Content Manager OnDemand for Multiplatforms Version 7.1 Release Notes

Content Manager OnDemand for Multiplatforms Version 7.1 Release Notes

Note

Before using this information and the product it supports, read the information in "Notices" on page 139.

Second Edition (April 2001)

This edition applies to IBM® Content Manager OnDemand for Multiplatforms, Version 7 Release 1 and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About the Release Notes

The information about OnDemand contained in this Release Notes file was not available at the time that the OnDemand product publications were produced. Please make a note of these items, which supersede the information in the OnDemand product publications. For the latest information about OnDemand, see the OnDemand Web site at:

`http://www.software.ibm.com/data/ondemand`

Change history

1. Version 7.1.0.0 First release March 30, 2001.
2. Version 7.1.0.1 Second Edition April 20, 2001. Added or updated the following information:
   - ACIF indexing. Added the "INDEXOBJ parameter" on page 3.
   - "Chapter 2. Adobe software" on page 5. Added a note about the `ARSPDF32.API` file. Removed Adobe Business Tools from the list, because it is no longer provided by Adobe. New customers should use Adobe Acrobat Version 5 to view PDF documents from the client. Added a note about processing PDF input files with the administrative client.
   - "Chapter 6. Hardware and software requirements" on page 13. Updated the requirements for the CICS/ESA® client. Added a note about the OnDemand PDF Indexer feature. Added a note about Adobe Acrobat Version 5. Added a note about processing PDF input files with the administrative client.
   - "Chapter 10. PDF indexing" on page 33. Added a section on how to use the graphical indexer to generate indexing information for PDF input files.
   - "Chapter 12. Support" on page 41. Added the component IDs for the clients and the server.
   - Xenos transform. Replaced the sample script file in "Indexing and converting input data" on page 116.

Part 1. General release notes

Chapter 1. ACIF indexing

Concatenating resources to an AFP™ file

Note: Add the following section to Chapter 6, Hints and Tips, in the *Indexing Reference*, SC27–0842–00.

A resource group can be created and stored in a file by using the ARSACIF program. The resource file and the AFP file can then be concatenated together to form a file that can be processed by the indexing program. The following lists the parameters used to create a resource file using the ARSACIF program. The parameters process an AFP file named `credit.afp`, which is an AFP file that contains no indexing information or inline resources. The example is for an AIX® system. For this example, the output file and the index file that ACIF usually generates are not needed; all resources are assumed to be in the directory named by the USERLIB parameter.

Contents of the ACIF parameter file `parms.acif`:

```
CC=YES
CCTYPE=A
RESTYPE=OVLY,PSEG,FEDF
INPUTDD=credit.afp
OUTPUTDD=/dev/null
INDEXDD=/dev/null
RESOBJDD=credit.res
USERLIB=/usr/resources
```

Command used to generate the resource group file:

```
arsacif parmdd=parms.acif
```

Command to concatenate the resource group file and the AFP file:

```
cat credit.res credit.afp > credit.out
```

You can then process the `credit.out` file with the indexing program, using indexing information that is specified in a `credit.ind` file.

INDEXOBJ parameter

The INDEXOBJ parameter now includes support for stapling on document boundaries when processing for Infoprint® Manager.

Important: This function should not be used with OnDemand.

ACIF normally removes any Begin/End Document structured fields from the input file and generates a single BDT/EDT for the entire output because MO:DCA indexes are relative to the Begin Document structured field. However, the stapling function uses BDT/EDT to indicate document boundaries for stapling. A new indexing option has been added to allow ACIF to pass through any BDT/EDT and not create it's own. This file is suitable for printing, but should not be used with indexing because the resultant index will not be MO:DCA compliant and may not be processed correctly by programs which use the index, such as OnDemand.

To enable BDT/EDT pass through, specify the BDTLY option on the INDEXOBJ parameter. For example:

```
INDEXOBJ=BDTLY
```

Inline resources

Note: Add the following section to Chapter 6, Hints and Tips, in the *Indexing Reference*, SC27–0842–00.

There is currently a restriction in ACIF that requires inline resources used with the input file to be in a particular order. The order (which is the same order that ACIF produces in the Resource Object file) is for all resources included or used by another resource to appear inline before the resource that uses it. For example, if an overlay includes a coded font, then the coded font must include a code page and character set. The inline resources must be in the following order:

code page

character set

coded font

overlay

Otherwise, because ACIF does not look ahead in the inline resources, it will try to read the included resources from a resource library, and if that fails, will end with an error indicating that the resource was not found.

Chapter 2. Adobe software

This section contains the latest information about the Adobe software that can be used with OnDemand.

- Adobe Acrobat. Licensed software to view and create PDF files. You can use Adobe Acrobat to do the following:
  - You can integrate Adobe Acrobat with the OnDemand client. If the Data Type of the OnDemand application is PDF, when a user retrieves a PDF document from the system, the client loads the PDF document into the client viewing window. IBM recommends that you purchase Adobe Acrobat for your users that need to view PDF documents.
  - You can integrate Adobe Acrobat with the administrative client. If you plan to use the report wizard or the graphical indexer to process PDF input files, then you must first install Adobe Acrobat on the PC from which you plan to run the administrative client.
  - You can use Adobe Acrobat to generate PDF files. If you need to create PDF input files for the OnDemand PDF Indexer, a separately priced feature of OnDemand, then you should obtain the Acrobat software from Adobe.

  Important: OnDemand provides the `ARSPDF32.API` file to enable PDF viewing from the clients. If you install the clients after you install Adobe Acrobat, then the installation program will copy the API file to the Acrobat plug-in directory. If you install the clients before you install Adobe Acrobat, then you must copy the API file to the Acrobat plug-in directory. Also, if you upgrade to a new version of Acrobat, then you must copy the API file to the new Acrobat plug-in directory. The default location of the API file is `\Program Files\IBM\OnDemand32\PDF`. The default Acrobat plug-in directory is `\Program Files\Adobe\Acrobat x.y\Acrobat\Plug_ins`, where `x.y` is the version of Acrobat, for example, `4.0`, `5.0`, and so forth.
- Adobe Type Manager (ATM). Licensed software to manage Adobe Type 1 fonts. If your users need to view AFP documents that use the IBM Core Outline Fonts or the Sonoran Metric Outline Fonts that are provided with OnDemand, then they will need ATM installed on their systems. On Windows® 2000 systems, ATM is part of the base operating system. If you plan to use the OnDemand Windows client with other operating systems, then you should obtain the appropriate version of the ATM software from Adobe.

- Adobe Reader. Free of charge software (available for download from the Adobe Web site) to view PDF documents. The OnDemand client can start Reader if the Data Type of the OnDemand application is User Defined and you specify a File Type of PDF and associate the PDF file type with Reader on the client operating system.

You can find out more about Adobe software from Adobe on the Web at:

```
http://www.adobe.com
```

Chapter 3. ARS.CFG

For UNIX® servers, the following parameters have been changed in or added to the ARS.CFG file in Version 7.1.

ARS_DB_ENGINE
Use to specify the database manager. See **DB_ENGINE**.

ARS_MESSAGE_OF_THE_DAY
Use to show the message of the day. Set to the full path name of a file that contains the message that you want to show. For example:

```
ARS_MESSAGE_OF_THE_DAY=/opt/ondemand/tmp/message.txt
```

The contents of the message file can contain a maximum of 1024 characters of text. The administrative client and the Windows client show the message after the user logs on to the server. To close the message box and continue, the user must click OK. If you do not specify a message file, then the normal client processing occurs.

ARS_NUM_DBSRVR
The default value is now 4 (four).

ARS_ORACLE_HOME
Use to specify the base installation directory for Oracle. The default value is:

```
ARS_ORACLE_HOME=/oracle
```

Replace the string `/oracle` with the name of the base installation directory for Oracle.

ARS_STORAGE_MANAGER
A new option **TSM** has been added. You should specify `ARS_STORAGE_MANAGER=TSM` to link the server program to an archive storage manager. The **ADSM** option will continue to be supported for existing customers.

DB_ENGINE
A new parameter **ARS_DB_ENGINE** has been added to specify the database manager. The **DB_ENGINE** parameter will continue to be supported for existing customers.

Chapter 4. ARSADMIN program

The –h parameter is now required for all functions except compress and decompress.

Chapter 5. DB2® UDB Version 7.1 information

Note: The latest update to DB2 UDB Version 7.1 is Fix Pack 2 (two), released in February 2001.

Windows servers

This section lists the changes to DB2 information in *Installation and Configuration Guide for Windows Servers*, GC27–0835–00.

1. Chapter 5. OnDemand system administrator account. This is the user account that you will use to install DB2. Grant the following user rights to the OnDemand system administrator account in Windows:
   - Act as part of the operating system
   - Create a token object
   - Increase quotas
   - Log on as a service
   - Replace a process level token
2. Chapter 7. Installing DB2. To install DB2 on the library server, follow the instructions in *DB2 Universal Database® for Windows V7 Quick Beginnings*, GC09–2971–00 to perform a typical installation and verify the installation.

Chapter 6. Hardware and software requirements

This section lists the changes and additions to the hardware and software requirements listed in *Introduction and Planning Guide*, GC27–0839–00, *Installation and Configuration Guide for UNIX Servers*, GC27–0834–00, and *Installation and Configuration Guide for Windows Servers*, GC27–0835–00.

1. OnDemand supports the IBM @server pSeries and IBM @server xSeries product lines.
2. For Oracle users, OnDemand now requires Oracle 8i Release 3 (8.1.7.0) or later. See "Chapter 9. Oracle information" on page 25 for more information about using Oracle with OnDemand.
3. OnDemand now requires Sun Solaris Version 8 or later on Sun servers.
4. IBM no longer distributes or supports the OS/2® client or supports the Windows client on Windows 95.
5. For the Windows client and the administrative client, OnDemand now requires Microsoft® Windows NT® Version 4.0 SP5 or later, Windows 98, or Windows 2000.
6. For Windows NT servers, OnDemand now requires Microsoft Windows NT Server Version 4.0 SP5 or later.
7. The OnDemand CICS/ESA client now requires OS/390® Version 2 Release 6 and CICS/ESA Version 4.1.0.
8. The OnDemand PDF Indexer feature is included in the OnDemand media pack. However, customers are not authorized to install or use the OnDemand PDF Indexer feature that is included in the OnDemand media pack unless they first purchase the appropriate Proofs of Entitlement for the feature.
9. New customers should use Adobe Acrobat Version 5 to view PDF documents from the client. See "Chapter 2. Adobe software" on page 5 for more information.
10. If you plan to use the report wizard or the graphical indexer to process PDF input files, then you must first install Adobe Acrobat. See "Chapter 2. Adobe software" on page 5 for more information.

Chapter 7. Migrating from previous versions of OnDemand

If you are migrating your OnDemand system to Version 7.1, then you should backup your databases and other files that are critical to the operation of the system before you install OnDemand Version 7.1. See "Preparing databases for migration" on page 16 and "Preparing archive storage for migration" on page 16 for more information.

Important: You can migrate directly from Version 2.2.1.0 or later to Version 7.1. If you are running a version earlier than Version 2.2.1.0, then:

- For UNIX servers, you must upgrade to Version 2.2.1.0 before you migrate to Version 7.1
- For Windows servers, you must first upgrade to Version 2.2.0.15 and then upgrade to Version 2.2.1.0 before you migrate to Version 7.1

For information about upgrading to Version 2.2.1.0, see the README file that is provided with Version 2.2.1.0. You can get the README file from IBM service on the Web at:

`ftp://service.software.ibm.com/software/ondemand/fixes/v221`

The enhancements included in Version 7.1 require changes to the OnDemand system tables (the database). "Migrating from a previous version" on page 16 describes the steps that you must complete to migrate from a previous version of OnDemand.

OnDemand has always provided backward compatibility between clients and servers. However, please note the following:

- The library server and the object servers must always use the same version of the product.
- The administrative client and the library server should always use the same version of the product.
- You cannot use an administrative client prior to Version 7.1 to administer a Version 7.1 server.
- When a new version of the client and the server contains new function, you should not use the new function unless you use the same (new) version of both the client and the server. For example, if you plan to use the new PNG data type, then you must use both the Version 7.1 or later client and the Version 7.1 or later server.

Preparing databases for migration

Follow the instructions in the information that is provided with the database management product that you are using with OnDemand to prepare the OnDemand database for migration. For example, if you are using DB2, see the *Quick Beginnings* publication for your library server operating system. See the section titled Migrating from a previous release of DB2.

In general, IBM recommends that you create a full offline backup of the database on removable media. No processes or users should be connected to the database, except for the backup task. Save the backup copy in a safe location. You will need the backup copy if you want to return to the previous version.

Preparing archive storage for migration

Follow the instructions in the information that is provided with the archive storage management product that you are using with OnDemand to prepare the archive storage manager database for migration. For example, if you are using TSM, see the *Quick Start* publication for the TSM server operating system. See the section titled Migrating from a previous version of TSM.

In general, IBM recommends that you create a full backup of the archive storage manager database on removable media. Save the backup copy in a safe location. You will need the backup copy if you want to return to the previous version.

You should also save copies of the archive storage manager configuration and operation files. For example, if you are using TSM, then save the following files: `DSMSERV.DSM`, `DSMSERV.OPT`, `DSM.OPT`, `DSM.DB2.OPT`, the device configuration file, the volume history file, and any scripts that you may have written. You should also save information about the archive storage devices that are attached to the system (for example, in AIX, save the output from `lsdev –Cc tape` and `lsdev –Cc library`).

Migrating from a previous version

This section describes how to migrate from Version 2.2.1 to Version 7.1. If you need to migrate more than one instance, then you must repeat the steps for each instance. If your OnDemand system contains servers on more than one physical node or workstation, then you must repeat the steps on each node or workstation.

1. Verify that the server meets all of the hardware, software, and memory requirements to install OnDemand. For more information, see “Chapter 6. Hardware and software requirements” on page 13 and the *Introduction and Planning Guide*, GC27–0839–00.

2. Backup databases. See “Preparing databases for migration” on page 16 and “Preparing archive storage for migration” on page 16.
3. If you are migrating a UNIX server, then make a copy of the following files: ARS.CFG, ARS.INI, ARS.CACHE, ARS.DBFS, ARSLOAD.CFG, and ARS_ADSM and any user exit programs and scripts that you may have written. If you are migrating a Windows server, then you should backup the Registry and save any user exit programs that you may have written.
4. Make a backup copy of the following files: ARSLOG, ARSPRT.
5. If required, upgrade and configure the database manager software. See your database manager product information for details
6. If required, upgrade and configure the archive storage manager software. See your archive storage manager product information for details.
7. If you are migrating a UNIX server, then run the ARSLINK program to delete the links to the Version 2.2.1 server programs. First, change to the OnDemand configuration file directory. (The configuration file directory is /usr/lpp/ars/config in AIX and /opt/ondemand/config in HP-UX and Sun Solaris.) Then run the following command:

   ```
   arslink -u
   ```

   Note: The ARSLINK program no longer exists in Version 7.1; the bin/srvr directory no longer exists and there are no longer symbolic links to the ARSADMIN, ARSMAINT, ARSOBJD, ARSSOCKD, and ARSTBLSP programs.
8. Remove any OnDemand Version 2.2.1 software from the server. The uninstall process that you use will vary, depending on the operating system of the server. However, you need to delete any prior versions of the client and the server software from the system before you continue.
9. Install the OnDemand Version 7.1 software. See the *Installation and Configuration Guide* for details.
10. If you are migrating a UNIX server, then configure the following files: ARS.CFG, ARS.INI, ARS.CACHE, ARS.DBFS, ARSLOAD.CFG, and ARS_ADSM. Use the information from the files that you saved in step 3.
11. Reconfigure any scripts and recompile any user exit programs that you may have written for previous versions to work with Version 7.1. Use the information from the files that you saved in step 3.
12. Configure the following files: ARSLOG, ARSPRT. Use the information from the files that you saved in step 4.
13. Verify initialization processes and scheduled tasks. For example, you may need to modify /etc/inittab and crontab.
14. Upgrade the OnDemand system tables. Complete the following steps:

    __ Step a. Open a window with a command prompt.

__ Step b. Create a temporary directory. Create the directory on a drive that has at least 100 MB of free space. (The exact amount of temporary space required during the migration will depend on the number of user-defined application groups and folders and the number of annotations that you have stored in the system.)

__ Step c. Make the temporary directory the current directory.

__ Step d. For Windows servers that use SQL Server, complete the following steps:

1) Run the ARSDB program to drop the configuration indexes. First, start the OnDemand for WinNT Command Window. Then enter the following command at the prompt:

```
arsdb -ev
```

2) Use the ISQL tool to alter the OnDemand tables. You can run the ISQL tool interactively and enter the SQL statements to alter the tables at the prompt or you can run the ISQL tool and specify the name of an input file that contains the SQL statements.

   To enter the SQL statements at the prompt, first enter the ISQL command:

```
isql -E
```

   Then, at the prompt, enter the SQL statements shown in Figure 1 on page 19. Replace the string `ondemand_database` in the USE statement with the name of the OnDemand database in SQL Server (for example, `ARCHIVE`). Replace the string `instance_owner` in each of the ALTER TABLE statements with the name of the instance owner of the OnDemand database in SQL Server (for example, `ODADMIN`).

```
USE ondemand_database

GO

ALTER TABLE instance_owner.arsag ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arsag ADD upd_date BIGINT NULL
ALTER TABLE instance_owner.arsag ADD last_doc_date BIGINT NULL
ALTER TABLE instance_owner.arsag ADD migr_srvr_str VARCHAR(254) NULL

GO
ALTER TABLE instance_owner.arsann ADD table_name VARCHAR(18) NULL
ALTER TABLE instance_owner.arsann ADD doc_exp_date BIGINT NULL

GO
ALTER TABLE instance-owner.arsapp ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arsapp ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arsfol ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance-owner.arsfol ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arsgroup ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arsgroup ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arsnode ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arsnode ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arsprt ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arsprt ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arsres ADD add_date BIGINT NULL

GO
```

Figure 1. Altering the OnDemand Tables in SQL Server (Part 1 of 2)

```
ALTER TABLE instance_owner.arsset ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arsset ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arssys ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arssys ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arsuser ADD email VARCHAR(254) NULL
ALTER TABLE instance_owner.arsuser ADD upd_userid VARCHAR(128) NULL
ALTER TABLE instance_owner.arsuser ADD upd_date BIGINT NULL

GO

ALTER TABLE instance_owner.arsann ALTER COLUMN time_stamp BIGINT NOT NULL

GO

ALTER TABLE instance_owner.arsfolfldusr ALTER COLUMN min_int BIGINT NOT NULL
ALTER TABLE instance_owner.arsfolfldusr ALTER COLUMN max_int BIGINT NOT NULL

GO

ALTER TABLE instance_owner.arsload ALTER COLUMN start BIGINT NOT NULL
ALTER TABLE instance_owner.arsload ALTER COLUMN stop BIGINT NOT NULL
ALTER TABLE instance_owner.arsload ALTER COLUMN exp_date BIGINT NOT NULL

GO

ALTER TABLE instance_owner.arsseg ALTER COLUMN start_date BIGINT NOT NULL
ALTER TABLE instance_owner.arsseg ALTER COLUMN stop_date BIGINT NOT NULL
ALTER TABLE instance_owner.arsseg ALTER COLUMN post_date BIGINT NOT NULL
ALTER TABLE instance_owner.arsseg ALTER COLUMN closed_date BIGINT NOT NULL
ALTER TABLE instance_owner.arsseg ALTER COLUMN reimported_date BIGINT NOT NULL
ALTER TABLE instance_owner.arsseg ALTER COLUMN last_update BIGINT NOT NULL
ALTER TABLE instance_owner.arsseg ALTER COLUMN last_backup BIGINT NOT NULL
ALTER TABLE instance_owner.arsseg ALTER COLUMN last_stats BIGINT NOT NULL

GO

ALTER TABLE instance_owner.arssys ALTER COLUMN time_out BIGINT NOT NULL

GO

ALTER TABLE instance_owner.arsuser ALTER COLUMN last_update BIGINT NOT NULL

GO
```

Figure 1. Altering the OnDemand Tables in SQL Server (Part 2 of 2)

To specify the name of a file that contains the SQL statements, enter:

```
isql.exe -E -i sqlserver.txt
```

Where `sqlserver.txt` is the name of the file that contains the SQL statements. See Figure 1 on page 19 and remember to replace the string `ondemand_database` in the USE statement with the name of the OnDemand database in SQL Server (for example, `ARCHIVE`) and replace the string `instance_owner` in each of the ALTER TABLE statements with the name of the instance owner of the OnDemand database in SQL Server (for example, `ODADMIN`).

__ Step e. Export the OnDemand system tables using the `arsdb` program.

Note: In the steps that follow, replace the string `../ondemand/bin` with the name of the OnDemand program directory on the system:

AIX	/usr/lpp/ars/bin
HP-UX	/opt/ondemand/bin
Sun Solaris	/opt/ondemand/bin
Windows	\Program Files\IBM\OnDemand for Windows NT\bin

If the database manager is DB2 or SQL Server, then enter:

```
../ondemand/bin/arsdb -lxv
```

If the database manager is Oracle, then enter:

```
../ondemand/bin/arsdb -xv
```

__ Step f. Drop the old tables:

```
../ondemand/bin/arsdb -dv
```

__ Step g. Create the new tables and indexes:

```
../ondemand/bin/arsdb -rtv
```

__ Step h. Import the old table information into the new tables. If the database manager is DB2 or SQL Server, then enter:

```
../ondemand/bin/arsdb -ilv
```

If the database manager is Oracle, then enter:

```
../ondemand/bin/arsdb -iv
```

__ Step i. Last, run maintenance on the new tables. If the database manager is DB2 or Oracle, then enter:

```
/opt/ondemand/bin/arsdb -mv
```

If the database manager is Oracle, then enter:

```
../ondemand/bin/arsdb -sv
```

15. Verify your OnDemand installation. You may need to restart your system before you verify the installation.
16. Optionally, create a full offline backup image of the database.

Chapter 8. Naming rules

This section lists the changes to the rules for naming objects in OnDemand, found in the *Introduction and Planning Guide*, GC27–0839–00.

- User ID. Can contain one to 128 characters.
- Password. Can contain one to 128 characters. However, the password authentication that is built into OnDemand verifies only the first eight characters that are entered by the user. The additional characters are provided for customers who choose to implement their own password security by enabling the logon user exit. Contact the IBM support center for more information about the logon user exit.

  Note: If you enable the logon user exit, you should set the Minimum Password Length option to Permit Blank Password so that OnDemand security does not validate passwords that are entered by your users (when they set or change a password). Also, OnDemand security ignores the Maximum Password Age option when you enable the logon user exit.
- Group. Can contain one to 128 characters.
- Server Queue. Can contain one to 60 characters.
- Storage Node Logon. Can contain one to 128 characters.
- Storage Node Password. Can contain one to 128 characters.

Chapter 9. Oracle information

This section provides additional information about using Oracle with OnDemand. This section also includes corrections to the Oracle information that is contained in the OnDemand product publications. Please make a note of these items, which supersede the information in the OnDemand product publications.

Field types and field sizes

Table 1 lists the types of database fields supported by OnDemand and the number of bytes required to hold a value in each type of field. For decimal fields, the actual field size will vary, depending on the average precision for each column. IBM recommends that you read your Oracle information about the NUMBER datatype.

Table 1. Database Field Types and Sizes

OnDemand Field Type	SQL Type	Oracle Datatype	Length
Small Integer	SQL_SMALLINT	NUMBER	7 bytes
Integer	SQL_INT	NUMBER	11 bytes
Decimal	SQL_FLOAT	NUMBER	4 – 22 bytes (maximum precision is 32)[1]
String (Fixed)	SQL_CHAR	CHAR	1 – 254; 1 byte per character declared, even if partially used
String (Variable)	SQL_VARCHAR	VARCHAR2	1 – 254; 1 byte per character plus 4 bytes overhead; unused characters do not consume storage

1. Oracle stores decimal values in a variable length format. Each value is stored in scientific notation, with one byte used to store the exponent and up to twenty bytes to store the mantissa. The resulting value is limited to 32 digits of precision. Oracle does not store leading and trailing zeros. For example, the number `412.50` is stored in a format similar to `4.125 * 100`, with one byte used to store the exponent (`2`) and two bytes used to store the three significant digits of the mantissa (`4, 1, 2`). Taking this into account, the column data size for a particular decimal value `NUMBER (p)`, where `p` is the precision of a given value (scale has no effect), can be calculated using the formula: `1 + FLOOR( p/2 ) + 2`. Therefore, the system requires a minimum of four bytes to hold a decimal value.

Table 1. Database Field Types and Sizes (continued)

OnDemand Field Type	SQL Type	Oracle Datatype	Length
Date	SQL_SMALLINT	NUMBER	7 bytes
Time	SQL_INT	NUMBER	11 bytes
Date/Time	SQL_INT	NUMBER	11 bytes
Date/Time (TZ)	SQL_INT	NUMBER	11 bytes

Estimating database storage space

Estimating the size of database objects is an imprecise undertaking. Overhead caused by disk fragmentation, free space, and the use of variable length fields (including numbers) make size estimation difficult, because there is such a wide range of possibilities for field types and row lengths. After initially estimating your database size, you should create a test database and populate it with representative data.

Figure 2 shows a formula that you can use to estimate disk storage space for the database. You should use the formula to estimate the amount of database storage space for each of the reports that you plan to load on the system. You can use the formula for reports that contain logical items, such as statements and policies, and for reports that contain sorted transaction data.

Note: The formula was derived in part from information provided by Oracle. For more information, or if you have special requirements or if you need to do more, see the Oracle product information. Also, the formula does not include space requirements related to file management overhead required by the operating system, including file block size and directory control space.

```
TableSize = ( Sum of column lengths ) + 3 + ( Number of columns * 2 )

IndexSize = ( Index 1 length + 8 ) + ( Index 2 length + 8 ) + ...

DatabaseSize = ( ( TableSize + 40 ) * 1.2 ) + ( IndexSize * 1.2 )
                   * Number of indexed items per month
                        * Number of months to keep index in database
```

Figure 2. Formula to Estimate Database Storage Space

- `Index n length` is the size of a database field for which you want OnDemand to build an index. For example, an integer field requires four bytes to hold the index value. Oracle requires an additional eight bytes for each index that you define.

- OnDemand adds approximately 40 bytes of control information to each row in a table.
- The formula includes a 20 percent buffer for overhead.
- When the report contains logical items, the Number of indexed items per month is the number of statements, policies, and so forth.
- When the report contains sorted transaction data, the Number of indexed items per month is the number of groups of indexed pages. By default, the system indexes a report in groups of 100 pages. You can specify the size of an indexed group of pages when you index a report with ACIF.

The following example shows how to estimate the database storage space required for a report that contains logical items, such as statements. The example is for indexing one million items per month and keeping the index data in the database for 24 months. Table 2 lists the database fields used in the example. Approximately 3.75 GB of disk space is required to maintain the index data in the database.

```
TableSize = ( 59 + 3 + ( 4 * 2 ) ) = 70

IndexSize = ( 12 + 8 ) ) = 20

DatabaseSize = ( ( 70 + 40 ) * 1.2 ) + ( 20 * 1.2 ) ) = 156
                 * 1000000 = 156000000
                   * 24  =  3744000000
```

Figure 3. Estimating Database Storage Space — Part I

Table 2. Estimating Database Storage Space — Part II

Field Name	Field Type	Field Size	Index or Filter
Report Date	Date	7 bytes	Filter
Account Number	Fixed String	12 bytes	Index
Invoice Balance	Decimal	9 bytes	Filter
Customer Name	Fixed String	31 bytes	Filter

Estimating the size of rollback segments

The database storage space requirements for your OnDemand system include a rollback segment, an area on your disk subsystem that is used when OnDemand makes changes to the database. The size of the rollback segment should be based on the types of transactions that run against the database.

In general, there are two types of transactions in OnDemand: loading data into the system and deleting data from the system. Loading data into the

system is a batch job of (usually) long running transactions. The OnDemand actions that load data into the system include the ARSLOAD program and the ARSADMIN LOAD command. Deleting data from the system can be either a long running transaction (for example, when deleting an entire report from the system) or a short transaction (for example, when deleting a document from the system). The OnDemand actions that delete data from the system include the ARSADMIN UNLOAD command, the ARSDOC DELETE command, and the ARSMAINT program. Data is deleted by the ARSMAINT program when you set an expiration type in your application groups.

Because of this mix of transaction sizes, most customers should plan a large rollback segment that can handle transactions of any size. Most customers should plan to allocate enough storage space for a rollback segment that can hold the transactions for the largest input file that will be loaded into the system.

To estimate the size of the rollback segment, you need to consider three factors:

1. The number of documents that are loaded into the system during the single largest load process.
2. The number of bytes that are allocated to the user-defined database fields for the application group that is associated with the single largest load process.
3. The 40 bytes of system information that OnDemand adds to each database row.

Once you know these values, you can use the formula shown in Figure 4 to estimate the size of the rollback segment that is required for your system.

Note: The formula was derived in part from information provided by Oracle. For more information, or if you have special requirements or if you need to do more, see the Oracle product information. Also, the formula does not include space requirements related to file management overhead required by the operating system, including file block size and directory control space.

```
( Number of documents in largest load *
    ( Number of bytes in application group + 40 OnDemand overhead bytes ) )
        * 2 = estimated size of rollback segment
```

Figure 4. Formula for Estimating the Size of the Rollback Segment

For example, suppose that your largest OnDemand load is for a statement application that loads 150,000 statements in a single load file. The OnDemand application group database fields require approximately 50 bytes of database

storage per document. Figure 5 shows the example calculation, which requires approximately 27 MB of rollback segment space.

```
( 150,000 statements *
       ( 50 bytes in application group + 40 OnDemand overhead bytes ) )
             * 2 = 27,000,000
```

Figure 5. Example of Estimating the Size of the Rollback Segment

If you expect to delete data from the system by using the ARSDOC DELETE command (this is a somewhat unusual requirement), then instead of using the size of the single largest load file, you should substitute the largest number of records that you expect to delete during a delete process.

Tuning the database

In general, most customers use OnDemand in one of two ways:

- Long-term archive for larger reports. These customers create table spaces for each application group that they add to the system, load many rows into the database at one time, maintain data on the system for many months or years, and delete a report at a time from the system. These parameters usually result in very static data and a low maintenance operation.
  - A table space contains data from one application group.
  - Inserts are done by a high-volume batch process.
  - After a table reaches its Maximum Rows value, OnDemand closes the table and no additional inserts are made to the table. Closed tables remain available for queries until the data is removed from the system.
  - Data is removed from the system by dropping a table or deleting a large number of consecutive rows at a time.

  Because of the low rate of change within the tables, these customers should seldom or never need to tune their database.
- Short-term archive for smaller reports. These customers store all application group data in the SYSTEM table space (or in one or more DATA table spaces), load very few rows into the database at one time, maintain data on the system for a short period of time, or use the Delete Document method to remove data from the system. These parameters usually result in very dynamic data and a high maintenance operation.
  - A table space contains data from more than one application group.
  - Inserts are done by a low-volume batch process.
  - Inserts and deletes happen frequently.
  - Data is removed from the system by deleting one row at a time.

Because of the high rate of change within the tables, these customers should plan to tune their database on a time-based schedule, such as a weekly or nightly process. **Note:** Some customers tune their database every day. However, most customers do not tune their database until the optimizer ignores the current set of statistics or generates an inefficient plan. IBM recommends that these customers tune the database just before they plan to take an offline backup of the database. This schedule keeps the optimization information up to date and minimizes the impact to system availability, because a system outage is already planned.

Tuning the database is done by collecting statistics on the tables, which can provide faster access to the data, thereby improving performance. Statistics on tables are gathered by using the ANALYZE command. When you analyze a table, its associated indexes are automatically analyzed as well. The frequency with which you analyze the tables depends on the rate of change within the tables. **Note:** If you collect statistics and do not notice a visible performance improvement, then dropping and recreating the indexes to your tables may help. Customers in a high maintenance operation may need to periodically rebuild the indexes on their most active tables. See your Oracle information for details about rebuilding indexes.

OnDemand provides two programs to collect statistics on database tables: the ARSDB program and the ARSMAINT program.

Using the ARSDB program

You can use the ARSDB program to collect statistics on the OnDemand system tables, such as the user table, the group table, the application group table, and so forth. For most customers, the OnDemand system tables require very little maintenance. You can probably schedule the ARSDB program to collect statistics once a month (or less often).

The syntax is:

```
/opt/ondemand/bin/arsdb <options>
```

The options are:

–e Drop configuration indexes

–r Create configuration indexes

–s Collect statistics

Using the ARSMAINT program

You can use the ARSMAINT program to maintain the tables that contain user-defined application group data. User-defined application groups are the application groups that you define to the system. Customers in a high maintenance operation should run the ARSMAINT program on a regular schedule.

The syntax is:

```
/opt/ondemand/bin/arsmaint <options>
```

The options are:

–d, –i Expire index data from the database. The **–i** parameter expires index data that has been imported from archive storage. If you do not migrate index data to archive storage, then you do not need to specify the **–i** parameter.

–e Migrate index data from the database to archive storage. If you do not migrate index data to archive storage, then you do not need to specify the **–e** parameter.

–r Collect statistics. **Note:** The ARSMAINT program collects statistics only on the tables that have changed since the last time that statistics were collected. OnDemand keeps information about all of its tables, including the last time that it modified a table and the last time that it collected statistics on a table.

–g applGroup
Process the tables for the specified application group. If you do not specify this parameter and name an application group, then the ARSMAINT program processes all of the user-defined application groups.

Chapter 10. PDF indexing

Note: The OnDemand PDF Indexer feature is included in the OnDemand media pack. However, customers are not authorized to install or use the OnDemand PDF Indexer feature that is included in the OnDemand media pack unless they first purchase the appropriate Proofs of Entitlement for the feature.

Overview

This section describes how to use the graphical indexer to create indexing information for PDF input files.

Important: If you plan to use the report wizard or the graphical indexer to process PDF input files, then you must first install Adobe Acrobat on the PC from which you plan to run the administrative client. OnDemand provides the `ARSPDF32.API` file to enable PDF viewing from the client. If you install the client after you install Adobe Acrobat, then the installation program will copy the API file to the Acrobat plug-in directory. If you install the client before you install Adobe Acrobat, then you must copy the API file to the Acrobat plug-in directory. Also, if you upgrade to a new version of Acrobat, then you must copy the API file to the new Acrobat plug-in directory. The default location of the API file is `\Program Files\IBM\OnDemand32\PDF`. The default Acrobat plug-in directory is `\Program Files\Adobe\Acrobat x.y\Acrobat\Plug_ins`, where `x.y` is the version of Acrobat, for example, `4.0`, `5.0`, and so forth.

Beginning with Version 7.1, you can define indexing information in a visual environment. You begin by opening a sample input file with the graphical indexer. You can run the graphical indexer from the report wizard or by choosing the sample data option from the Indexing Information page of the application. After you open an input file in the graphical indexer, you define triggers, fields, and indexes. The PDF indexer uses the triggers, fields, and indexes to locate the beginning of a document in the input data and extract index values from the input data. Once you have defined the triggers, fields, and indexes, you can save them in the application so that OnDemand can use them later on to process the input files that you load into the system.

You define a trigger, field, or index by drawing a box around a text string with the mouse and then specifying properties. For example, to define a trigger that identifies the beginning of a document, you could draw a box around the text string `Account Number` on the first page of a statement in the

input file. Then, on the Add a Trigger dialog box, you would accept the default values provided, such as the location of the text string on the page. When processing an input file, the PDF indexer attempts to locate the specified string in the specified location. When a match occurs, the PDF indexer knows that it has found the beginning of a document. The fields and indexes are based on the location of the trigger.

The PDF file that you open with the graphical indexer should contain a representative sample of the type of input data that you plan to load into the system. For example, the sample input file must contain at least one document. A good sample should contain several documents so that you can verify the location of the triggers, fields, and indexes on more than one document. The sample input file must contain the information that you need to identify the beginning of a document in the input file. The sample input file should also contain the information that you need to define the indexes. When you load an input file into the system, the PDF indexer will use the indexing information that you create to locate and extract index values for each document in the input file.

Using the graphical indexer

The following example describes how to use the graphical indexer from the report wizard to create indexing information for an input file. The indexing information consists of a trigger that uniquely identifies the beginning of a document in the input file and the fields and indexes for each document.

1. To begin, start the administrative client.
2. Log on to a server.
3. Start the report wizard by clicking the Report Wizard icon on the toolbar. The report wizard opens the Sample Data dialog box.
4. Click Select Sample Data to open the Open dialog box.
5. Type the name or full path name of a file in the space provided or use the Look in or Browse commands to locate a file.
6. Click Open. The graphical indexer opens the input file in the report window.
7. Press F1 to open the main help topic for the report window. The main help topic contains general information about the report window and contains links to other topics that describe how to add triggers, fields, and indexes. Under Options and Commands, click Indexer Information page to open the Indexing Commands topic. (You can also use the content help tool to display information about the icons on the toolbar.) Under Tasks, Indexer Information page, click Adding a trigger (PDF).
8. Close any open help topics and return to the report window.
9. Define a trigger.

- Find a text string that uniquely identifies the beginning of a document. For example, `Account Number`, `Invoice Number`, `Customer Name`, and so forth.
- Using the mouse, draw a box around the text string. Start just outside of the upper left corner of the string. Click and hold mouse button one. Drag the mouse towards the lower right corner of the string. As you drag the mouse, the graphical indexer uses a dotted line to draw a box. When you have enclosed the text string completely inside of a box, release the mouse button. The graphical indexer highlights the text string inside of a box.
- Click the Define a Trigger icon on the toolbar to open the Add a Trigger dialog box. Verify the attributes of the trigger. For example, the text string that you selected in the report window should be displayed under Value; for Trigger1, the Pages to Search should be set to Every Page. Click Help for assistance with the other options and values that you can specify.
- Click OK to define the trigger.
- To verify that the trigger uniquely identifies the beginning of a document, first put the report window in display mode. Then click the Select tool to open the Select dialog box. Under Triggers, double click the trigger. The graphical indexer highlights the text string in the current document. Double click the trigger again. The graphical indexer should highlight the text string on the first page of the next document. Use the Select dialog box to move forward to the first page of each document and return to the first document in the input file.
- Put the report window in add mode.

10. Define a field and an index.
    - Find a text string that can be used to identify the location of the field. The text string should contain a sample index value. For example, if you want to extract account number values from the input file, then find where the account number is printed on the page.
    - Using the mouse, draw a box around the text string. Start just outside of the upper left corner of the string. Click and hold mouse button one. Drag the mouse towards the lower right corner of the string. As you drag the mouse, the graphical indexer uses a dotted line to draw a box. When you have enclosed the text string completely inside of a box, release the mouse button. The graphical indexer highlights the text string inside of a box.
    - Click the Define a Field icon on the toolbar to open the Add a Field dialog box.
    - On the Field Information page, verify the attributes of the index field. For example, the text string that you selected in the report window should be displayed under Reference String; the Trigger should

identify the trigger on which the field is based. Click Help for assistance with the options and values that you can specify.

- On the Database Field Attributes page, verify the attributes of the database field. In the Database Field Name space, enter the name of the application group field into which you want OnDemand to store the index value. In the Folder Field Name space, enter the name of the folder field that will appear on the client search screen. Click Help for assistance with the other options and values that you can specify.
- Click OK to define the field and index.
- To verify the locations of the fields, first put the report window in display mode. The fields should have a blue box drawn around them. Next, click the Select tool to open the Select dialog box. Under Fields, double-click Field 1. The graphical indexer highlights the text string in the current document. Double click Field 1 again. The graphical indexer should move to the next document and highlight the text string. Use the Select dialog box to move forward to the each document and display the field. Then return to the first document in the input file.
- Put the report window in add mode.

11. Click the Display Indexer Parameters tool to open the Display Indexer Parameters dialog box. The Display Indexer Parameters dialog box lists the indexing parameters that the PDF indexer will use to process the input files that you load into the application. At a minimum, you need one trigger, one field, and one index. See the *Indexing Reference* for details about the indexing parameters.
12. When you have finished defining all of the triggers, fields, and indexes, close the report window.
13. Click Yes to save the changes to the indexer parameters.
14. On the Sample Data window, click Next to continue with the report wizard.

Chapter 11. Publications CD-ROM

The OnDemand publications CD-ROM contains the OnDemand books in HTML and PDF formats. The OnDemand books on the CD-ROM are provided in several languages.

CD-ROM directory structure

The HTML files on the CD-ROM have an extension of HTM. The PDF files on the CD-ROM have an extension of PDF. The files are located in one of the following directories. On UNIX systems:

```
/cdrom/<language>/html/<book>
/cdrom/<language>/pdf
```

On Windows systems:

```
x:\<language>\html\<book>
x:\<language>\pdf
```

Where:

/cdrom
Refers to your mount point (UNIX)

x: Refers to your CD-ROM drive (Windows)

<language>
Refers to the language code, and is named with one of the codes listed in Table 3.

Table 3. Directory Language Identifier

Directory	Language	Directory	Language
ARA	Arabic	FRA	French
CHS	Simplified Chinese	FRC	Canadian French
CHT	Traditional Chinese	JPN	Japanese
DEU	German	KOR	Korean
ENU	English	NLD	Dutch
ESP	Spanish	PTB	Brazilian Portuguese

<book>
Refers to the HTML directory for each book, and named using the five-character identifier assigned to each book (see Table 4 on page 38).

Notes:

1. The directory names may appear in uppercase or lowercase, depending on your operating system.
2. Not all of the OnDemand books that are viewable from the CD-ROM are translated into all of the supported languages. If a particular book is translated into a particular language, then it is in the language directory specific to that language. This means, for example, that the English language books are in the \enu directory on Windows, and in the /enu directory on UNIX, whereas the French books are in the \fra directory on Windows, and in the /fra directory on UNIX. Whenever a book is not available in a specific language, the English book is provided.

OnDemand publications

Table 4 shows the OnDemand books that are on the CD-ROM and the directory name (HTML) for each publication.

Table 4. OnDemand Publications

Title	Directory Name
Introduction and Planning Guide	ars1p
Installation Guide for UNIX Servers	ars1u
Installation Guide for Windows Servers	ars1w
Administrator's Guide	ars1b
Indexing Reference	ars1d
User's Guide	ars5q
Windows Client Customization Guide	ars5u
Web Enablement Kit Installation and Configuration Guide	ars5y

Note: Not all of the books are translated into all of the supported languages. Whenever a book is not available in a specific language, the English book is provided.

Viewing the HTML books

The books included with OnDemand are in HTML soft copy format. The HTML format enables you to browse the information and provides hypertext links to related information. It also makes it easier to share the books across your organization.

You can view the HTML books with any browser that conforms to HTML Version 3.2 specifications.

To view the HTML books:

1. Insert the OnDemand publications CD-ROM into the CD-ROM drive. On UNIX systems, mount the CD-ROM. See your operating system documentation for the mounting procedures.
2. Start the browser.
3. Click File->Open and open the desired book from one of the following locations:
   - On UNIX systems:

     `/cdrom/%L/HTML/<book>` directory on the CD-ROM, where `/cdrom` represents the mount point of the CD-ROM, `%L` represents the three-character country code that represents your language (for example, ENU for English), and `<book>` represents the directory name. See Table 3 on page 37 for a list of the country codes and Table 4 on page 38 for a list of directory names.
   - On Windows systems:

     `x:\language\HTML\<book>` directory, where `x` represents the CD-ROM drive, `language` represents the three-character country code that represents your language (for example, ENU for English), and `<book>` represents the directory name. See Table 3 on page 37 for a list of the country codes and Table 4 on page 38 for a list of directory names.
4. The main HTML file for each book is INDEX.HTM. Load this file into your browser to view the book.

You can also copy the HTML files from the CD-ROM to a local or network drive and browse them from there.

Printing the PDF books

If you prefer to have printed copies of the books, you can print the PDF files found on the OnDemand publications CD-ROM. Using the Adobe Reader, you can print either the entire book or a specific range of pages. For the file name of each book on the CD-ROM, see Table 4 on page 38.

You can obtain the latest version of Adobe Reader from Adobe on the Web at:

`http://www.adobe.com`

The PDF files are included on the OnDemand publications CD-ROM with a file extension of PDF. To access the PDF files:

1. Insert the OnDemand publications CD-ROM into the CD-ROM drive. On UNIX systems, mount the CD-ROM. See your operating system documentation for the mounting procedures.
2. Start the Adobe Reader.
3. Open the desired PDF file from one of the following locations:

- On UNIX systems:

  `/cdrom/doc/%L/PDF` directory on the CD-ROM, where `/cdrom` represents the mount point of the CD-ROM and `%L` represents the three-character country code that represents your language (for example, ENU for English). See Table 3 on page 37 for a list of the country codes.

- On Windows systems:

  `x:\doc\language\PDF` directory, where `x` represents the CD-ROM drive and `language` represents the three-character country code that represents your language (for example, ENU for English). See Table 3 on page 37 for a list of the country codes.

You can also copy the PDF files from the CD-ROM to a local or network drive and print them from there.

Ordering the printed books

You can order the printed OnDemand books from IBM Publications on the Web at:

`http://www.elink.ibmlink.ibm.com/pbl/pbl`

Chapter 12. Support

Support for the OnDemand product is available from the Web. Click **Support** from the product Web site at:

`http://www.ibm.com/software/data/ondemand/`

The IBM support center maintains product updates for OnDemand. You can obtain the latest product updates from IBM service on the Web at:

`ftp://service.software.ibm.com/software/ondemand/fixes/v71`

If you encounter problems or errors running any of the OnDemand programs, you can call the IBM support center to obtain software program and defect support. The phone number for the IBM support center is 1–800–237–5511. The OnDemand program ID is 5697–G34. The component ID for the OnDemand for Multiplatforms Version 7.1 server is 5697G3400. The component ID for the OnDemand Version 7.1 clients is 5697G3401.

Chapter 13. TSM Version 4.1 information

License registration has changed for UNIX servers running TSM Version 4.1.

- You no longer have to register a network license.
- The client and extended device support license names have changed from those documented in item 5 on page 61 of *Installation and Configuration Guide for UNIX Servers*, GC27–0834–00. A client is now a Managed System; devices that required extended (or advanced) device support now require a Managed Library license.
- The following example shows how to register the licenses for ten managed systems (known as client nodes in TSM V3.7) and one managed library for an optical library (that required the extended device support license in TSM V3.7) on a TSM V4.1 server:

  ```
  register license file(10mgsyslan.lic)
  register license file(1library.lic)
  ```

  See the TSM V4.1 *Quick Start* publication for your server operating system for more information about registering licenses.

Chapter 14. Web Enablement Kit

This section describes updates to the OnDemand Web Enablement Kit *Installation and Configuration Guide*, SC27–1000–01.

Xenos transform

Beginning with Version 7.1, you can configure the WEK feature to use the Xenos transform. You can use the Xenos transform to convert AFP and Metacode documents that you retrieve from an OnDemand server into PDF documents and send the converted documents to the Web browser. See "Viewing and printing with the Web Enablement Kit" on page 98 for an overview of using the Xenos transform. See "Chapter 29. Configuring the WEK" on page 125 for details about configuring the various components of the WEK feature to use the Xenos transform.

ADDFIELDSTODOCID parameter

If the `Annotation Flags in document database table` field is set to Yes, then you must specify ADDFIELDSTODOCID=1. You can set the `Annotation Flags in document database table` field on the Database Information dialog box, from the General page in application groups (click Advanced to open the Database Information dialog box).

API reference

Add Annotation

The default value for the `_copy` parameter should be `off`, not 0 (zero). To specify that the annotation should remain attached to the document, specify `on`, not 1 (one).

If you specify `_html=*` (asterisk), then ODWEK uses a file named ADDNOTE.HTML, not ADDNOTE.HTM.

Change Password

If you specify `_html=*` (asterisk), then ODWEK uses a file named CHGPASSWORD.HTML, not CHGPASSWORD.HTM.

Document Hit List

If you specify `_html=*` (asterisk), then ODWEK uses a file named DOCHITLIST.HTML, not DOCHITLIST.HTM.

Logoff

If you specify _html=* (asterisk), then ODWEK uses a file named LOGOFF.HTML, not LOGOFF.HTM.

Logon

If you specify _html=* (asterisk), then ODWEK uses a file named LOGON.HTML, not LOGON.HTM.

Print Document

If you specify _html=* (asterisk), then ODWEK uses a file named PRINTDOC.HTML, not PRINTDOC.HTM.

When the specified printer (_printer) is a FAX or Printer with Information, then you can specify the following additional parameters:

Table 5. Print Document Function — New Parameters

Parameter=Value	Purpose
_recv_name=value	The receiver's name
_recv_comp=value	The receiver's company name
_recv_fax=value	The receiver's FAX number
_send_name=value	The sender's name
_send_comp=value	The sender's company name
_send_tel=value	The sender's phone number
_send_fax=value	The sender's FAX number
_send_cover=value	A user-defined overlay that the Header Page Exit program merges with the values of the other parameters to produce a cover page for the document
_subject=value	A string that represents the subject of the document
_notes=value	A string that represents a note about the document

Retrieve Document

If you specify _html=* (asterisk), then ODWEK uses a file named RETRIEVE.HTML, not RETRIEVE.HTM.

Search Criteria

If you specify _html=* (asterisk), then ODWEK uses a file named SEARCHCRIT.HTML, not SEARCHCRIT.HTM.

Update Document

If you specify `_html=*` (asterisk), then ODWEK uses a file named UPDATE.HTML, not UPDATE.HTM.

View Annotations

If you specify `_html=*` (asterisk), then ODWEK uses a file named GETNOTES.HTML, not GETNOTES.HTM.

AFP to PDF transform

The following parameter can now be specified in the AFP2PDF configuration file, AFP2PDF.INI:

AllObjects

The `AllObjects` parameter determines how the WEK will process documents that are stored as large objects in OnDemand. If you specify `0` (zero), then the WEK will retrieve only the first segment of a document. If you specify `1` (one), then the WEK will retrieve all of the segments and convert them before sending the document to the client. **Note:** If you enable large object support for very large documents, then your users may experience a significant delay before they can view the document at the client.

Chapter 15. Windows client customization

The following methods have been added to the OLE control section of the *Windows Client Customization Guide*.

- GetDocType (see "GetDocType" on page 50)
- GetTypeForDoc (see "GetTypeForDoc" on page 52)

GetDocType

short **GetDocType**(
long **Index**,
variant * **pType**,
lpUnknown **pExtension**)

Parameters

Index
Specifies the zero-based index of a document within the document list of the active folder. If this value is less than zero, then the open document is used.

pType
Points to a variable to receive the document type of the specified document. On return, this variable is set to type VT_I2. The document type will be one of the document type values, such as ARS_OLE_DOC_TYPE_AFP, found in ARSOLEEX.H.

pExtension
Points to a string to receive the file extension of the document. This value is returned only if the document type is ARS_OLE_DOC_TYPE_USER_DEF.

Description

Retrieves the document type. If the document type is ARS_OLE_DOC_TYPE_USER_DEF, then the file extension is also retrieved.

Return Value

Refer to Return Code in the *Windows Client Customization Guide* for an explanation of the return code.

See Also

GetTypeForDoc

Restrictions

This method is intended for use with C/C++.

Examples

The following example retrieves the document type of the third item in the document list.

```
VARIANT type;
char extension[20];
CArsOle * pArsCtrl;
short rc;

.
.
```

```
.

rc = pArsCtrl->GetDocType( 2, &type, extension );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
```

GetTypeForDoc

short **GetTypeForDoc**(
long **Index**,
variant * **pType**,
BSTR * **pExtension**)

Parameters

Index
Specifies the zero-based index of a document within the document list of the active folder. If this value is less than zero, then the open document is used.

pType
Points to a variable to receive the document type of the specified document. On return, this variable is set to type VT_I2. The document type will be one of the document type values, such as ARS_OLE_DOC_TYPE_AFP, found in ARSOLEEX.H.

pExtension
Points to a BSTR to receive the file extension of the document. This value is returned only if the document type is ARS_OLE_DOC_TYPE_USER_DEF.

Description

Retrieves the document type. If the document type is ARS_OLE_DOC_TYPE_USER_DEF, then the file extension is also retrieved.

Return Value

Refer to Return Code in the *Windows Client Customization Guide* for an explanation of the return code.

See Also

GetDocType

Restrictions

This method is intended for use with Visual Basic.

Examples

The following example retrieves the document type of the third item in the document list.

```
Dim rc As Integer
Dim type As Variant
Dim ext As String

.
.
.
```

```
rc = ArsOle.GetTypeForDoc (2, type, ext)
if rc <> ARS_OLE_RC_SUCCESS THEN
  MsgBox "ERROR"
  End
Endif
```

Part 2. System administration

Chapter 16. Overview

Applications, application groups, folders, storage sets, and printers are the objects that represent how OnDemand stores, manages, retrieves, views, and prints reports and index data. Controlling and limiting access to the reports and index data is accomplished by defining users and groups and giving them the level of authority that is required to meet the data security strategy of an organization.

Chapter 17. User types

OnDemand provides the ability to centralize or decentralize the administration of the system. OnDemand also provides the flexibility to control access to objects from different levels. The most basic level of control is how the user is defined to the system. When a user is added, a user type is specified. Each user type has a different level of authority:

System Administrator
A system administrator has the highest level of authority on the system. A system administrator can perform all tasks on all of the objects that are defined to an OnDemand system. The objects are users, groups, applications, application groups, folders, storage sets, and printers. The tasks are add, update, delete, copy, export, create a report, and view properties. A system administrator also has the authority to modify the system parameters.

Application Group/Folder administrator
An application group/folder administrator has the authority to perform all tasks on all of the applications, application groups, and folders that are defined to an OnDemand system. The tasks are add, update, delete, copy, export, create a report, and view properties.

User Administrator
A user administrator has the authority to perform all tasks on all of the users that are defined to an OnDemand system. The tasks are add, update, delete, copy, export, create a report, and view properties. **Note:** A user administrator cannot create or delete a system administrator or an application group/folder administrator or change the user type of a system administrator or an application group/folder administrator.

User A user has the lowest level of authority on the system. A user does not have access to any object on the system and therefore, cannot perform any tasks. The user must be given the authority to access an object and to perform a task on the object.

Chapter 18. Authority

A user's authority can be extended beyond the authority that is built into the user type of the user. Depending on the user type, up to four additional levels of authority can be given:

Create Users

An application group/folder administrator or a user can be given the authority to add users to the system. By default, when a user is added to the system, the user that performs the add task automatically has the authority to perform all other tasks on the user. The tasks are update, delete, generate a report, and view properties. When a user is added, the system automatically gives the user with Create Users authority the permission to access and administer the newly created user. (See "Chapter 19. Permissions" on page 65 for information about permissions at the user level.) If the permissions are later taken away, then the user with Create Users authority no longer has access to the user or the authority to administer the user.

A user with Create Users authority is similar to a user administrator in that both can create users. However, they differ because a user with Create Users authority can access and administer only the users that they create, so long as their access and administrator authority is not taken away. **Note:** A user with Create Users authority cannot create or delete a system administrator, an application group/folder administrator, or a user administrator or change the user type of a system administrator, an application group/folder administrator, or a user administrator.

A user with Create Users authority also has the authority to perform copy and export tasks on a user because both tasks add a user. An export task involves two userids: one on the server where the user exists and one on the server where the user will be added. The userid on the server where the user will be added must have the authority to add a user.

Create Groups

A user administrator, an application group/folder administrator, or a user can be given the authority to add groups to the system. The only other type of user that can add groups is a system administrator

By default, when a group is added to the system, the user with Create Groups authority is designated as the group owner. The group owner has the authority to perform all other tasks on the group. The tasks are update, delete, generate a report, and view properties. If the

owner is changed to a different user or group, then the system automatically takes the authority to perform tasks on the group away from the user that originally created the group.

In general, a user has access to a group if the user is a system administrator, the owner of the group, a member of the group that has been designated as the owner, or a member of the group.

A user with Create Groups authority also has the authority to perform copy and export tasks on a group because both tasks add a group. An export task involves two userids: one on the server where the group exists and one on the server where the group will be added. The userid on the server where the group will be added must have the authority to add a group.

Create Application Groups

A user administrator or a user can be given the authority to add application groups to the system. By default, when an application group is added to the system, the user that performs the add task automatically has the authority to perform all other tasks on the application group. The tasks are update, delete, generate a report, and view properties. When the application group is added, the system automatically gives the user with Create Application Groups authority the permission to access and administer the newly created application group. (See "Chapter 19. Permissions" on page 65 for information about permissions at the application group level.) If the permissions are later taken away, then the user with Create Application Groups authority no longer has access to the application group or the authority to administer the application group.

A user with Create Application Groups authority is similar to an application group/folder administrator in that both can create application groups. However, they differ because a user with Create Application Groups authority can access and administer only those application groups that they create, so long their access and administrator authority is not taken away. An application group/folder administrator can access and administer all of the application groups that are defined to the system.

Because applications are considered by OnDemand to be part of an application group, the permissions for accessing and administering applications are defined by the permission for the application group. For this reason too, a user with Create Application Groups authority can also create applications for the application group.

A user with Create Application Groups authority also has the authority to perform copy and export tasks on an application group because both tasks add an application group. An export operation involves two userids: one on the server where the application group

exists and one on the server where the application group will be added. The userid on the server where the application group will be added must have the authority to add an application group.

Create Folders

A user administrator or a user can be given the authority to add folders to the system. By default, when a folder is added to the system, the user that performs the add task automatically has the authority to perform all of the other tasks on the folder. The tasks are update, delete, generate a report, and view properties. When a folder is added, the system automatically gives the user with Create Folders authority the permission to access and administer the newly created folder. (See "Chapter 19. Permissions" on page 65 for information about permissions at the folder level.) If the permissions are later taken away, then the user with Create Folders authority no longer has access to the folder or the authority to administer the folder.

A user with Create Folders authority is similar to an application group/folder administrator in that both can create folders. However, they differ because a user with Create Folders authority can access and administer only the folders that they create, so long as their access and administrator authority is not taken away. An application group/folder administrator can access and administer all of the folders that are defined to the system.

A user with Create Folders authority also has the authority to perform copy and export tasks on a folder because both tasks add a folder. An export task involves two userids: one on the server where the folder exists and one on the server where the folder will be added. The userid on the server where the folder will be added must have the authority to add a folder.

Chapter 19. Permissions

Access and control of users, groups, applications, application groups, storage sets, folders, and printers can be given at various levels. In “Chapter 17. User types” on page 59, the level of control was determined by the user type of the user. In this section, a different level of control is described. Permissions on an object can be set from the Permissions page of the object. The following topics describe the objects and the permissions that can be set.

Users

A user can be given the authority to view documents that have been archived. A user can also be given the authority to perform administrative tasks on the system. A user can retrieve a list of users from the server with the administrative client. The list contains the users that the user has the authority to access. If the user is a system administrator or a user administrator, then the user also has the authority to administer any user in the list.

Access authority means the user can see the user in any list that contains users and can print or view the properties of the user. Access authority can be given to an individual user or to a group.

Access authority is especially helpful to application group/folder administrators, because they can give any user in the list access to application groups and folders. This is also true for group owners; access authority allows them to add users to the groups that they own.

Administrator authority of a user can be given to another user or to a group. Having administrator authority for a user means that the user or group given the authority can delete or update the user.

Groups

Groups can be created by a system administrator or a user that has Create Groups authority. Updating or deleting a group can be performed by a system administrator or the group owner. A group owner can be a user or another group. Allowing the group owner to be another group provides the ability to allow multiple users to administer the group.

To see a group in a list, a user must be a member of the group, the owner of the group, or a system administrator. For example, an application group/folder administrator can give groups access to application groups and folders. However, the application group/folder administrator must be able to

see the groups in the list on the Permissions page of the application group or folder. This means that the application group/folder administrator must be a member of any group that requires access authority to an application group or a folder.

A system administrator, a group owner, or a member of the group can view the properties of the group and generate and print reports.

Applications

Because applications are considered by OnDemand to be part of an application group, the permissions for accessing and administering applications are defined by the permission for the application group. (See "Application Groups".)

Application Groups

Application Groups can be created by system administrators, application group/ folder administrators, and users with Create Application Groups authority. After an application group is created, only a system administrator, an application group/folder administrator, a user with administrator authority for the application group, or a member of a group that has administrator authority for the application group can update or delete the application group. When a user with Create Application Groups authority creates an application group, the system automatically gives the user the authority to administer the application group. The user can update or delete the application group, so long as the administrator authority is not taken away.

To allow other users to see an application group in a list with the administrative client or to search for documents with the end-user client, the users must be given access authority to the application group. A user with access authority can also print or view the properties of the application group. Access is given on the Permissions page of an application group. There are three ways to give a user access to an application group:

1. Add the user's userid to the access list.
2. Add the name of a group to which the user belongs to the access list. The user and all of the other members of the group will have access to the application group.
3. Set the access permission for the reserved name *PUBLIC. All users on the system will have access to the application group. (The *PUBLIC name is used to set permissions for all users on the system.)

The levels of authority within the application group have a precedence order in which the permissions are enforced. The permissions that have been set for a user take precedence over any permissions that have been set for any

groups that the user may belong to. User permissions also take precedence over permissions that have been set using the *PUBLIC name.

A user can also be given the authority to save a specific set of viewing attributes such as zoom, background color, and so forth. The viewing attributes can be used when a document is viewed with the end-user client. The set of viewing attributes, called a logical view, is accessible only to the user that created the logical view.

Document and Annotation permissions can also be set for users by using the *PUBLIC name, group names, and userids. Document permissions include add, delete, update, view, copy, print, and FAX. Annotation permissions include add, delete, update, view, and copy.

Folders

Folders can be created by system administrators, application group/folder administrators, and users with Create Folders authority. After a folder is created, only a system administrator, an application group/folder administrator, a user with administrator authority for the folder, or a member of a group that has administrator authority for the folder can update or delete the folder. When a user with Create Folders authority creates a folder, the system automatically gives the user the authority to administer the folder. The user can update or delete the folder, so long as the administrator authority is not taken away.

To allow other users to see a folder in a list with the administrative client or to open a folder with the end-user client, the users must be given access authority to the folder. A user with access authority can also print or view the properties of the folder. Access is given on the Permissions page of a folder. There are three ways to give a user access to a folder:

1. Add the user's userid to the access list.
2. Add the name of a group to which the user belongs to the access list. The user and all of the other members of the group will have access to the folder.
3. Set the access permission for the reserved name *PUBLIC. All users on the system will have access to the folder. (The *PUBLIC name is used to set permissions for all users on the system.)

The levels of authority within the folder have a precedence order in which the permissions are enforced. The permissions that have been set for a user take precedence over any permissions that have been set for any groups that the user may belong to. The user permissions also take precedence over permissions that have been set using the *PUBLIC name.

In addition to allowing a user to access or administer a folder, a user can also be given the authority to customize the appearance of the folder search and display fields with the administrative client. The authority can be given to a specific user or to a group. If the authority is given to a group, then any member of the group has the authority to customize the appearance of the search and display fields. Only the authorized user or members of the group see the customized search and display fields with the end-user client. All other users will see the search and display fields that have been defined using the *PUBLIC name in the folder.

A user can also be given the authority to save a specific set of search criteria when using the end-user client. The user can restore the set of search criteria when needed, into the search fields of a folder. The set of search criteria, called a named query, can be made available to all of the users that have access to the folder (a public named query) or it can be made available only to the user that created the named query (a private named query). When giving Named Query authority to a user, the user can be given the authority to view named queries, but not create them; the user can also be given the authority to create public named queries, private named queries, or both. A user can get Named Query authority from a group, if the group has been given Named Query authority for the folder.

Storage sets

Any user on the system can view the properties of a storage set or generate a report about a storage set. However, only a system administrator can add, delete, update, copy, or export a storage set.

Printers

Printers are maintained by system administrators. Only a system administrator can add, delete, update, copy, or export a printer. By default, only system administrators can see printers in a list and therefore, only a system administrator can view the properties of a printer or generate a report about a printer. However, it is almost always necessary for other users to have access to printers. For this reason, a user or group can be given access to a particular printer. A printer's access list is maintained on the Permissions page under printers, by using the administrative client. Access to a printer can be given to all users and groups defined to the system, individual groups, and individual users.

Limiting access to printers provides the ability to control which printers can be used to print archived documents by OnDemand users. For example, suppose that there is a printer in the customer service department. Only people in the customer service department should be permitted to print on

the printer. You can accomplish this by creating an OnDemand group that contains only the department members and giving only that group access to the printer.

Chapter 20. System administration

OnDemand provides the ability to centralize or decentralize the administration of the system. A centralized environment means that one type of user, a system administrator, controls the creation and access to all of the objects defined on the system. A decentralized environment means that the tasks of the system administrator are divided and assigned to other users. The responsibilities of the other users may vary from user administration, group administration, application group administration, folder administration, or any combination of the administrative tasks.

The decision to centralize or decentralize the administration of the system should be made before objects are added to the system. While the decision is reversible, the amount of work required to change from one type of administration to the other can be significant if a large number of users, groups, folders, and application groups have already been added.

There are many ways to decentralize the administration of the system, because of the various user types and the additional authority levels that can be specified for users. Two specific models will be discussed in this section: the Object Type model and the Object Owner model.

- In the Object Type model, all of the objects on the system are logically grouped into administrative domains according to the type of the object. The administrator of a domain maintains all of the objects within the domain. For example, an application group/folder administrator maintains all of the application, application group, and folder objects on the system.
- In the Object Owner model, the objects on the system are logically grouped into administrative domains according to the creator/owner of the object. An administrator maintains only the objects that they create. For example, a user with create application groups and create folders authority can maintain only the applications, application groups, and folders that they created. The Object Owner model can be used to separate the objects on the system into logical parts, such as a department, a company, or some other entity. Each part is independent of the other and should be maintained separately. Each part typically requires two administrative users. One user has the responsibility for creating and maintaining users and groups. The other user has the responsibility for creating and maintaining applications, application groups, and folders. However, you can also define one user with the authority to create and maintain users, groups, applications, application groups, and folders. In effect, the one user would be the system administrator for a logical part of the system.

Object Type model

In the Object Type model, the system administrator defines two new users. One user is responsible for administering applications, application groups, and folders and is defined as an application group/folder administrator. The second user is responsible for administering users and groups and is defined as a user administrator with Create Groups authority. Table 6 shows the administrative users and the tasks assigned to the users.

Table 6. Administrator Roles in the Object Type Model

User Type	Tasks
System Administrator	Create an application group/folder administrator Create a user administrator with Create Groups authority Create and maintain storage sets Create and maintain system printers
User Administrator with Create Groups authority	Create and maintain users Create and maintain groups
Application Group/Folder Administrator	Create and maintain application groups Create and maintain applications Create and maintain folders

When maintaining application groups and folders, the application group/folder administrator must give other users access to the application groups and folders. The recommended and simplest way to do this task is to give access to a group, rather than to individual users. No additional work is required by the application group/folder administrator when another user needs access to the application group or folder. When a new user is added to the group, the user automatically gets access to the application group or folder. Adding the user to the group is the responsibility of the user administrator since the user administrator owns all of the groups in this model.

Another reason for giving groups rather than individual users access to application groups and folders is that the application group/folder administrator does not have access to the users and groups in this model. Because the application group/folder administrator must first be given access to any users or groups that require access to application groups or folders, it is simpler and less time consuming to give access to a few groups rather than hundreds or even thousands of users. The application group/folder administrator is given access to a group by adding the application

group/folder administrator to the group. This task is done by the user administrator with Create Groups authority. As a group member, the application group/folder administrator will be able to see the group in the list and will therefore be able to give the group access to any application groups and folders on the system.

To give an application group/folder administrator access to a user, the user administrator with Create Groups authority must update each user and give the application group/folder administrator access to the user. Once access has been given, the application group/folder administrator will be able to see the user in the list and will therefore be able to grant the user access to any application groups and folders on the system. Again, this is not the recommended approach because this task will have to be repeated each time that a user is added to the system.

Object Owner model

In the Object Owner model, the system administrator defines two users for each logical part of the system. One user is responsible for maintaining the users and groups for a logical part of the system. The other user is responsible for maintaining the applications, application groups, and folders for a logical part of the system. The Object Owner model allows you to store data from several sources on one OnDemand system and let only one set of users access each set of data. Table 7 shows the administrative users and the tasks assigned to the users.

Table 7. Administrator Roles in the Object Owner Model

User Type	Tasks
System Administrator	Create a user with Create Users and Create Groups authority Create a user with Create Application Groups and Create Folders authority Create and maintain storage sets Create and maintain system printers
User with Create Users and Create Groups authority	Create and maintain users Create and maintain groups
User with Create Application Groups and Create Folders authority	Create and maintain application groups Create and maintain applications Create and maintain folders

In addition to the tasks listed in Table 7 under System Administrator, it is also necessary for the system administrator to give the user with Create Users and Groups authority access to the user with Create Application Groups and

Folders authority. Otherwise, the user that creates groups will not be able to add the user that creates application groups and folders to any groups. To simplify the explanation that follows, the user with Create Users and Groups authority will be called the user administrator and the user with Create Application Groups and Folders authority will be called the application group/folder administrator.

When maintaining application groups and folders, the application group/folder administrator must give access to application groups and folders to other users on the system. The recommended and simplest way to do this task is to give access to a group, rather than to individual users. No additional work is required by the application group/folder administrator when another user needs access to the application group or folder. When a new user is added to the group, the user automatically gets access to the application group or folder. Adding the user to the group is the responsibility of the user administrator since the user administrator owns the groups in this model.

Another reason for providing access to application groups and folders from a group rather than to an individual user is that the application group/folder administrator does not have access to the users and groups in this model. Since the application group/folder administrator must first be given access to any users or groups that require access to application groups or folders, it is simpler and less time consuming to give access to a few groups rather than hundreds or even thousands of users. The application group/folder administrator is given access to a group by adding the application group/folder administrator to the group. This is done by the user administrator in this model. As a group member, the application group/folder administrator will be able to see the group in the list and will therefore be able to grant the group access to the application groups and folders that have been defined by the application group/folder administrator in this model.

To give an application group/folder administrator access to a user, the user administrator must update each user and give the application group/folder administrator access to the user. Once access has been given, the application group/folder administrator will be able to see the user in the list and will therefore be able to grant the user access to the application groups and folders. Again, this is not the recommended approach because this task will have to be repeated each time that a user is added to the system.

To illustrate how the Object Owner model can be used, assume that a company installs an OnDemand system to provide data archival and retrieval services for other organizations. The company provides the hardware and software required to administer the system and archive and retrieve the data. An administrator from each organization defines application groups and folders for their data. Another administrator defines the users that can access

the data. The system must be able to limit access to an organization's application groups and folders. Only users defined by an organization should have access to the application groups and folders that are owned by the organization. The system must also be able to limit access to the data. Only users defined by an organization should have access to the data that is owned by the organization. By using the Object Owner model, both requirements can be met.

Chapter 21. Summary

There can be many different variations of the two models that have been described. For example, in the Object Owner model, rather than one user administering both application groups and folders, one user can be defined to administer application groups and another user can be defined to administer folders. Choosing the right model or variation is an important decision that should be made early in the planning process. Changing to a different model later is not impossible but may require additional work if there are a large number of objects defined on the system.

Chapter 22. Helpful hints

1. To simplify the task of providing access to application groups and folders, give access to a group rather than a user. When a new user needs access, add the user to the group.
2. To allow an application group/folder administrator to see groups in the permissions list, add the application group/folder administrator to the groups that require access to application groups and folders.
3. To allow multiple users to administer the same groups, create a group of users and make that group the group owner for any groups that need to be administered by multiple users.
4. The Create Groups authority is most effectively used if it is combined with the Create Users authority or added to a user administrator. Because the purpose of a group is to give a set of users permissions to another object, it is not very useful if the user that creates the group does not have access to any users. Otherwise, the user that creates a group must be given access to each user that needs to be added to the group.

Part 3. Xenos transform

Chapter 23. Understanding Xenos

Beginning with OnDemand Version 7.1, you can process certain types of input files with a transform program that is provided by the Xenos Group. This document gives an overview of the Xenos transform, explains the functions that the Xenos transform can perform, and describes different scenarios for processing your input files.

Important: Before you attempt to use the Xenos transform on your system, you must obtain the transform program, license, and documentation from the Xenos Group. The Xenos Group can also provide education and other types of help and support for processing input files with the transform program.

The Xenos transform is a batch application program that lets you process several different types of print files. The Xenos transform provides converting, indexing, and resource collecting capabilities that let you archive and retrieve documents.

With the Xenos transforms you can:

- Convert AFP data to PDF, Metacode to AFP, Metacode, and PDF, and PCL to PDF.
- Index input data to enhance your ability to view, archive, or retrieve individual pages or groups of pages from large print files.
- Collect the resources needed for printing or viewing a document, so that you can print and view the exact document, possibly years after its creation.

The Xenos transform accepts data from your application programs in these formats:

- AFP data
- Metacode print files
- PCL print files

The Xenos transform can process application print data and resources to produce these files:

- Index file
- Document file
- Resource file

With the files that the Xenos transform creates, you can store the data into OnDemand and then do the following:

- Use the Windows client to search for and retrieve, view, and print the documents.
- Use the OnDemand Web Enablement Kit feature to search for and retrieve, view, and print the documents.

Figure 6 on page 85 shows a high-level overview of how the Xenos transform fits into an OnDemand system for creating, indexing, viewing, and printing documents. The figure shows the resources and the print data, which can be provided by various products, that can flow into the ARSLOAD program for processing. If the Indexer that is specified in OnDemand is Xenos, then the ARSLOAD program calls the Xenos transform with parameter and script files that you create. The files that the Xenos transform produces can then be processed by the ARSLOAD program for archiving and the client programs for viewing and printing. If you plan to use the OnDemand Web Enablement Kit (WEK) feature, then you can retrieve and transform documents that are stored in OnDemand into files that can be viewed by the applets and plug-ins for the Web browsers that are used in your organization. For example, you could use the Xenos transform with the ARSLOAD program to process and load Metacode print files. Then, you could use the WEK to retrieve a Metacode document from the system, call the Xenos transform to convert the Metacode document into a PDF file, and send the PDF file to the browser.

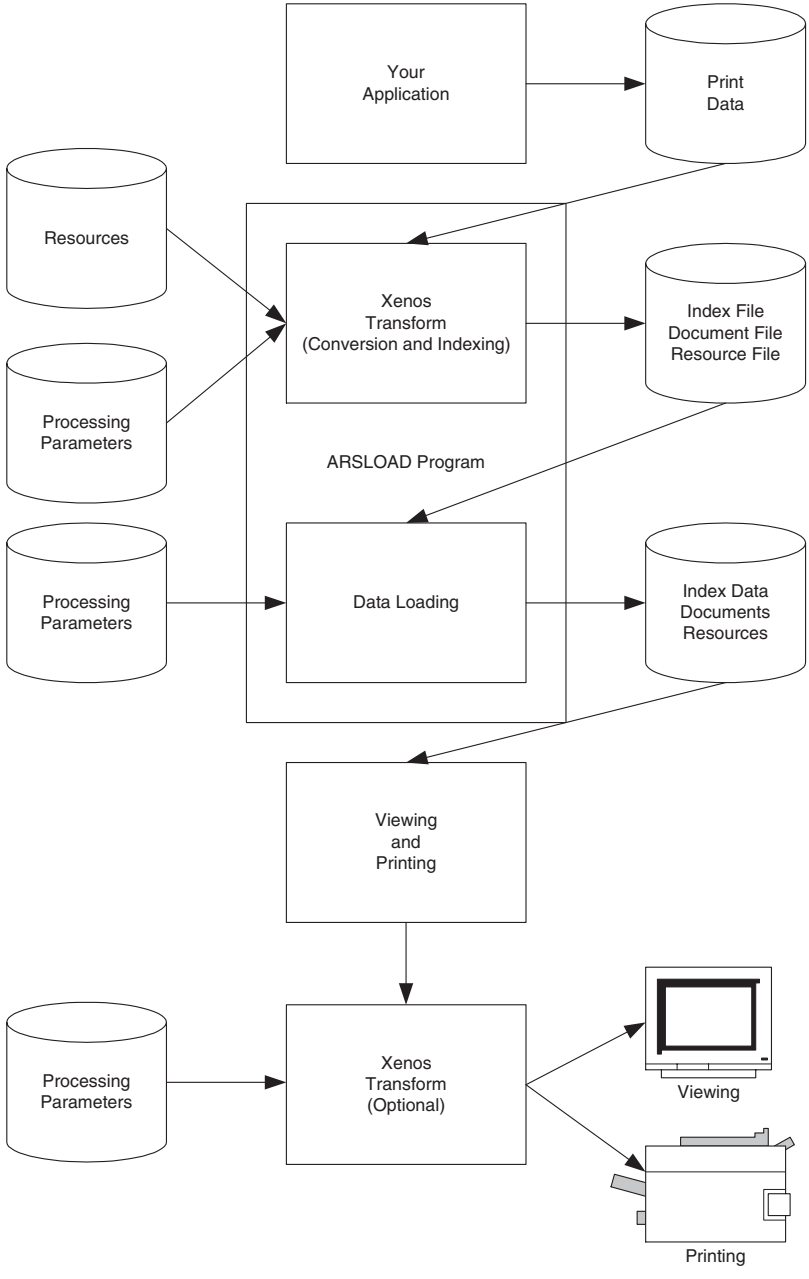


Figure 6. How the Xenos Transform fits into the OnDemand Environment

Chapter 24. Xenos transform

You can use the Xenos transform to perform these functions:

- Convert data streams
- Index documents
- Collect resources

Convert data streams

Most OnDemand customers will use the Xenos transform for the following types of data conversions:

- AFP data to PDF
- Metacode to AFP
- Metacode to Metacode
- Metacode to PDF
- PCL to PDF

The following sections describe each type of data. Please see the Xenos documentation to better understand the data formats and for more information about the transforms.

AFP data to PDF

The AFP to PDF transform converts AFP data streams into Adobe PDF documents. Linedata and mixed mode files that print on AFP printers through PSF are supported, as well as fully-composed MO:DCA-P files. The PDF output can be viewed at the Windows client by using the Adobe Business Tools. The PDF output can be viewed at the Web browser by using the Acrobat plug-in.

To run the AFP to PDF conversion in OnDemand, set the Data Type on the View Information page to PDF and set the Indexer on the Indexing Information page to Xenos. You must also specify processing parameters on the Indexing Information page.

Metacode to AFP

The Metacode to AFP transform converts Xerox Metacode and Linedata Conditioned Data Streams (LCDS) into AFP documents. Linedata and mixed mode files that print on Metacode Printers are supported, as well as pure Metacode files. The AFP output can be viewed at the Windows client. The AFP output can be viewed at the Web browser by using the AFP plug-in.

To run the Metacode to AFP conversion in OnDemand, set the Data Type on the View Information page to AFP and set the Indexer on the Indexing Information page to Xenos. You must also specify processing parameters on the Indexing Information page.

Metacode to Metacode

The Metacode to Metacode transform converts Xerox Metacode and Linedata Conditioned Data Streams (LCDS) to Metacode. Linedata and mixed mode files that print on Metacode Printers are supported, as well as pure Metacode files. At first, the Metacode to Metacode transform might seem a bit redundant, converting a format that already prints on the destination printer to the same format. But the input to the transform can be inefficient linedata or very obscure metacode, where the resulting output Metacode is efficient and in a predictable format, which allows individual pages or groups of pages in the output Metacode to be indexed and retrieved.

To run the Metacode to Metacode conversion in OnDemand, set the Data Type on the View Information page to Metacode and set the Indexer on the Indexing Information page to Xenos. You must also specify processing parameters on the Indexing Information page.

Metacode to PDF

The Metacode to PDF transform converts Xerox Metacode and Line Conditioned Data Streams (LCDS) into Adobe PDF documents. Linedata and mixed mode files that print on Metacode printers are supported, as well as pure Metacode files. The PDF output can be viewed at the Windows client by using the Adobe Business Tools. The PDF output can be viewed at the Web browser by using the Acrobat plug-in.

To run the Metacode to PDF conversion in OnDemand, set the Data Type on the View Information page to PDF and set the Indexer on the Indexing Information page to Xenos. You must also specify processing parameters on the Indexing Information page.

PCL to PDF

The PCL to PDF transform converts Hewlett Packard Printer Control Language (PCL) print files into Adobe PDF documents. The term PCL refers to the compound data stream used by the Hewlett Packard (HP) printers. The transform accepts most PCL 4 or 5 designed for HP desktop printers; the transform does not support the HP PGL or HP Deskjet formats. The PDF output can be viewed at the Windows client by using the Adobe Business Tools. The PDF output can be viewed at the Web browser by using the Acrobat plug-in.

To run the PCL to PDF conversion in OnDemand, set the Data Type on the View Information page to PDF and set the Indexer on the Indexing Information page to Xenos. You must also specify processing parameters on the Indexing Information page.

Index documents

The Xenos transform can index input files. When indexing with the Xenos transform, you can divide a large print file into smaller, uniquely identifiable units, called *groups*. For example, you can use the Xenos transform to divide a large print file that was created by a bank statement application into individual groups by generating group indexes that define the group boundaries in the file. A group is a named collection of sequential pages, which, in this example, consists of the pages describing a single customer's account. For example, a bank statement application probably produces a large printout consisting of thousands of individual customer statements. You can think of these statements as smaller, separate units, each uniquely identifying an account number, date, Social Security number, or other attributes.

Using the Xenos transform, you can create an OnDemand generic index file. The index file lets you:

- Retrieve individual statements from storage volumes, based on an account number or any other attribute.
- More rapidly access the statements for viewing by, for example, the Windows client.
- Archive individual statements or the entire indexed print file for long-term storage and subsequent data management and reprinting, even years after its creation.

The Xenos transform can create a generic index file for the following types of input files:

- AFP data
- Metacode
- PCL data

The Xenos transform lets you generate the group indexes by extracting values that are present in the input data itself, when the data has been formatted so that the Xenos transform can reliably locate the values. This kind of indexing is called *indexing with data values*.

Indexing with data values

Some applications such as payroll or accounting statements contain data that might be appropriate to use for indexing tags. In the bank statement example, the account number is a type of data value that you might want to tag. You can then store a single customer's account statement using the account

number, and you can retrieve and view the same statement using the account number. If the data value that you want to use in an indexing tag is consistently located in the same place for each statement, then you can specify parameters to the Xenos transform that create a separate group of pages for each statement.

Collect resources

The Xenos transform can determine the list of resources needed to view or print the print file and collect the resources from the specified libraries. After you load the document and the resources into the system, you can view or print the document with fidelity. This Xenos function is especially valuable if the resources are not present on the designated platform in a distributed print environment.

When you store a document in OnDemand, you can also store the resources (such as logos and other types of images) in the form in which they existed when the file was printed. By storing the original resources, you can reproduce the document with fidelity at a later date, even if the resources have changed since that time. For example, suppose that a page segment contains the signature of an officer of a company and is included in the print data. When someone else replaces the officer, current print files must reference the new officer's signature, but older files must reference the former officer's signature.

The type of resources that the Xenos transform collects from the specified libraries is based on the parameters that you specify. When the Xenos transform processes a print file, it:

- Identifies the resources requested by the print file.
- Collects the resources from the specified libraries.
- Creates a resource file

Summary

Figure 7 shows the first part of the load process – you run the ARSLOAD program to process an input file and the Indexer that is specified to OnDemand is Xenos.

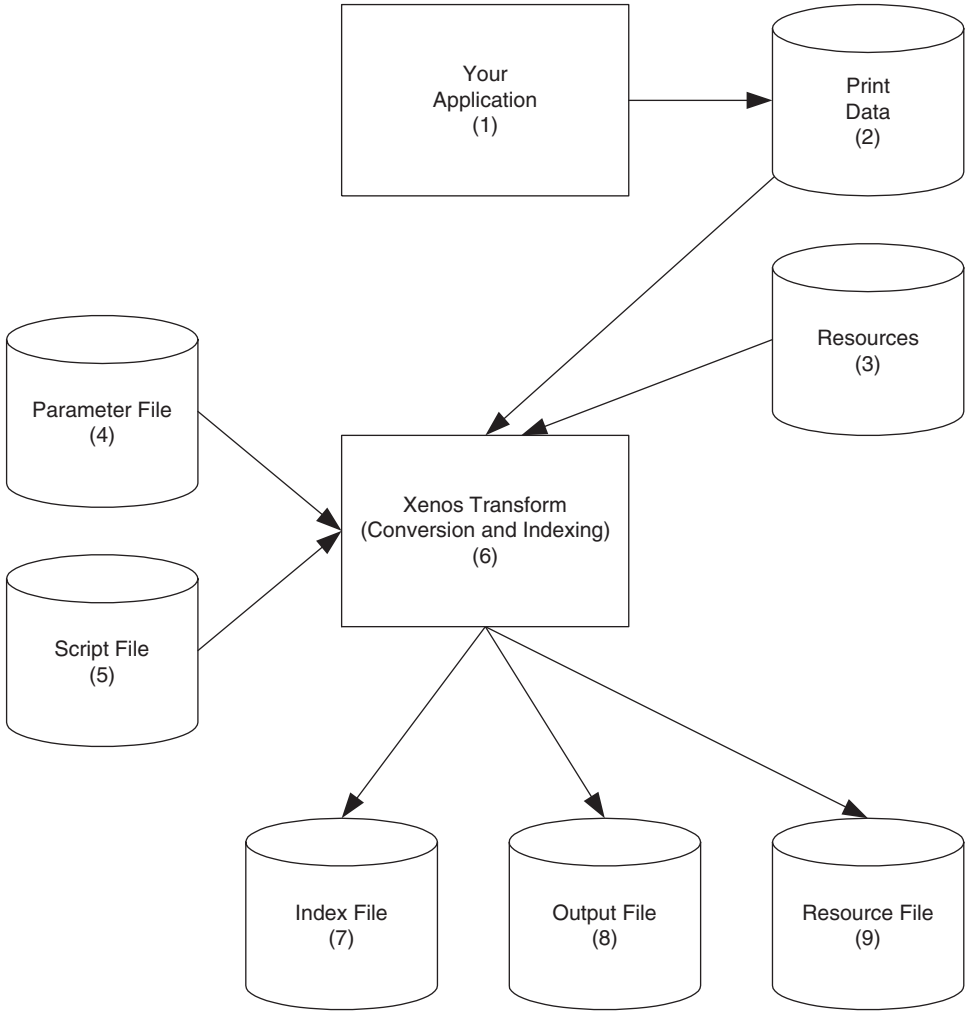


Figure 7. Using the Xenos Transform to Prepare Files for Loading and Viewing

1. The process begins with your application, which is the program that generates your print data.
2. The print data can be AFP, Metacode, or PCL.
3. Resources are stored in resource libraries, by for example, Infoprint Manager.

4. You create the Parameter File, which the Xenos transform uses to locate index data in the input file, convert the input to a specified type of output file, and so forth. See Figure 17 on page 117 for an example of a parameter file.
5. You create the Script File, which the Xenos transform uses to create the OnDemand generic index file and the other output files. See Figure 18 on page 119 for an example of a script file.
6. The ARSLOAD program calls the Xenos transform, to index the print data, convert the input file to the specified type of output, and collect the resources.
7. The Index File contains the index data that is in the OnDemand generic index format.
8. The Output File contains the indexed groups of pages, also known as documents in OnDemand.
9. The Resource File contains the resources that are required to view and print the converted documents.

Chapter 25. Loading data

Figure 8 shows the second part of the load process – the ARSLOAD program processes the files that were created by the Xenos transform.

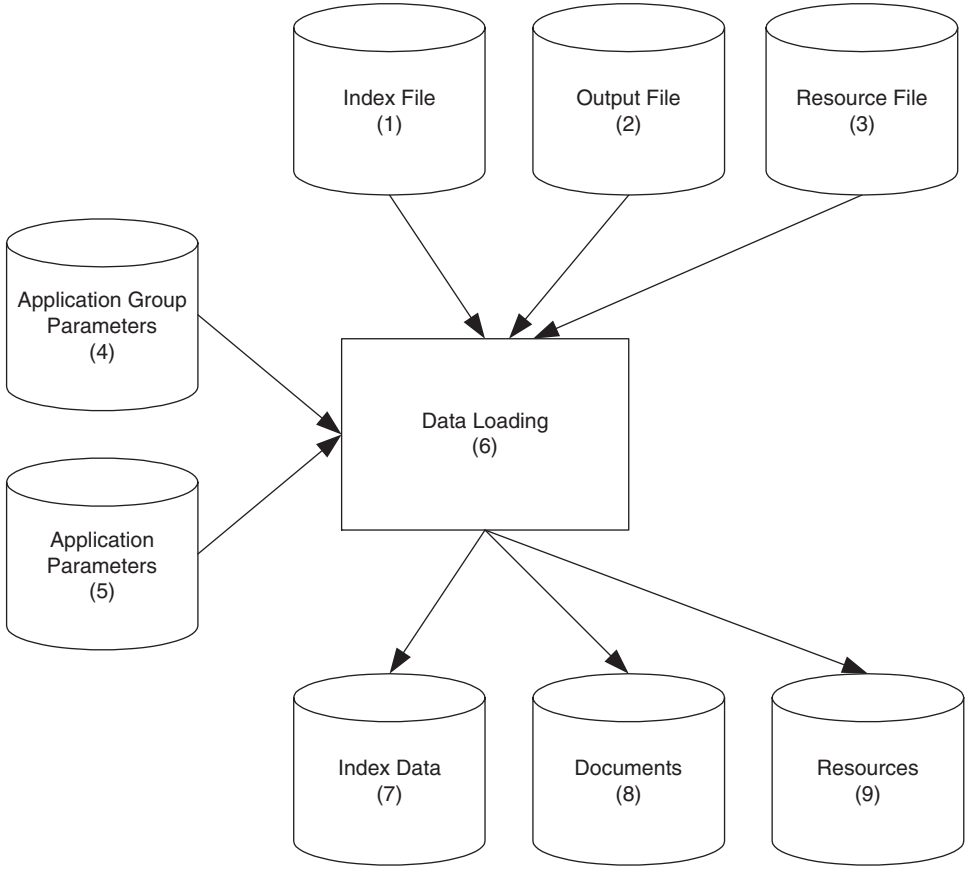


Figure 8. Loading Data into OnDemand

1. The Index File contains the index data that is in the OnDemand generic index format.
2. The Output File contains the indexed groups of pages, also known as documents in OnDemand.
3. The Resource Files contains the resources that are required to view and print the documents.

4. You create the application group parameters, which include the database and storage management information that the ARSLOAD program uses to process the input files. For example, the database fields that you define for the application group will hold the index field values that the Xenos transform extracted from the original print data.
5. You create the application parameters, which include the type of data and the indexer information that the ARSLOAD program uses to process the input files. For example, on the View Information page in applications, you specify the type of data that will be stored in OnDemand, that is, the output from the Xenos transform (AFP, Metacode, or PDF); on the Indexing Information page, you must specify Xenos as the indexing program and specify the parameters that are used to process the input file and create the output file, collect the resources, and generate the index data. See "Specifying parameters to OnDemand" on page 102 for information about creating the application parameters. Figure 11 on page 105 and Figure 12 on page 106 show examples of the Indexing Information page.
6. The ARSLOAD program stores the index data into the database and loads the documents and resources on storage volumes.
7. The Index Data is loaded into the database.
8. The Documents are loaded on to storage volumes.
9. The Resources are loaded on to storage volumes.

Chapter 26. Scenarios for using Xenos

You can use the Xenos transform to process your files for:

- Viewing and printing locally with the Windows client
- Viewing and printing locally with Web Enablement Kit feature

Viewing and printing with the Windows client

Figure 9 on page 97 shows the steps that you take to view and print documents locally with the Windows client.

1. The process begins with the Index Data (1a), Documents (1b), and Resources (1c) that you loaded into the database and on to storage volumes with the ARSLOAD program.
2. Using the Windows client program, your users search for and view the indexed documents. They can also print documents on local printers from the Windows client.

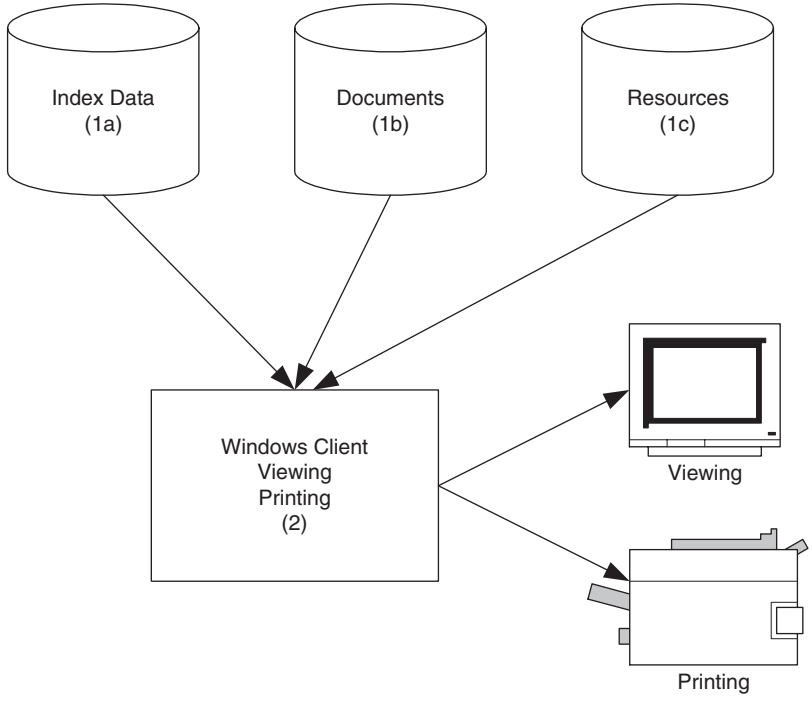


Figure 9. Viewing and Printing with the Windows Client

Viewing and printing with the Web Enablement Kit

Figure 10 on page 99 shows the steps that you take to view and print documents locally with the WEK.

1. The process begins with the Index Data (1a), Documents (1b), and Resources (1c) that you loaded into the database and on to storage volumes with the ARSLOAD program.
2. In the ARSWWW.INI file, you specify information about the documents that the WEK retrieves from the server. For each type of document that you plan to retrieve from OnDemand, you can specify an output format, that is, the format of the document as it is viewed at the Web browser. You also specify a program to convert between the input format and the output format. For example, you can specify a section of the ARSWWW.INI file to describe the process for converting Metacode documents that are stored in OnDemand into PDF documents that are viewed at the Web browser. See "Configuring the ARSWWW.INI file" on page 125 for information about how to configure the ARSWWW.INI file to support the Xenos transform.
3. In the ARSXENOS.INI file, you specify the name of the parameter file and the script file that are used by the Xenos transform to process the input document and create the output document that is sent to the Web browser. See "Configuring the ARSXENOS.INI file" on page 129 for an example of the ARSXENOS.INI file.
4. When the WEK retrieves a document, it checks the ARSWWW.INI file to determine if document conversion is required.
5. If document conversion is required, then the WEK sends the document to the Xenos transform. Otherwise, the WEK sends the document to the Web browser.
6. You create the Parameter File, which provides information about the type of conversion that the Xenos transform performs. See "Example of a parameter file" on page 131 for an example of a parameter file.
7. You create the Script file, which the Xenos transform uses to create the output document. See "Example of a script file" on page 133 for an example of a script file.
8. The Xenos transform converts the document from the format in which it was stored on the server to a format that can be viewed with an applet or plug-in that is available at the Web browser.
9. The user views the document with a Web browser. The user can also print documents on local printers from the Web browser.

See "Chapter 29. Configuring the WEK" on page 125 for more information about configuring the WEK to use the Xenos transform.

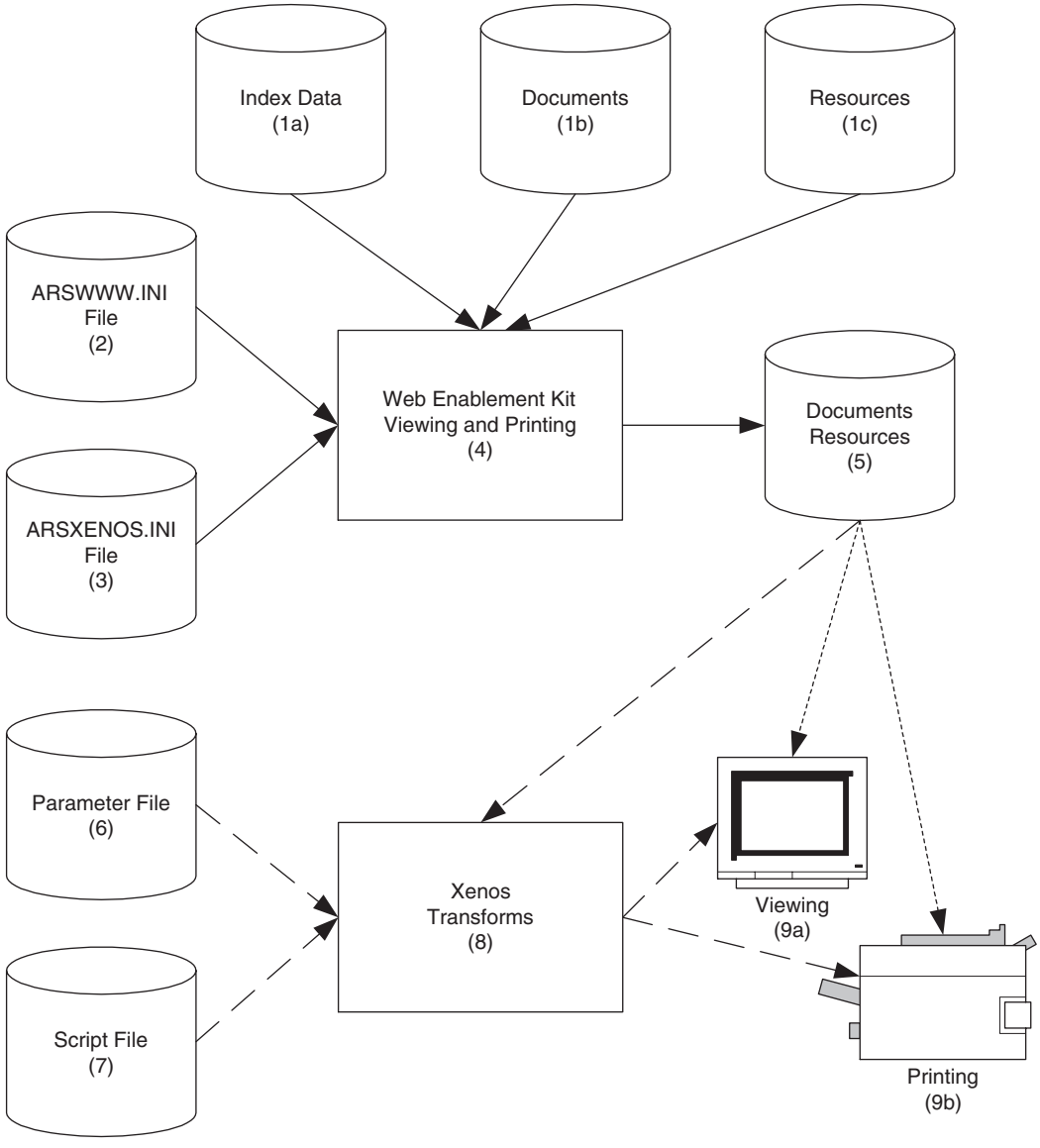


Figure 10. Viewing and Printing with the Web Enablement Kit

Chapter 27. How to specify parameters to the Xenos transform

This section describes how to specify parameters to the Xenos transform to assist you with developing indexing parameters and to specify the parameters that are used by the ARSLOAD program and the Xenos transform to load input data into the system. See "Chapter 29. Configuring the WEK" on page 125 for information about how to specify parameters when you use the WEK to run the Xenos transform.

There are two parts to specifying the parameters that are used by the ARSLOAD program and the Xenos transform.

- First, you should process some sample input data to determine the locations of the text strings that the Xenos transform uses to identify groups and locate the field index values. You can run the Xenos transform from the command line and specify the parameters for this step.
- Then you should specify the parameters that are used by the ARSLOAD program to call the Xenos transform and specify the parameter file and script file that the Xenos transform uses to process the input data and create the output files. You can specify all of the parameters for this step by using the administrative client.

Processing sample data

You can use the Xenos transform (the JS program) to help you determine the location of the text strings in the input data. The JS program processes one or more pages of an input file and generates an output file. The output file contains one record for each text string on a page. Each record contains the coordinates for a box imposed over the text string (upper left, lower right).

The process works as follows:

- Obtain a printed copy of the sample input file.
- Identify the string values that you want to use to locate fields
- Identify the number of the page on which each string value appears. The number is the *sheet number*, not the page identifier. The sheet number is the order of the page as it appears in the file, beginning with the number `0` (zero), for the first page in the file. A page identifier is user-defined information that identifies each page (for example, iv, 5, and 17-3).
- Process one or more pages of the input file with the JS program.

- In the output file (that is identified by the FDDLOUTPUT parameter), locate the records that contain the string values and make a note of the coordinates.
- Create FIELD parameters using the coordinates, page number, and string value.

See "Generating the locations of text strings on a page" on page 111 for an example of how to run the JS program to process sample data.

For information about the parameters that you can specify, including options and data values, and the script file, see your Xenos documentation.

Specifying parameters to OnDemand

The indexing parameters and other processing parameters that are used by the ARSLOAD program and the Xenos transform are part of the OnDemand application. The administrative client provides an edit window that you can use to maintain the parameters for the application. The parameters are:

- The license file. You must always specify the name of the Xenos license file by using the `OnDemandXenosLicenseFile` parameter. For example:

  ```
  OnDemandXenosLicenseFile=c:\tmp\dmlic_IBM_NT.txt
  ```

  The file that you specify must be a valid license file that you obtained from Xenos.
- The warning level. The `OnDemandXenosWarningLevel` parameter determines how the ARSLOAD program will handle return codes from the Xenos transform. For example:

  ```
  OnDemandXenosWarningLevel=4
  ```

  The Xenos transform sets a return code after it converts an input file. Use this parameter to specify the maximum return code that the ARSLOAD program will consider good and continue with the load process. For example, if you specify 4 (four), then the return code that is set by the Xenos transform must be four or less; otherwise, the load will fail. The default value is zero. If you do not specify this parameter, then the Xenos transform must set a return code of zero. Otherwise, the load will fail. See the Xenos documentation for details about return codes.
- The parameter file. You can use one of two methods to specify the parameters: the *file name* method and the *details* method.

  With the file name method, you specify the name of the parameter file. The parameter file that you specify must contain all of the indexing and conversion parameters and other parameters that the Xenos transform uses to process the input data. Use the `OnDemandXenosParmFile` parameter to specify the name of the parameter file. For example:

```
OnDemandXenosParmFile=c:\tmp\sample.par
```

Figure 11 on page 105 shows an example of the Indexing Information page when you use the file name method.

With the details method, you must specify all of the parameters in the edit window within the application. Enclose the parameters inside of the OnDemandXenosParmBegin and OnDemandXenosParmEnd parameters. For example:

```
OnDemandXenosParmBegin
fddmslib  = 'c:\program files\documorph\dmsl.lib'
scriptvar = ( 'Parser', 'META' )
scriptvar = ( 'Generator', 'PDF' )
scriptvar = ( 'NumberOfFields', 2 )
     .
     .
     .
OnDemandXenosParmEnd
```

Figure 12 on page 106 shows an example of the Indexing Information page when you use the details method. **Note:** The example is for demonstration purposes only; not all of the parameters are shown in the edit window.

- The script file. As with the parameter file, you can use the file name method or the details method to specify the script. The script file that you specify must contain all of the code that the Xenos transform uses to generate the OnDemand generic index file and the other output files. Use the OnDemandXenosScriptFile parameter to specify the name of the parameter file. For example:

  ```
  OnDemandXenosScriptFile=c:\tmp\sample.dms
  ```

  Figure 11 on page 105 shows an example of the Indexing Information page when you use the file name method.

  With the details method, you must specify all of the script statements in the edit window within the application. Enclose the script statements inside of the OnDemandXenosScriptBegin and OnDemandXenosScriptEnd parameters. For example:

  ```
  OnDemandXenosScriptBegin
  TRUE = 1;
  FALSE = 0;
  call dm_Initialize
  rc = dm_SetPart( par_h, 'fdinput', inputfile )
       .
       .
       .
  OnDemandXenosScriptEnd
  ```

Figure 12 on page 106 shows an example of the Indexing Information page when you use the details method. **Note:** The example is for demonstration purposes only; not all of the script statements are shown in the edit window.

See "Indexing and converting input data" on page 116 for an example of a parameter file and a script file to convert and index data and generate the OnDemand generic index file and the other output files.

For information about the parameters that you can specify, including options and data values, and the script file, see your Xenos documentation.

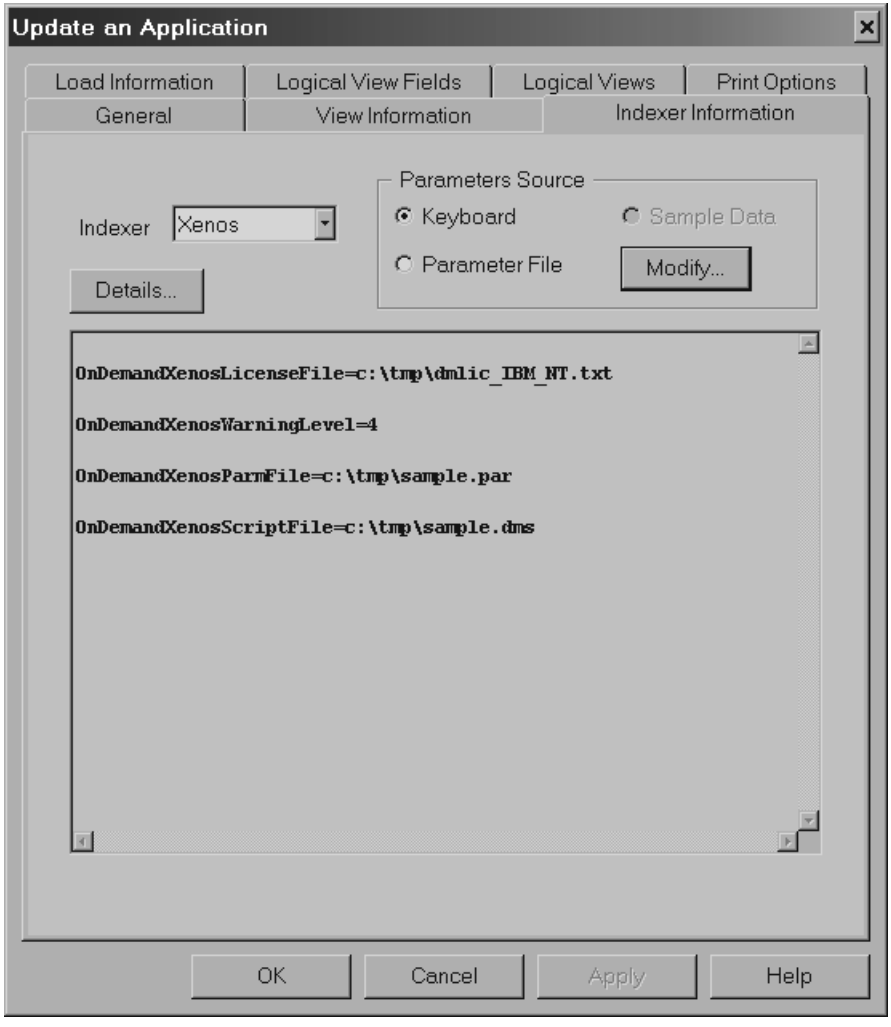


Figure 11. Application Indexing Information – Specifying File Names

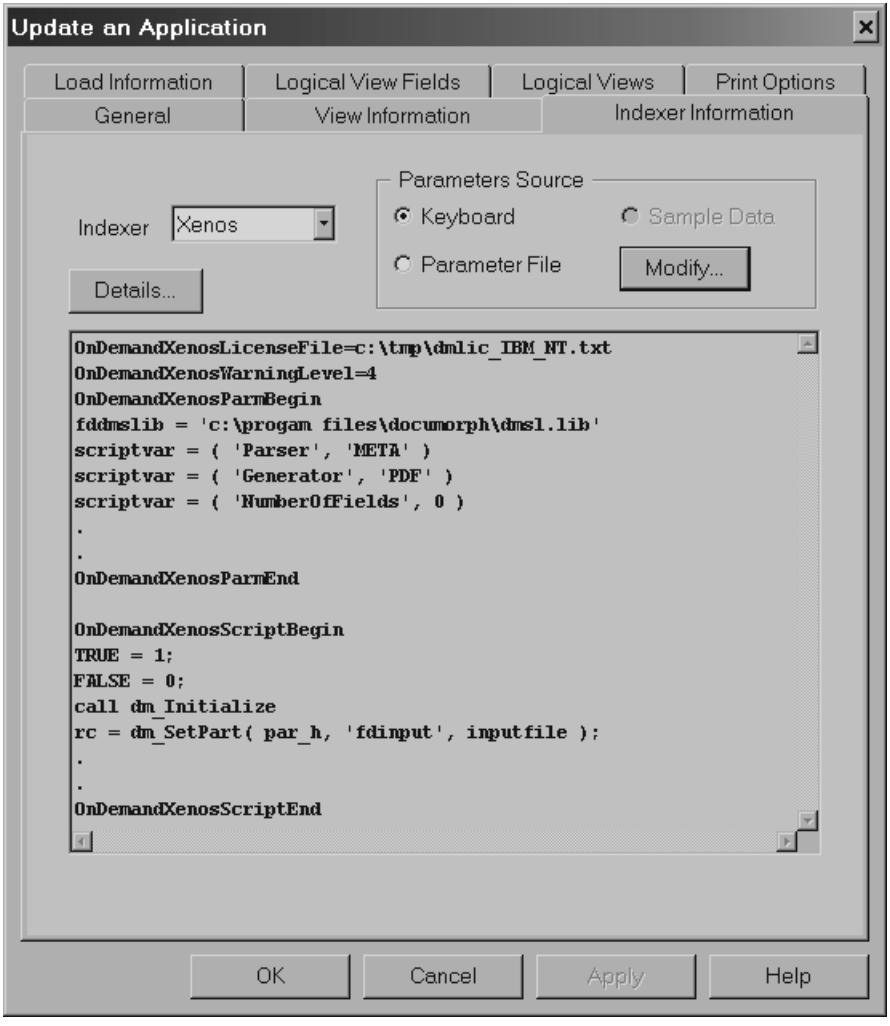


Figure 12. Application Indexing Information – Specifying Details

Chapter 28. JS program reference

Purpose

The JS program is the main Xenos transform program. You can use the JS program in the following ways:

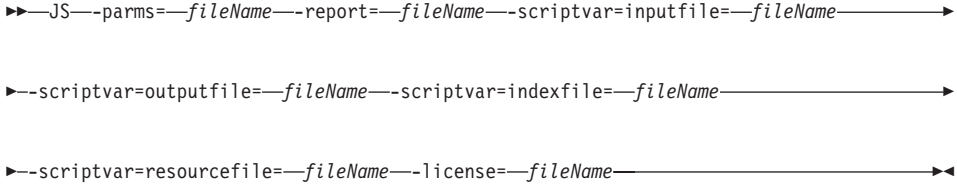
- Manually run the JS program to print the locations of the text strings found on the pages of an input file.
- Automatically call the JS program from the ARSLOAD program to process an input file, generate an OnDemand generic index file and the other output files, convert the input data, and load the data into the system.

Note: See "Chapter 29. Configuring the WEK" on page 125 for information about how to configure the WEK to run the Xenos transform.

Syntax

Note: The following shows the syntax that is used by the ARSLOAD program to call the JS program.

Important: All of the parameters are required.



Description

The JS program can be used to convert various types of input data into formats that are supported by OnDemand. The JS program can extract index data from the input data and collect the resources that are required to view and print the data. The JS program can be used to do the following:

- Print the locations of the text strings found on the pages of an input file. You can use the JS program to help define the indexing fields for your input data. When you define indexing fields, you must specify the location of the string value used to locate the field as a coordinate system imposed on the page. For each string value, you must identify the upper left and

lower right position on the page. To help you create the indexing parameters, you can use the JS program to process a sample input file and list the text strings found on the pages of the input file and the locations of the text strings. When you run the JS program to print the locations of the text strings, you must specify the FDDLOUTPUT parameter. The FDDLOUTPUT parameter identifies the name of the file that will contain the text strings and the location information.

- Convert input data, generate an index file, and collect resources. The ARSLOAD program will automatically call the JS program if the Indexer that is specified on the Indexer Information page under applications is Xenos. You can also run the JS program from the command line to generate an index file manually.

  Before you load an input file into the system, you must create an OnDemand application. The application includes the processing parameters that are used by the ARSLOAD program to call the JS program to convert and index the input file and collect resources. When the ARSLOAD program processes an input file and the indexer that is specified on the Indexing Information page in the application is Xenos, the ARSLOAD program automatically calls the JS program to process the input file. The JS program processes the input file with parameters that you specified for the application. Indexing parameters determine the location of the index data on a page. Conversion parameters determine the type of output data that is generated. The JS program extracts the index data from the input file and converts the input data into the specified output format. After indexing and converting the data, OnDemand loads the index data into the database and the indexed groups on to storage volumes.

Parameters

Note: The following shows the parameters that are used by the ARSLOAD program to call the JS program.

Important: All of the parameters are required.

–parms=

Use to specify the file name or full path name of the file that contains the names of the parameter file and the script file. The parameter file contains the parameters that are used to index and convert the input data. The script file creates the OnDemand generic index file and the other output files. See Figure 15 on page 112 and Figure 17 on page 117 for examples of parameter files. See Figure 16 on page 114 and Figure 18 on page 119 for examples of script files. On UNIX servers, file and path names are case sensitive.

Note: For information about the parameters that you can specify, including options and data values, see your Xenos documentation.

–report=
Use to specify the file name or full path name of the file to which a job report will be written. The report will contain a list of the parameters used to process the input file, any error messages, and a list of resources used to process the input file. On UNIX servers, file and path names are case sensitive.

–scriptvar=inputfile=
Use to specify the file name or full path name of the file that contains the input data to process. On UNIX servers, file and path names are case sensitive.

–scriptvar=outputfile=
Use to specify the file name or full path name of the file that will contain the converted output data. On UNIX servers, file and path names are case sensitive.

Note: If you are using the JS program to generate the text strings and locations, then you can discard the output file.

–scriptvar=indexfile=
Use to specify the file name or full path name of the file that will contain the index data that will be loaded into the OnDemand database. On UNIX servers, file and path names are case sensitive.

–scriptvar=resourcefile=
Use to specify the file name or full path name of the file that will contain the resources that were collected. On UNIX servers, file and path names are case sensitive.

–license=
Use to specify the file name or full path name of the file that contains the license information required to run the Xenos programs. On UNIX servers, file and path names are case sensitive.

Note: You must obtain a valid license file from Xenos.

Examples

The syntax of the JS program is the same, whether you use the program to print the text strings and the location information or you use the program to generate the index and output files to be loaded into OnDemand. The only variables in the process are the type of conversion to run and the indexing parameters that the JS program uses to process the input data, which are specified in the processing parameter file, and the script file.

Figure 13 shows an example of how to run the JS program.

```
js -parms="c:\documorph\sampletests\parms.auto"
   -report="c:\documorph\sampletests\sample.rep"
   -scriptvar=inputfile="c:\documorph\sample.mta"
   -scriptvar=outputfile="c:\documorph\sample.out"
   -scriptvar=indexfile="c:\documorph\sample.ind"
   -scriptvar=resourcefile="c:\documorph\sample.res"
   -license="c:\documorph\dmlic_IBM_NT.txt"
```

Figure 13. Running the JS Program

In Figure 13:

- The file that contains the name of the parameter file and the script file is `parms.auto`
- The file that the job report will be written to is `sample.rep`
- The file that contains the input data to process is `sample.mta`
- The file that will contain the converted data to be loaded into the system is `sample.out`
- The file that will contain the index data that will be loaded into the database is `sample.ind`
- The file that will contain the resources that will be loaded into the system is `sample.res`
- The file that contains the Xenos license information is `dmlic_IBM_NT.txt`

Figure 14 shows the contents of the example `parms.auto` file.

```
fdjobparm='\documorph\sampletests\meta2pdf\sample.par'
fddmscript='\documorph\sampletests\meta2pdf\sample.dms
```

Figure 14. Sample Parameter File

In Figure 14, `sample.par` is the name of the file that contains the processing parameters and `sample.dms` is the name of the script file.

Generating the locations of text strings on a page

This example shows how to use the JS program to process a Metacode input file and produce the text strings and location information. You can use the information generated in this step to specify the indexing information that is used by the JS program to extract index data from the input files that you want to load on the system (see "Indexing and converting input data" on page 116).

- Figure 15 on page 112 shows the processing parameters. Note that all of the parameters that are required to process an input file and produce the text strings and the location information are not shown in the example. See your Xenos documentation for a complete list of indexing parameters, options, and data values and more detailed information.

  Notes:

  1. The script variables `Parser` and `Generator` determine the type of conversion taking place. The script variable `Parser` represents the input file format and the script variable `Generator` represents the output file format. To perform an AFP to PDF conversion, specify the following:

     ```
     scriptvar=('Parser', 'AFP')
     scriptvar=('Generator', 'PDF')
     ```

     To perform a Metacode to AFP conversion, specify the following:

     ```
     scriptvar=('Parser', 'META')
     scriptvar=('Generator', 'AFP')
     ```

  2. The script variable `NumberOfFields` must be set to `0` (zero) when running the JS program to produce text string coordinates.
  3. The `STARTPAGE` and `STOPPAGE` parameters determine how many pages of the input file to process. In the example, the entire input file will be processed. To process a range of pages, specify the number of the starting page to process and the number of pages to process. For example, if you specify `STARTPAGE=0` and `STOPPAGE=10`, then the JS program will process the first eleven pages of the input file.

     Note: The input data must contain the information that the JS program uses to determine when one page ends and another page begins. For example, if the input data contains carriage control characters, then a Skip-to-Channel One carriage control character signals the beginning of a new page.
  4. The `FDDLOUTPUT` parameter determines the file name or full path name of the file that will contain the text strings and the location information. The `FDDLOUTPUT` parameter is required when you run the JS to generate the text strings and the locations.

- Figure 16 on page 114 shows the script file. **Note:** The script file does not create an index file, because an index file is not needed for this step.

```
/* Sample processing parameters for Meta2PDF transform */

JS:

/* DM Script Library - XG supplied functions */
fddmslib = 'c:\program files\documorph\dmsl.lib'

scriptvar = ('Parser', 'META')
scriptvar = ('Generator', 'PDF')
scriptvar = ('NumberOfFields', 0)

METADL-METAP:

/* Metacode Parser Options */
startjdl   = GTIJDL
startjde   = START
startpage  = 0
stoppage   = 0
position   = WORD
native     = NO
cc         = YES
shade      = NONE
dashline   = NO
ldmethod   = LARGEST
cctran     = NONE

/* File Defs */
fdfnt      = 'c:\documorph\resources\meta\%s.fnt'
fdjsl      = 'c:\documorph\resources\meta\%s.jsl'
respectshift = NO

/* MTA2PDF Options */
fdfonttab  = 'c:\documorph\sampletests\meta2pdf\newfont.tab'
```

Figure 15. Sample Parameters for the Xenos Transform (Part 1 of 2)

```
PDFGEN-PDFOUT:

/* PDF Out Generator Options */
/* output file name being set in the script */
offset     = (0,0)
scaleby    = 100
border     = NONE
compress = (NONE,NONE,NONE)
orient     = AUTO
pdfauthor  = 'Xenos Group'
pdfopenact = '/FitH 800'
bmorder = (AsIs,AsIs,AsIs)

METADL-DLIST:

parmdpi = 300
pagefilter = all
resfilter = all
fddloutput = sample.dl

fieldname = (PAGE)
fieldword = (20, and, %20)
fieldphrase = (%500, OneSpace)
fieldpara = (%500)
```

Figure 15. Sample Parameters for the Xenos Transform (Part 2 of 2)

```
TRUE = 1;
FALSE = 0;

call dm_Initialize

par_h = dm_StartParser(Parser);
gen_h = dm_StartGenerator(Generator);

rc = dm_SetParm(par_h, 'fdinput', inputfile);

/* start the DASD Distributors */
dasd_h = dm_StartDistributor("DASD");

/* open output file */
rc = dm_DASDOpen(dasd_h, '{GROUPFILENAME}'outputfile);


/* initialize */
file_open = FALSE

dlpage = dm_GetDLPage(gen_h);
```

Figure 16. Sample JS Program Script File (Part 1 of 2)

```
do while(dlpage <> 'EOF')
  if file_open = FALSE then do
    if(generator = 'PDF') then do
      rc = dm_PDFGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'AFP') then do
      rc = dm_AFPGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'META') then do
      rc = dm_METAGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    file_open = TRUE
  end

  if(generator = 'PDF') then do
    rc = dm_PDFGenWrite(gen_h, dlpage );
  end
  else if(generator = 'AFP') then do
    rc = dm_AFPGenWrite(gen_h, dlpage );
  end
  else if(generator = 'META') then do
    rc = dm_METAGenWrite(gen_h, dlpage );
  end

  dlpage = dm_GetDLPage(par_h);
end

if file_open = TRUE then do
  if(generator = 'PDF') then do
     rc = dm_PDFGenClose( gen_h )
  end
  else if(generator = 'AFP') then do
     rc = dm_AFPGenClose( gen_h )
  end
  else if(generator = 'META') then do
     rc = dm_METAGenClose( gen_h )
  end
end

rc = dm_DASDClose( dasd_h )
return;
```

Figure 16. Sample JS Program Script File (Part 2 of 2)

Indexing and converting input data

This example shows how to use the JS program to process a Metacode input file and produce an index file and a converted output file that can be loaded into OnDemand.

- Figure 17 on page 117 shows the processing parameters. Note that all of the parameters that are required to process an input file are not shown in the example. See your Xenos documentation for a complete list of processing parameters, options, and data values and more detailed information.

  Notes:

  1. The script variable `NumberOfFields` determines the number of index fields.
  2. The script variables `Field.1` and `Field.2` identify the names and locations of the index fields, using the text string coordinates that were generated in "Generating the locations of text strings on a page" on page 111.
  3. The `fieldlocate` parameter determines the *anchor* character string that the Xenos transform will search for. The Xenos transform will locate the index fields based on the location of the anchor string.
  4. The `fieldbase` parameters identify the locations of index fields using the specified offset from the anchor string.
  5. The `fieldname` parameters identify the names of the index fields.
  6. The `STARTPAGE` and `STOPPAGE` parameters determine how many pages of the input file to process. In the example, the entire input file is processed.

     Note: The input data must contain the information that the JS program uses to determine when one page ends and another page begins. For example, if the input data contains carriage control characters, then a Skip-to-Channel One carriage control character signals the beginning of a new page.
  7. The absence of the `FDDLOUTPUT` parameter means that the JS program will not generate the text strings and the location information.
  8. For information regarding the `FieldLocate`, `FieldName`, and `FieldBase` parameters, refer to your Xenos documentation.
- Figure 18 on page 119 shows the script file.
- Figure 19 on page 123 shows the OnDemand generic index file that was created by the script file.

```
/* Sample indexing parameters for META2PDF transform */

JS:

/* DM Script Library - XG supplied functions */
fddmslib = 'c:\program files\documorph\dmsl.lib'

scriptvar = ('Parser', 'META')
scriptvar = ('Generator', 'PDF')
scriptvar = ('NumberOfFields', 2)
scriptvar = ('Field.1','Name')
scriptvar = ('Field.2','Policy')

METADL-METAP:

/* Metacode Parser Options */
startjdl   = GTIJDL
startjde   = START
startpage  = 0
stoppage   = 0
position   = WORD
native     = NO
cc         = YES
shade      = NONE
dashline   = NO
ldmethod   = LARGEST
cctran     = NONE

/* File Defs */
fdfnt      = 'c:\documorph\resources\meta\%s.fnt'
fdjsl      = 'c:\documorph\resources\meta\%s.jsl'
respectshift = NO
```

Figure 17. Sample Parameters for the Xenos Transform (Part 1 of 2)

```
/* MTA2PDF Options */
fdfonttab  = 'c:\documorph\sampletests\meta2pdf\newfont.tab'

PDFGEN-PDFOUT:

/* PDF Out Generator Options */
/* output file name being set in the script */
offset     = (0,0)
scaleby    = 100
border     = NONE
compress   = (NONE,NONE,NONE)
orient     = AUTO
pdfauthor  = 'Xenos Group'
pdfopenact = '/FitH 800'
bmorder    = (AsIs,AsIs,AsIs)

METADL-DLIST:

parmdpi    = 300
pagefilter = all
resfilter  = all

fieldname   = (PAGE)
fieldword   = (20, and, %20)
fieldphrase = (%500, OneSpace)
fieldpara   = (%500)

/* field 1 - extract name */
fieldlocate = ('InsName', 'Insured')
fieldname   = ('Name')
fieldbase   = ('InsName', +275,
               '=',    -35,
               '=',    +800,
               '=',    +30)

/* field 2 - extract policy number */
fieldname   = ('Policy')
fieldbase   = ('InsName', +300,
               '=',       +100,
               '=',       +800,
               '=',       +170)
```

Figure 17. Sample Parameters for the Xenos Transform (Part 2 of 2)

```
TRUE = 1;
FALSE = 0;

call dm_Initialize

par_h = dm_StartParser(Parser);
gen_h = dm_StartGenerator(Generator);

rc = dm_SetParm(par_h, 'fdinput', inputfile);

/* start the DASD Distributors */
dasd_h = dm_StartDistributor("DASD");
index_h = dm_StartDistributor("DASD");

/* open output and index files */
rc = dm_DASDOpen(dasd_h, '{GROUPFILENAME}'outputfile);
rc = dm_DASDOpen(index_h, indexfile );

/* initialize */
do i = 1 to NumberOfFields
  fieldvaluesave.i = ""
end
file_open = FALSE
save_BytesWritten = 0
crlf = '0d0a'X

/* write preamble to the index file */
rc = dm_DASDWrite(index_h, "COMMENT: OnDemand Generic Index File Format"||crlf);
rc = dm_DASDWrite(index_h, "COMMENT: File generated by the xenos process"||crlf);
dt = DATE('N');
ts = TIME('N');
rc = dm_DASDWrite(index_h, "COMMENT:" dt||" "||ts||crlf);
rc = dm_DASDWrite(index_h, "COMMENT:"||crlf);
rc = dm_DASDWrite(index_h, "CODEPAGE:819"||crlf||crlf);

dlpage = dm_GetDLPage(gen_h);

do while(dlpage <> 'EOF')
  if file_open = FALSE then do
    if(generator = 'PDF') then do
      rc = dm_PDFGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'AFP') then do
    rc = dm_AFPGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'META') then do
      rc = dm_METAGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    if rc = 0 then do
      file_open = TRUE
```

Figure 18. Sample JS Program Script File (Part 1 of 4)

```
      end
    end
    do i = 1 to NumberOfFields
    fieldvalue.i = dm_GetText( dlpage, field.i, First )
  end

  docbreak = 0
  do i = 1 to NumberOfFields
    if fieldvalue.i <> "" then do
      /* if there is no previous value, save the current value */
      if fieldvaluesave.i = "" then do
        fieldvaluesave.i = fieldvalue.i
      end
      else
      /* if there is a previous value, see if the new value is different */
      if fieldvaluesave.i <> fieldvalue.i then do
        docbreak = 1
      end
    end
  end
if docbreak = 1 then do
    if(generator = 'PDF') then do
       rc = dm_PDFGenClose( gen_h )
    end
    else if(generator = 'AFP') then do
       rc = dm_AFPGenClose( gen_h )
end
    else if(generator = 'META') then do
       rc = dm_METAGenClose( gen_h )
    end
    file_open = FALSE

    /* write out index values to the index file */
    do i = 1 to NumberOfFields
      field_name = "GROUP_FIELD_NAME:"||field.i
      rc = dm_DASDWrite( index_h, field_name||crlf)
      field_value = "GROUP_FIELD_VALUE:"||fieldvaluesave.i
      rc = dm_DASDWrite( index_h, field_value||crlf)
    end
```

Figure 18. Sample JS Program Script File (Part 2 of 4)

```
    /* replace index values with the new values */
    do i = 1 to NumberOfFields
      if fieldvalue.i <> "" then do
        fieldvaluesave.i = fieldvalue.i
      end
    end

    rc = dm_DASDSize(dasd_h)
    BytesWritten = dm_size
    length = BytesWritten - save_BytesWritten
    offset = BytesWritten - length
    save_BytesWritten = BytesWritten

    group_offset = "GROUP_OFFSET:"||offset
    rc = dm_DASDWrite( index_h, group_offset||crlf)
    group_length = "GROUP_LENGTH:"||length
    rc = dm_DASDWrite( index_h, group_length||crlf)
    group_filename = "GROUP_FILENAME:"||outputfile
    rc = dm_DASDWrite( index_h, group_filename||crlf||crlf)

if(generator = 'PDF') then do
    rc = dm_PDFGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'AFP') then do
      rc = dm_AFPGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'META') then do
      rc = dm_METAGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    if rc = 0 then do
      file_open = TRUE
    end
  end

  if(generator = 'PDF') then do
      rc = dm_PDFGenWrite(gen_h, dlpage );
  end
```

Figure 18. Sample JS Program Script File (Part 3 of 4)

```
  else if(generator = 'AFP') then do
    rc = dm_AFPGenWrite(gen_h, dlpage );
  end
  else if(generator = 'META') then do
    rc = dm_METAGenWrite(gen_h, dlpage );
  end

  dlpage = dm_GetDLPage(par_h);
end

if file_open = TRUE then do
  if(generator = 'PDF') then do
     rc = dm_PDFGenClose( gen_h )
  end
  else if(generator = 'AFP') then do
     rc = dm_AFPGenClose( gen_h )
  end
  else if(generator = 'META') then do
     rc = dm_METAGenClose( gen_h )
  end
end

/* write out final index values to the index file */
do i = 1 to NumberOfFields
  field_name = "GROUP_FIELD_NAME:"||field.i
  rc = dm_DASDWrite( index_h, field_name||crlf)
  field_value = "GROUP_FIELD_VALUE:"||fieldvaluesave.i
  rc = dm_DASDWrite( index_h, field_value||crlf)
end

rc = dm_DASDSize(dasd_h)
BytesWritten = dm_size
length = BytesWritten - save_BytesWritten
offset = BytesWritten - length
save_BytesWritten = BytesWritten

group_offset = "GROUP_OFFSET:"||offset
rc = dm_DASDWrite( index_h, group_offset||crlf)
group_length = "GROUP_LENGTH:"||length
rc = dm_DASDWrite( index_h, group_length||crlf)
group_filename = "GROUP_FILENAME:"||outputfile
rc = dm_DASDWrite( index_h, group_filename||crlf)

rc = dm_DASDClose( dasd_h )
rc = dm_DASDClose( index_h )
return;
```

Figure 18. Sample JS Program Script File (Part 4 of 4)

```
COMMENT: OnDemand Generic Index File Format
COMMENT: This file has been generated by the xenos process
COMMENT: 20 Feb 2001 14:16:18
COMMENT:
CODEPAGE:819

GROUP_FIELD_NAME:Name
GROUP_FIELD_VALUE:Ward T. Jones, MD.
GROUP_FIELD_NAME:Policy
GROUP_FIELD_VALUE:0030-334-33
GROUP_OFFSET:0
GROUP_LENGTH:111783
GROUP_FILENAME:\documorph\sampletests\meta2pdf\sample.out

GROUP_FIELD_NAME:Name
GROUP_FIELD_VALUE:WARD T. JONES, MD
GROUP_FIELD_NAME:Policy
GROUP_FIELD_VALUE:0030-334-33
GROUP_OFFSET:111783
GROUP_LENGTH:339119
GROUP_FILENAME:\documorph\sampletests\meta2pdf\sample.out

GROUP_FIELD_NAME:Name
GROUP_FIELD_VALUE:Mike R. Smith.
GROUP_FIELD_NAME:Policy
GROUP_FIELD_VALUE:0333-888-45
GROUP_OFFSET:450902
GROUP_LENGTH:106689
GROUP_FILENAME:\documorph\sampletests\meta2pdf\sample.out

GROUP_FIELD_NAME:Name
GROUP_FIELD_VALUE:MIKE R. SMITH
GROUP_FIELD_NAME:Policy
GROUP_FIELD_VALUE:0333-888-45
GROUP_OFFSET:557591
GROUP_LENGTH:338985
GROUP_FILENAME:\documorph\sampletests\meta2pdf\sample.out

GROUP_FIELD_NAME:Name
GROUP_FIELD_VALUE:Barbara  L. Schuster
GROUP_FIELD_NAME:Policy
GROUP_FIELD_VALUE:4567-001-77
GROUP_OFFSET:896576
GROUP_LENGTH:106705
GROUP_FILENAME:\documorph\sampletests\meta2pdf\sample.out

GROUP_FIELD_NAME:Name
GROUP_FIELD_VALUE:BARBARA  L. SCHUSTER
GROUP_FIELD_NAME:Policy
GROUP_FIELD_VALUE:4567-001-77
GROUP_OFFSET:1003281
GROUP_LENGTH:334110
GROUP_FILENAME:\documorph\sampletests\meta2pdf\sample.out
```

Figure 19. Generic Index File Produced by the JS Program Script

Chapter 29. Configuring the WEK

If you plan to use the WEK feature on your system, then you can retrieve and transform documents that are stored in OnDemand into files that can be viewed by the applets and plug-ins for the Web browsers that are used in your organization. For example, you could use the Xenos transform with the ARSLOAD program to process and load Metacode print files. Then, you could use the WEK to retrieve a Metacode document from the system, call the Xenos transform to convert the Metacode document into a PDF file, and send the PDF file to the browser.

This section describes how to configure the WEK to run the Xenos transform. This section contains the following topics:

- Configuring the ARSWWW.INI file
- Configuring the ARSXENOS.INI file
- Example of a parameter file
- Example of a script file
- Changes to the Retrieve Document API

Configuring the ARSWWW.INI file

You must make the following changes to the ARSWWW.INI file to enable the Xenos transform:

- Add the XENOS section
- Specify a MIME content type for Metacode documents
- Specify an AFP transform option
- Specify a Metacode transform option

[XENOS]

The XENOS section contains the parameters that are used by the Xenos transform. You can run the Xenos transform from the WEK to convert AFP and Metacode documents and resources into PDF documents that can be viewed with the Adobe Acrobat plug-in.

Notes:

1. To convert AFP and Metacode documents, an administrator must obtain the Xenos transform from the Xenos Group and install and configure it on the server. See your Xenos representative for more information about the Xenos transform. An administrator must also provide configuration options for the Xenos transform. See “Configuring the ARSXENOS.INI file” on page 129 for more information about the configuration file.

2. To convert AFP documents with the Xenos transform, you must specify the AFPVIEWING=XENOS parameter in the DEFAULT BROWSER section (or other browser sections). See "AFPVIEWING" on page 127 for details. (If you plan to use the Retrieve Document API, then you should specify the _afp=XENOS parameter. See "Changes to the Retrieve Document API" on page 135 for details.)
3. To convert Metacode documents with the Xenos transform, you must specify the METAVIEWING=XENOS parameter in the DEFAULT BROWSER section (or other browser sections). See "METAVIEWING" on page 128 for details. (If you plan to use the Retrieve Document API, then you should specify the _meta=XENOS parameter. See "Changes to the Retrieve Document API" on page 135 for details.)
4. By default, the WEK uses the Adobe Acrobat plug-in to view converted documents.
5. If a document is stored in OnDemand as a large object, then the value of the ALLOBJECTS parameter determines how the WEK handles the document. See "Configuring the ARSXENOS.INI file" on page 129 for details.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

CONFIGFILE
The configuration file that contains the options used by the Xenos transform to convert AFP and Metacode documents and resources into PDF documents that can be viewed with the Adobe Acrobat plug-in. "Configuring the ARSXENOS.INI file" on page 129 provides information about the configuration file that is provided with OnDemand.

This parameter has a global scope, and you specify it only once in the XENOS section.

This parameter is optional.

Example:

```
[XENOS]
CONFIGFILE=arsxenos.ini
```

INSTALLDIR
The directory that contains the Xenos transform programs and configuration files. Specify the full path name of the directory on the Web server.

Note: Verify the permissions of the directory that you specify. The processes that run WEK programs must read the install directory.

This parameter has a global scope, and you specify it only once in the XENOS section.

This parameter is optional.

Example:

```
[XENOS]
INSTALLDIR=/usr/lpp/ars/documorph
```

Specifying a MIME content type for Metacode

If you plan to process Metacode documents with the Xenos transform, then you need to add the following parameter to the MIMETYPES section of the ARSWWW.INI file.

META

The MIME content type for Metacode documents. By default, there is no MIME content type associated with Metacode documents.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional. However, if you do not specify this parameter, then the WEK sets the MIME content type to `application/unknown`.

Example:

```
[MIMETYPES]
META=application/unknown
```

Specifying the AFP transform option

If you plan to process AFP documents with the Xenos transform, then you need to set the value of the AFPVIEWING parameter in the DEFAULT BROWSER (or other browser sections) of the ARSWWW.INI file.

AFPVIEWING

When the WEK retrieves an AFP document from the OnDemand server, the value of this parameter determines what action, if any, that the WEK takes before sending the document to the client. For example, to convert AFP documents to PDF with the Xenos transform, specify AFPVIEWING=XENOS so that the WEK will call the Xenos transform to convert the AFP document before sending it to the client.

This parameter has a global scope, and you specify it only once in the DEFAULT BROWSER section. When using the Retrieve Document function, you can override the specified action with the `_afp` parameter.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
AFPVIEWING=XENOS
```

Specifying the Metacode transform option

If you plan to process Metacode documents with the Xenos transform, then you need to set the value of the METAVIEWING parameter in the DEFAULT BROWSER (or other browser sections) of the ARSWWW.INI file.

METAVIEWING

When the WEK retrieves a Metacode document from the OnDemand server, the value of this parameter determines what action, if any, that the WEK takes before sending the document to the client. For example, to convert Metacode documents to PDF with the Xenos transform, specify METAVIEW=XENOS so that the WEK will call the Xenos transform to convert the Metacode document before sending it to the client.

You can set the parameter to one of the following values:

NATIVE The WEK extracts and will uncompress Metacode documents and their resources from OnDemand.

Note: If you specify METAVIEWING=NATIVE, then verify that the MIME content type for Metacode documents identifies the viewer that you want to use. See "Specifying a MIME content type for Metacode" on page 127 for details.

XENOS The WEK calls the Xenos transform to convert Metacode documents to PDF documents.

Note: If you specify METAVIEWING=XENOS, then verify that the MIME content type for PDF documents identifies the viewer that you want to use. See the PDF parameter in the MIMETYPES section of the ARSWWW.INI file and the *Web Enablement Kit: Installation and Configuration Guide* for details.

This parameter has a global scope, and you specify it only once in the DEFAULT BROWSER section. When using the Retrieve Document function, you can override the specified action with the `_meta` parameter.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
METAVIEWING=XENOS
```

Configuring the ARSXENOS.INI file

The Xenos transform converts AFP and Metacode documents and resources into PDF documents. The Xenos transform is licensed software from the Xenos Group. An administrator must install and configure the Xenos transform on the Web server. See your Xenos representative for more information about the Xenos transform. An administrator must also specify configuration options for the documents and resources that you plan to process with the Xenos transform. This section describes how to specify the configuration options.

Note: In this document, the name ARSXENOS.INI refers to the configuration file. To specify the file that contains the configuration options, see "CONFIGFILE" on page 126.

The ARSXENOS.INI file provides configuration options for the Xenos transform. You typically configure the ARSXENOS.INI file with options for specific OnDemand applications. However, you can also provide a set of default options. The Xenos transform uses the default options when it converts documents from applications that are not identified in the ARSXENOS.INI file.

The following topics provide additional information about the ARSXENOS.INI file:

- Specifying the ARSXENOS.INI file
- Viewing converted documents

Note: To convert AFP documents with the Xenos transform, you must also specify the `AFPVIEWING=XENOS` parameter in the DEFAULT BROWSER section (or other browser sections) of the ARSWWW.INI file. See "AFPVIEWING" on page 127 for details. (If you plan to use the Retrieve Document API, then you should specify the `_afp=XENOS` parameter. See "Changes to the Retrieve Document API" on page 135 for details.) To convert Metacode documents with the Xenos transform, you must also specify the `METAVIEWING=XENOS` parameter in the DEFAULT BROWSER section (or other browser sections) of the ARSWWW.INI file. See "METAVIEWING" on page 128 for details. (If you plan to use the Retrieve Document API, then you should specify the `_meta=XENOS` parameter. See "Changes to the Retrieve Document API" on page 135 for details.)

Specifying the ARSXENOS.INI file

The following is an example of an ARSXENOS.INI file:

```
[CREDIT-CREDIT]
ParmFile=c:\documorph\afp2pdf\sample.par
ScriptFile=c:\documorph\noindex.dms
LicenseFile=c:\program files\documorph\dmlic.txt
OutputType=pdf

[default]
ParmFile=c:\documorph\afp2pdf\sample.par
ScriptFile=c:\documorph\noindex.dms
LicenseFile=c:\program files\documorph\dmlic.txt
OutputType=pdf
AllObjects=0
WarningLevel=4
```

The structure of the file is similar to a Windows INI file, and contains one stanza for each OnDemand application and one default stanza. The title line of the stanza identifies the application group and application. For example, the title line:

```
[CREDIT-CREDIT]
```

Identifies the CREDIT application group and the CREDIT application. Use the – (dash) character to separate the names in the title line. The names must match the application group and application names in OnDemand. If the application group contains more than one application, then create one stanza for each application.

The parameters that you specify in the `[default]` stanza are used by the Xenos transform to process documents from applications that are not identified in the ARSXENOS.INI file. The default parameters are also used if an application stanza does not include one of the parameters.

The `ParmFile` parameter identifies the full path name of the file that contains the parameters that are used by the Xenos transform to convert the document. See the Xenos documentation for details about the parameters that you can specify in the parameter file. “Example of a parameter file” on page 131 shows an example of a parameter file.

The `ScriptFile` parameter identifies the full path name of the file that contains the script statements that are used by the Xenos transform to create the output file. See the Xenos documentation for details about the script file. “Example of a script file” on page 133 shows an example of a script file.

The `LicenseFile` parameter identifies the full path name of a valid license file that you obtained from Xenos.

The `OutputType` parameter determines the type of conversion that the Xenos transform performs. You must set this parameter to PDF.

The `AllObjects` parameter determines how the WEK will process documents that are stored as large objects in OnDemand. If you specify `0` (zero), then the WEK will retrieve only the first segment of a document. If you specify `1` (one), then the WEK will retrieve all of the segments and convert them before sending the document to the client. **Note:** If you enable large object support for very large documents, then your users may experience a significant delay before they can view the document at the client.

The `WarningLevel` parameter determines how the WEK will handle return codes from the Xenos transform. The Xenos transform sets a return code after each document is converted. Use this parameter to specify the maximum return code that the WEK will consider good and send the converted document to the client. For example, if you specify `4` (four), then the return code that is set by the Xenos transform must be four or less; otherwise, the WEK will not send the converted document to the client. The default value is zero. If you do not specify this parameter, then the Xenos transform must set a return code of zero. Otherwise, the WEK will not send the converted document to the client. See the Xenos documentation for details about return codes.

Viewing converted documents

To view converted documents with the Adobe Acrobat plug-in, you must obtain the plug-in for the browsers that are used by your organization.

Example of a parameter file

Figure 20 on page 132 shows an example of a parameter file that is used by the Xenos transform to convert AFP documents that are stored in OnDemand to PDF documents that the WEK sends to the Web browser.

The script variables `Parser` and `Generator` determine the type of conversion taking place. The script variable `Parser` represents the input file format and the script variable `Generator` represents the output file format. To perform an AFP to PDF conversion, specify the following:

```
scriptvar=('Parser', 'AFP')
scriptvar=('Generator', 'PDF')
```

To perform a Metacode to PDF conversion, specify the following:

```
scriptvar=('Parser', 'META')
scriptvar=('Generator', 'PDF')
```

```
/* Sample processing parameters for the Xenos transform */

JS:

/* DM Script Library - XG supplied functions */
fddmslib = 'c:\program files\documorph\dmsl.lib'

scriptvar = ('Parser', 'AFP')
scriptvar = ('Generator', 'PDF')

AFPDL-AFPP:
/* AFP Parser Options */
formdef = f1a10111
pagedef = p1a06462
CC      = on
trc     = off
startpage = 0
stoppage  = 0
native    = no
position  = word

/* File Defs */
FDpagesegs  = 'c:\Documorph\Resources\afp\%s.psg'
FDafpfonts  = 'c:\Documorph\Resources\afp\%s.fnt'
FDpagedefs  = 'c:\Documorph\Resources\afp\%s.pde'
FDformdefs  = 'c:\Documorph\Resources\afp\%s.fde'
FDoverlays  = 'c:\Documorph\Resources\afp\%s.ovr'

FDfontcor  = 'c:\documorph\newfont.tab'

PDFGEN-PDFOUT:

/* PDF Out Generator Options */
offset     = (0,0)
scaleby    = 100
border     = NONE
compress = (NONE,NONE,NONE)
orient     = AUTO
pdfauthor  = 'Xenos Group'
pdfopenact = '/FitH 800'
bmorder = (AsIs,AsIs,AsIs)
```

Figure 20. Sample Parameters for the Xenos Transform

Example of a script file

Figure 21 shows an example of a script file that is used by the Xenos transform to generate the output file that the WEK sends to the Web browser. **Note:** The script file does not create an index file, because an index file is not needed for this step.

```
TRUE = 1;
FALSE = 0;

call dm_Initialize

par_h = dm_StartParser(Parser);
gen_h = dm_StartGenerator(Generator);

rc = dm_SetParm(par_h, 'fdinput', inputfile);

/* start the DASD Distributors */
dasd_h = dm_StartDistributor("DASD");

/* open output file */
rc = dm_DASDOpen(dasd_h, '{GROUPFILENAME}'outputfile);


/* initialize */
file_open = FALSE

dlpage = dm_GetDLPage(gen_h);
```

Figure 21. Sample JS Program Script File (Part 1 of 2)

```
do while(dlpage <> 'EOF')
  if file_open = FALSE then do
    if(generator = 'PDF') then do
      rc = dm_PDFGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'AFP') then do
      rc = dm_AFPGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    else if(generator = 'META') then do
      rc = dm_METAGenOpen(gen_h, '{GROUPFILEENTRY}'outputfile);
    end
    file_open = TRUE
  end

  if(generator = 'PDF') then do
    rc = dm_PDFGenWrite(gen_h, dlpage );
  end
  else if(generator = 'AFP') then do
    rc = dm_AFPGenWrite(gen_h, dlpage );
  end
  else if(generator = 'META') then do
    rc = dm_METAGenWrite(gen_h, dlpage );
  end

  dlpage = dm_GetDLPage(par_h);
end

if file_open = TRUE then do
  if(generator = 'PDF') then do
     rc = dm_PDFGenClose( gen_h )
  end
  else if(generator = 'AFP') then do
     rc = dm_AFPGenClose( gen_h )
  end
  else if(generator = 'META') then do
     rc = dm_METAGenClose( gen_h )
  end
end

rc = dm_DASDClose( dasd_h )
return;
```

Figure 21. Sample JS Program Script File (Part 2 of 2)

Changes to the Retrieve Document API

This section describes the changes that were made to the Retrieve Document API to support the Xenos transform.

1. The **_afp** parameter now accepts a value of **XENOS**. When you specify **_afp=XENOS**, the WEK calls the Xenos transform to convert the AFP document that was retrieved from the system into a PDF document that will be sent to the Web browser.
2. The **_meta** parameter was added. You can use the **_meta** parameter to specify the transform that should take place when the WEK retrieves a Metacode document from the system. You can specify one of the following values:

   NATIVE The WEK extracts and will uncompress the Metacode document and its resources from OnDemand.

   Note: If you specify `_meta=NATIVE`, then verify that the MIME content type identifies the viewer that you want to use (see "Specifying a MIME content type for Metacode" on page 127 for more information).

   XENOS The WEK calls the Xenos transform to convert the Metacode document to PDF.

Part 4. Appendixes

Notices

Copyright statement

Trademarks

AFP, AIX, CICS/ESA, DB2, DB2 Universal Database, IBM, IBM @server, Infoprint, OS/2, OS/390, pSeries, and xSeries are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Adobe, the Adobe logo, Acrobat, and the Acrobat logo are trademarks of Adobe Systems Incorporated, which may be registered in certain jurisdictions.

Microsoft, Windows, Windows NT are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

Other company, product, and service names may be trademarks or service marks of others.

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.