Integrating IBM Tivoli Workload Scheduler and Content Manager OnDemand to Provide Centralized Job Log Processing

- Solution to integrate ITWS Suite and Content Manager OnDemand
- Quick and centralized access to ITWS and ITWS for z/OS logs
- Scripts provided for additional customization

Vasfi Gucer
Cheryl Brown
Budi Darmawan
Doroti Almeida Dias Garcia
Tina Lamacchia
Martin Pepper
Pete Soto
Michael R Tucci
Bob Watling

ibm.com/redbooks
Integrating IBM Tivoli Workload Scheduler and Content Manager OnDemand to Provide Centralized Job Log Processing

October 2003
## Contents

**Figures** ................................................................. vii
**Tables** ................................................................ xi
**Notices** ................................................................. xiii
**Trademarks** ......................................................... xiv
**Preface** ................................................................. xv
  The team that wrote this redbook ........................................ xv
  Become a published author ........................................... xvii
  Comments welcome ................................................... xvii

**Chapter 1. Overview and benefits** ................................... 1
  1.1 IBM DB2 Content Manager OnDemand overview ............... 3
  1.2 Servers ................................................................ 5
  1.3 OnDemand concepts ................................................... 6
    1.3.1 Definition files ................................................ 6
    1.3.2 Indexing methods ............................................. 8
    1.3.3 OnDemand Web Enablement Kit .......................... 12
  1.4 IBM Tivoli Workload Scheduler overview ....................... 13
  1.5 IBM Tivoli Workload Scheduler concepts ........................ 16
    1.5.1 IBM Tivoli Workload Scheduler for z/OS. ............... 17
    1.5.2 IBM Tivoli Workload Scheduler for Distributed ....... 18
    1.5.3 IBM Tivoli Workload Scheduler in an end-to-end configuration .... 20

**Chapter 2. Integrating OnDemand and IBM Tivoli Workload Scheduler** .... 23
  2.1 Introduction .......................................................... 24
  2.2 The business case of integration ................................... 25
    2.2.1 Benefits of ITWS/OnDemand integration in a nutshell .... 26
  2.3 Distributed environment ............................................. 27
    2.3.1 Data to be collected ......................................... 28
    2.3.2 Data collection process ....................................... 28
  2.4 z/OS and end-to-end environment ................................... 29
    2.4.1 Data to be collected ......................................... 29
    2.4.2 Data collection process ....................................... 30

**Chapter 3. IBM Tivoli Workload Scheduler for z/OS implementation** ........ 31
  3.1 Overview ............................................................. 32
  3.2 Architecture ........................................................ 34
TWSmerge script: tws_ondemand_twsmerge.sh ................................. 207
Database script: tws_ondemand_database.sh (pull method) ................. 208
Plan script: tws_ondemand_plan.sh ............................................ 208
Windows scripts ........................................................................ 209
tws_ondemand_netman.cmd ......................................................... 209
tws_ondemand_twsmerge.cmd ....................................................... 209
tws_ondemand_database.cmd ....................................................... 210
tws_ondemand_plan.cmd ............................................................ 210

Appendix B. Flowcharts ............................................................... 213
Environmental variables ............................................................. 215
Push method job stdlist real-time or batch mode ................................. 216
Push method batch mode netman, TWSmerge, database, and plan .......... 218
Push method batch mode ............................................................. 221
Pull method monitor ................................................................. 222
Pull method event.log and real time .............................................. 223
Pull method convert logs ............................................................. 226
Pull method UNIX netman .......................................................... 234
Pull method UNIX TWSmerge ..................................................... 235
Pull method UNIX database audit .................................................. 236
Pull method UNIX plan audit ......................................................... 237
Pull method Windows Netman ...................................................... 238
Pull method Windows TWSmerge ................................................ 239
Pull method Windows database audit ............................................. 240
Pull method Windows plan audit ................................................. 241

Appendix C. Additional material .................................................. 243
Locating the Web material .......................................................... 243
Using the Web material ............................................................. 243
  System requirements for downloading the Web material .................. 244
  How to use the Web material ..................................................... 244

Abbreviations and acronyms ........................................................ 245

Related publications ................................................................. 247
IBM Redbooks ......................................................................... 247
Other publications ..................................................................... 247
Online resources ....................................................................... 248
How to get IBM Redbooks .......................................................... 248

Index ......................................................................................... 249
Figures

1-1 Typical OnDemand system ................................................. 4
1-2 Relationship between folders, application groups, and applications . . . 8
1-3 Applications and documents ............................................. 10
1-4 IBM Tivoli Workload Scheduler for z/OS structure ...................... 18
1-5 IBM Tivoli Workload Scheduler structure (distributed) ............... 20
1-6 IBM Tivoli Workload Scheduler end-to-end structure ................... 22
3-1 OnDemand for z/OS architecture ........................................ 33
3-2 OnDemand and TWS for z/OS (one JES domain) schematic ............... 34
3-3 OnDemand and TWS for z/OS (two or more JES domains) schematic . 35
3-4 TWS4ZOS.ZIP file structure .............................................. 41
3-5 OnDemand administrator panel ............................................ 45
3-6 ARSYSPIN (APKACIF) exit creation job flow ............................. 48
3-7 MSGCLASS files that have been collected for a given day ............... 64
3-8 Supported DB and Planning reports for TWS for z/OS .................... 72
3-9 Example OnDemand screen showing the TWS for z/OS reports ........... 73
3-10 OnDemand screen showing the TWS for z/OS EQMLOGS collected .... 76
4-1 Our lab environment for the ITWS/OnDemand integration solution .... 80
4-2 TWS4DIST.ZIP file structure ............................................. 85
4-3 Scenario for each CPU pushing data to the OnDemand Server .......... 110
4-4 Scenario showing a combination of Push and Pull methods ............... 111
5-1 ITWS and OnDemand integration (ITWS end-to-end environment) .... 115
6-1 Starting the OnDemand client program .................................. 119
6-2 Logging on to the OnDemand Server .................................... 120
6-3 Open a Folder window ..................................................... 122
6-4 Search Criteria and Document List dialog box .......................... 123
6-5 Wildcard usage example ................................................... 129
6-6 Searching for documents operation: date and time range ............... 130
6-7 Searching for documents operation: a certain date ..................... 131
6-8 Sort operation ............................................................... 132
6-9 Print dialog ................................................................. 136
6-10 TWSMERGE logs .......................................................... 138
6-11 Viewing the TWSMERGE log ............................................ 139
6-12 Selecting the next page ................................................... 140
6-13 Save a Named Query ...................................................... 141
6-14 Select a Named Query .................................................... 142
6-15 Find operation ............................................................. 143
6-16 Adding a note to a document ............................................. 146
6-17 Viewing a note ............................................................. 148
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-44</td>
<td>Pull method UNIX netman flowchart</td>
<td>235</td>
</tr>
<tr>
<td>7-45</td>
<td>Pull method UNIX twsmerge flowchart</td>
<td>236</td>
</tr>
<tr>
<td>7-46</td>
<td>Pull method UNIX audit database flowchart</td>
<td>237</td>
</tr>
<tr>
<td>7-47</td>
<td>Pull method UNIX plan audit flowchart</td>
<td>238</td>
</tr>
<tr>
<td>7-48</td>
<td>Pull method Windows Netman flowchart</td>
<td>239</td>
</tr>
<tr>
<td>7-49</td>
<td>Pull method Windows TWSmerge flowchart</td>
<td>240</td>
</tr>
<tr>
<td>7-50</td>
<td>Pull method Windows database audit</td>
<td>241</td>
</tr>
<tr>
<td>7-51</td>
<td>Pull method Windows plan audit</td>
<td>242</td>
</tr>
</tbody>
</table>
Integrating IBM Tivoli Workload Scheduler Suite and Content Manager OnDemand
## Tables

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Additional documentation references</td>
<td>37</td>
</tr>
<tr>
<td>3-2</td>
<td>APKACIF exit condition code handling</td>
<td>51</td>
</tr>
<tr>
<td>3-3</td>
<td>ARSYSPIN keywords and descriptions</td>
<td>52</td>
</tr>
<tr>
<td>3-4</td>
<td>ARSYSPIN JCL explained</td>
<td>61</td>
</tr>
<tr>
<td>6-1</td>
<td>Search operators</td>
<td>126</td>
</tr>
<tr>
<td>6-2</td>
<td>Search wildcards</td>
<td>128</td>
</tr>
<tr>
<td>6-3</td>
<td>T format string</td>
<td>133</td>
</tr>
<tr>
<td>6-4</td>
<td>Examples of using the T format string</td>
<td>133</td>
</tr>
<tr>
<td>6-5</td>
<td>Options to view documents when AutoView is selected</td>
<td>137</td>
</tr>
<tr>
<td>7-1</td>
<td>Date format symbols</td>
<td>164</td>
</tr>
</tbody>
</table>
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.
# Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

<table>
<thead>
<tr>
<th>Advanced Function Presentation™</th>
<th>IBM®</th>
<th>Print Services Facility™</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFP™</td>
<td>iSeries™</td>
<td>Redbooks™</td>
</tr>
<tr>
<td>AIX®</td>
<td>Lotus®</td>
<td>Redbooks (logo)™</td>
</tr>
<tr>
<td>AS/400®</td>
<td>Maestro™</td>
<td>S/390®</td>
</tr>
<tr>
<td>CICS®</td>
<td>MVS™</td>
<td>Tivoli Enterprise™</td>
</tr>
<tr>
<td>DB2 Universal Database™</td>
<td>NetView®</td>
<td>Tivoli Enterprise Console®</td>
</tr>
<tr>
<td>DB2®</td>
<td>Notes®</td>
<td>Tivoli®</td>
</tr>
<tr>
<td>@server™</td>
<td>OS/390®</td>
<td>VTAM®</td>
</tr>
<tr>
<td>Infoprint®</td>
<td>OS/400®</td>
<td>WebSphere®</td>
</tr>
</tbody>
</table>

The following terms are trademarks of other companies:

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

This IBM Redbook implements a solution that integrates IBM® Tivoli® Workload Scheduler and IBM DB2® Content Manager OnDemand products to provide centralized output processing for job logs (job outputs, message files, and audit files) from IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduler for z/OS. The solution provides immediate benefit by integrating the job logs into an online, electronic information archive and retrieval system, which is used for quick search and problem resolution purposes.

As part of the solution, we cover defining and implementing the required infrastructure, to access these reports from a simple user interface. We also include examples and scenarios for using this solution.

We include all scripts that make up this solution so that you will be able customize the solution according to your needs.

We anticipate that the solution covered in this book will provide great value for IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduler for z/OS customers who are planning to deploy a centralized job logging and browsing system.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Austin Center.

Vasfi Gucer is an IBM Certified Consultant IT Specialist working at the International Technical Support Organization, Austin Center. He worked with IBM Turkey for 10 years, and has been with the ITSO since January 1999. He has more than 10 years of experience in systems management, networking hardware, and distributed platform software. He has worked on various Tivoli customer projects as a Systems Architect in Turkey and the U.S. Vasfi is also a Certified Tivoli Consultant.

Cheryl Brown is an independent consultant for the IBM DB2 Content Manager OnDemand for Multiplatforms product. She is an IBM retiree who became a contractor working on the development of OnDemand education for seven years, beginning in 1996. In addition to education, she has also provided consulting to IBM customers both in the US and abroad.
Budi Darmawan is a Tivoli Specialist at the International Technical Support Organization, Austin Center. He writes extensively and teaches IBM classes worldwide on Tivoli, DB2 database, and OS/390®. Before joining the ITSO in February 1999, he worked in IBM Global Services, Indonesia as the lead solution architect for Tivoli system management and business intelligence services. Budi is also a Tivoli Certified Instructor and a Tivoli Certified Enterprise Consultant.

Doroti Almeida Dias Garcia is IT Specialist working in Tivoli Customer Support in Brazil. She has three years of experience with Tivoli products, focusing on Tivoli Workload Scheduler and User Administration. She holds a degree in Mathematical Science and she has an MBA in E-Management Information Technology from Fundação Getulio Vargas. Her areas of expertise includes the AIX® operational system.

Tina Lamacchia has been in the IT realm for 18 years. Her career started with programming, and then administrating hundreds of UNIX® systems. She has been with Tivoli Systems for six years. She has worked with IBM Tivoli Workload Scheduler for nine years as a customer and as an IBM Tivoli Workload Scheduler advocate. Currently, she is working in the SWAT team in Austin to assist with sales of various deployments in IBM Tivoli Workload Scheduler and all PACO products.

Martin Pepper does EMEA pre-sales technical support for OnDemand for Multiplatforms and is based out of Warwick in the United Kingdom. He has four years of experience in the Content Management field. He holds a Masters degree in Computer Science from the University of Manchester. Martin is an IBM certified AIX system administrator and a certified solution expert for OnDemand for Multiplatforms.

Pete Soto is currently the L2 Senior Software Engineer for IBM Tivoli Workload Scheduler. He has worked for IBM for nine years in training, implementing, and supporting IBM Tivoli Workload Scheduler. He also supported the Tivoli Software Distribution and Tivoli Inventory products. He is Tivoli Certified for Tivoli Framework, Tivoli Software Distribution, Tivoli Inventory, and Tivoli Workload Scheduler. He has a BBA with a Major in Accounting and a Minor in Information Systems from the University of Texas at San Antonio.

Michael R. Tucci started with IBM Tivoli Software in Austin in 1999 as an L2 Software Engineer for IBM Tivoli Workload Scheduler. He has done extensive travel to customer sites as well as spent time in Rome, Italy testing IBM Tivoli Workload Scheduler 8.2 in a Windows® environment. He also assisted with the completion of the IBM Tivoli Workload Scheduler Installation Guide. Currently, he is responsible for Austin’s L2’s test environment with Windows, UNIX and Linux.

Bob Watling is a Systems Management Technical Consultant based in the United Kingdom with more than 30 years of Systems Management experience.
He has been the IBM representative at the UK-based OPC user group for more than 15 years and has accumulated a vast amount of knowledge and experience over this period of time in the area of batch management.

Thanks to the following people for their contributions to this project:

Wade Wallace
International Technical Support Organization, Austin Center

Wei-Dong Jackie Zhu
International Technical Support Organization, San Jose Center

Robert Haimowitz
International Technical Support Organization, Poughkeepsie Center

Jackie Biggs, Gregory Felderman, Warren Gill, Louise E. Hawley
IBM USA

Fabio Barillari, Maria Pia Cagnetta, Antonio Di Cocco, Riccardo Colella, Pietro Iannucci, Valeria Perticara
IBM Italy

**Become a published author**

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

**Comments welcome**

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:
Use the online Contact us review redbook form found at:

ibm.com/redbooks

Send your comments in an Internet note to:

redbook@us.ibm.com

Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. JN9B Building 003 Internal Zip 2834
11400 Burnet Road
Austin, Texas 78758-3493
Overview and benefits

In this chapter, we provide an overview of the IBM DB2 Content Manager OnDemand (OnDemand) system and the IBM Tivoli Workload Scheduler product. We also provide information that can help you better understand how these products work. This information is important, because you have to have a good understanding of both products to be able to successfully implement the IBM DB2 Content Manager OnDemand and IBM Tivoli Workload Scheduler integration solution (hereafter referred to as the ITWS/OnDemand integration solution). This integration is the main topic of this book. If you already have experience with both products, you can skip this chapter and go to Chapter 2, “Integrating OnDemand and IBM Tivoli Workload Scheduler” on page 23.

First, we describe how OnDemand manages reports and indexes data. We have included important information about how OnDemand, its database manager, and the storage manager work together to index, load, and retrieve reports and log files. Also, we provide a list of tasks that OnDemand administrators typically perform to manage an OnDemand system.

Second, we describe how IBM Tivoli Workload Scheduler is structured and how it can schedule your entire enterprise.

The following sections will be covered in this chapter:

- Section 1.1, “IBM DB2 Content Manager OnDemand overview” on page 3
- Section 1.2, “Servers” on page 5
- Section 1.3, “OnDemand concepts” on page 6
Section 1.4, “IBM Tivoli Workload Scheduler overview” on page 13
Section 1.5, “IBM Tivoli Workload Scheduler concepts” on page 16
1.1 IBM DB2 Content Manager OnDemand overview

IBM DB2 Content Manager OnDemand is an archival and retrieval system for documents, log files, and reports. It provides online access to hard copy and is an excellent tool for microfiche replacement, providing instant access to information. An OnDemand system can support small office environments and large enterprise installations with hundreds of system users. It can dramatically improve productivity and customer service in many businesses by providing fast access to information stored in the system.

OnDemand processes the print output of application programs, extracts index fields from the data, stores the index information in a relational database, and stores one or more copies of the data in the system. With OnDemand, you can archive newly created and frequently accessed reports on high speed, disk storage volumes and automatically migrate them to other types of storage volumes as they age.

OnDemand fully integrates the capabilities of the Advanced Function Presentation™ (AFP™) data stream, including management of resources, indexes, and annotations, and supports full fidelity reprinting and faxing of files to devices attached to a PC, OnDemand Server, or other server on the network.

OnDemand provides administrators with tools to manage OnDemand Servers, authorize users to access OnDemand Servers and data stored in the system, and back up the database and data storage.

OnDemand provides users the ability to view, print, send, attach electronic notes, and fax copies of log files and reports.

OnDemand offers several advantages to IBM Tivoli Workload Scheduler users; it can:

- Easily locate data without specifying the exact report.
- Retrieve the pages of the report that you need without processing the entire report.
- View selected data from within a report.

Note: Part of this chapter has been taken from the redbook Content Manager OnDemand Guide, SG24-6915. For more information on IBM DB2 Content Manager OnDemand, you can refer to this redbook.
OnDemand provides an information management tool that will increase your effectiveness when working with customers. It achieves this by doing the following:

- Integrates data created by application programs into an online, electronic information archive and retrieval system.
- Provides controlled and reliable access to all reports of an organization.
- Retrieves data when needed.
- Provides a standard, intuitive client with features such as thumbnails, bookmarks, notes, and shortcuts.

These features mean that OnDemand can help you quickly retrieve the specific page of a report that you need to provide fast customer service.

An OnDemand system consists of client programs and server programs that communicate over a network running the TCP/IP communications protocol. An example of a typical OnDemand system is shown in Figure 1-1.

![Figure 1-1 Typical OnDemand system](image)

**OnDemand client programs**

OnDemand client programs run on CICS® terminals and Windows operating system PCs attached to a network communicating with the OnDemand Servers. Using the client program, users construct queries to initiate the search and then are presented with a list of documents that match the query. Once a document list has been displayed, users have the following options:
1.2 Servers

The OnDemand Server environment includes a library server and one or more object servers residing on one or more nodes connected to a TCP/IP network. An OnDemand Server is supported in one of the following operating systems:

- AIX
- Sun OS
- HPUX
- OS/400®
- z/OS®

OnDemand library server
The OnDemand library server manages control information in the database about the OnDemand users and the reports stored on the system. The database manager provides the database engine and utilities to administer the database. The library server processes client logons, queries, and print requests and updates to the database. The major functions that run on the library server are the request manager, the database manager, and the server print manager.

**Note:** An OnDemand system consists of only one library server.

OnDemand object server
The OnDemand object server maintains documents on cache storage volumes and, optionally, works with an archive storage manager to maintain documents on archive media, such as optical and tape storage libraries. An object server loads data, retrieves documents, and expires documents. The major functions that run on an object server are the cache storage manager, OnDemand data loading and maintenance programs, and optionally, the archive storage manager.

An OnDemand system consists of one or more object servers. An object server can operate on the same server as the library server, or on a separate server than the library server.

When a user submits a query, the client program sends a search request to the OnDemand library server. The library server returns a list of the documents that match the query to the user. When the user selects a document for viewing, the
client program retrieves a copy of the document from the object server where the document is stored, opens a viewing window, and displays the document.

1.3 OnDemand concepts

In order to load and view report files or log files, OnDemand definition files must be created. The terms application, application group, and folder represent these definition files. It is in these files that the determination is made as to how OnDemand stores, manages, retrieves, views, and prints reports and index data. When defining a new report or type of data to OnDemand, an administrator must create an application and assign the application to an application group. You may define one or more applications per application group.

**Note:** If an application group does not exist, the administrator must first create one.

Before users can search for and retrieve documents, an administrator must create or update a folder to use the application group and application.

1.3.1 Definition files

The following gives an introduction to the OnDemand definition files.

**Application**

Each application represents a unique report type. An application defines the following areas to OnDemand:

- **File characteristics** Describes the input file characteristic, that is, format of the data, the orientation of data on the page, the paper size, the record length, and the code page.
- **Index information** Includes parameters used by the indexing program to locate and extract index data.
- **Load instructions** Includes both pre- and post-processing parameters when loading index data in the database, date format settings, and data compression selection.
**Application group**

When searching for a report or log file, a SQL query is run against an application group database table. An application group defines three key areas to OnDemand:

- **Database tables**
  This includes retention of the indexes, database field attributes as well as database organization, table length, and size.

- **Data storage**
  This includes retention period of the data for short-term data storage and interfacing with storage manager for long-term data storage.

- **Access**
  Assigns access permissions to users or groups of users to query the database and access the report files.

When a report is loaded into OnDemand, you must identify the application group where OnDemand will load the index data and store the documents. An application group is a collection of one or more OnDemand applications with common indexing and storage management attributes. You typically group several different reports in an application group so that users can access the information contained in the reports with a single query.

**Folder**

A folder provides users with a convenient way to find related information stored in OnDemand, regardless of the source of the information or how the data was prepared. A folder is similar to a directory structure in a server where the files within that directory are virtual files, that is, a link to the actual physical location of the files are accessible. The following areas are defined through the folder:

- **Query fields**
  Defines which database fields within the application group will be searched and all attributes about those fields, for example, default value and search operators (*Equal To* or *Less Than*, and so on)

- **Access**
  Assign access permissions to users or groups of users to query the database and access the report files.

A folder allows an administrator to set up a common query screen for several application groups that may use different indexing schemes so that a user can retrieve the data with a single query. For example, a folder called Log Files might contain NETMAN, JOBMAN, and MAILMAN entries in addition to Standard List Log Files, which represents information stored in different application groups, defined in different applications, and created by different programs.

Figure 1-2 on page 8 illustrates the concepts described in this section.
1.3.2 Indexing methods

OnDemand provides two methods of indexing data:

- **Document indexing** is used for reports that contain logical items, such as log file information. Each of the items in a report can be individually indexed on values, such as server job number, job name, start date and time, and end date and time. With document indexing, the user does not necessarily need to know about reports or report cycles to retrieve a document from OnDemand.

- **Report indexing** is used for reports that contain many pages of the same kind of data, such as a merge transaction log. Each line in the report usually identifies a specific transaction, and it would not be cost effective to index each line. OnDemand stores the report as groups of pages and indexes each group. When reports include a sorted transaction value (for example, start date), OnDemand can index the data on the transaction value. This is done by extracting the beginning and ending transaction values for each group of pages and storing the values in the database. This type of indexing lets users retrieve a specific transaction value directly.

The reports that are stored in OnDemand must be indexed. OnDemand supports several types of index data and indexing programs. The AFP Conversion and Indexing Facility (ACIF) can be used to index S/390® line data, ASCII data, and AFP files, collect resources required to view the reports, and convert line data.
files to AFP data. The OS/400 Indexer can be used to index a variety of data types and is the most common OnDemand indexer for OS/400 spooled files. The OnDemand PDF Indexer can be used to create index data for Adobe Acrobat PDF files. The OnDemand Generic Indexer can be used to create index data for almost any other type of data, such as HTML documents, Lotus® WordPro documents, and TIFF files.

**Note:** The ITWS/OnDemand solution uses ACIF. See “Indexer information” on page 161 for a more detailed explanation about indexing.

For OS/400, there is the OS/400 Indexer that can index a variety of data types for OS/400 spooled files. Refer to *IBM Content Manager OnDemand for Multiplatforms Indexing Reference Version 7.1*, SC27-0842, *IBM Content Manager OnDemand for iSeries Common Server: Indexing Reference*, SC27-1160, and *IBM Content Manager OnDemand for z/OS and OS/390: Indexing Reference*, SC27-1375 for details about the indexing programs provided with OnDemand for various platforms.

**Documents**

OnDemand documents represent indexed groups of pages. Typically, an OnDemand document is a logical section of a larger report, such as a log file within a report of hundreds of log files. An OnDemand document can also represent a portion of a larger report. For reports that do not contain logical groups of pages, such as merge transaction logs, OnDemand can divide the report into groups of pages. The groups of pages are individually indexed and can be retrieved much more efficiently than the entire report. Documents are usually identified by date, with one or more other fields, such as CPU name.

Figure 1-3 on page 10 illustrates OnDemand applications and documents. An administrator could define the Log File application for a report that contains logical items. The Log File application uses the document indexing method to divide the report into documents. Each log file in the report becomes a document in OnDemand. Users can retrieve a statement by specifying the date and any combination of name and number.
An administrator could define the MERGE FILE application for a report that contains lines of sorted transaction data. The MERGE FILE application uses the report indexing method to divide the report into documents. Each group of 100 pages in the report becomes a document in OnDemand. Each group is indexed using the first and last sorted transaction values that occur in the group (in the merge file the start date is used as a transaction value). Users can retrieve the group of pages that contains a specific start date and specifying the CPU name. OnDemand retrieves the group that contains the value entered by the user.

The basic OnDemand configuration is a library server and an object server on the same physical system or node. This single library/object server configuration supports the database functions and cache storage on one system. You can add an archive storage manager to the single library/object server configuration to maintain documents on archive media.

You can also configure your OnDemand system with a library server on one node and one or more object servers on different nodes. This configuration is known as a distributed library/object server system. The distributed library/object server configuration supports caching of documents on different servers. You can add an archive storage manager to one or more of the object servers to maintain documents on archive media attached to different servers.
An OnDemand Server environment contains several components:

- A request manager that provides client, network, and operating system services, security, and accounting. The request manager resides on the library server.
- A database manager that maintains the index data for the reports that you store on the system. The database manager is a relational database management product, such as DB2. The database manager resides on the library server.
- Database control information about the users, groups, application groups, applications, folders, storage sets, and printers that you define on the system. The control information determines who can access the system, the folders that a user can open, and the application group data that a user can query and retrieve. The database resides on the library server.
- A cache storage manager that maintains documents in cache storage. Cache storage is for high-speed access to the most frequently used documents.
- An archive storage manager, which is an optional part of the system. The archive storage manager is for the long-term storage of one or more copies of documents on archive media, such as optical and tape storage libraries.
- A download facility that automatically transfers spool files to a server at high speed. We recommend that you use Download for OS/390, a licensed feature of Print Services Facility™ (PSF) for OS/390. Download provides the automatic, high-speed download of JES Spool files from an OS/390 system to OnDemand Servers. The download facility is not applicable to iSeries™.
- Data loading programs that can be set up to automatically store report data into application groups and update the database. The data loading programs can run on any OnDemand Server.
- Archived reports and resources.
- A server print facility that allows users to reprint a large volume of documents at high speed. OnDemand uses Infoprint®, which must be purchased separately, to manage the server print devices.
- OnDemand management programs to maintain the OnDemand database and documents in cache storage.
- A system logging facility that provides administrators with tools to monitor server activity and respond to specific events as they occur. The interface to the system logging facility is through the system log folder and the system log user exit.

**Management programs**
OnDemand provides programs to maintain and optimize the database and maintain documents on cache storage. An administrator usually determines the
processing parameters for these programs, including the frequency with which the programs should run. When you create an application group, you specify other parameters that these programs use to maintain the report data stored in the application group.

For example, when creating an application group, the administrator specifies how long documents should be maintained on the system and whether index data should be migrated from the database to archive media. The programs use the information to migrate documents from cache storage to archive media, delete documents from cache storage, migrate index data from the database to archive media, and delete index data from the database. These functions are useful because OnDemand can reclaim the database and cache storage space released by expired and migrated data. We recommend that you configure your OnDemand system to automatically start these management programs on a regular schedule, usually once every night or week.

The archive storage manager deletes data from archive media when it reaches its storage expiration date. An administrator defines management information to the archive storage manager to support the OnDemand data it manages. The management information includes the storage libraries and storage volumes that can contain OnDemand data, the number of copies of a report to maintain, and how long to keep data in the archive management system.

OnDemand and the archive storage manager delete data independently of each other. Each uses its own criteria to determine when to remove documents. Each uses its own utilities and schedules to remove documents. However, for final removal of documents from the system, you should specify the same criteria to OnDemand and the archive storage manager.

### 1.3.3 OnDemand Web Enablement Kit

The OnDemand Web Enablement Kit (ODWEK) is an optional feature of OnDemand that allows people to use a Web browser to access data stored in an OnDemand system. For example, you can provide some people with the Uniform Resource Locator (URL) of a Web page that permits them to log on to an OnDemand Server; you can provide other people with the URL of a Web page that permits them to search a specific folder. ODWEK verifies that the user information is valid on the OnDemand Server, such as permission to access the server and data stored in an application group. After the user submits a search, ODWEK displays a Web page that contains a list of the documents that match the query. The user selects a document to view and ODWEK sends the document to the browser.
ODWEK contains several components:

- **The Web server program.** The server program uses standard OnDemand interfaces and protocols to access data stored in an OnDemand Server. No additional code is needed on the OnDemand Server to support ODWEK. You can use one of the following Web server programs to control ODWEK:
  - CGI program. The CGI program runs against an HTTP server, such as the IBM HTTP Server.
  - Java™ servlet. The servlet runs on a Java-enabled HTTP server with a Java application server, such as the IBM WebSphere® Application Server.

- **The AFP Web Viewer.** The AFP Web Viewer lets users search, retrieve, view, navigate, and print AFP documents from a Web browser. The AFP Web Viewer must be installed on the viewing workstations.

- **The Image Web Viewer.** The Image Web Viewer lets users search, retrieve, view, navigate, and print BMP, GIF, JPEG, PCX, and TIFF documents from a Web browser. The Image Web Viewer must be installed on the viewing workstations.

- **The Line Data Java applet.** The Line Data applet lets users view line data documents from a Web browser.

- **The AFP2HTML Java applet.** The AFP2HTML applet lets users view the output generated by the IBM AFP2WEB Transform service offering. The AFP2WEB Transform converts AFP documents and resources into HTML files that can be displayed with the AFP2HTML applet. If you plan to use the AFP2HTML applet, then you must obtain the AFP2WEB Transform from IBM and install and configure it on the Web server.

### 1.4 IBM Tivoli Workload Scheduler overview

The importance of end-to-end business systems management is recognized across the industry; batch workload is the backbone of business applications and therefore drives the business. As a crucial part of business systems, the workload must be managed seamlessly from end to end.

IBM Tivoli Workload Scheduler is the state-of-the-art production workload manager that is designed to help you meet your present and future data processing challenges. Its scope encompasses your entire enterprise information system, including heterogeneous environments.
The workload scheduler suite simplifies systems management across heterogeneous environments by integrating systems management functions. There are three main components to the suite:

- Tivoli Workload Scheduler for z/OS (The scheduler in OS/390 and z/OS environments)
- Tivoli Workload Scheduler (The scheduler in distributed environments)
- Tivoli Job Scheduling Console (The common user interface for both Tivoli Workload Scheduler for z/OS and Tivoli Workload Scheduler)

**Automatic driver**

IBM Tivoli Workload Scheduler provides leading-edge solutions to problems in production workload management. It can automate, plan, and control the processing of your enterprise’s entire production workload, not just the batch subset. IBM Tivoli Workload Scheduler functions as an “automatic driver” for your production workload to maximize the throughput of work and optimize your resources, but also allows you to intervene manually as required. When IBM Tivoli Workload Scheduler interfaces with other system management products, it forms part of an integrated automation and systems management platform for your DP operation.

IBM Tivoli Workload Scheduler can manage the complete flow of work through your enterprise’s entire DP operation, on both local and remote systems.

**Single point of control**

From a single point of control, IBM Tivoli Workload Scheduler analyzes the status of the production work and drives the processing of the workload according to installation business policies. It supports a multiple-end-user environment, enabling distributed processing and control across sites and departments within your enterprise.

**Enterprise application integration**

ITWS provides advanced integration with enterprise applications like SAP R/3, Oracle Applications, and PeopleSoft. ITWS’s underlying architecture ensures that business and mission-critical applications run within an infrastructure that is secure, fault-tolerant, and scalable.

**Systems management integration**

Solutions to today’s systems management problems require an integration of application programs and processes. IBM Tivoli Workload Scheduler interfaces directly with some of the z/OS products as well as with a number of other IBM products to provide a comprehensive, automated processing facility and an integrated approach for the control of complex production workloads.
For example:

- *IBM DB2 Content Manager OnDemand* has the ability to archive not only the output from jobs being processed under the control of IBM Tivoli Workload Scheduler but also the logs and reports produced by the scheduler itself. Using the Windows client or WEB access feature, users can historically search the DB2 database by specifying the same user defined keywords as those used when the data was archived. Searching for archived output by Jobname, date and time of run, completion code, or even freeform strings are all possible.

- Workload Scheduler sends workload plan information and *IBM Tivoli Business Systems Manager* is the solution to unifying the management of business systems. IBM Tivoli Workload Scheduler for z/OS has been enhanced to support monitoring from Tivoli Business Systems Manager. From Tivoli Business Systems Manager, you can monitor status changes to jobs, addition of jobs to the plan, and alert conditions relating to job scheduling.

- Workload Scheduler can initiate software distributions done by *IBM Tivoli Configuration Manager*; Configuration Manager can install and deploy Workload Scheduler Agents.

- Workload Scheduler for z/OS can send job related information to *Information Management* for service related interactions. Information Management supports the administration of the systems management process of an enterprise’s hardware, software, and related resources. An interface with Information Management is provided for reporting problems detected while processing the production workload.

- Workload Scheduler schedules *IBM Tivoli Data Exchange* to manage the customers’ data movement among systems.

- Integrating Workload Scheduler with *IBM Tivoli NetView®* gives network managers the ability to monitor and diagnose Workload Scheduler networks from a NetView management node. It includes a set of submaps and symbols to view scheduler networks topographically, and determine the status of job scheduling activity and critical scheduler processes on each workstation. Menu actions are provided to start and stop scheduler processing, and to run *conman* on any workstation in the network.

- *System Automation for 390 (SA/390)* initiates automation procedures that perform operator functions to manage OS/390 components, data sets, and subsystems. SA/390 includes an automation feature for IBM Tivoli Workload Scheduler for z/OS.

- The *IBM Tivoli Enterprise™ Console (TEC)* is a powerful, rules-based event management application that integrates network, systems, database, and application management. It offers a centralized, global view of your computing
enterprise while ensuring the high availability of your application and computing resources. Tivoli Enterprise Console® acts as a central collection point for alarms and events from a variety of sources, including those from Tivoli applications. IBM Tivoli Workload Scheduler runs a Tivoli Enterprise Console adapter that reads events from the IBM Tivoli Workload Scheduler log file. Enterprise Console in turn can initiate actions in Workload Scheduler based upon the collective correlation of other events.

- Workload Scheduler can initiate and monitor workstation backup and recovery activities by integrating with IBM Tivoli Storage Manager.
- IBM Tivoli Monitoring can report on the status and health of Workload Scheduler’s management processes and queue files.
- IBM Tivoli Decision Support for z/OS helps you effectively manage the performance of your system by collecting performance data in a database and presenting the data in a variety of formats for use in systems management. Decision Support uses data from IBM Tivoli Workload Scheduler for z/OS to produce summary and management reports about the production workload, both planned and actual results.
- Removable Media Manager (RMM) works with restart and cleanup to assist with the management of data sets in a z/OS environment. Removable Media Manager verifies that a data set exists on a volume and then takes user-requested actions on the data set. Consequently, IBM Tivoli Workload Scheduler for z/OS works with Removable Media Manager to properly mark and expire data sets as needed.
- Workload Scheduler will store batch job run statistics such as job duration, CPU usage, and status information in the IBM Tivoli Data Warehouse. There are many predefined reports provided with the Data Warehouse for most common reporting needs.

This document is specifically targeted at the integration between OnDemand and IBM Tivoli Workload Scheduler and/or IBM Tivoli Workload Scheduler for z/OS.

1.5 IBM Tivoli Workload Scheduler concepts

This topic summarizes the differences between IBM Tivoli Workload Scheduler for z/OS, IBM Tivoli Workload Scheduler for Distributed, and IBM Tivoli Workload Scheduler for an end-to-end configuration. For a more detailed explanation, you can refer to the End-to-End Scheduling with Tivoli Workload Scheduler 8.1, SG24-6022 redbook or refer to the IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduler for z/OS product manuals directly.
1.5.1 IBM Tivoli Workload Scheduler for z/OS

IBM Tivoli Workload Scheduler for z/OS has been scheduling and controlling batch workloads in data centers since 1977. Originally called Operations Planning and Control (OPC), the product has been extensively developed and extended to meet the increasing demands of customers worldwide. An overnight workload consisting of 100,000 production jobs is not unusual, and IBM Tivoli Workload Scheduler for z/OS can easily manage this kind of workload.

IBM Tivoli Workload Scheduler for z/OS databases contains information about the work that is to be run, when it should run, and the resources that are needed and available. This information is used to calculate a forward forecast called the long term plan. The data center staff can check this forecast to confirm that the desired work is being scheduled when required. The long term plan usually covers a period of time from four to twelve weeks. A second plan is produced that uses the long term plan and databases as input. The current plan usually covers 24 hours and is a detailed production schedule. IBM Tivoli Workload Scheduler for z/OS uses this to submit jobs to the appropriate processor at the appropriate time. All jobs in the current plan have IBM Tivoli Workload Scheduler for z/OS status codes that indicate the progress of work. When a job's predecessors are complete, IBM Tivoli Workload Scheduler for z/OS considers it ready for submission. It checks that all requested resources are available, and when these conditions are met, it causes the job to be submitted.

Structure

IBM Tivoli Workload Scheduler for z/OS consists of a controller and one or more trackers (Figure 1-4 on page 18). The controller runs on an z/OS system. The controller manages the IBM Tivoli Workload Scheduler for z/OS databases and the long term and current plans. The controller schedules work and cause jobs to be submitted to the appropriate system at the appropriate time. Trackers are installed on every system managed by the controller. The tracker is the link between the controller and the managed system. The tracker submits jobs when the controller instructs it to do so, and it passes job start and job end information back to the controller. The controller can schedule jobs on z/OS systems as well as distributed systems. The IBM Tivoli Workload Scheduler for z/OS controller use fault tolerant workstations for job scheduling on distributed systems. A fault tolerant workstation is actually a IBM Tivoli Workload Scheduler Fault Tolerant Agent. For more details on the use of fault tolerant workstations, see 1.5.3, “IBM Tivoli Workload Scheduler in an end-to-end configuration” on page 20.
1.5.2 IBM Tivoli Workload Scheduler for Distributed

IBM Tivoli Workload Scheduler is descended from the Unison Maestro™ program. Unison Maestro was developed by Unison Software on Hewlett-Packard's MPE operating system. It was then ported to UNIX and Windows. In its various manifestations, IBM Tivoli Workload Scheduler has a 15 year track record. During the processing day, IBM Tivoli Workload Scheduler's production control programs manage the production environment and automate most operator activities. It prepares jobs for execution, resolves interdependencies, and launches and tracks each job. Because jobs begin as soon as their dependencies are satisfied, idle time is minimized and throughput improves significantly. Jobs never run out of sequence, and, if a job fails, IBM Tivoli Workload Scheduler handles the recovery process with little or no operator intervention.

There are two basic aspects to job scheduling in IBM Tivoli Workload Scheduler: the database and the plan. The database contains all the definitions for scheduling objects, such as jobs, job streams, resources, and workstations. It
also holds statistics of job and job stream execution, as well as information on the user ID that created an object and when an object was last modified. The plan contains all job scheduling activity planned for a period of one day. In IBM Tivoli Workload Scheduler, the plan is created every 24 hours and consists of all the jobs, job streams, and dependency objects that are scheduled to execute for that day. All job streams for which you have created a run cycle are automatically scheduled and included in the plan. As the day goes by, the jobs and job streams that do not execute successfully can be rolled over into the next day’s plan.

**Structure**

IBM Tivoli Workload Scheduler consists of a Master Domain Manager (MDM) that contains the centralized database files used to document scheduling objects. It creates the production plan at the start of each day and performs all logging and reporting for the network. All communications to agents are routed through the Domain Manager (DM), which is the management hub in a domain. The network can be managed by a mix of agents. Fault tolerant agents are capable of resolving local dependencies and launching their jobs should a network interruption cause a loss of communication with their Domain Managers, because each one is given a set of scheduling instructions at the beginning of every processing day. IBM Tivoli Workload Scheduler Version 7.0 introduced a new Java GUI, the Job Scheduling Console (JSC). The current version of JSC has been updated with several new specific IBM Tivoli Workload Scheduler functions. The JSC provides a common interface to both IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduler for z/OS.

Figure 1-5 on page 20 shows the IBM Tivoli Workload Scheduler Distributed architecture.
1.5.3 IBM Tivoli Workload Scheduler in an end-to-end configuration

In an end-to-end environment, E2E scheduling directly connects IBM Tivoli Workload Scheduler Domain Managers and their underlying agents and domains to IBM Tivoli Workload Scheduler for z/OS. IBM Tivoli Workload Scheduler for z/OS is seen by the distributed network as the Master Domain Manager.

IBM Tivoli Workload Scheduler for z/OS also creates the production plan for the distributed network and sends it to the Domain Managers. The Domain Managers send a copy of the plan to each of their agents and subordinate Domain Managers for execution.

The IBM Tivoli Workload Scheduler Domain Managers function as the broker systems for the distributed network by resolving all dependencies for their subordinate managers and agents. They send their updates (in the form of events) to IBM Tivoli Workload Scheduler for z/OS so that it can update the plan accordingly. IBM Tivoli Workload Scheduler for z/OS handles its own jobs and notifies the Domain Managers of all the status changes of the IBM Tivoli Workload Scheduler for z/OS jobs that involve the IBM Tivoli Workload Scheduler plan. In this configuration, the Domain Managers and all the distributed agents recognize IBM Tivoli Workload Scheduler for z/OS as the Master Domain Manager.
Manager and notify it of all the changes occurring in their own plans. At the same time, the agents are not permitted to interfere with the IBM Tivoli Workload Scheduler for z/OS jobs, since they are viewed as running on the master that is the only node that is in charge of them.

IBM Tivoli Workload Scheduler for z/OS also allows you to access job streams and add them to the current plan in IBM Tivoli Workload Scheduler for z/OS. In addition, you can build dependencies among IBM Tivoli Workload Scheduler for z/OS job streams and IBM Tivoli Workload Scheduler jobs. From IBM Tivoli Workload Scheduler for z/OS, you can monitor and control the distributed agents.

In the IBM Tivoli Workload Scheduler for z/OS current plan, you can specify jobs to run on workstations in the IBM Tivoli Workload Scheduler network. IBM Tivoli Workload Scheduler for z/OS passes the job information to the Symphony file in the IBM Tivoli Workload Scheduler for z/OS server, which in turn passes the Symphony file to the IBM Tivoli Workload Scheduler Domain Managers (DMZ) to distribute and process. In turn, IBM Tivoli Workload Scheduler reports the status of running and completed jobs back to the current plan for monitoring in the IBM Tivoli Workload Scheduler for z/OS engine.

**Structure**

Figure 1-6 on page 22 shows an end-to-end environment where a IBM Tivoli Workload Scheduler network is managed by IBM Tivoli Workload Scheduler for z/OS.
This concludes our discussion on IBM DB2 Content Manager OnDemand and IBM Tivoli Workload Scheduler. In the next chapter, we will start discussing the ITWS/OnDemand integration solution.

For more information on IBM DB2 Content Manager OnDemand, please refer to the *Content Manager OnDemand Guide*, SG24-6915.

For more information on IBM Tivoli Workload Scheduler, please refer to *End-to-End Scheduling with Tivoli Workload Scheduler 8.1*, SG24-6022.
Chapter 2. Integrating OnDemand and IBM Tivoli Workload Scheduler

In this chapter, we introduce the solution of conceptional integration of centralized log storage on DB2, and discuss this application’s role in the construction of the OnDemand event-driven scripts and data-transfer methodology. We also discuss the benefits of integration of OnDemand and IBM Tivoli Workload Scheduler and the pros and cons of various strategies that can be used for implementing this solution.

This chapter provides the following sections:

- Section 2.1, “Introduction” on page 24
- Section 2.2, “The business case of integration” on page 25
- Section 2.3, “Distributed environment” on page 27
- Section 2.4, “z/OS and end-to-end environment” on page 29
2.1 Introduction

In the past several years, Web-based integration of database access has increasingly attracted attention, mainly because of its ability to let administrators, managers, and end users access information from virtually any workstation with an Internet connection, thus allowing experts to view data and solve problems faster and easier.

OnDemand offers substantially higher control of data sharing and access over the conventional Job Scheduling Console (JSC) software. Data searches can be customized, and visualization of the data is quick and easy. The ITWS/OnDemand integration solution stores the Tivoli Workload Scheduler and IBM Tivoli Workload Scheduled for z/OS logs in an OnDemand database to facilitate all these benefits.

The IBM Content Manager portfolio provides the enterprise content management infrastructure to store, access, and manage the full spectrum of digital information generated in e-business, from vertical line of business applications to customer service, enterprise resource planning and supply chain management, to digital asset management and web content management.

This integration can accelerate business process automation across the enterprise for small, medium, and large e-businesses. The Content Manager portfolio manages and integrates access to various forms of information, and for this exercise, we will discuss the IBM Tivoli Workload Scheduler process outputs.

These are three points of integration:

- Application integration: Bills, statements and transaction report jobs generated on different host systems can be captured in the content repository for central access to all documents. These reports can be available when the TWS jobs are completed successfully.

- Scalable infrastructure: The ability of the high performance content management systems to store literally billions of individual pages according to corporations requirements, which dictates the availability of these documents for time periods often exceeding seven years.

- Document display: Full page display eliminates the requirement for online record oriented systems to access multiple screens. All of the information required to solve a problem or answer a question is contained on a single document. This will eliminate logging into various systems to retrieve standard lists for one or multiple jobs.

Our discussion on strategies will surround a business case and its proven capability to increase productivity while managing information with IBM Tivoli
2.2 The business case of integration

Content management systems deliver proven results. You can link invoice and bill readiness with transaction report data processes, such as IBM Tivoli Workload Scheduler (TWS). It can create a significant return on investment by:

- Increasing productivity by as much as 30 percent
- Eliminating return phone calls by 90 percent
- Enabling customer self service by allowing Internet access to invoices and statements versus paper mail outs

These benefits are as much of a customer relationship management as an e-business document exchange or electronic bill and presentment, as they provide much of the cost justification that is generally ignored in today’s corporate return on investment calculations for these types of systems. The qualitative benefits of an improved business process, superior customer service/self-service, and fast access to data makes the case for content management even more compelling.

Content management systems leverage this fact and simply use the same document format in which data is originally delivered to the customer to index, store, retrieve, and view information. Even though this document content is a key element of customer service, the scheduling community is just starting to exploit and integrate this readily available, off-the-shelf technology. This information black hole exists because most organizations are too focused on the online legacy or enterprise resource planning system database, believing that it is the only available solution to their information delivery problem. They have not recognized the availability of affordable technology and customer-friendly systems that deliver document information in an easily viewable format, the one in which they were originally created, and it is all Web based. Lastly, OnDemand was chosen due to the speed of the document capture, indexing, and Web transformation process. When the system retrieves a bill, it will automatically transform it to HTML, eliminating what would have been time consuming and costly development costs in creating entirely new Internet format software.

The competitive advantage of OnDemand is that, with its robust performance and multi-function object management architecture, it has exceptional capability. The software operates on Windows, NT/2000 and UNIX servers, as well as on OS/390, z/OS, and OS/400 (iSeries), platforms. UNIX servers include Sun Solaris, HP-UX, and IBM AIX. The client software is the same across all platforms, ensuring that the input and retrieval screens are the same, regardless
of the host application server. There is also software for both the IBM iSeries (formerly AS/400®) 5250 and CICS 3270 native terminals to preserve investments.

### 2.2.1 Benefits of ITWS/OnDemand integration in a nutshell

Why would you want to integrate IBM Tivoli Workload Scheduler and OnDemand and what possible benefits could there be?

Let us suppose you wanted to answer some of the following questions:

- Has a particular job or job stream ever been scheduled within the last year?
- Who was the last person to have scheduled a particular on request job or job stream and when?
- Which users have updated a particular job stream in the database over the last month?
- On the last run of a particular job, what was the name of the input file that was used?
- Has a particular error message appeared in any message logs over the last seven days?
- On the last run of this yearly job, did it complete successfully with a zero return code and what was the actual duration?
- How many times in the last month has a particular job run and what was the return code on each execution?
- How can I store all the various outputs from my enterprise scheduler in one central location?

These types of questions are pretty typical and can all be answered using the solution provided by this redbook. In the examples above, we have used last year, month, and 7 days as time frame variables, but of course these are user defined and entered at search time. The only limitation would be ensuring the data you need to be able to search for is kept for the relevant amount of time.

**Note:** In the sample definitions we have provided, the default retention period for IBM Tivoli Workload Scheduler data collected into OnDemand has been set to 365 days. This can be changed for individual reports by the OnDemand administrator using the administration dialogues.

If you are already an OnDemand user, you will already know how simple it is to use the client or optional Web-based dialogue to retrieve the information you require with just a few simple keystrokes.
If you are not already an OnDemand user and would like to find out more, then please contact your local IBM representative to check out how affordable this solution can be.

In the following sections we will investigate different strategies to transport IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduled for z/OS logs in an OnDemand database.

### 2.3 Distributed environment

Our approach for the OnDemand solution in a distributed environment is to join an existing IBM Tivoli Workload Scheduler environment on multiplatforms with a dedicated DB2 server. The OnDemand concept will work in conjunction with the IBM Tivoli Workload Scheduler environment to pull IBM Tivoli Workload Scheduler logs from every IBM Tivoli Workload Scheduler Fault Tolerant Agent (FTA) and push these logs into a centralized storage system, organized by DB2, as they are created or configured to transfer the data at specified times. There will also be a discussion on when it is the best time to use the Push methodology as well.

In the following section, we introduce OnDemand's programming model and give a sketch of its architecture.

OnDemand will utilize a series of scripts executed by IBM Tivoli Workload Scheduler to:

- Test connectivity to the OnDemand Server.
- Pull IBM Tivoli Workload Scheduler logs from IBM Tivoli Workload Scheduler Domain Managers and FTAs.
- Push IBM Tivoli Workload Scheduler logs to the OnDemand Server.
- Update database with current job information.

The installation consists of the OnDemand Client Software on the designated Data Pulling system (this system must be an IBM Tivoli Workload Scheduler full-status FTA or Master). The scripts are executed by IBM Tivoli Workload Scheduler in the form of jobs in a job stream and executed daily.

The scripts may be modified to run in real-time mode or batch mode, depending on the customer's environment and needs. These variables will determine the processing time of when the IBM Tivoli Workload Scheduler logs are transported to the OnDemand Server. Think of how these choices can be integrated into your other document view solutions.
2.3.1 Data to be collected

We have supplied OnDemand application groups, application names, folders, and scripts to support the capture of the following outputs:

- The TWSmerge log
- The Netman log
- The Plan auditlog (or Plan log)
- The Database auditlog (or Database log)

2.3.2 Data collection process

In the distributed environment, we are using two methods to either push or pull the data for the distributed related output in either real-time or batch modes.

**Real-time mode**

In the real-time mode, all IBM Tivoli Workload Scheduler logs from the Master Domain Manager, Domain Managers, FTAs, Extended Agents, and Standard Agents are pulled as they are created, and immediately pushed to the OnDemand server.

**Batch mode**

In batch mode, the IBM Tivoli Workload Scheduler logs are queued for transport at a designated time in intervals. This mode is configurable by scheduling the pull and push of the IBM Tivoli Workload Scheduler logs at a specified time whenever convenient for specific environments.

**Note:** In real-time mode, the following logs are pulled only once a day regardless of any customization:

- TWSmerge logs
- Netman logs
- Plan logs
- Database logs

These logs are updated by IBM Tivoli Workload Scheduler throughout the day and cannot be pulled in real time; therefore, these logs only will be pulled at the beginning of the new IBM Tivoli Workload Scheduler day, at which time new logs are created.
**Pull method**
Here we discuss the pulling of IBM Tivoli Workload Scheduler logs from all IBM Tivoli Workload Scheduler servers. This operation is accomplished by a series of scripts executed by IBM Tivoli Workload Scheduler as jobs in a job stream. For this reason, the Pull Server must also be an IBM Tivoli Workload Scheduler FTA configured with Full Status and Resolve Dependencies, as well as the OnDemand Client. The Pull Server is also pushing the data to the OnDemand Server.

The Pull method works in real-time mode only; therefore, IBM Tivoli Workload Scheduler logs being pulled into this server are immediately pushed to the OnDemand Server, as long as the link to the OnDemand Server is established. For further details of the Pull method, see Chapter 4, “IBM Tivoli Workload Scheduler Distributed implementation” on page 77.

**Push method**
Here we discuss pushing the IBM Tivoli Workload Scheduler logs from every FTA in the IBM Tivoli Workload Scheduler network. With this method, the OnDemand Client software must be installed on every IBM Tivoli Workload Scheduler FTA. A job stream containing jobs that execute the scripts locally on each IBM Tivoli Workload Scheduler FTA must be scheduled to launch daily. This method can be used in either real-time mode or batch mode. For further details of the Push method, see Chapter 4, “IBM Tivoli Workload Scheduler Distributed implementation” on page 77.

**2.4 z/OS and end-to-end environment**
In both the z/OS and the end-to-end environment, we recommend that the OnDemand database (DB2) resides on the z/OS machine where the primary IBM Tivoli Workload Scheduler controller is situated due to its unprecedented reliability, scalability, and performance.

**2.4.1 Data to be collected**
We have supplied OnDemand application groups, application names, and folders to support the capture of the following outputs:

- The SYSLOG produced by any batch job that specifies a MSGCLASS reserved for use by OnDemand.
- Certain reports produced by IBM Tivoli Workload Scheduler itself, such as Daily and Long Term Planning jobs that specify a SYSOUT CLASS and FORM reserved for use by OnDemand.
Message logs produced by IBM Tivoli Workload Scheduler started tasks, such as the Controller and Tracker that specify a SYSOUT CLASS and FORM, are reserved for use by OnDemand.

The output from the above is written to the JES Spool and awaits collection by OnDemand processing.

### 2.4.2 Data collection process

In the z/OS and end-to-end environment, we are using two methods to either push or pull the data for z/OS related output. The method of collecting distributed data in an end-to-end environment will be the same as described earlier in the distributed section.

**Push method: real-time mode**

The MSGCLASS and SYSOUT classes/forms defined for use by OnDemand cause the output from these jobs to be pushed out to the JES Spool, and started tasks continuously running on the z/OS machine where the JES Spool is located search for these classes and push the data directly into OnDemand database. The started tasks used for this purpose are known as ARSYSPIN and ARSLOAD.

**Pull method: batch mode**

In certain situations, it may be advantageous not to write a user report directly to sysout but to write the report to a z/OS data set instead for collection into OnDemand at a later time. In this case, we use the Pull method to locate this report and store it in the OnDemand database. You will see later in 3.4.8, “Install the ARSBATCH procedure” on page 66 that this is achieved by way of the ARSBATCH procedure. This procedure can be invoked as a batch job in its own right or can be added as a step to an existing job.

Once you have decided on your architecture, the scripts will simply integrate this solution, which strategically can be added to facilitate other company services.
IBM Tivoli Workload Scheduler for z/OS implementation

This chapter discusses in detail the steps required to implement the integration between IBM Tivoli Workload Scheduler for z/OS and OnDemand for z/OS.

If you are using IBM Tivoli Workload Scheduler to schedule an end-to-end (E2E) environment, z/OS and distributed, then you should also read Chapter 5, “IBM Tivoli Workload Scheduler end-to-end environment” on page 113 and Chapter 4, “IBM Tivoli Workload Scheduler Distributed implementation” on page 77, which deal with implementing IBM Tivoli Workload Scheduler for Distributed and E2E.

In this chapter, the following topics are covered:

- Section 3.1, “Overview” on page 32
- Section 3.2, “Architecture” on page 34
- Section 3.3, “Prerequisites” on page 36
- Section 3.4, “Implementation” on page 38
3.1 Overview

In the z/OS environment, IBM Tivoli Workload Scheduler can schedule operations that consist of batch jobs that run on the JES processors throughout the enterprise. These batch jobs produce output via the JES MSGCLASS parameter and are written to the JES Spool. Keeping this log in an archive database for later retrieval or audit purposes can be very advantageous and this is where the OnDemand process can help.

In addition to the MSGCLASS referred to above, IBM Tivoli Workload Scheduler also produces its own message logs (EQQMLOG) and Batch Reports that can also aid in problem determination and satisfy audit requirements.

Section 3.4, “Implementation” on page 38 describes how we capture and store the SYSOUT (MSGCLASS) from the z/OS jobs that are scheduled by IBM Tivoli Workload Scheduler, the message logs (EQQMLOG) produced by IBM Tivoli Workload Scheduler tasks, such as Controller and Trackers, and the output from planning and database reports from the IBM Tivoli Workload Scheduler system itself.

Once captured and archived in the OnDemand database, users can easily retrieve the data by use of the OnDemand client or Web based dialogues. The Web based dialogue is an optional feature that can be purchased and implemented as part of the OnDemand solution.

Finding the required data from within the archive database is a simple process. By selecting filtering criteria such as Date and/or Time as well as other useful keys, which can be defined within the OnDemand system, such as Jobname, Return Code, DDNAME, Machine ID, and so on, speedy recovery of the required data can occur. Once a log (or logs) has been retrieved using the selection criteria defined via the user dialogue, additional processing, such as sorting the results or searching for a specific string, are all possible.

Some useful examples could be:

- Search for all IBM Tivoli Workload Scheduler controller logs produced over the last seven days and look for a specific error message.
- Scan the IBM Tivoli Workload Scheduler audit reports for the last month to check for the activity of a certain user.
- Scan the Daily Planning Reports for the last 12 months to check to see if a particular job or application was ever scheduled.
- Search for archived Application Descriptions for audit purposes.
- Search for archive Operator Instructions for audit purposes.
In our IBM Tivoli Workload Scheduler and OnDemand integrated environment, OnDemand can capture data in one of three ways:

- The ARSYSPIN started task, which monitors the JES SPOOL for a specific SYSOUT class. Typically, this is the output specified by the MSGCLASS parameter on the jobcard of a z/OS scheduled batch job.
- The ARSLOAD started task, which monitors the JES SPOOL for a specific SYSOUT class and FORMS. Typically, this would be used for user defined reports or message logs.
- The ARSBATCH procedure for reading z/OS datasets contains user reports instead of reading the reports directly from the JES Spool.

Sample procedures for ARSYSPIN, ARSLOAD, ARSBATCH, and the ARSSOCKD started task, which handles the communication between the z/OS system and the OnDemand Server running in UNIX System Services (USS), can be found within the implementation section along with a sample job that invokes the ARSBATCH procedure.

Figure 3-1 shows a general overview of this integration.

![Figure 3-1 OnDemand for z/OS architecture](image-url)
3.2 Architecture

In this section, we will describe the detailed architecture of the OnDemand and IBM Tivoli Workload Scheduler for z/OS integration, starting with the single z/OS domain implementation.

3.2.1 Single z/OS domain

Figure 3-2 indicates the architecture used for an OnDemand and IBM Tivoli Workload Scheduler for z/OS installation within one JES domain.

You should note the following in Figure 3-2:

- Controller, Tracker, and DB2 are all in the same domain.
- A z/OS job being scheduled by the controller can produce MSGCLASS and SYSOUT(FORM EQQR) written to the JES Spool and, optionally, user reports written to z/OS datasets.
- The controller and tracker tasks write their EQQMLOG to JES Spool via the form name EQQM.
The JES Spool output is processed by two separate started tasks known as ARSYSPIN and ARSLOAD.

The ARSYSPIN started task collects MSGCLASS=W output and loads it into the OnDemand DB2 database.

The ARSLOAD started task collects SYSOUT=Y, FORMS EQQR, and EQQM and load it into the OnDemand DB2 database.

The ARSBATCH procedure, which can be run as a batch job in its own right or can be added as a step to an existing job, reads user reports that have been previously written to a z/OS dataset instead of directly to the JES Spool.

### 3.2.2 Multiple z/OS domains

Figure 3-3 indicates the architecture used for an OnDemand and IBM Tivoli Workload Scheduler for z/OS installation within two or more z/OS domains.
You should note the following in Figure 3-3 on page 35:

- The z/OS Domain1 is explained in Figure 3-2 on page 34.
- The z/OS Domain 2 is identical to the z/OS Domain 1 (some symbols such reports for EQQM and EQQR, have been left off for ease of reading), but you should note that we have another z/OS tracker running work in Domain 2 and that this tracker is connected to the controller in Domain 1.
- Each Domain, with its own JES Spool, would also have its own ARSYSPIN, ARSLOAD and ARSBATCH processes, and these processes would each write data back to the OnDemand DB2 database, in this case, residing in Domain 1.

### 3.3 Prerequisites

The integration referred to by this redbook assumes you already have IBM Tivoli Workload Scheduler, OnDemand, and DB2 installed on the relevant platforms (in this case, z/OS) and that your are at the latest maintenance level for all of these products.

#### 3.3.1 Software versions and releases tested

In our testing environment, we used the following versions of these programs with all the maintenance levels applied:

- IBM Tivoli Workload Scheduler for z/OS Version 8.2 (will also work with IBM Tivoli Workload Scheduler Version 8.1 for z/OS)
- OnDemand for OS/390 and z/OS Version 7.1
- Database Manager 2 (DB2) for z/OS Version 7 at recommended service upgrade RSU0303
- z/OS Operating System Version 1.4 at recommended service upgrade RSU0305
- COBOL Enterprise Edition Version 3.2

#### 3.3.2 JES2 sysout classes

For our integration solution, we require two JES sysout classes to be reserved for OnDemand processing: One to enable z/OS MSGCLASS output collection and the other for user reports that can be specified along with a special JES FORM name. For example: MSGCLASS=W and SYSOUT=(Y,,EQQR), SYSOUT=(Y,,EQQM).
You do not have to use the output classes of W or Y, but if you do substitute either of these, then remember to take this fact into consideration when reading the examples.

**Important:** The forms names of EQQR and EQQM have been specifically created for use within our integration scenario and cannot be changed unless you change the application group and application name definitions within the OnDemand administration dialogue.

### 3.3.3 Additional documentation

Table 3-1 lists useful Web site addresses that can be used to obtain additional information and documentation if required.

<table>
<thead>
<tr>
<th>Description</th>
<th>Web address</th>
</tr>
</thead>
</table>

One document that we have used in implementing this solution is the *OnDemand for OS/390 and z/OS - JES Spool Data Capture - ARSYSPIN Reference*. Some information from this document has already been included within this chapter, but you might still want to obtain the original document. For example, if you want to know how to provide user written exits for use with the ARSYSPIN procedure, which is referred to later, you need to consult that document.
The following steps describe what you need to do to obtain this document:

1. Go to the following Web site:
   http://www-1.ibm.com/support/docview.wss?uid=swg27002351
2. Locate the ARSYSPIN.ZIP file on this site.
3. Double-click the ARSYSPIN.ZIP file and open it from its current location.
4. Open the ARSYSPIN.pdf file and then save it to your own location for future reference. You can ignore all the other files contained within this ZIP directory, since they are not used as part of this solution. However, we will be downloading and unzipping another file later as part of the implementation process to import the application groups and folders specifically created for the ITWS/OnDemand integration.

### 3.4 Implementation

This topic describes the steps necessary to implement the IBM Tivoli Workload Scheduler OnDemand integration. It does not, however, address the installation and implementation of the DB2, OnDemand, and IBM Tivoli Workload Scheduler for z/OS products themselves.

To obtain information related to the installation and customization of these products, you should refer to the relevant documentation.

#### 3.4.1 Overview of the implementation

In order to integrate the IBM Tivoli Workload Scheduler for z/OS and OnDemand products, the following steps are necessary:

1. Ensure that you have a fully working DB2 system running on z/OS that can be used to archive the z/OS output using the OnDemand Server.
2. Ensure that you have a fully working OnDemand system that can communicate with the DB2 system used to archive OnDemand reports.
3. Ensure that you have a fully working IBM Tivoli Workload Scheduler for the z/OS scheduling environment.
4. Install the ARSSOCKD started task, if it is not already completed, as part of the original OnDemand installation.
5. Install the supplied application groups and folders:
   a. Download the zip file containing the OnDemand application group and folder definitions used for the ITWS/OnDemand integration.
   b. Unzip the downloaded export file referred to above.
c. Import the definitions previously unzipped.

6. Ensure you have valid user IDs and passwords allocated with enough authority to be able to add data to the OnDemand database.

7. Ensure you have valid user IDs and passwords allocated with enough authority to be able to view data in the OnDemand database.

8. Implement ARSYSPIN:
   a. Compile and assemble the ARSYSPIN exit.
   b. Create the ARSYSPIN parameter file.
   c. Modify as required and install the ARSYSPIN started task procedure.

9. Modify as required and install the ARSLOAD started task procedure.

10. Modify as required and install the ARSBATCH procedure.

11. Optionally, modify the IBM Tivoli Workload Scheduler started task parameter members for Controller, Tracker, E2E Server, Datastore, JSC, and so on, if you wish to store the EQQMLOGS from these tasks.

12. Optionally, modify the IBM Tivoli Workload Scheduler batch job skeletons and/or any scheduled batch jobs used for running IBM Tivoli Workload Scheduler reports, such as Current and Long Term Planning, to ensure the report can be processed by OnDemand. This can take the form of:
   a. Ensuring the report is written to a specified data set for later processing.
   b. Ensuring the report is written to a specified sysout class and form name.

13. Optionally, modify the MSGCLASS parameter of any batch jobs running under the control of IBM Tivoli Workload Scheduler that you wish to archive within the OnDemand database.

Each of the above steps is outlined in detail in the following sections.

### 3.4.2 Install the ARSSOCKD procedure

This is the primary task enabling OnDemand in the z/OS address space to communicate with the OnDemand Server residing in UNIX Systems Services (USS). The OnDemand Server handles the transactions it receives and processes these transactions by sending the data to DB2. This started task may have already been installed within your installation, but if not, Example 3-1 on page 40 shows a sample ARSSOCKD that you must modify to suit your own installation requirements.
Example 3-1  ARSSOCKD procedure

```plaintext
//ARSSOCKD PROC
//ARSSOCKD EXEC PGM=ARSSOCKD,REGION=0M,TIME=NOLIMIT
//STEPLIB DD DISP=SHR,DSN=ARS.SARSLOAD
// DD DISP=SHR,DSN=DB7G7.SDSNEXIT
// DD DISP=SHR,DSN=DB7G7.SDSNLOAD
// DD DISP=SHR,DSN=APK.ACIF.SAPKMOD1
//DSNAOINI DD PATH='/usr/lpp/ars/config/cli.ini'
//SYSPRINT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=* 
```

**Important:** In order for the ARSSOCKD task to successfully initialize a valid DB2 system, as pointed to by the DSNAOINI path name and file, it must have already been started on the same machine as ARSSOCKD. Details on how to configure the ARSSOCKD task can be found within the *IBM Content Manager OnDemand for z/OS and OS/390 Configuration Guide Version 7.1*, GC27-1373.

### 3.4.3 Download, unzip, and import the group and folder definitions

As part of the configuration of ARSYSPIN, at least one application, application group, and folder must be created so that the spool file data can be indexed and loaded into OnDemand.

The application defines the characteristics of the data, such as data type, record length, record delimiter, code page, and so on. It is also where the indexer parameters are defined for indexing.

The application group defines the database fields, including the field name, type of field, length, and so on. It also contains storage management information, such as data retention periods and archive media information.

The folder defines how the search and display fields will be presented in the Windows client so that individual spool files can be retrieved and viewed.

In this redbook, we provide sample applications, application groups, and folders that can be used when setting up ARSYSPIN. The samples can be used with minimal changes or they can be used as a template for creating a customized set of application, application group, and folder definitions. The only required change is in the application group. The storage management information of the application group needs to be updated to reflect the storage set and data retention values required for the user's environment. Another change may be required if the default code page used in the application definition is not correct for the user's environment. The sample application definitions use a code page.
value of 1047. If the data to be captured is in a code page other than 1047, the code page must be changed in the application definition.

Application, application group, and folder definitions samples are provided as a zip file (TWS4ZOS.zip) with this redbook. To download the zip file, use the instructions in Appendix C, “Additional material” on page 243.

Figure 3-4 shows the structure of the TWS4ZOS.zip file.

The following information describes how to define an application, application group, and folder using the samples:

1. Using a binary file transfer mechanism (such as FTP), transfer the TWS4ZOS.ZIP file to a temporary or working directory on your PC.

2. Execute PKZIP (or its equivalent) to extract the contents of the TWS4ZOS.ZIP file and recreate the proper directory structure. For example, the following command will create a directory named TWS4ZOSSAMP anchored within the root directory of the C: drive:

   \texttt{pkzipc -ext -dir c:\TWS4ZOS.ZIP c:}\]

   The -ext parameter indicates that PKZIPC is to extract the contents of the TWS4ZOS.ZIP file.

   The -dir parameter indicates that the directory structure contained in the TWS4ZOS.ZIP file is to be recreated.
The TWS4ZOSSAMP directory will, in turn, contain three subdirectories named LOAD, TABLE, and VIEW. The files contained in this set of directories comprise the system tables that define a local server.

The local server contains the definitions for the sample application, application group, and folder.

The following steps describe how to create a local server, export the definition files to the *OD390 OnDemand Server*, and update those definition files to comply with your server environment.

**Important:** The following tasks must be performed by a user with administration authority.

1. Start the Administrative Client.
   a. From the Windows task bar button, select **Start → Programs → IBM OnDemand32 → OnDemand administrator.**

2. Add the local server to the server list.
   a. Highlight **OnDemand Servers**.
   b. From the File menu, select **New Server**.
   c. Type TWS4ZOSSAMP in the Server field.
   d. Change the Protocol to Local by selecting **Local** from the Protocol combo box.
   e. Enter the full path name of the TWS4ZOSSAMP directory (or click on the **Browse** button and expand the directory list for the drive where the TWS4ZOSSAMP directory was added by clicking on the + sign next to the drive name; select the TWS4ZOSSAMP directory and click on the **OK** button.
   f. Click on the **OK** button to add the local server.

3. Log on to the TWS4ZOSSAMP server.
   a. Double click the left mouse button on the TWS4ZOSSAMP server.
   b. Enter admin as the user ID and leave the password blank (no password is required), then click **OK** to logon to the server.

4. Export the sample application group to the OD/390 server.
   a. Select **Application Groups**, which is located below the TWS4ZOSSAMP server name in the left pane of the window.
b. Select the application group you wish to load in the right pane of the window and press the right mouse button. Select Export from the pop-up menu.

**Note:** The supplied application groups are MSGCLASS, EQQR and EQQM, and you will need to repeat this section (4) down to section(7) for each application group in turn.

c. Select the name of your OD/390 server from the Server combo box.

d. Click on the No Storage Set check box.

e. Click on the Export button to display the Logon dialog box for the OD/390 server.

f. If you are currently logged on the OD/390 server, verify that the user ID has authority to create application groups and folders. Otherwise, complete the logon in the following step.

g. Enter a user ID and password for the OD/390 server and click on the OK button to log on to the server.

h. Click on the OK button to complete the export. (The sample application is automatically exported with the application group.)

5. Export the sample folder to the OD/390 OnDemand Server.

   a. Repeat all the steps in Item 4 substituting folder for application group. (Logging on again is not necessary, unless you logged off after the last export operation.)

6. Update the selected application group on the OD/390 server to specify a Storage set.

   a. Point to the OD/390 server in the left pane of the window and expand the server by clicking on the + sign.

   b. Select the Application Groups category located below the OD/390 server name in the left pane of the window.

   c. Select Update from the pop-up menu for the application group you currently installing.

   d. Click on the Storage Management tab to display Storage Management information.

   e. Select the name of an appropriate storage set from the Storage Set Name combo box. OnDemand uses the storage set to maintain the spool file data in cache storage and in archive storage.
f. (Optional) If necessary, change the parameters that identify how long data is maintained in cache and how long the data and indexes will be retained in OnDemand.

g. Click on the OK button to update the application group with the changes.

7. (Optional) If necessary, update the application you are currently installing on the OD/390 server to change the code page.

   a. Point to the OD/390 server in the left pane of the window and expand the server by clicking on the + sign.

   b. Select the Applications category located below the OD/390 server name in the left pane of the window.

   c. Select Update from the pop-up menu for the application you are currently installing.

   d. Click on the View Information tab to display information used to display the data.

   e. Change the code page field value from 1047 to the code page of the data (for example 037, 500).

   f. Click on the OK button to update the application with the change.

After the application, application group, and folder have been defined, the OnDemand Server is ready to begin receiving indexing and loading jobs from ARSYSPIN. Figure 3-5 on page 45 indicates the application groups, applications, and folders that have been created for this redbook once your import has been successfully completed. If you are already an OnDemand user, then you will probably have additional entries be displayed. This is, of course, as expected.
3.4.4 Create user IDs and passwords for the OnDemand database

In the following sample procedures for ARSYSPIN, ARSLOAD, and ARSBATCH, the user ID and password have been hardcoded within the procedure itself or, as in the case of the ARSYSPIN procedure, the parameter file the procedure uses. This can be changed, if required, to pick up the user ID and password from a file stored within the USS portion of the system. In any case, it is recommended that the user ID you specify within these tasks only has enough authority to be able to ‘ADD’ data to the OnDemand database, thus reducing the chance of any misuse in the case of non-authorized users obtaining the password. User IDs and passwords are administered through the OnDemand Administration dialogue.
3.4.5 Overview of the JES Spool data capture facility (ARSYSPI

The JES Spool Data Capture facility provides a means to collect and consolidate JES Spool (SYSOUT) data sets into one (or more) files so they can be archived by OnDemand.

The facility executes as a started task in its own address space. Throughout the remainder of this document, the facility is referred to by its program name, ARSYSPI.

A control statement file is used to provide ARSYSPI parameters. These parameters specify JES Spool file selection criteria (for example, from which SYSOUT classes output is to be selected) as well as other operational characteristics.

ARSYSPI creates an intermediate output file that contains one or more spool file(s) from one or more job(s). Each captured spool file is bracketed by a pair of separator records. A Begin separator record is written before the first record of the spool file and an End separator record is written after the last record of the spool file data. The separator records contain information that can be used to construct index values to facilitate the retrieval and viewing of the captured data sets after they have been stored by OnDemand.

The intermediate output file is indexed and stored in OnDemand using the ARSLOAD program. ARSYSPI invokes ARSLOAD when sufficient data has been captured in the intermediate output file. ARSLOAD indexes the data by calling an indexer program (usually APKACIF) to extract the index values from the data and store them in an index file. ARSLOAD then adds the indexes to the database and loads the data onto archive media.

A sample APKACIF input exit is provided with ARSYSPI, which illustrates a technique for inserting a data record into the APKACIF input stream for the purpose of providing additional indexing information. This sample exit scans the JES job log and system messages spool files (if they are present) to locate critical system and job processing messages (for example, messages which

Note: In our samples, we are using a user ID of IBM Tivoli Workload Scheduler and a password of ondemand. Remember that if you want to create a new user ID such as TWS that limits the authority to be able to just add data to the database, then once you have created the user with a password name of your choice, in our case, ondemand, then you should log on to this user initially through the server dialogue to reset the password. Again you can use the same name (ondemand). You will now be ready to use it in the ARSLOAD, ARSBATCH, and ARSYSPI tasks.
indicate the execution of job steps that terminated via ABEND or with non-zero completion codes), extract related values, and place this information in the inserted record for subsequent indexing.

After the captured spool data has been stored into OnDemand, ARSYSPIN repeats the process of collecting and consolidating other spool files so they too can be archived in OnDemand. The cycle of collecting, consolidating, and loading data continues until the program is terminated via the MVS™ STOP command.

**Note:** For more information on how to obtain additional details about the ARSYSPIN task, please refer to 3.3, “Prerequisites” on page 36.

### 3.4.6 Install the ARSYSPIN started task

The ARSYSPIN started task is used to continually check for a specified sysout class and forms number that has been defined as the MSGCLASS on the JOBCARD of the selected production jobs that we wish to archive. Throughout the examples in this document, we are looking for MSGCLASS=W with a FORM name of STD. Any MSGCLASS sysout meeting this criteria will be selected for processing and loaded into the OnDemand database using the application group MSGCLASS and application name of TWSJOBS.

Figure 3-6 on page 48 shows a schematic of the following implementation tasks that are required to fully implement the ARSYSPIN started task.
In order to install the ARSYSPIN started task we have to:

- Compile and assemble the CEEUOPT csect, as shown in Example 3-2.

**Example 3-2  CEEUOPT csect assembly job**

```plaintext
//UOPTJCL JOB (999,POK),'CM ONDEMAND',CLASS=A,MSGCLASS=T,
// NOTIFY=&SYSUID,TIME=1440,REGION=OM
/*JOBPARM L=999,SYSAFF=SC64
*/
/* TAKEN FROM CEE.SCEESA(CEEUOPT)
*/

//STEP1 EXEC PGM=ASMA90,PARM='DECK,NOBJECT'
//SYSPRINT DD SYSOUT=*  
//SYSUT1 DD UNIT=VIO,SPACE=(CYL,(1,1))
//SYSUT2 DD UNIT=VIO,SPACE=(CYL,(1,1))
//SYSUT3 DD UNIT=VIO,SPACE=(CYL,(1,1))
//SYSLIB DD DSN=CEE.SCEEMAC,DISP=SHR
// DD DSN=SYS1.MACLIB,DISP=SHR
//SYSIN DD DISP=SHR,DSN=ARSUSER.JCL.CNTL(CEEUOPT)
```
The ARSSPVIN sample APKACIF input exit is written as a COBOL main program. In order to prevent the Language Environment® from creating and destroying the COBOL run-time environment each time ARSSPVIN is called, a CEEUOPT CSECT must be assembled and link-edited with the COBOL object code.

Constructing a CEEUOPT CSECT is documented in *OS/390 Language Environment for OS/390 Customization Guide*, SC28-1941. A sample CEEUOPT CSECT can be found in data set CEE.SCEESAMP(CEEUOPT) and is documented in the JES Spool Data Capture document referred to in 3.4, “Implementation” on page 38, should you wish to see it. We recommend you take a copy of this source file and store it in your own user dataset prior to assembly. In our example we used DSN=ARSUSER.JCL.CNTL(CEEUOPT).

**Important:** You can use a copy of the CEE.SCEESAMP(CEEUOPT) as a model, but you *must* be sure that the RTEREUS=(ON) option is specified.

IBM also recommends that the ALL31(ON) option be specified, but this is not required.

In addition, you *must* be sure that the resulting module is link-edited as NOT RE-ENTRANT and NOT REUSABLE. This is required to allow the local variables within the COBOL exit code to retain their values across multiple invocations.

**Important:** Please ensure you read and understand the section entitled “Special Considerations for APKACIF Exits Written in Cobol”, which can be found in *OnDemand for OS/390 and z/OS - JES Spool Data Capture - ARSYSPIN Reference*, found at:


Compile and assemble the APKACIF indexing exit known as ARSYSPIN, as shown in Example 3-3.

*Example 3-3  ARSYSPIN (APKACIF exit) compile and assembly job*

```
//SPVINJCL JOB (999,POK), 'CM ONDEMAND', CLASS=A, MSGCLASS=T,
// NOTIFY=&SYSUID,TIME=1440, REGION=OM
/*JOBPARM L=999, SYSAFF=SC64
//*
// JCLLIB ORDER=(ARSUSER.JCL.CNTL)
//STEPI EXEC IGyWCL,
```
PARM.COBOL='NODYNAM,LIB,LIST,MAP,OBJECT,APOST,TRUNC(BIN)',
PARM.LKED='XREF,LET,LIST,MAP,rmode=24,amode=31,norent,noreus'
//COBOL.SYSLIB DD DISP=SHR,DSN=ARS.SARSINST
//             DD DISP=SHR,DSN=CEE.SCEESAMP
//COBOL.SYSIN  DD DISP=SHR,DSN=ARSUSER.JCL.CNTL(ARSSPVIN)
//LKED.SYSLIB  DD
//             DD DISP=SHR,DSN=ARS.SARSLMD5
//LKED.SYSLMOD DD DISP=SHR,DSN=ARSUSER.LOADLIB
//LKED.SYSIN   DD *
INCLUDE ARSOBJ(CEEUOPT)
ENTRY ARSSPVIN
NAME ARSSPVIN(R)
/*
//ARSOBJ    DD DISP=SHR,DSN=ARSUSER.OBJ

ARSSPVIN is a sample APKACIF input exit provided with ARSYSPIN to add additional index values into the data using a trailer record. Trailer records are inserted at the end of the JESMSGLG or JESYSMSG data and reflect the highest severity of the condition code that has been found in any of the message logs within the file being processed.

The input exit begins by examining each record in the data file and looking for a separator record that identifies an appropriate Spool file (JESMSGLG or JESYSMSG). When such a Spool file has been found, each subsequent record is examined to look for specific messages or a specific string in the record that contains a condition code. If the message contains a condition code and the condition code is non-zero, the condition code is saved and is written later as part of the trailer record information. If the message indicates a JCL error, a condition code is set to 'JCLE' and saved. When all of the records have been processed in the current JESYSMSG log, a trailer record is written. If no errors have occurred, the condition code will have a value of '0000' in the trailer record. Once a trailer record has been written, processing begins again, looking for a separator record that identifies the next JESMSGLG or JESYSMSG message log in the file.

There are several different types of condition codes that can occur. For each type, the highest value of the condition code is saved as the records are being processed. Later, when the trailer record is written, a determination is made as to which condition code should be used. The determination is based on the order of items shown in Table 3-2 on page 51.
Table 3-2  APKACIF exit condition code handling

<table>
<thead>
<tr>
<th>Component</th>
<th>Condition code</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>Non-zero condition code extracted from message IEF472I (for example, SYSTEM=0C3).</td>
</tr>
<tr>
<td>User</td>
<td>Non-zero condition code extracted from message IEF472I (for example, USER=0008). Non-zero condition code extracted from record containing &quot;USER COMPLETION CODE=&quot;.</td>
</tr>
<tr>
<td>Step</td>
<td>Non-zero condition code extracted from message IEF142I (for example, COND CODE 0012).</td>
</tr>
<tr>
<td>JCL</td>
<td>JCL error message found: IEE122I, IEE124I, IEF373I, IEF452I, IEFC452I, IEE453I.</td>
</tr>
</tbody>
</table>

The job start and stop times (extracted from messages IEF375I and IEF376I, respectively) are also included in the trailer record.

**Important**: Note the inclusion of the csect, CEEUOPY, which was previously compiled and assembled.

In order to run the ARSYSPIN procedure successfully, we must also define the parameter file to be used by the ARSYSPIN task. Example 3-4 is a sample ARSYSPIN parameter file referred to by the ARSYPARM dd statement.

**Example 3-4  ARSYSPIN parameters used for TWS for z/OS integration**

* ARSYSPIN Parameters
  ODUSER=TWS
  ODUSERPW=ONDEMAND
  ODHOST=9.12.6.31
  DSNPFX=ARSUSER
  LOADPGM=ARSLOAD
  JOBBREAK=YES
  TEMPPATH=/tws/arstmpwrk
  TEMPUNIT=SYSDA
  JESCLASS=W
  APPLGROUP="MSGCLASS"
  APPL="TWSJOBS"
  SELFORM=(STD)
ARSYSPIN parameter explanations

Table 3-3 explains all the possible parameters that can be supplied to the ARSYSPIN program.

Table 3-3   ARSYSPIN keywords and descriptions

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPL</td>
<td>Specifies the OnDemand application name. See the description of the ARSLOAD -a parameter in the <em>IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1</em>, SC27-1374. Must be supplied within quotes. Example: APPL=&quot;MSGCLASS&quot;.</td>
</tr>
<tr>
<td>APPLGROUP</td>
<td>Specifies the OnDemand application group name. See the description of the ARSLOAD -g parameter in the <em>IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1</em>, SC27-1374. Must be supplied within quotes. Example: APPLGROUP=&quot;TWSJOBS&quot;.</td>
</tr>
<tr>
<td>CAPDSKEEP</td>
<td>Specifies if the intermediate capture data sets are to be kept or deleted after being processed by ARSLOAD. The principal use of keeping an intermediate capture data set is to facilitate the construction of indexing parameters with the OnDemand Report Wizard. Example: CAPDSKEEP=YES (CAPDSKEEP is ignored if the DSNPFX parameter is not specified).</td>
</tr>
</tbody>
</table>
**DSNPFX**

Specifies the left-most 26 characters of the data set name string used when creating intermediate data sets. A given intermediate data set contains the captured spool file data that is used as input to the load program.

The right-most two qualifiers of the data set name are constructed to be of the form: Dyyyddd.xhmmssth.

The "yyyyddd" portion of the first of these qualifiers is the current system date. The second of these qualifiers is the current system time, where the "x" is one of the letters "A", "B", or "C", corresponding to the first digit of the hour being 0, 1, or 2, respectively.

When the DSNPFX parameter is specified, the effective data set disposition is a function of the CAPDSKEEP parameter. When CAPDSKEEP=YES is specified, the effective disposition is NEW,CATLG,CATLG. When CAPDSKEEP=NO is specified, the effective disposition is NEW,DELETE,CATLG.

If the DSNPFX parameter is not specified, the intermediate data sets are allocated as temporary data sets with an effective disposition of NEW,DELETE,DELETE.

Example: To specify the leftmost portion of the generated data set name string as OPS.TRACKING, you would enter DSNPFX=OPS.TRACKING. If the current system date is day 95 of year 2002 and the time is 11:30:15.65, the generated data set name is OPS.TRACKING.D2002095.B1301565.

---

**ERROPT**

Specifies the action to be taken by ARSYSPIN when spool file I/O errors are encountered. The operand must be one of the following:

- **ABEND**: ARSYSPIN terminates processing via ABEND. All of the spool files that had been captured in the current cycle remain available for selection.

- **ACCEPT**: The error is treated as though an End-of-File condition had been encountered. Processing continues normally.

- **HOLD**: The error is treated as though an End-of-File condition had been encountered. The spool file that encountered the error is placed in a System Hold (JES2 HOLDRC=6) state. The remainder of processing continues normally.

---

### Keyword Description

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNPFX</td>
<td>Specifies the left-most 26 characters of the data set name string used when creating intermediate data sets. A given intermediate data set contains the captured spool file data that is used as input to the load program. The right-most two qualifiers of the data set name are constructed to be of the form: Dyyyddd.xhmmssth. When the DSNPFX parameter is specified, the effective data set disposition is a function of the CAPDSKEEP parameter. When CAPDSKEEP=YES is specified, the effective disposition is NEW,CATLG,CATLG. When CAPDSKEEP=NO is specified, the effective disposition is NEW,DELETE,CATLG. If the DSNPFX parameter is not specified, the intermediate data sets are allocated as temporary data sets with an effective disposition of NEW,DELETE,DELETE. Example: To specify the leftmost portion of the generated data set name string as OPS.TRACKING, you would enter DSNPFX=OPS.TRACKING. If the current system date is day 95 of year 2002 and the time is 11:30:15.65, the generated data set name is OPS.TRACKING.D2002095.B1301565.</td>
</tr>
<tr>
<td>ERROPT</td>
<td>Specifies the action to be taken by ARSYSPIN when spool file I/O errors are encountered. The operand must be one of the following: ABEND: ARSYSPIN terminates processing via ABEND. All of the spool files that had been captured in the current cycle remain available for selection. ACCEPT: The error is treated as though an End-of-File condition had been encountered. Processing continues normally. HOLD: The error is treated as though an End-of-File condition had been encountered. The spool file that encountered the error is placed in a System Hold (JES2 HOLDRC=6) state. The remainder of processing continues normally.</td>
</tr>
<tr>
<td>Keyword</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>JESCLASS</td>
<td>Identifies the set of SYSOUT classes to be processed by ARSYSPIN. The operand should be specified as a string, which may consist of the characters A through Z and the digits 0 through 9. Example: To indicate that the SYSOUT classes H, K, and N are to be processed, specify JESCLASS=HKN.</td>
</tr>
<tr>
<td>JESNAME</td>
<td>Identifies the name of the Job Entry Subsystem with which ARSYSPIN is to interact. If JES3 is specified, then ARSYSPIN must itself be executing under the control of JES3. If this condition is not satisfied, ARSYSPIN will terminate by ABEND (U0039, Reason Code x’0490). Example: To indicate that ARSYSPIN should interact with a Job Entry Subsystem named JEST, specify JESNAME=JEST.</td>
</tr>
</tbody>
</table>
| JOBBREAK     | JOBBREAK indicates if a given collection of captured spool files contains the data for one job (JOBBREAK=YES) or for multiple jobs (JOBBREAK=NO). When JOBBREAK=YES is specified, the MAXDORM and MAXSFC parameters are ignored. When JOBBREAK=NO is specified, the MAXDORM and MAXSFC parameters work in concert as follows:  

Spool files for a given job that satisfy the specified selection criteria are gathered into the current collection. Once JES indicates there are no additional Spool files for the job currently being processed, the MAXSFC limit value is checked. If that limit has now been reached or exceeded, the current collection is regarded as complete and the load program (usually ARSLOAD) is invoked to process (that is, store) the collection.  

If the MAXSFC limit has not been reached, a timer is set whose value (in seconds) was specified via the MAXDORM parameter. If additional Spool files (for any job which satisfies the selection criteria) become available before this timer expires, those data sets are added to the current collection. Once the timer expires, the current collection is regarded as complete.  

Regardless of whether JOBBREAK=YES or JOBBREAK=NO is specified, the Spool files for a given job are always contained in the same collection.  

Example: To indicate that the Spool files for multiple jobs are to be gathered into a single collection, the approximate maximum number of Spool files is 100, and the maximum amount of time to wait for the arrival of additional Spool files is two minutes. We would specify JobBreak=No, MAXSFC=100, and MaxDorm=120. |
**LOADPGM**

Provides the name of the program to be used to process the captured Spool file collections. The principal use of this parameter (combined with specifying CAPDSEKEEP=YES) is to facilitate the construction of indexing parameters with the OnDemand Report Wizard.

The specified program is invoked as a subtask (via ATTACH) within the ARSYSPIN address space and must reside within an APF Authorized library. The execution environment at entry is: PSW: Problem State, Primary ASC Mode, same PSW Key as ARSYSPIN. APF Authorized and DU-AL are not shared with ARSYSPIN.

The PARM text presented to the specified program is what would normally be presented to ARSLOAD. See the *IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1*, SC27-1374 for further details regarding ARSLOAD command line parameters. The default is ARSLOAD.

Example: Retain the intermediate capture data set and specify IEFBR14 as the load program. The capture data set can then be used as input to the OnDemand Report Wizard to more easily create indexing specifications. For example, CAPDSEKEEP=YES and LOADPGM=IEFBR14.

**MAXDORM**

Specifies the maximum amount of time (in seconds) that ARSYSPIN is to wait for the arrival of additional Spool files that satisfy the current selection criteria. Once this period elapses, the current collection is regarded as complete.

This parameter is honored only when JOBBREAK=NO is specified; otherwise, if specified, it is ignored. See the description of the JOBBREAK and MAXSFC parameters in this table for further details regarding how these parameters interact with one another. The default is 120.

Example: To indicate that Spool files for multiple jobs are to be gathered into a single collection, the approximate maximum number of Spool files is 150, and the maximum amount of time to wait for the arrival of additional Spool files is five minutes. For example, Specific JobBreak=No, MAXSFC=150 and MaxDorm=300.
### Keyword Description

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXLPP</td>
<td>Specifies the size of a logical page in terms of lines. When this parameter is specified, ARSYSPIN maintains the notion of a logical page by &quot;counting lines&quot;. If a page break does not occur within the data stream before the MAXLPP limit is reached, ARSYSPIN forces a page break. A page break within the data stream is indicated when a given Spool file possesses either ASA or machine control characters and one of the following control characters is encountered:</td>
</tr>
<tr>
<td></td>
<td>- x'F1': ASA: Skip to Channel 1 and print</td>
</tr>
<tr>
<td></td>
<td>- x'8B': Machine: Skip to Channel 1 immediately</td>
</tr>
<tr>
<td></td>
<td>- x'89': Machine: Print and skip to Channel 1</td>
</tr>
<tr>
<td></td>
<td>If this parameter is not specified or if an operand value of zero is specified, line counting is not performed.</td>
</tr>
<tr>
<td></td>
<td>Maximum Length: 15 digits.</td>
</tr>
<tr>
<td></td>
<td>Default: 0.</td>
</tr>
<tr>
<td></td>
<td>Example: To indicate that line counting is to be performed and the logical page size is 70 lines, specify MAXLPP=70.</td>
</tr>
</tbody>
</table>
### MAXSFC

Specifies the maximum number of Spool files to be gathered into a single collection. Once this limit is reached or exceeded, the current collection is regarded as complete.

This parameter is honored only when JOBBREAK=NO is specified; otherwise, if specified, it is ignored. See the description of the JOBBREAK and MAXDORM parameters in this table for further details regarding how these parameters interact with one another.

Note that the Spool files for a given job are always contained in the same collection. Therefore, the specified limit should be regarded as an approximate upper bound and it is possible that the MAXSFC value may be exceeded.

**Maximum Length:** 15 digits  
**Maximum Value:** 768  
**Default:** 100

**Example:** To indicate that Spool files for multiple jobs are to be gathered into a single collection, the approximate maximum number of Spool files is 150, and the maximum amount of time to wait for the arrival of additional Spool files is five minutes, specify JobBreak=No, MAXSFC=150 and MaxDorm=300.

### ODHOST

Specifies the name of the OnDemand Library Server. See the *IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1, SC27-1374* for a description of the ARSLOAD -h parameter. The Maximum Length is 128 characters. Example: To specify the name of the OnDemand Library Server as MyLibServer, enter ODHOST=MyLibServer.

### ODINSTANCE

Specifies the OnDemand Instance name. See the *IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1, SC27-1374* for a description of the ARSLOAD -l parameter. The Maximum Length is 20 characters. Example: To specify the OnDemand Instance name as Production, enter ODINSTANCE=Production.

### ODUSER

Specifies an OnDemand User name (with LOAD authority only) on whose behalf ARSLOAD is to execute. See the *IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1, SC27-1374* for a description of the ARSLOAD -u parameter. The Maximum Length is 20 characters. Example: To specify the OnDemand User Name as Admin, enter ODUSER=Admin.
<table>
<thead>
<tr>
<th><strong>Keyword</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>ODUSERPW</td>
<td>Specifies the logon password for the OnDemand User name specified by the ODUSER parameter. See the <em>IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1</em>, SC27-1374 for a description of the ARSLOAD -p parameter. The Maximum Length is 20 characters. By default, the minimum length is 8. Example: To specify the OnDemand User Name as Admin with a logon password of REDBOOK, enter ODUSER=Admin and ODUSERPW=REDBOOK.</td>
</tr>
<tr>
<td>OUTSEP</td>
<td>Indicates if the ARSYSPIN generated separator records that surround captured Spool files are to be generated. Example: To indicate that separator records are not to be produced, enter OUTSEP=No.</td>
</tr>
<tr>
<td>REPORTID</td>
<td>Provides a string that is placed in the separator records that surround each captured Spool file. This parameter is provided solely for compatibility with a similar function offered in OnDemand for OS/390 Version 2. The Maximum Length is eight characters. Example: To specify an ID string of Sample, enter REPORTID=Sample.</td>
</tr>
<tr>
<td>RSADEXIT</td>
<td>Indicates if the Report Specifications Archive Definition exit is to be enabled. See the OnDemand Configuration Guide for details regarding the operation and other characteristics of this logical exit point.</td>
</tr>
<tr>
<td></td>
<td>Note: Support for the Report Specifications Archive Definition exit was introduced by APAR PQ70058. Do not request the enablement of this logical exit point if support for this APAR is not installed on your system.</td>
</tr>
<tr>
<td></td>
<td>Example: To indicate that Report Specifications Archive Definition exit is to be enabled, enter RSADEXIT=Yes.</td>
</tr>
<tr>
<td>SELDEST</td>
<td>Specifies a JES destination ID to be used as a Spool file selection filtering criteria. See the description of the OUTPUT JCL statement DEST parameter in <em>z/OS MVS JCL Reference</em>, SA22-7597 for details regarding JES destination IDs. Wild card characters may be specified. If the SELDEST parameter is not specified, then selection filtering by destination ID is not performed. The Maximum Length is 18 characters. Example: To indicate that Spool files are to be selected, which were to be routed to user SIMPSON on system POKVMB, enter SelDest=POKVMB.SIMPSON.</td>
</tr>
<tr>
<td>Keyword</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>SELFORM</td>
<td>Specifies a set of up to eight Forms names to be used as Spool file selection filtering criteria. See the description of the OUTPUT JCL statement FORMS parameter in z/OS MVS JCL Reference, SA22-7597 for details regarding Form names. Wild card characters may be specified. If the SELFORM parameter is not specified, then selection filtering by Form names is not performed. The Maximum Length is up to eight 8-character comma separated forms names. Example: To indicate that Spool files are to be selected that have an associated Forms name of either GREENBAR or any string beginning with the characters XYZ, enter SELFORM=(GREENBAR, XYZ*).</td>
</tr>
<tr>
<td>SELWTR</td>
<td>Specifies a SYSOUT Writer name to be used as a Spool file selection filtering criteria. See the description of the OUTPUT JCL statement FORMS parameter in z/OS MVS JCL Reference, SA22-7597 for details regarding SYSOUT Writer names. Wild card characters may be specified. If the SELWTR parameter is not specified, then selection filtering by Writer name is not performed. The Maximum Length is 8 characters. Example: To indicate that Spool files are to be selected, which have an associated Writer name of JCLWTR, enter SELWTR=JCLWTR.</td>
</tr>
<tr>
<td>TEMPPATH</td>
<td>Identifies the path to an HFS directory where temporary files created during the indexing and store processing are to be placed. See the IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1, SC27-1374 for a description of the ARSLOAD -c parameter. The Maximum Length is 255 characters. Example: To indicate that temporary HFS files are to be placed in the /usr/tmp directory, enter TEMPPATH=/usr/tmp.</td>
</tr>
<tr>
<td>TEMPUNIT</td>
<td>Identifies the MVS Unit Name to be used when dynamically allocating intermediate data sets. The Maximum Length is 8 characters and the default is SYSALLDA. Example: To specify an MVS Unit Name of SYSDA, enter TEMPUNIT=SYSDA.</td>
</tr>
<tr>
<td>USERSTRING</td>
<td>Specifies an arbitrary string that is presented to all ARSYSPIN user exits. The operand is accepted as is (that is, without case conversion). ARSYSPIN itself neither interprets nor modifies this string. The Maximum Length is 128 characters and is contained within quotes. For example: USERSTRING=&quot;TWS integration&quot;.</td>
</tr>
</tbody>
</table>
The UXnn statement is used to specify the state of an ARSYSPIN logical exit point and provides the name of a user written module to be associated with that exit point. The nn portion of the keyword is a two digit decimal number corresponding to a logical exit point. The operand must be specified in one of the following forms. (Note that the first suboperand is a boolean value.)

- (NO): Indicates the logical exit point is permanently disabled. (NO,name) Associates the module named name with the logical exit point, but the exit point is currently disabled. The exit point may subsequently be enabled or disabled via an MVS Modify command.

- (YES,name): Associates the module named name with the logical exit point, and the exit point is currently enabled. The exit point may subsequently be disabled or enabled via an MVS Modify command.

Note: Support for the changing of the state of an exit routine through the use of the MVS Modify command is not currently provided. This support will be provided in a subsequent PTF.

The maximum length of the name suboperand is 8 characters.

Example: To indicate User Exit 1 is never to be invoked, enter UX01=(NO).

Example: To indicate User Exit 2 is never to be invoked, enter UX02=(NO).

Example: To associate the module named MYUX03 with logical exit point 3, but with that exit point currently disabled, which means that module MYUX03 will not be invoked, enter UX03=(NO,MYUX03).

Example: To enable logical exit point 3 and associate the module named MYUX03 with that exit point, enter UX03=(YES,MYUX03).

ZSTRING Specifies a mixed case string to be used as the operand of the arsload -z command line argument. See the IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1, SC27-1374 for further details regarding the intended use of the arsload -z command line argument. The Maximum Length is 128 characters and the value must be contained with quotes, for example, ZSTRING="ODFDBOwner".
Install the ARSYSPIN procedure. Example 3-5 is a sample of the ARSYSPIN procedure that needs to be modified to suite your installation standards.

Example 3-5  ARSYSPIN procedure

```
//ARSYSPIN PROC
//ARSYSPIN EXEC PGM=ARSYSPIN,REGION=0M,TIME=NOLIMIT
//STEPLIB  DD DISP=SHR,DSN=ARS.SARSLOAD
//    DD DISP=SHR,DSN=CEE.SCEERUN
//    DD DISP=SHR,DSN=CBC.SCBCCMP
//    DD DISP=SHR,DSN=DB2G7.SDSNEXIT
//    DD DISP=SHR,DSN=DB2G7.SDSNLOAD
//    DD DISP=SHR,DSN=APK.ACIF.SAPKMOD1
//    DD DISP=SHR,DSN=ARSUSER.LOADLIB
//SYSIN    DD DUMMY,DCB=(LRECL=80,BLKSIZE=80,RECFM=FB)
//SYSPRINT DD SYSOUT=*                        
//SYSPRINT DD SYSOUT=*                        
//SYSOUT DD SYSOUT=*                          
//DSNAOINI DD PATH='/usr/lpp/ars/config/cli.ini'
//CEEDUMP  DD SYSOUT=*                       
//ARSYLIST DD SYSOUT=*                       
//ARSYPARM DD DISP=SHR,DSN=ARSUSER
```

Table 3-4 explains the ARSYSPIN JCL.

Table 3-4  ARSYSPIN JCL explained

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXEC</td>
<td>The EXEC statement should specify a REGION size of 0M. Currently, no PARM text is expected. If PARM text is supplied, it is ignored. However, to ensure compatibility with future enhancements, the EXEC statement PARM parameter should not be specified</td>
</tr>
<tr>
<td>Keyword</td>
<td>Explanation</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>STEPLIB</td>
<td>Depending upon your system configuration, the STEPLIB DD statement may or may not be required. If a STEPLIB DD statement is needed, all of the referenced libraries must be APF Authorized. The CEE.* and CBC.* libraries provide the C and other Language Environment run-time support modules. The DB2 libraries provide DB2 run-time support modules. The *.SARSLOAD library is where ARSYSPIN and other Related OnDemand modules (such as ARSLOAD) reside. The *.SAPKMOD1 library is where APKACIF resides.</td>
</tr>
<tr>
<td>SYSIN</td>
<td>This DD statement is used by the C run time for the STDIN stream. The statement should be specified exactly as shown.</td>
</tr>
<tr>
<td>SYSPRINT</td>
<td>This DD statement is used by the C run time for the STDOUT stream. See the OS/390 Language Environment for OS/390 and VM Programming Guide, SC28-1939 regarding the characteristics of the file referenced by this DD statement.</td>
</tr>
<tr>
<td>SYSOUT</td>
<td>This DD statement is used by the C run time for the STDERR stream. See the OS/390 Language Environment for OS/390 and VM Programming Guide, SC28-1939 for details regarding the characteristics of the file referenced by this DD statement.</td>
</tr>
<tr>
<td>CEE DUMP</td>
<td>This DD statement is used by the Language Environment run time for the production of serviceability and traceback information in the event of a run-time detected failure. See the OS/390 Language Environment for OS/390 and VM Programming Guide, SC28-1939 for details regarding the characteristics of the file referenced by this DD statement.</td>
</tr>
</tbody>
</table>
Figure 3-7 on page 64 shows an example OnDemand screen showing the MSGCLASS files that have been collected for a given day.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARSYLIST</td>
<td>The file referenced by this DD statement contains the messages produced by ARSYSPIN. Any file supported by QSAM may be specified. This file is always written as RECFM=FBA, LRECL=133. If a BLKSIZE value is not supplied, a system determined BLKSIZE value is used.</td>
</tr>
<tr>
<td>ARSYPARM</td>
<td>The file referenced by this DD statement contains the control statement stream. Any file supported by QSAM may be specified. LRECL: Any record length up to a maximum value of 512 is accepted. The smallest record length is that which is needed to fully contain a keyword/operand pair. From a practical point of view, the smallest record length is 80. RECFM: Any of the following are supported: RECFM=F, RECFM=FB, RECFM=V, RECFM=VB.</td>
</tr>
</tbody>
</table>
3.4.7 Install the ARSLOAD procedure

The ARSLOAD procedure has been created to allow the collection of user reports directly from the JES Spool running continuously as a started task.

The reports are located on the JES Spool and processed by the OnDemand indexing feature using the keys of DDNAME (*dataset*) and FORM. The indexing feature is used to enable OnDemand to know which fields are to extracted and which are to be indexed.

The OnDemand application group defines the fields to be indexed and the application name defines where within the input stream these fields can be located.
An application group can contain one or more Applications but, by definition, this means that all application names within the same application group must have the same indexing even if the values to be obtained for the index are not necessarily in the same position within the input stream.

For example, if we define an application group and we define the index fields to be DATE, TIME, Report Title, and Department, then every application name defined within the same application group would have to be able to extract these values from the input stream even though the data may be in different columns or rows.

In the ARSLOAD sample procedure shown in Example 3-6, we have specified that the application group name will be derived from the FORM name (G=FORMS) specified on the sysout statement of the users JCL and that the application name (A=DATASET) will be derived by using the DDNAME that defines the user dataset. The class we will be searching for is specified by the C=Y parameter, in this case, sysout class Y.

**Example 3-6  ARSLOAD procedure**

```plaintext
//ARSLOAD  PROC I='9.12.6.31', /* TCP/IP Server Address */
//         H=ARSDBASE, /* OnDemand Server Name */
//         P=ONDEMAND, /* UserID password */
//         G=FORMS, /* Formname defines GROUP */
//         A=DATASET, /* DD NAME defines APPLIC */
//         T=10, /* Polling time interval */
//         U=TWS, /* OnDemand UserID */
//         C=Y /* SYSOUT CLASS */
////////////////////////////////////////////////////////////////////////////////////////
// ARSLOAD for collecting Reports from the JES Spool
////////////////////////////////////////////////////////////////////////////////////////
//ARSLOAD EXEC PGM=ARSLOAD,REGION=0M,TIME=NOLIMIT,
// PARM=('/-h &H -I &I -u &U -p &P -C &C -G &G -A &A -t &T')
//STEPLIB DD DISP=SHR,DSN=ARS.SARSLOAD
//         DD DISP=SHR,DSN=DB2G7.SDSNEXIT
//         DD DISP=SHR,DSN=DB2G7.SDSNLOAD
//         DD DISP=SHR,DSN=APK.ACIF.SAPKMOD1
//         DD DISP=SHR,DSN=ARSUSER.LOADLIB
//SYSPRINT DD SYSOUT=*,RECFM=FBA,LRECL=121,BLKSIZE=6050
//SYSOUT DD SYSOUT=* 
```

For example, a DD statement in a users JCL stream similar to //ADREPORT DD SYSOUT=(Y,,EQQR) would cause OnDemand to expect an application group named EQQR (the FORM name) and within that application group, an application called ADREPORT (the DDNAME containing the sysout parameters).
3.4.8 Install the ARSBATCH procedure

The ARSBATCH procedure has been created to allow the user to process reports written to datasets instead of directly from the JES Spool. The processing is very similar to that employed by the ARSLOAD procedure, except that in the ARSBATCH environment, the parameter string and JCL is slightly different.

In the sample ARSBATCH procedure shown in Example 3-7, we have specified that the application group name will be derived from the FORM name (G=EQQR) used on sysout statements of the users JCL and that the application name (A=UNKNOWN by default) will be supplied when we invoke the ARSBATCH procedure. In the ARSBATCH procedure, we are not searching for a specific sysout class, but we are using the -s INPUT parameter to read a user report from the DDNAME in the ARSBATCH task known as INPUT. We will supply the dataset name to be processed when we execute the ARSBATCH procedure by using the FILE variable.

Example 3-7  ARSBATCH procedure

```
// ARSBATCH PROC H=ARSDBASE, /* OnDemand Server id */
// U=TWS,  /* OnDemand UserID */
// G=EQQR, /* OnDemand Group Name */
// A=UNKNOWN, /* OnDemand Applic name */
// P=ONDEMAND, /* UserID password */
```
Unlike the ARSLOAD procedure which, once started, will continue to run until stopped by the z/OS stop command, the ARSBATCH task will end immediately after it has processed the INPUT file.

Notes:
- You should note the use of the upper and lower case characters used for the special keywords within the parameter string, for example, -h, -n, -g, -a, and -s INPUT. The use of upper and lower case is very significant and can cause unexpected results if the incorrect values are used.
- The use of the -n character denotes that the input file is not to be deleted after it has been processed by OnDemand.
- The use of the lower case -g and -a in this example differs from the upper case values used in the ARSLOAD procedure.

ARSBATCH execution

The ARSBATCH procedure is run as a batch job in its own right or as an additional step to an existing job. In the sample given in Example 3-8, we are running the ARSBATCH procedure to read from IBM Tivoli Workload Scheduler planning and database reports that were written to datasets instead of the JES Spool. Running the ARSLOAD started task can achieve the same function as the ARSBATCH process, but has been added into this document for completeness.
3.4.9 Summary of started tasks

Example 3-9 contains a sample display of the tasks that should be running to support the IBM Tivoli Workload Scheduler for z/OS and OnDemand integration. It shows a list of typical IBM Tivoli Workload Scheduler for z/OS tasks, DB2 address spaces, and OnDemand procedures, all of which need to be active to support the integrated environment.

Example 3-9   WS, DB2 and OnDemand started tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TWSC</td>
<td>TWS for z/OS Controller</td>
</tr>
<tr>
<td>TWSCJSC</td>
<td>TWS for z/OS JSC server</td>
</tr>
<tr>
<td>TWSCTCP</td>
<td>TWS for z/OS E2E server</td>
</tr>
<tr>
<td>TWSDDST</td>
<td>TWS for z/OS datastore server</td>
</tr>
<tr>
<td>TWST</td>
<td>TWS for z/OS Tracker Agent</td>
</tr>
<tr>
<td>DB7GDBM1</td>
<td>DB2 manager</td>
</tr>
<tr>
<td>DB7GDIST</td>
<td>DB2 DDF</td>
</tr>
<tr>
<td>DB7GIRLM</td>
<td>DB2 lock manager</td>
</tr>
<tr>
<td>DB7GMSTR</td>
<td>DB2 control</td>
</tr>
<tr>
<td>DB7GSPAS</td>
<td>DB2 stored procedures</td>
</tr>
<tr>
<td>ARSLOAD</td>
<td>ARS task for handling special forms</td>
</tr>
<tr>
<td>ARSSOCKD</td>
<td>ARS DB2 communicator</td>
</tr>
<tr>
<td>ARSYSSPIN</td>
<td>ARS task for handling MSGCLASS sysout</td>
</tr>
</tbody>
</table>
3.4.10 Collecting ITWS for z/OS reports from the JES Spool

Once you have all the necessary tasks started and you have imported the application groups and folders, you are now ready to change the relevant JCL in order to start archiving your IBM Tivoli Workload Scheduler for z/OS reports.

In Example 3-10, we have modified the JCL for the IBM Tivoli Workload Scheduler for z/OS print Application Descriptions job so that the report will be archived to OnDemand.

Example 3-10  TWS for z/OS print Application Descriptions report to JES Spool

```
//ADPRT    EXEC PGM=EQQBATCH,PARM='EQQADPRT',REGION=2048K
//EQQMLIB  DD DISP=SHR,DSN=EQQ.SEQQMSG0
//EQQPARM  DD DISP=SHR,DSN=EQQUSER.TWS810.PARM(BATCHOPT)
//ADREPORT DD SYSOUT=(Y,,EQQR),
//   DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6050)
//ADREPORT DD DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6050),
//   SPACE=(CYL,(3,1)),DISP=(NEW,DELETE),UNIT=SYSDA
//EQQMLOG  DD SYSOUT=* 
//SYSOUT   DD SYSOUT=* 
//SYSPRINT DD SYSOUT=* 
//SYSMDUMP DD DISP=MOD,DSN=EQQUSER.TWS810.SYSDUMPB
//EQQDUMP  DD SYSOUT=* 
//EQQDMSG  DD SYSOUT=* 
//ADPRIN   DD DCB=(LRECL=200,BLKSIZE=3200,RECFM=FB),
//   SPACE=(CYL,(3,1)),DISP=(NEW,DELETE),UNIT=SYSDA
//ADPROUT  DD DCB=(LRECL=200,BLKSIZE=3200,RECFM=FB),
//   SPACE=(CYL,(3,1)),DISP=(NEW,DELETE),UNIT=SYSDA
//ADPRWK01 DD SPACE=(CYL,(3,1)),UNIT=SYSDA
//ADPRWK02 DD SPACE=(CYL,(3,1)),UNIT=SYSDA
//ADPRWK03 DD SPACE=(CYL,(3,1)),UNIT=SYSDA
//EQQADDS  DD DSN=EQQUSER.TWS810.AD,DISP=SHR
//EQQAD2DS DD DSN=EQQUSER.TWS810.AD,DISP=SHR
//ADUSERDS DD DSN=NULLFILE,DISP=SHR
//SYSIN    DD *
1100000000
```

You can see in Example 3-10 that the ADREPORT DD statement has been modified to ensure we collect the data and this statement is shown below for clarification:

```
//ADREPORT DD SYSOUT=(Y,,EQQR)
```

Specifying the sysout CLASS of Y and the FORM name of EQQR will ensure the ARSLOAD procedure will find the data and know how to archive and index it correctly.
3.4.11 Collecting ITWS for z/OS reports from datasets

In Example 3-11, we have modified the JCL for the IBM Tivoli Workload Scheduler for z/OS print Application Descriptions job so that the report will be written to a dataset.

Example 3-11  Modified JCL for TWS for z/OS print Application Descriptions job

```
//TWSRESX JOB 'ACCNT#',REGION=6072K,NOTIFY=TWSRES1,MSGCLASS=W
//*OPC BATCHJOBS
//*JOBPARM SYSAFF=SC64
//DELETE   EXEC PGM=IEFBR14
//OLDLIST DD DSN=EQQUSER.TWS810.ADPRINT,
//         UNIT=SYSDA,SPACE=(TRK,0),DISP=(MOD,DELETE)
//ALLOC   EXEC PGM=IEFBR14
//NEWLIST DD DSN=EQQUSER.TWS810.ADPRINT,
//         UNIT=SYSDA,DISP=(,CATLG),SPACE=(CYL,(2,5),RLSE),
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=12100)
//ADPRT   EXEC PGM=EQQBATCH,PARM='EQQADPRT',REGION=2048K
//EQQMLIB DD DISP=SHR,DSN=EQQ.SEQQMSG0
//EQQPARM DD DISP=SHR,DSN=EQQUSER.TWS810.PARM(BATCHOPT)
//ADREPORT DD DSN=EQQUSER.TWS810.ADPRINT,DISP=SHR,
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6050)
//ADREPORT2 DD DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6050),
//          SPACE=(CYL,(3,1)),DISP=(NEW,DELETE),UNIT=SYSDA
//EQQMLOG DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSDUMP DD DISP=MOD,DSN=EQQUSER.TWS810.SYSDUMPB
//EQQDUMP DD SYSOUT=*  
//EQQDMSG DD SYSOUT=*  
//ADPRIN DD DCB=(LRECL=200,BLKSIZE=3200,RECFM=FB),
//         SPACE=(CYL,(3,1)),DISP=(NEW,DELETE),UNIT=SYSDA
//ADPROUT DD DCB=(LRECL=200,BLKSIZE=3200,RECFM=FB),
//         SPACE=(CYL,(3,1)),DISP=(NEW,DELETE),UNIT=SYSDA
//ADPRWK01 DD SPACE=(CYL,(3,1)),UNIT=SYSDA
//ADPRWK02 DD SPACE=(CYL,(3,1)),UNIT=SYSDA
//ADPRWK03 DD SPACE=(CYL,(3,1)),UNIT=SYSDA
//EQQADDS DD DSN=EQQUSER.TWS810.AD,DISP=SHR
//EQQAD2DS DD DSN=EQQUSER.TWS810.AD,DISP=SHR
//ADUSERDS DD DSN=NULLFILE,DISP=SHR
//SYSIN    DD *
```

You can see in the example that the ADREPORT DD statement has been modified to ensure we collect the data to a new dataset:

```
//ADREPORT DD DSN=EQQUSER.TWS810.ADREPORT,DISP=SHR,
```
Writing the report in this way means we can collect the data at later stage by using the ARSBATCH procedure and at the same time keep the dataset intact.

To collect the TWS for z/OS Application Description report and load it into the OnDemand DB2 database, we would execute the ARSBATCH procedure:

```
// EXEC ARSBATCH,FILE='EQQUSER.TWS810.ADPRINT',A=ADREPORT
```

### 3.4.12 ITWS for z/OS database and planning reports

Figure 3-8 on page 72 lists the IBM Tivoli Workload Scheduler for z/OS reports supported by this redbook and identifies the SYSOUT class and FORM and DDNAME to be modified. In the case of the Daily Planning reports, the SYSPRINT statement refers to the definition used in the reporting step. At the time of the writing of this redbook, this was the last step of the job, and was known as DPREPORT.
You should also modify any IBM Tivoli Workload Scheduler for z/OS skeletons and any scheduled IBM Tivoli Workload Scheduler reporting jobs as required, making sure to modify the correct dd statement and correct sysout class and forms name.

Figure 3-9 on page 73 shows what a possible on demand screen could look like to a user searching for all IBM Tivoli Workload Scheduler for z/OS reports defined to the system for a specific day. The pull down shows how individual report types can be filtered, but the default is to select all.
3.4.13 Collecting IBM Tivoli Workload Scheduler for z/OS log files (EEQMLOG)

In order to collect the message log files created by the IBM Tivoli Workload Scheduler for z/OS started tasks, such as CONTROLLERs, E2E servers, DATASTORE servers, JSC servers, and TRACKERs, we need to add some information into each of these tasks primary parameter file members in order for the OnDemand indexing function to store the relevant data. We also need to change the EEQMLOG dd statements within the relevant started tasks JCL to point to the correct forms name that we are going to use. The FORMS name we are going to use for this purpose has been defined as EEQM (the same as the OnDemand application group name) and the application name will be the same as the DD statement for message logs (EEQMLOG).
Example 3-12 and Example 3-13 show how the EQQMLOG statement would look in the IBM Tivoli Workload Scheduler for z/OS TRACKER started task and how it is used to accomplish the storage of its message log into the OnDemand and also an example of the modification (new comment line added) in the relevant parameter file.

Example 3-12  IBM Tivoli Workload Scheduler for z/OS Tracker procedure

```plaintext
//TWST      EXEC PGM=EQQMAJOR,REGION=0M,PARM='TRAP',TIME=1440
//***********************************************************
//* TRACKER.
//***********************************************************
//EQQMLIB   DD  DISP=SHR,DSN=EQQ.SEQQMSG0
//EQQMLOG   DD  SYSOUT=(Y,,EQQM)
//EQQPARM   DD  DISP=SHR,DSN=EQQUSER.TWS810.PARM
//EQQBRDS   DD  SYSOUT=(A,INTRDR)
//EQQEVD01  DD  DISP=SHR,DSN=EQQUSER.TWS810.EV     *OPTIONAL*
//EQQEVDS   DD  DISP=SHR,DSN=EQQUSER.TWS810.EV
//EQQSUDS   DD  DISP=SHR,DSN=EQQUSER.TWS810.SU
//EQQJCLIB  DD  DISP=SHR,DSN=EQQUSER.TWS810.JCLIB
//EQQINCKW  DD  DISP=SHR,DSN=EQQUSER.TWS810.INCWORK
//EQQSTC    DD  DISP=SHR,DSN=EQQUSER.TWS810.STC
```

**Note:** Note the use of (Y,,EQQM) on the EQQMLOG DD statement.

Example 3-13  TWS for z/OS Tracker parameters

```plaintext
/* OPCOPTS: run-time options for the TRACKER processor                          */
/*********************************************************************************/
OPCOPTS  OPCHOST(NO)
/* ARS-ONDEMAND: TASK=TWST , TYPE=TRACKER , MVSID=&SYSNAME
ERDRTASK(0)
ENTRTASK(YES)    EWTRPARAM(STDWTR)
JCCTASK(NO)
NCFTASK(NO)
*/

JOBOPTS  JOBLOGRETRIEVAL(IMMEDIATEALL)
TRROPTS  HOSTCON(XCF)

XCFOPTS  GROUP(TWS810)
         MEMBER(&SYSNAME.TWST)
```
**Attention:** See the inclusion of the comment line that is used to pass indexing information to OnDemand. This should *never* be placed before the first control statement and should *always* appear in the first 255 lines of the EQQMLOG listing. Failure to observe this will result in incorrect indexing.

The /* ARS-ONDEMAND: comment starting in column 1 contains positional parameters and must be coded, as indicated in Example 3-14.

**Example 3-14  * ARS-ONDEMAND: comment**

```plaintext
/* must be present and start in column ................. 01 length 02
ARS-ONDEMAND: must be present and start in column .... 04 length 13
The started TASK name must be present and start in column 23 length 08
The TWS log TYPE name must exist and start in column .... 38 length 12
&SYSNAME must be present and start in column .......... 58 length 20
```

The TYPE field is used to identify the server type producing the log. The following types are supported by this redbook, but others can be added as required:

- CONTROLLER
- TRACKER
- DATASTORE
- JSC
- E2E

The task name field is used to identify the started task producing the log that is being archived, and the &SYSNAME variable, which is resolved by the z/OS itself, indicates the machine where the task is running.

Figure 3-10 on page 76 shows what a possible OnDemand screen could look like to a user searching for all message logs defined to the system for a specific day. The pull down shows how individual server types can be filtered, but the default is to select all.
Figure 3-10 OnDemand screen showing the TWS for z/OS EQMLOGS collected
IBM Tivoli Workload Scheduler Distributed implementation

This chapter describes the procedure to implement the Integration of IBM Content Manager OnDemand for Multiplatforms Version 7.1 (OnDemand) and IBM Tivoli Workload Scheduler Version 8.2 in a distributed environment.

In this chapter, the following topics will be discussed:

- Section 4.1, “Overview” on page 78
- Section 4.2, “Prerequisites” on page 79
- Section 4.3, “Implementation” on page 88
- Section 4.4, “Scenarios” on page 109
- Section 4.5, “Troubleshooting” on page 111
4.1 Overview

IBM Tivoli Workload Scheduler scheduled operations that include the launching of scheduled, adhoc jobs and managing daily operations can produce vast amounts of logs, such as job execution logs (stdlist file) and IBM Tivoli Workload Scheduler control processes, such as TWSmerge, Netman process (Netman), and Auditing (audit database and plan).

The integration of OnDemand and IBM Tivoli Workload Scheduler will consolidate all these logs and archive them to a central location that will then facilitate the retrieval of information in these logs. The timeliness or availability of this data will depend on the architecture that is adopted and the frequency of the updates. It maybe possible to access the job stdlist on the OnDemand Server minutes after the job has completed.

Once the data is archived, the data can be viewed via the OnDemand Client or Web-based dialogues (OnDemand Web Enablement Kit). The log that will be viewed or queried will determine the type of information that will be available, the search criteria, and the format of the output. Just think of the flexibility this provides to your user community, other scheduling teams, and your executive management.

This solution can be implemented on UNIX (AIX, HP-UX, Sun Solaris, and Linux) and requires the use of scripts to extract and load data, IBM Tivoli Workload Scheduler jobs, IBM Tivoli Workload Scheduler job streams, and OnDemand application templates for different types of logs that will be archived and templated for viewing data. The scripts, IBM Tivoli Workload Scheduler jobs, and IBM Tivoli Workload Scheduler job streams will be on the IBM Tivoli Workload Scheduler Agents, and OnDemand application templates will be on the OnDemand Server.
4.2 Prerequisites

Before any discussion of the solution, prerequisites must be identified for integrating IBM DB2 Content Manager OnDemand for Multiplatforms Version 7.1 (OnDemand) and Tivoli Workload Scheduler Version 8.2.

The integration referred by this redbook assumes that the IBM DB2 Content Manager OnDemand for Multiplatforms Version 7.1 (OnDemand), DB2 Universal Database™ Enterprise Edition Version 7 (DB2 is included with OnDemand) and Tivoli Workload Scheduler Version 8.2 are installed, configured, have the latest maintenance patches, and are working correctly.

Figure 4-1 on page 80 shows our lab environment for the ITWS/OnDemand integration solution.
4.2.1 Hardware requirements

This solution requires that you have at least one UNIX CPU with Tivoli Workload Scheduler Version 8.2 installed and this CPU be at least a Fault-Tolerant Agent (FTA).

Note: If the UNIX scripts have been converted for use with Windows systems, this requirement of a UNIX FTA may not be necessary, depending on the adopted architecture.

This UNIX FTA must have enough disk storage space to accommodate storing of logs that may be in the process of being archived to the OnDemand Server. If either the network or the OnDemand Server is down, this UNIX FTA should have...
enough disk space to accommodate these files until the network and OnDemand Server are up and able to process the data.

4.2.2 Software requirements

This solution requires that the following products, with their respective version and maintenance patches be applied:

- IBM Tivoli Workload Scheduler Version 8.2
- IBM DB2 Content Manager OnDemand for Multiplatforms Version 7.1 (OnDemand) with patch 7.1.014
- DB2 Universal Database Enterprise Edition Version 7 (DB2) with upgrade RSU0303

Figure 4-1 on page 80 shows our lab environment for the ITWS/OnDemand integration solution.

4.2.3 IBM Tivoli Workload Scheduler specific requirements

The IBM Tivoli Workload Scheduler Version 8.2 environment must be defined to use the *expanded database* by default, since this solution uses long job and job stream names; otherwise, the provided job and job stream definitions will need to be renamed accordingly.

**Note:** Expanded databases is the default setting for new installations of IBM Tivoli Workload Scheduler Version 8.2. Verify if expanded databases are already being used by reviewing the TWShome/mozart/globalopts file if you are upgrading from Tivoli Workload Scheduler Version 7.0 or 8.1. For more details, refer to Chapter 1, “Introduction” in *IBM Tivoli Workload Scheduler Planning and Installation Guide Version 8.2*, SC32-1273.

The merge stdlists option in the TWShome/localopts file should be set to the default value of yes. This is necessary because all IBM Tivoli Workload Scheduler control processes, except Netman, can send their console messages to a single standard list file called TWSmerge file. The console messages from Netman process are sent to a file called Netman.

4.2.4 Environmental variables requirements

Since the implementation of this solution requires the use of scripts that require different settings, options, and environmental settings, a script with this information called tws_ondemand_env.sh will be sourced by the scripts to set up necessary variables that will be used by them. This facilitates the setting of
options to one file and prevents the need to modify individual scripts. Most of the
script names used will have the prefix of tws_ondemand_ and should be in a
central location.

The environment script (tws_ondemand_env.sh) is used to define variables that
will be used by other scripts. Most of the values must remain unchanged unless
otherwise stated in the following sections. The environment script has four main
sections: LOAD, ARSLOAD, ONDEMAND SERVER, and EVENT MONITOR,
which are variables that will require your customization. These are the only
sections that may be customized.

**LOAD variable**
The LOAD variable is used to identify if the information that will be loaded to the
OnDemand Server will be in real-time or batch mode. The default setting of
LOAD_TYPE variable is REAL and it means that job stdlists will be loaded to
OnDemand Server as soon as the job completes, provided that the network and
OnDemand Server are available.

As discussed in step d of “Push method and additional requirements” on page 92
and in step d of “Pull method and additional requirements” on page 99, the
OnDemand process called ARSLOAD is invoked at system startup and must be
running for the loads to OnDemand Server to occur.

If you do not want to load job stdlists to the OnDemand Server once jobs
complete, but defer it to a later time, you will need to set LOAD_TYPE as BATCH
and create a job stream to schedule a job that loads all job stdlists to the
OnDemand Server at a specified time. You must not perform step d of “Push
method and additional requirements” on page 92.

**ARSLOAD variables**
The ARSLOAD variables are used to define the location of a job execution’s
output and error logs, and may be customized. The ARSLOAD_SERVER
variable must be defined as the OnDemand Server’s host name. It will be used
by the ARSLOAD program to load data to the OnDemand Server.

| Note: | If the ARSLOAD_SERVER variable is not set as the OnDemand Server's host name, attempts to load data will fail. |

| Note: | The ARSLOAD_TWS and ARSLOAD_TYPE variables must not be changed, because their values are used by other scripts. |
ONDEMAND SERVER variables
The ONDEMAND SERVER variables are used to identify the OnDemand Server's IP address and port number. The ODSERVER_IP variable must be set to the OnDemand Server's IP address. It is used to determine if the OnDemand Server is available to receive loads. The ODSERVER_PORT will not require customization if the default port number 1445 is used. The OnDemand Server install operation uses this port number as the default port. Contact your OnDemand administrator if you are unsure if the OnDemand Server is using the default port number.

EVENT MONITOR variables
The EVENT MONITOR variables are used to identify the location of specific files. The BMEVENT_FILE variable does not need to be customized and is used to identify the location of the event.log file.

The BMEVENT_CPULIST variable is used to identify the location of the cpulist file, which will contain the list of workstations (CPUs) to be managed or non-managed. This variable does not need to be customized unless your file location changes.

The BMEVENT_TYPE variable is used to denote that the cpulist file will contain either an include or workstations to be managed or non-managed, respectively. The default value is “1” (include), and it means that all workstations listed in the cpulist file will have their data loaded to the OnDemand Server.

Note: If the cpulist file does not exist or does not have content, all workstations defined in the IBM Tivoli Workload Scheduler environment will have their data loaded to the OnDemand Server.

If the BMEVENT_TYPE value is “0” (exclude), it means that all workstations listed will not have their data loaded to the OnDemand Server.

Note: Regardless of the value set for BMEVENT_TYPE, if the cpulist file does not exist or does not have content, all workstations defined in the IBM Tivoli Workload Scheduler environment will have their data loaded to the OnDemand Server.

4.2.5 OnDemand specific requirements
The implementation of this solution requires that you install the provided applications that will identify the data that will be loaded onto the OnDemand Server and how it will be viewed.
A zip file (TWS4DIST.zip) has been provided that contains the required OnDemand definition files for loading and viewing the IBM Tivoli Workload Scheduler job logs. The implementation of this solution requires that you install the definition files by completing the following steps:

1. Download the zip file containing the OnDemand application group and folder definitions used for the OnDemand and IBM Tivoli Workload Scheduler integration.

2. Unzip the downloaded export file referred to above.

3. Import the previously unzipped definitions.

4. Ensure you have valid user IDs and passwords in OnDemand to authorize loading data into the database.

5. Ensure you have valid user IDs and passwords in OnDemand to search and view data in the database.

Once these steps have been completed, OnDemand will be ready to automatically load and retrieve IBM Tivoli Workload Scheduler log file.

### 4.2.6 Download, unzip, and import the OnDemand definitions

In this redbook, we provide sample applications, application groups, and folders that can be used when setting up the OnDemand and IBM Tivoli Workload Scheduler integration. The samples can be used with minimal changes or they can be used as a template for creating a customized set of definitions. For a description of these categories, see 1.3.1, “Definition files” on page 6.

The only required change is in the application group. The storage management information of the application group needs to be updated to reflect the storage set and data retention values required for the user's environment. Application, application group, and folder definitions samples are provided as a zip file (TWS4DIST.ZIP) with this redbook. To download the zip file, use the instructions in Appendix C, “Additional material” on page 243.

Figure 4-2 on page 85 shows the structure of the TWS4DIST.ZIP file.
The following information describes how to define an application, application group, and folder using the samples:

1. Using a binary file transfer mechanism (such as FTP), transfer the TWS4DIST.ZIP file to a temporary or working directory on your PC.

2. Execute PKZIP (or its equivalent) to extract the contents of the TWS4DIST.ZIP file and recreate the proper directory structure. For example, the following command will create a directory named TWS4DISTSAMP anchored within the root directory of the C: drive:

   \[ pkzipc -ext -dir c:\TWS4DIST.ZIP c:\ \]

   - The -ext parameter indicates that PKZIPC is to extract the contents of the TWS4DIST.ZIP file.
   - The -dir parameter indicates that the directory structure contained in the TWS4DIST.ZIP file is to be recreated.

The TWS4DISTSAMP directory will, in turn, contain three subdirectories named LOAD, TABLE, and VIEW. The files contained in this set of directories comprise the system tables that define a local server.

The local server contains the definitions for the sample application, application group, and folder.

The following steps describe how to create a local server, export the definition files to the distributed OnDemand Server, and update those definition files to comply with your server environment.
1. Start the Administrative Client.
   a. From the Windows task bar button, select **Start → Programs → IBM OnDemand32 → OnDemand administrator**.

2. Add the local server to the server list.
   a. Highlight **OnDemand Servers**.
   b. From the File menu, select **New Server**.
   c. Type **TWS4DISTAMP** in the Server field.
   d. Change the Protocol to Local by selecting **Local** from the Protocol combo box.
   e. Enter the full path name of the TWS4DISTAMP directory (or click on the **Browse** button and expand the directory list for the drive where the TWS4DISTAMP directory was added by clicking on the + sign next to the drive name; select the **TWS4DISTAMP** directory and click on the **OK** button).
   f. Click on the **OK** button to add the local server.

3. Log on to the TWS4DISTAMP server.
   a. Double-click the left mouse button on the TWS4DISTAMP server.
   b. Enter **admin** as the user ID and leave the password blank (no password is required), then click **OK** to log on to the server.

4. Export the sample application groups to the OnDemand Server.
   a. Select the application groups located below the TWS4DISTAMP server name in the left pane of the window.
   b. Highlight all the application groups you wish to export in the right pane of the window, then press the right mouse button. Select **Export** from the pop-up menu. The supplied application groups are **stdlist, netman, twsmerge, database, and plan**.
   
   **Note:** Each application group must be exported to the distributed OnDemand Server.

   c. Select the name of your distributed server from the Server combo box.
   d. Click on the **No Storage Set** check box.
e. Click on the **Export** button to display the Logon dialog box for the distributed server.

i. If you are currently logged on to the distributed server, verify that the logged on user ID has the authority to create application groups and folders. Otherwise, complete the logon in the following step.

ii. Enter a user ID and password for the distributed server and click on the **OK** button to log on to the server.

f. Click on the **OK** button to complete the export.

---

**Important:** Any applications linked to an application group will automatically be exported at the same time. If you do not need a particular application, e.g., no Linux FTA's, simply delete the Linux application. However, if you delete the application group, the application will also be deleted.

5. Export the TWS Job Logs sample folder to the distributed OnDemand Server.

   a. Repeat all the steps in Item 4, substituting *folder* for *application group*. (Logging on again to the distributed server is not necessary unless you logged off after the last export operation.)

6. Update the selected application group on the distributed OnDemand Server to specify a Storage Set.

   a. Point to the distributed server in the left pane of the window and expand the server by clicking on the + sign.

   b. Select the **Application Groups** category located below the distributed OnDemand Server name in the left pane of the window.

   c. For each of the application groups that were exported in Item 4 above, select **Update** from the pop-up menu for the application group you are currently updating.

   d. Click on the **Storage Management** tab to display Storage Management information.

   e. Select the name of an appropriate storage set from the Storage Set Name combo box. OnDemand uses the storage set to maintain the log file data in cache storage and in archive storage.

   f. (Optional) If necessary, change the parameters that identify how long data is maintained in cache and how long the data and indexes will be retained in OnDemand.

   g. Click on the **OK** button to update the application group with the changes.

7. (Optional) If necessary, update the application you are currently updating on the distributed OnDemand Server to change the code page.
a. Point to the distributed OnDemand Server in the left pane of the window and expand the server by clicking on the + sign.

b. Select the Applications category located below the server name in the left pane of the window.

c. Select Update from the pop-up menu for the application you are currently updating.

d. Click on the View Information tab to display information used to display the data.

e. Change the code page field value from 1047 to the code page of the data (for example 037, 500)

f. Click on the OK button to update the application with the change.

After the application, application group, and folder have been defined, the OnDemand Server is ready to begin receiving indexing and loading jobs.

The loading of data to the OnDemand Server requires that it be loaded by a valid OnDemand user. The solution requires that you create an OnDemand user called tws whose only function will be to load the data to the OnDemand Server. This user will not be able to do anything else. Contact your OnDemand administrator for assistance or refer to 7.2, “User and group administration” on page 171.

### 4.3 Implementation

This section will discuss two methods (Push and Pull) to implement the Integration of IBM Content Manager OnDemand for Multiplatforms Version 7.1 (OnDemand) and IBM Tivoli Workload Scheduler Version 8.2.

The Push method will always be used by at least one system and the use of the Pull method is optional. Various scenarios are possible; we will identify some of the possible combinations that may be considered.

Although there are only two methods (Push and Pull), there are three types of CPUs: PUSH-CPUs, PULL-CPUs, and MANAGED-CPUs.

The PUSH-CPUs refer to the IBM Tivoli Workload Scheduler Agents that will be pushing data to the OnDemand Server either via real-time or batch mode. PUSH-CPUs can be any UNIX Master Domain Manager, Backup Master, UNIX Domain Manager UNIX FTA, or UNIX Standard Agent. Windows FTAs may also be PUSH-CPUs if the UNIX scripts and logic for this solution have been converted for Windows.
The PULL-CPUs refer to the IBM Tivoli Workload Scheduler Agents that will be extracting data from MANAGED-CPUs that will later be loaded to the OnDemand Server. PULL-CPUs can be any UNIX Master Domain Manager, Backup Master, UNIX Domain Manager, or UNIX FTA.

The MANAGED-CPUs refer to the Extended Agents, including Oracle, PeopleSoft and SAP, Windows Standard Agents, Linux FTAs, Linux Standard Agents, Linux Extended Agents, Windows Master Domain Managers, Backup Masters, Windows Domain Managers, or Windows FTAs that will be managed by PULL-CPUs.

Each of the above CPUs and their requirements will be discussed later in the chapter.

Note: The Windows systems are included as MANAGED-CPUs because this solution does not have the scripts converted to a format that will work on a Windows system. If these scripts are developed, a Windows system could be a PUSH-CPU.

4.3.1 Method type to use

The method type to use will depend on your IBM Tivoli Workload Scheduler environment, workstation disk space, workstation capacity, and the need for timeliness access to the information.

The Push method should be the desired approach for loading data to the OnDemand Server. Use of this method loads the job stdlist to the OnDemand Server when the job completes and loads IBM Tivoli Workload Scheduler logs (TWSmerge, Netman, Audit Database, and Plan logs) to the OnDemand Server once a day. The advantage of this method is that each PUSH-CPU is responsible for its own jobs and it will not impact any other IBM Tivoli Workload Scheduler Agent.

Note: To ensure completeness of TWSmerge log, Netman log, Audit Plan, and Database logs, these logs will be archived the day after their production day. This means that the loading of IBM Tivoli Workload Scheduler logs (TWSmerge log, Netman log, Audit Plan, and Database logs) will be from the previous day, so there may be a 24 hour delay in the availability of these logs and information on the OnDemand Server.

The Pull method requires that a PULL-CPU pull all job stdlists (TWSmerge log, Netman log, Audit Plan, and Database log) (if they exist) from its MANAGED-CPUs and load them to the OnDemand Server. Use of this method
will require that the PULL-CPUs have sufficient disk space to store pending data loads until they get loaded onto the OnDemand Server.

**Important:** If the OnDemand Server is down and the IBM Tivoli Workload Scheduler Agents are linked, the number and size of these files could have an impact on the PULL-CPU.

Possible candidates for the PULL-CPU are UNIX Master Domain Managers, UNIX Backup Masters, UNIX Domain Managers, and UNIX FTA’s. These CPUs would then manage their agents. If the PULL-CPUs are UNIX Domain Managers, they could manage the CPUs within their domain. If you want to dedicate one CPU to be responsible for pulling data from all its MANAGED-CPUs, it must be a PULL-CPU.

**Important:** If you have a Windows Master, you will need to have it managed by a PULL-CPU.

The only task that must be performed on the MANAGED-CPUs is to launch a job that will contain the TWSmerge log, Netman log, Audit Plan, and Database logs (if they exist), generating a job stdlist. This information will be pulled by the PULL-CPU later.

### 4.3.2 Push method

The Push method discussed in this section will use the real-time frequency for loading the job stdlists and use batch mode for loading the TWSmerge log, Netman log, and, when applicable, the Audit Plan and Database logs.

The Push method is one of the most efficient methods of loading logs to the OnDemand Server because when a job completes, its job stdlist is loaded to the OnDemand Server automatically. If this method is implemented for each UNIX Master Domain Manager, UNIX Backup Master, UNIX Domain Manager, UNIX Fault Tolerant Agent, and UNIX Standard Agent, each system will be responsible for loading its logs to the OnDemand Server. As soon as the load is complete, the job stdlists will be available for viewing or querying on the OnDemand Server.
Push method and real-time operations
As discussed earlier, there are two options for moving data from IBM Tivoli Workload Scheduler to the OnDemand Server: real-time or batch mode. In the real-time option, the job stdlists will be loaded to the OnDemand Server as soon as the job completes, provided that the network and OnDemand Server are available.

Real-time loading requires that the ARSLOAD program be configured as documented in step 1.d of “Push method and additional requirements” on page 92. The OnDemand ARSLOAD program is invoked at system startup and must be running for the loads on the OnDemand Server to occur as soon as the jobs complete.

Under normal operations, a PUSH-CPU will launch a IBM Tivoli Workload Scheduler job. This job will execute the modified jobmanrc script that will initiate scripts that start the load process when the job completes. The scripts will determine if the OnDemand Server is available; if the server is available, the job’s stdlist will be loaded onto the OnDemand Server.

If the network is down or the OnDemand Server is unavailable, a link to the original job stdlist called os_type.stdlist.yyyymmdd.Opid_num.hhmm.log will be created in the TWSHome/stdlist/ondemand directory. These files will be loaded to the OnDemand Server by the ARSLOAD program that is running in the background when the OnDemand Server is available.

The Push method is the most efficient method for loading data, since the pushing of the jobs stdlists is spread over the entire day.

Push method and batch mode operations
The other option for loading data to the OnDemand Server is the batch mode. When the jobs complete, a link to the original job stdlist called os_type.stdlist.yyyymmdd.Opid_num.hhmm.log will be created in the TWSHome/stdlist/ondemand directory.

Note: The solution provided for the Push method was designed for UNIX. Because of the many possible options for shell or programming languages that can be used to create and run scripts on Windows systems, this solution did not include converting the UNIX scripts to a format that would work on Windows. The UNIX scripts may be converted for Windows, in which case the Windows systems (Master Domain Manager, Backup Master, Domain Manager, and Fault Tolerant Agent) will then be able to function as PUSH-CPU. If the scripts are converted, please see the notes in the scripts that apply to PUSH-CPUs and replace the logic where for variable OS=win-p.
The `os_type` prefix will be used by the ARSLOAD program to determine what application will be used to load data to the OnDemand Server: `linux` for Linux or `unix` for AIX, HP-UX or Sun Solaris, and `win` for Windows.

In addition to job stdlists, all PUSH-CPU's will need to load their IBM Tivoli Workload Scheduler logs (Netman, TWSmerge, Audit Database and Plan logs) on to the OnDemand Server. This is accomplished by creating a job called `tws_ondemand_logpush.sh`, whose main function is to create ARD file for each IBM Tivoli Workload Scheduler log from the previous production day, which is loaded on the OnDemand Server by the ARSLOAD program.

**Note:** Because of the constant updates to these files, it is better to schedule the loading of the previous days logs. Since this job always loads the previous days logs, availability of this information will not reflect the current status.

**Push method and additional requirements**

The following is a list of requirements for the Push method. Perform all of them in each PUSH-CPU:

1. OnDemand: Push-CPU
   a. The implementation of the Push method requires the OnDemand Client be installed on each system that will be pushing the data. This means that you must touch each UNIX CPU to install the OnDemand Client software.
   
   **Note:** For assistance, refer to the *IBM Content Manager OnDemand for Multiplatforms Installation and Configuration Guide for UNIX Servers Version 7.1*, GC27-0834.

   b. Define an OnDemand user ID and password with Administrator authority and ADD access only. Refer to 7.2, “User and group administration” on page 171 for more information.

c. Configure the `arsload.cfg` file.
   i. Log on as the root user.
   ii. Change to the `OnDemandhome/config` directory.
   iii. Make a copy of the `arsload.cfg` file that was supplied by IBM by running the following command:

   ```bash
   cp arsload.cfg arsload.cfg.orig
   ```

   iv. Make changes to the `arsload.cfg` file with a standard text editor, such as `vi`. 
v. Change the user ID and password. The user and password must be
the same as the one mentioned in 4.2.6, “Download, unzip, and import
the OnDemand definitions” on page 84.

vi. Save your changes and exit the editor.

vii. Set the user file permissions so that only root can read and write and
all the users can only read the arsload.cfg. The permissions should
appear as follows:

```
-rw-r--r-- 1 root system 643 Aug 07 16:00 arsload.cfg
```

**Note:** For details about arsload.cfg file, please refer to Chapter 16,
“Automating data loading” in *IBM Content Manager OnDemand for
Multiplatforms Installation and Configuration Guide for UNIX Servers
Version 7.1*, GC27-0834.

d. Configure the INIT file to automatically start the ARSLOAD program during
operating system initialization. If batch mode is being used, please skip
this step.

**Note:** For OnDemand and TWS integration, you have to configure the
INIT file as follows:

```
ars6:2:once:OnDemandhome/bin/arsload -fv -h <OnDemand_Server_hostname>
-t 300 -A MVS -G JOBNAME -U OnDemandhome/config/arsload.cfg -d
TWShome/stdlist/ondemand.
```

For details about automating the ARSLOAD program, please refer to
Chapter 16, “Automating data loading”, of the *IBM Content Manager
OnDemand for Multiplatforms Installation and Configuration Guide for
UNIX Servers Version 7.1*, GC27-0834.

e. Create `OnDemandhome/logs`, `OnDemandhome/logs/ondemand`, and the
`OnDemandhome/scripts` directory:

```
# mkdir OnDemandhome/logs
# mkdir OnDemandhome/logs/ondemand
# mkdir OnDemandhome/scripts
```

f. Set the permissions, owner, and groups to allow the tws user to access
the directories:

```
# chown tws.tivoli OnDemandhome/logs
# chown tws.tivoli OnDemandhome/logs/ondemand
# chown tws.tivoli OnDemandhome/scripts
# chmod 777 OnDemandhome/logs
# chmod 777 OnDemandhome/logs/ondemand
# chmod 755 OnDemandhome/scripts
```
The permissions should appear as in Example 4-1.

Example 4-1  The permissions after executing the chown commands

<table>
<thead>
<tr>
<th>Permissions</th>
<th>Owner</th>
<th>Group</th>
<th>Size</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>drwxrwxrwx</td>
<td>2 tws</td>
<td>tivoli</td>
<td>512</td>
<td>Aug 07 16:00</td>
</tr>
<tr>
<td>drwxrwxrwx</td>
<td>2 tws</td>
<td>tivoli</td>
<td>512</td>
<td>Aug 07 16:00</td>
</tr>
<tr>
<td>drwxr-xr-x</td>
<td>2 tws</td>
<td>tivoli</td>
<td>512</td>
<td>Aug 07 16:00</td>
</tr>
</tbody>
</table>

The permissions should appear as in Example 4-2.

Example 4-2  The permissions after executing the chown commands

- rwxr-xr-x 1 tws tivoli 2185 Aug 07 16:00 tws_ondemand_env.sh
- rwxr-xr-x 1 tws tivoli 3163 Aug 07 16:00 tws_ondemand_logpush.sh
- rwxr-xr-x 1 tws tivoli 2150 Aug 07 16:00 tws_ondemand_real.sh

2. IBM Tivoli Workload Scheduler: PUSH-CPU

   a. Ensure that IBM Tivoli Workload Scheduler Version 8.2 is installed, configured, and working properly.

   b. Insert the lines shown in Example 4-3 prior to the last line, exit $UNISON_RETURN, in the TWS_home/jobmanrc file.

Example 4-3  Lines that are inserted to the TWS_home/jobmanrc file

```
TWS_MAIN_OUTPUT="$MAEHOME/scripts/logs/tws_ondemand_main.out"
export TWS_MAIN_OUTPUT

trap '([sleep 10; nohup $MAEHOME/scripts/tws_ondemand_main.sh $UNISON_STDLIST $UNISON_JOBNUM $UNISON_SCHED_DATE >$TWS_MAIN_OUTPUT 2>&1 & ] &) &' 0
```

The permissions should appear as in Example 4-2.
c. Change the merge stdlists parameter to yes in the TWShome/localopts file, as shown in Example 4-4.

Example 4-4  Part of the localopts file

```plaintext
# TWS localopts file defines attributes of this Workstation.
#
#------------------------------------------------------------
# Attributes of this Workstation:
#
merge stdlists = yes
```

d. Create the TWShome/scripts, TWShome/scripts/logs and TWShome/stdlist/ondemand directories:

```plaintext
# mkdir TWShome/scripts
# mkdir TWShome/scripts/logs
# mkdir TWShome/stdlist/ondemand
```

e. Set the permissions, owner, and group to allow the tws user access the directories:

```plaintext
# chown tws.tivoli TWShome/scripts
# chown tws.tivoli TWShome/scripts/logs
# chown tws.tivoli TWShome/stdlist/ondemand
# chmod 755 TWShome/scripts
# chmod 777 TWShome/scripts/logs
# chmod 777 TWShome/stdlist/ondemand
```

The permissions should appear as in Example 4-5.

Example 4-5  The permissions after executing the chown commands

```
-rwxr-xr-x 2 tws tivoli 512 Aug 07 16:00 scripts
... 
-rwxrwxrwx 2 tws tivoli 512 Aug 07 16:00 logs
... 
-rwxrwxrwx 2 tws tivoli 512 Aug 07 16:00 ondemand
```

f. Copy the tws_ondemand_main.sh script to the TWShome/scripts directory:

```plaintext
# cp directory/tws_ondemand_main.sh TWShome/scripts
```

g. Set the permissions, owner, and group name to allow the tws user to access the directory:

```plaintext
# chown tws.tivoli TWShome/scripts/tws_ondemand_main.sh
# chmod 755 TWShome/scripts/tws_ondemand_main.sh
```

The permissions should appear as follows:

```
-rwxr-xr-x 1 tws tivoli 736 Aug 07 16:00 tws_ondemand_main.sh
```
h. Create the ODLOGPUSH job, which executes
   OnDemandhome/scripts/tws_ondemand_logpush.sh, as shown in
   Example 4-6.

Example 4-6  ODLOGPUSH job

<PUSH-CPUname>#ODLOGPUSH
   SCRIPTNAME "OnDemandhome/scripts/tws_ondemand_logpush.sh"
   STREAMLOGON "tws"
   DESCRIPTION "Generate and Push all TWS Logs to OnDemand Server"
   RECOVERY STOP

i. Create the ODPUSH job stream, which launches the ODLOGPUSH job,
   as shown in Example 4-7.

Example 4-7  ODPUSH job stream

SCHEDULE <PUSH-CPUname>#ODPUSH
   ON EVERYDAY
   *
   * This schedule must exist on all PUSH-CPUs
   *
   * It will be launch every day as soon as Jnextday completes
   *
   * This will launch a job that extracts the TWSmerge log, Netman log,
   * Audit Database and Plan logs (if they exist) and loads them to the
   * OnDemand Server
   :
   ODLOGPUSH
   END

4.3.3 Pull method

The Pull method discussed in this section requires that a PULL-CPU have Tivoli
Workload Scheduler Version 8.2 installed and working properly and be at least
an UNIX FTA. It will pull all job stdlists and IBM Tivoli Workload Scheduler logs
from their MANAGED-CPUs. These MANAGED-CPUs consist of Extended
Agents, including Oracle, PeopleSoft and SAP, Linux FTAs, Linux Extended
Agents, Linux Standard Agents, Windows Master and Backup Master Domain
Managers, Windows Domain Managers, Windows FTAs, and Windows Standard
Agents. The pulling of the data is accomplished by using IBM Tivoli Workload
Scheduler conman commands and Event Monitoring.
The PULL-CPU will contain a process called tws_ondemand_event.sh, which monitors a TWShome/event.log file. The location of this file is defined in the TWShome/BmEvents.conf file. The event.log contains a summary of all activity on a IBM Tivoli Workload Scheduler environment.

When jobs complete, either with a state of SUCC or ABEND, the monitoring process will invoke a procedure that extracts individual job stdlists from all MANAGED-CPUs and redirects their output to files that are stored in the TWShome/stdlist/ondemand directory. These files will be loaded shortly after their creation to the OnDemand Server throughout the production day.

Special IBM Tivoli Workload Scheduler jobs will run on the MANAGED-CPUs, which will echo the contents of TWSmerge log, Netman log, Audit Database, and Plan logs (if they exist) as standard output into the job stdlist. Since these jobs are being monitored by tws_ondemand_event.sh on a PULL-CPU, the monitoring process will be able to determine if these jobs contain IBM Tivoli Workload Scheduler logs and will handle them accordingly.

For more details about the cpulist file, see “EVENT MONITOR variables” on page 83.

**Pull method and real-time operations**

As stated earlier, the PULL-CPU will pull all job stdlists and IBM Tivoli Workload Scheduler logs for the CPUs that it manages.

The MANAGED-CPUs are determined by the cpulist file, which contains a list of CPUs that will be managed or non-managed by a PULL-CPU, based on the value in the BMEVENT_TYPE variable. For more details about the cpulist file and BMEVENT_TYPE variable, see “EVENT MONITOR variables” on page 83.

The cpulist file is used in conjunction with the Event Monitoring (see step d on page 102) which receives job information for all jobs that run on the IBM Tivoli Workload Scheduler environment. When jobs complete on a MANAGED-CPU, the PULL-CPU extracts the job stdlist and stores it in the TWShome/stdlist/ondemand directory. The naming convention used will be os_type.stdlist.yyyymmdd.MANAGED_CPUname.Opid_num.hhmm.ARD.
Where:

**os_type** Either linux, unix, or win

**log_type** stdlist

**MANAGED_CPUname** IBM Tivoli Workload Scheduler CPUname

**Opid_num** O+process number

Files with an extension of .ARD will be loaded to the OnDemand Server throughout the day, provided that the ARSLOAD program is configured as documented in step “d on page 100. The OnDemand ARSLOAD program is invoked at system startup and must be running for the loads on OnDemand Server to occur.

The real-time method will be the recommended option for PULL-CPUs because of the amount of job stdlists that will be retrieved and loaded to the OnDemand Server.

**Important:** PULL-CPUs will not need the modified jobmanrc, as the job stdlist would be loaded twice, once by the jobmanrc and then also by the tws_ondemand_event.sh process.

**Pull method and MANAGED-CPU operations**

Operations for MANAGED-CPUs will be limited to Windows Master Domain Managers, Windows Domain Managers, Windows FTAs, and Extended Agents, including Oracle, PeopleSoft and SAP, and Windows Standard Agents. The PULL-CPU will retrieve job stdlists from all MANAGED-CPUs and IBM Tivoli Workload Scheduler logs from only Windows Master Domain Managers, Windows Domain Managers, Windows FTAs, and Windows Standard Agents. IBM Tivoli Workload Scheduler jobs will be required to retrieve TWSmerge log, Netman log, Audit Database, and Plan logs (see step 3 on page 105 for job specifics).

The purpose of these jobs will be to list the contents of each of these IBM Tivoli Workload Scheduler logs. The jobs will have specific names that will be monitored by the Event Monitoring (tws_ondemand_event.sh) process on the PULL-CPU. This process will then extract the log information and create a file called

`os_type.log_type.yyyymmdd.MANAGED_CPUname.log_type_upper.log.ARD`

that will then be loaded to the OnDemand Server.
Where:

- **os_type**: Either linux, unix, or win
- **log_type**: Either twsmerge, netman, database, or plan
- **MANAGED_CPUname**: IBM Tivoli Workload Scheduler workstation name
- **log_type_upper**: Either TWSMERGE, NETMAN, DATABASE, or PLAN

### Pull method and additional requirements

As explained before, the implementation of the Pull method requires that the PULL-CPU be a UNIX system.

The following is a list of requirements for the Pull method:

**1. OnDemand: PULL-CPU**

a. Implementation of the Pull method requires that the OnDemand Client must be only installed on the PULL-CPU that will be pulling the data from its MANAGED-CPUs.

**Note:** For assistance, refer to the *IBM Content Manager OnDemand for Multiplatforms Installation and Configuration Guide for UNIX Servers Version 7.1, GC27-0834.*

b. Define an OnDemand user ID and password with Administrator authority and ADD access only. Refer to 7.2, “User and group administration” on page 171 for more information.

c. Configure the arsload.cfg file:

   i. Log on as the root user.
   
   ii. Go to the OnDemandhome/config directory.
   
   iii. Make a copy of the ARSLOAD.CFG file that was supplied by IBM:

   ```
   cp arsload.cfg arsload.cfg.orig
   ```
   
   iv. Make changes to the ARSLOAD.CFG file with a standard text editor, such as vi.
   
   v. Change the user ID and password. The user and password must be the same one in 4.2.6, “Download, unzip, and import the OnDemand definitions” on page 84.
   
   vi. Save your changes and exit the editor.
vii. Set the user file permissions so that only root can read and write and all the users can only read the ARSLOAD.CFG. The permissions should appear as follows:

```
-rw-r--r--1 root system 643 Aug 07 16:00 arsload.cfg
```

**Note:** For details about the arsload.cfg file, please refer to Chapter 16, “Automating data loading”, in *IBM Content Manager OnDemand for Multiplatforms Installation and Configuration Guide for UNIX Servers Version 7.1*, GC27-0834.

d. Configure the INIT file to automatically start the ARSLOAD program during operating system initialization.

**Note:** For the integration of OnDemand and IBM Tivoli Workload Scheduler, you have to configure the INIT file as follows:

```
ars6:2:once:OnDemandhome/bin/arsload -fv -h <OnDemand_Server_hostname> -t 300 -A MVS -G JOBNAME -U OnDemandhome/config/arsload.cfg -d TWShome/stdlist/ondemand.
```

For details about automating the ARSLOAD program, please refer to Chapter 16, “Automating data loading”, of the *IBM Content Manager OnDemand for Multiplatforms Installation and Configuration Guide for UNIX Servers Version 7.1*, GC27-0834.

e. Create the OnDemandhome/logs, OnDemandhome/logs/ondemand, and OnDemandhome/scripts directories:

```
# mkdir OnDemandhome/logs
# mkdir OnDemandhome/logs/ondemand
# mkdir OnDemandhome/scripts
```

f. Set the permissions, owner, and groups to allow the tws user to access the directories:

```
# chown tws.tivoli OnDemandhome/logs
# chown tws.tivoli OnDemandhome/logs/ondemand
# chown tws.tivoli OnDemandhome/scripts
# chmod 777 OnDemandhome/logs
# chmod 777 OnDemandhome/logs/ondemand
# chmod 755 OnDemandhome/scripts
```

The permissions should appear as in Example 4-8 on page 101.
g. Copy the `tws_on_demand_convert.sh`, `tws_on_demand_env.sh`, `tws_on_demand_event.sh`, and `tws_on_demand_monitor.sh` scripts to the `OnDemandHome/scripts` directory:

```bash
# cp directory/tws_on_demand_convert.sh OnDemandHome/scripts
# cp directory/tws_on_demand_env.sh OnDemandHome/scripts
# cp directory/tws_on_demand_event.sh OnDemandHome/scripts
# cp directory/tws_on_demand_logpush.sh OnDemandHome/scripts
# cp directory/tws_on_demand_monitor.sh OnDemandHome/scripts
# cp directory/tws_on_demand_real.sh OnDemandHome/scripts
```

h. Set the permissions, owner, and groups to allow the `tws` user to execute them:

```bash
# chown tws.tivoli OnDemandHome/scripts/tws_on_demand_convert.sh
# chown tws.tivoli OnDemandHome/scripts/tws_on_demand_env.sh
# chown tws.tivoli OnDemandHome/scripts/tws_on_demand_event.sh
# chown tws.tivoli OnDemandHome/scripts/tws_on_demand_monitor.sh
# chmod 755 OnDemandHome/scripts/tws_on_demand_convert.sh
# chmod 755 OnDemandHome/scripts/tws_on_demand_env.sh
# chmod 755 OnDemandHome/scripts/tws_on_demand_event.sh
# chmod 755 OnDemandHome/scripts/tws_on_demand_monitor.sh
```

The permissions should appear as in Example 4-9.

---

2. IBM Tivoli Workload Scheduler: PULL-CPU

   a. Ensure that IBM Tivoli Workload Scheduler Version 8.2 is installed, configured, and working properly.

   b. Change the merge stdlists parameter to yes in `TWSHome/localopts`, as in Example 4-10 on page 102.
Example 4-10  Part of the localopts file

# TWS localopts file defines attributes of this Workstation.
#
#------------------------------------------------------------
# Attributes of this Workstation:
#
merge stdlists                          =yes


c. Define the FULLSTATUS and RESOLVEDEP parameters as ON in the PULL-CPU workstation definitions, as in Example 4-11.

Example 4-11  PULL-CPU workstation definitions

CPUNAME <PULL-CPU name>
   DESCRIPTION "Fault-Tolerant Agent"
   OS  UNIX
   NODE <hostname>
   DOMAIN MASTERDM
   TCPADDR <port number>
      FOR MAESTRO
      AUTOLINK ON
      RESOLVEDEP ON
      FULLSTATUS ON

Note: Use of the Pull method requires that the PULL-CPU be a UNIX FTA, with the FULLSTATUS and RESOLVEDEP options in its CPU definition, in order to pull the logs from its MANAGED-CPUs. This is necessary because PULL-CPU must receive all information for IBM Tivoli Workload Scheduler jobs.

d. Event Monitoring

If Event Monitoring is already being performed, then you will need to include additional monitored events for jobs with a state for “SUCC”. This is necessary, since one of the default monitored events is for jobs that ABEND.

Event Monitoring uses the TWShome/BmEvents.conf file to determine which events get monitored and what log gets generated with that information. Since TWShome/BmEvents.conf identifies the log file, the tws_ondemand_env.sh script, which is executed by most of the scripts, will be able to extract the log. The default name for this file is TWShome/event.log.

If Event Monitoring is not being performed, or it uses a pipe to monitor events instead of a log file, it will be necessary to configure the
TWShome/BmEvents.conf file. There will be changes to the OPTIONS, LOGGING, EVENTS, and FILE options sections in the BmEvents.conf file:

i. Copy the BmEvents.conf file from the TWShome/OV directory to the TWShome directory:

```
# cp TWShome/OV/BmEvents.conf TWShome
```

ii. Change the parameters shown in Example 4-12 in the TWShome/BmEvents.conf file.

Example 4-12  BmEvents.conf file

```
# @(#) $Header:/usr/local/SRC_CLEAR/maestro/JSS/maestro/Netview/RCS/BmEvents.cnf,v 1.6 1996/12/16 18:19:50 ee viola_thunder $
# This file contains the configuration information for the BmEvents module.
# The lines in this file can contain the following:

OPTIONS=MASTER
# MASTER This tells batchman to act as the master of the network and 
# information on all cpus are returned by this module.
# OFF This tells batchman to not report any events.
# default on the master cpu is to report all job scheduling events
# for all cpus on the Maestro network (MASTER); default on other cpus
# is to report events for this cpu only.

LOGGING=ALL
# ALL This tells batchman all the valid event numbers are reported.
# KEY This tells batchman the key-flag filter is enabled
# default is ALL for all the cpus
# #
#

EVENTS=51 101 102 104 105 111 151 152 155 201 202 203 204 251 252 301
# <n> is a valid event number (see Maestro.mib traps for the valid
# event numbers and the contents of each event.
# default is 51 101 102 105 111 151 152 155 201 202 203 204 251 252 301

# To communicate with OperationsCenter using the demonstration log
# file encapsulation the following is required:
FILE=TWShome/event.log
```
e. Create the IBM Tivoli Workload Scheduler job ODMONITOR, which runs
OnDemandhome/scripts/tws_ondemand_monitor.sh (see Example 4-13).

**Example 4-13  ODMONITOR job**

```plaintext
<PULL-CPUname>#ODMONITOR
  SCRIPTNAME "OnDemandhome/scripts/tws_ondemand_monitor.sh"
  STREAMLOGON "tws"
  DESCRIPTION "Monitor tws_ondemand_event.sh script"
  RECOVERY STOP
```

f. Create the IBM Tivoli Workload Scheduler job stream ODPULL, which
runs the ODMONITOR job (see Example 4-14).

**Example 4-14  ODPULL job stream**

```plaintext
SCHEDULE <PULL-CPUname>#ODPULL
ON EVERYDAY
*
  * This schedule must exist on PULL-CPU which will be pulling data
  * from its MANAGED-CPUs
  *
  * The ODMONITOR job will monitor tws_ondemand_event.sh script which
  * read each line of event.log file specified in the BmEvents.conf
  * and pull any job stdlist for MANAGED-CPUs

ODMONITOR EVERY 10
END
```
3. IBM Tivoli Workload Scheduler: UNIX MANAGED-CPUs
   a. Ensure that IBM Tivoli Workload Scheduler Version 8.2 is installed, configured, and working properly.
   b. Create the following IBM Tivoli Workload Scheduler jobs:
      i. ODNETMAN, which runs
         TWShome/scripts/tws_ondemand_netman.sh (see Example 4-16)
      ii. ODTWSMERGE, which runs
         TWShome/scripts/tws_ondemand_twsmerge.sh (see Example 4-17)
      iii. ODDATABASE, which runs
         TWShome/scripts/tws_ondemand_database.sh (see Example 4-18)
      iv. ODPLAN, which runs TWShome/scripts/tws_ondemand_plan.sh (see Example 4-19 on page 105)

Example 4-16 ODNETMAN job
<MANAGED-CPUname>#ODNETMAN
   SCRIPTNAME "TWShome/scripts/tws_ondemand_netman.sh"
   STREAMLOGON "tws"
   DESCRIPTION "Netman log"
   RECOVERY STOP

Example 4-17 ODTWSMERGE job
<MANAGED-CPUname>#ODTWSMERGE
   SCRIPTNAME "TWShome/scripts/tws_ondemand_twsmerge.sh"
   STREAMLOGON "tws"
   DESCRIPTION "TWSmerge log"
   RECOVERY STOP

Example 4-18 ODDATABASE job
<MANAGED-CPUname>#ODDATABASE
   SCRIPTNAME "TWShome/scripts/tws_ondemand_database.sh"
   STREAMLOGON "tws"
   DESCRIPTION "Audit Database log"
   RECOVERY STOP

Example 4-19 ODPLAN job
c. Create the IBM Tivoli Workload Scheduler job stream ONDEMAND, which launches ODNETMAN, ODTWSMERGE, ODDATABASE, and ODPLAN (see Example 4-20).

Example 4-20 ONDEMAND job stream

SCHEDULE <MANAGED-CPUname>#ONDEMAND
ON EVERYDAY

* This schedule must exist on all MANAGED-CPUs where data must be
* pulled from
:
* The job names may not be changed because the script
* tws_ondemand_event.sh which runs on the PULL-CPU is looking
* for these job names
*
* The following jobs will do a cat of the respective log. It will
* have a header and trailer text to denote the start and stop
* of the log

ODNETMAN
ODTWSMERGE
ODDATABASE
ODPLAN

END

d. Create the TWSHome/scripts, TWSHome/scripts/logs, and
TWSHome/stdlist/ondemand directories:

# mkdir TWSHome/scripts
# mkdir TWSHome/scripts/logs
# mkdir TWSHome/stdlist/ondemand

e. Set the permissions, owner, and groups to allow the tws user to access
the directories.

# chown tws.tivoli TWSHome/scripts
# chown tws.tivoli TWSHome/scripts/logs
# chown tws.tivoli TWSHome/stdlist/ondemand
# chmod 755 TWSHome/scripts
# chmod 777 TWSHome/scripts/logs
# chmod 755 TWSHome/stdlist/ondemand

The permissions should appear as in Example 4-21.
Example 4-21  The permissions after executing the chown commands

<table>
<thead>
<tr>
<th>Mode</th>
<th>Owner</th>
<th>Group</th>
<th>Size</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>drwxr-xr-x</td>
<td>tws</td>
<td>tivoli</td>
<td>512</td>
<td>Aug 07 16:00</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxrwx</td>
<td>tws</td>
<td>tivoli</td>
<td>512</td>
<td>Aug 07 16:00</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxr-xr-x</td>
<td>tws</td>
<td>tivoli</td>
<td>512</td>
<td>Aug 07 15:59</td>
</tr>
</tbody>
</table>

f. Copy the tws_ondemand_netman.sh, tws_ondemand_twsmerge.sh, tws_ondemand_database.sh, and tws_ondemand_plan.sh scripts to the TWS/home/scripts directory:

```
# cp directory/tws_ondemand_netman.sh TWS/home/scripts
# cp directory/tws_ondemand_twsmerge.sh TWS/home/scripts
# cp directory/tws_ondemand_database.sh TWS/home/scripts
# cp directory/tws_ondemand_plan.sh TWS/home/scripts
```

g. Set the permissions, owner, and groups to allow the tws user to access the directories:

```
# chown tws.tivoli TWS/home/scripts/tws_ondemand_netman.sh
# chown tws.tivoli TWS/home/scripts/tws_ondemand_twsmerge.sh
# chown tws.tivoli TWS/home/scripts/tws_ondemand_database.sh
# chown tws.tivoli TWS/home/scripts/tws_ondemand_plan.sh
# chmod 755 TWS/home/scripts/tws_ondemand_netman.sh
# chmod 755 TWS/home/scripts/tws_ondemand_twsmerge.sh
# chmod 755 TWS/home/scripts/tws_ondemand_database.sh
# chmod 755 TWS/home/scripts/tws_ondemand_plan.sh
```

The permissions should appear as in Example 4-22.

Example 4-22  The permissions after executing the chown commands

<table>
<thead>
<tr>
<th>Mode</th>
<th>Owner</th>
<th>Group</th>
<th>Size</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>drwxr-xr-x</td>
<td>tws</td>
<td>tivoli</td>
<td>495</td>
<td>Aug 07 16:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxr-xr-x</td>
<td>tws</td>
<td>tivoli</td>
<td>509</td>
<td>Aug 07 16:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxr-xr-x</td>
<td>tws</td>
<td>tivoli</td>
<td>515</td>
<td>Aug 07 16:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxr-xr-x</td>
<td>tws</td>
<td>tivoli</td>
<td>479</td>
<td>Aug 07 16:00</td>
</tr>
</tbody>
</table>

4. IBM Tivoli Workload Scheduler: Windows MANAGED-CPUs

a. IBM Tivoli Workload Scheduler Version 8.2 is installed, configured, and working properly.

b. Create the following IBM Tivoli Workload Scheduler jobs:

i. ODNETMAN, which runs 
   TWS/home\scripts\tws_ondemand_netman.cmd (see Example 4-23 on page 108)

ii. ODTWSMERGE, which runs 
    TWS/home\scripts\tws_ondemand_twsmerge.cmd (see Example 4-24)
iii. ODDATABASE, which runs
   TWShome\scripts\tws_ondemand_database.cmd (see Example 4-25)
iv. ODPLAN, which runs TWShome/scripts/tws_ondemand_plan.cmd
    (see Example 4-26)

Example 4-23  ODNETMAN job

<MANAGED-CPUname>#ODNETMAN
   SCRIPTNAME "<drive>:\TWShome\scripts\tws_ondemand_netman.cmd"
   STREAMLOGON "tws"
   DESCRIPTION "Netman log"
   RECOVERY STOP

Example 4-24  ODMERGE job

<MANAGED-CPUname>#ODTWSMERGE
   SCRIPTNAME "<drive>:\TWShome\scripts\tws_ondemand_twsmerge.cmd"
   STREAMLOGON "tws"
   DESCRIPTION "TWSmerge log"
   RECOVERY STOP

Example 4-25  ODDATABASE job

<MANAGED-CPUname>#ODDATABASE
   SCRIPTNAME "<drive>:\TWShome\scripts\tws_ondemand_database.cmd"
   STREAMLOGON "tws"
   DESCRIPTION "Audit Database log"
   RECOVERY STOP

Example 4-26  ODPLAN job

<MANAGED-CPUname>#ODPLAN SCRIPTNAME
   "TWShome\scripts\tws_ondemand_plan.cmd"
   STREAMLOGON "twsuser"
   DESCRIPTION "Audit Plan log"

C. Create the IBM Tivoli Workload Scheduler job stream ONDEMAND, which
   runs ODDATABASE, ODMERGE, ODNETMAN, and ODPLAN (see
   Example 4-27 on page 109).
Example 4-27  ONDEMAND job stream

SCHEDULE <MANAGED-CPUname>\#ONDEMAND
ON EVERYDAY

* This schedule must exist on all MANAGED-CPUs where data must be
* pulled from:
* The job names may not be changed because the script
* tws_ondemand_event.sh which runs on the PULL-CPU is looking
* for these job names
*
* The following jobs will do a cat of the respective log. It will
* have a header and trailer text to denote the start and stop
* of the log

ODMERGE
ODNETMAN
OPPLAN
ODDATABASE

END

d. Create the TWShome\scripts, TWShome\scripts\logs, and
   TWShome\stdlist\ondemand directories:
   
   # mkdir <drive>:\TWShome\scripts
   # mkdir <drive>:\TWShome\scripts\logs
   # mkdir <drive>:\TWShome\stdlist\ondemand

e. Copy the tws_ondemand_netman.cmd, tws_ondemand_twsmerge.cmd,
   tws_ondemand_database.cmd, and tws_ondemand_plan.cmd scripts to
   the TWShome\scripts directory:
   
   # cp directory\tws_ondemand_netman.sh <drive>:\TWShome\scripts
   # cp directory\tws_ondemand_twsmerge.sh <drive>:\TWShome\scripts
   # cp directory\tws_ondemand_database.sh <drive>:\TWShome\scripts
   # cp directory\tws_ondemand_plan.sh <drive>:\TWShome\scripts

4.4 Scenarios

Candidates for PUSH-CPUs will be any UNIX Master, UNIX Domain Manager,
UNIX Backup Master, UNIX Domain Manager, UNIX FTA, or UNIX Standard
Agent. A Windows FTA can also be a PUSH-CPU if the UNIX scripts and logic
have been converted to Windows. As stated before, PUSH-CPUs require
installation of the OnDemand Client.
Since all candidates for PUSH-CPUs are UNIX, a scenario for their representation will consist of each PUSH-CPU pushing data to the OnDemand Server, as in Figure 4-3.

![Figure 4-3](image)

Since PULL-CPUs manage MANAGED-CPUs, one scenario that PULL-CPUs can be used in is to dedicate certain PULL-CPUs for managing Windows FTAs only.

Another scenario would be to have the UNIX Domain Manager managing the Windows FTAs, Standard Agents, and Extended Agents under its domain. This will permit the grouping of MANAGED-CPUs by domain.

The ideal situation would be a combination of both of the methods: Push and Pull, as depicted in Figure 4-4 on page 111. All UNIX IBM Tivoli Workload Scheduler Agents could be set up to use the Push method. The number of MANAGED-CPUs would determine the number of PULL-CPUs required.
4.5 Troubleshooting

If the loads are failing, you might check the following:

- Is the OnDemand server host name specified for the ARSLOAD_SERVER variable in the tws_ondemand_env.sh script? Correct it if necessary.

- Is there disk space available to ARD files?

- Directory permissions must be valid for the IBM Tivoli Workload Scheduler user (in our scenario, tws).

- Is the OnDemand Server available?
  Check if the OnDemand Server is up and running by issuing `telnet OD_server_IP port_number` such as:
  
telnet <OD_server_IP> 1445
- Check the log files in the OnDemandhome/logs/ondemand and TWShome/scripts/logs directories.
IBM Tivoli Workload Scheduler end-to-end environment

This chapter discusses, in detail, the steps required, over and above those already described in the earlier chapters, to implement the integration between IBM Tivoli Workload Scheduler for z/OS and OnDemand for z/OS in an end-to-end environment.

The end-to-end environment implies you are going to run all your production workloads, z/OS and distributed, from the IBM Tivoli Workload Scheduler for z/OS scheduling engine, which will utilize the Fault Tolerant Agent technology, and that you will have one central OnDemand DB2 database for storing the appropriate output.

In this chapter, the following topics are covered:

- Section 5.1, “Overview” on page 114
- Section 5.2, “Architecture” on page 114
- Section 5.3, “Prerequisites” on page 115
- Section 5.5, “Implementation” on page 116
5.1 Overview

In an end-to-end scenario, the tasks required to store the output being collected in both the z/OS and Distributed environments are exactly the same as those already defined in the specific chapters for each environment with one exception. In an end-to-end environment, our recommendation to have the OnDemand DB2 database residing on z/OS means that the address of the server you define in the distributed implementation instructions should now point to the server name and address of the z/OS OnDemand Server machine.

5.2 Architecture

Figure 5-1 on page 115 indicates the architecture used for an OnDemand and IBM Tivoli Workload Scheduler for z/OS installation within two or more domains in a true end-to-end configuration. Detailed explanations for both the Domain 1 and Domain 2 environments can be found within Chapter 3, “IBM Tivoli Workload Scheduler for z/OS implementation” on page 31. Details on how the Distributed environment is set up can be found in Chapter 4, “IBM Tivoli Workload Scheduler Distributed implementation” on page 77.
Chapter 5. IBM Tivoli Workload Scheduler end-to-end environment

5.3 Prerequisites

The prerequisites are defined in Chapter 4, “IBM Tivoli Workload Scheduler Distributed implementation” on page 77 and Chapter 3, “IBM Tivoli Workload Scheduler for z/OS implementation” on page 31.

5.4 Data collected

The data collected is the same as that defined in Chapter 4, “IBM Tivoli Workload Scheduler Distributed implementation” on page 77 and Chapter 3, “IBM Tivoli Workload Scheduler for z/OS implementation” on page 31.
5.5 Implementation

The implementation is the same as that described in Chapter 4, “IBM Tivoli Workload Scheduler Distributed implementation” on page 77 and Chapter 3, “IBM Tivoli Workload Scheduler for z/OS implementation” on page 31 with the following difference.

The OnDemand Server pointed to when installing the IBM Tivoli Workload Scheduler for Distributed OnDemand integration should now point to the z/OS OnDemand Server residing in the UNIX System Services (USS) portion of the z/OS machine.
Reporting: IBM Content Manager user interface

This chapter describes the graphical user interface (GUI) that guides users with point-and-click functionality, which makes Content Manager OnDemand easy to navigate by end users and easy to administer by your IT staff.

The following topics are covered in this chapter:

- Section 6.1, “IBM Content Manager OnDemand user interfaces” on page 118
- Section 6.2, “Using the OnDemand client program” on page 119
6.1 IBM Content Manager OnDemand user interfaces

IBM Content Manager OnDemand has two user interfaces that are used to search for and retrieve reports stored on OnDemand systems.

OnDemand client program
These programs operate on personal computers running on Windows. Using the client program, users can search for and retrieve reports stored on the system. Specifically, users can construct queries and search for reports, retrieve documents from OnDemand, view, print, and fax copies or pages of documents, and attach electronic notes to pages of a document.

OnDemand client programs and servers communicate over a computer network supported by TCP/IP. When a user submits a query, the client program sends a search request to the OnDemand library server. The library server returns a list of the documents that match the query to the user. When the user selects a document for viewing, the client program retrieves a copy of the document from the object server where the document is stored, opens a viewing window, and displays the document.

Note: The OnDemand client program was designed for use on PC-compatible platforms only.

OnDemand Web Enablement Kit
The OnDemand Web Enablement Kit (ODWEK) is an optional feature of OnDemand that allows people to use a Web browser to access data stored in an OnDemand system. For example, you can provide some people with the Uniform Resource Locator (URL) of a Web page that permits them to log on to an OnDemand Server; you can provide other people with the URL of a Web page that permits them to search a specific folder. ODWEK verifies that the user information is valid on the OnDemand Server, such as permission to access the server and data stored in an application group. After the user submits a search, ODWEK displays a Web page that contains a list of the documents that match the query. The user selects a document to view and ODWEK sends the document to the browser.

For information on installing and customizing ODWEK, refer to IBM Content Manager OnDemand for Multiplatforms Web Enablement Kit Implementation Guide Version 7.1, SC27-1000.

In this redbook, we used the OnDemand client program as a user interface in our scenarios, but when you implement the solution in your environment, you can use the OnDemand Web Enablement Kit instead. You can also use a
combination of both user interfaces (OnDemand client program for some users and OnDemand Web Enablement Kit for others). Note that the look and feel of both user interfaces is very similar, so if you become experienced with one of these programs, you can easily use the other with no difficulty.

### 6.2 Using the OnDemand client program

In the following sections, we will show how you can use the OnDemand client program. We installed our OnDemand client program on a Windows 2000 workstation.

#### 6.2.1 Starting the IBM OnDemand Interface (GUI)

In order to start the OnDemand client program from the Start menu do the following:

1. Select the Start button on the Taskbar and select Programs → OnDemand 32 → OnDemand32 English, as shown in Figure 6-1.

![Figure 6-1   Starting the OnDemand client program](image)

2. The Logon to a Server dialog box appears in the foreground of the OnDemand work space, as in Figure 6-2 on page 120.
Figure 6-2  Logging on to the OnDemand Server

OnDemand needs to know certain information to begin communicating with a server, and it collects this information in the Logon to a Server dialog box.

**Note:** You can also start the OnDemand program from the Run dialog by selecting **Start → Run** from your Windows workstation and entering \OnDemand Path\arsgui32.exe, where OnDemand Path is the path to your OnDemand client.

3. If your OnDemand administrator has set up default servers, the names of the default servers are listed in the pull-down list in the Server entry field. To access the pull-down list, choose the arrow to the right of the Server entry field and select a name from the list. If you cannot find the server listed, press the **Update Servers...** button and the Update Servers dialog box appears. See 6.2.2, “Updating server information” on page 121 for more information on adding a server to the list in the Logon to a Server dialog box, or changing information for a server that is listed.

4. Type your OnDemand user ID and password in the spaces provided. Press the Tab key to move from one field to another or use the mouse to select them.

5. Choose **OK** or press Enter to log on to the specified server. OnDemand will verify the user ID and password on the Server.

**Note:** The logon command is only available when you are not currently logged on to a server.
6.2.2 Updating server information

Use the Update Servers option from the Logon to a Server dialog box to add a server to the list of servers you can access, to change information for a server that you can access, or to delete a server from the list of servers that you can access. You can change information for TCP/IP servers and local servers.

Note: The first time you start OnDemand, the Server entry field and the pull-down list box may not contain any entries. Check with your OnDemand administrator for server names and other information you need to enter before you begin.

6.2.3 Opening folders

Your OnDemand administrator stores information in related collections called folders. Think of a folder as a container for related information, such as statements, invoices, or correspondence, just like a manila folder is used in a metal filing cabinet. When you open a folder, you have access to the information contained in that folder. For example, a billing folder could contain the reports for customer transactions over the past two years.

OnDemand displays the Open a Folder dialog box only if there is more than one folder from which to choose. Otherwise, OnDemand displays the Search Criteria and Document List dialog box immediately after logging on the server. If you do not have to choose folders, skip the following steps and proceed to searching for documents.

To open a folder on your OnDemand client, follow these steps:

1. Select File → Open Folder. The folders that you can access appear in the Open a Folder dialog box (Figure 6-3 on page 122).
2. To open a folder, double-click a folder name or choose a folder name and then choose **Open**. The Search Criteria and Document List dialog box appears, as in Figure 6-4 on page 123.

---

**Notes:**

- Recall from 1.3, “OnDemand concepts” on page 6 that folders allow an administrator to set up a common query screen for several application groups that may use different indexing schemes, so that a user can retrieve the data with a single query.

- If you do not see a folder you want to select in the Open a Folder window, the reason might be that it is a secondary folder. In that case, use the **Show Folders** option to list all folders, including secondary folders. When you log on to a server, the client lists only the primary folders. To list all folders, select **All**. If there are no secondary folders defined on the server, then the client does not display the Show Folders option. An administrator determines which folders are secondary folders.

- You can have more than one folder open at a time. When you are working with multiple folders, OnDemand lists the folders you have open in the Window menu.
3. From the Search Criteria and Document List dialog, specify the search criteria for the documents you want to view. This task is called *searching for documents*. OnDemand lists the documents that match your search criteria. You then use this same dialog box to select specific documents for viewing or printing. This task is called *Selecting and Viewing Documents*.

The following is a description of the options in this dialog:

- **Search Field Name**: Your OnDemand administrator creates search field names. Each folder can have different search field names. You can type values for any combination of search fields. You may need to type a value in at least one search entry field for OnDemand to search for documents. Check with your OnDemand administrator for any search requirements in your office.

- **Search Operator**: Select a search operator for each search field. OnDemand uses standard search operators to search for documents. Choose the button to the right of the search field name and OnDemand displays the list of
operators valid for that field. Select an operator by choosing one of the buttons or typing the keyboard shortcut.

- **Search Entry Field**: Type numbers and text strings into search entry fields. Use wildcards to expand searches.
- **Logical**: Choose the OR button to connect search fields using the OR logical operator. Choose the AND button to connect search fields using the AND logical operator.
- **Append**: Use to place new search results after those already in the Document List.
- **AutoScroll**: Use to automatically move to the end of the Document List as search results are found. This is useful when you use the Append check box to add search results to those already in the Document List.

**Note**: The AutoScroll check box is not included on the Search Criteria and Document List dialog box when the information in the Document List is generated in sorted order.

The following is a description of the commands in this dialog:

- **Search**: Use to search the OnDemand Server for documents that match the search criteria you specified. First, type text and numbers in one or more search entry fields on the Search Criteria and Document List dialog box, and then choose **Search**.

- **Clear All Fields**: Use to clear all search entry fields. Use this command to erase the values in all the entry fields on the Search Criteria and Document List dialog box with a single click of the mouse.

- **Restore Defaults**: Use to restore the default values (if any) to the search entry fields. Restore the search entry fields to the values that appeared on the Search Criteria and Document List dialog box when you started OnDemand or first selected a folder. Ask your OnDemand administrator about setting default values for search fields.

- **Close Folder**: Use to remove the Search Criteria and Document List dialog box from the display and return to the Open a Folder dialog box. When you are working with a single folder, use Close Folder to close the folder you are using and remove the Search Criteria and Document List dialog box from the display. The Open a Folder dialog box appears, so that you can make another selection.

When you are working with multiple folders, use Close Folder to close the folder you are using, and the Open a Folder dialog box appears over the Search Criteria and Document List dialog box for the previous folder you had open (if you want to open a different folder). Otherwise, choose the Cancel.
button on the Open a Folder dialog box, and you can work with the previous folder.

- **View All Selected:** Use to view the list of documents that you selected. By default, the client displays each selected document in a separate viewing window. If you select View Combined Documents, then the client concatenates the documents and displays them in a single viewing window.

- **Print All Selected:** Use to print the list of documents you selected. OnDemand displays the Print dialog box.

- **Sort List:** Use to change the order of the items in the Document List.

If you want to concatenate the selected documents and display them as a single document in one viewing window, you can use the View Combined Documents option from the Options menu.

All of the documents that you select must contain line data and have the same physical attributes, such as carriage control, record format, and record length. You can use the local print and send functions to print, FAX, or e-mail the concatenated document.

The View Combined Documents command does not work with large objects.

**Note:** Large objects are very large documents that, when loaded on the server, are divided into segments to provide better retrieval performance. As you move to pages in the document, OnDemand automatically retrieves the required segment from the server.

### 6.2.4 Searching for documents

Search for documents using the Search Criteria and Document List dialog box. The name of the folder you selected appears in the title bar of the Search Criteria and Document List dialog box.

**Search field names**

Search field names are the key to finding documents on the OnDemand Server. The names represent pieces of information in a document. OnDemand uses the values you type in search entry fields to match to documents on the server. Your OnDemand administrator creates search field names. Each folder can have different search field names. You can search for documents using any combination of search fields. OnDemand needs a value in at least one entry field to search for documents. Ask your OnDemand administrator about any special requirements for typing search strings for the folders on your system.
Search operators on the Select Operator dialog box

OnDemand uses standard search operators to search for documents. Choose a search operator button (to the right of the search field name) and OnDemand displays the Select Operator dialog box. The list of operators valid for that field appears in the dialog box. Choose an operator button or type the keyboard shortcut for an operator name.

Table 6-1 describes the operators you may use with OnDemand.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal To</td>
<td>Search for the exact value specified.</td>
</tr>
<tr>
<td>Not Equal To</td>
<td>Search for values that do not match the value specified.</td>
</tr>
<tr>
<td>Greater Than</td>
<td>Search for values greater than the value specified.</td>
</tr>
<tr>
<td>Greater Than Or Equal To</td>
<td>Search for values greater than or equal to the value specified.</td>
</tr>
<tr>
<td>Less Than</td>
<td>Search for values less than the value specified.</td>
</tr>
<tr>
<td>Less Than Or Equal To</td>
<td>Search for values less than or equal to the value specified.</td>
</tr>
<tr>
<td>Like</td>
<td>Search for values that match the pattern specified.</td>
</tr>
</tbody>
</table>

When you use the Like search operator, OnDemand inserts the % wildcard after the character or characters that you enter and searches for all values that start with the characters you specified. You can also insert the % wildcard before the characters you specify to search for all values that end with the characters you specified.
Choose **Cancel** to remove the Select Operator dialog box from the display without changing the search operator.

The following are examples of search entry fields:

- Type JONES for Customer Name and select the **Equal To** search operator. OnDemand searches for documents with JONES as the Customer Name value.

- Type 001-001-0094 for Account and select the **Equal To** search operator. OnDemand searches for documents with 001-001-0094 as the Account value.

- Type 1000 for Balance and select the **Greater Than** search operator. OnDemand searches for documents with a Balance greater than 1000.

### Operator Meaning

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Like</td>
<td>Search for values that do not match the pattern specified. When you use the Not Like search operator, OnDemand inserts the % wildcard after the character or characters you enter and searches for all the values that do not start with the characters you specified. You can also insert the % wildcard before the characters you specify to search for all values that do not end with the characters you specified.</td>
</tr>
<tr>
<td>Between</td>
<td>Search for a range of values using a lower and upper value.</td>
</tr>
<tr>
<td>Not Between</td>
<td>Search for a range of values outside a lower and upper value.</td>
</tr>
<tr>
<td>In</td>
<td>Match one or more of the values specified in the set.</td>
</tr>
<tr>
<td>Not In</td>
<td>Match none of the values specified in the set.</td>
</tr>
</tbody>
</table>

**Note:** You can prefix money values with + (plus) and - (minus) characters, for example, +1000 or -1000.

You can type and search for currency values that include two places to the right of the decimal point, for example, 1000.00.
6.2.5 Search with wildcards

Use wildcards with the Like and Not Like search operators to represent any search character. In OnDemand, the % (percentage) and the _ (underscore) are wildcard characters. The % matches any number of characters in a field. For example, san% can find San Francisco and Santa Rosa but not Asante. The _ matches a single character in a field. For example, to_ can find top and tow, but not today.

The _ (underscore) character is a wildcard that matches a single letter, number, or other special character. The % (percentage) character is a wildcard that matches zero or more letters, numbers, or other special characters. Wildcards are valid only in string (character) fields. In the second example in Table 6-2, assume the target search field contains letters and numbers defined as a string field. A string can be composed of letters, numbers, and special characters.

Table 6-2 Search wildcards

<table>
<thead>
<tr>
<th>Wildcard</th>
<th>Usage</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>%</td>
<td>Like MARS%</td>
<td>Find MARS, MARSCH, MARSHELL, MARSTEENS, ...</td>
</tr>
<tr>
<td>_</td>
<td>Like 1_0</td>
<td>Find 100, 120, 130...190</td>
</tr>
</tbody>
</table>

Figure 6-5 on page 129 shows an example of wildcard usage, where we search for job names starting with M82#S. You can see from the pictures that only jobs names starting with M82#S are listed.
6.2.6 Searching with dates and date ranges

OnDemand validates date fields using the system date format. The default date format is m/d/yy (no leading zero for day and month), where m is a month from 1 to 12, d is a day from 1 to 31, and yy is a year from 01 to 99.

Type date strings use MM/DD/YY and MM-DD-YY formats, where MM is a month from 1 to 12, DD is a day from 1 to 31, and YY is a year from 01 to 99. OnDemand requires - (dash) or / (forward slash) characters between MM and DD and DD and YY.

Note: Within OnDemand, the minimum date for the year is 1970.

Check with your OnDemand administrator about any special requirements for searching with date and date ranges.

Figure 6-6 on page 130 shows an example of a Searching for documents operation with a given date and time range.
Figure 6-6  Searching for documents operation: date and time range

Figure 6-7 on page 131 shows an example of a Searching for documents operation only for a certain date.
You also have the option of sorting the documents according to several fields (ascending or descending order) by selecting the **Sort List** button. Figure 6-8 on page 132 shows a sort operation.
**T date search option**

To retrieve documents that contain the current system date, type T in the date search field and set the search operator to **Equal To**.

Type the letter T in the date search field and set the search operator to **Equal To**. When you run the search, OnDemand retrieves the documents that contain the current system date.

You can also use the following patterns when you use the T date search option:

T \{ + or - \} # \{ D or M or Y \}

The braces denote groups of optional parameters for the T format string; choose one of the symbols in the group. If you leave out + or -, OnDemand assumes +; if you leave out D, M, or Y, OnDemand assumes D. The T format string is case insensitive; you can type T or t; D or d.

Table 6-3 on page 133 describes the T format string.
Table 6-3  T format string

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>Current date (required)</td>
</tr>
<tr>
<td>+</td>
<td>Forward from the current date</td>
</tr>
<tr>
<td>-</td>
<td>Backward from the current date</td>
</tr>
<tr>
<td>#</td>
<td>Represents the number of days, months, or years (required)</td>
</tr>
<tr>
<td>D</td>
<td>Days (# days)</td>
</tr>
<tr>
<td>M</td>
<td>Months (# months)</td>
</tr>
<tr>
<td>A</td>
<td>Years (# years)</td>
</tr>
</tbody>
</table>

Table 6-4 describes examples of the T format string.

Table 6-4  Examples of using the T format string

<table>
<thead>
<tr>
<th>T String</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-6M</td>
<td>Six months prior to the current date</td>
</tr>
<tr>
<td>T+30D</td>
<td>Thirty days from the current date</td>
</tr>
<tr>
<td>T30</td>
<td>Thirty days from the current date (same as above)</td>
</tr>
<tr>
<td>T-1Y</td>
<td>One year prior to the current date</td>
</tr>
</tbody>
</table>

To retrieve the most recently loaded documents, type L in the date search field, set the date search operator to Equal To, and enter a value in at least one other search field.

L date search option

Use the L date search option to retrieve the most recently loaded document from a set of documents that contain the same index values.

To use the L date search option, you must:

- Type the letter L in the date search field
- Set the date search operator to Equal To
- Enter a value in at least one other search field

When you run the search, OnDemand lists only the most recently loaded documents that match the search criteria.
6.2.7 Search logical operators

OnDemand connects search fields using the AND/OR logical operators on the Search Criteria and Document List dialog box. The standard search connects fields with AND (that is, the search must satisfy all search fields). The OR logical operator means the search must satisfy at least one of the search criteria.

Choose the AND button to connect search fields using the AND logical operator. Choose the OR button to connect search fields using the OR logical operator.

**Note:** Even when you choose the OR logical operator, the date segment field is always ANDed with the rest of the search fields.

The following are some search examples:

- Searching for documents with a specific date: To retrieve documents with a specific date from the OnDemand Server, type the specific date in the Date entry field and use the Equal To search operator, if it is not already selected. Choose the Search button.

- Searching for a set of documents: To retrieve a set of documents for a specific account, set the Account search operator to Equal To and the Date search operator to Between. Type the account in the Account entry field. Type begin and end dates in the Date entry fields. Choose the Search button.

- Searching using pattern matching: You can use the Like and Not Like search operators to retrieve documents that match (or do not match) a pattern. You must use a search wildcard with these operators. OnDemand uses the following SQL pattern matching wildcards: _ (underscore) to represent any single letter, number, or special character, and % (percentage) to represent zero or more occurrences of any letter, number, or special character.
Searching using negative operators: You can exclude certain documents from the ones OnDemand retrieves. Use the Not Equal, Not In, Not Like, and Not Between search operators to tell OnDemand to retrieve only those documents that do not meet the search criteria.

For example, assume you want OnDemand to retrieve documents where the account is not in 000-000-100 and 000-000-199. First, set the Account search operator to **Not In**. Then, type 000-000-100 and 000-000-199 in the Account entry fields. Date is a required field in the sample folder.

### 6.2.8 Printing selected documents

Select the documents to print from the Document List and use the **Print All Selected** button to send print requests for a list of documents to the server.

OnDemand prints the documents on a printer attached to your PC (your local printer) or a printer attached to the server. If you have questions about printing, please ask your OnDemand administrator.

To print selected documents from the Document List on the Search Criteria and Document List dialog box, follow these steps:

1. Select documents from the Document List using one of the following methods:
   - Choose a single document you would like to print.
   - Place the pointer on the first document you would like to print. Click and hold down the mouse button. Drag the pointer to select a group of documents you would like to print. Release the mouse button.
   - Place the pointer on the first document you would like to print. Click the mouse button. Move the pointer to the last document you would like to print. Press and hold down the Shift key. Click the mouse button. Release the Shift key.
   - Place the pointer on the first document you would like to print. Click the mouse button. Move the pointer to the next document you would like to print. Hold down the Ctrl key and click the mouse button. Release the Ctrl key. Repeat to select additional documents.

2. Choose the **Print All Selected** button. The Print dialog box appears, as in Figure 6-9 on page 136, with the All Pages button selected and the other Pages choices grayed out.
3. Select the **Local** button to send a print request for the selected documents to a printer attached to your PC. Select the **Server** button to send a print request for the selected documents to a printer attached to your server.

**Note:** If you have questions about printers and printer ports, ask your OnDemand administrator.

4. If the printer you want to use is not displayed in the entry field, select a printer from the list.

5. Choose the number of copies you would like to print.

6. Choose the **Print** button to send the documents to the printer, close the Print dialog box, and return to the Search Criteria and Document List dialog box.

### 6.2.9 Selecting and viewing documents

Select documents for viewing using the Document List section of the Search Criteria and Document List dialog box.
The number of documents that you can view concurrently is determined by the availability of system resources and the type of document. Typically, fewer Advanced Function Presentation (AFP) formatted documents can be displayed at once, while a larger number of line data documents can be displayed at once. Experiment to determine how many documents you can display at one time. OnDemand displays each document you select in a separate window.

If you selected the AutoView command on the Options menu, you have the following choices when viewing documents (see Table 6-5).

Table 6-5  Options to view documents when AutoView is selected

<table>
<thead>
<tr>
<th>View document option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>AutoView is not enabled; OnDemand does not automatically display a document found by a search.</td>
</tr>
<tr>
<td>First Document</td>
<td>OnDemand automatically displays the first document from the list of documents that satisfy the search criteria.</td>
</tr>
<tr>
<td>Single Document</td>
<td>When OnDemand finds only one document that satisfies the search criteria, OnDemand automatically displays the document for you.</td>
</tr>
</tbody>
</table>

6.2.10 How to select documents for viewing

To select documents for viewing, follow these steps:

1. Select documents from the Document List on the Search Criteria and Document List dialog box using one of the following methods:
   - Choose a single document you would like to view.
   - Place the pointer on the first document you would like to view. Click the mouse button. Move the pointer to the next document you would like to view. Hold down the Ctrl key and click the mouse button. Release the Ctrl key. Repeat to select additional documents.
   - Place the pointer on the first document you would like to view. Click and hold down the mouse button. Drag the pointer to select a group of documents. Release the mouse button.
   - Place the pointer on the first document you would like to view. Click the mouse button. Move the pointer to the last document you would like to include in the group. Press and hold down the Shift key. Click the mouse button. Release the Shift key.
2. Choose **View All Selected**. By default, the client displays the selected documents, each in a separate window. When you are selecting multiple documents for viewing, OnDemand displays the following message in the first section of the status bar:

```
Document Retrieval is proceeding for document name.
```

If you select **View Combined Documents**, then the client concatenates the documents and displays them in a single viewing window.

Here is an example of selecting and viewing documents for a single document, that is, TWSMERGE log. In Figure 6-10, we search the TWSMERGE logs on eastham for CPU type UNIX and dates range between Aug 14th and Aug 19th.

![TWS Merge Logs - Search Criteria and Document List](image)

**Figure 6-10  TWSMERGE logs**

We select the first TWSMERGE log in the list by double-clicking on it or by selecting **View All Selected**. Figure 6-11 on page 139 shows the first page of the output. Note the thumbnails at the bottom, which show different pages of the document.
Figure 6-11 Viewing the TWSMERGE log

We click on thumbnail 2 to browse the second page in Figure 6-12 on page 140.
6.2.11 Saving and recalling successful queries

When you have entered a set of search criteria on the Search Criteria and
Document List dialog box that is especially useful, you can give that set of search
criteria a name and reuse that set of search criteria at another time. OnDemand
calls this set of search criteria and its associated name a named query. The
name can be up to 30 characters long, and it can consist of any characters on
your keyboard.

Tip: Before naming a set of search criteria, choose the Search command to
make sure that you have entered valid information in your entry fields.
OnDemand includes query commands on the File menu and related toolbar buttons on the OnDemand toolbar that you can use to name and reuse successful queries. You need permission to use the query commands or toolbar buttons. If the query commands or toolbar buttons are grayed out, you do not have permission to use them. Contact your OnDemand administrator if you need to use the query commands or toolbar buttons.

Figure 6-13 shows an example of how to save a query. After writing your search arguments, select **Save a Named Query** from the File menu.

![Save a Named Query](image)

Your query will be saved. If you want to retrieve the query, select **Select a Named Query** from the File menu, and the OnDemand client program will show you the available queries, as in Figure 6-14 on page 142.
6.2.12 Finding information in documents or notes

You can locate information in OnDemand with the **Find** command from the following places:

- Go To dialog box
- Notes dialog box
- Open a Folder dialog box search menu
- Select Recipient Information dialog box

**Locating information in the document you are viewing**

To locate information in the document you are viewing, follow these steps:

1. Select **Find** from the Search menu or choose the toolbar button. The Find dialog box appears.

2. You can specify a search string in three ways:
   - Type the information you want to find in the String entry field. OnDemand searches for the string exactly as you type it, including spaces.
   - Click on the right side of the String entry field to select from previous entries.
   - When you want to search for information in a specific field, choose the **Field** check box and select a field to search in the document you are viewing. OnDemand searches for the information you specify in the field you selected.
3. Click on the **Find** button. OnDemand searches the document and displays the page that contains the information. OnDemand highlights the information with a yellow background.

4. Select **Find Next** or **Find Previous** from the Search menu to display the requested occurrence of the information.

Figure 6-15 shows a Find operation. Note the highlighted sentence with a yellow background pointing the first occurrence of the searched string.

![Figure 6-15 Find operation](image-url)
Here are the options you can use when locating information in the document you are viewing:

- **String**: Contains the information (search string) to be found. Up to 32 search strings can be saved in the drop-down list. The least recently used string will drop out of the list, and the new one will be added to the top of the list. The string can include blanks, but a match might not be found, even if the two strings are separated by a blank in the document. The Find dialog box searches the data portion of the list of folders, the document, the note, or the list of recipients; it does not search the artwork included with the document. Artwork can include such things as images, electronic forms, or logos.

- **Case Sensitive**: Locates the search string with the combination of upper- and lowercase letters.

- **Field**: Searches a field for the search string, allowing you to limit the scope of a search. For example, you want OnDemand to show you the occurrences of the value 1000 only when it appears in the Loan Amount field. The fields you can search are listed in the Field list.

Here are the commands you can use when locating information in the document you are viewing:

- **Find**: Locates the first folder name, page, note, or item in the document list that contains the string that you specified. The first occurrence of the string will be highlighted.

- **Find All**: Locates all occurrences of the search string in a line data document or in the document list. In a line data document, after completing the search, OnDemand splits the viewing window into two parts: the document and the list of occurrences. You can move directly to an occurrence in the document by double-clicking on the line number in the occurrence window. If the document you are viewing was loaded using large object support, then OnDemand can automatically search the entire document. From the document list, the client selects all of the items that contain the search string that you specified.

- **Cancel**: Closes the Find dialog box without performing a search.

**Note**: The Find command searches the data portion of the document; it does not search the artwork included with the document. Artwork can include such things as images, electronic forms, or logos.

**Note**: To enter characters that are not available on your keyboard, press and hold the Alt key and use the numeric keypad to type in the ASCII code number (33 to 255) for the character, then release the Alt key.
Locating information in notes you are viewing
To locate information in notes you are viewing, follow these steps:

1. Select **Find** on the Notes dialog box. The Find dialog box appears.
2. You can specify a search string in two ways:
   - Type the information you want to find in the String entry field. OnDemand searches for the string exactly as you type it, including spaces.
   - Click the down arrow at the right side of the String entry field to select from previous entries.
3. Click on the **Find** button. OnDemand searches the note and highlights the first occurrence of the information.
4. Select the **Find Next** or **Find Previous** button on the Notes dialog box to highlight the requested occurrence of the information. OnDemand removes the highlighting from the current string and highlights the next or previous string.

6.2.13 Adding and viewing notes

Notes are comments attached to the document you are viewing. You can display notes that already exist and create new notes. OnDemand retrieves any notes attached to a document when you select the document for viewing.

Adding notes
You need permission to add notes. If the Add Note button is grayed out when you access the Notes menu or the Notes dialog box, you do not have permission to add notes. Contact your OnDemand administrator if you need to add notes.

1. You can create a note in two ways:
   - When viewing documents, select **Notes** on the OnDemand menu bar and then select the **Add Note** button. (The two choices on the Notes menu are **View Notes** and **Add Note**.) When you select **Add Note**, the Add a Note dialog box appears.
   - Choose the toolbar button. The Add a Note dialog box appears.

   **Note:** When a note already exists for a document, the Notes dialog box appears instead. Choose the **Add Note** button on the Notes dialog box to add a new note to a document.

2. Type your comment in the Add a Note dialog box.
3. Verify the Sharing options. For example, choose **Public** if you want everyone who can open the document you are viewing to be able to read your note.
4. Choose **Note can be copied to another server** if you want to be able to copy the note to another server.

5. Finish the note.

**Important:** After you save a note, you cannot change it. Therefore, check the contents of your note carefully before you save it. Choose **Save** to save the note, exit from the Add a Note dialog box, and return to the document.

Figure 6-16 shows an example of adding a note to a job log document.

---

**Note:** OnDemand places a note mark, also referred to as a yellow document icon, in the first column of the Document List as an indication that a note has been saved for a document. You use the Document Page Note Marks on the Options menu or the toolbar button on the OnDemand toolbar to specify how you want note marks displayed.

**Viewing notes**
You need permission to display or view notes. If the toolbar button or the View Notes button on the Notes menu is grayed out, you do not have permission to
display or view notes. See your OnDemand administrator if you need to display or view notes.

To display or view a note, follow these steps:

1. You can display notes attached the document you are viewing in three ways:
   - When viewing documents, select Notes on the OnDemand menu bar and then select View Notes. (The two choices on the Notes menu are View Notes and Add Note.) When you select View Notes, the Notes dialog box appears.
   - Choose the toolbar button to display the Notes dialog box if a note has been saved with the document you are viewing.
   - When the document you are viewing displays note marks (which resemble Post-It notes), click on the note mark, and OnDemand displays the Notes dialog box showing the oldest note associated with the note mark. Use the Auto Note Retrieval and Document Page Note Marks buttons to see where notes are attached to the document you are viewing.

2. When the Notes dialog box is active, you can select any of the command buttons that are active.

3. To add a note to a document when the Notes dialog box is active, choose the Add Note button. The Add a Note dialog box appears.

4. Type your comment in the Add a Note dialog box.

5. Verify the Sharing options. For example, choose Public if you want everyone who can open the document you are viewing to be able to read your note.

6. Choose Note can be copied to another server if you want to be able to copy the note to another server.

7. Finish the note.

8. Choose the OK button to close the Notes dialog box.

Figure 6-17 on page 148 shows an example of viewing a note.
You need permission to delete notes. If the **Delete Notes** button is grayed out when you access the Notes dialog box, you do not have permission to delete notes or all the notes have been deleted. Contact your OnDemand administrator if you need to delete notes.

To delete a note, follow these steps:

1. When you are viewing notes for a document and you want to delete a note, choose the **Delete Note** button on the Notes dialog box. The Delete a Note dialog box appears.

2. Select the note that you want to delete and choose the **Delete** button. The header on the note changes from blue to red.

---

**Note:** A note mark, also referred to as a yellow document icon, in the first column of the Document List indicates that a note has been saved for a document.
3. The next time you open this document, the note will be removed from the Notes dialog box.

4. Continue viewing the notes or choose the OK button to close the Notes dialog box and return to the document you are viewing.

Notes dialog box
When a note or comment has been saved with a document and you select View Notes from the Notes menu or you choose the toolbar button, the Notes dialog box appears. The Notes dialog box lists the most recent note at the top of the dialog box. For more information on viewing notes, see “Viewing notes” on page 146.

Here are the commands that you can use in the Notes Dialog box:

- **OK**: Use to close the Notes dialog box and return to the document you are viewing.

- **Add Note**: Use to add a note or comment to the document you are viewing. When you select Add Note, the Add a Note dialog box appears. You need permission to add notes. If the Add Notes command is grayed out, you do not have permission to add notes. Contact your OnDemand administrator if you need to add notes.

- **Delete Notes**: Use to delete a note or comment associated with the document you are viewing. When you select Delete Note, the Delete a Note dialog box appears. You need permission to delete notes. If the Delete Notes button is grayed out when you access the Notes dialog box, you do not have permission to delete notes. Contact your OnDemand administrator if you need to delete notes.

- **Sort Notes**: Use to sort the notes associated with the document you are viewing. When you select Sort Notes, the Sort Notes dialog box appears.

- **Print Note**: Use to print the note you are viewing. When you select Print Note, the Print dialog box appears.

- **Print All Notes**: Use to print all the notes you are viewing. When you select Print All Notes, the Print dialog box appears.

- **Copy**: Use to copy a highlighted note or comment to the clipboard. You need permission to copy notes. If the Copy button is grayed out when you access the Notes dialog box, you do not have permission to copy notes. Contact your OnDemand administrator if you need to copy notes.

- **Find**: Use to search for a text string in the note or comment you are viewing. When you select Find, the Find dialog box appears.

- **Find Previous**: Use to repeat the last find operation, starting with the previous note.
Find Next: Use to repeat the last find operation, starting with the next note.
Administration of the solution

An extremely important aspect of an OnDemand system is the effective design and implementation of a strategy regarding system administration from a report perspective and from a user authority and responsibility perspective. The focus of this strategy should be to ensure that the system is planned in a manner that provides the greatest functionality and the best performance as the system matures.

The OnDemand and IBM Tivoli Workload Scheduler integration solution also requires administration from the OnDemand perspective.

In this chapter, we discuss the following topics:

- Section 7.1, “Report administration” on page 152
- Section 7.2, “User and group administration” on page 171
7.1 Report administration

Report design and definition is key to a successful implementation of an OnDemand system. Knowledge of the data that is to be indexed, loaded, and retrieved, along with knowledge of OnDemand best practices, results in the most efficient and user friendly system possible. In this section, we consider the processes that are followed when defining an OnDemand report, along with hints and tips to help in the design and implementation process.

The system components that are required for creating, retrieving, and viewing an OnDemand report are a storage set, an application group, an application, and a folder. These elements, in combination, allow the OnDemand administrator to define and create a report definition that can then be used to index and load data into OnDemand so that the end user can search for and retrieve a report.

Figure 7-1 illustrates the relationship of these elements in a typical OnDemand system.
7.1.1 Storage sets

A storage set is defined for each unique storage retention requirement. An application group then declares which storage set will manage the report data being loaded. A storage set contains one or more storage nodes that can be used by several application groups that have the same archive storage requirements. For example, a storage set can be used to maintain data from different application groups that need to retain documents for the same length of time and need the data to be kept on the same type of media.

Different storage sets can be created to handle different data retention requirements. One storage set could be set up to maintain data on cache-only storage, another could be set up to point to an archive storage to maintain data for three years on optical media. Business practices and legal requirements determine the storage management design required.

A more in-depth look into storage management can be found in Chapter 5, “Storage management”, of the Content Manager OnDemand Guide, SG24-6915. However, excerpts from that chapter are repeated here to introduce the various report administration related topics.

7.1.2 Application groups

An application group is a collection of one or more applications that contain common indexing and storage management requirements. The application group contains the database information that is used to load, search for, and retrieve reports. The application group defines the database fields that are to be loaded into the database. Security access to the database tables is also defined here. In the following sections, we take a closer look at aspects of application group definition, which can contribute to a successful OnDemand system implementation.

Database information

The database information category found under General → Advanced → Database Information menu of the application group definition requires that decisions be made concerning database organization, annotation references, and data management for each database table (Figure 7-2 on page 154). These values are important to system performance and maintenance and are described in the following sections.
Maximum rows

The *maximum rows* value determines how many data rows will be loaded into each *physical* database table. Each application group generates a single *logical* table, although it may be made up of multiple physical tables. It is used for segmenting the index data tables and determining when to open a new one.

The default value of 10 million rows is recommended for balancing the performance of data loads and queries. The number of rows specified should be large enough to handle the largest possible input report file, for example, if a single input report file resulted in 5 million logical items, the maximum number of rows should be at least 5,000,000. The value should be decreased if there is a small amount of data associated with the application group, thereby increasing query performance without adversely affecting data load performance.

Loads per database table

The number of loads per database table can be set to multiple or single. For the IITWS/OnDemand solution, we recommend *multiple loads* per database table. If multiple loads is chosen, rows are appended to the open table. When the current application group table reaches the maximum rows value, the table is closed and a new table is opened.

Annotation

OnDemand allows notes to be added, printed, and viewed by a user accessing reports. These notes are stored in a separate database table, but a link is generated to connect a note with a specific document within an application group. Select Yes if you want an icon to be placed on the document list to
indicate a note has been attached to a document (Figure 7-3). If No is selected, you must view the document before knowing a note has been added.

Notes are not viewable from the document list unless you are using the Web browser client.

Figure 7-3  Annotation flags in document database table: Yes

**Data Management**
Do not change this option. OnDemand is the default.

**Storage management**
The storage management settings (Figure 7-4) determine how long report data and indexes will be kept in cache (short-term) storage before being expired. There are also options that determine how soon data will be migrated to archive (long-term) storage after the report load is completed.

Figure 7-4  Application group storage management
**Cache data**
In OnDemand, *cache* refers to disk drives on the object server. Normally, data is stored in cache for the expected period of time for retrievals to occur. When data is required to be stored for a longer period of time (usually for legal purposes), data is accessed from archive storage media (tapes or optical).

The default setting for the IITWS/OnDemand solution is to use a Cache Only storage set, retaining the data for one year from the date the report is loaded.

**Life of data and indexes**
The life of data and indexes settings determine the length of time that database indexes are maintained on an OnDemand system before being deleted from an application group. Indexes should *always* be kept the same length of time as the data in cache.

**Expiration type**
The expiration type determines how rows are expired from the database tables. The default setting is load. Each time a report file is loaded into OnDemand, a variable number of rows are also loaded into the application group tables. When the retention period has been met, all rows associated with that load are expired at one time.

**Permissions**
Permission to access the database tables is administered with the application group. In addition, document access settings are defined here (Figure 7-5 on page 157).
Application group access

There are three settings for this category. First, highlight the user ID (or group of users) from the Users/Group list, then select Add. While the user ID(s) is highlighted in the Defined list, select the permission settings appropriate. An explanation of these settings follows:

Access

User is authorized to query the database.

Administrator

User is authorized to make changes to the application group definition.

Logical Views

User is authorized to create and save their own viewing choices of the data.

Note: *PUBLIC refers to any user who has an OnDemand user ID. The IITWS/OnDemand solution grants *PUBLIC access and the authority to view, copy, and print/fax documents; also, the authority to add and view notes.

When selecting access or administrator permissions, there are default document and notes selections. These can be overridden by an administrator of the application group.

The Add setting authorizes a user ID to load data into the application group.
**Field information**

Field information (Figure 7-6) is used to define the attributes of the database fields that make up the OnDemand report index data. These attributes determine the characteristics of the index data and control many aspects of loading and processing data in the system. When an user enters a value(s) in the folder search field, the search is conducted against the application group database fields. A database field must be added for each required search field.

**Note:** Be sure that all of the database fields needed are included *before* the application group is added to the OnDemand system. Database fields *cannot* be added or removed *after* the application group has been created.

![Figure 7-6](image)

*Figure 7-6  Application group field information*

Next, we consider the following field information attributes in detail:

- **Type**
- **Segment**
- **Application ID field**

**Type**
The type attribute determines the manner in which a database field is used by OnDemand. There are three main types of attributes: *index*, *filter*, and *not in*...
database (Figure 7-7). The IITWS/OnDemand solution does not use Not in Database.

**Figure 7-7  Type attribute**

A field should have a type of index if it is used to uniquely identify a document or if it is frequently used when searching for documents. Designating a field as an index serves to enhance query performance but increases overhead during loading and database maintenance. All search fields with the index-type designation are used as the primary sort field during the search (Figure 7-8). In Figure 7-8, Job Number is defined as an index database field. There is only one log file with a job number of 24886.

**Figure 7-8  OnDemand results using index field search**

A field should have a type of filter if it does not uniquely identify a report but is used in conjunction with an index field during folder queries. For example, you may search for a log file with a unique job number on a specific day; however, if you search for all log files on a specific day, it could result in thousands of matches. In Figure 7-9 on page 160, using only the filter date field resulted in 960 database matches.
Segment

Segment is the date or date/time field that is used to identify which physical table should be searched during a folder query (Figure 7-10). When the maximum number of rows is met, the database manager closes the current table in the application group and automatically opens a new one to continue loading rows. Each physical table is called a table segment. OnDemand can determine the correct table segment to search based on this setting. Only one date field may be defined as segment and should always have a type of filter.

Application ID field

When an application group is defined to store data and index fields from more than one report file, the application ID field is used to identify which application is required for each different report file (Figure 7-11 on page 161). The database mapping fields determine the value stored in the database and also the value to be displayed in the folder. For example, if a separate application were required for loading report files from two separate FTA workstations, the application would be identified in the database by its unique database value. A query could then be made against a specific workstation.
7.1.3 Applications

An application defines the data that is to be indexed and loaded. Each unique report type, that is, a report that has a unique layout, will require a separate application. The indexing definition is specified in the Indexer Information category. If the report has the same layout as another report, but the data needs to be handled differently, then a separate application will be required to handle the unique processing. This processing will be defined in the Load Information category of the application.

Indexer information

As previously mentioned in 1.1, “IBM DB2 Content Manager OnDemand overview” on page 3, OnDemand is an archival and retrieval system. In order to retrieve and view a report, there must be some way of locating that report, that is, reports stored in OnDemand must be indexed. Index information is extracted from the data stream and then stored in the OnDemand database along with storage management information.

The primary indexer in OnDemand is called ACIF (see “Indexer information” on page 161). An administrator defines the index fields and other processing parameters that ACIF uses to locate and extract index information from reports with the OnDemand Administrative client (Figure 7-12 on page 162). The administrative client includes a report wizard that lets you create ACIF indexing parameters by visually marking up sample report data.
Based on the parameters specified in the application, the indexer parses through the report file data stream, attempting to match a character string (or strings). Once the character string is located, the indexes are extracted.

As part of the ITWS/OnDemand solution, you must download a zip file. This file includes the necessary application definitions required for loading all collected IBM Tivoli Workload Scheduler data. See 2.4.1, “Data to be collected” on page 29 for a list of this data. If there are additional IBM Tivoli Workload Scheduler report files you would like to load into OnDemand, you will need to add a new application with the proper indexing parameters. Refer to the IBM Content Manager OnDemand for Multiplatforms Indexing Reference Version 7.1, SC27-0842 for more detailed indexing information.

**Load information**

Load information specifies the processing and resource information that the OnDemand loader uses to load the input data onto storage volumes and the associated index data into the OnDemand database. File format, preprocessor,
and postprocessor parameters (Figure 7-13) are defined as part of load information.

- **File format:** Provides settings that control how the OnDemand system compresses and stores documents and resources. The OD77 setting provides maximum compression of the data for storage. When retrieving the log file or report, a compressed copy of the document will be transferred to the user’s workstation and the OnDemand client will decompress it for viewing.

![Figure 7-13 Application Load Information: File Format](image)

- **Preprocessor:** Specifies processing that is carried out on database fields prior to storing index data. For example, you may want to remove the dashes from a system number (111-2-333) in order to store it as a numeric (1112333). Simply enter the character to be removed in the character removal section of each database field (Figure 7-14).

![Figure 7-14 Application Load Information, Preprocessor Parameters](image)

Also, dates can be presented in a variety of ways, for example, using all numerics, or a combination of numerics and alphabetic characters or special characters. To identify a string of data as a date field so that it can be used in segmenting the database, the OnDemand indexer needs to have the format
of the character string defined. The formatting codes used by OnDemand to convert data into a date field are identified in Table 7-1.

Table 7-1  Date format symbols

<table>
<thead>
<tr>
<th>Format symbol</th>
<th>Description</th>
<th>Format specified</th>
<th>Format example</th>
</tr>
</thead>
<tbody>
<tr>
<td>%m</td>
<td>Two digit month</td>
<td></td>
<td>08</td>
</tr>
<tr>
<td>%d</td>
<td>Two digit day</td>
<td></td>
<td>04</td>
</tr>
<tr>
<td>%y</td>
<td>Two digit year</td>
<td>%m-%d-%y</td>
<td>08-04-03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mm-dd-yy</td>
</tr>
<tr>
<td>%f</td>
<td>One digit month</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>%e</td>
<td>One digit day</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>%Y</td>
<td>Four digit year</td>
<td>%f-%e-%Y</td>
<td>7-4-2003</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>m-d-YYYY</td>
</tr>
<tr>
<td>%B</td>
<td>Month spelled out</td>
<td>%B %e, %Y</td>
<td>August 4, 2003</td>
</tr>
<tr>
<td>%b</td>
<td>Month abbreviated</td>
<td>%b %d, %Y</td>
<td>Aug 04, 2003</td>
</tr>
<tr>
<td>%H</td>
<td>Hours</td>
<td></td>
<td></td>
</tr>
<tr>
<td>%M</td>
<td>Minutes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>%S</td>
<td>Seconds</td>
<td>%H,%M,%S</td>
<td>08.30.59</td>
</tr>
<tr>
<td></td>
<td>*Specify any character between each unit</td>
<td>%S %H %M</td>
<td>59 08 30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>%M-%S-%H</td>
<td>30-59-08</td>
</tr>
<tr>
<td>%T</td>
<td>Hour Minute Second</td>
<td>%T</td>
<td>12:25:02</td>
</tr>
<tr>
<td></td>
<td>*Assumes the ' ' character between each unit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>%j</td>
<td>Julian date</td>
<td>%j %Y</td>
<td>231 2003</td>
</tr>
<tr>
<td></td>
<td>*Year must also be specified</td>
<td>%j %y</td>
<td>231 03</td>
</tr>
</tbody>
</table>

You can create your own date or time format by replacing what is displayed with the format of the date/time that you require. Date strings are not case sensitive. For example, you can specify %B for dates that contain strings like August, AUGUST, or august.

Note: Only the format symbols from the standard formats that are listed can be used.
If the IBM Tivoli Workload Scheduler report files are generated from multiple processors and the date is presented in a different format, then a separate application will be required to define the different formats.

- Postprocessor: Specifies a system command or exit program which will be run against an index file before the index records are loaded into the database. No postprocessor command is required for the ITWS/OnDemand solution.

**Large object support**

Large object support is specified when a single log file is very large and there is no logical way to break it up into smaller individual log files. The ITWS/OnDemand solution does not require the use of large object support.

### 7.1.4 Folders

A folder is the interface that allows a user to search for reports and documents that have been stored in the OnDemand system. The user enters search criteria in the folder search fields; a document list is constructed based on the results of the search. The folder is linked to one or more application group database tables. When the end user selects **Search**, an Structured Query Language (SQL) request is made against one or more physical database tables; if there are any indexes that match, a pointer to the document will be displayed.

The link to an application group is defined in the General category (Figure 7-15).

![Folder General information](image)

*Figure 7-15  Folder General information*

The folder can be customized to provide the look and feel that is desired for the end users of the system, including the order of the search and document fields,
displaying a default value in one or more search field, or whether an entry is required in a particular search field before selecting Search. We will now consider several folder parameter settings that can have an impact on document retrieval performance.

**Display document location**
The Display Document Location setting (Figure 7-15 on page 165) results in an icon appearing next to each entry in a document list indicating whether the item is stored on cache or on archive media. If the cache icon is displayed, the retrieval time will be much shorter than if the archive media icon is displayed. (See Figure 7-16.)

![Cache icon and Archive media icon]

*Figure 7-16 Display Document Location icons*

Enabling this feature can result in degraded search performance because the storage location information for every item returned to the document list must be retrieved from the OnDemand object server. The default setting is to leave this setting blank (do not display the document location).

**Note search**
The note search setting (see Figure 7-15 on page 165) can determine when a search for any attached notes is performed. The possible options are:

- Hit list: When a folder query is run. This setting provides the same result as the annotation setting=yes setting in the application group, but has a direct performance impact when generating the document list.
- Retrieve: OnDemand searches for annotations when the user selects a document for display. When viewing the document, **Notes** will flash in the bottom window bar of the document. Retrieve is the default option.
- Note: OnDemand searches for annotations when the user selects the note command while viewing a document.

We recommend setting the annotation parameter in the Application Group → General → Advanced settings to handle annotation storage and display. When the application group annotation parameter is set to YES, an annotation flag is set in the database when a user adds an annotation to a document. This recommended setting does not create any significant performance impact.
Permissions
The Permissions folder (Figure 7-17) category determines three key options for the end user:

- Access to the folder search screen
- Access to the named query function
- Determining the scope of a search

![Folder permissions](image)

**Authority**
Granting a user ID, or a group of user IDs, access to the folder determines whether the folder will be listed when a user logs on. Without folder access, a user is unable to search for documents. To authorize a user ID folder access, highlight the user id from the list of users and groups displayed, then select Add. While the user ID is highlighted, select Access.

**Note:** When providing a user with folder access, you must also provide application group access.

**Named queries**
When a user initiates a search with the same field values on a regular basis, those search values can be saved and recalled, thus saving the user data entry time. To allow this function to be available, users must be granted permission to create named queries. An explanation of the three settings follows:
Public
Create named queries that any user having access to the folder will be able to use. Only other users with public authority will be able to update or delete a public query.

Private
Create named queries for your own use and allow others to use the query. May not create, update, or delete any query except for any other private query.

View
Allows a user to view both public and private named queries, but may not create, update, or delete their own or any other query.

**Maximum hits**
The *maximum hits* setting defines the maximum number of document list entries to be displayed after a folder query (Figure 7-17 on page 167). An explanation of the three choices is as follows:

- **No limit**
  Return all items matching the search fields.

- **No hits**
  Return no items regardless of the number of matches against the database.

- **__ hits**
  Sets an upper limit of items displayed regardless of the number of matches against the database. When you select the radio button, you must enter a value in the adjacent open field.

No limit is the default. Limiting the scope of a search by restricting the number of hits that can be returned from a query may prevent performance degradation. For example, if the only search criteria entered was a date range over the last five years and there were 20 table segments to be scanned, the resulting number of documents could be many thousands.

**Field definition**
The search fields listed for each folder are named in the Field Definition category (Figure 7-18 on page 169). Once a folder field has been defined, none of its attributes (field type or mapping type) can be changed with the exception of the name itself. You may not remove nor add a field once the folder definition has been saved.
To change a field name, select the field from the list of folder fields (this causes the name to be placed in the Name box on the left side). Click at the end of the name until you have a typing cursor. Once you have edited the name, simply click on the previous name (under the list of Folder Fields) to effect the change.

**Field type**
This must match the field type defined in the application group. For example, if the Job Number field in the application group was defined as an integer data type, then it must also be defined as an integer field type in the folder. The list of valid field types will vary slightly with the database being used in your OnDemand system.

**Mapping type**
This setting declares the relationship between a folder field and an application group database field. The default setting is Single. An explanation of the mapping types follows:

- **Single** The folder field will be linked to a single database field for each application group listed in the folder.
**Range**

The folder field will be linked to two database fields. The two database fields contain only the low and high values of an ascending sorted list of numbers. When a search value is entered, a match occurs if the value falls within the range of the low and high values. This field type has been used for the Start Date and End Date in the TWSMERGE file.

**Operator Or**

The folder field will be linked to two database fields. This allows a user to enter a single search value without knowing where the actual value is stored.

**Text search**

When a folder field is added, the assumption is that a corresponding database field within an application group will actually be searched. However, sometimes the information you are attempting to locate cannot easily be extracted from the input file. The text search folder field type is used to search documents that contain a specific word or phrase before the document hit list is built. Only documents that contain the specified word or phrase will be returned as part of the document list. The search takes place on the server.

Text search is performed only on the documents that first match the criteria from the database. For example, if the other query fields are date and job number, OnDemand will first locate those documents. Once those documents are identified, a text search will be performed. If the document contains the text search string, it will be added as part of the document list. Text search fields do not need to be mapped to database fields.

The only valid search operator is EQUAL. Wildcard searches and pattern searches are not allowed. Text search is not case sensitive.

There are other text search limitations based on the type of the documents to be searched and the platform OnDemand is running. Refer to the latest OnDemand documentation for details. Users should always fully qualify the queries to bring back only the specific documents that they need to view. Any sort of wildcard search in conjunction with a text search could have a severe impact on performance.

**Note:** You can still perform a standard search with this folder. You do not have to specify a text search every time.

**Annotation search**

A folder field type with either Annotation Text Search or Annotation Color Search can be added to allow for searching by either the text inside a note attached to a
document or the color of the note that was added. Neither of these options impact the search performance.

7.2 User and group administration

Security for data stored in an OnDemand system occurs at several levels. The highest level is, of course, access to OnDemand through a user ID and password. In addition, you can assign administrative function authorization to users. After that, security can be applied at the folder level, database table level, or individual documents.

A general philosophy of centralized administrative control or distributed administrative control should be decided upon based on your OnDemand environment. We will address each of those philosophies in order for you to make an educated decision.

Because the ITWS/OnDemand solution to centralize job log files into a single repository has an expected limited viewing audience, you may want a single IBM Tivoli Workload Scheduler administrator to manage the OnDemand definition files. We will look at adding users and grouping those users with common access requirements.

7.2.1 User types, authorities, and functions

When OnDemand is first installed, there will be only one user ID defined: admin with system administrator authority. You will need to log on as admin (not case sensitive, and no password is required) in order to add additional users.

There are five different types of users in an OnDemand system. Each has a different level of access, authority, and responsibility in the system. See Figure 7-19 for an example of user types.

![Figure 7-19 User types](image-url)
Where:

**User**
End users log in and query the system to retrieve documents and reports for viewing. Most OnDemand users fall into this category.

**User admin**
User administrators add, update, or delete end users or other user administrators to the system.

**Report admin**
Report administrators can add, update, or delete the application groups, applications, and folders necessary for loading and viewing data.

**System admin**
System administrators have authority for all system functions and can grant other users the authority to perform various tasks.

The system administrator is the only level of authority that can grant create group authority, create storage sets, and define system printers.

**User plus**
This is a *basic user* for whom you want to grant additional authority within a department-level setting. If the additional authority is Create Users, then this user ID can only add, update and delete users within their span of control, that is, those users that were specifically added by them. This differs from the user admin who can add, update, or delete a user regardless of how they were added, as long as the authority level does not exceed user admin.

If the additional authority is Create Folders, then this user ID can only add, update, and delete folders within their span of control, that is, those folders that were specifically added by them.

A user may have more than one plus authority.

Once the administrative tasks and levels of authorities are understood, decisions must be made concerning the span of control in the system. Is it better to have one user control all access and functions in the OnDemand system or is it better to spread the administrative tasks among several users in order to smooth the workload based on system requirements?

If you decide that one user should control all access and functions in the OnDemand system, simply use the admin user to manage your system.
If you decide to have one user from each IBM Tivoli Workload Scheduler function, (for example, operations, system administration, management, and so on), then the User Plus authority level is best. This will prevent an administrator from one functional area affecting definitions from another area.

### 7.2.2 OnDemand system parameters

Adjust the OnDemand system parameters to comply with your corporate security requirements before adding users. Figure 7-20 displays the OnDemand default system parameters.

![OnDemand default system parameters](image)

*Figure 7-20  OnDemand default system parameters*

**Note:** Inactivity Time Out is measured in requests to the server. Each search is a server request.

### 7.2.3 Add a user

Adding users to the OnDemand system must be done using the OnDemand administrator client. Follow these steps:

**Important:** You need to create a user ID for use when automatically loading the IBM Tivoli Workload Scheduler job logs. The user IDs should have only Basic User authority and be named as follows:

- For z/OS OnDemand: `TWS`
- For Distributed OnDemand: `tws`

1. Select the **Users** category in the administrator.
2. Select **File** → **New User**.
3. Enter the new user name and a password (twice).
4. Do not change the UID number that is automatically assigned.

5. Enter any additional information you may want, such as description, and so on.

6. Select **OK**.

![Figure 7-21 Add a user](image)

When adding a user, the only required entries are user ID, password, and selecting a user type. Any additional information, such as name, department number, phone number, and so on, is optional.

### 7.2.4 Organizing users into groups

Often, users with similar job descriptions need access to similar data. These users are placed into **groups**. If there are 15 people needing access to the same reports, placing them into one group means granting access once to the group instead of 15 individual users. Also, when a new user is added, adding that user into the group means they automatically acquire the group’s access permissions.

**Add a group**

The only users that have the authority to add groups are either the system administrators or the user plus group. Figure 7-23 on page 176 shows the window you use for adding a group.
Figure 7-22 Add a group

The steps necessary for creating a group and assigning users into that group are:

1. Select the **Groups** category in the OnDemand administrator.
2. Select **File → New Group**.
3. Enter the group name.
4. Do not change the GID number that is automatically assigned.
5. Enter any additional information you may want, such as description, and so on.
6. Highlight the user names to be added from the list of users.
7. Select **Add**.
8. Select **OK**.

To define access either to the application group or folder, the group name will appear with the list of users (see Figure 7-23 on page 176). All groups will have a + in front of the group name.
<table>
<thead>
<tr>
<th>Users/Groups</th>
<th>Defined</th>
</tr>
</thead>
<tbody>
<tr>
<td>+Accounting</td>
<td>*PUBLIC</td>
</tr>
<tr>
<td>+SalesDept</td>
<td>+TWSusers</td>
</tr>
<tr>
<td>BOB</td>
<td>ADMIN</td>
</tr>
<tr>
<td>DOROTI</td>
<td>CHERYL</td>
</tr>
<tr>
<td>MIKE</td>
<td>ITSO</td>
</tr>
<tr>
<td>PETE</td>
<td>MARTIN</td>
</tr>
<tr>
<td>TINA</td>
<td>TWS</td>
</tr>
</tbody>
</table>

*Figure 7-23  Groups and users list*
Scripts used in the solution

This appendix describes the scripts used in the ITWS/OnDemand integration solution. You can also refer to flowcharts of the scripts in Appendix B, “Flowcharts” on page 213.

Note: All scripts that are shipped with this redbook are offered on an AS IS basis.

This appendix has the following sections:

- “Setup variables: tws_ondemand_env.sh” on page 179
- “jobmanrc” on page 180
- “Real script: tws_ondemand_real.sh” on page 189
- “Logpush script: tws_ondemand_logpush.sh” on page 191
- “Batch script: tws_ondemand_batch.sh” on page 194
- “Monitor script: tws_ondemand_monitor.sh” on page 196
- “Event Monitor script: tws_ondemand_event.sh” on page 197
- “BmEvents.conf” on page 200
- “Convert script: tws_ondemand_convert.sh (pull method)” on page 201
- “Netman script: tws_ondemand_netman.sh” on page 207
- “TWSmerge script: tws_ondemand_twsmerge.sh” on page 207
- “Database script: tws_ondemand_database.sh (pull method)” on page 208
- “Plan script: tws_ondemand_plan.sh” on page 208
- “tws_ondemand_netman.cmd” on page 209
- “tws_ondemand_twsmerge.cmd” on page 209
- “tws_ondemand_database.cmd” on page 210
- “tws_ondemand_plan.cmd” on page 210
UNIX scripts

In the following sections, we list the UNIX scripts and a configuration file used in the ITWS/OnDemand integration solution.

Setup variables: tws_ondemand_env.sh

#!/bin/ksh
#
#    Setup variables - tws_ondemand_env.sh
#

# TWS variables
#
WHOAMI=`whoami`
MAESTROHOME=`finger $WHOAMI | grep Directory | cut -d " " -f 2`
MAESTROHOME_STDLIST="$MAESTROHOME/stdlist"
MAESTROHOME_ONDEMAND="$MAESTROHOME_STDLIST/ondemand"
UNISON_STDLIST=$1
UNISON_JOBNUM=$2
UNISON_SCHED_DATE=$3

export WHOAMI MAESTROHOME MAESTROHOME_STDLIST MAESTROHOME_ONDEMAND
UNISON_STDLIST UNISON_JOBNUM UNISON_SCHED_DATE

#
# LOAD variables
#
LOAD_TYPE="REAL"    # <<< ATTENTION: set to "REAL" (default) or "BATCH" >>>

export LOAD_TYPE

#
# ARSLOAD variables
#
ARSLOAD_CONVERT_OUTPUT="$PRODUCT_DIR/logs/ondemand/tws_ondemand_convert.out"
ARSLOAD_FAILED="$PRODUCT_DIR/logs/ondemand/tws_ondemand_arsload.failed"
ARSLOAD_OUTPUT="$PRODUCT_DIR/logs/ondemand/tws_ondemand_arsload.out"
ARSLOAD_REAL_OUTPUT="$PRODUCT_DIR/logs/ondemand/tws_ondemand_arsload_real.out"
ARSLOAD_TMP_OUTPUT="$PRODUCT_DIR/logs/ondemand/tws_ondemand_arsload_tmp.out"
ARSLOAD_SERVER="tws3.itsc.austin.ibm.com"

# <<< ATTENTION: set to ONDEMAND_SERVER_HOSTNAME >>>
ARSLOAD_TWS="stdlist"
ARSLOAD_TYPE=$OS
export ARSLOAD_CONVERT_OUTPUT ARSLOAD_FAILED ARSLOAD_OUTPUT ARSLOAD_REAL_OUTPUT
ARSLOAD_TMP_OUTPUT ARSLOAD_SERVER ARSLOAD_TWS ARSLOAD_TYPE

# # ONSERDAND SERVER variables
#
ODSERVER_IP="9.3.4.187" # <<< ATTENTION: set to ONDEMAND_SERVER_IP_ADDRESS >>>
ODSERVER_PORT=1445     # <<< ATTENTION: set to ONDEMAND_SERVER_PORT_NUMBER >>>
ODSERVER_TELNET="$PRODUCT_DIR/logs/ondemand/tws_ondemand_telnet.out"

export ODSERVER_IP ODSERVER_PORT ODSERVER_TELNET

# # EVENT MONITOR variables
#
BMEVENT_FILE=`grep "^FILE" $MAESTROHOME/BmEvents.conf | tail -1 | cut -d= -f2`
BMEVENT_CPULIST="$PRODUCT_DIR/scripts/cpulist"
BMEVENT_TYPE=1         # <<< ATTENTION: include = 1 (default) / exclude = 0 >>>

export BMEVENT_FILE BMEVENT_CPULIST BMEVENT_TYPE

# # WORK variables
#
DATE=`datecalc today - 1 days pic YYYYMMDD`
TERM="vt100"

COUNT_0=`echo $UNISON_STDLIST | wc -c`
COUNT_1=`expr $COUNT_0 - 1`
COUNT_2=`expr $COUNT_0 - 5`
UNISON_END=`echo $UNISON_STDLIST | cut -c$COUNT_2-$COUNT_1`
ODSERVER_SKIP="false"

export DATE TERM UNISON_END ODSERVER_SKIP

###############################################################################
END

jobmanrc

#!/bin/sh
#@(#) $Id: jobmanrc.sh,v 9.9 1999/05/06 21:56:07 pl Exp $
#
# This script is invoked by jobman when launching a job. The job's
# script file is $1
#
## WARNING:
## This shares the message set with 'jobmanrc.csh'. Any change
## in message set id or message ids needs to be applied to other
## script (i.e., jobmanrc.csh) also.
##
## jobmanrc message catalog definitions.
##
## message set id.
##
## MAE_JOBMANRC_SET=226
## MAE_COPYRIGHT_SET=234
##
## message ids.
##
## MSG_STARTING_JOB=5501
## MSG_END_OF_JOB=5502
## COPYRIGHT_CONST=1
##
## Ensure MAEHOME is set even if jobmanrc is invoked
## without the complete pathname.
##
## if [ "`dirname $0`" = "." ]
## then
##   MAEHOME=`pwd`
## else
##   MAEHOME=`dirname $0`
## fi
##
## Set LANG to "C", if it is not set or set to ",", as catopen does
## not assume a default for %L specified in the template for NLSPATH.
##
## if [ -z "$LANG" ]
## then
##   LANG="C"; export LANG
## fi
##
## set NLSPATH so that jobmanrc (any user) can access the catalogs.
##
## NLSPATH=$MAEHOME/catalog/%L/%N.cat:$NLSPATH; export NLSPATH
MECHO=$MAEHOME/bin/mecho; export MECHO

# Changed for TWS 8.2 Mar #
VERSION="8.2"
PATCH=`echo 'Revision: 9.9 '$Revision: 9.9 $'|awk '{ print $2 }'`
BANNER="TWS for UNIX/JOBNANRC $VERSION"
COPYRIGHT="`MECHO $MAE_COPYRIGHT_SET $COPYRIGHT_CONST`"

# If it's just a version check, terminate.
if [ "$1" = "-V" ]
then
  echo "$BANNER"
  echo "$COPYRIGHT"
  echo 'Revision: 9.9 $'
  exit 0
fi

UNISON_DIR=`dirname $0`

# CONFIGURATION ENVIRONMENT VARIABLES.
#

# VARIABLE: USE_EXEC
#

# USE_EXEC is used to eliminate extra shell processes. If it is set
# to "YES" the exec command will be used to start the script. If a
# sub-shell is requested (see SHELL_TYPE), the shell being used will
# be exec'ed. If a local jobmanrc file is used exec will be used
# there as well. Certain options will override this one, such as
# MAIL_ON_ABEND being set to anything but "NO".
#
# DEFAULT: "YES"
# CRON Equivalent: "YES"
USE_EXEC="NO"# Changed to implement OnDemand/TWS

# VARIABLE: UNISON_EXIT
#

# UNISON_EXIT is used to set the behavior of the script in case of an
# error occurring. If UNISON_EXIT is set to "YES" then the first
# command which returns a non-zero exit code in the script will cause
# the script to terminate. The script would then return the same exit
# code as the command which resulted in the "abend". Setting
# UNISON_EXIT to "NO" will result in the script continuing after a
# command returns a non-zero exit code. All other values are
# equivalent to "NO".
#
# DEFAULT: "NO"
# CRON Equivalent: "NO"
UNISON_EXIT="NO"

# VARIABLE: LOCAL_RC_OK
#
# LOCAL_RC_OK is used to allow or disallow the use of local .jobmanrc files by individual users. If LOCAL_RC_OK is set to "YES" and there is a .jobmanrc file in $HOME it will be executed passing $UNISON_JCL as its first argument. If LOCAL_RC_OK is set to "NO" the presence of a .jobmanrc is ignored and $UNISON_JCL is executed directly. Setting LOCAL_RC_OK to any other value is equivalent to "NO".
#
# DEFAULT: "YES"
# CRON Equivalent: "NO"
# Restrictions: The logon user must have read and execute access to .jobmanrc
LOCAL_RC_OK="YES"

if [ "$LOCAL_RC_OK" = "YES" ]
then
  if [ -f $UNISON_DIR/localrc.allow ]
  then
    if grep $LOGNAME $UNISON_DIR/localrc.allow >/dev/null;
    then
      echo
    else
      LOCAL_RC_OK="NO"
    fi
  elif [ -f $UNISON_DIR/localrc.deny ]
  then
    if grep $LOGNAME $UNISON_DIR/localrc.deny >/dev/null;
    then
      LOCAL_RC_OK="NO"
    fi
  fi
fi

if [ "$LOCAL_RC_OK" = "YES" ]
then
  if [ -x .jobmanrc ]
  then
    LOCAL_RC_OK="YES"
  else
    LOCAL_RC_OK="NO"
  fi
fi

# VARIABLE: MAIL_ON_ABEND
# MAIL_ON_ABEND is used to cause a message to be mailed when a job
# fails to run correctly (returns a non-zero exit code). Setting
# MAIL_ON_ABEND to "YES" will cause a message to be mailed to the
# mailbox for the logon user. This message will indicate the name of
# the stdlist file to be examined to determine the cause of the
# failure. Setting it to "NO" will skip the mail step. Any other
# value will be considered to be a user id to mail output to. A list
# could be specified if several users should be notified.
#
# DEFAULT: "NO"
# CRON Equivalent: "YES"
MAIL_ON_ABEND="NO"

if [ "${MAIL_ON_ABEND}" != "NO" ]
then
    USE_EXEC="NO"
fi

# VARIABLE: SHELL_TYPE
#
# SHELL_TYPE is used to configure the method of executing the job
# script ($UNISON_JCL). Setting it to "STANDARD" will cause the
# script to be executed with the shell specified in the first line of
# the script (or "/bin/sh" if none is there) echoing commands to the
# standard list. This is the most "MPE-like" option. Setting it to
# "USER" will cause the script to be executed with $UNISON_SHELL
# echoing commands to the standard list. This is only slightly less
# "MPE-like" then "STANDARD". Using "SCRIPT" will cause the script to
# be executed as directly using the standard shell protocol for
# determining which shell to use. This last option will not echo
# commands to the standard list; only output from the commands will be
# shown. This is the mose "CRON-like" option. Any other value is
# equivalent to "STANDARD".
#
# DEFAULT: "SCRIPT"
# CRON Equivalent: "SCRIPT"
SHELL_TYPE="SCRIPT"

# Other environment variables used are:
#
# HOME The logon user's home directory.
# LOGNAME The logon user's name.
# PATH /bin:/usr/bin
# TZ The timezone in effect.
# UNISON_SHELL The logon user's login shell (not used by jobman).
# UNISON_CPU The maestro CPU name this machine
# UNISON_HOST The maestro CPU name for this machine's host.
# UNISON_JOB The full jobname for the job (<cpu>#<sched>.<jobname>).
# UNISON_JOBNUM The job number for the job.
# UNISON_MASTER The CPU name of the network master.
# UNISON_RUN The current maestro run number.
# UNISON_SCHED The maestro schedule name.
# UNISON_SCHED_DATE The maestro schedule date.
# UNISON_STDLIST The pathname of the job's stdlist file.
# UNISON_SCHED_EPOCH The epoch for the maestro date and time.

UNISON_RC=$0
UNISON_JCL=$1;export UNISON_JCL

[ -d /usr/ucb ] && PATH=$PATH:/usr/ucb

# export UNISON_SCHED_DATE, if it is set in the environment
if [ ! -z "$UNISON_SCHED_DATE" ]
then
  export UNISON_SCHED_DATE
fi

# UNISON_STDLIST is now set in the job's environment by jobman

# For more information, see the Maestro users guide.
#

echo "$BANNER"
echo "$COPYRIGHT"

echo "`$MECHO $MAE_JOBMANRC_SET $MSG_STARTING_JOB $UNISON_RC $UNISON_JCL`"

echo

# Set the shell flags to echo all commands
SHELL_FLAGS="-x"

# Set the shell flags to exit on first error
if [ "$UNISON_EXIT" = "YES" ]
then
  SHELL_FLAGS="$SHELL_FLAGS -e"
fi

# Decide whether $UNISON_JCL should be executed as a script or a
# command. If it is explicitly set up as a command (as determined by
# the jobinfo program) or if it contains spaces or tabs, the
# SHELL_TYPE is forced to "SCRIPT" so that it will simply be invoked
# as a command. This way there are no extra shell processes unless
# the command happens to be a script. If you want the IS_COMMAND
# environment variable available for use in local jobmanrc files the
# export command below should be uncommented.
IS_COMMAND=`$UNISON_DIR/bin/jobinfo IS_COMMAND`

if [ "$IS_COMMAND" = "YES" ]
then
  # The job was explicitly set up as a command.
  SHELL_TYPE="SCRIPT"
else
echo "$UNISON_JCL"|egrep ".\[ \].\;
then
  # The UNISON_JCL contained a space or tab.
  IS_COMMAND="YES"
  SHELL_TYPE="SCRIPT"
fi

# export IS_COMMAND

# Set up which shell to use.

case $SHELL_TYPE in
  STANDARD)
    if [ ""head -1 $1|cut -c1-2"" = ":!" ]
    then
      USE_SHELL="head -1 $1|cut -c3-" $SHELL_FLAGS"
    else
      USE_SHELL="/bin/sh $SHELL_FLAGS"
    fi
  ;;
  USER)
    USE_SHELL="$UNISON_SHELL $SHELL_FLAGS"
  ;;
  SCRIPT)
    USE_SHELL=""
  ;;
esac

# Add the exec command if USE_EXEC is selected.

if [ "$USE_EXEC" = "YES" ]
then
  EXECIT="exec"
else
  EXECIT=""
fi
# Execute the job's script file, via local .jobmanrc, if it exists.

if [ "$LOCAL_RC_OK" = "YES" ]
then
    # Execute the local jobmanrc file. Exec it if USE_EXEC is specified
    # and use the appropriate shell. The JCL string is passed in $1.
    # Whether or not this is a command is passed in $2 to be used in
    # coding the local jobmanrc. The local jobmanrc will probably end in
    # something like:
    #
    # if [ "$2" = "YES" ]
    # then
    #   eval $1
    # else
    #   exec $1
    # fi
    #
    # but it could also be:
    #
    # eval $1
    #
    # if you don't need to eliminate the extra shell process when
    # executing scripts.
    #
    $EXECIT $USE_SHELL $HOME/.jobmanrc "$UNISON_JCL" $IS_COMMAND
    UNISON_RETURN=$?
else
    if [ -z "$USE_SHELL" ]
    then
        if [ "$IS_COMMAND" = "YES" ]
        then
            eval $UNISON_JCL
            UNISON_RETURN=$?
        else
            $EXECIT $UNISON_JCL
            UNISON_RETURN=$?
        fi
    else
        $EXECIT $USE_SHELL "$UNISON_JCL"
        UNISON_RETURN=$?
    fi
fi

# Mail a message to user or to root if the job fails.

if [ "$MAIL_ON_ABEND" = "YES" ]
then
if [ $UNISON_RETURN -ne 0 ]
then
    mail $LOGNAME <<-!
    $UNISON_JOB
    \'$UNISON_JCL\' failed with $UNISON_RETURN
    Please review $UNISON_STDLIST!
fi
elif [ "$MAIL_ON_ABEND" = "ROOT" ]
then
    if [ $UNISON_RETURN -ne 0 ]
    then
        mail root <<-!
        $UNISON_JOB
        \'$UNISON_JCL\' failed with $UNISON_RETURN
        Please review $UNISON_STDLIST!
    fi
fi
elif [ "$MAIL_ON_ABEND" != "NO" ]
then
    if [ $UNISON_RETURN -ne 0 ]
    then
        mail $MAIL_ON_ABEND <<-!
        $UNISON_JOB
        \'$UNISON_JCL\' failed with $UNISON_RETURN
        Please review $UNISON_STDLIST!
    fi
fi

echo "~$MECHO $MAE_JOBMANRC_SET $MSG_END_OF_JOB~"


################################ Content Manager OnDemand Server

MAESTROHOME_MAIN_OUTPUT="$MAEHOME/scripts/logs/tws_ondemand_main.out"
export MAESTROHOME_MAIN_OUTPUT

trap '(sleep 10; nohup $MAEHOME/scripts/tws_ondemand_main.sh $UNISON_STDLIST
$UNISON_JOBNUM $UNISON_SCHED_DATE >$MAESTROHOME_MAIN_OUTPUT 2>&1 & ) & ' 0
Main script: tws_ondemand_main.sh

#!/bin/ksh
#
# Main script -  tws_ondemand_main.sh
#

OS=${(uname)}
if [[ ${OS} = AIX ]] ; then
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
elif [[ ${OS} = HP-UX ]] ; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = SunOS ]] ; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = Linux ]] ; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
else
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
fi

export OS PRODUCT_DIR

. $PRODUCT_DIR/scripts/tws_ondemand_env.sh $1 $2 $3

nohup $PRODUCT_DIR/scripts/tws_ondemand_real.sh $UNISON_STDLIST $UNISON_JOBNUM $UNISON_SCHED_DATE >> $ARSLOAD_REAL_OUTPUT 2>&1 &

Real script: tws_ondemand_real.sh

#!/bin/ksh
#
# Real script -  tws_ondemand_real.sh
#

```bash
OS=$(uname)
if [[ $OS = AIX ]]; then
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
elif [[ $OS = HP-UX ]]; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ $OS = SunOS ]]; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ $OS = Linux ]]; then
    OS=linux
    PRODUCT_DIR=/opt/ondemand
else
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
fi

export OS PRODUCT_DIR

. $PRODUCT_DIR/scripts/tws_ondemand_env.sh $1 $2 $3

if [ $LOAD_TYPE = "REAL" ]
then
    nohup telnet $ODSERVER_IP $ODSERVER_PORT > $ODSERVER_TELNET 2>&1 &
    sleep 1
    cat $ODSERVER_TELNET | grep Escape > /dev/null 2>&1
    if [ $? = 0 ]
    then
        $PRODUCT_DIR/bin/arsload -nfv -h $ARSLOAD_SERVER -g $ARSLOAD_TWS -a $OS -U
        $PRODUCT_DIR/config/arsload.cfg $UNISON_STDLIST > $ARSLOAD_TMP_OUTPUT 2>&1
        if [ $? = 0 ]
        then
            echo "\n\n==================================================
===== 
\narsload: Processing SUCCESSFUL for file $UNISON_STDLIST"
            echo "\n\n==================================================
===== 
        else
            exit 0
        fi
    else
        fi
    fi
fi
```

190 Integrating IBM Tivoli Workload Scheduler Suite and Content Manager OnDemand
Logpush script: tws_ondemand_logpush.sh

#!/bin/ksh -x
#
# Netman / TWSMERGE / Audit Database and Plan script - tws_ondemand_logpush.sh
#

OS=$(uname)
if [[ $(uname) = AIX ]]; then
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
elif [[ $(uname) = HP-UX ]]; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ \${OS} = SunOS ]] ; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
else
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
fi

export OS PRODUCT_DIR

. $PRODUCT_DIR/scripts/tws_ondemand_env.sh

#####   NETMAN.log
#####

UNISON_STDLIST=$MAESTROHOME_STDLIST/logs/$DATE_NETMAN.log
UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/$OS.netman.$DATE_NETMAN.log.twsod.ARD

export UNISON_STDLIST UNISON_STDLIST_ARD

LOG="NETMAN"
ARSLOAD_HOSTNAME=`head -1 $UNISON_STDLIST | awk -F"\|" '{ printf("%-16s| ", $5) }'`

if [ -f $UNISON_STDLIST ]
then
    echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD
    if [ $? = 0 ]
    then
        cat $UNISON_STDLIST >> $UNISON_STDLIST_ARD
    if [ $? != 0 ]
    then
        echo "netman(cat): Processing FAILED for file $UNISON_STDLIST_AR
        fi
    else
        echo "netman(echo): Processing FAILED for file $UNISON_STDLIST_AR
    fi

#####   TWSMERGE.log
#####

UNISON_STDLIST=$MAESTROHOME_STDLIST/logs/$DATE_TWSMERGE.log
UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/OS\twsmerge.$DATE\TWSMERGE.log.twso\d.ARD

export UNISON_STDLIST UNISON_STDLIST_ARD

LOG="TWSMERGE"
ARSLOAD_HOSTNAME=`head -1 $UNISON_STDLIST | awk -F"|" '{ printf("%-16s| ", $5) }'`

if [ -f $UNISON_STDLIST ]
then
  echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1
  if [ $? = 0 ]
  then
    cat $UNISON_STDLIST >> $UNISON_STDLIST_ARD 2>&1
    if [ $? != 0 ]
    then
      echo "twsmerge(cat): Processing FAILED for file $UNISON_STDLIST_ARD"
      fi
    else
      echo "twsmerge(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
      fi
  fi
fi

#####
#####   Audit Database
#####

UNISON_STDLIST=$MAESTROHOME/audit/database/$DATE
UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/OS\database.$DATE\DATABASE.log.twso\d.ARD

export UNISON_STDLIST UNISON_STDLIST_ARD

LOG="DATABASE"
ARSLOAD_HOSTNAME=`hostname | awk '{ printf("%-16s ", $0) }'`

if [ -f $UNISON_STDLIST ]
then
  echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1
  if [ $? = 0 ]
  then
    cat $UNISON_STDLIST >> $UNISON_STDLIST_ARD 2>&1
    if [ $? != 0 ]
    then
      echo "audit database(cat): Processing FAILED for file $UNISON_STDLIST_ARD"
      fi
    else
      echo "audit database(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
      fi
  fi
fi
Batch script: tws_ondemand_batch.sh

#!/bin/ksh
#
# Batch script - tws_ondemand_batch.sh
#

OS="$uname"
if [[ "$OS" = AIX ]]; then
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
elif [[ "$OS" = HP-UX ]]; then
    OS=unix
else
    echo "audit plan(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
fi
fi
fi

####
#####   Audit Plan
####

UNISON_STDLIST=$MAESTROHOME/audit/plan/$DATE
UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/$OS.plan.$DATE.PLAN.log.twsod.ARD
export UNISON_STDLIST UNISON_STDLIST_ARD

LOG="PLAN"

ARGSLOAD_HOSTNAME=`hostname | awk '{ printf("%-16s|", $0) }'`

if [ -f $UNISON_STDLIST ]
then
    echo "$ARGSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1
    if [ $? = 0 ]
    then
        cat $UNISON_STDLIST >> $UNISON_STDLIST_ARD 2>&1
        if [ $? != 0 ]
        then
            echo "audit plan(cat): Processing FAILED for file $UNISON_STDLIST_ARD"
        fi
    else
        echo "audit plan(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
    fi
else
    echo "audit database(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
fi
fi

Batch script: tws_ondemand_batch.sh

#!/bin/ksh
#
# Batch script - tws_ondemand_batch.sh
#

OS="$uname"
if [[ "$OS" = AIX ]]; then
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
elif [[ "$OS" = HP-UX ]]; then
    OS=unix
else
    echo "audit plan(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
fi
fi
fi

####
#####   Audit Plan
####

UNISON_STDLIST=$MAESTROHOME/audit/plan/$DATE
UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/$OS.plan.$DATE.PLAN.log.twsod.ARD
export UNISON_STDLIST UNISON_STDLIST_ARD

LOG="PLAN"

ARGSLOAD_HOSTNAME=`hostname | awk '{ printf("%-16s|", $0) }'`

if [ -f $UNISON_STDLIST ]
then
    echo "$ARGSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1
    if [ $? = 0 ]
    then
        cat $UNISON_STDLIST >> $UNISON_STDLIST_ARD 2>&1
        if [ $? != 0 ]
        then
            echo "audit plan(cat): Processing FAILED for file $UNISON_STDLIST_ARD"
        fi
    else
        echo "audit plan(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
    fi
else
    echo "audit database(echo): Processing FAILED for file $UNISON_STDLIST_ARD"
fi
fi
PRODUCT_DIR=/opt/ondemand
eif [[ ${OS} = SunOS ]] ; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = Linux ]] ; then
    OS=linux
    PRODUCT_DIR=/opt/ondemand
else
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
fi

export OS PRODUCT_DIR
.
$PRODUCT_DIR/scripts/tws_ondemand_env.sh

ls -R $MAESTROHOME_ONDEMAND | grep ARD > /dev/null 2>&1

if [ $? -ne 0 ]
then
    echo "It does not have *.ARD file to load ..."
    exit 0
fi

while [ $ODSERVER_SKIP = "false" ]
do

    nohup telnet $ODSERVER_IP $ODSERVER_PORT > $ODSERVER_TELNET 2>&1 &
    sleep 1

    cat $ODSERVER_TELNET | grep Escape > /dev/null 2>&1

    if [ $? = 0 ]
    then
        ODSERVER_SKIP="true"
        echo "OnDemand Server is up"
    else
        echo "OnDemand Server is down"
        continue
    fi

    for UNISON_STDLIST in `find $MAESTROHOME_ONDEMAND -name ".*ARD"`
do

        $PRODUCT_DIR/bin/arsload -fv -h $ARSLOAD_SERVER -U $PRODUCT_DIR/config/arsload.cfg -A MVS -G JOBNAME $UNISON_STDLIST >> $ARSLOAD_ARD_OUTPUT 2>&1

        if [ $? != 0 ]

Monitor script: tws_ondemand_monitor.sh

#!/bin/ksh
#
# Event Monitor script - tws_ondemand_monitor.sh
#

OS=$(uname)
if [[ ${OS} = AIX ]]; then
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
elif [[ ${OS} = HP-UX ]]; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = SunOS ]]; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = Linux ]]; then
    OS=linux
    PRODUCT_DIR=/opt/ondemand
else
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
fi

export OS PRODUCT_DIR

. $PRODUCT_DIR/scripts/tws_ondemand_env.sh

ps -ef | grep tws_ondemand_event.sh | grep -v grep >/dev/null 2>&1

if [ $? != 0 ]
then
    echo "Process tws_ondemand_event.sh is down, starting it"
    nohup $PRODUCT_DIR/scripts/tws_ondemand_event.sh >/dev/null 2>&1 &
fi
Event Monitor script: tws_ondemand_event.sh

#!/bin/ksh -x
#
# Event Monitor script - tws_ondemand_event.sh
#

OS=$(uname)
if [[ ${OS} = AIX ]] ; then
  PRODUCT_DIR=/usr/lpp/ars
elif [[ ${OS} = HP-UX ]] ; then
  PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = SunOS ]] ; then
  PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = Linux ]] ; then
  PRODUCT_DIR=/opt/ondemand
else
  PRODUCT_DIR=/usr/lpp/ars
fi

export OS PRODUCT_DIR

. $PRODUCT_DIR/scripts/tws_ondemand_env.sh

trap endwatch 0 1 2 15

getstd()
{
  if [ -f $BMEVENT_CPULIST ]
  then
    if [ `cat $BMEVENT_CPULIST | wc -l` = 0 ]
    then
      if [ $BMEVENT_TYPE = 1 ]
      then
        FOUND=0
      else
        FOUND=1
      fi
    else
      grep -i -w $JOBCPUNAME $BMEVENT_CPULIST
      FOUND=$?
    fi
  else
    if [ $BMEVENT_TYPE = 1 ]
    then
      FOUND=0
    else
      $FOUND=$?
    fi
  fi
}


FOUND=1
fi
fi

EXPR=`expr $FOUND + $BMEVENT_TYPE`

if [ $EXPR = 1 ]
then
  MAESTROLINES=0
  EVENT_DATE=`echo $EVENT_TIMESTAMP | cut -c1-8`
  EVENT_TIME=`echo $EVENT_TIMESTAMP | cut -c9-12`
  HOST="$MAESTROHOME/bin/cpuinfo $JOBCPUNAME | grep HOST | cut -d" " -f2`
  CPU_TYPE="$MAESTROHOME/bin/cpuinfo $JOBCPUNAME | grep INFO | cut -d" " -f2`
  if [ "$CPU_TYPE" = "" ]
  then
    CPU_TYPE="$MAESTROHOME/bin/cpuinfo $HOST | grep INFO | cut -d" " -f2`
  fi
  if [ $CPU_TYPE = "Windows" ]
  then
    OS="win"
  else
    if [ $CPU_TYPE = "Linux" ]
    then
      OS="linux"
    else
      OS="unix"
    fi
  fi
  export OS

  if [ $JOBNAME = "ODNETMAN" -o $JOBNAME = "ODTWSMERGE" -o $JOBNAME = "ODDATABASE" -o $JOBNAME = "ODPLAN" ]
  then
    nohup $PRODUCT_DIR/scripts/tws_ondemand_convert.sh $JOBNAME $CPUNAME $UNISON_FILENAME $OS >> $ARSLOAD_CONVERT_OUTPUT 2>&1 &
  else
    UNISON_FILENAME=$MAESTROHOME_ONDEMAND/$OS\.stdlist.$EVENT_DATE.$JOBCPUNAME.$JOBNUM.$EVENT_TIME.ARD
    echo "Writing job stdlist for $JOBCPUNAME to $UNISON_FILENAME"
    $MAESTROHOME/bin/conman "sj $JOBCPUNAME$JOBNUM;single;stdl" 2>/dev/null | grep -v "^sj" > $UNISON_FILENAME
  fi
fi

}

m101()
{
    CPUNAME=$1
    SCHEDNAME=$2
    JOBNAME=$3
    JOBCPUNAME=$4
    JOBNUM=$5
    EVENT_TIMESTAMP=${12}

    export CPUNAME SCHEDNAME JOBNAME JOBCPUNAME JOBNUM EVENT_TIMESTAMP

    getstd "$CPUNAME $SCHEDNAME $JOBNAME $JOBCPUNAME $JOBNUM $EVENT_TIMESTAMP"
}

m104()
{
    CPUNAME=$1
    SCHEDNAME=$2
    JOBNAME=$3
    JOBCPUNAME=$4
    JOBNUM=$5
    EVENT_TIMESTAMP=${12}

    export CPUNAME SCHEDNAME JOBNAME JOBCPUNAME JOBNUM EVENT_TIMESTAMP

    getstd "$CPUNAME $SCHEDNAME $JOBNAME $JOBCPUNAME $JOBNUM $EVENT_TIMESTAMP"
}

endwatch()
{
    echo ending mwatch!
    exit 0
}

# main part is here

tail -f $BMEVENT_FILE | while read line
do
eval m`echo $line | sed 's/\+/\\+/g' 2>/dev/null

done
# OPTIONS=MASTER|OFF

# MASTER This tells batchman to act as the master of the network and
# information on all cpus are returned by this module.
#
# OFF This tells batchman to not report any events.
#
# default on the master cpu is to report all job scheduling events
# for all cpus on the Maestro network (MASTER); default on other cpus
# is to report events for this cpu only.

# LOGGING=ALL|KEY

# ALL This tells batchman all the valid event numbers are reported.
#
# KEY This tells batchman the key-flag filter is enabled
#
# default is ALL for all the cpus

# SYMEVNTS=YES|NO

# YES tells batchman to report a picture of job status events as soon as the
# new plan is
# generated. It is valid only for key-flagged jobs with LOGGING=KEY
#
# NO does not report these events. It is the default.

EVENTS = 51 101 102 104 105 111 151 152 155 201 202 203 204 251 252 301

# <n> is a valid event number (see Maestro.mib traps for the valid event
# numbers and the contents of each event.
#
# default is 51 101 102 105 111 151 152 155 201 202 203 204 251 252 301

# These can be followed with upto 5 different notification paths in the
# following format:
# PIPE=<filename> This is used for communicating with a Unison Fifo file.
# The format of this is a 4 byte message len followed by the message.
# FILE=<filename> This is for appending to the end of a regular file.
# This file will be truncated on each new processing day.
# MSG=<filename%-.msg> This is used for communicating with a Unison Message file.
# The default event strings encoding is UTF-8.
# Use the following keywords instead of the previous ones, if you want events written in local language:
# PIPE_NO_UTF8=<filename>
# FILE_NO_UTF8=<filename>
# MSG_NO_UTF8=<filename%-.msg>

# To communicate with Unison snmp agent, the following is required:
# PIPE=/usr/lib/maestro/MAGENT.P
PIPE=/usr/lib/maestro/MAGENT.P

# To communicate with OperationsCenter using the demonstration log file encapsulation the following is required:
FILE=/disk01/mae82/maestro/event.log

Convert script: tws_ondemand_convert.sh (pull method)

#!/bin/ksh -x
#
# Netman / TWSmerge / Audit Database and Plan script - tws_ondemand_convert.sh
# (pull method)
#
OS=$(uname)
if [[ ${OS} = AIX ]]; then
    OS=unix
    PRODUCT_DIR=/usr/lpp/ars
elif [[ ${OS} = HP-UX ]]; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = SunOS ]]; then
    OS=unix
    PRODUCT_DIR=/opt/ondemand
elif [[ ${OS} = Linux ]]; then
    OS=linux
    PRODUCT_DIR=/opt/ondemand
else
    OS=unix

PRODUCT_DIR=/usr/lpp/ars
fi

export OS PRODUCT_DIR

UNISON_JOBNAME=$1
UNISON_CPUNAME=$2
UNISON_STDLIST=$3
UNISON_OS=$4

export UNISON_JOBNAME UNISON_CPUNAME UNISON_STDLIST UNISON_OS

. $PRODUCT_DIR/scripts/tws_ondemand_env.sh $UNISON_STDLIST

#####
#####   NETMAN.log
#####

if [ $UNISON_JOBNAME = "ODNETMAN" ]
then

UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/$UNISON_OS\,.netman.$DATE\,$UNISON_CPUNAME\.NETMAN.log.ARD

export UNISON_STDLIST_ARD

COUNT=0

if [ -f $UNISON_STDLIST ]
then
    while read LINE
    do
        echo $LINE | grep **ODLOGSTART**
        if [ $? = 0 ]
        then
            COUNT=1
        else
            if [ $COUNT = 1 ]
            then
                echo $LINE | grep **ODLOGSTOP**
                if [ $? = 0 ]
                then
                    break
                else
                    echo $LINE >> $UNISON_STDLIST_ARD.TMP
                fi
            fi
        fi
    fi
done < $UNISON_STDLIST

if [ $COUNT = 1 ]
then
    LOG="NETMAN"
    ARSLOAD_HOSTNAME=`head -1 $UNISON_STDLIST_ARD.TMP | awk -F"|" '{
        printf("%-16s| ", $5),'
    '}
    echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1
fi

if [ $? = 0 ]
then
    cat $UNISON_STDLIST_ARD.TMP >> $UNISON_STDLIST_ARD 2>&1
fi

if [ $? = 0 ]
then
    rm -f $UNISON_STDLIST_ARD.TMP >/dev/null 2>&1
fi
fi
fi
fi

if [ $UNISON_JOBNAME = "ODTWSMERGE" ]
then

UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/$UNISON_OS\twmerge.$DATE\$UNISON_CP\UNAME\TWSMERGE.log.ARD

export UNISON_STDLIST_ARD

COUNT=0

if [ -f $UNISON_STDLIST ]
then
    while read LINE
    do
        echo $LINE | grep **ODLOGSTART**
        if [ $? = 0 ]
        then
            COUNT=1
        else
            if [ $COUNT = 1 ]
            then
                echo $LINE | grep **ODLOGSTOP**
            fi
        fi
    done
fi
if [ $? = 0 ]
then
  break
else
  echo $LINE >> $UNISON_STDLIST_ARD.TMP 2>&1
fi
fi
done < $UNISON_STDLIST

if [ $COUNT = 1 ]
then
  LOG="TWSMERGE"
  ARSLOAD_HOSTNAME=`head -1 $UNISON_STDLIST_ARD.TMP | awk -F"" " ^{ printf("%-16s| ", $5) }``
echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1

  if [ $? = 0 ]
  then
    cat $UNISON_STDLIST_ARD.TMP >> $UNISON_STDLIST_ARD 2>&1
    if [ $? = 0 ]
    then
      rm -f $UNISON_STDLIST_ARD.TMP >/dev/null 2>&1
    fi
  fi
fi
fi
fi

#####
##### Audit Database
#####

if [ $UNISON_JOBNAME = "ODDATABASE" ]
then

  UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/$UNISON_OS\database.$DATE\$UNISON_CP\UNAME\DATABASE.log.ARD

  export UNISON_STDLIST_ARD

  COUNT=0

  if [ -f $UNISON_STDLIST ]
  then
    while read LINE
    do
      
    fi
  fi

204  Integrating IBM Tivoli Workload Scheduler Suite and Content Manager OnDemand
Appendix A. Scripts used in the solution

```bash
echo $LINE | grep **ODLOGSTART**
if [ $? = 0 ]
then
  COUNT=1
else
  if [ $COUNT = 1 ]
  then
    echo $LINE | grep **ODLOGSTOP**
    if [ $? = 0 ]
    then
      break
    else
      echo $LINE >> $UNISON_STDLIST_ARD.TMP 2>&1
      fi
    fi
  fi
done < $UNISON_STDLIST

if [ $COUNT = 1 ]
then
  LOG="DATABASE"
  ARSLOAD_CPUNAME="$MAESTROHOME/bin/cpuinfo $UNISON_CPUNAME | grep NODE | cut -d" " -f2"
  ARSLOAD_HOSTNAME=`echo $ARSLOAD_CPUNAME | awk '{ printf("%-16s| ", $0)'
  echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1
  if [ $? = 0 ]
  then
    cat $UNISON_STDLIST_ARD.TMP >> $UNISON_STDLIST_ARD 2>&1
    if [ $? = 0 ]
    then
      rm -f $UNISON_STDLIST_ARD.TMP >/dev/null 2>&1
    fi
  fi
fi
fi

#####
#####   Audit Plan
#####
if [ $UNISON_JOBNAME = "ODPLAN" ]
then

UNISON_STDLIST_ARD=$MAESTROHOME_ONDEMAND/$UNISON_OS\plan.$DATE\.$UNISON_CPUNAME\E\PLAN.log.ARD
```
export UNISON_STDLIST_ARD
COUNT=0

if [ -f $UNISON_STDLIST ]
then
  while read LINE
  do
    echo $LINE | grep **ODLOGSTART**
    if [ $? = 0 ]
    then
      COUNT=1
    else
      if [ $COUNT = 1 ]
      then
        echo $LINE | grep **ODLOGSTOP**
        if [ $? = 0 ]
        then
          break
        else
          echo $LINE >> $UNISON_STDLIST_ARD.TMP 2>&1
          fi
        fi
      fi
  done < $UNISON_STDLIST

  if [ $COUNT = 1 ]
  then
    LOG="PLAN"
    ARSLOAD_CPUNAME="$MAESTROHOME/bin/cpuinfo $UNISON_CPUNAME | grep NODE |
    cut -d' "' -f2"
    ARSLOAD_HOSTNAME=`echo $ARSLOAD_CPUNAME | awk '{ printf("%-16s| ", $0)
    }'`
    echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD 2>&1
    if [ $? = 0 ]
    then
      cat $UNISON_STDLIST_ARD.TMP >> $UNISON_STDLIST_ARD 2>&1
      if [ $? = 0 ]
      then
        rm -f $UNISON_STDLIST_ARD.TMP >/dev/null 2>&1
      fi
    fi
  fi
fi
Netman script: tws_ondemand_netman.sh

#!/bin/ksh
#
# Netman script  -  tws_ondemand_netman.sh
# (pull method)
#

WHOAMI=`whoami`
MAESTROHOME=`finger $WHOAMI | grep Directory | cut -d " " -f 2`
MAESTROHOME_STDLIST="$MAESTROHOME/stdlist"

DATE=`datecalc today - 1 days pic YYYYMMDD`
MAESTROHOME_NETMAN=$MAESTROHOME_STDLIST/logs/$DATE_NETMAN.log

if [ -f $MAESTROHOME_NETMAN ]
then
    echo "**ODLOGSTART**"
    cat $MAESTROHOME_NETMAN
    echo "**ODLOGSTOP**"
else
    echo "No `echo $MAESTROHOME_NETMAN` exists"
fi

TWSmerge script: tws_ondemand_twsmerge.sh

#!/bin/ksh
#
# Twsmerge script  -  tws_ondemand_twsmerge.sh
# (pull method)
#

WHOAMI=`whoami`
MAESTROHOME=`finger $WHOAMI | grep Directory | cut -d " " -f 2`
MAESTROHOME_STDLIST="$MAESTROHOME/stdlist"

DATE=`datecalc today - 1 days pic YYYYMMDD`
MAESTROHOME_TWSMERGE=$MAESTROHOME_STDLIST/logs/$DATE_TWSMERGE.log

if [ -f $MAESTROHOME_TWSMERGE ]
then
    echo "**ODLOGSTART**"
    cat $MAESTROHOME_TWSMERGE
    echo "**ODLOGSTOP**"
else
else
Database script: tws_ondemand_database.sh (pull method)

#!/bin/ksh
#
#  Database script -  tws_ondemand_database.sh
#  (pull method)
#

WHOAMI=`whoami`
MAESTROHOME=`finger $WHOAMI | grep Directory | cut -d " " -f 2`
MAESTROHOME_AUDIT_DATABASE="$MAESTROHOME/audit/database"

DATE=`datecalc today - 1 days pic YYYYMMDD`
MAESTROHOME_DATABASE=$MAESTROHOME_AUDIT_DATABASE/$DATE

if [ -f $MAESTROHOME_DATABASE ]
then
  echo "**ODLOGSTART**"
  cat $MAESTROHOME_DATABASE
  echo "**ODLOGSTOP**"
else
  echo "No `echo $MAESTROHOME_DATABASE` log exists"
fi

Plan script: tws_ondemand_plan.sh

#!/bin/ksh
#
#  Plan script -  tws_ondemand_plan.sh
#  (pull method)
#

WHOAMI=`whoami`
MAESTROHOME=`finger $WHOAMI | grep Directory | cut -d " " -f 2`
MAESTROHOME_AUDIT_PLAN="$MAESTROHOME/audit/plan"

DATE=`datecalc today - 1 days pic YYYYMMDD`
MAESTROHOME_PLAN=$MAESTROHOME_AUDIT_PLAN/$DATE
if [ -f $MAESTROHOME_PLAN ]
then
    echo "**ODLOGSTART**"
    cat $MAESTROHOME_PLAN
    echo "**ODLOGSTOP**"
else
    echo "No `echo $MAESTROHOME_PLAN` log exists"
fi

Windows scripts

In the following sections, we list the Windows scripts used in the ITWS/OnDemand integration solution.

tws_ondemand_netman.cmd

@echo off

set TWS_HOME=c:\win32app\TWS\mae82\maestro

FOR /F "Tokens=*" %%i in ('%TWS_HOME%\bin\datecalc today -1 days pic YYYYMMDD')
do set DDATE=%%i

set MAESTROHOME_NETMAN=%TWS_HOME%\stdlist\logs\%DDATE%_NETMAN.log

if exist %MAESTROHOME_NETMAN% goto EXISTFILE
goto END

:EXISTFILE
echo "**ODLOGSTART**"
type %MAESTROHOME_NETMAN%
echo "**ODLOGSTOP**"

:END

tws_ondemand_twsmerge.cmd

@echo off

set TWS_HOME=c:\win32app\TWS\mae82\maestro
FOR /F "Tokens=**" %%i in (\%TWS_HOME\%bin\datecalc today -1 days pic YYYYYMMDD')
do set DDATE=%%i

set MAESTROHOME_TWSMERGE=\%TWS_HOME\%stdlist\logs\%DDATE\_TWSMERGE.log

if exist \%MAESTROHOME_TWSMERGE% goto EXISTFILE
goto END

:EXISTFILE
echo "**ODLOGSTART**"
type \%MAESTROHOME_TWSMERGE%
echo "**ODLOGSTOP**"

:END

tws_onondemand_database.cmd

@echo off

set TWS_HOME=c:\win32app\TWS\mae82\maestro

FOR /F "Tokens=**" %%i in (\%TWS_HOME\%bin\datecalc today -1 days pic YYYYYMMDD')
do set DDATE=%%i

set MAESTROHOME_DATABASE=\%TWS_HOME\%audit\database\%DDATE%

if exist \%MAESTROHOME_DATABASE% goto EXISTFILE
goto END

:EXISTFILE
echo "**ODLOGSTART**"
type \%MAESTROHOME_DATABASE%
echo "**ODLOGSTOP**"

:END

tws_onondemand_plan.cmd

@echo off

set TWS_HOME=c:\win32app\TWS\mae82\maestro

FOR /F "Tokens=**" %%i in (\%TWS_HOME\%bin\datecalc today -1 days pic YYYYYMMDD')
do set DDATE=%%i
set MAESTROHOME_PLAN=%TWS_HOME%\audit\plan\%DDATE%

if exist %MAESTROHOME_PLAN% goto EXISTFILE
goto END

:EXISTFILE
echo "**ODLOGSTART***"
type %MAESTROHOME_PLAN%
echo "**ODLOGSTOP***"

:END
Flowcharts

This appendix provides flowcharts of the scripts used in the ITWS/OnDemand integration solution. The flowcharts in this appendix will help you better understand the scripts that make up the ITWS/OnDemand integration solution.

In this appendix, we provide the following flowcharts:

- "Environmental variables" on page 215
- "Push method job stdlist real-time or batch mode" on page 216
- "Push method batch mode netman, TWSmerge, database, and plan" on page 218
- "Push method batch mode" on page 221
- "Pull method monitor" on page 222
- "Pull method event.log and real time" on page 223
- "Pull method convert logs" on page 226
- "Pull method UNIX netman" on page 234
- "Pull method UNIX TWSmerge" on page 235
- "Pull method UNIX database audit" on page 236
- "Pull method UNIX plan audit" on page 237
- "Pull method Windows Netman" on page 238
- "Pull method Windows TWSmerge" on page 239
- “Pull method Windows database audit” on page 240
- “Pull method Windows plan audit” on page 241
Environmental variables

Figure 7-24 and Figure 7-25 on page 216 show the flowcharts of environmental variables processing (tws_ondemand_env.sh).

Figure 7-24  tws_ondemand_env.sh flowchart
Environmental Variables

# ONDEMAND SERVER variables

ODSERVER_IP= <<<< ATTENTION: set to ONDEMAND_SERVER_IP_ADDRESS >>>
ODSERVER_PORT= <<<< ATTENTION: set to ONDEMAND_SERVER_PORT_NUMBER >>>
ODSERVER_TELNET="$PRODUCT_DIR/logs/ondemand/tws_ondemand_telnet.out"

export OSDERVER_IP OSDERVER_PORT OSDERVER_TELNET

# EVENT MONITOR variables

BMEVENT_FILE="grep ""--FILE" $MAESTROHOME/BmEvents.conf | tail -1 | cut -d= -f2"  
BMEVENT_CPULIST="$PRODUCT_DIR/scripts/cpu1ist"
BMEVENT_TYPE=1 <<<< ATTENTION: set to include=1 (default) / exclude=0 >>>

export BMEVENT_FILE BMEVENT_CPULIST BMEVENT_TYPE

# WORK variables

DATE="date calc today -1 days pic VYYMMDD`
TERM="vt100"

COUNT_0="echo $UNISON_STDLIST | wc -c"
COUNT_1="expr $COUNT_0 - 1"
COUNT_2="expr $COUNT_0 - 5"
UNISON_END="echo $UNISON_STDLIST | cut -c$COUNT_2-$COUNT_1"
ODSERVER_SKIP="false"

export DATE TERM UNISON_END OSDERVER_SKIP

Figure 7-25  tws_ondemand_env.sh flowchart

Push method job stdlist real-time or batch mode

Figure 7-26 on page 217 shows the flowchart of Push method job stdlist real-time or batch mode processing (jobmanrc).
Appendix B. Flowcharts

Figure 7-26  Push method job stdlist real-time or batch mode flowchart

Note: The `arsload` command will use the `$UNISON_STDLIST` file and this file will remain on the system even after a successful load. Users are responsible for deleting these files from the system as part of their normal housekeeping procedures.

Note: The `$UNISON_STDLIST_ARDB` file will be created in the $MAESTROHOME/stdlist/ondemand directory and the background ARSLOAD program will push this file to the OnDemand Server.

For more details about the ARSLOAD program, please see Chapter 16, “Automating data loading” in *IBM Content Manager OnDemand for Multiplatforms Installation and Configuration Guide for UNIX Servers Version 7.1*, GC27-0834
Push method batch mode netman, TWSmerge, database, and plan

Figure 7-27 through Figure 7-30 on page 221 show the flowcharts of the Push method batch mode netman, TWSmerge, database, and plan processing (tws_ondemand_logpush.sh).

![Flowchart of Push method batch mode netman, TWSmerge, database, and plan processing](flowchart.png)
Figure 7-28  Push method batch mode netman, TWSmerge, database, and plan
Figure 7-29  Push method batch mode netman, TWSmerge, database, and plan
Figure 7-30  Push method batch mode netman, TWSmerge, database, and plan

Push method batch mode

Figure 7-31 on page 222 shows the flowchart of Push method batch mode processing (tws_onedemand_batch.sh).
Figure 7-31  Push method batch mode flowchart

Pull method monitor

Figure 7-32 on page 223 shows the flowchart of Pull method monitor processing (tws_ondemand_monitor.sh).
Figure 7-32  Pull method monitor flowchart

Pull method event.log and real time

Figure 7-32 through Figure 7-35 on page 226 show the flowcharts of Pull method event.log and real-time processing (tws_ondemand_event.sh).
Figure 7-33  Pull method event.log and real-time flowchart
Figure 7-34  Pull method event.log and real-time flowchart
Pull method convert logs

Figure 7-36 on page 227 through Figure 7-43 on page 234 show the flowcharts of Pull method convert logs processing (tws_ondemand_convert.sh).
Figure 7-36  Pull method convert logs flowchart
PULL method convert logs

LOG="NETMAN"

ARSLOAD_HOSTNAME=`head -1 $UNISON_STDLIST_ARD.TMP | awk -F";" '{ printf("%16s | ", $5)}`
echo "$ARSLOAD_HOSTNAME$LOG" > $UNISON_STDLIST_ARD

$?=0

Y

N

cat $UNISON_STDLIST_ARD.TMP >> $UNISON_STDLIST_ARD

$? != 0

N

Y

rm -f $UNISON_STDLIST_ARD.TMP

2

Figure 7-37  Pull method convert logs flowchart
Appendix B. Flowcharts

Figure 7-38  Pull method convert logs flowchart
Figure 7-39  Pull method convert logs flowchart
Figure 7-40  Pull method convert logs flowchart
Integrating IBM Tivoli Workload Scheduler Suite and Content Manager OnDemand

Figure 7-41  Pull method convert logs flowchart
PULL method convert logs

Figure 7-42  Pull method convert logs flowchart
Pull method UNIX netman

Figure 7-44 on page 235 shows the flowchart of Pull method UNIX netman processing (tws_ondemand_netman.sh).
Pull method UNIX TWSmerge

Figure 7-44 shows the flowchart of Pull method UNIX TWSmerge processing (tws_ondemand_twsmerge.sh).
Pull method UNIX database audit

Figure 7-46 on page 237 shows the flowchart of Pull method UNIX database audit processing (tws_ondemand_database.sh).
Pull method UNIX plan audit

Figure 7-47 on page 238 shows the flowchart of Pull method UNIX plan audit processing (tws_ondemand_plan.sh).
Pull method Windows Netman

Figure 7-48 on page 239 shows the flowchart of Pull method Windows Netman processing (tws_ondemand_netman.cmd).
Pull method Windows TWSmerge

Figure 7-49 on page 240 shows the flowchart of Pull method Windows TWSmerge processing (tws_ondemand_twsmerge.cmd).
Pull method Windows database audit

Figure 7-50 on page 241 shows the flowchart of Pull method Windows database audit processing (tws_ondemand_database.cmd).
Pull method Windows plan audit

Figure 7-50 shows the flowchart of Pull method Windows plan audit processing (tws_on-demand_plan.cmd).
PULL method
WIN plan

```
@echo off
set TWS_HOME=%drive%\%TWS_home%
set TWS_HOME=%TWS_HOME%/bin\datecalc today -1 days pic YYYYYMMDD') do set DDATE=%1
set MAESTROHOME_PLAN=%TWS_HOME%\std\dist\logs\%DDATE%
```

```
if exist %MAESTROHOME_PLAN%
  goto :EXISTFILE
```

```
;!EXISTFILE
  echo ""OUTPUTSTART"
  type %MAESTROHOME_PLAN%
  echo ""OUTPUTSTOP"
```

```
:END
```

Figure 7-51  Pull method Windows plan audit
Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG246629

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the Additional materials and open the directory that corresponds with the redbook form number, SG246629.

Using the Web material

The additional Web material that accompanies this redbook includes the following files:

SG246629.zip  Includes zipped OnDemand application groups, folders, and scripts for the distributed environment
(TWS4DST.zip) and zipped OnDemand application groups and folders for the TWS z/OS environment (TWS4ZOS.zip)

System requirements for downloading the Web material

The following system configuration is recommended:

- **Hard disk space:** 10 MB minimum
- **Operating System:** Windows/UNIX

How to use the Web material

Create a subdirectory (folder) on your workstation as per the specific implementation instructions, and unzip the contents of the Web material zip file into this folder.
# Abbreviations and acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACF</td>
<td>Advanced Communications Function</td>
</tr>
<tr>
<td>AFP</td>
<td>Advanced Function Presentation</td>
</tr>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>CORBA</td>
<td>Common Object Request Broker Architecture</td>
</tr>
<tr>
<td>DM</td>
<td>Domain Manager</td>
</tr>
<tr>
<td>DMTF</td>
<td>Desktop Management Task Force</td>
</tr>
<tr>
<td>E2E</td>
<td>End-to-End</td>
</tr>
<tr>
<td>FTA</td>
<td>Fault Tolerant Agent</td>
</tr>
<tr>
<td>GIF</td>
<td>Graphics Interchange Format</td>
</tr>
<tr>
<td>HFS</td>
<td>Hierarchical File System</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machines Corporation</td>
</tr>
<tr>
<td>ISPF</td>
<td>Interactive System Productivity Facility</td>
</tr>
<tr>
<td>ITSO</td>
<td>International Technical Support Organization</td>
</tr>
<tr>
<td>ITWS</td>
<td>IBM Tivoli Workload Scheduler</td>
</tr>
<tr>
<td>JCL</td>
<td>Job Control Language</td>
</tr>
<tr>
<td>JES</td>
<td>Job Execution Subsystem</td>
</tr>
<tr>
<td>JPG</td>
<td>Joint Photographic Experts Group</td>
</tr>
<tr>
<td>JSC</td>
<td>Job Scheduling Console</td>
</tr>
<tr>
<td>JSS</td>
<td>Job Scheduling Services</td>
</tr>
<tr>
<td>MDM</td>
<td>Master Domain Manager</td>
</tr>
<tr>
<td>OD390</td>
<td>OnDemand Server</td>
</tr>
<tr>
<td>ODWEK</td>
<td>OnDemand Web Enablement Kit</td>
</tr>
<tr>
<td>OPC</td>
<td>Operations, Planning, and Control</td>
</tr>
<tr>
<td>PID</td>
<td>Process ID</td>
</tr>
<tr>
<td>PSP</td>
<td>Preventive Service Planning</td>
</tr>
<tr>
<td>PTF</td>
<td>Program Temporary Fix</td>
</tr>
<tr>
<td>RFC</td>
<td>Remote Function Call</td>
</tr>
<tr>
<td>RTM</td>
<td>Recovery and Terminating Manager</td>
</tr>
<tr>
<td>SMF</td>
<td>System Management Facility</td>
</tr>
<tr>
<td>SMP</td>
<td>System Modification Program</td>
</tr>
<tr>
<td>SMP/E</td>
<td>System Modification Program /Extended</td>
</tr>
<tr>
<td>STLIST</td>
<td>Standard List</td>
</tr>
<tr>
<td>TMR</td>
<td>Tivoli Management Region</td>
</tr>
<tr>
<td>USS</td>
<td>UNIX System Services</td>
</tr>
<tr>
<td>VTAM®</td>
<td>Virtual Telecommunications Access Method</td>
</tr>
<tr>
<td>X-agent</td>
<td>Extended Agent</td>
</tr>
<tr>
<td>XCF</td>
<td>Cross-system Coupling Facility</td>
</tr>
</tbody>
</table>
Integrating IBM Tivoli Workload Scheduler Suite 8.2 and Content Manager OnDemand to provide
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 248. Note that some of the documents referenced here may be available in softcopy only.

- Content Manager OnDemand Guide, SG24-6915
- End-to-End Scheduling with Tivoli Workload Scheduler 8.1, SG24-6022

Other publications

These publications are also relevant as further information sources:

- IBM Content Manager OnDemand for iSeries Common Server Indexing Reference Version 5 Release 2, SC27-1160
- IBM Content Manager OnDemand for Multiplatforms Indexing Reference Version 7.1, SC27-0842
- IBM Content Manager OnDemand for Multiplatforms Installation and Configuration Guide for UNIX Servers Version 7.1, GC27-0834
- IBM Content Manager OnDemand for Multiplatforms Web Enablement Kit Implementation Guide Version 7.1, SC27-1000
- IBM Content Manager OnDemand for z/OS and OS/390 Administration Guide Version 7.1, SC27-1374
- IBM Content Manager OnDemand for z/OS and OS/390 Configuration Guide Version 7.1, GC27-1373
- IBM Content Manager OnDemand for z/OS and OS/390 Indexing Reference Version 7.1, SC27-1375
- IBM Tivoli Workload Scheduler Planning and Installation Guide Version 8.2, SC32-1273

OS/390 Language Environment for OS/390 Customization Guide, SC28-1941

OS/390 Language Environment for OS/390 and VM Programming Guide, SC28-1939

z/OS MVS JCL Reference, SA22-7597

IBM Content Manager OnDemand Messages and Codes, SC27-1379, found at:

OnDemand for OS/390 and z/OS - JES Spool Data Capture - ARSYSPIN Reference, found at:
http://www-1.ibm.com/support/docview.wss?uid=swg27002351

Online resources

These Web sites and URLs are also relevant as further information sources:

- IBM DB2 Content Manager OnDemand Product Family: OnDemand Web Enablement Kit

- IBM DB2 Content Manager OnDemand for z/OS
  http://www-3.ibm.com/software/data/ondemand/390/

- IBM DB2 for z/OS and OS/390

- IBM Redbooks
  http://www.redbooks.ibm.com

- IBM Tivoli Workload Scheduler for z/OS

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks
Index

A
ABEND 97, 102
ADD access 99
Administrator authority 99
Adobe Acrobat 9
Advanced Function Presentation 3
AFP 3
AFP Conversion and Indexing Facility 8
AFP Web Viewer 13
AFP2HTML 13
AFP2WEB Transform 13
AIX 78, 92
Annotation parameter 166
Annotations 3
APKACIF indexing exit 49
APKACIF input exit 46
Application 6, 161, 172
Application group 6–7, 153, 172
Application group advanced setting 166
Application ID 160
application name definition 37
Archive media 5, 12
Archive storage 153, 155
Archive storage manager 5, 10–12
ARSBATCH 33
ARSLOAD 33, 35, 82
ARSLOAD process 91
ARSLOAD program 98
ARSSOCKD 33
ARSSOCKD started task 38
ARYSPIN 33, 35
AS IS basis 177
Audit Plan 90
Auditing 78
Authority 171
AutoScroll 124
AutoView 137
Binary file transfer mechanism 85
BMEVENT_CPU_LIST variable 83
BMEVENT_FILE 83
BMEVENT_TYPE 83
BMEVENT_TYPE variable 83
BmEvents.conf 97, 102
BMP 13
Bookmark 4
broker systems 20
business case of integration 25

C
Cache data 156
Cache storage 11
Cache storage manager 5, 11
CEEUOPT csect 48
centralized storage system 27
CGI program 13
Client programs 4
COBOL Enterprise Edition 36
COBOL object code 49
Concept
  application 6
  application group 6–7
  document 9
  document indexing 8
  folder 7
  management programs 11
  report indexing 8
  server 5
Control information 11
Controller 32, 34
CPU 89
CPU usage 16
cpulist file 83
current plan 21
customer transactions 121

D
Daily Planning Reports 32
Data
  attributes 158
  data type 158
deletion 12
loading 7
segmenting 154
Data collection process 28, 30
Data loading programs 11
Data Pulling system 27
Data type
  filter 158
  index 158
  not in database 158
Database auditlog 28
Database control information 11
Database logs 90
Database manager 5, 11
Database Manager 2 36
Datastore 39
DB2 address space 68
DB2 upgrade-RSU0303 81
DB2 Version 7 81
DDNAME 32
default port number 83
default servers 120
Defining
  application 161
  folder 165
  note search 166
  storage set 153
distributed agents 21
DM
  See Domain Manager
Document 9
  removal 12
Document indexing 8
Document location 166
Domain Manager 19–20, 28
Download facility 11

E
E2E Server 39
End separator record 46
end-to-end configuration 20
default port number 83
default servers 120
default port number 83
default servers 120
Defining
  application 161
  folder 165
  note search 166
  storage set 153
distributed agents 21
DM
  See Domain Manager
Document 9
  removal 12
Document indexing 8
Document location 166
Domain Manager 19–20, 28
Download facility 11

E
E2E Server 39
End separator record 46
end-to-end configuration 20
default port number 83
default servers 120
default port number 83
default servers 120
Defining
  application 161
  folder 165
  note search 166
  storage set 153
distributed agents 21
DM
  See Domain Manager
Document 9
  removal 12
Document indexing 8
Document location 166
Domain Manager 19–20, 28
Download facility 11

E
E2E Server 39
End separator record 46
end-to-end configuration 20
default port number 83
default servers 120
default port number 83
default servers 120
Defining
  application 161
  folder 165
  note search 166
  storage set 153
distributed agents 21
DM
  See Domain Manager
Document 9
  removal 12
Document indexing 8
Document location 166
Domain Manager 19–20, 28
Download facility 11

E
E2E Server 39
End separator record 46
end-to-end configuration 20
end-to-end environment 21, 29
environment script 82
EQQMLOG 32
EQQMLOGS 39
EVENT MONITOR variables 83
Event Monitoring 96, 102
EVENT option 104
event.log 83
EVENTS parameter list 104
expanded database 81
Expiration type 156
Extended Agent 28, 96

F
Fault Tolerant Agent 19, 27
fault tolerant workstations 17
Faxing documents 3
Field Definition 168
  Annotation Search 170
  Field Type 169
  Mapping Type 169
  Text search 170
Field information 158
FILE parameter 104
flowcharts 177
Folder 6–7, 165, 172
folder 121
FTA 27
FTP 85
Full Status 29
FULLSTATUS 102
functions 171

G
GIF 13
globalopts file 81
group administration 171

H
Hardware requirements 80
Hewlett-Packard 18
Hit list 166
HP-UX 78, 92

I
IBM 32
IBM Content Manager OnDemand for Multiplatforms
  Version 7.1 79
IBM DB2 Content Manager OnDemand 15
IBM HTTP Server 13
IBM Tivoli Business Systems Manager 15
IBM Tivoli Configuration Manager 15
IBM Tivoli Data Exchange 15
IBM Tivoli Data Warehouse 16
job stream 81
job stream definitions 81
job stream execution 19
JOBCARD 47
jobmanrc script 91
Jobname 32
JPEG 13
JSC
  See Job Scheduling Console

L
large object 125, 165
Large object support 165
Library server 5, 10
  functions 5
Life of data and indexes 156
Line Data Java applet 13
Linux 78, 92
Linux Extended Agent 89, 96
Linux FTA 89
Linux Standard Agent 96
Linux Standard Agents 89
Load 7, 153, 156, 162
  postprocessor 165
  preprocessor 163
Load information 162
Loads per database table 154
local servers 121
localopts 81
long job 81
Lotus WordPro 9

M
Machine ID 32
Maestro 18
Maintenance 153
manage OS/390 components 15
MANAGED-CPU 88, 97
Management programs 11
Master Domain Manager 19–20, 28
Maximum hits 168, 171
  __ hits 168
  No hits 168
  No limit 168
MDM
  See Master Domain Manager
merge stdlists 81
MPE operating system 18

MSGCLASS 34
Multiple z/OS domains 35

N
Named Queries 167
  Private 168
  Public 167
  View 168
naming convention 97
Netman log 28, 90
Netman process 78
information black hole 25
Note 166
Note search 166

O
Object server 5, 10
  functions 5
OD/390 server 43
ODSERVER_IP 83
ODSERVER_PORT 83
ODWEK 12
OnDemand administrator 1, 83, 121
OnDemand Client 99
OnDemand client program 119
OnDemand configuration
  distributed 10
  single 10
OnDemand database 32
OnDemand Generic Indexer 9
OnDemand management programs 11
OnDemand PDF Indexer 9
OnDemand Server 27–29, 82
OnDemand Server components
  cache storage manager 11
  data loading program 11
  database manager 11
  download facility 11
  request manager 11
ONDEMAND SERVER variables 83
OnDemand Server's IP address 83
OnDemand Server's port number 83
OnDemand system parameters 173
OnDemand Web Enablement Kit 12, 78
Operations Planning and Control 17
Operator Instructions 32
Optical 5
Oracle 96
OS/400 Indexer  9
os_type prefix  92

P
PCX  13
PDF  9
PeopleSoft  96
Performance  153–154, 166
plan  21
Plan auditlog  28
Plan log  92
Prerequisites  79
print output  3
Print Services Facility  11
production day  89
PSF  11
pull  28
Pull method  29, 96
Pull method - batch mode  30
Pull Server  29
PULL-CPU  88, 97
push  28
Push method  29, 90, 92
Push method - real-time mode  30
PUSH-CPU  88, 92

Q
query  140

R
real-time  82
real-time mode  27
Redbooks Web site  248
Redbooks Web site  xviii
Removable Media Manager  16
Report  6–8
Report administration  152
Report design and definition  152
Report administrator  172
Report indexing  8
Report wizard  161
Reprinting documents  3
Request manager  5, 11
Resolve Dependencies  29
RESOLVEDEP  102
Resource  3
Return Code  32
RSU0303  36

S
SAP  96
searching for documents  123
Secondary folder  170
Segment  160
Segmenting index data  154
Selecting and Viewing Documents  123
selection criteria  32
Server  10
Server library server  5
Server object server  5
Server entry field  121
Server print facility  11
Server print manager  5
Server programs  4
Shortcut  4
Show Folders  122
Software requirements  81
spool files  46
Standard Agent  28
started task  68
stdlist  89
Storage
Storage archive media  5
Storage cache  11
Storage optical  5
Storage tape  5
Storage management  155
Storage manager
Storage archive storage manager  5, 10–12
Storage cache storage manager  5, 11
Storage set  43, 153, 172
SUCC  97
Sun Solaris  78, 92
Symphony file  21
SY SORT  32
System administrator  172
System Automation for  390  15
System logging  11
system messages  46
System printer  172

T
Tape  5
TCP/IP  4
TCP/IP servers  121
Test connectivity 27
Text search 170
Thumbnail 4
TIFF 9, 13
timeliness 89
toolbar 141
Tracker 32, 34
tracker tasks 34
Transaction 8
Transaction log 9
Transaction value 10
tws user 95
tws_ondemand_env.sh 81
tws_ondemand_event.sh 97
tws_ondemand_main.sh 95, 107, 109
TWS4DIST.zip 84
TWS4ZOS.ZIP 41
TWSmerge 92
TWSmerge file 81
TWSmerge log 28, 90

U
UID number 174
Unison Maestro 18
Unison Software 18
UNIX Backup Master 90
UNIX Domain Manager 90
UNIX Fault Tolerant Agent 90
UNIX Master Domain Manager 90
UNIX Standard Agent 90
UNIX System Services 33, 116
Update Servers 120
User access 7
User administration 171
User administrator 172
User type 171
USS 116

W
Web server program 13
Windows 92
Windows Domain Manager 96
Windows Master 90, 96
Windows Standard Agent 89, 96
workstation capacity 89
workstation disk space 89

Z
z/OS Domain 36
Integrating IBM Tivoli Workload Scheduler and Content Manager OnDemand to Provide Centralized Job Log Processing
Integrating IBM Tivoli Workload Scheduler and Content Manager OnDemand to Provide Centralized Job Log Processing

Solution to integrate ITWS Suite and Content Manager OnDemand

Quick and centralized access to ITWS and ITWS for z/OS logs

Scripts provided for additional customization

This IBM Redbook implements a solution that integrates IBM® Tivoli® Workload Scheduler and IBM DB2® Content Manager OnDemand products to provide centralized output processing for job logs (job outputs, message files, and audit files) from IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduler for z/OS. The solution provides immediate benefit by integrating the job logs into an online, electronic information archive and retrieval system, which is used for quick search and problem resolution purposes.

As part of the solution, we cover defining and implementing the required infrastructure, to access these reports from a simple user interface. We also include examples and scenarios for using this solution.

We include all scripts that make up this solution so that you will be able customize the solution according to your needs.

We anticipate that the solution covered in this book will provide great value for IBM Tivoli Workload Scheduler and IBM Tivoli Workload Scheduler for z/OS customers who are planning to deploy a centralized job logging and browsing system.

For more information: ibm.com/redbooks