORTTO,%-9.8s,Report
    FIELD = SECURITY,%-65.64s,Security
```

```
FIELD = MESSAGE,%-129.128s,Message
```

NOTE: The system will not display the password even if you add it to the INI file.

Start each option with *FIELD=*. The order in which you define the list determines the order of the columns—the first column appears to the left of the window.

The right side defines the DFD field name, followed by the format and the column heading description. You can use these formats:

- Percent signs (%) indicate the opening of the format string.
- A dash (-) indicates you want the text left justified. Omit the dash for right justification.

The formats are followed by a *precision.length* indicator. This indicator makes sure all data is displayed at a consistent length within that column.

The *s* at the end of each format tells the system it is a text string. This is the only option you can use for these fields.

NOTE: The size of the window does not vary, if you display more information than will fit, use the scroll bar to see the additional columns.

Centering Form Titles

To center form titles set the following INI option.

```
< Control >
    CenterFormTitles = Yes/No
```

The default, *No*, left justifies form titles.

Customizing the Send and CC Confirmation Messages

Use the KeyIDTitles option to customize the text that appears on the Send and CC confirmation messages. For example, if you want the system to display a message similar to this one:

```
Package CC'd to Tyler, Susan.
Customer ID is 33333
```

You would set the option as shown here:

```
< DLGTitles >
    KeyIDTitle = Customer ID
```

The default is *KeyID*.

Turning Off the Page Break Option

Use the LimitPageBreak option to turn off the option to insert a page break (Insert, Break, Page Break) when a text area field does not have the proper options set:

```
< Control >
```

```
LimitPageBreak = Yes
```

This INI option also disables the CTRL+ENTER shortcut.

Returning to the Prior Window

Use the Automatic Return feature to have the system redisplay the window as it was before the user made his or her last selection. You turn this feature on or off using an INI option. By default this option is set to *No*. To set feature, use these INI options:

```
< AutoReturn >
    FormSelection  = Yes
    FormsetRetrieve= Yes
    WIPEdit        = Yes
```

When you use the Automatic Return feature with the WIP List window, (see Using the WIP List and the Automatic Return Feature on page 184 for more information) you return to the WIP window instead of the Form Selection window, Edit window, Retrieve window, or View Batch Queue window.

The FormSelection option returns you to the Form Selection window if the you selected the File, New option. The WIP List option returns you to the WIP List - Edit window if you selected the WIP, WIP List or File, Edit option to open a form set.

Displaying the Form Description on the Print Notification Window

Use this option to show the form description instead of the form name on the Print Notification window:

```
< Control >
    PrintNotifyText = Description/Name
```

The default is *Name*. The system looks in the FORM.DAT file for the form description data instead of the form name and displays the form description name when you are printing form sets, if you set this option to *Description*.

Preventing Users from Retrieving Forms During Form Selection

Use the following INI option to hide the Retrieve button on the Form Selection window.

```
< FormSelection >
    HideRetrieve = Yes
```

If you omit this option, the Retrieve button appears on the Form Selection window.

Displaying Grid Lines

If you want to add an option for displaying grid lines, add this line to your MEN.RES file:

Add this line

```
POPUP "&Options" 1056 "Options"
    BEGIN
        MENUITEM "&Field Template"  1057    "NULL" "NULL"
        MENUITEM "&Auto Focus"      1059    "NULL" "NULL"
        MENUITEM "&Information"     1062    "NULL" "NULL"
        MENUITEM "Fixed &Edits"     1077    "NULL" "NULL"
        MENUITEM "Fixed &Prompt"    1084    "NULL" "NULL"
        MENUITEM "&Grid"            1015    "NULL" "NULL"
```

This adds the Grid option to the Options menu.

Handling Banner Pages

You can use the following options to control how the system handles banner pages:

```
< Banner >
    Enabled        = No
    Banner         = filename.fap
    FilterRecips   = recip;recip;recip;
```

Option	Description
Enabled	Enter Yes so you do not have to define the Name option. This tells the system to instead use the Banner option. The default is No.
Banner	Setting this option to *banner* provides support for legacy systems. You can, however, enter the name of a FAP file.
FilterRecips	Use this option to specify recipients who should receive banner pages. If not defined, all recipients receive a banner page. If you enter more than one recipient, separate the recipient names with semicolons.

Showing the Prior Policy Number on Renewals

When users renew a policy they retrieve the original policy from archive and then assign it a new policy number on the Forms Selection window. To add a field to the banner page that automatically displays the previous policy number, create a field on the banner page called *PRVPOL*.

In your INI file, include the new field in the Banner control group, as shown here:

```
< Banner >
    Field      = PRVPOL
    Field      = POLICY NBR
    Field      = INSURED NAME
    Name       = PFORM
    Size       =
```

In the BannerProc field be sure to include the following:

```
< AFEProcedures >
    BannerProc = TRNw32->TRNSetBannerFormInfo
```

Also you must enter the following line in the FormSelection group of the INI file:

```
< FormSelection >
    OriginalPolicyNumber = PRVPOL
```

NOTE: The name *PRVPOL* is user defined. It does not have to be PRVPOL.

Customizing Recipient Selection on the Print Window

You can use the following INI options to customize how users select recipients:

```
< PrtType:XXX >
    SelectRecipients   = Yes
    AllRecipients      = No
```

Option	Description
SelectRecipients	Enter Yes to have the system pre-select this option when the Print window appears.
AllRecipients	Enter Yes if you want the All Recipients field pre-selected so the system will print copies for all recipients. Enter No if you want to make the user choose the individual recipients for which you want to print copies. If you enter Yes for the SelectRecipients option and omit this option, the system defaults to pre-selecting the All Recipients field.

Specifying Recipient Print Order

Use the RecipientPrintOrder option to choose the recipient print order when printing from archive:

```
<   Printer >
    RecipientPrintOrder = Company,INSURED,PRODUCER
```

This example specifies the order in which forms or the form set should print. If you append *Yes* to the entry, a window appears which lets users rearrange the order. Here is an example:

```
<   Printer >
    RecipientPrintOrder = Company,INSURED,PRODUCER;YES
```

Customizing the Select Recipients Option

The select recipients functions (AFEViewRecipients, AFEViewNextRecipient, and DefaultRecipView) let you choose which recipient forms view. These functions, located in the MEN.RES file, are called from the Formset, Select Recipients options. These functions determine the options which appear in the Select Recipients window. If you want to see a certain recipient repeatedly, such as the INSURED recipient, you can add a customized MEN.RES function to view that recipient's forms.

AFEViewRecipients

AFEViewRecipients is the MEN.RES function that specifies one or more recipients for display in the Select Recipients window. The following is an example of how to set this option in the MEN.RES file:

```
MENUITEM "Select Insured" 187 "AFEW32->AFEViewRecipients" "INSURED"
MENUITEM "Select Others" 188 "AFEW32->AFEViewRecipients" "AGENT,
HOME OFFICE"
```

In the second line, there are two recipients. If you need to see two or more recipients, enter those recipients on the same line and separate them with commas or semicolons. You can change the control ID, but it must remain in the 151-200 range because a form set has to be open to use these functions.

AFEViewNextRecipient

The AFEViewNextRecipient function displays the next recipient's form. Here is an example of how to set up this option in the MEN.RES file:

```
MENUITEM "Select Next" 190 "AFEW32->AFEViewNextRecipient" "View the
next recipient"
```

The control ID must be within the range of 151-200.

DefaultRecipView

The DefaultRecipView function works in Archive Retrieval. If you want to display a form set from Archive Retrieval in a particular recipient's view, enter the following INI option in the ArcRet group:

```
< ArcRet >
    DefaultRecipView =
```

Set the values for this option to the recipients you want to see. If there is more than one recipient, separate them by commas. For example, to display the retrieved forms for the Insured and the Agent, enter the following:

```
< ArcRet >
    DefaultRecipView = INSURED, AGENT
```

Forcing Users to Select from Tables

Some form fields let data entry users select entries from drop-down tables. By default, the system also lets users type in entries. You can add the following option to your FSISYS.INI or FSIUSER.INI file to force data entry users to select from the options listed in the table. When set to Yes, this option removes the ability to type in an entry.

```
< Control >
    ForceTableOnlySelection = Yes
```

When set to Yes, this option removes the ability to type in an entry. If you omit this option or set it to No, users can select an entry from the table or type in an entry.

Optimizing the Use of Tables

For performance purposes, you can include one of these INI options to specify that entry table files will use the old or new format, which was introduced in version 10.1. (*Do not* include both options.)

```
< Tables >
    OldFormatOnly = Yes
    NewFormatOnly = Yes
```

For instance, if you are doing a lot of entry table lookups, your tables are located on a network drive, and the tables are a mix of both old and new format tables, performance can be affected because the system has to check the format of each table.

If, however, you can use one of these new options to tell the system that all tables are in the same format, it can omit that query and performance improves. Specify only the option that applies. If you omit both options, the system first checks to see if the table is in the new format. If not, then it checks to see if the table is in the old format.

Keep in mind that if you include one of these options, all of your tables must be in that format. For instance if you set the OldFormatOnly option to Yes, all of your tables must be in the old format. If you later decide to convert your tables to the new format, you must remove this option and, to get the same performance gain, include the NewFormatOnly option.

Printing "Draft" or "Reprinted" on Form Sets

To have the system automatically print *Draft* on form sets which have not been completed or *Reprinted* on form sets printed from Archive, add the following options in your FSISYS.INI file:

```
< AfeProcedures >
    INIT=CSTW32->CSTInitImageHandler
< Control >
    PrintDraft=Yes
< ArcRet >
    PrintReprinted=Yes
```

The INIT option specifies the code module to use. The PrintDraft option tells the system to add *Draft* to the SendCopyTo field instead of the recipient name during normal WIP printing.

NOTE: If you print the form set from the Complete window, the system will name the recipients as usual.

The PrintReprinted option tells the system to include *Reprinted* in the SendCopyTo field when it prints an archived form set.

Printing Through Menu and Toolbar Options

Using the MEN.RES file, you can add a menu or toolbar option that lets you print immediately without having to choose the print destination and which lets you select printer definitions that are not available on the Print window.

For each menu option, you specify the printer type in the MEN.RES file that you want to use and, optionally, whether you want the Print window to appear.

Here are some example MEN.RES file changes:

```
MENUITEM "PCL" 191 "AFEW32->AFEDirectPrint" "PCL"
MENUITEM "GDI" 192 "AFEW32->AFEDirectPrint" "GDI,N"
MENUITEM "Other" 193 "AFEW32->AFEDirectPrint" "OTHER,T,F"
```

These three menu items all use the same AFE function. An associated toolbar icon would need to specify the appropriate value (191, 192, or 193) to invoke one of these items.

The syntax of the last part of the menu line is the key. The first section up to the comma (if included) specifies the PrtType control group to use from the INI file. This does not have to be one of the printer types you specify in your Printers control group. This feature lets you have print groups that do not appear on the printer selection list of the Print window.

So, using the example above, these three menu items refer to the following printer groups: PrtType:PCL, PrtType:GDI, and PrtType:Other. The two remaining options are optional. Each is separated from the PrtType name and each other by a comma. Each option is a Yes or No (True or False) answer to the SuppressWindow and SuppressDlg options. By default the answer to both questions is True— which tells the system to suppress the windows.

NOTE: You can spell out the entries, such as *GDI,No*, to override the option, or you can simply enter a single character, such as *Y*, *N*, or *T*, *F*.

If you disable the Suppress option, one or both of the Print windows may appear. In the example above, the PCL print would display a window. GDI print would display a print window but would attempt to suppress the Windows GDI print window. Because the second option was omitted it defaults to True. The Other option suppresses the GUI window, or the *T*, but allows the Windows GDI print window to appear because of the *F* in the second position.

Remember if you are printing to the GDI device and have not specified enough information in the INI file, or incorrect information in the INI, the Windows' Print window may still appear.

Directing Print Streams

You can specify the folder you would like to send a print stream to while on the Print window. You use this INI control group to specify choices:

```
< Print_To_Folder >
    Test_Folder    = D:\DAP\Test_Folder
    Example_Folder = D:\DAP\Example_Folder
    Sample_Folder  = D:\DAP\Sample_Folder
```

The names of the folders you specify, such as Test_Folder, appear as options the user can choose from on the Print window. You can select one folder per print. You can also print to the same folder as many times as necessary. The system creates a unique file name for each print stream which cannot be overwritten. The names of these print streams are a combination of the user name, the date (using format E—Mar062003), and the time (192459 is 07:24:59).

Generating Print Stream Names Automatically

You can use the GenerateFileName option to tell the system to automatically generate names for print streams. The file name is comprised of the user ID, the date (such as Mar062003), and the time (such as 192459, which is 07:24:59 pm).

```
< Printer >
    GenerateFileName = Yes
```

When you set the option to Yes, the device on the Printer window is grayed out and the system generates a print file in the current directory.

Controlling Pagination when Editing a Form Set

You can use the AutoPagination option to control pagination when you edit a form set in Documaker Workstation or iDocumaker Workstation.

```
< EntryOptions >
    AutoPagination = No
```

Option	Description
AutoPagination	Enter No to turn off automatic re-pagination when editing a form set. The default is Yes.

NOTE: Keep in mind you can also use the AutoPagination option in the Control control group to prevent the system from re-paginating the form set when an image grows because of expanding text in a text area. Here is an example:

```
< Control >
    AutoPagination = No
```

This tells the system not to re-paginate the form.

Using Tab to Page through a Form Set

If you want to use the TAB key to move to the next page while in Archive View mode, set the following INI option. The default is *Yes*.

```
< ArcRet >
    TabNextPage = Yes
```

Viewing Print-only Forms

By default, the system skips forms on which no entry is required. These forms are called print-only forms. If you want to view these forms, set this INI option to *No*:

```
< Control >
    ActivateFirstField=Yes
```

The default (*Yes)* tells the system to display the first form with an active entry field. If set to *No,* the system activates the first *viewable* page of a form. *Viewable* pages are set by the DisplayPrintOnly or the DisplayAllEntryFormPages options.

To avoid viewing all forms (the DisplayPrintOnly option is set to Yes), use this option:

```
< Control > or < ArcRet >
    DisplayAllEntryFormPages = No
```

The default is *No*. Set this option to *Yes* if you want all pages with images marked for *Entry* or *Entry and Print* to appear. Assigning this option under the Control group affects Entry modes. Assigning it under the ArcRet control group affects Archive/Retrieval.

NOTE: If the DisplayPrintOnly option is set to *Yes* the appropriate group, the system ignores the ActivateFirstField option.

Customizing Units of Measure

By default, the system uses inches as the unit of measure. This means your rulers and grid lines are marked in increments based on inches. You can select a different unit of measure by modifying your FSIUSER.INI or FSISYS.INI files. This example shows how to set the rulers and grid to centimeters:

```
< HorizontalRuler >
   Type = 4
< VerticalRuler >
   Type = 4
```

Your other options are explained in this table:

To choose...	Set Type equal to...
none	0
FAP units (1/2400 of an inch)	1
Points	2
Inches	3
Metric (centimeters)	4
Picas	5
1/8 inch increments	6
1/6 inch increments	7

NOTE: You have the same unit of measurement options in the Image Editor. When you create an image, the system stores all measurement information in FAP units. Based on the user's INI settings, the system switches to the user's unit of measure for entry and display purposes. For example, if you create an image using inches, a data entry user in Documaker Workstation sees centimeters as the unit of measure if his or her INI files are set for centimeters.

Enabling Spell Checking

If you want to give users a spell check menu option you must make the following entry in the MEN.RES file:

```
MENUITEM "Spell Check All Fields..." 1087 "NULL" "NULL"
```

It doesn't matter which main menu you put this option under but our recommendation is to use either the WIP or the Formset menu. Users will be able to spell check data just entered into a field as well as imported data or data assigned in other ways (such as with a DAL script). Enabling spell checking through this method makes the checker available for all entry fields even if the form's creator has the spell option turned off in a field.

Setting Spell Check Options

You can use the SuggestDepth option in the Spell control group to control how the spell checker works. These options control how quickly and how completely the spell checker searches for errors.

```
< Spell >
    SuggestDepth = 0
```

To have the spell checker work	Enter
more quickly, but find fewer items	0 (zero)
fairly quickly and fairly accurately	50
slowly, but very accurately	100

The default is zero (0). If your users notice the spell checker is not finding errors in commonly used words, try setting the spell checker at a different setting.

Anchoring the Check Spelling Window

You can anchor the Check Spelling window and not have it float, depending on the location of the text it highlights. To anchor the window, add this option to your FSIUSER.INI or FSISYS.INI file:

```
< Spell >
    FixedDialogPos = Yes
```

The default is No.

Setting the Default Language

The base system uses US English as the default language for the spell checker. The spell checker supports these languages: Brazilian Portuguese, European Portuguese, Norwegian, Danish, US English, UK English, Swedish, Spanish, Italian, German, French, Finnish, and Dutch. It also supports hyphenation in these languages.

To have the spell checker default to a different language add the MainDicts option to your FSISYS.INI or FSIUSER.INI file, as shown here:

```
< Spell >
    MainDicts = ssceXX.tlx, ssceXX2.clx
```

You can choose from these languages:

For this language...	Enter...
Danish	ssceda.tlx, ssceda2.clx
Dutch	sscedu.tlx, sscedu2.clx
Finnish	sscefi.tlx, sscefi2.clx
French	sscefr.tlx, sscefr2.clx
German	sscege.tlx, sscefe2.clx

For this language...	Enter...
Italian	ssceit.tlx, ssceitl.clx
Norwegian	sscenb.tlx, sscenb2.clx
Portuguese (Brazil)	sscepb.tlx, sscepb2.clx
Portuguese (European)	sscepo.tlx, sscepo2.clx
Spanish	sscesp.tlx, sscesp2.clx
Swedish	sscesw.tlx, sscesw2.clx
UK English	sscebr.tlx, sscebr2.clx
US English	ssceam.tlx, ssceam2.clx

Setting a Default Locale

You can use the Locale option in the FSISYS.INI, FSIUSER.INI, or FAPCOMP.INI file to specify a default locale. The FSISYS.INI and FSIUSER.INI files are used by all Documaker software. The FAPCOMP.INI file is used by Docucreate.

In this INI file	The option...
FSISYS.INI FSIUSER.INI	Affects month naming in charts. For example if you set the locale to ARS (Argentina, Spanish) and you set your chart labels to month (short or long), the month name will be in Spanish.
	Lets you specify the default country for the Spelling Options window.
	Affects DAL date functions if the locality was not specified in the DAL script.
FAPCOMP.INI	Lets you specify the default country for the Spelling Options window.
	Lets you control the default date format based on the locale code you select.
	Affects month naming in charts. For example if you set the locale to ARS (Argentina, Spanish) and you set your chart labels to month (short or long), the month name will be in Spanish.
	Lets you control the paper size in the Image Editor and Form Set Manager. The system defaults to A4 or letter, depending on the locale code you select.
	Lets you control the default date format, based on the country code you select.

```
< Language >
    Locale = XXX
```

XXX represents the country code.

You can choose from these locale codes:

Country	Language	Code
Argentina	Spanish	ARS
Australia	English	AUD
Austria	German	ATS
Belgium	Dutch	BED
Belgium	French	BEF
Bolivia	Spanish	BOB
Brazil	Portuguese	BRC
Canada	English	CAN

Country	Language	Code
Canada	French	CAD
Chile	Spanish	CLP
Columbia	Spanish	COP
Costa Rica	Spanish	CRC
Denmark	Danish	DKK
Ecuador	Spanish	ECS
France	French	FLX
Finland	Swedish	FMK
Germany	German	DEM
Guatemala	Spanish	GTQ
Iceland	Icelandic	ISK
Indonesia	Indonesian	IDR
Italy	Italian	TL
Ireland	English	IEP
Liechtenstein	German	CHL
Luxembourg	German	LUX
Luxembourg	French	FRF
Mexico	Spanish	MXN
The Netherlands	Dutch	NLG
New Zealand	English	NZD
Norway	Norwegian	NOK
Panama	Spanish	PAB
Paraguay	Spanish	PYG
Peru	Spanish	PES
Portugal	Portuguese	PTE
South Africa	Affrikaans	ZAA
South Africa	English	ZAR
Spain	Basque	ESB

Country	Language	Code
Spain	Catalan	ESC
Spain	Spanish	ESP
Sweden	Swedish	SEK
Switzerland	German	CHF
Switzerland	French	CHH
Switzerland	Italian	CHI
United Kingdom	English	GBP
United States	English	USD
Uruguay	Spanish	UYU
Venezuela	Spanish	VEB

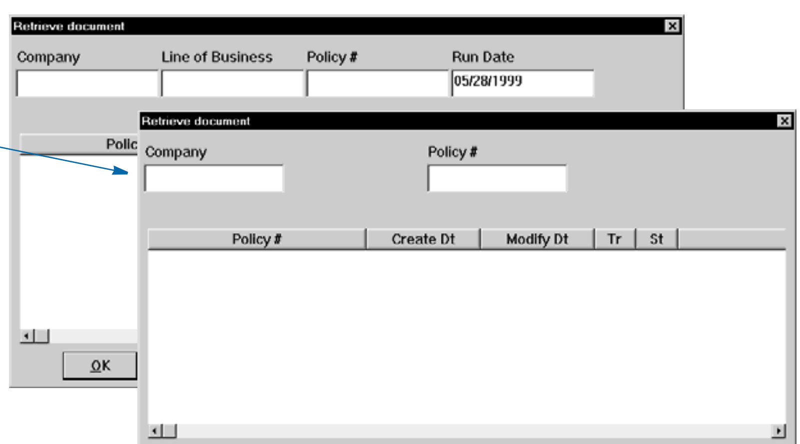
Removing Fields from the Retrieve Document Window

You can add the following options in the ArcRet control group to remove key fields from the Retrieve Document window. You can also turn off the display of the Run Date field.

```
< ARCRET >
;   HideKey1    = Yes
    HideKey2    = Yes
;   HideKeyID   = Yes
    HideRunDate= Yes
```

In the example shown above, the Key2 field and the Run Date field are hidden. The semi-colons comment out the other options. If you omit these options, comment them out, or set them to No, the system displays the field. Here is an example of the result:

The options above tell the system to hide the Key2 (Line of Business) and Run Date fields, as shown here.



MAKING SURE REQUIRED FIELDS HAVE DATA

You can include the RACCheckRequiredFields function in your MEN.RES or WIPEDIT.RES file to have the system make sure required fields have data.

This table describes what the RACCheckRequiredFields function does:

If...	The function displays
The required fields have data	*All required fields have data.*
One or more of the required fields is missing data	*Data required for [name of field].* The system makes the first required field with missing data the current field.

Setting up a required fields check with WIP Edit

There is a RACLIB function that should be placed in the WIPEDIT.RES file. This function may send a response back to WIPCTL or it may display a message box, depending on whether it is executed from WIP Edit or AFEMAIN. It is set up like any MEN.RES function and must be identified by a command number not used by any other function in the WIPEDIT.RES file. The setup in the WIPEDIT.RES file must occur regardless of the way you start the required field check.

Here is an example line from the WIPEDIT.RES file:

```
MENUITEM "&CheckRequiredFields" 263 "RACW32->RACCheckRequiredFields"
"Check Required"
```

You do not need to make changes to IDS.

There are two ways to start the required field check:

- From a menu
- From an ASP or JSP page

Starting from a menu

To let users start a required field check from the menu add the following line to the WIPEDIT.RES file. The number 263 may vary but it should not be used for another function in the WIPEDIT.RES file. This line tells the system to include a Check Required Fields option on the menu that appears when you right click the mouse.

```
MENUITEM "&Check Required Fields" 263 "RACW32-
>RACCheckRequiredFields" "Check Required"
```

Starting from an ASP page

Follow these steps to start a required field check from an ASP page.

1 Register the WIPCTL.DLL file.

NOTE: The WIPCTL program (WIPCTL.DLL) contains the WIP Edit plug-in which lets custom web applications send messages to WIP Edit to do things like zoom in, zoom out, advance to the next page or form, and so on. This component must be registered with regsvr32. See Using Web-Enabled Solutions for more information on the WIPCTL feature.

2 Set up the WIPEDIT.RES file to execute the required field check from the menu.

3 The web page must contain a script like the following. When this script executes, the required field check will occur. The variable rspmsg will contain either “All required fields have data” or an error indicating the field that needs data. The RACCheckRequiredFields function will be executed in this case as well but its response is returned to the script, not displayed in a message box.

```
< Script Language="VBScript" >
    Function SendToWipedit
        Dim rspmsg
        set aspobj = CreateObject("Wipctl.WipEd.1")
        aspobj.cmdWithMessage 263,rspmsg
        set aspobj = Nothing
        MsgBox(rspmsg)
    End Function
</Script>
```

Automatically Re-Paginating Images

Beginning with version 11.2, when an image changes size, Documaker Workstation automatically re-paginates pages that consist of multiple floating images.

Understanding the System

This applies to both Documaker Workstation and the WIP Edit plug-in. Documaker Server waits until the end of its processing to handle pagination. Earlier versions of Documaker Workstation did not re-paginate a form when a page consisted of multiple images and one of those images changed size.

If a text area causes an image to grow or shrink — due to embedded fields changing or because it was a multi-line text field — the form that contains the image automatically re-paginates.

If you designed the form using Documaker Studio, the positioning information you established is reapplied to the form each time changes in a text area cause the dimensions of an image to change. You can, however, use the AutoPagination option to disable automatic re-pagination:

```
< Control >
    AutoPagination = Yes
```

Option	Description
AutoPagination	Enter No if you do not want the system to automatically re-paginate when image dimensions change. The default is Yes.

Keep in mind...

- The system honors positioning information designed into the form via Documaker Studio. Positioning information is stored via SetOrigin rules.
- If a form consists of multiple images on a page, but those images comprise more space than defined for the page size, the system automatically paginates that page and moves the images that did not fit to a new page.

- If an image grows to push another image such that its positioning rule causes it to encroach on a defined footer or the bottom of the page, that image is moved to the next page and the entire form will be have the SetOrigin rules reapplied.
- When designing a form, avoid having a footer image that uses a relative position. This ultimately means there can only be one image on the page that is not a header or footer. Footers, typically should be placed using a rule that makes sure it has a relationship to the bottom of the page.
- When images shrink (due to text area shrinking) an image from the next page may be brought back to the current page. In other words, images can not only flow to the next page, but they can come back when space allows.
- Remember that the positioning (SetOrigin) defined in the form is applied. So although there may appear to be a space large enough to hold an image, you also have to account for any additional adjustments applied by the SetOrigin rules.

  For example, suppose you have an image that is 2 inches in height, but the SetOrigin rule for that image specifies a relative placement 1/2 inch from the previous image. In this case, 2 1/2 inches of space is required for that image to fit on the page. If there is less than 2 1/2 inches remaining before encountering a footer image or the bottom of the page layout, then that image will move to the next page.
- Images can only flow to and from pages that were created during pagination. If a page was specifically designed into the form via Documaker Studio, then no images will move onto that page from a prior page. Images can only move to or from pages that were created by overflowing their defined page.

Using the Complete and Exit Option

You can add an option to your MEN.RES file which lets you complete the current form set and then exit the system. This option combines the Complete and Exit menu options.

If there is an error on the form set, the system will not complete the form set and exit the system until you correct the error.

Understanding the System

This option is typically used with iDocumaker Workstation or iPPS implementations but it can also be used if you have created specialized applications that launch entry via the RACLIB interface. Documaker Workstation and PPS users can also use this option if users typically exit the application after completing a form set.

To add the Complete and Exit option to your system, you must first add the following line to the MEN.RES file:

```
MENUITEM "Complete and &Exit" 106 "AFEOS2->AFEComplShutDown"
"Complete the form set and exit the system"
```

You can add this line, or a line similar to it, under any menu group you like. You can also change the text of the option.

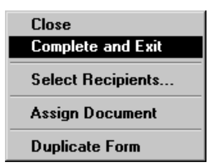
NOTE: In this example, *106* is used as the ID for the menu function. You can assign any ID between 100 and 200, as long as it is not used by another menu function.

The ampersand (&) indicates that the next character is the accelerator for this menu option. You can omit the ampersand if you like.

For instance, you could place this option on the Formset menu, as shown here:

```
POPUP "Form&set" 170 "Formset"
    BEGIN
        MENUITEM "&Close" 1069 "Close document" "NULL"
        MENUITEM "Complete and &Exit" 106 "AFEOS2->AFEComplShutDown"
"Complete the form set and exit the system"
        SEPARATOR
        MENUITEM "Select &Recipients..." 187 "AFEW32-
>AFESelectRecipients" "View forms by recipient"
        SEPARATOR
        MENUITEM "&Assign Document" 150 "AFEW32->AFEAssignDocument"
"Assign Document"
        SEPARATOR
        MENUITEM "&Duplicate Form" 104 "AFEW32->AFEDuplicateForm"
"Duplicate current form"
    END
```

To see this menu:



Using Workstations as Print Servers

You can configure a workstation to function as a *print server*. When you do this, the user can continue entering data while the system prints forms. The print server controls the flow of printer data and monitors the available drive space for the spool file. Based on a value you set, the print server will not send additional print data if it begins to run out of space. The print server performs a similar function for archive operations — monitoring available drive space based on a value you set.

The print server can run on one workstation for all users or on each workstation. The only limitation for having one print server for several workstations is that all workstations must have the same network drive mappings or all resource libraries must be located on the same drive. The print server *must* have access to the same WIP data and resource libraries as do the workstations.

Setting Up INI Options

To set up a print server, you must add the following control group and options (sample 32-bit WIndows values are shown):

```
< BatchPrint >
    Program            = PRSRVW32.EXE
    Path               = K:\FAPDEMO\BATCH\
    AutoStart          = No
    SpoolDrive         = P:
    SpaceNeeded        = 10000000
    ArchiveDrive       = R: or
    ArchiveSpaceNeeded = 5000000
```

Option	Description
Program	The name of the print server program. Enter PRSRVW32.EXE for 32-bit Windows environments.
Path	The full path to the batch print queue. All workstations using the print server *must* point to the same directory.
AutoStart	Tells 32-bit Windows environments whether or not to start the print server in the background.
SpoolDrive	The drive where the network spooler is located.
SpaceNeeded	Defines how much space (in bytes) must be free on the network drive on which the spooler is located. You define the drive using the SpoolDrive option.
ArchiveDrive	The drive where the archive files are located.
ArchiveSpaceNeeded	Defines how much space (in bytes) must be free on the network drive on which the archive files are located. You define the drive using the ArchiveDrive option.

Setting Up INI Options for Multiple Environments

If you have multiple environments, such as Windows XP and Windows 2000, you can set up substitutions for the Program option. This option defines the print server program.

NOTE: You must make sure the appropriate print server program is located on each workstation.

Instead of having a separate Program option setting for each platform, you can enter **PRSRVW32.EXE** as the setting for the Program option in the BatchPrint control group and include the following control group and option to the same INI file:

```
< Windows32Subs >
    PRSRVW32.EXE = PRSRVW32.EXE
```

Setting Up Email Support

Documaker Workstation includes an application independent interface to electronic-mail systems. Currently, Documaker Workstation supports Lotus VIM (cc:Mail) and Microsoft MAPI (MSMail, WFW Mail, MS Exchange) through DLLs you can install.

Only core level functions are addressed in each supported email system. This lets Documaker Workstation support a large number of email products without requiring you to have a specific implementation or version of the email software. Documaker Workstation uses an email system's transport abilities, it does not attempt to be an alternate email application interface.

Most installed email products require no changes to enable email services in Documaker Workstation. This means you can install Documaker Workstation in environments where a supported email product has already been established.

As implemented, Documaker Workstation does not maintain a continuous open *session* with the email system. Instead, specific user-initiated events occur which tell Documaker Workstation to open an email session, perform a task, and then close the session.

Using Lotus VIM (cc:Mail)

The Vendor-Independent Messaging (VIM) API is the only interface available on Windows workstations. Lotus offers two products using the VIM interface: cc:Mail and Lotus Notes.

Documaker Workstation expects to load the VIM.DLL and uses VIM Specification 1.0 functions. These core level functions are usually provided by any VIM mail application. If you use a 32-bit version of Documaker Workstation with a 16-bit version of cc:Mail see Using 16-bit VIM Under 32-bit Windows on page 232.

Documaker Workstation automatically substitutes the correct module name of the email DLL if you use a basic name. *VMMail*, is the MailFunc function that serves as the entry point to this DLL.

Troubleshooting

If an email transmission generates an error code such as:

```
Platform error
FAP Error Code = 1
VIMLIB
Receive failed to complete successfully.
GUI Error Code = 0xfffffff
```

there are several possible explanations for the failure. Check the following possibilities for a solution to the problem:

- VIMLIB indicates the user is attempting to use cc:Mail or Lotus Notes to mail from WIP. Check to make sure the correct VIM DLLs are installed. If cc:Mail or Notes has not been installed, the VIM DLLs may not be there.
- If you have the VIM DLLs installed, it is possible that you don't have the VIM DLLs in your path and therefore they will not load.
- If you are trying to use Notes, there is also a Notes requirement that the NOTES.INI be located in the path as well. If you don't intend to do VIM, then you need to change your INI MAILTYPE = statement to point to your MSM (Microsoft Mail) INI group.

You can use the following INI option to turn on debug log tracing the Email Print Driver:

```
< Debug_Switches >
    EPT_Debug = Yes
```

The EPT_Debug option generates addition trace information about errors within that driver. This includes outputting the names of the printer and email drivers that are referenced.

If you need error trace information from the referenced email driver, activate the appropriate option for that email driver. For instance, use the VIM_Debug option to get additional error information for the Lotus (Notes or cc:Mail) interface driver.

Using 16-bit VIM Under 32-bit Windows

Windows XP and Windows 2000 are 32-bit environments that can run both 16-bit and 32-bit Windows software. Although these platforms can run both program types, 32-bit programs cannot automatically link directly with 16-bit programs. For this reason, use an email program that's designed for the same platform your system runs on.

NOTE: Under Windows XP or Windows 2000, you cannot use a 16-bit Windows version of cc:Mail with a 32-bit version of Documaker Workstation.

Lotus does offer a 32-bit version of cc:Mail, however, Lotus has also made publicly available several Windows 32-bit DLLs which you can use to communicate with your existing cc:Mail post office without upgrading. You can download these DLLs via anonymous FTP from:

ftp.support.lotus.com

Look for the ccapi32.zip file in the following directory:

/pub/comm/ccmail/dev_tools

NOTE: These files belong to and are made available by Lotus. Lotus may, at any time, withdraw their availability and require you to upgrade.

When you unzip the CCAPI32.ZIP file, you will find these DLLs:

- MEMB632.DLL
- CHRSET32.DLL
- VIM32.DLL

Place these DLLs in your cc:Mail program directory or at least in the path searched. Once installed, you will be able to use the Windows 32-bit version of our program to send and receive WIP via cc:Mail.

Using Microsoft MAPI

The Microsoft Messaging Application Interface (MAPI) is only available for Microsoft Windows environments. This interface supports Microsoft Mail, Windows for Workgroups Mail, Outlook, and Microsoft Exchange.

Documaker Workstation expects to load the MAPI.DLL (or MAPI32.DLL under Windows) and only requires a subset of the MAPI API specification called *Simple MAPI*.

Documaker Workstation automatically substitutes the correct module name of a DLL if you use a *basic* name. For MAPI support, set the Module option as shown here to support all environments:

```
Module=MSMW32
```

MSMWIN.DLL is the module name of the DLL for 16-bit email systems. *MSMW32.DLL* is the module name of the DLL for 32-bit email systems (MS Exchange).

MSMMail is the MailFunc function that serves as the entry point to this DLL.

Debugging

You can use the following INI options for debugging. These options turn on debug log tracing for Microsoft Mail (MSM_Debug = Yes) and the Email Print Driver (EPT_Debug = Yes):

```
< Debug_Switches >
    MSM_Debug = Yes
    EPT_Debug = Yes
```

Use the EPT_Debug option to generate addition trace information about errors within that driver. This includes outputting the names of the printer and email drivers that are referenced.

If you need error trace information from the referenced email driver, activate the MSM_Debug option for the Microsoft Mail (Outlook or MAPI) interface driver.

Using Microsoft Exchange Mail Server

If you are running Windows 2000, you will probably be using Microsoft Exchange as your email carrier. In this case, make sure you have created a profile for your mailbox name. Your mailbox name is the name or ID that you use to log into the email system.

If you do not have a profile for your mailbox name, follow these steps:

1. Open the Properties window for Microsoft Exchange and click the Show Profiles button.

   If an item in the profile list has your mailbox name, you do not have to make any changes, click Cancel. If not, click Add.

2. On the Input Setup Wizard window, check the appropriate option to indicate this profile should use Microsoft Mail.

   If Microsoft Mail is not one of the options, you must insert the Windows 2000 setup disk and add the software support for this option.

3. Make sure the Use the Following Information Services button is activated, then click Next.

4. Enter a profile name that matches your mailbox name—the ID you type on the email login window. Then click Next.

5. Depending upon whether Microsoft Mail was the only service you selected, the next few windows may vary. Each collects important information that is specific to each service that you identified.

The Microsoft Mail windows will collect the Post Office location, your user name, and make you type your password to verify who you are. Then you identify where to locate your personal address book and folders.

You may be asked if you want to convert old information. Before converting, check with your email administrator. If you are unsure, do not convert.

6 Decide whether Microsoft Exchange should be started when Windows 2000 starts. Documaker Workstation does not require you to do this, so it is your decision.

Setting Up Documaker Workstation

Enabling email support within Documaker Workstation is easy. All you have to do is set up two control groups and options in the FSISYS.INI or FSIUSER.INI files that specify which email system is installed and a few other access parameters.

The Mail control group identifies the specific control group in the INI file which contains the email system requirements. This control group should appear as shown here:

```
< Mail >
    MailType        = xxx
    SendForms       =
    ReceiveForms    =
```

Option	Description
MailType	*xxx* identifies a specific INI control group appended with the MailType option, such as MailType:CCM for cc:Mail or MailType:SMTP for SMTP.
SendForms	Enter Yes if you want to send FAP files along with the WIP policies. The default is No which means that you will receive the forms into WIP but no FAP files.
ReceiveForms	Enter Yes if you want the recipient to receive the FAP files into the forms directory of the MRL.

The MailType:xxx control group contains the system requirements necessary to load and use the email system. In addition, you can specify default user login information in this control group. This control group appears as shown below:

```
< MailType:xxx >
    Module          =
    MailFunc        =
    Name            =
    UserID          =
    Password        =
    From            =
    AltFrom         =
    DataPath        =
    ShowDataPath    =
    HiddenMsgSupport=
```

Option	Description
Module	The Documaker Workstation DLL that supports the email system.

Option	Description
MailFunc	Exported DLL function name of the email handler.
Name	Name of the system (Identifies the system on internal dialogs).
UserID	Default user ID for logging in.
Password	Default password.
From	For SMTP, this option lets you specify who the email was from.
AltFrom	For SMTP, this option lets you specify an alternative from address, indicating where the email was from.
DataPath	Default data path for the email system, if required.
ShowDataPath	Set to No to suppress the Data Path field on the email logon window. For some email systems, such as Microsoft Mail or Lotus Notes, the Data Path field is not applicable.
HiddenMsgSupport	Set to No if you are having trouble retrieving messages into WIP. See the following Understanding the System for more information.

The Module and MailFunc options are required for all email implementations. Some systems may require you to define one or more of the other options in the INI file. Most implementations simply use the additional login information to either eliminate or pre-fill login windows.

Migrating from cc:Mail to Outlook

If you are migrating users from cc:Mail to Microsoft Outlook, keep in mind that cc:Mail uses Skywire Software's VIM library which does not support the application (hidden) mail-type that Outlook (MAPI) does.

For this reason, if someone from cc:Mail sends a form set to an Outlook client, the message ends up in the regular Inbox and does not use the application mail-type Documaker Workstation normally expects.

Use this INI option to have MAPI support check for messages with Documaker Workstation's application mail-type and the Inbox for messages of our Subject type.

```
< MailType:MSM >
   ReceiveFromInbox = Yes
```

When set to Yes, this option tells the system to check for all types of messages, not just our application-type messages.

Understanding the System

When possible, the system tries to send WIP transfers as *private application* mail, so the email transferring the WIP does not appear in the user's inbox. The system checks for emailed WIP and receives them automatically or manually, depending on how you set it up.

Some email systems do not support private application mail. In those systems, it really does not matter how you set the HiddenMsgSupport option.

Some email systems that do support private application mail can be configured in such a way as to make it impossible for the system to retrieve the WIP transfers. Since the email is never visible in any inbox, the messages can go unnoticed. It is in these instances, that you would set the HiddenMsgSupport option to No. This lets the system retrieve messages into WIP.

When you have the HiddenMsgSupport option set to No or you are using an email system that does not support private application mail, the messages that contain the WIP transfer file are visible in the user's inbox. It is important that the user does not redirect or remove these messages manually within the email application, but instead continues to use the system to receive the messages into WIP. The system will remove the messages from the user's Inbox after receipt.

For example, assume you are using cc:Mail as the email application. Your INI file would include the following control groups and options:

```
< Mail >
    MailType    = CCM
< MailType:CCM >
    Name        = cc:Mail
    Module      = VIMWIN
    MailFunc    = VMMail
    UserID      = ABR
    Password    = PWORD
    DataPath    = V:\CCDATA
```

Logging In

If the options in the control groups do not provide enough information for your email system to login successfully, a login window appears. On this window, the user can enter the user ID, password, and any other required information.

Once a successful login occurs, the login information is stored in memory. Subsequent email operations use this information instead of the information in the INI file during when the user attempts to login. This approach minimizes the number of times the user must fill in the login window.

Once the user exits Documaker Workstation, this information is removed from memory—the user will have to re-enter the information the next time he or she starts Documaker Workstation and attempts to login to the email system.

Note: If you are using Microsoft Exchange as your email carrier and you cannot login, see Using Microsoft Exchange Mail Server on page 233 for more information.

Setting Up Addresses

When a Documaker Workstation email message requires a destination address (a recipient), a window appears which lets the user select a recipient. Documaker Workstation retrieves this list of recipients from the email system. The list usually corresponds to the default post office opened during login.

Documaker Workstation returns only one email address from the selection window.

Adding Information to an Email Subject Line

You can use the Append_Subject INI option to add information to the subject line when sending emails via Documaker Workstation and Documaker Server. This option will work with all features that use the email system, such as EPT, routing, and so on.

Here are some examples of how you can use this option:

```
< MailType:MSM >
    Append_Subject = [ ~Key1 | ~USERID ]
```

or

```
    Append_Subject = ~Key1 | ~USERID
```

or

```
    Append_Subject = ~Key1
```

or

```
    Append_Subject = [ ~DALRUN ]
```

or

```
    Append_Subject = Skywire
```

In these examples, the tilde (~) in front of Key and USERID tells the system to retrieve the Key1 and user ID values from the form set currently loaded in memory. If no form set is in memory, the value for the Append_Subject option will be blank.

The example *~DALRUN* tells the system to run a DAL script if necessary.

The pipe symbol (|) between KEY1 and USERID is not required. It is only there to separate the two values. For example, if the value for ~KEY1 is *Skywire*, and the value for ~USERID is *James Brown*, the subject line will look like this:

```
FILE_FAPMSG [ Skywire| James Brown]
```

Without the pipe symbol (|), the subject line will look like this:

```
FILE_FAPMSG [ Skywire intJames Brown]
```

Customizing the Confirmation Message

When using Documaker Workstation to send a WIP document from one system user to another, you can specify what should appear in the confirmation message.

For example, to specify Account Number:

```
Package CC'd to Tyler, Susan.
Account Number is 33333
```

Set the KeyIDTitle option in the DlgTitles control group:

```
< DlgTitles >
    KeyIDTitle = Account Number
```

The default is KeyID.

CHAPTER 5

Importing and Exporting Information

The system's data import and export features make the data entry process quicker, more efficient, and more accurate. During an import, the system automatically fills specific variable fields with data from another application.

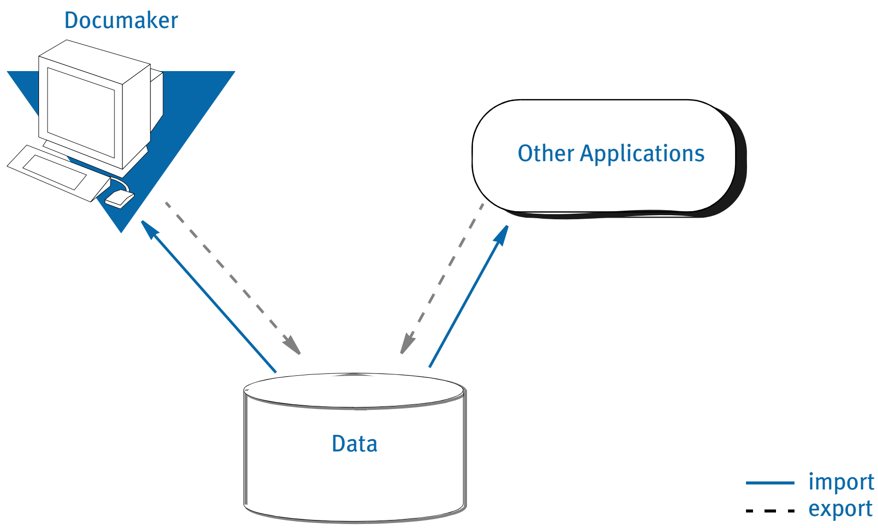
These features reduce the need to enter data multiple times and help avoid data entry mistakes. Likewise, you can export data from the system to other applications. The export process provides the same benefits as the import process. This chapter includes information on...

In addition, Skywire Software has available technical documentation you can request from your sales representative or from Support Services.

Importing and Exporting Files

When users export data, they send a file to a specified directory so other applications can use that same data. The system's import and export functions work with the manual data entry features. You can import and apply common data, such as a name and address, then manually add the information specific to the form set. By importing common data and manually entering the remaining data, you can save time and reduce data entry errors.

The export function lets you extract data from the system, and then use the data in another application.



The system includes two import functions. You can use the Standard Import to import files which contain data for specific form sets. And, you can use a Selective Import to select import files that are specific to form set transaction types and user access levels.

NOTE: When you import multi-page images, the system includes after the field name a pipe symbol (|) with an *M* and the page number. Here is an example:

\NA=qvrfldlM2\

Standard import

A *standard import* file contains data for a specific form set. The import file contains a header record which the system uses to match the data with the appropriate form set. When you select a Standard Import function, the system displays a window which contains a list of import files. You must know what data the import files contain, and what type of form set transaction the data relates to, to choose the proper import file.

Selective import

A *selective import* function contains a driver file and import files. The driver file filters the import files based on the user's form set selections. The driver file recognizes the current user and the selected transaction type, and displays only the import files which correspond to that user ID and form set transaction. The driver file can include a brief description of each import file, which helps the user select the appropriate file.

Export file

An *export file* contains data from the system. You can apply a generic format to this data so other applications can read the exported data.

Setting Up the INI Files

You must perform these tasks to enable and configure the import and export functions.

- Modify the FSISYS.INI file
- Modify the FSIUSER.INI file

This chapter focuses on setting up your system so you can import and export information and outlines the settings your INI files must contain. For more information about modifying INI file settings, see Configuring INI Files on page 128.

Each resource library contains two INI files which you must configure to use the import and export features. You enable or disable import and export options and select the import and export file formats in the FSISYS.INI file. In the FSIUSER.INI file, you specify the directory where the system stores the files.

Note: You must configure the INI files for (and in) each resource library for which you create import and export files.

Setting Up the FSISYS.INI File

When you set up your resource libraries, you may have several import files. If this is the case, your users can select from a list of files to import.

You enable or disable import and export functionality, and define the format of import and export files in the ImportFormats and ExportFormats control groups in the FSISYS.INI file. These control groups contain options which define import and export formats and function calls. The control groups contain option lines for specific import and export formats. Each option line includes these semicolon delimited parameters. Here is an example:

```
01=;TD;Standard Import;TRNW32->TRNImport;
02=;TD;Selective Import;TRNW32->TRNSelImport;
```

Syntax

```
;_NOT_USED;_DESCRIPTION;_DLL NAME->FUNCTION;
```

Parameter	Description
01	Any two-character numeric value for sorting purposes
NOT USED	This field is not used by the system. You can use this field for a code for the file format. If not used, place a semicolon to mark the field.
DESCRIPTION	The import format description which appears in the import window.
DLL-> FUNCTION	The Dynamic Link Library (DLL) and function to call when a user selects an import file. *Do not modify this parameter.*

If either the ImportFormats or the ExportFormats control group is missing from the FSISYS.INI file, or if a group contains no options, the import or export feature is disabled.

NOTE: For more information about INI files, see Configuring INI Files on page 128.

CONFIGURING IMPORT FORMAT OPTIONS

Configuring import format options lets you enable and modify specific import formats. The SAMPCO resource library installed with the system contains a default import file format. You can modify the default formats to customize the import format selections displayed for the users. You can also modify the numeric value to display a format in a different order.

Setting Up Import Formats

Follow these steps to set up an import format:

1. Open the FSISYS.INI file in the resource library for which you want to use import files. You can use any text editor to open this file.
2. Locate the ImportFormats control group. Most text editors have a find or search function you can use to quickly find this group heading.
3. Add or delete the following lines, depending on the import format you want to use:

For this import format	Enter...
Standard	**01=;TD;Standard Import;TRNW32->TRNImport;**
Selective	**02=;SI;Selective Import;TRNW32->TRNSelImport;**

Modifying Import Format Settings

Follow these steps to change import format settings:

1. Open the FSISYS.INI file in the resource library for which you want to use import files. You can use any text editor to open this file.
2. Locate the ImportFormats control group. Most text editors have a find or search function you can use to quickly find this group heading.
3. Select the format option you want to modify.
4. Change only the numeric and the description parameters as desired, using the table in the previous topic as a guideline. The format description appears in the Import window when you select the import option.

CONFIGURING EXPORT FORMAT OPTIONS

Configuring export format options lets you enable and modify specific export formats. You can create export files that are compatible with the import feature or using an older method with or without headers.

The standard export format is compatible with the import format. This format contains all of the key information used to create the form set, as well as, field data grouped by the form and image on which they occurred.

The older export methods are supported for legacy systems. If headers are included, the headers indicate the specific form set where the exported data originated. This format does not include any form information and omits much of the form set definition information. If you enable export files with or without headers, you should indicate this fact in the format description.

Setting Up Export Formats

Follow these steps to set up export formats:

1. Open the FSISYS.INI file in the resource library for which you want to use export files. You can use any text editor to open this file.
2. Locate the ExportFormats control group. Most text editors have a find or search function you can use to quickly find this group heading. Then add or delete the following lines to indicate the export formats you want to use:

For this export format	Enter...
Standard export format	**01=;V2;Export Version 2;TRNW32->TRNExportV2;**
Old method, without headers	**02=;NH;Export no Header;TRNW32->TRNExportNoNA;**
Old method, with headers	**03=;TD;Standard Export;TRNW32->TRNExport;**

Modifying Export Format Settings

Follow these steps to change export format settings:

1. Open the FSISYS.INI file in the resource library for which you want to use export files. You can use any text editor to open this file.
2. Locate the ExportFormats control group. Most text editors have a find or search function you can use to quickly find this group heading.
3. Select the format option you want to modify.
4. Change only the numeric value and the description, using the following table as a guide. The format description appears in the Export window when you select the export option.

Parameter	Description
01	Any two-character numeric value for sorting purposes
DESCRIPTION	The description of the import format which appears in the import window.

SETTING UP THE FSIUSER.INI FILE

Since both standard and selective imports use the same import files, you can store these files in the same directory. You specify the directory paths in the FSIUSER.INI file.

Configuring Import Options

There are several options you can use to configure the import feature so it works the way you need it to work. Simply omit any option you do not want to use. Here's how to find the section of the INI file which contains import options:

1. Open the FSIUSER.INI file in the resource library for which you want to use import files. You can use any text editor to open this file.
2. Locate the ImpFile_cd control group. Most text editors have a find or search function you can use to quickly find this group heading.

If not found, you will need to add this group to your INI file. Simply type in the text, exactly as shown above, and be sure to include brackets (< ImpFile_cd >).

The import options follow the group heading. Usually, the order in which the options appear does not matter. In some cases, however, INI options are dependent on other INI options. These dependencies are discussed in the description of each option.

Setting Up the Default Import File Name

```
< ImpFile_cd >
    File = file name
```

Substitute a valid file name where indicated. There is no default for this option.

The system uses the file name you enter as the default in the File window the user sees when selecting the import file.

Setting Up the Default Import Directory

```
< ImpFile_cd >
    Path = MSTRRES\(resource library name)
```

Substitute a valid drive and directory path where indicated. There is no default for this option.

Specifying the import directory lets you map where your import files are located. If you omit this option, the system looks for your import files in the working directory specified during installation (see Installing the Software on page 8 for more information).

The system uses the path you enter here to determine the default directory displayed in the File window. If you use the File option and the file name specified there also includes a path, that path *overrides* one entered here.

Setting Up the Default Import File Extension

```
< ImpFile_cd >
    Ext = .DAT
```

Substitute a valid file extension where indicated. Be sure to include the period. The default is *.DAT*.

On the File window, the system will use this default extension to fill the selection list with available files.

If you use the File option and the file name you entered there includes an extension, that extension overrides one entered here.

Setting Up the Driver File for Selective Import

```
< ImpFile_CD >
    SelectionFile = driver file name
```

Substitute a file name where indicated. This file name indicates the driver file you want to use for selective imports. There is no default for this option.

If you enter a valid driver file name here, the system will not display the File window. If you omit this option or if the file name you enter is invalid, the system will display the File window so the user can select a driver file to import.

Preventing Users from Changing Imported Data

```
< ImpFile_CD >
    ProtectFlds = Yes/No
```

Substitute Yes or No where indicated. The default is Yes, which means the contents of imported fields cannot be changed from within the system.

This option defaults to Yes because import data typically comes from another application. If the data is incorrect, most users prefer to correct the data in the original application and re-import the data into the system.

If you are using the import file to provide default data, and you do not mind if users change the imported data, set this option to No.

NOTE: Keep in mind that normally the system only lets users change a global field the first time the field is accessed. Your system may function differently if it has been customized.

Ignoring Invalid Groups in Import Headers

```
< ImpFile_CD >
    IgnoreInvalidGroup = Yes/No
```

Substitute Yes or No where indicated. The default is No. All Key1 and Key2 group names must correspond to those specified in the FORM.DAT file.

If you turn this option on, which it *is not* recommended, the system attempts to import the data into the current form set. Since the system does not check the form group information, there is no way to make sure the data in the import file is valid for the current form set. Any information in the import file which does not correspond with fields in the form set simply disappears.

Importing and Exporting a Form Set that has Images added using a DAL Function

You can use the DAL functions AddImage and AddForm to add images or forms to a form set and then export the information. To use this capability, you must use the Standard Export Version 2 export option and add the IgnoreInvalidImage option.

For example, first make sure your ExportFormats control group looks like the one shown here:

```
< ExportFormats >
    01 = ;V2;New Std Export w Header  ;TRNW32->TRNExportV2;
    02 = ;TD;Old Std Export w Header  ;TRNW32->TRNExport;
    03 = ;NH;Old Std Export w/o Header;TRNW32->TRNExportNoNA;
```

Then, add the IgnoreInvalidImage option to tell the import function to include any images added to the form set by the AddImage or AddForm DAL functions.

```
< ImpFile_CD >
    IgnoreInvalidImages = Yes
```

Be sure to set this option to Yes to avoid an error message. The default is No.

CONFIGURING EXPORT OPTIONS

As with import, there are several options you can use to configure the export feature so it works the way you need it to work. Simply omit any option you do not want to use. Here's how to find the section of the INI file which contains export options:

1. Open the FSIUSER.INI file in the resource library for which you want to use export files. You can use any text editor to open this file.

2. Locate the ExpFile_CD control group. Most text editors have a find or search function you can use to quickly find this group heading.

If not found, you will need to add this control group to your INI file. Simply type in the text, exactly as shown above, and be sure to include braces ([ExpFile_CD]).

The export options follow the group heading. Usually, the order in which the options appear does not matter. In some cases, however, INI options are dependent on other INI options. These dependencies are discussed in the description of each option.

Setting Up the Default Export File Name

```
< ExpFile_CD >
    File = file name
```

Substitute a valid file name where indicated. There is no default for this option.

The system uses the file name you enter as the default in the File window the user sees when selecting the export file.

Setting Up the Default Export Directory

```
< ExpFile_CD >
    Path = MSTRRES\(resource library name)
```

Substitute a drive and directory path where indicated. There is no default.

Specifying the export directory lets you map where your export files are located. If you omit this option, the system looks for your export files in the working directory specified during system setup (see Installing the Software on page 8 for more information).

The system uses the path you enter here to determine the default directory displayed in the File window. If you use the File option and the file name specified there also includes a path, that path overrides one entered here.

Setting Up the Default Export File Extension

```
< ExpFile_CD >
    Ext = .OUT
```

Substitute a valid file extension where indicated. Be sure to include the period. The default is *.OUT*.

On the File window, the user can use the List of Files field to display only those files with the export file extension.

If you use the File option and the file name you entered there includes an extension, that extension overrides one entered here.

Appending to an Existing Export File

```
< ExpFile_CD >
    AppendedExport = Yes/No
```

Substitute Yes or No where indicated. The default is No.

If you set this option to Yes and the export file exists, the system does not display the Confirm Overwrite message to the user. The system simply appends the current form set data to the existing file.

If the file name you specified does not exist, the system creates a new file using that name.

Suppressing the Confirm Overwrite Message

```
< ExpFile_CD >
    Overwrite = Yes/No
```

Substitute Yes or No where indicated. The default is No.

Enter Yes to turn off the Confirm Overwrite message if a user chooses an existing file as the export file.

The system ignores this option if the AppendedExport option is set to Yes. Appending to an export file implicitly means the export file will not be overwritten.

Suppressing the Export File Selection Window

```
< ExpFile_CD >
    SuppressDlg = Yes/No
```

Substitute Yes or No where indicated. The default is No.

If the File option was not defined, or if it was defined improperly, the window appears regardless of what you enter here.

The File Selection window is also affected by the AppendedExport option and the Overwrite option. If you set either of these options to Yes, setting the SuppressDlg option to Yes tells the system not to display the File Selection window.

If you set those options to No, the File Selection window appears unless the export file does not exist. This prevents the user from accidentally overwriting a valid export file without confirmation.

Controlling the Default Export Button on the Complete Window

Please note that this option *does not* appear under the ExpFile_cd control group. Instead, it appears in the Complete control group, which is usually located in the FSISYS.INI file.

```
< Complete >
    ExportOnComplete = Yes/No
```

By default, the check box is enabled and visible on the Complete window. The default is no (unchecked). If you set this option to Yes, the system checks the Export option on the Complete window.

Optionally, you can hide or disable the control so the user cannot change the setting. Hide the option by including *,hidden* after the answer, as shown below:

```
ExportOnComplete = Yes,hidden
```

To let the user see the option, but not change it, include *,disabled* after the answer:

```
ExportOnComplete = Yes,disabled
```

Exporting Recipient Information

Use this option to determine whether recipient information is exported. The default is No.

```
< ExpFile_CD >
    AFEExportRecips = Yes/No
```

If you set this option to Yes, recipients are written for each form listed inside angle brackets. Recipients are separated by commas and each recipient's copy count is shown in parentheses. This option lets you import recipient information that has been changed in a form set. For example:

```
\NA=\;SAMPCO;LB1;DEC PAGE;<AGENT(1),COMPANY(2),INSURED(1)>
```

If an image on a form has a recipient with a different copy count from the form, that image's differing recipients are written. Here is an example:

```
\NA=qmdc3\<COMPANY(1)>
```

There is no option for importing recipient information from the import file, since the information is either in the file or not.

Listing the Export Functions You want to Use

The system lets you specify a list of export functions you want the system to execute. You can use this capability for a variety of purposes. For instance, you can create a backup copy of an exported file in a second directory or you can run multiple exports using different export methods.

To install this feature, add the following INI option:

```
< ExportFormats >
    99 =;ME;Multi-Export;TRNW32->TRNMultiExport;
```

If you have other export functions listed under this control group and you want to remove those functions, insert a semicolon as the first character on the line to *comment out* that line. You can also delete the line from the INI file.

Keep in mind:

- The INI file is sorted when loaded. This lets you control which option appears first, second, third, and so on. For instance, you can use numbers as the option indicator left of the equal sign (01 =, 02 =, and so on) to set the order of the options.
  - The semicolons identify and separate each portion of the definition.
  - *ME* is a token that represents this export method. You can change the text, but make sure it is unique.
- The text *Multi-Export* appears in the Export area on the Complete Form Set window. You can enter any description you want, but space is limited in the export list display area. Make sure the text you enter fits.
- The final portion of the definition

  ```
  TRNW32->TRNMultiExport
  ```

  must be copied exactly as shown. This identifies the DLL to load and the function that performs this feature. This information is case sensitive.

Calling Export Operations

This export feature does not export data. Instead, it lets you call multiple export operations during a single event. For instance, you can use this feature to call the standard V2 export multiple times or in combination with other export routines.

Use the MultiExportList control group to list the export operations you want to call:

```
< MultiExportList >
    01 = TRNW32->TRNExportV2
    02 = TRNW32->TRNExportV2;EXPORT2;EXPFILE_CD;
    03 = TRNW32->TRNExportDS
```

The options in this control group are sorted by the value to the left of the equal sign.

Each option must list the export DLL and function name you want to execute. The export functions you list here do not have to be defined in the ExportFormats control group shown earlier. The method of defining this information is:

```
DLL->FunctionName
```

Notice the example above actually names the export function, *TRNExportV2*, twice in the list. Depending upon how you have configured your INI options for each export function, this may or may not result in a separate export file.

If you want to temporarily change the export options used by the called function, you can include two optional parameters, separated by semicolons, after the export function name. You can see an example of this in this option line:

```
02 = TRNW32->TRNExportV2;EXPORT2;EXPFILE_CD;
```

Each parameter names a control group that contains the options you want to temporarily use.

The first parameter names a *source* control group that contains alternate INI options you want to apply to the *destination* control group in the second parameter. The destination control group is the control group normally associated with the export operation you want to call.

The above example tells the system to copy the INI options found in the Export2 control group to the ExpFile_CD control group. The standard V2 export function, TRNExportV2, uses the ExpFile_CD control group to define its options.

During the first export, the options remain as is. During the second export operation, the system first copies the alternate options from the source control group into the destination control group used by the export function.

This option switching process replaces any INI options which have the same name and adds missing INI options to the destination group. Once the export function completes, the original INI values from the destination group are restored. This makes sure your default INI options are always left intact once the export operations are completed.

NOTE: Any INI options that exist in the destination group and are not replaced by an option in the source group are left unchanged. This means that you only have to include in the source control group those options you want to alter for the subsequent export operation.

Keep in mind

- The individual export functions listed under the MultiExportList control group do not know they are being called indirectly or in series. Each operates normally unless you override their INI options by substituting INI options as discussed previously. Normal operation can include displaying a window to ask for an export file name or for permission to overwrite an existing file.

  So, if you want your exports to work without user intervention, check the INI options for each export and use those that answer or suppress questions which have to be answered for the export to operate.

- Each export operation can generate error messages. The MultiExport feature calls all export functions in the list regardless of any errors which occur inside the subordinate export functions. If one of the export operations fails, this does not necessarily prevent the other operations from being called.
- If an export function named in an option in the MultiExportList control group cannot be located, an error message appears and that function is skipped.
- If you define a *source* control group without naming a *destination* control group, that will cause an error and the export function will be skipped.
- Do not include the multi-export function in the list declared in the MultiExportList control group.
- Do not name the MultiExportList control group as a destination control group on any of the listed export functions.
- If the source and destination control group are the same, the export is called without doing any substitutions.
- If you specify a destination control group without specifying a source control group, the export function is called and any substitutions are omitted.

Export Functions and INI Control Groups

This table lists the various export functions and the corresponding control groups which contain the options that apply to these functions.

Function	Control group	Description
TRNW32–>TRNExportV2	ExpFile_CD	The standard V2 (version 2) export.
TRNW32–>TRNExportDS	ImpExpCombined	The full-document export or combined-WIP export.
TRNW32–>TRNExportFields	ExportFields	The field data export used to limit exporting to certain fields and WIP record information.
WXMW32–>WXMExportXML	XML_Imp_Exp	The XML document export function. This export is included with Documaker licenses, but sold separately with other products, such as PPS.

Using the Field-only Export Option

The standard export feature outputs all form and field information contained in a form set. In some cases, you may want to extract only certain field information (like accounting data) from the form set without having to deal with all of the field and header information included in the standard export. You can use the Field-only export option to limit the exported output to the fields you choose.

By installing and using the Field-only export option, you can create an export file which contains only the information you want. After you install this export method, the Field-only export option appears on the Complete Formset window, along with the other export options.

This export method lets you specify which fields should be written to the export file. Each field is located by name, regardless of which image contains the information. Rather than output the same field and data numerous times, this export option only writes the first occurrence of any specified field in the form set.

In addition, if you later plan to import this information using another program, you can define an *alias* name which can be written to the export file. This means that you are not limited to using the field names chosen by the form designers.

The field-only export option uses many of the same INI options specified for a standard export method. Detail information about all options are included in this topic.

Installing the Field-only Export Option

To turn on this feature, first locate the ExportFormats control group. This control group defines the export methods available to the user on the Complete Formset window. It should look something like this:

```
< ExportFormats >
    01 = ;TD;Standard Export;TRNW32->TRNExportV2;
```

Then follow these steps:

1. To add this new export option, edit the file and include the following line under the ExportFormats control group:

   ```
   02 = ;FX;Field-only Export;TRNW32->TRNExportFields;
   ```

   There are several important things to note about this new line.

   - The INI file is sorted when loaded. Therefore, you can control which option appears first, second, third, and so on. This is accomplished in this example by using numbers as the option indicator -- 01 comes before 02. If you wish the new option to be first, you can simply change the numbers used for each option.
   - The semicolons are necessary to identify and separate each portion of the definition.
   - The first bit of information, FX, is a token that represents this export method.
   - The option text that appears in the Export area on the Complete Formset window occurs next. This text can be any text description. Please note, there is limited space in the export list to display the text description, so test to make sure the entire description can be seen on the window.

- The final portion of the definition must be copied exactly as shown. This identifies the DLL to load and the function that performs this specific feature. This information is case sensitive, so you should spell it exactly as defined here—otherwise the option will fail to load when chosen.

2 Next, one of the INI files will need to be changed. The FSIUSER.INI file typically contains *individual* user options, while the FSISYS.INI file contains the *universal* options shared by all users. The file you elect to change should be based upon whether you intend for all users or a select few to use the field-only export option.

NOTE: Control group options in the FSIUSER.INI file work with options in the FSISYS.INI file and in some cases, override those settings. This means that if you define the same control group in both INI files, they combine to form one control group when loaded.

3 In the INI file you chose, add the ExportFields control group. For the settings that can be specified under this control group, see Setting Up the Field-only Export Option on page 253.

SETTING UP THE FIELD-ONLY EXPORT OPTION

This export feature supports many of the same INI options that can be specified for the standard export option. The main difference is that the control group name for this feature is *ExportFields* instead of *ExpFile_CD,* which is used for the standard export feature.

Specifying a default file name

```
< ExportFields >
    File = file name
```

There is no default for this option. If you omit this option, the user must specify a file name for export. If you enter a valid file name for this option, the File window used for selecting the export file name defaults to this value. Optionally, you can suppress the File window if a valid file name is specified.

Specifying a default path

```
< ExportFields >
    Path = path name
```

There is no default for this option and without it, the current working directory is assumed. Setting the Path option tells the system to automatically write the export files to this directory, or the initial directory presented in the File window.

NOTE: If you enter a file name *and* full path for the File option, that path overrides your entry for this option.

Specifying a default file extension

```
< ExportFields >
    Ext = .exp
```

The default for this option is *.EXP*. To use another extension, simply specify it using this option.

NOTE: If you enter a file name *and* an extension for the File option, that extension overrides your entry for this option.

Appending to an existing export file

```
< ExportFields >
    AppendedExport = Yes
```

The default setting is No. If you turn this option on and an existing file is selected (or specified by an INI option), no overwrite confirmation message is shown to the user. The current form set data is appended to the existing file.

Suppressing the overwrite file confirmation message

```
< ExportFields >
    Overwrite = Yes
```

The default is No. Enabling this option prevents the overwrite confirmation message from appearing if the user chooses an existing file as the export file.

This option is ignored if you activate the AppendedExport option. Appending to an export file implicitly means it will not be overwritten.

Suppress the Export File window

```
< ExportFields >
    SuppressDlg = Yes
```

The default is No, which means the window will appear. If an INI option has not been defined, or defined improperly for the default export file name, the window appears despite the setting of this option.

The File Selection window is also affected by the AppendedExport and Overwrite options. If either options is enabled (Yes) the SuppressDlg option prevents the File Selection window from appearing. If neither of these options are enabled, the window is only suppressed if the specified export file does not exist. This prevents users from accidentally destroying a potentially valid export file without confirmation.

Identifying the fields to export

```
< ExportFields >
    FIELD = field name ; alias
    FIELD = field name2 ; alias2
    ...
```

For each field that should be exported from the form set you must include a *FIELD=* line in the INI file. You may export as many fields as necessary, but each line must begin with the *FIELD=* statement.

The *field name* reference in the example should be replaced with an actual variable field name contained on a image within the form set. To determine the field names, it may be necessary to load the image in Documaker Studio or Image Editor and check the properties of the given fields.

The alias reference is optional and should be separated by a semicolon from the image field name, if used. This identifies a different name to write in the export file, rather than the name of the actual field from the image. Use the alias feature when you are exporting information which will be imported into another application that uses different names for its fields.

For instance, assume the line reads as follows:

```
FIELD = Acct#;Account Number
```

This line would locate the Acct# field in the form set. If found, the data would be written to the export file using the name *Account Number*.

Other important notes about these settings:

- The order of the fields in the export file matches the order they are defined within the INI control group.
- If an alias is not provided, the exported field name will be the same as the image field name.
- Including a semicolon after the image field name, but not providing any text for the alias suppresses the output of a field name in the export file. The backslash, (\), will still precede the data for the field, even when the field name is suppressed.
- Any field that is not located within the form set is omitted in the export file.
- The field data written to the export file appears just as it does in the form set. Fields without data are written to the export file with no data following the backslash.
- When specifying an alias, spaces before and after the semicolon divider are ignored. This means that *IMGFLD;EXPFLD* is the same as *IMGFLD ; EXPFLD*.
- Image field names are not case sensitive. *IMGFLD* is equivalent to *imgfld* and will locate the same field within the form set. However, the case of the field name or alias in the INI file is preserved in the output file.

Defining header and trailer information

```
< ExportFields >
   START = text
   END   = text
```

These settings are optional and independent of each other. You can specify a START without an END or vice versa. No default is provided for the START and END options.

Since this export method does not include header information from the referenced form set, it may be necessary to identify the starting and ending locations of export information. This is probably most important when you are using the AppendedExport option, which means that more than one form set's information will be written to the same file.

The text specified on these options can be any ASCII string. The text is written exactly as it is specified in the INI file. START is written before the first field's data from the form set is written. END is written after all field data has been written.

The START and END text is written to the export file whether any field data is written from the given form set.

Exporting to a single line

```
< ExportFields >
   SingleLine = Yes (or No)
```

This options defaults to No and is not enabled unless you change it in the INI file.

Normally, each field's name and data is written to a different text line within the export file. By enabling the SingleLine option, this will condense the exported field information into a single line of output. This option is provided for those environments where this method of export is more desirable.

By default, when the SingleLine option is in use, each field name and data set is separated by a semicolon. You can change this field separator using the Separator option.

```
< ExportFields >
   Separator = text
```

This additional option is only recognized when you specify the SingleLine option. The text assigned to the statement is used to separate each field name and data set written to the export file.

The text can be any ASCII string value you want to specify. For example...

*Separator = **!***

tells the system to write the text

!

between each field. Here's another example...

Separator = NEXT FIELD=

tells the system to write the text

NEXT FIELD=

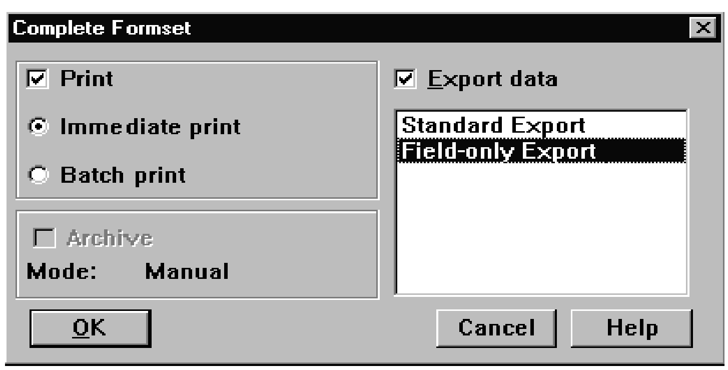
between each field.

Other important notes about these options:

- The separator text is only written when the SingleLine option is set to Yes.
- The separator text is only written between the field information sets. If you use a START option, the system omits the separator between the START text and the first field. Likewise, if you specify an END option, the system omits the separator between the last field output and the trailing text.
- The SingleLine option is probably of most value when using the AppendedExport option and writing more than one form set to the same export file.

Using the Field-only Export Option

Once installed, the option appears as a choice in the Export list, as shown below.



Choosing this option tells the system to write to the export file those fields specified in the ExportFields control group.

By default, the field information is written to the export file in the format shown below:

```
FIELD NAME\FIELD DATA
```

The field's name is separated from its associated data via the backslash (\) character. Using some of the other options provided with this feature, it is possible to change the output in several ways. For information about additional format options, see Setting Up the Field-only Export Option on page 253.

Setting Up Multiple Import Sessions

There may be occasions when users need to import data from more than one directory. For example, you may have one directory set up for form sets with a status code of *WIP* and another directory for form sets with a status code of *BatchPrint*.

To handle this requirement, you can define more than one set of options for auto-import in the INI file. The following examples show how to change the default AUTOIMPORT settings to accommodate this:

```
< TimerFuncs >
    01=;0;0;300;TRNW32->TRNAutoImport;
    02=;0;0;300;TRNW32->TRNAutoImport; \AUTOBATCH
```

In this example, definition *01=* assumes AutoImport is the control group that identifies the behavior for this auto-import registration. The second definition, *02*, names an alternate control group, called AutoBatch, which contains the definitions for that registration.

Each auto-import control group, such as AUTOIMPORT and AUTOBATCH in this example, can contain different auto-import options.

Notice that the same auto-import function is used for all definitions. You can set the TMRLIB flags (STATE, URGENCY, and SECONDS) as needed for each control group.

Once the system activates TRNAutoImport, it identifies the specific auto-import control group and options before it searches for import files.

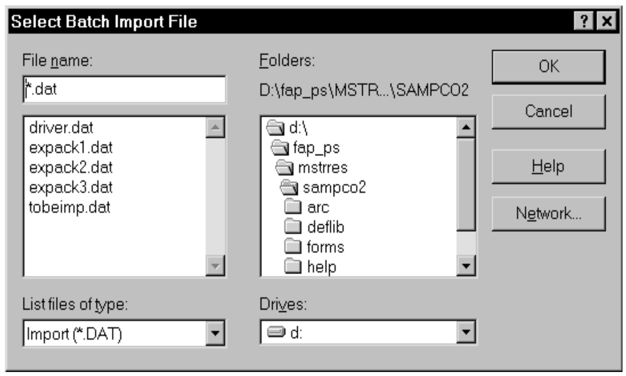
For more information about timed service functions, see Setting Up Timed Service Functions on page 162.

Batch Importing from a File

Batch importing from a file lets you import multiple form sets from a single file and place the individual form sets in the WIP list. When you batch import from a file, the system imports both the forms and the data. Users can then open the form sets from the WIP list and complete data entry. See Importing and Exporting Files on page 240 for more information on setting up import and export files.

Follow these steps to perform a batch import:

1. Choose Tools, Batch Import from File. The Select Batch Import File window appears.

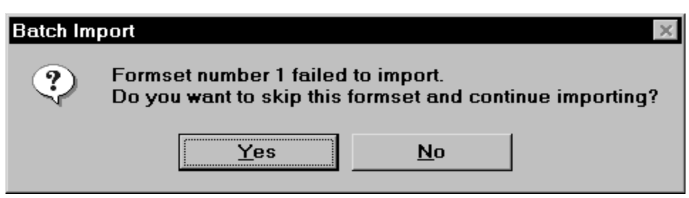


2. Enter or select the batch import file. You can change the drive and directory if necessary. Click Ok.

Note: The system defaults to the working directory. You can change the drive and directory to locate a batch import file, but if you try to import form sets with key fields that do not match the current resource directory, the import will not be successful.

3. The system displays a message confirming the number of successful form set imports. Click Ok to close the message.

4. If there are data or key field discrepancies in any of the form sets, the system displays a warning message either telling you the form set number is not unique or stating there were no successful imports. If the problem is in a single form set, the system displays this message:



5. Click Yes to continue importing the form sets, or No to cancel the import. If you choose Yes, the system will place the form sets in the WIP list and display a message confirming the number of successful imports.

6. You can choose WIP, WIP List to view the form sets in the WIP list:

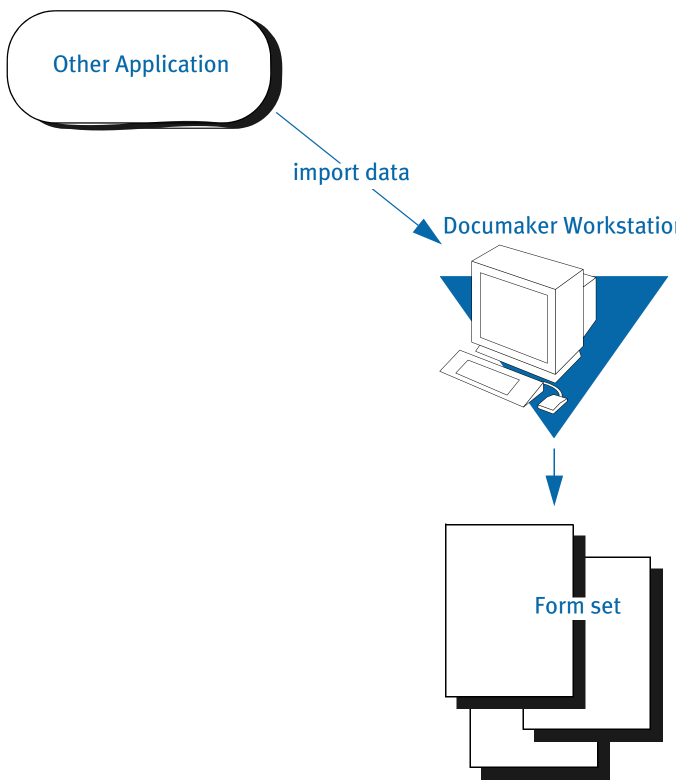
Creating a Standard Import File

Creating import and export files lets you automatically import data from an external application or data created during entry, into specific form sets in the system. You can also export data from the system to use in other applications or to import back into the system. When you create an import file, you must format the import data and set up the file so that the system knows which form set to apply the data to.

In addition to standard import files, you can also create a selective import driver file which displays applicable import files for user selection, based on the company, transaction, and user ID.

After you and the users have selected the forms for data import, your next task is to extract the applicable data from the host application system.

For a standard import file, you extract data to apply to a single form set. The type of data you import normally includes common data that does not vary. Common data can include company, line of business, policy, recipient, agent, lien holder, policy dates, and other general information. In addition, is it possible for the import file to designate which forms should be included in the form set. When importing data from a standard import file, the user must know what the import file contains to import it to the appropriate form set.



After you extract the host data, you must format it so the system can recognize and import the data to the correct forms, and to the correct fields on the forms.

You can format the text using any text editor. You may find it helpful to first map the data on paper, using an actual form as a guide, then enter the formats into the text editor.

NOTE: When creating files for batch import, follow the instructions for creating standard import and export files (with a header). For more information on configuring the system to append form set data to an existing export file, see Importing and Exporting Files on page 240.

For information on batch importing, see Batch Importing from a File on page 259. For information on exporting recipient information, see Exporting Recipient Information on page 249.

Sample Standard Import File

An import file is an ASCII text file which contains form set information. The file is formatted into lines with each line terminated by a carriage return/line feed (\r\n) combination. Each line is interpreted as one of these basic components:

- Header Record lines
- Form/image lines
- Field Definition lines

Some import files will only contain *header record* lines and f*ield definition* lines. Files may contain form/image lines if specific form information is to be included.

One header record is normally required, but any number of header records may be included. If more than one header record is included, they should all include the same KEY1 definition. Group all header records at the top of the file, because the system will not recognize additional headers once it encounters field or form information.

Header records consist of elements separated by semicolons (:). In addition, the line must start with a semicolon. The components of a header record are shown here:

```
;KEY1;KEY2;KEYID;TRANSCODE;STATUSCODE;DESCRIPTION
```

The first three components are required while the last three are optional. Note that all components are *positional*. This means that should a component be omitted, include the semicolon following the component. For instance, all of the following are valid header representations.

```
;KEY1;KEY2;KEYID;
;KEY1;KEY2;KEYID;TRANSCODE;
;KEY1;KEY2;KEYID;TRANSCODE;STATUSCODE;
;KEY1;KEY2;KEYID;;;DESCRIPTION
```

Component	Description
KEY1	a valid Key1 (Company) defined in the FORM.DAT file.
KEY2	a valid Key2 (Line of Business) defined in the FORM.DAT for that KEY1 (Company).
KEYID	the WIP KeyID (such as a policy number) provided on the Form Selection window.
TRANSCODE	corresponds to a valid transaction type defined by the user in INI files.

Component	Description
STATUSCODE	a valid WIP status identifier which is also defined in the INI files.
DESCRIPTION	text to be assigned as the WIP description.

Form/image lines define which forms and images on those forms to preselect for the user. Field information which is read after reading a form line is assumed to refer to fields on that form/image. Form lines have the following syntax:

```
\NA=name\;KEY1;KEY2;FORMNAME;
```

The first backslash (\) indicates this is a form/image line and must be followed by *NA=*.

KEY1 and KEY2 represent similar values to that shown for header records. Along with the FORMNAME (form's name), the system uses KEY1 and KEY2 to locate the specific FORM.DAT line that represents that form.

If name is included, it defines the image to locate on the current form. This component may be omitted to represent the form named for selection.

```
\NA=\;KEY1;KEY2;FORMNAME;
```

In addition, if the following line is included, this simply defines the next image on the previous form to locate.

```
\NA=name\
```

Field Definition lines refer to any line that does not begin with a semicolon or backslash. They take the following form:

```
FIELDNAME\FIELDDATA
```

Where FIELDNAME represents a field that is defined globally (if no current image or form has been named), or belongs to the last form/image that was named. The backslash (\ ) separates the field's name from the field's data.

FIELDDATA is any valid field (already formatted) for use by the named field.

```
POLICY CNT\31
POLICY EFFECT\04/25/06
POLICY EXPIR\04/25/07
```

In the sample data records above, each line is a field name, followed by field data:

- Policy Count = 31
- Policy Effective Date = 4/25/06
- Policy Expiration Date = 4/25/07

Field	Length	Description
Variable Field Name	32	Field name as it appears on the applicable form. Field names cannot contain a backslash (\).
Variable Field Data	255	Data corresponding to the field name. Field data cannot contain a backslash (\). Make field data lengths equal to or less than field lengths on the form. If data is greater than the form field lengths, the system truncates the data.

NOTE: If multiple fields with the same name are defined, the data in the last field with that name will import to all the similarly named fields in your form/image.

This example shows a formatted import file:

Header record line 1	;GENERAL AGTS INS;COMML PACKAGE;CPP_94;NB;WIP;Sample;
	;GENERAL AGTS INS;GENERAL;GEN_94;
	INSURED NAME\Bob Bradshaw
Header record line 2	POLICY NBR\332-224521-NR
Form/image line 1	\NA=IMG1\;GENERAL AGTS INS;COMML PACKAGE;FORM1;
	POLICY CNT\31
	EFFECT DATE\04/25/06
Field data lines	EXPIR DATE\04/25/07
	\NA=IMG2\;GENERAL AGTS INS;COMML PACKAGE;FORM2;
	POLICY CAN DATE\9/11/03
	POLICY END DATE\9/11/04
Form/image line 2	POLICY AUD DATE\9/11/06
Form/image line 3	\NA=IMG2B\
	\NA=IMG2B\
	LOB1\AUTOMOBILE
	PREMIUM 1\1111.1
Form/image line 4	POLICY FEE\44.4
	\NA=IMG3\;GENERAL AGTS INS;GENERAL;FORM3;AGENT NUM\77
	AGENT NAME\JOE SMITH
Form/image line 5	AGENT ADDRS1\1234 CANTERBURY LANE
	\NA=IMG3\;GENERAL AGTS INS;GENERAL;FORM3;
	AGENT CITY/ST\ANYWHERE/USA
	AGENT PHONE\5551234
Form/image line 6	LOSS PAYEE\THE BANK, INC.

In this example, the form set contains two KEY2 groups (COMML PACKAGE and GENERAL) for the same KEY1. This normally indicates a packaged policy situation.

Field data defined before any form/image lines represents globally defined fields. Any form in the form set which has one of these globally defined fields will be prefilled with the imported data.

FORM1 and FORM2, are selected from the COMML PACKAGE group. Two copies of FORM3 are selected from the GENERAL group.

FORM2 identifies three images, IMG2, IMG2B, and IMG2B. The last two images are named the same, but no data is imported for the first occurrence of this image.

Rules for import files

Here are the rules for the import files:

- If no current form/image line has been identified, assume the fields are *form set global*.
- If a form is named and an image is not, assume fields have a *form* scope and will propagate to any identically named fields on that form.
- The first form/image line found must have the additional KEY1, KEY2, and FORMNAME fields. The program takes this information and searches the FORM.DAT for the appropriate form. Until another form is named, the system assumes the data following this line belongs to the named form/image.
- All forms identified by the import file are automatically selected for the user. This causes the form to appear with a check mark on the Form Selection window.
- Forms can be identified without additional image or field data.

Listing the global fields

You can use the OutputFieldScope option to list the global scope fields near the beginning of the FAP file:

```
< ExpFile_CD >
   OutputFieldScope = Yes
```

Here is an example:

```
;FORMMAKER PACKAGE;COMMERCIAL PACKAGE;EXPSCOPE=YES;NB;W  ;;
;FORMMAKER PACKAGE;GENERAL LIABILITY;EXPSCOPE=YES;NB;W  ;;
RENEWAL NBR\1098212-12
POLICY NBR\1098212
INSURED NAME\Fred Scope
ADDR1\1120 Global Scope Drive
ADDR2\Suite 3020
CITY\Atlanta
EFFECT DATE\02/17/01
EXPIRE DATE\02/17/06
BUSINESS DESCRP\Car Insurance
COMM PROP PREM\      1000.00
\NA=\;FORMMAKER PACKAGE;COMMERCIAL PACKAGE;FIL 1010 04 92;
\NA=\;FORMMAKER PACKAGE;GENERAL LIABILITY;CG DEC;
FORM LINE1\FCG 0001 04 93, FCG 0010 11 92, FCG 2100 01 93, FCD 0000
04 93
\NA=CGDEC~1\
\NA=\;FORMMAKER PACKAGE;GENERAL LIABILITY;FCG 0001 04 93;
\NA=CLF00\
\NA=CLF00~2\
\NA=\;FORMMAKER PACKAGE;GENERAL LIABILITY;FCG 0010 11 92;
\NA=\;FORMMAKER PACKAGE;GENERAL LIABILITY;FCG 2100 01 93;
\NA=\;FORMMAKER PACKAGE;GENERAL LIABILITY;FCD 0000 04 93;
\NA=CL000\
```

Global fields

Keep in mind this option only applies to standard V2 imports and that the system only lists the fields once per form set.

TESTING A STANDARD IMPORT FUNCTION

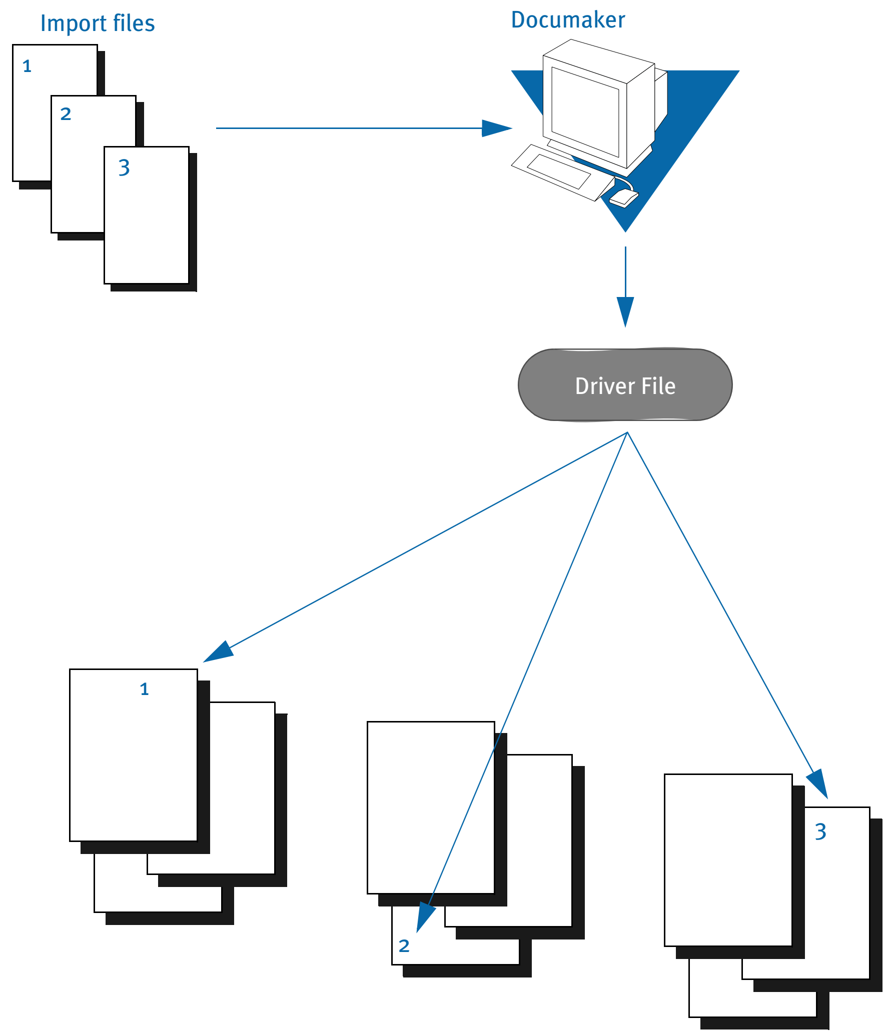
To make sure the data imports successfully, test the import files you create. Before you perform the test, first check the FSIUSER.INI and FSISYS.INI files to make sure import (and export) functions are enabled in the system. Also make sure at least one user ID is loaded into the system for import function initiation.

To test a standard import function, follow the steps for selecting forms, in Chapter 4 of the Documaker Workstation User Guide. If you created multiple import files, make sure you test them all.

Creating a Selective Import File

A selective import function uses a driver file that filters import data. When performing a selective import function, the user selects the driver file from the list of import files. The driver file recognizes the selected transaction type, the company name, and the user's access level, and displays only the import files that apply to that combination of variables.

If your company uses numerous import files, the user can narrow the file choices, to display only one or two that are appropriate for the selected form set type. Choosing a selective import function takes the guesswork out of identifying the correct file to import. The selective import function supports multiple users who access the driver file simultaneously.



Sample Selective Driver File

The file format for selective import data is exactly the same as for standard import data. The import driver file format (fields) differs slightly. Like the standard import file, a driver file contains ASCII characters with semicolon delimited fields, and carriage return/line feed (\r\n) record delimiters.

Each record line in the driver file normally contains five fields: Resource Library (Company) Name, Policy Transaction Name, Import File Name (including the path), User ID, and File Description. The company library name entered should match the name of the resource library where you store the file.

These five fields indicate the form set that receives the import data. The file description indicates the form set that receives the import data, to ensure importing into the correct form set. Each two-line record indicates the particular form set to which you import related data. You cannot import form sets to different resource libraries. Here is a sample driver file.

```
;SAMPCO;NEWFMSET;\PPSWIN\MSTRRES\SAMPCO\
IMPORT3.DAT;DOCUCORP;General Information

;SAMPCO;ENDORSE;\PPSWIN\MSTRRES\SAMPCO\
IMPORT4.DAT;USER1;Health endorsement
```

NOTE: In the actual driver file, each record appears on a single line.

Formatting the Driver Import File

The table below contains syntax and format guidelines for the import driver file. Refer to this table as you create the driver records. You can assign any name to the driver file, as long as the users know the file name. Create the driver file as you would create a standard import file, using a text editor.

Field	Length	Description
Company	8	Company library name of the DOS batch file (no extension). This batch file calls a specific resource library directory (ppswin95\dll).
Transaction	8	Transaction type, enter: NEWFMSET - New form set ENDORSE - Endorsement RENEWAL - Renewal
Import File	127	Import file name, include the path and subdirectories
User ID	64	User ID for individual who completes the transaction
Description	35	Description of the import file Driver file parameters, appears in the Import Selection window

Testing a Selective Import

The requirements for testing a selective import function are the same as for a standard import function. Confirm that the import function is enabled in the FSIUSER.INI file for each company library you designated in the import driver file.

To test a selective import function, follow the steps for selecting forms, in Chapter 3 of the Documaker Workstation User Guide.

When a user selects the driver file from the Import File window, the driver file checks for the selected transaction type and user ID. The driver file then checks the transaction types you defined in each driver record for a match with the user selected transaction. By process of elimination, the driver file then displays only the import files applicable to the user's selected transaction type, and to the user's ID level.

Importing Information Directly into Archive

The system lets you import data directly into archive using a timed service function that automatically archives all records waiting in the Manual Archive queue.

This function performs the manual steps of choosing the WIP, Manual Archive option, selecting all records on the window, and then clicking Ok.

Use an INI setting similar to the following to set up this feature:

```
< TimerFuncs >
    08 =;0;0;100;AFEW32->AFEAutoArchive;
```

Where the first three values (*;0;0;100;*) represent:

;*State*;*Urgency*;*Seconds/Time-of-day*;*Function*

Value	Description
State	always set to zero (0)
Urgency	zero (0) or 1, where zero means the call can be skipped if the user is engaged in an operation and 1 means to call the feature as soon as the user has finished the current operation.
Seconds or Time-of-day	Enter the number of seconds to call the function at a specific interval, such as **300** for every five minutes. Or, enter the specific time of the day at which you want the system to call this function. The system uses a 24-hour clock, so enter **13:00** to indicate 1 pm.
Function	*AFEW32->AFEAutoArchive* indicates the timed service function. Enter this value exactly as shown. Without this value the automatic archive feature will never work. An error message will be displayed if this value is incorrect.

If you use this feature with the Automatic Import (to import records with a Manual Archive status), you can have the system archive imported files without requiring user intervention.

For more information on other timed service functions, see Setting Up Timed Service Functions on page 162.

Importing Global Data from Archive

This feature lets you select a form set from the Archive window and, instead of retrieving the entire original form set as it is done with Archive Retrieval, only import the global data from that form set. This data (without the forms) is then available for new form sets.

Using this feature, you do not have to keep the forms stored in the archive form set to make sure the data propagates onto the newer forms.

You set up the Import from Archive feature using INI files. The description of the Retrieve from Archive option in the Import list is based on your entries in the INI options.

To install the Import from Archive feature, add these INI options:

```
< ImportFormats >
    02 =;IA;Import From Archive;TRNW32->TRNImpDatFromArchive;
```

You can change the text that appears in the Select Import window as necessary.

NOTE: This feature does not select forms for you; it simply imports the data designated as global. You use the Scope field in the field's Properties window in Documaker Studio or Image Editor to designate a field as global.

Importing Data with Forms and Images

If the form and image names match, the system can transfer data to the Entry system when importing forms and images from archive.

For instance, if the archived form set consists of *FORM_A* and *FORM_B* and you currently have *FORM_A* and *FORM_B_v2* selected, the system transfers the form data from *FORM_A*, plus any global fields set up for the entire form set. Since *FORM_B* does not match *FORM_B_v2*, no form or image level information is transferred between the form sets. Similar name matching is required at the image levels.

To use this feature, you must include these INI options:

```
< ImportFromArchive >
  SelectForms    = Yes
  TransferRecord = Yes
```

The default for both options is No, to be compatible with the original release of the ImportFromArchive feature discussed in the previous section (Importing Global Data from Archive on page 270).

If you set the SelectForms option to Yes, the import operation reselects the forms based upon those found in the archive. Any prior form select made by the user is removed.

If you set the TransferRecord option to Yes, the archive index record information is transferred to the WIP record information and the system updates the Form Selection window to show any changed values. For instance, you would set this option to Yes if you want the Key1, Key2, KeyID, and Description fields from the archived record used on your current form selection.

NOTE: Multi-line text fields will retain paragraph markers but will not retain any text formatting.

Exporting When Manually Archiving

You can have the system export a form set when it manually archives the form set. Using INI options, you can make the system wait until archive is about to occur before it exports the form set. This makes sure the exported data accurately represents the archived document.

The system checks the ArchiveExport option to determine which export routine you want to use. It then loads the form set into memory. This is necessary for the export function, but not for archival.

NOTE: Using the ArchiveExport method does not affect the export feature on the Complete window. This is a separate export from the one started there.

The export function is called before the archival functions are called. This way, if an error occurs during the export, the archival process stops and returns the error. The form set is still in WIP, so you can correct the error.

To use this feature, include the ArchiveExport option in the Complete control group, as shown in this example:

```
< Complete >
    ArchiveExport = DLL->FunctionName;SourceINI;DestINI;
```

The ArchiveExport option identifies the DLL and name of the export function you want to use. The export function does not have to be defined in the ExportFormats control group. Define the export function name the same way you define other entry hooks — a DLL name followed by the name of the name of a function to dynamically call to do the export. Here is an example:

```
< Complete >
    ArchiveExport = TRNW32->TRNExportV2;Export2;ExpFile_CD;
```

To handle different export options used when called in this manner, you can include two optional parameters after the export function name. Separate the parameters with semicolons. Each parameter names an INI control group that contain various options.

The first or source control group (*Export2* in the example) specifies the INI control group which contains alternative INI options you want to apply to the second INI control group (*ExpFile_CD* in the example). The second or destination control group should be the control group normally associated with the export function you want to use.

In the example above, the system copies the INI settings found under the Export2 control group to the ExpFile_CD control group. The standard V2 export function then uses the ExpFile_CD control group to find its options.

For instance, you might define the following:

```
< Export2 >
    Overwrite = Yes
    SuppressDlg = Yes
    File = ~HEXTIME .EXP
< ExpFile_CD]
    Overwrite = No
    File = Output.EXP
    Path = .\data\
```

NOTE: You only have to include in the source control group options you want to add or replace options in the destination control group.

Once the export function completes, the system restores the original INI options and values from the destination group. This makes sure your default INI options are left intact once the export operation has finished.

So, using the example above, after the substitution takes place you would have the following options defined for the standard export:

```
< ExpFile_CD >
    Overwrite   = Yes
    SuppressDlg= Yes
    File        = ~HEXTIME .EXP
    Path        = .\data\
```

By restoring the original INI options and values for the destination control group, the system lets manually called export functions work with a standard set of options, while allowing the ArchiveExport method to be more automated.

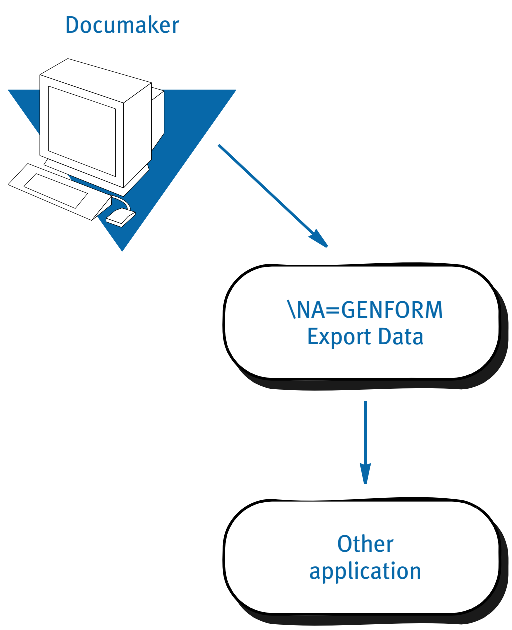
Keep in mind that the export function does not know it is being called indirectly. It will operate in its normal fashion. So, unless you override INI options as described here, the system may ask for an export file name or other information.

To further automate exports, check the INI options available for the export function and define those that answer or suppress questions for the export process so it can operate without user input.

Creating Export Files

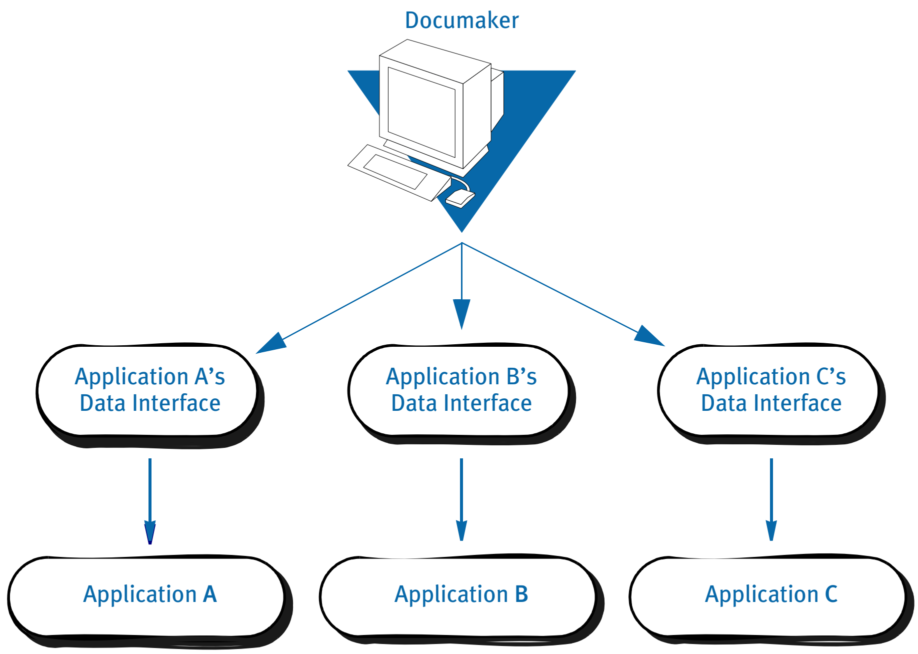
Creating an export file lets you extract form set data from the system and use the data in other applications or import the data back into the system. The export file contains the generically formatted system data that other applications read.

You can create two different kinds of export files: with headers or without headers. Depending on your system configuration, your export options may include Export Version 2. This export file is the same as the Standard Export (with a header). An export file with a header lets you indicate the form (or form set) where the data originated. An export file without a header does not include the form information. An export file without a header lets you export only the form data.



To create an export file, you can extract data that users have manually entered into the system, or data that was imported from another application.

The following illustration shows the flow of data when you create an export file without a header.



* The Application Data Interface is a program written by the application to read system export file data.

Creating an export file without a header lets you extract raw data from the system, without indicating the image where the data originated. With this option, users can use the exported data in various other applications.

Sample Export File Format (with Headers)

This example shows part of a sample export file with a header. The export file uses the exact format as a standard import file. Each record line contains the variable field name and the variable field data, separated by a backslash (\). The export file also contains the header as record line 1. If a form appears multiple times in a form set, the export file contains recurring variable field data for the duplicate forms.

Header record line 1

```
\NA=GENFORM\
INSURED NAME\John Smith
POLICY NBR\06871
```

Variable field data lines

```
PERSONAL LIMIT\300
EACH OCCR LIMIT\400
FIRE DAMG LIMIT\500
MED EXPEN LIMIT\600
```

Header record line 2

```
\NA=TESTFORM\
INSURED NAME\Richard Smith
POLICY NBR\28754
```

PERSONAL LIMIT\1000
EACH OCCR LIMIT\500
FIRE DAMG LIMIT\500
Variable field data lines
MED EXPEN LIMIT\2000

A form set may contain multiple forms. You can create an export file containing data corresponding to each form. The header records separate the data by image.

Below are two general guidelines to consider when creating export files, both with or without a header:

- If you create an export file corresponding to an import file, you can use the same name for both files, but use different extensions, or possibly use separate directories. For example...

```
(Library Name).IMP = Import Extension
(Library Name).OUT = Export Extension
```

- If you do not use an import file, the system prompts you to enter an export file name. If you use an import file, the system uses the file name of the import file also as the export file name.

Formatting the Export File

The following tables contain a section of the sample export file from the previous page, and the file format. Refer to these tables as you create the export data file. The headers separate the form data.

```
\NA=GENFORM\
INSURED NAME\John Smith
POLICY NBR\06871
PERSONAL LIMIT\300
\NA=TESTFORM\
INSURED NAME\Richard Smith
POLICY NBR\28754
PERSONAL LIMIT\1000
```

Field	Length	Description
Header	20	The name of the image in a form or form set where the export data originated. Header name is always in standard format: \NA= followed by the image name.
Variable Field Name	15	Field name as it appears on the applicable form. Same format as Standard Import file.
Variable Field Data	150	Data corresponding to the field name. Same format as Standard Import file.

Sample Export File Format (without Headers)

The format for an export file without a header is identical to an export file with a header, except the file does not include \NA records as headers. Each record line contains the variable field name, followed by the variable field data, separated by a backslash.

```
INSURED NAME\John Smith
POLICY NBR\06871
PERSONAL LIMIT\300
EACH OCCR LIMIT\400
FIRE DAMG LIMIT\500
MED EXP LIMIT\600
```

Field	Length	Description
Variable Field Name	15	Field name as it appears on the applicable form. Same format as export with Header file.
Variable Field Data	150	Data corresponding to the field name. Same format as export with Header file.

Testing an Export Function

Always perform a test export to make sure the export file you create contains the data you intended and the data is in the correct format. Test your application data interface to make sure your application system correctly imports the system data you exported. See the Documaker Workstation User Guide for instructions on importing and exporting.

Importing And Exporting WIP, NAFile, And POLFile Information

The Entry system includes an import/export format that combines WIP, NAFILE, and POLFILE information. This format exports information from the Entry system which will then be imported into the Rules Processor.

This feature uses many of the same INI options as standard import and shares the same options with a standard export file (export with headers). The import function conforms to standard Documaker Workstation import requirements. The input to this function is a single file which contains WIP record information appended with standard NA and POL form set data. For more information on formatting import/export files, see Setting Up Import Formats on page 242 and Setting Up Export Formats on page 243.

To use this new format type, you must add the following options to your INI file:

```
< ImportFormats >
    02 = ;DS;Full Document Import;TRNW32->TRNImportDS;
< ExportFormats >
    04 = ;DSFull Document Export;TRNW32->TRNExportDS;
```

You can also add these options in the ImpExpCombined control group:

```
< ImpExpCombined >
    File        =
    Path        =
    Ext         =
    SuppressDlg =
    WIPHeader   =
    Separator   =
    Field       =
```

Option	Description
File	(optional) Enter the name of the import or export file.
Path	(optional) Enter the path for the import or export file if not included in the File option.
Ext	(optional) Enter the default extension of the file. Defaults to *DS*.
SuppressDlg	(optional) Set to Yes to prevent the File Selection window from appearing if enough information is specified in the INI file. The default is No.
WIPHeader	(optional) Enter a text string to indicate where WIP record transactions begin. The default is *WIP*.
Separator	(optional) Enter a text string to indicate an alternative separator for WIP fields. The default separator is a quotation mark (“).You could use another character as the separator, such as a semicolon (;).

Field	For each field that should be imported/exported to the WIP record line, you must include a line in the INI file. You can import as many fields as necessary. Begin each line with *FIELD=* and use this syntax:
	`Field = The WIP field name;stringformat`
	The string format is the standard C sprintf function format. Here are some examples:
	`Field =KEY1;%-3.3s`
	`Field =KEYID;%-10.10s`

For each field that should be exported to the WIP record line, you must include a line in the INI file. You may export as many fields as necessary. Begin each line with *FIELD=* and use this syntax:

```
FIELD=WIP field name;formatstring
```

Here are some examples:

```
FIELD=KEY1;%-3.3s
FIELD= ;,
FIELD=KEYID;%-10.10s
```

Mapping Alternate WIP Index Columns from Imported WIP Data

You can add INI options to the INI control group associated with the import method you are using to identify globally-defined fields from which you want data transferred to the WIP index record.

Here is an example of how you would define one of these INI options:

```
WIPField = FAPField;DFDField
```

Option	Description
WIPField	FAPField is the name of a field stored in the global form set dictionary. The FAPField definition must name a field that is defined as Formset Global (globally) in the import file DFDField is the name of a WIP DFD field.

By default, the system looks for the Key1, Key2, and KeyID fields using their standard WIP names. After that, any additional WIPField definitions are located and mapped accordingly.

Here is an example:

```
< ImpFile_CD>
    WIPField = INSURED NAME;DESC
```

This definition looks for the globally-defined document field INSURED NAME and maps it into the WIP record DESC column. Since the DESC column is normally one of the WIP columns that would be automatically mapped in the V2 header, this definition overrides that and stores the value of the global field INSURED NAME in its place.

Exporting Files Created by AutoImport, AutoPrint, or AutoArchive

When using AutoImport, AutoPrint, and AutoArchive, you can have the system automatically export the resulting file to another system. To use this feature, add the following option to your FSIUSER.INI or FSISYS.INI file:

```
< AutoImport >
    CompleteOnSuccess = Yes
```

NOTE: If you specify an alternate name for your AutoImport control group, add this new option under that group name.

The default for this option is No, which tells the system not to automatically export the file once the transactions have been processed.

Use these Complete control group options to automate the window in version 10.0.

```
< Complete >
    PrintOnComplete = Yes
    ExportOnComplete = Yes
    ArchiveOnComplete = Yes
    SuppressDialog = Yes
```

Set up the ExpFile_CD control group up to automatically complete.

```
< ExpFile_CD >
    File = EXPORT
    EXT = .OUT
    SuppressDlg = Yes
    AppendedExport = Yes
```

NOTE: When operating in this manner, the first export function in your list is the one the system uses. Make sure you only have one export format defined, such as the ExportFormats control group, or define the default option so it appears at the top of the list.

Finally, go to your Printer groups and make sure these options are set:

```
< Printer >
    SuppressDialog = Yes
    PrtType = PCL
```

NOTE: If you are using the GDI output or redirecting raw PCL through the GDI device, make sure you set the necessary options in that PrtType control group to automatically complete windows that appear.

Sending messages to a log file

Use the SuppressAllMessages option to send most error messages to a log file instead of having them displayed on screen. Keep in mind that unless you periodically check the log file for errors, you will not know errors are occurring. Here is an example:

```
< AutoImport >
    SuppressAllMessages = Yes
```

Working With XML Files

You can now import and export XML files while using Documaker Workstation and you can send and receive XML messages. Setting up the new import and export capabilities is similar to setting up any import/export file format.

NOTE: The ability to work with XML files is included in Documaker Workstation, but must be purchased separately by PPS users. You must also have a Docupresentment license to use the messaging features in the WXMEntryHookExtXMLLoad function because it calls Docupresentment files. Contact your sales representative for more information.

This feature uses these functions:

Function	Description
WXMImportXML	This function lets you import data from an XML file into a form set.
WXMExportXML	This function lets you export data from a form set to an XML file.
WXMEntryHookExtXMLLoad	This function lets you send messages from the system to any type of message server.

Setting Up Documaker Workstation

To use the import and export functions, you must also add this control group and options to your FSISYS.INI or FSIUSER.INI file:

```
< XML_IMP_EXP >
    Ext         =
    File        =
    Path        =
    SuppressDlg=
```

Option	Description
Ext	(optional) Enter the extension for the output files. The default is XML.)
File	(optional) Enter a file name, such as XMLEXP. If you omit this option the system prompts the user to enter the file name.
Path	(optional) Enter the path, such as \xmlfile. If you omit this option, the system defaults to the current directory.
SuppressDlg	(optional) Enter Yes to suppress the File Selection window. The default is No.

Follow the instructions below to complete the import, export, and messaging setup.

Setting up the XML export format

Follow these steps to set up the XML export format:

1. Open the FSISYS.INI file in the resource library for which you want to use export files. You can use any text editor to open this file.
2. Locate the ExportFormats control group. Most text editors have a find or search function you can use to quickly find this group heading. Then add the following line:

For this export format	Enter...
XML	**09=;XM;XML Export;WXMW32->WXMExportXML**

This assumes **09** is not already being used. Here is an example:

```
< ExportFormats >
    09=;XM;XML Export;WXMW32->WXMExportXML
```

Setting up the XML import format

Follow these steps to set up the XML import format:

1. Open the FSISYS.INI file in the resource library for which you want to use export files. You can use any text editor to open this file.
2. Locate the ImportFormats control group. Most text editors have a find or search function you can use to quickly find this group heading. Then add the following line:

For this import format	Enter...
XML	**09=;XM;XML Import;WXMW32->WXMImportXML**

This assumes **09** is not already being used. Here is an example:

```
< ImportFormats >
    09=;XM;XML Import;WXMW32->WXMImportXML
```

Setting up the XML message format

To send a message from Documaker Workstation to a message handling program such as IDS or MQSeries, you must add the EntryFormset INI option. Follow these steps:

1. Open the FSISYS.INI file in the resource library for which you want to use export files. You can use any text editor to open this file.
2. Locate the AFEProcedures control group. Most text editors have a find or search function you can use to quickly find this group heading. Then add the following option:

For this option	Enter...
EntryFormset	**WXMW32->WXMEntryHookExtXMLLoad**

Here is an example:

```
< AFEProcedures >
    EntryFormset = WXMW32->WXMEntryHookExtXMLLoad
```

Setting Up IDS

If you are using IDS as the message server, you must also add the INI options shown below to let Documaker Workstation retrieve an archived record from IDS and load data into a form set before any data is entered by a user.

The archived record is retrieved using the Key1, Key2, and KeyID entered on the New Form Set window. For this to happen, you must set up the following request type in the DOCSERV.INI file for IDS:

```
< ReqType:GetXML>
    function = atcw32->ATCLogTransaction
    function = atcw32->ATCLoadAttachment
    function = atcw32->ATCUnloadAttachment
    function = dprw32->DPRSetConfig
    function = dprw32->DPRLocateOneRecord,Key1,Key2,KeyID
    function = dprw32->DPRRetrieveFormset
    function = dprw32->DPRPrint
    function = dprw32->DPRProcessTemplates
    function = atcw32->ATCSendFile, DOCC_XML, SENDBACKPAGE, TEXT
```

You can use any name for the archive library, as long as the same MRL name is used in Documaker Workstation.

You can set up the new function as an entry hook:

```
< AFEProcedures >
    EntryFormset = WXMW32->WXMEntryHookExtXMLLoad
```

If you set it up as an entry hook, you must also set up these INI options:

```
< XML_IMP_EXP >
    DSIUseNTUserID         =
    DSIVARS                =
    DSIIgnoreTimeoutError  =
    DSIAttachedVarFile     =
    DSIImportLevel         =
    DSITimeout             =
    DSIReqType             =
    DSIRecordDFD           =
```

Option	Description
DSIUseNTUserID	(optional) Set this option to Yes to use the NT user ID. The default is No. This gives you a way to pass the NT user ID in the queue instead of the normal DMWS ID.
DSIVARS	(optional) Enter *variable;value*, where *variable* is the variable name and *value* is its value. This lets you identify a constant list of variables to be sent in the queue.
DSIIgnoreTimeoutError	(optional) Enter Yes to continue processing if a timeout occurs. The default is No. This gives you a way to ignore a timeout when waiting on a return queue.
DSIAttachedVarFile	(optional) The default is DOCC_XML. Set this option to the attachment name if it differs from DOCC_XML. This gives you a way to specify the variable name the XML file is attached to.

Option	Description
DSIImportLevel	(optional) This option is typically used by programmers. Enter 2 if you want the hook to operate on the FAP_MSGOPEN level. Enter 3 if you want it to operate on the FAP_MSGRUN level. The default is 2.
DSITimeout	(optional) Enter the number of milliseconds you want for the timeout. The default is 60000 milliseconds or 60 seconds.
DSIReqType	(optional) Enter the name of the request type of the message placed in the queue. The default is GETXML.
DSIRecordDFD	(optional) Enter the name of a DFD file. The system tries to match variable fields sent in the request to field values in this DFD file. It then attaches the DFD record to the end of the message.

If the request for an XML file comes back with an error, as opposed to a time-out, IDS displays an error message.

XML File Format

Here is an example of the format of the XML file the system creates:

```
<?xml version="1.0" encoding="UTF-8"?>
<DOCUMENT TYPE="RPWIP" VERSION="10.2">
 <DOCSET NAME="">
  <FIELD NAME="POLICY NBR">P1234-1</FIELD>
  <FIELD NAME="RENEWAL NBR">1234-2</FIELD>
  <FIELD NAME="AGENT&apos;S NBR">6789</FIELD>
  <FIELD NAME="EFFECT DATE">10/1/02</FIELD>
  <FIELD NAME="EXPIRE DATE">10/1/03</FIELD>
  <FIELD NAME="INSURED NAME">John A. Doe</FIELD>
  <FIELD NAME="ADDR1">2345 Anystreet</FIELD>
  <FIELD NAME="CITY">Anytown</FIELD>
  <FIELD NAME="STATE">GA</FIELD>
  <FIELD NAME="ZIP CODE">30339</FIELD>
  <FIELD NAME="BUSINESS DESC1">Business</FIELD>
  <FIELD NAME="BUSINESS DESC2">Personal</FIELD>
  <FIELD NAME="BUSINESS DESC3">Property</FIELD>
  <FIELD NAME="DATE">09/27/02</FIELD>
  <GROUP NAME="" NAME1="DOCUCORP PACKAGE"
   NAME2="PROFESSIONAL INSURANCE">
   <FORM NAME="Professional Dec">
    <DESCRIPTION>Professional Declarations
     </DESCRIPTION>
    <FIELD NAME="FORM LINE1">Form Letter</FIELD>
    <RECIPIENT NAME="AGENT" COPYCOUNT="1"/>
    <RECIPIENT NAME="HOME OFFICE" COPYCOUNT="1"/>
    <RECIPIENT NAME="INSURED" COPYCOUNT="1"/>
    <SHEET>
     <PAGE>
      <SECTION NAME="profdec"/>
     </PAGE>
    </SHEET>
   </FORM>
   <FORM NAME="Form Letter">
    <DESCRIPTION>Form Letter</DESCRIPTION>
    <RECIPIENT NAME="AGENT" COPYCOUNT="1"/>
    <RECIPIENT NAME="HOME OFFICE" COPYCOUNT="1"/>
    <RECIPIENT NAME="INSURED" COPYCOUNT="1"/>
    <SHEET>
     <PAGE>
      <SECTION NAME="let~tbl">
       <FIELD NAME="Coverage">Automobile</FIELD>
       <FIELD NAME="Extra">
        <P><FONT SIZE="12"
        FACE="Univers ATT" COLOR="#FF0000">Text in
        multiline variable field.</FONT>
        </P>
       </FIELD>
      </SECTION>
     </PAGE>
```

Form set global data

Group

Form global fields

Form

Page

Recipient information

Multi-page form

Multi-page image

Image local fields

Multi-line field

Indicates a second page

```
        <PAGE>
          <SECTION NAME="let~tbl">
            <DAPINSTANCE VALUE="2"/>
            <DAPOPTIONS VALUE="M"/>
          </SECTION>
        </PAGE>
      </SHEET>
    </FORM>
  </GROUP>
 </DOCSET>
</DOCUMENT>
```

Keep in mind...

- DAPOPTIONS should have a value of *M* for multi-page FAP images. There are other FAP image options, but only *M* is applicable in XML.

  Use DAPINSTANCE to provide a page number for multi-page FAP images. If the image does not span multiple pages, omit the DAPINSTANCE value.

- When you have multiple XML transactions within a single file, separate each transaction with a line feed. This is a requirement of Documaker software, not the XML parser.

- Although you do not have to include line feeds inside the XML for a transaction, Docucorp suggests you add a line feed after each element tag. This makes it easier to read the file and helps in debugging your XML. A message like

  ```
  Line 255, column 8, syntax is incorrect
  ```

  is easier to diagnose than

  ```
  Line 1, column 156780, syntax is incorrect.
  ```

TRANSFORMING XML FILES

You can export an XML file with XSLT transformation. This lets you transform the output XML file into another format, such as HTML or text. The final output format is determined by the XSLT template you choose.

To enable this export add the following INI option to the ExportFormats control group:

```
< ExportFormats >
    01=;MX;Export with XSL;WXMW32->WXMExportWithXSL
```

and then add these options to support the new option above:

```
< ExportWithXSL:MX >
    XSLTName        =
    Executable      =
    Debug           =
< ExportWithXSL >
    XSLTName        =
    Executable      =
    Debug           =
```

Option	Description
XSLTName	Enter the full or relative path and name of the XSL style sheet.
Executable	Enter the full path and name of the XSLTW32.EXE program. If you omit this option, the program will use the entry for the Executable option in the ExportWithXSL control group. If that option is missing, the program will query the full path of the XSLTW32.EXE program.
Debug	Enter Yes to send debug output to the trace file.

Appending output transformations

You can append multiple XSLT output transformations to the same file using this INI option:

```
< ExpFile_cd >
     AppendedExport = Yes
```

NOTE: The default control group used by the WXMExportWithXSL rule is the ExportWithXSL control group. If you specify another control group in your PPS selection, such as ExportWithXSL:M1, and it is either not found in the INI file or one of the expected options for that group is missing, the system tries to read the default value from the ExportWithXSL control group.

Here is an example. This example transforms an XML export into a semicolon-delimited output file you can import into Excel. It also uses the in-process executable XSLTW32.EXE for the transformation.

First, you need these INI options:

```
< ExportWithXSL:M1 >
   XSLTName = x:\rp\mstrres\aeic\xsl\output1.xsl
   Debug = no
```

And this XSL style sheet:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">

  <xsl:output method="text" encoding="ISO-8859-1" />

  <!-- global variables -->



  <xsl:template match="/">
    <xsl:call-template name="process"/>
  </xsl:template>


  <xsl:template name="process">

         <xsl:variable name="semicolon" select="';'"/>
         <xsl:variable name="root" select="DOCUMENT/DOCSET"/>
         <xsl:variable name="policy" select="$root/
FIELD[@NAME='POLICY']"/>
         <xsl:variable name="insnam" select="$root/
FIELD[@NAME='INSNAM']"/>
         <xsl:variable name="insnam2" select="$root/
FIELD[@NAME='INSNAM2']"/>
         <xsl:variable name="insad1" select="$root/
FIELD[@NAME='INSAD1']"/>
         <xsl:variable name="insad2" select="$root/
FIELD[@NAME='INSAD2']"/>
         <xsl:variable name="inszip" select="$root/
FIELD[@NAME='INSZIP']"/>
         <xsl:variable name="agent" select="$root/
FIELD[@NAME='AGENT']"/>
         <xsl:variable name="effdte" select="$root/
FIELD[@NAME='EFFDTE']"/>
         <xsl:variable name="expdte" select="$root/
FIELD[@NAME='EXPDTE']"/>
         <xsl:variable name="cddesc" select="$root/
FIELD[@NAME='CDDESC_BUSDSC']"/>
         <xsl:variable name="premo_prop" select="$root/
FIELD[@NAME='PREMO_PROP']"/>
         <xsl:variable name="advprem" select="$root/
FIELD[@NAME='ADVPREM']"/>
         <xsl:variable name="totpre" select="$root/
FIELD[@NAME='TOTPRE']"/>
         <xsl:variable name="galmt" select="$root/
FIELD[@NAME='GALMT']"/>
         <xsl:variable name="prcolmt" select="$root/
FIELD[@NAME='PRCOLMT']"/>
         <xsl:variable name="pailmt" select="$root/
FIELD[@NAME='PAILMT']"/>
         <xsl:variable name="perocc" select="$root/
FIELD[@NAME='PEROCC']"/>
         <xsl:variable name="fdlmt" select="$root/
FIELD[@NAME='FDLMT']"/>
         <xsl:variable name="medlmt" select="$root/
FIELD[@NAME='MEDLMT']"/>


         <xsl:value-of select="concat($policy, $semicolon)"/>
```

```
          <xsl:value-of select="concat($insnam, $semicolon)"/>
          <xsl:value-of select="concat($insnam2, $semicolon)"/>
          <xsl:value-of select="concat($insad1, $semicolon)"/>
          <xsl:value-of select="concat($insad2, $semicolon)"/>
          <xsl:value-of select="concat($inszip, $semicolon)"/>
          <xsl:value-of select="concat($agent, $semicolon)"/>
          <xsl:value-of select="concat($effdte, $semicolon)"/>
          <xsl:value-of select="concat($expdte, $semicolon)"/>
          <xsl:value-of select="concat($cddesc, $semicolon)"/>
          <xsl:value-of select="concat($premo_prop, $semicolon)"/>
          <xsl:value-of select="concat($advprem, $semicolon)"/>
          <xsl:value-of select="concat($totpre, $semicolon)"/>
          <xsl:value-of select="concat($galmt, $semicolon)"/>
          <xsl:value-of select="concat($prcolmt, $semicolon)"/>
          <xsl:value-of select="concat($pailmt, $semicolon)"/>
          <xsl:value-of select="concat($perocc, $semicolon)"/>
          <xsl:value-of select="concat($fdlmt, $semicolon)"/>
          <xsl:value-of select="concat($medlmt, $semicolon)"/>

       <xsl:text>&#xA;</xsl:text>

  </xsl:template>




</xsl:stylesheet>
```

And this XML export file:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <DOCUMENT TYPE="RPWIP" VERSION="10.2">
- <DOCSET NAME="">
  <FIELD NAME="POLICY">A108</FIELD>
  <FIELD NAME="INSNAM">SAM MALONE</FIELD>
  <FIELD NAME="INSNAM2">CHEERS, INC.</FIELD>
  <FIELD NAME="NEW">X</FIELD>
  <FIELD NAME="INSAD1">123 MAIN ST</FIELD>
  <FIELD NAME="INSAD2">SUITE 100</FIELD>
  <FIELD NAME="INSCTY">ATLANTA</FIELD>
  <FIELD NAME="INSST">GA</FIELD>
  <FIELD NAME="INSZIP">23033</FIELD>
  <FIELD NAME="AGENT">12345</FIELD>
  <FIELD NAME="AGYNAM">Docucorp Insurance Agency</FIELD>
  <FIELD NAME="AGYAD1">2727 Paces Ferry Road S.E.</FIELD>
  <FIELD NAME="AGYAD2">Suite II-900</FIELD>
  <FIELD NAME="AGYCTY">Atlanta</FIELD>
  <FIELD NAME="AGYST">GA</FIELD>
  <FIELD NAME="AGYZIP">30339</FIELD>
  <FIELD NAME="PRMSTE">GA</FIELD>
  <FIELD NAME="EFFDTE">07/05/2003</FIELD>
  <FIELD NAME="EXPDTE">07/05/2004</FIELD>
  <FIELD NAME="TERM">366 DAYS</FIELD>
  <FIELD NAME="CDDESC_BUSDSC">BAR & GRILL</FIELD>
```

```
  <FIELD NAME="PREMO_PROP">12,000.00</FIELD>
  <FIELD NAME="ADVPREM">12,000.00</FIELD>
  <FIELD NAME="FEEDESC1">Policy Tax</FIELD>
  <FIELD NAME="FEEDESC1 TAX">3%</FIELD>
  <FIELD NAME="FEEAMT1">360.00</FIELD>
  <FIELD NAME="FEEDESC2">Stamping Fee</FIELD>
  <FIELD NAME="FEEAMT2">250.00</FIELD>
  <FIELD NAME="OTHCHG">610.00</FIELD>
  <FIELD NAME="TOTPRE">12,610.00</FIELD>
  <FIELD NAME="CSIGNEDLOC">Atlanta, GA</FIELD>
  <FIELD NAME="SIGNED DATE">07/30/2003</FIELD>
  <FIELD NAME="SIGNED TIME">09:25:18</FIELD>
  <FIELD NAME="OPINIT">DOCUCORP</FIELD>
  <FIELD NAME="SIGNATURE">Authorized Representative</FIELD>
  <FIELD NAME="GALMT">1,000,000</FIELD>
  <FIELD NAME="PRCOLMT">1,000,000</FIELD>
  <FIELD NAME="PAILMT">1,000,000</FIELD>
  <FIELD NAME="PEROCC">1,000,000</FIELD>
  <FIELD NAME="FDLMT">1,000,000</FIELD>
  <FIELD NAME="MEDLMT">1,000,000</FIELD>
- <GROUP NAME="" NAME1="American Equity" NAME2="INTERLINE">
- <FORM NAME="FS100 10-2000">
  <DESCRIPTION>Schedule of Forms/End</DESCRIPTION>
  <FIELD NAME="FORM DESC LINE">Forms Applicable - INTERLINE</FIELD>
  <FIELD NAME="FORM DESC LINE #003">A100J 02-1999 Policy Jacket -
AEIC</FIELD>
  <FIELD NAME="FORM DESC LINE #004">A100 03-1997 Common Policy Dec -
AEIC</FIELD>
  <FIELD NAME="FORM DESC LINE #005">A101 03-1997 Minimum Earned
Premium Endt</FIELD>
  <FIELD NAME="FORM DESC LINE #006">A104 10-1998 Service of Suit</
FIELD>
  <FIELD NAME="FORM DESC LINE #007">IL0017 11-1998 Common Policy
Conditions</FIELD>
  <FIELD NAME="FORM DESC LINE #008">IL0021 04-1998 Nuclear Energy
Liab Excl Endt</FIELD>
  <FIELD NAME="FORM DESC LINE #010">Forms Applicable - GENERAL
LIABILITY</FIELD>
  <FIELD NAME="FORM DESC LINE #012">CL150 01-2000 General Liab
Coverage Part</FIELD>
  <FIELD NAME="FORM DESC LINE #013">L003 03-1997 Amendment of Premium
Condition</FIELD>
  <FIELD NAME="FORM DESC LINE #014">L005 01-2000 Contractual Liab
Limitation</FIELD>
  <FIELD NAME="FORM DESC LINE #015">L007 07-1998 Ded Liab Ins-w/Costs
per Claim</FIELD>
  <FIELD NAME="FORM DESC LINE #016">L150 01-2000 Additional
Exclusions</FIELD>
  <FIELD NAME="FORM DESC LINE #017">CG0001 07-1998 Comm General Liab
Cov Form</FIELD>
  <FIELD NAME="FORM DESC LINE #018">CG2160 09-1998 Excl - Year 2000
Computer Prob</FIELD>
  <RECIPIENT NAME="EXTRA COPY" COPYCOUNT="1" />
  <RECIPIENT NAME="GENERAL AGENT" COPYCOUNT="1" />
  <RECIPIENT NAME="HOME OFFICE" COPYCOUNT="1" />
  <RECIPIENT NAME="ORIGINAL" COPYCOUNT="1" />
  <RECIPIENT NAME="RETAIL AGENT" COPYCOUNT="1" />
- <SHEET>
```

```
- <PAGE>
  <SECTION NAME="FORMSCHA" />
  </PAGE>
  </SHEET>
  </FORM
  </DOCSET>
  </DOCUMENT>
```

The output file looks like this:

```
A108;SAM MALONE;CHEERS, INC.;123 MAIN ST;SUITE 100;23033;12345;07/
05/2003;07/05/2004;;   12,000.00;
12,000.00;;1,000,000;1,000,000;1,000,000;1,000,000;1,000,000;1,000,
000;
```

You can import this file into an Excel spreadsheet.

Multiple User and Networking Issues

In a networked, multi-user environment you must carefully handle files to avoid conflicts. Two users trying to write to the same file at the same time is an obvious problem. In addition, problems can arise if one user tries to read a file, while another is writing to it. The import and export features of the system avoid these types of file conflicts.

When a user imports a file, the system opens the file and prevents or delays other users from writing to that file. Likewise, while a user is writing and exporting a file, other users are prevented or delayed from reading or writing to that same file. Multiple users can, however, read from the same import file at the same time.

When a conflict occurs, the workstation attempts to secure the file (in the proper mode) for up to 15 seconds. If the file becomes available within that time, the requested operation continues as expected. If, however, the file does not become available, the operation fails and the system displays a message. If the user is exporting a file, the message tells the user the export failed and asks if the user wants to try again. If the user is importing a file, the message simply states that the import file is invalid.

To change the delay or wait time, set the FSIWAIT environment variable to the number of seconds you want the system to wait. Place this setting in your AUTOEXEC.BAT file. For example, enter...

SET FSIWAIT=number of seconds

If you set FSIWAIT too low, operations may fail more frequently on a heavily used network. If you set it too high, the user may become concerned the program is no longer functioning.

NOTE: If you set FSIWAIT to zero, the system uses the default of 15 seconds instead. The least amount of wait time you can specify is one second.

Most of the time, users will be unaware of any delay because the import and export processes typically take only a second or two. The file is then released to other users.

Problems may still arise in environments where users open the import or export file in a file editor or some other program that keeps the file opened.

CHAPTER 6

Maintaining Your System

System maintenance ensures the integrity and smooth functionality of Documaker Workstation. System supervisors have access to maintenance functions unavailable to general system operators primarily through the Tools menu.

Maintenance functions include activities such as compressing the database, deleting form sets to maximize the system's processing efficiency, maintaining user IDs, and troubleshooting potential user ID problems. To make sure the system is fully operational at all times, you should schedule maintenance activities on a regular basis.

In addition to maintenance activities, the Tools menu lets system supervisors increase forms processing efficiency by importing multiple form sets to the WIP menu for data entry and by letting supervisors set up routing slips to direct the work flow.

This chapter discusses...

All activities focus on using the system on single user workstations. If you use the system on a networked system, your maintenance activities may differ slightly. Consult your network documentation for additional information.

Handling Unknown Users

You can use the AFEOS2->AFEVerifyWIPUserID function to display a window which shows WIP assigned to users unknown to the system. An *unknown* user is a user not defined in the UserInfo database.

To use this function, add the following to your MEN.RES file. You can place this function anywhere you like, such as on the Tools menu.

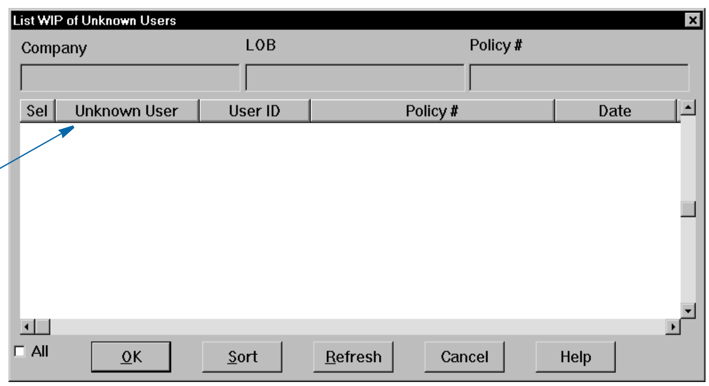
```
MENUITEM "List WIP of Unknown Users" 283 "AFEOS2->AFEVerifyWIPUserID"
"List WIP of Unknown Users" 0
```

You can name the menu option and window any way you like. In the example shown above, it is set to *List WIP of Unknown Users*. You can also enter any unused menu ID in the range of 201-300. This range of menu IDs are inactive when a form set is open and active when no form sets are open.

The zero (0) indicates the security level of the user who can use this function. Level 0 is typically used for supervisor-type functions such as this.

When you set up the function and then choose the List WIP of Unknown Users option, the WIP window that appears is a variation of the Assignment window which includes a new column listing users unknown to the system.

This column lists users unknown to the system.



You can select multiple WIP records assign those records to the appropriate users. This window works like the normal Assignment window except it does not let you assign the WIP to a remote user via email.

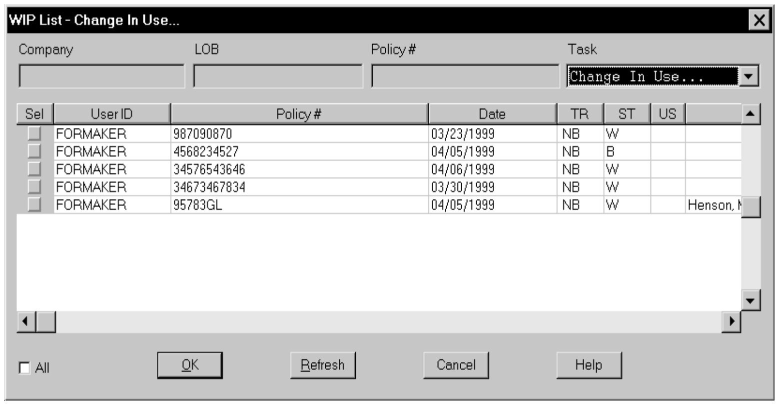
Understanding the System

The CurrUser field from the WIP records tells the system which users are *unknown*. The system uses the CurrUser field as a key to find the user's ID in the UserInfo database.

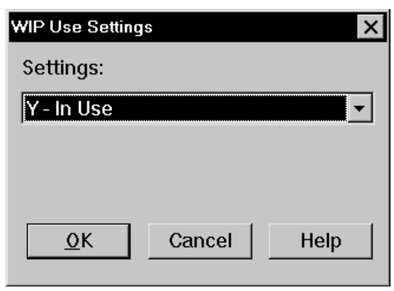
Changing the In Use Status

Changing a form set's In Use status lets you unlock a document and make it available for editing in the WIP list. You must have a system supervisor access level to perform this task.

To change a document's In Use status, choose the WIP, WIP List option. Then select the Change In Use option from the Task field. The Change In Use window appears.



1 Click the document whose In Use status you want to change; then, click Ok. The WIP Use Settings window appears.



2 Click the In Use status you want for the document; then, click Ok. The system returns to the Change In Use window with the new In Use status displayed.

NOTE: A blank space in the In Use column indicates a document is not in use; therefore, choosing *Not in Use* will leave the column blank.

---

3 Click the upper right corner of the Edit In Use window, or click Cancel, to close the window.

After changing a document's In Use status you may need to change the form set's status to place it back in the WIP list for editing. See the Documaker Workstation User Guide for information on this task.

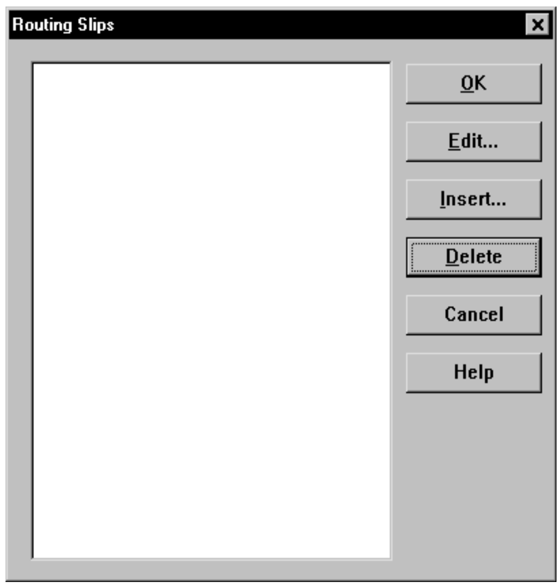
Setting Up Routing Slips

Setting up routing slips lets you set up an email directory for sending documents. You can group recipients by department, or organize them in any way. When you send documents using routing slips, the system sequentially sends a document to each individual in the order they appear on your list. The system then routes the document back to you.

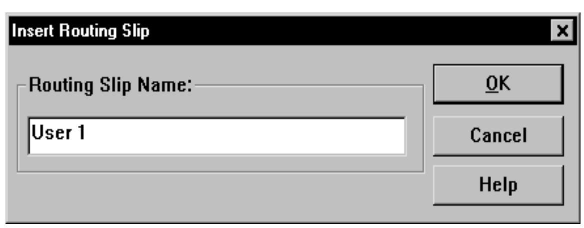
Note: *All* documents must be closed *before* you set up a routing slip.

Follow these steps to set up a routing slip:

1 Choose Tools, Routing Slips. The Routing Slips window appears.

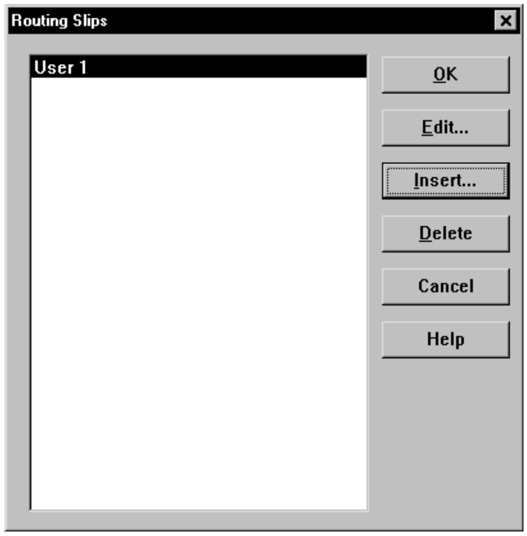


2 Click Insert. The Insert Routing Slip window appears.



3 Type the title of the routing slip in the Routing Slip Name field; then, click Ok. The name appears in the routing slip list.

Note: The system limits the number of characters that display in the routing slip list to only eight, including spaces.



You can now add recipient names to your routing slip. See the next topic for more information.

Adding Routing Slip Recipients

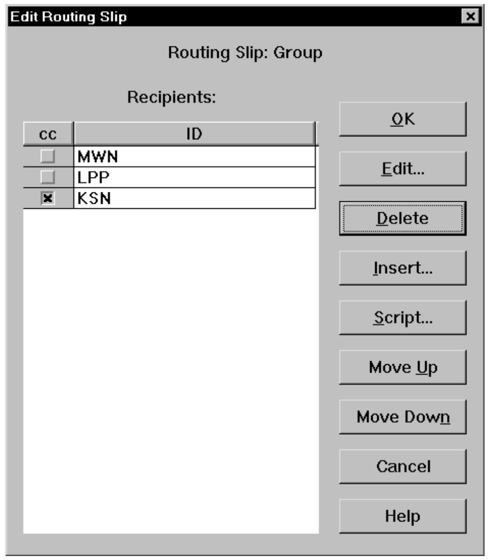
Adding routing slip recipients lets you designate and group individuals to whom you email documents. You can use the routing slip recipient list to direct the flow of work and to provide read-only copies of documents at chosen stages in the work flow. You can also insert a DAL script in the routing slip recipient list to direct the work flow based on values in a certain field of the document.

You must first set up a routing slip before you can add recipients. See the previous topic for information about setting up routing slips.

NOTE: You must set up your INI files to use email options before you set up recipients.

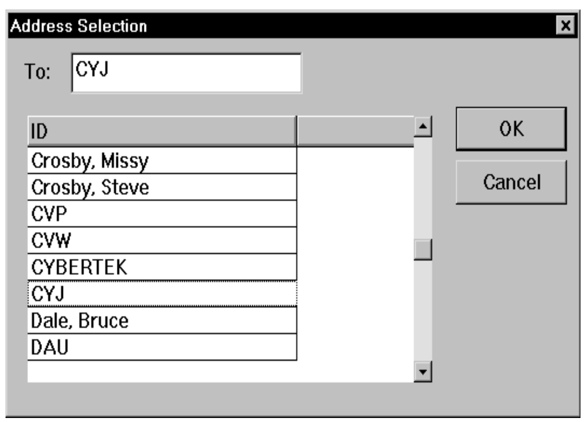
Follow these steps to add routing slip recipients:

1 In the Routing Slips window, click the routing slip to which you want to add recipients; then, click Edit. The Edit Routing Slip window appears.



2 In the Edit Routing Slip window click Insert. The first time you add a recipient, the system displays your email login window.

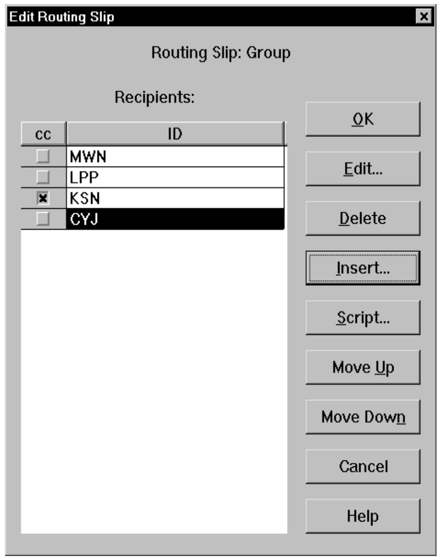
3 Type your email user ID and password; then, click Ok. The system displays the Address Selection window.



4 Type or select your first routing slip recipient; then, click Ok to add the recipient to your list. The Edit Routing Slip window displays your recipients.

5 Repeat Steps 2 and 4 for each recipient you want to add.

NOTE: The system will route the documents in the order in which they appear in the recipient list. To change the order, click the recipient you want to change; then, click Move Up or Move Down. Click Cancel at any time to exit the window.



6 To provide a read-only copy of the form set to a recipient, click the CC column next to the recipient's name. Read-only copies cannot have data added or modified; the recipient can only view and/or print the document. Recipients with the CC column checked receive a copy of the document the recipient above received. When the first recipient completes work on the document, it is routed to the next recipient in the list.

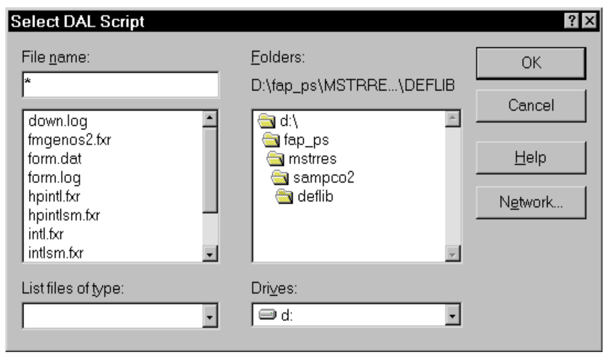
This table shows how routing occurs:

cc	ID	Receives...
	First Person	the document and completes his work. The document is then routed to *Second Person*.
	Second Person	*First Person's* completed document and enters data in his portion of the document. The package is then routed to *Third Person*.
X	John Doe	a read-only copy of the same document *Second Person* received.
X	Jane Doe	a read-only copy of the same document *Second Person* received.
	Third Person	*Second Person's* completed work and enters data in his portion of the document. The package is then routed back to the originator of the package.

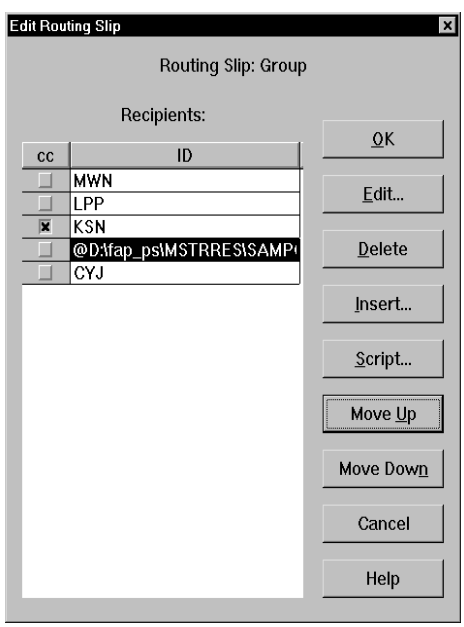
NOTE: You can set up routing slips in which all recipients receive read-only copies of the form set. Simply click the CC column next to each recipient's name and the system broadcasts a read-only copy to everyone in the routing slip.

7 If you want to add a DAL script to the recipient list, click the Script button. DAL scripts can be written and inserted in the recipient list to route the document according to values entered in certain variable fields. When the linear recipient above the DAL script completes work on the document, the script reads the values returned in the specified fields and routes the document to the recipient designated by the returned values. When you click Script, the Select DAL Script window appears.

NOTE: A DAL script is an ASCII text file which can be written using any text editor. For information on DAL functions and file format, see the DAL Reference. For instructions on placing the script in the master resource library, see Using Resource Libraries on page 17. For information on configuring the INI files, see Configuring INI Files on page 128.



8 Enter the name of the DAL script in the File Name field, or select the script from the file list. Change the drive and directory if necessary. Click Ok; the system returns to the Edit Routing Slip window.



9 The DAL script appears in the recipient list. When the previous recipient completes work on the document, the system reads the DAL script and routes the document accordingly. To change the DAL script's position in the recipient list, select the script; then, click Move Up or Move Down.

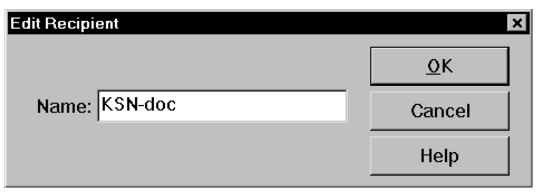
NOTE: The DAL script is not copied and sent with the document. The script must already be present in the recipient's files to execute. For information on DAL scripts, see the DAL Reference.

10 When you have completed adding recipients to your routing slip, click Ok to return to the Routing Slips window.

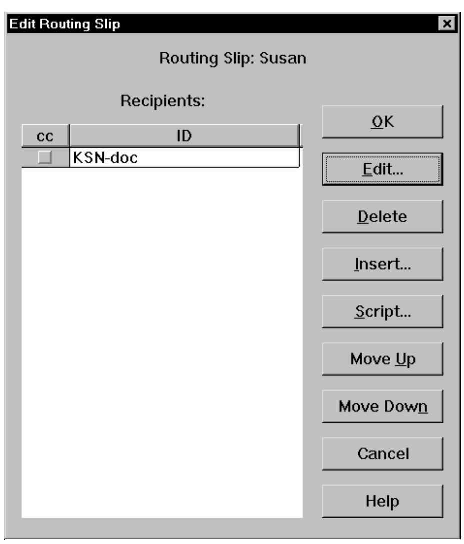
11 Click Ok in the Routing Slips window to return to the system's main window. When you email documents, you can send them using routing slips.

Editing Routing Slip Recipients

You can edit the data associated with a routing slip recipient. You may want to edit a recipient if the email address is incorrect, or to add information related to the recipient. To edit a recipient, click the recipient in the Edit Routing Slip window; then, click Edit. The Edit Recipient window appears.



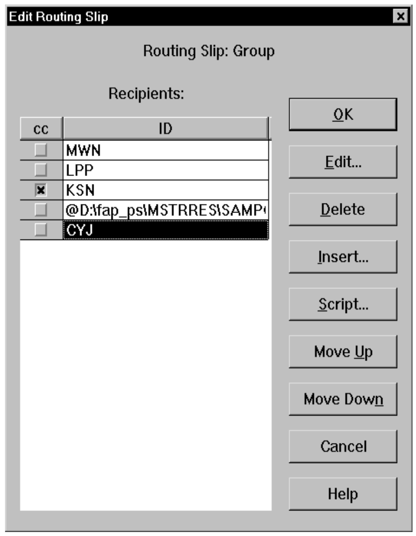
Edit the recipient data as necessary; then, click Ok. Your changes appear in the Edit Routing Slip window.



Deleting Routing Slip Recipients

You can delete routing slip recipients to allow for personnel changes and transfers.

To delete a routing slip recipient, open the Routing Slip window and click the routing slip which contains the recipients you want to delete. Click Edit to display the Edit Routing Slip window.



Click the recipient you want to delete; then, click Delete. The system removes the recipient from the routing slip.

Deleting Routing Slips

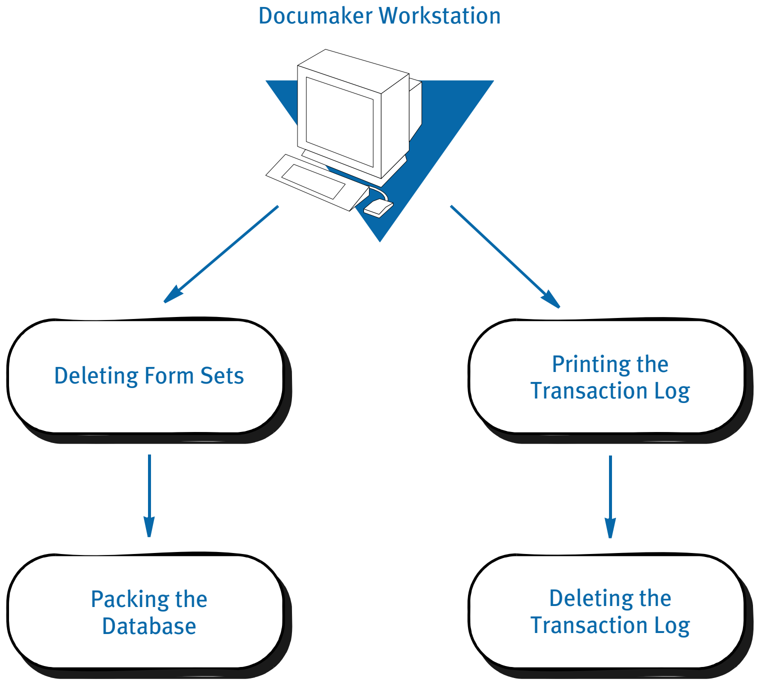
Deleting routing slips lets you remove an entire routing slip from your list. When you delete a routing slip, you delete all recipients' names within the name of your routing slip.

To delete a routing slip, choose Tools, Routing Slips. The system displays the Routing Slips window.

Click the routing slip you want to delete; then, click Delete. The system removes the routing slip from the Routing Slips window and you can no longer use it.

Maintaining Form Sets

Maintaining form sets in Documaker Workstation lets you monitor storage and track transactions. By monitoring and managing storage, you enhance your system performance and efficiency. Tracking transactions lets you view and print status reports of user and form set transactions to assist you with maintenance.



Deleting Form Sets

Deleting form sets lets you delete any unused, incorrect, or outdated form sets from the system. Remember that when you delete the form sets, you delete the data from the library you opened when logging onto the system.

If you want to delete form sets from several libraries, you must delete them within each selected library. Archived form sets cannot be deleted. Although users can no longer access deleted form sets, the form set record remains in the system.

See Packing the WIP Database on page 305, for information on removing all deleted form set references from the system.

Form Set Status

Before deleting a form set, you should understand the status codes associated with form sets:

- **W (Work-in-Process)** - The system lets users save form sets to Work-in-Process prior to printing and archiving. The user can later edit and print the form set. After the user prints a completed form set, the system changes the form set status from W to AR (Archived) as defined in the FSIUSER.INI file (refer to Configuring INI Files on page 128, for more information on FSIUSER.INI options).

- **P (Printed)** - The system lets users print completed WIP form sets without archiving them if the FSIUSER.INI file is configured as *Archival Mode = MANUAL*.
- **B (Batch Print Queue)** - The system lets users send form sets to a batch print queue for printing at a later time. The user cannot edit form sets with status B. The user can, however, change status B to W (Work-in-Process) and then edit the form set. The system archives status B form sets after printing them.
- **AR (Archived)** - The system archives form sets for permanent storage after printing via the Complete window. Users cannot edit the form set, but they can view or print the form set. Users cannot delete archived form sets.
- **User Defined Codes** - The system lets you define status codes in the FSIUSER.INI file. These codes appear in the WIP status list. See Configuring INI Files on page 128, for more information.

Deleting Form Sets in WIP

Deleting form sets in WIP lets you delete form sets from the combined Work-in-Process list. The combined Work-in-Process list includes form sets saved to WIP and form sets sent to the batch print queue. Deleting form sets removes the form sets from the WIP list, but does not delete the form set records from the system.

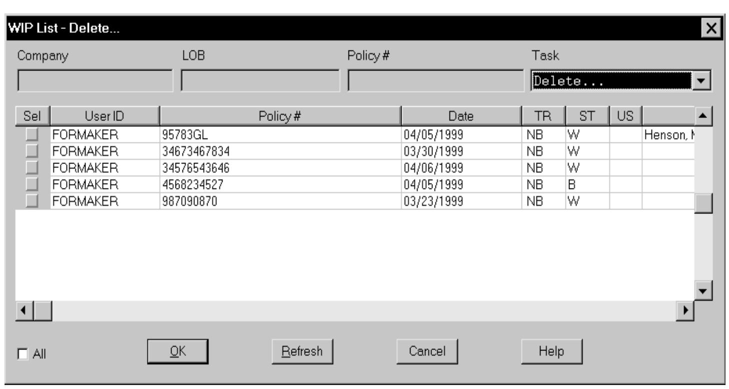
NOTE: Packing the WIP database lets you permanently delete the form set records from your system. See Packing the WIP Database on page 305 for more information.

You should check with users to determine if their WIP data needs to be saved before you delete it. If your system is set for manual archive, completed and printed form sets appear in the WIP list with a P (Printed) status code, but have not been archived. If data needs to be saved, you should manually archive or backup that form set data in WIP before you delete other form sets.

Follow these steps to delete form sets in WIP:

1 Start the system using the resource library icon from which you want to delete form sets in WIP.

2 Choose the WIP, WIP List option. Then choose the Delete option from the Task field. The Delete window appears, containing the combined WIP list of all users' form sets.



3 Select the form sets you want to delete by clicking the Sel column in the WIP list. You can also select and deselect form sets by using the directional arrows to move to the form set line. Then press the SPACEBAR to select or deselect.

4 Click Ok after you have selected all the form sets you want to delete. The Delete Confirmation window appears.

5 Click Yes to delete your selected form sets, or No to retain the form sets in WIP.

PACKING THE WIP DATABASE

Packing the WIP database lets you remove all deleted form set references from the system. Your system's processing efficiency corresponds to your database size. When you delete form sets, users cannot access the form sets. The record of the form set, however, still resides in the system, using hard disk space. Packing the WIP database removes the form set record from your system. Pack the WIP database after deleting form sets, or at any time your system seems sluggish.

NOTE: You should pack the database only when all users are logged off the system. Additionally, you should note that packing the WIP database can take an extended period of time, depending on the number of deleted form sets, so you may want to schedule this activity for after business hours.

Follow these steps to pack the WIP database:

1 Start the system using the company resource library icon for which you want to pack the WIP database.

2 Choose the Tools, Database Maintenance option. Then choose the Pack Database option. A confirmation window appears. Click Yes to pack the database, or No to cancel. The system displays a message to tell you if the operation was successful.

MAINTAINING THE TRANSACTION LOG

The transaction log is a database which lets you review and track all user activity within a given resource library. The table below lists each transaction log heading and its description:

Log Heading	Description
Date Range Requested	Lists all activities performed within a specified start and end date
Company	Company or client associated with a form set
Line of Business	Specific business or industry associated with a form set. Examples: General Liability, Property, Inland Marine
Policy Number	User-defined identifier (alphanumeric) associated with a specific form set

Log Heading	Description
Activity	Type of process or transaction associated with a particular form set, such as Create, Update, Complete, or Delete
User ID	User ID of the user performing the activity
Date	Date the user performed a specific transaction
Time	Time the user performed a specific transaction

Managing the Log File

You can control the amount of information stored in this file, the name of the file, and where the file is located.

Determining what actions to track

The FSISYS.INI or FSIUSER.INI file settings in the AFELOG control group let you determine what activities the system tracks in the log file. You have these options:

Option	Tells the system to make a log entry when a user...	Default
1	Creates a transaction	Yes
2	Updates a transaction	No
3	Deletes a transaction	Yes
4	Saves a transaction	Yes
5	Completes a transaction	Yes
6	Receives a transaction	Yes
7	Sends a CC copy of a transaction	Yes
8	Sends a form set	Yes
9	Assigns a form set	Yes
10	Archives a form set	Yes
11	Prints a form set	Yes
12	Changes the status of a form set	Yes
13	Exits a form set without saving it	No
14	Retrieves a form set	No
15	Aborts the system	Yes

You only need to include changes to the defaults in the INI file—you do not have to include default settings.

To turn on any option that is off (No) by default, enter the number shown above followed by *Yes*. To turn off any option that is on (Yes) by default, enter the number shown above followed by *No*.

For example, to have the system log all updates to transactions and to accept all other defaults, you would enter:

```
< AFELog >
    2 = Yes
```

Typically, you would leave this option set to No (off) because it results in very large log files.

Specifying where the log file is stored

The default log file is named *AFELOG*. The AFELOG file is, by default, stored in the WIP directory. You can, however, use the following INI options to specify another file name or location for the log file.

```
< AFELog >
    File = file name
    Path = directory path
```

If you enter a path for the FILE option, your entry overrides any path you specify in the PATH option. If your entry for the FILE option does not include a path, the system uses the path you enter for the PATH option, if any.

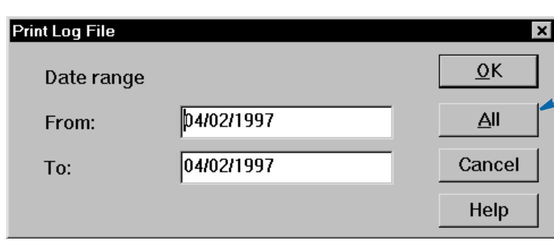
NOTE: See Configuring INI Files on page 128, for more information on customizing the transaction log.

Printing the Transaction Log

Printing the transaction log lets you view all user activity within a form library, for a specific date range. Since you cannot display this information, you must print the transaction log to track user activity.

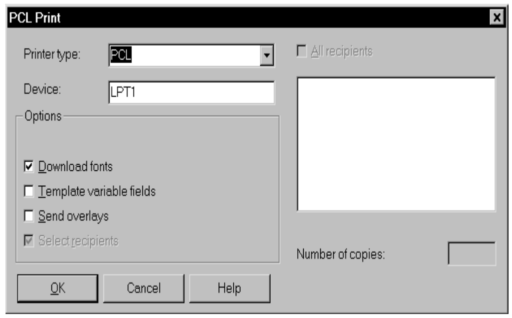
To print the transaction log:

1. Start the system using the Company Resource Library icon from which you want to print the transaction log.
2. Choose Tools, Print Log File. The system displays the Print Log File window.



To print the entire transaction log since the last transaction log deletion, click the All button.

3. In the From field, type the beginning date for the range of user transactions that you want to print.
4. In the To field, type the ending date for the range of user transactions you want to print, and click Ok. The Print window appears.



You can change the printer type or printer port in the appropriate fields. For an explanation of Print window options, see the Documaker Workstation User Guide.

5 Click Ok to print the transaction log. If there are no transactions within the dates you specified, the system displays a message telling you just that.

Deleting the Transaction Log

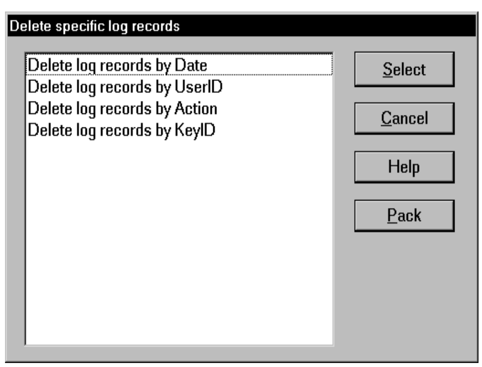
Deleting the transaction log lets you clear all the entries from the system's transaction log. Remember that each form library has its own transaction log. If you want to delete the transaction log from several libraries, you must do so within each selected library.

Since the system records every user action performed, the transaction log grows rapidly. We suggest that you print and delete the transaction log on a regular basis.

To delete the transaction log:

1 Start the system using the Company Resource Library icon from which you want to delete the transaction log.

2 Choose the Tools, Database Maintenance option. Then choose the Delete Log Records option. The Delete Specific Log Records window appears.



3 Select the criteria you want to use to delete records. For example, you can delete log records by date, user ID, action, or by KeyID. Then click Select. The delete window appears.

NOTE: Click Pack to remove from the system all previously deleted log records.

4 Select the records you want to delete and then click Ok. The system shows you the number of records deleted and returns you to the delete window. Click Cancel to exit.

Modifying FORM.DAT Files

The FORM.DAT file (*form set definition table*) defines how images are organized within forms, and how those forms are grouped into form sets. In addition, the FORM.DAT file defines the recipients for your form sets. You use the Form Set Manager, which is part of Docucreate, to create and modify FORM.DAT files.

You modify the FORM.DAT file whenever you add new forms. If you receive forms from an external forms provider, normally you will receive the complete FORM.DAT file with those forms. You should not modify a FORM.DAT file you have received from an external source, unless you are creating forms to add to that resource library.

Each resource library contains a FORM.DAT file which defines each form available in that library. The FORM.DAT file is located in the DEFLIB directory of your resource library.

Refer to the Docucreate User Guide for information on how to use the Form Set Manager.

Using a Text Editor to Modify the FORM.DAT File

The FORM.DAT file is a semicolon-delimited file which can be edited using a text editor, *but this is not recommended*. Furthermore, Support Services will not be able to help you if problems arise.

If your license does not include the Form Set Manager and you choose to modify the FORM.DAT file using a text editor, you must change *each* field and replace the information with your new library name, new forms, images, and designated recipient copies.

In addition, you make sure any information you add to the FORM.DAT file is placed in the appropriate location. For example, if you are adding a form to a form set, you must locate the form set information in the FORM.DAT file and add the new form in the appropriate spot within the form set.

It is *imperative* that you create the file in the exact format shown on the following pages. A single misplaced semicolon *corrupts* the FORM.DAT file, so you must modify it carefully. Consider these guidelines and recommendations when creating or modifying a FORM.DAT file:

- *Always* create a backup copy of the FORM.DAT file before you modify it.
- Use a text editor that saves files as *straight ASCII text*.
- Insert semicolons (;) between *each* element on a record line, and braces ([ ]) between *each* image recipient list.
- *Do not* use tabs or any text formatting when you create a FORM.DAT file.

Follow these steps to create or modify the FORM.DAT file:

1. Open the FORM.DAT file in the DEFLIB directory of your new master resource library directory.

NOTE: If you choose to modify an existing FORM.DAT file, first copy the file from the DEFLIB directory of the SAMPCO resource library.

2 Using the format guidelines that follow, modify the existing FORM.DAT file, or create a new FORM.DAT file using the sample FORM.DAT file as a template.

Sample FORM.DAT File

The following example shows the first two lines of a sample FORM.DAT file. A FORM.DAT file contains one record line corresponding to each form in the library.

Line 1

```
;DOCUCORP PACKAGE;COMMERCIAL PACKAGE;DEC PAGE;DEC
PAGE;RX;;CPDEC~1|D[INSURED(1),HOME OFFICE(1),AGENT(1)];
```

Each semicolon delimited field represents a particular piece of information on a form. Each FORM.DAT record line contains two distinct sections of information:

- The *form section* associates the form with a company and line of business (all semicolon (;) delimited fields) and includes the name and description of the form.
- The *image section* details each image, image recipients, and number of recipient image copies .
    - A pipe (|) separates the image name from its related information.
    - Recipient information appears within braces ([ ]).
    - Recipient image copies appear within parentheses (( )).
    - A comma (,) separates recipient image copies from the next recipient name.
    - A backslash (\) at the end of a line indicates that the definition continues on the next line.

This example shows a FORM.DAT record format:

Form section

```
;COMPANY;LOB;EXTERNAL NAME;DESCRIPTION;
FORM OPTIONS;COPY COUNT
```

Image section

```
;IMAGE1|IMAGEOPTIONS[RECIPIENT1(COPIES),...RECIPIENTn
(COPIES)]/IMAGE2|OPTIONS[RECIPIENT1(COPIES),...
RECIPIENTn(COPIES)]/...IMAGEn|OPTIONS[RECIPIENT2
(COPIES)];
```

Formatting the FORM.DAT File

The following tables contain FORM.DAT file layouts and guidelines for the form section and the image section. Refer to these tables as you create the FORM.DAT file.

```
;COMPANY;LOB;EXTERNAL NAME;DESCRIPTION;
FORM OPTIONS;COPY COUNT
```

Field	Length	Description
Company	20	Company name
LOB	20	Line of business
External Name	20	Common form name associated with form

Field	Length	Description
Description	30	Form description, external name is often used for form description
Form Options: Select up to five options per form		
R	1	Required form, includes pull and electronic forms
P	1	Pull form, includes forms unavailable in electronic form
M	1	Repeatable form, form can appear multiple times within a form set.
X	1	Master Dec Page, include in a package policy
S	1	Sub Dec Page, include one per form set within a package.
D	1	Monoline Dec Page, first page of a single form set
Copy Count	N/A	Not applicable, add a semicolon to mark the field

NOTE: *Do not* use the following form option combinations in the FORM.DAT file: DM, XM, SM, DX, DS, SX

```
;IMAGE|IMAGEOPTIONS[RECIPIENT1(COPIES),...RECIPIENTn
(COPIES)]/IMAGE2|OPTIONS[RECIPIENT1(COPIES),...
RECIPIENTn(COPIES)]/...IMAGEn|OPTIONS[RECIPIENT2
(COPIES)];
```

The FORM.DAT file considers the image section (all image names, recipients, and image copies) as one field. Form definitions that exceed 1,024 characters must be broken into several lines using the continuation character (\).

Section	Description
Image 1... Image n	Image name without an extension. Maximum length is 8.
Image Options	Defines the image type. Select one option from each category View/Print, Page, and Form.
View/Print	A = Print only, cannot display or enter data onto the form C = Data entry only, cannot print the form D = Data entry and print E = View only, cannot enter data or print the form H = View and print, cannot enter data on the form
Page	F = Front page of a duplexed page, must also use back page option. Also use with multi-page images B = Back page of a duplexed page, must also use front page option

Section	Description
Form	L = Letter size. System default - 8-1/2” x 11" G = Legal size, 8-1/2” x 14" I = A4 size. Standard European size J = Executive size. European size
Recipient 1...n	Those receiving image copies. Propagates to SEND COPY TO: field. The maximum length is 20.
Copy Count	Default # of copies for recipient. Each recipient may receive a varied number of copies. Length is 1.

Setting Up Bookmarks and Pull Forms

Pull forms indicate where a preprinted (not printed by this system) form should be placed in a form set. Bookmarks define how the system should note the placement of pull forms. You define bookmarks using these INI options:

```
< Bookmark >
    Name    =
    Size    =
```

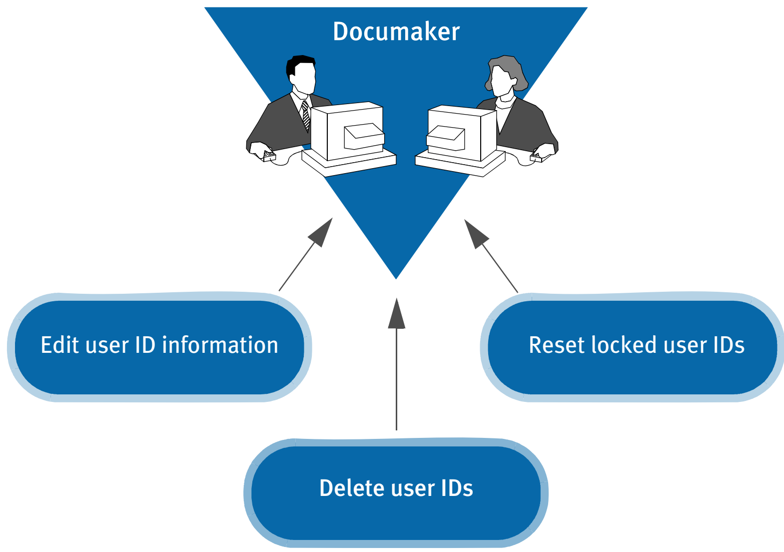
Option	Description
Name	Enter the name of the FAP file to print on the bookmark page.
Size	Choose the paper size from these options: L = Letter size G= Legal size The default is letter size.

You define the pull forms used in the form set in the FORM.DAT file. For each pull form, you must indicate the name of the form and its recipients. You must also mark it as a pull form (P).

To have the system include the pull form, it must be marked as required in the FORM.DAT file or selected by the user on the Form Selection window.

Maintaining User Information

Maintaining user information is an important facet of system maintenance. As a system supervisor, you are responsible for making sure user information is correctly set up and properly maintained in the system. Maintenance includes editing and deleting user IDs and resetting locked user IDs as needed.



Maintaining User ID Information

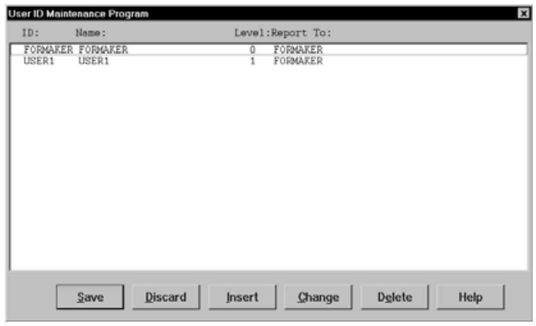
Maintaining user ID information lets you update the user IDs and related information within your system. User ID maintenance includes editing user ID information and deleting specific user IDs. For information on adding a new user, see Setting Up Multiple Users on page 10.

Editing User ID Information

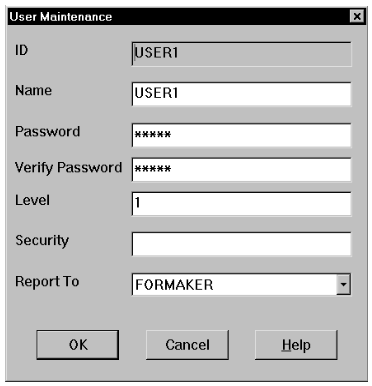
Editing user ID information lets you change the user information associated with a particular ID. You can change a user's name, password, access level rights, security level, and the person to whom the user reports. When editing user ID information, do not change the ID itself. Changing an ID results in the inability to open form sets stored under the particular user ID. If you must change a user ID, delete the ID, then create a new one.

To edit user ID information:

1 Choose Tools, User ID Maintenance, to display the User ID Maintenance Program window.



2 Click the user for which you want to edit information, and click Change to display the User Maintenance window.



3 In the Name field, type the user's name with any changes or corrections.

NOTE: Use this step to alter a current user's name, not to add a new user to the system. See Setting Up Multiple Users on page 10, for information about adding a new user.

4 Type the user's new password in the Password field. If you don't type a new password, the user's password remains the same.

5 Type a number between 1 and 9 to assign a new user access level in the Level field, and press TAB twice to skip the Security field (future use). Zero (0) is reserved for supervisor access level.

NOTE: You can enable or disable main menu options depending on a user's access level.

6 Click the scroll arrow next to the Report To field to display a list of users. Select the user's new supervisor from the list.

7 Click Ok to update the user information. The system returns you to the User ID Maintenance Program window.

8 Click Save to record the updated user ID information. The system displays the Save Confirmation window.

9 Click Yes to save the updated user ID information. Click No to cancel the changes.

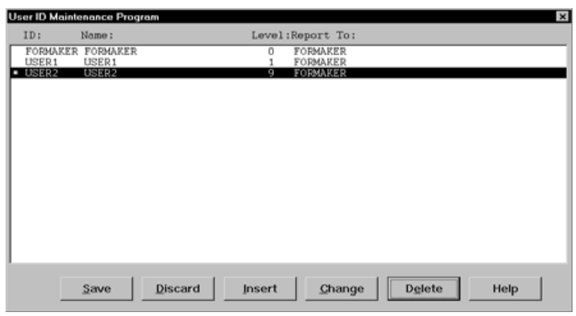
NOTE: To revert your most recent unsaved settings to their prior status, click Discard in the User ID Maintenance Program window. Discard lets you make changes to a user ID, then quickly revert to the prior settings if you change your mind.

Deleting a User ID

Deleting a user ID lets you delete a specific user ID from the system. When you delete a user ID, the user cannot start the system.

For instructions on deleting user IDs using SuperUser access, see Deleting Users on page 318. Follow these steps to delete a user ID:

1 Choose Tools, User ID Maintenance to display the User ID Maintenance Program window.



2 Click the user you want to delete, then click Delete. An asterisk (*) appears next to the selected ID.

NOTE: The Delete button is a toggle. The system deselects the user ID you marked for deletion if you click Delete again.

3 Click Save to remove the user ID from the User ID Maintenance Program list, and to display the Save Confirmation window.

4 Click Yes to save the updated user ID information, or No to cancel your changes.

NOTE: To revert your most recent unsaved settings to their prior status, click Discard in the User ID Maintenance Program window.

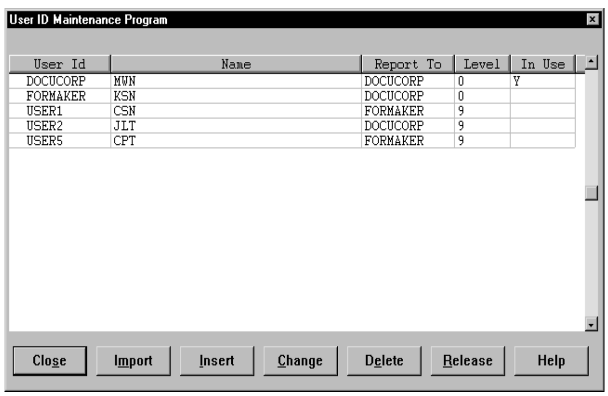
USING SUPERUSER ACCESS

With SuperUser access, the system displays an enhanced version of the User ID Maintenance Program window. SuperUsers must have a security level of zero (0) and the SupportSuperUser option must be configured in the INI file. See Setting Up SuperUsers on page 12.

The SuperUser window lets you...

- Search for a user by name or user ID by double-clicking on the appropriate column heading. You can enter a partial text string or an exact match.
- Import user information from a database or text file.
- Reset locked user IDs.

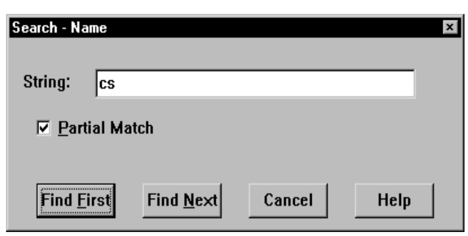
There is no Save button in this window because the system saves your information as you enter it.



To change the security level so users with a security level between 1 and 9 can use these features, see Assigning Access Rights on page 13.

Searching for a User

You can double click on any of the column headings in the User ID Maintenance window to do a full or partial text search in that field.



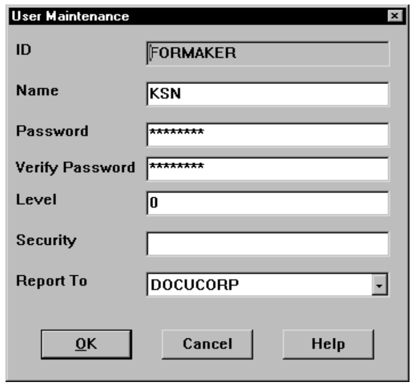
Enter the text you want to search for, indicate whether or not the system should look for partial matches, and click Find First.

Changing User Information

Once you have selected the user you want to change...

- Highlight the user and click the Change button or
- Double-click on the user you want to change

This brings up the User Maintenance window where you can change the user's data.



Deleting Users

To delete a user, select the user in the User ID Maintenance window and click delete. The system asks you to confirm the deletion. You can set the following option to suppress the confirmation window.

```
< UserInfo >
    SuppressDeleteDialog = Yes
```

The system warns you if you try to delete yourself.

If other users are reporting to the deleted user, the system asks you who to reassign as the users' report to person and displays a list of all users.

Replacing a User in a Report To List

For example, suppose the CSN user ID has three users that report to it. If you have SuperUser rights and you delete CSN from the User ID Maintenance window, the following window appears (assuming CSN is not currently logged in):

Select the appropriate report-to person to replace CSN and click OK. The users that once reported to CSN now report to the new user.

If you want to suppress the User Replace List window, add this INI option:

```
< UserInfo >
```

```
SuppressReportToDlg = Yes
```

If you set the SuppressReportToDlg option to Yes, the system sets the current user as the new report to person for each person in the deleted user's report to list.

Importing User Information

To use the Import button, the UserImport functions must be configured in the INI file. See Setting Up SuperUsers on page 12.

With SuperUser access, you can import user information into the system user database from a...

- Text file. The text file should have the following format.

```
UserID,Name,Password,ReportTo,Level
```

You can leave a parameter blank as long as you include the comma separators. You can only have one user per line in the text file.

- Database file.

Follow these steps to import user information:

1 In the User ID Maintenance window, click Import. The Import Formats window appears.

2 Select an import format and click Ok. The Select Import File window appears.

3 Choose the file to import and click Ok. If the system notes an existing user ID among those being imported, it asks you to choose one of the following options:

Choose	To
Skip	Skip the current record being imported.
Skip All	Skip all records that conflict with an existing user ID
Replace	Update the existing record with the record being imported.
Replace All	Update all existing records with new records which have the same ID.
Cancel	Stop the import process. Records imported before you clicked cancel remain in the database.

After you are finished importing new IDs, you return to the User ID Maintenance window.

NOTE: If you are importing a file which contains a record with a report to person who is not in the database, the system produces a warning and skips that record. You can use the following option to suppress this feature so the record is imported even if the report to person does not exist.

```
< UserInfo >
    ImportBadReportTo = Yes
```

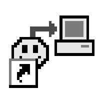
The default (No) is to show the warning and not import the record.

Resetting Locked User IDs

If the system is shut down inadvertently, such as when there is a loss of power, user IDs can become locked. Once locked, you cannot use the ID to start the system. Instead, when the user attempts to restart the system, it displays a message stating the user ID is already in use. This occurs because the system assumes the user is still logged on. Resetting a locked user ID lets you unlock the ID so it can be used again.Resetting a Locked User ID

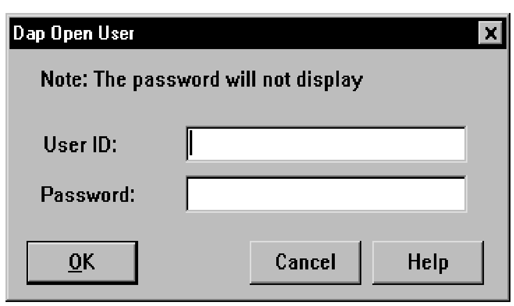
With SuperUser access, you can select a user in the User ID Maintenance window and click the Release button to reset the user ID. The system displays a message confirming the user ID has been reset.

To reset a locked user ID using the OpenUser program, click the OpenUser icon:



or run the openuw32.exe program from the working directory used by your Documaker Workstation icon.

The Open User window appears:



1 Type the locked user ID in the User ID field.

2 Type the user password in the Password field. Click Ok. The system displays a message confirming the user ID has been reset. Click Ok.

You can now open the system using the user ID and password.

Splitting Archives

The Archive Maintenance option on the Tools menu lets you split an archive into two files based on criteria you specify.

You can use this option for a variety of purposes, such as...

- If you need to free space or speed retrieval, you can extract older form sets and store them separately on CD or other media
- If your company is closing one location, you can split that location's archives for distribution to the remaining locations.

You specify the criteria the system will use to split the archive by selecting a cut-off date or by using a DAL script. Using a DAL script lets you split an archive based on any field in the form sets in the archive. This illustration shows what happens:

Original Archive

Transaction #	Date
2170001	02172005
2170002	02172005
2170003	02182005

Before splitting an archive, always back up the archive file.

Failure to back up an archive before performing a split could result in the loss of information crucial to your company.

If you simply split this archive using a cut-off date of 03012005, the result is...

Original Archive

Transaction #	Date
2170001	02172005
2170002	02172005
2170003	02182005

Newly-created Archive

Transaction #	Date

Note that the system marks as deleted the records it split from the original archive. Though marked as deleted, those records still exist in the original archive and can be accessed. If you also elected to purge the original archive, the result is shown below:

Original Archive

Transaction #	Date
2170001	02172005
2170002	02172005
2170003	02182005

Newly-created Archive

Transaction #	Date

As shown above, purging an archive file after performing a split deletes archive information and index entries from your original archive.

The blank lines in the archives are there for illustrative purposes only. The system does not allow blank records to exist in an archive file.

Making a backup of the archive file, CAR files, and the archive index before you begin this operation protects you in case of a power outage or some other event which would result in the loss of information crucial to your company.

NOTE: You can also use the ARCSPLIT command-line utility to split an archive. The Archive Maintenance option simply provides a more user-friendly way of using the ARCSPLIT utility. The ARCMERGE utility lets you combine two archives. There is no graphical user interface for the ARCMERGE utility. For more information, see the Docutoolbox Reference.

SPLITTING AN ARCHIVE

The system includes a convenient way for you to split archive files. To add this feature onto your system menu, include the following line in your MEN.RES file:

```
MENUITEM “&Split Archive...” 9299 “AGIW32->AGISplitArchive”
“Archive split utilities” 0
```

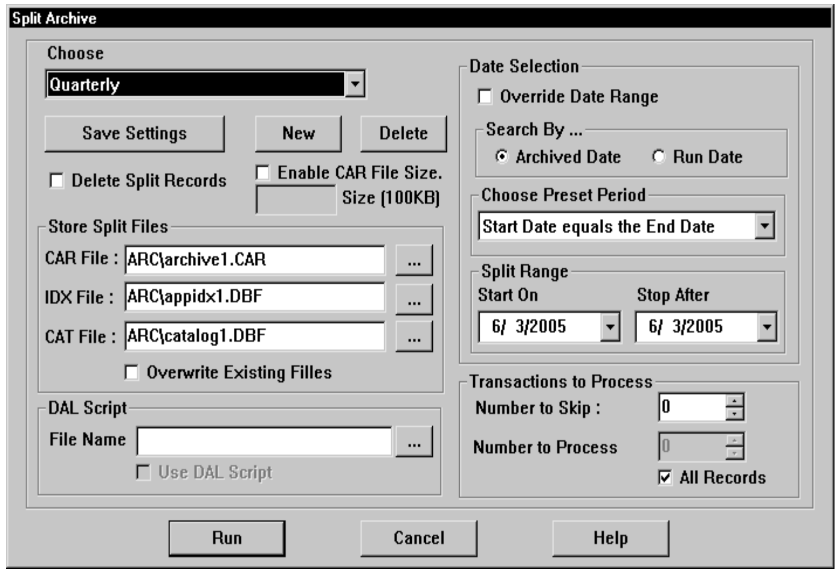
Here is an example:

```
POPUP “&Tools” 255 “Utility Programs”
  BEGIN
    MENUITEM “&Split Archive...” 9299 “AGIW32->AGISplitArchive”
       “Archive split utilities” 0
    SEPARATOR
    ...
```

NOTE: This option performs the same task as the ARCSPLIT utility. For information on the new parameters for this utility, see the Docutoolbox Reference.

When you choose the Split Archive option from the Tools menu, as shown in the example, the system prompts you to enter a name for your split archive settings:

Once you enter a name for the settings, the Split Archive window appears:



On this window you can tell the system how to split an archive, save those settings for future use, and split an archive file. When splitting an archive file, you can specify a period of time in which the archive should be split. For instance, you can split a file based on the archived date or run date, select from predefined periods, or enter a specific date range. You can also specify which records to process and where the system should store the files it creates.

NOTE: If no settings exist, the system prompts you to enter a name for the new settings you are creating.

Use these fields to tell the system how to split the archive:

Field	Description

Choose Settings	Lists the different settings. You can add or delete these settings, based on your security settings.
	You determine who can change these settings using the AllowChanges option, as shown here:
	`< ArcSplit >`
	`AllowChanges = Yes`
	The default is No.

Delete Split Records	Check this field to delete the split records from the original file. Be sure to *back up* your archive files before doing this.
Enable CAR File Size	Check this field to specify a limit to the size of the CAR files the system will create. Specify the file size in the field below the check box.
Store Split Files	Here you tell the system where you want to store the split files. Enter the file name and full path. If you omit the file name, an error occurs when you click the Run button.

Field	Description
Use DAL Script	Select the DAL script you want the system to run and check the Use DAL Script field. The system compiles the DAL script to enhance optimization.
Override Date Range	Check this field if you want to override the date range specified in the Split Range fields.
Search By	Select which date to base the search on. You can choose between the archive date or the run date. The archive date reflects when the record was archived. The run date tells you when the record was last modified.
Choose Preset Period	Select from predefined ranges such as less than or equal to 90 days, greater than or equal to 60 days, and so on. Your choice is based on the date you enter as the starting date in the Split Range field.
Split Range	Enter the date on which to start and the date on which to stop. For instance, 01/01/06 to 12/31/06 covers an entire year.
Transactions to Process	Specify the record with which the system should start and the number of records you want to process. For instance, if you have a huge archive, you can limit processing time by limiting the number of records to process at one time. If you omit the starting record, the system starts with the first record that matches the criteria you set on the other fields.

NOTE: The settings are stored in the INI file. See INI options on page 325 for more information.

You can also use these buttons:

Button	Description
Save Settings	If you click this button, the current settings are saved to the INI file.
New	Click New to add a setting. The system prompts you for the name of the new setting and adds the name in the list of settings. Once you enter the name, you can begin with the current settings or start with no options selected.
Delete	Click Delete to remove the current settings and select the next setting on the list. If the list is empty, the system prompts you to create a new setting.
Run	Click Run to split an archive based on the setting you selected.
Cancel	Click Cancel to close the window without splitting an archive.

INI options

In addition to parameters, you can also set up INI options to specify criteria. The system looks in the FSIUSER.INI file for these options. The following INI options are required. These options tell the system where to find the source archive files you want to split.

```
< ArcRet >
    AppIdx  = arc\appidx.dbf
    ArcPath = arc\
    CARFile = archive
    CARPath = arc\
    Catalog = arc\catalog
```

The system also looks for options that specify the names of the split archive files and provide other information. These options are located in the ArcSplit and ArcSplitConfig control groups:

```
< ArcSplit >
    ArcSplitConfig      =
    DefConfig           =
< ArcSplitConfig:TEST1 >
    SplitDays           =
    RunDALScript        =
    DALScript           =
    RecordsToProcess    =
    RecordsToSkip       =
    SearchDateBy        =
    SplitAppIdx         =
    SplitCARFile        =
    SplitCatalog        =
    SplitFromDate       =
    SplitToDate         =
    PurgeRecords        =
    CARFileSize         =
    EnableCARFileSize   =
    AllowChanges        =
    LogFile             =
```

Option	Description
ArcSplit	
ArcSplitConfig	Lets you assign a name to your archive split settings. You can create as many ArcSplitConfig options as you need. Each group of settings represents a control group with individual options. Below is an example of a setting named *TEST1*.
DefConfig	If you set up multiple ArcSplitConfig options for Documaker Workstation, you can use this option to designate a default.
ArcSplitConfig:*TEST1*	
SplitDays	(optional) Enter the number of days you want the utility to add to the start date to determine the end date. The utility then splits the archive between the start and end dates.
RunDALScript	(optional) Enter Yes if you want the utility to run the DAL script you specified with the DALScript option.

Option	Description
DALScript	(optional) Enter the name of the DAL script you want to run. Use the RunDALScript option to tell the utility if it should run the DAL script you specify with this option.
RecordsToProcess	(optional) Enter the number of records to process. Use this option if you have a very large archive and you want to limit the number of records processed at one time.
RecordsToSkip	(optional) Enter the number of the record the utility to process first. If you omit this option, the utility starts with the first record.
SearchDateBy	(optional) Enter *RunDate* to search the records based on the RunDate. Enter *ArchivedDate* to search the records based on the date on which they were archived.
SplitAppIdx	(optional) Enter a name for the newly split IDX file, such as APPIDX1. The utility stores the data in the file you specify. If the file exists and you have set the /D parameter, the utility overwrites the file. If you have not set the /D parameter and the file exists, the utility stops.
SplitCARFile	(optional) Enter a name for the newly split CAR file, such as ARCHIVE1. The utility stores the data in the file you specify. If the file exists and you have set the /D parameter, the utility overwrites the file. If you have not set the /D parameter and the file exists, the utility stops.
SplitCatalog	(optional) Enter a name for the newly split catalog file, such as CATALOG1. The utility stores the data in the file you specify. If the file exists and you have set the /D parameter, the utility overwrites the file. If you have not set the /D parameter and the file exists, the utility stops.
SplitFromDate	(optional) Enter the date on which you want the split to begin. The default is the current date. The format is MM/DD/YYYY.
SplitToDate	(optional) Enter the date on which you want the split to end. The default is the current date. The format is MM/DD/YYYY.
PurgeRecords	(optional) Enter Yes if you want the utility to purge from the master archive the records it split from the archive. The default is No, which tells the utility to copy but not delete those records. The utility stores the copied records in the files and directories you specified.
CARFileSize	Enter a number between one (1) to 14,000 to define the size for the CAR file. If you enter one (1), the utility interprets that as 100,000 bytes (100 KB). If you enter 14,000, the utility interprets that as 1,400,000,000 bytes (1,4000,000 KB). Omit this option if the EnableCARFileSize option is set to No.

Option	Description
EnableCARFileSize	(optional) This option turns on and off the related radio button field on the Split Archive window and also tells the system whether to use the CARFileSize option. The default is No.
AllowChanges	(optional) Lets you change settings from a window. The default is No.
LogFile	(optional) Enter the name of the file into which the utility should write any error messages. Here is an example: `LogFile = c:\errlog.txt`

Using DAL scripts

You can use DAL scripts to when you split or back up an archive. The system makes available to the DAL script all of the APPIDX column values you specify. This script can only return Yes or No. If anything else is returned, the system defaults to No.

All field names specified in the script file must include the word *ARCSPLIT*, such as *ARCSPLIT.KEY1*. This is required in case multiple index files are in use.

NOTE: Refer to the DAL Reference for information on the DAL functions you can use to create the scripts.

Assume the following form sets are stored in the archive, the INI options are set as shown below, and the DAL scripts COMPANY.DAL and COMBINED.DAL exist in the DefLib directory.

KeyID	Key1	Key2	Date archived
AA	Sampco	LB1	03/01/1999
BB	Sampco	LB2	03/01/1999
CC	FSI	GL	03/02/1999
DD	FSI	GF	03/02/1999
EE	MyCompany	GO	03/03/1999

Also assume these INI options are set:

```
< ArcRet >
    SplitAppIdx     = arc1\AppIdx1.dbf
    SplitCARFile    = arc1\Archive1.car
    SplitCatalog    = arc1\Catalog1.
```

The COMPANY.DAL script looks like this:

```
If ARCSPLIT.Key1 = "FSI" then Return ("Yes");
                        Else Return ("No");
End
```

The COMBINED.DAL script looks like this:

```
if (ARCSPLIT.KEY1 = "FSI " AND ARCSPLIT.KEY2 = "GL ") then
Return ("YES");
Else
Return("NO");
End
```

NOTE: Make sure the value you specify matches the field length defined in the DFD (Database Field Definition) file. In this example, the field length of KEY1 is four characters and the search value should be "FSI " (with a space between I and the ending quotation mark) instead of "FSI".

Based on these assumptions, this table shows the results if you enter the following commands to run the ARCSPLIT utility:

If you enter...	The result is...
ARCSPLIT /sd=19990301 / ed=19990301 / ini=fsiuser.ini	Records AA and BB are written to the archive files (ARCHIVE.CAR, APPIDX1.DBF, CATALOG1.DBF, APPIDX1.MDX, and CATALOG1.MDX) in the arc1 directory. The records in the master archive are not changed.
ARCSPLIT /sd=19990301 / ed=19990301 / ini=fsiuser.ini	Records AA and BB are written to the archive files in the arc1 directory. The records in the master archive are not changed. The APPIDX1 and CATALOG1 files will be flat files.
ARCSPLIT /sd=19990301 / ed=19990301 / ini=fsiuser.ini /p	Records AA and BB are written to the archive files in the arc1 directory. These records are also deleted from the master archive.
ARCSPLIT /dal=deflib\company.dal ini=fsiuser.ini	Records CC and DD are written to the archive files in the arc1 directory. The records in the master archive are not changed.
ARCSPLIT /dal=deflib\combined.dal ini=fsiuser.ini	Record CC is written to the archive files in the arc1 directory. The records in the master archive are not changed.

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X