WebSphere Product Center: Support Guide

Version 5.0.1

Note! Before using this information and the product it supports, read the information in "Notices" at the end of this document.

7 September2004

This edition of this document applies to WebSphere Product Center (5724-I68), version 5.0.1, and to all subsequent releases and modifications until otherwise indicated in new editions.

Table of Contents

Ch 1 Websphere Product Center monitoring

WebSphere Product Center monitoring can be done through the use of the **rootadmin** and **rmi_status** scripts or through the GUI. There is no standalone-monitoring tool provided with WebSphere Product Center.

The creation of a monitoring tool is a beyond the scope of this document; however, there are several simple ideas which can be mentioned:

- Create a virtual host on an application server, or perhaps on a separate monitoring server. Because rmi_status and rootadmin.sh create system load, the use of a separate monitoring server may be more appropriate for many environments
- Use a scripting language such as Perl to create a CGI wrapper for rootadmin.sh which controls and displays status for services
- Create an alias in the web server which points to $TOP/logs for easy log file browsing
- Create a utility which parses the logs in $TOP/logs for exceptions and other errors and optionally checks service status. This utility could email or provide some other means of notification for the administrator in the event of errors. Run this utility out of cron

Obtain the short status of a service

To get the short status of a service, pass the following parameters:

-cmd=check -svc=<service name>

The short status return one of the following conditions:

running	The service is running and responding to a "heartbeat" function.
not found	The service is not found. The service might not have been started or it might have crashed.
	The service was found as being registered with the RMI

found but not responding	The service was found as being registered with the RMI registry, but it is not responding to the "heartbeat" function. The service might have to be restarted.

Obtain the long status of a service

To get the long status of a service, pass the following parameters to rootadmin.sh:

-cmd=status -svc=<service name>

It will produce an HTML file that can be viewed using any browser. On a terminal, use Lynx (or similar tool) to format the output.

The status gives an overview of the different threads running in the service, as well as a status of the database connections currently taken by the service.

Example:

To get the status of the scheduler:

rootadmin.sh -cmd=status -svc=scheduler > /tmp/sch_status.html; lynx /tmp/sch_status.html

or

rootadmin.sh -cmd=status -svc=scheduler > /tmp/sch_status.html; lynx -dump /tmp/sch_status.html

Note: The ">" used in the example above directs the status details to a file output location.

Database monitoring and management

Since the relational database is the main storage for the bulk of the product information content, it is important to provide management

actions that prevent any degradation and loss of performance.

Setting up alerts in WebSphere Product Center can provide notification on issues that may arise, which may be resolved before it gets out of hand. A monitoring system should also be implemented to constantly monitor the WebSphere Product Center database.

The following tasks should be performed on a regular basis.

- Allocate more space whenever necessary
- Apply fix packs/patch sets as required
- Startup and shutdown the database manager, database, and other services whenever necessary
- Analyze the database schema and collect statistics whenever necessary for better performance
- Re-organize the tables and indexes in regular intervals of time for better performance
- Check the status of backup jobs scheduled
- Restore and recover the database if necessary
- Tune the database performance

1. Allocate more space whenever necessary

Space management is an ongoing task for most of you. Unless you have a completely static database, tables and indexes will regularly grow, or shrink, in size. You need to ensure that sufficient space is available for this to occur without interruption to the ongoing processing. You also need to help ensure that the space is being used efficiently. You can use DB2 Control Center to allocate space when needed. You can also use command line inter to complete the same task.

2. Apply Fix Packs / Patch Sets

Fix packs / patch sets are database system vendor's mechanism for delivering fully tested and integrated product fixes on a regular basis. They provide bug fixes only; they do not include new functionality, and do not require certification on the target system. It is very important to apply the fix packs / patch sets when they are available to avoid any known problems with the database system. Contact database system

vendor for more information on the fixes.

3. Startup and shutdown the database manager and the database

It is required to shutdown the database manager / database as part of applying the fixes, moving the databases from one server to another server etc. You will have to startup/shutdown the database as and when needed.

4. Analyze the database schema and collect the statistics

Database schema is analyzed to collect the latest statistics about tables and indexes in the database. The cost-based optimization approach uses statistics to determine an estimate for the cost of each execution plan. Gather statistics on a regular basis to provide the optimizer with the best information about schema objects. For example, after loading a significant number of rows into a table, you should collect new statistics for the table.

To analyze database schema, run the shell script analyze_schema.sh located at $TOP/src/db/schema/util directory.

5. Re-organize the tables and indexes

It is recommended to re-organize the tables and indexes on regular intervals of time for better performance

With today's databases growing faster than ever, the typical DBA must spend a significant amount of time performing space management and reorganization to achieve optimal performance.

Optimal performance means optimal response time. But this can degrade due to a number of space management issues. Most of these issues fall in three main areas namely table-related issues, Stagnated indexes and I/O balancing and data partitioning

Table-related issues are well known to most DBAs. They include

underutilized space inside table blocks, chained rows, poor data proximity, and fragmented (overextended) tables.

The second major issue in the performance challenge is stagnated indexes ¾ indexes that have become large and sparsely populated.

This condition can severely degrade the performance of index range scans. It can also waste a substantial amount of disk space.

The third major issue in the performance challenge is I/O balancing and data partitioning. When objects that are frequently accessed reside in the same data file, I/O bottlenecks can result. Tools like reorgchk in DB2 will give you information about which objects need to be reorganized. There are many methods and tools available to re-organize database objects. Read database system vendor specific documentation on re-organizing tables and indexes.

6. Check the status of Backup jobs scheduled

Backups are an integral part of the restore and recovery process. Verify the status of all backup jobs to make sure they are running as scheduled.

Checking the status of backup depends on how you define the backup procedure and what tools are used to take backups. Please go through vendor database system vendor specific documentation on Backups for more information.

7. Restore and Recover the database

In the case of a database failure, determine the type and extent of failure. The analysis should dictate the steps taken to recover the system. Use the restore and recovery process as defined by your IT support group.

Restoring a physical backup is reconstructing it and making it available to the database server. To recover a restored data file is to update it using redo records, that is, records of changes made to the database after the backup was taken.

8. Tune the database performance

One of the biggest responsibilities of a DBA is to ensure that the database is tuned properly. Any RDBMS is highly tunable and allows the database to be monitored and adjusted to increase its performance.

One should do performance tuning for the following reasons:

- The speed of computing might be wasting valuable human time (users waiting for response);
- Enable your system to keep-up with the speed business is conducted; and

Optimize hardware usage to save money (companies are spending millions on hardware).

Refer to the product documentation that was provided with the DB2 product for more information on different methods available to tune the performance of the database.

Ch 2 WebSphere Product Center performance

Managing disk space

It is recommend having 30-50 gigabytes of usable space that is used for both the WebSphere Product Center middleware and temporary partitions.

- WebSphere Product Center middleware partition: 30GB is recommended. The size of the middleware is approximately 65-70 megabytes.
- Temp partition: 2-4 GB usable disk space (the partition is used to hold many large temporary work files that are created and deleted)

In a clustered configuration, shared storage is necessary for the application servers. The static HTML and image files can be synchronized used a utility such as rsync, but shared storage for the web servers are also recommended.

For the application servers, $TOP, the ftp directory and the web server's document root (the location of static HTML and images) are typically on the shared device while supporting applications such as Apache, the JDK and the application server are installed on local storage. Logs can be kept on local storage or shared storage. The temp directory as specified in common.properties should be local.

Temporary files

The following directories hold temporary run-time generated files and are located on the file system:

Note: The temporary file directories may be different depending on the version of WebSphere Product Center installed.

$TOP/public_html/created_files/distributor

- Purpose: For outbound ftp distributions, the queue manager will download a document from the database into this directory, and then ftp the file over to its destination
- Lifespan: The system administrator should delete all files in this directory during all scheduled application downtimes. Sort all files by date and delete anything older than the rolling period that the company has established for all files in this directory, with the recommended rolling period being 7 days.

  Example Using Linux

  cd $TOP/public_html/created_files/distributor

  find . -type f -mtime +7 -exec ls -l {} \; <-- to view which files would be deleted

  find . -type f -mtime +7 -exec rm -f {} \; <-- to delete the files.

$TOP/public_html/suppliers/company code/aggregated_files/

- Purpose: Uploaded import/export files that are fetched via an ftp fetch are placed into this directory.
- Lifespan: Do not delete this directory from the file system, but rather from the WebSphere Product Center GUI, if necessary.

$TOP/public_html/suppliers/company code/tmp_files:

- Purpose: This directory holds temporary work files.
- Lifespan: It is recommended to save the files in this directory for a few weeks and purge them on a regular basis. If desired, purge files automatically on an WebSphere Product Center restart or based on a defined schedule (i.e. files older than two weeks.)

$TOP/logs

- Purpose: This directory holds the WebSphere Product Center middleware logs.
- Lifespan: Sufficient disk space should be provided based on the

log level (error --> debug). The administrator has complete control over this directory size. It is recommended to have 2-3 gigabytes of disk space available, in the event that debug mode is needed. A day's worth of logs at a regular level will generate approximately 30-40 megabytes of logs, which can be setup to auto-purge.

Caching web pages

The default installation of WebSphere Product Center is set to direct proxy servers NOT to cache pages. Allowing the caching of pages will severely limit the ability to use the browser's back button, producing error messages and expired pages. If caching is desired, use the GUI navigational features and avoid the use of the Back button.

Edit the file: common.properties
Parameter: no_cache_directive=on/off

By default, the parameter is set to off

If set to on, it will set the parameters on the response to direct proxy servers not to cache the pages, and will severely limit the ability of the browser's back button

If set to off, the browser's back button is functional and will not cause errors to occur

---

Hardware specifications

Hardware specification should be chosen based on best practices, past experience, and capacity requirements in order to derive optimal performance from WebSphere Product Center.

Application Server

A majority of data objects in WebSphere Product Center are stored within the database server. For this reason, disk storage on the

application servers will only be used to store the OS components, WebSphere Product Center executables, 3rd party components, WebSphere Product Center temporary work files, and WebSphere Product Center logs.

The WebSphere Product Center middleware utilizes several J2EE components, which can each take a large amount of memory. WebSphere Product Center recommends having an application server with 4GBs of memory, of which 2.5GB would typically be utilized for an instance of the WebSphere Product Center middleware.

Database Server

The size of the database server depends on a variety of factors. These can include the number of catalog items, the number of attributes associated with each item, and the size of the catalog attributes.

A safe rule of thumb is to allocate 8kb of space for each attribute. For example, a catalog with 500,000 items, each having 14 attributes, requires the minimum database storage of 56GBs (500,000 item x 14 attributes x 8kb).

This space does not include what is needed for the database binaries, undo segments, temporary table spaces, etc.

Recommended Architecture

The option to utilize an optional scheduler server to handle background transactions is recommended if WebSphere Product Center is used to handle large batch jobs.

Ch 3 Database administration

Database user

The database user and password, which was created for the WebSphere Product Center installation, is defined in common.properities. Changing the password for the database user without updating the common.properties file will cause the WebSphere Product Center middleware to crash. If the password for the database user needs to be changed, be sure to update the property db_password in common.properties. Password authentication is at the operating system level in DB2.

Database backup

Backing up and recovering the database is one of the most critical operations that a database administrator (DBA) performs. For this reason, it is extremely important to implement a well-defined backup and recovery strategy. The following backup strategies are suggested to maintain optimal performance with WebSphere Product Center.

Physical backups

WebSphere Product Center recommends taking daily physical backup of the database. You can take offline physical backup (image backup) of the database or online physical backup (hot backup) of the database based on the availability of system down time. Most of the WebSphere Product Center databases are accessed 24/7 i.e. there may not be any downtime available to take offline backup of the database. Database must be running in logretain mode in DB2 to be able to take online backup of the database. Taking online backup of the database enables you to recover the database to its state at a particular point in time. Refer to the DB2 product documentation for more information.

Logical backups

Logical backups store information about the schema objects created for a database. Using DB2MOVE utility in DB2 you can selectively export specific objects for supplemental protection and flexibility in a database's backup strategy. Database exports are not a substitute for physical backups and cannot provide the same complete recovery advantages that the physical backup offers. Some times logical backups are very handy to setup QA or test instances with the production data. WebSphere Product Center DBM toolkit also has WebSphere Product Center specific instructions on taking logical backup of the WebSphere Product Center database schema.

Database health check

Checking the health of the database system at regular intervals is key to high availability of the system.

Setup DB2 health center alerts

Use the DB2 Health Center to monitor the state of the database environment and make any necessary changes when needed. Health monitor continuously monitors a set of health indicators. If the current value of a health indicator is outside the acceptable operating range defined by its warning and alarm thresholds, the health monitor generates a health alert. DB2 comes with a set of predefined threshold values, which you can later customize.

The following are some of the key tasks that you can perform with the Health Center:

- View the status of the database environment.
- View the alerts for an instance or a database
- View detailed information about alert, and recommended actions.
- Configure the health monitor settings for a specific object, and the default settings for an object type or for all objects within an instance.

- Select which contacts will be notified of alerts with an e-mail or pager message.
- Review the history of alerts for an instance.

Database management toolkit

There are several db management scripts available to manage the WebSphere Product Center Database. All these scripts are put together in the form of a toolkit.

Different tasks covered in the toolkit for DB2 are:

- Purge the data from WebSphere Product Center database
- Analyzing WebSphere Product Center Schema and collecting statistics
- Taking logical backup of WebSphere Product Center Schema
- Collecting the configuration details of the WebSphere Product Center database using sysinfo.sql

Ch 4 Document Store

The Document Store is the area within WebSphere Product Center where every incoming and every outgoing file is stored. This includes import feeds, scripts, reports, and specification files.

The GUI structure provides hyperlinks to files that are stored on the database, which are essentially pointers to the location of the files.

Directories

The document store is displayed in a file structure manner. Files can be accessed from the following Document Store directories:

archives	public_html
eventprocessor	schedule_logs
feed_files	scripts
ftp	tmp
params	users

Ftp and public_html are file system directories that are mounted into the Document Store. They are defined in $TOP/etc/docstore_mount.xml. This file provides the location of various OS File System mount points.

The variables used are "$ftp_root_dir" and "$supplier_base_dir", which are defined in the common.properties configuration file.

Architecture

The database has a tablespace designated for the files stored in the Document store. When a file is stored in the document store, a new record in the DB is created. The database stores the file as a BLOB (Binary Large Object) file.

A BLOB file refers to any random large block of bits that needs to be stored in a database, such as a picture or sound file. The essential point about a BLOB is that it's an object that cannot be interpreted within the database itself.

The database stores BLOBs within a tablespace in the database itself. The advantage of this method is that the database protects the data, using the database server mechanisms that protect all other types of table data, such as backup-and-recovery and security mechanisms.

Managing tablespace

Space management is an ongoing task. The Document Store table will grow or shrink in size. Ensure that sufficient space is available to support the large binary files without interruption to the ongoing processing. Also, ensure that the space is being used efficiently.

Deleting files

When WebSphere Product Center deletes a BLOB file and the corresponding references, the database engine does not free up the allocated space but rather reuses the space for new files.

Thus, each file is stored in a memory block and as the file is deleted, the memory block is reused as new files are added.

Optional GZIPing of BLOBs

To compress documents stored in BLOBs, do the following:

File to edit: common.properties

Parameter: gzip_blobs=true/false

- Valid values are true and false
- If absent, it defaults to false
- If true, documents stored in blobs are compressed

Defragmentation

Due to the multiple additions and deletions of files in the Document Store, the memory blocks can become fragmented. Fragmentation occurs naturally when you use a disk frequently, creating, deleting, and modifying files.

At some point, the operating system needs to store parts of a file in noncontiguous clusters. This is entirely invisible to users, but it can slow down the speed at which data is accessed because the disk drive must search through different parts of the disk to put together a single file.

To improve the Document Store performance, it is best to export and then import the DBL table using compress=y. This will chunk all of the files into one continuous cluster, thus increasing the time to import files.

Note: Depending on the allocation of tablespace, defragmentation may not be needed regularly. Monitor the disk speed regularly to determine if defragmentation of the disk space is needed.

Document Store frequently asked questions

Problem: Once the blobs are deleted, does the WebSphere Product Center speed continues to be impacted?

No. Once the rows are deleted, the slow Document Store pages improve.

Problem: Is the space still allocated causing slow exports/imports?

Yes. The only way to fix this is to export and import the DBL table with compress=y.

Ch 5 Backup and recovery

The particular backup method and software employed is beyond the scope of this document; however, backup concepts are presented here.

Backing up WebSphere Product Center consists of two components: backing up the file system directories where WebSphere Product Center is installed and backing up the database.

WebSphere Product Center backup

To backup WebSphere Product Center, simply backup the $TOP directory as defined in common.properties. Because files do change in these directories, daily backups are recommended. The institution of a backup schedule consisting of a regular full backup and daily incremental backups is recommended.

Database backup

The manner of backing up the database is well beyond the scope of this document, particularly because of the variety of methods available: exports, hot backups, cold backups, mirroring, etc. Whatever manner is chosen, the WebSphere Product Center database user's schema, as defined in common.properties, is all that must be backed up.

Because the database must be available for WebSphere Product Center to run, it is recommended that daily online or 'hot' backups be used. Exports or offline backups should also be taken regularly.

Please see the section "Database backup" for more information on database backups.

Recovery

Recovery can be separated into two categories: recovery of the WebSphere Product Center and supporting files; and recover of the database.

To recover WebSphere Product Center and supporting files, simply restore the missing files or directories to their original locations, and then start WebSphere Product Center.

To recover the database, take the following steps:

- Shutdown application server
- Shutdown the WebSphere Product Center middleware
- Restore the schema
- Start WebSphere Product Center middleware
- Start application server

Ch 6 WebSphere Product Center logger

WebSphere Product Center provides pre-configured files that generate logs, which can then be used to troubleshoot problems within WebSphere Product Center. This document provides an overview of the logging mechanism and explains how to setup the log files.

WebSphere Product Center services log configuration files

The following files control various subsystems within the entire WebSphere Product Center. The location of the generated log is defined in each file.

Note: All paths are relative to $TOP

/etc/logs/eventprocessor.log.xml

/etc/logs/scheduler.log.xml

/etc/logs/system.log.xml

/etc/logs/appsvr.log.xml

/etc/logs/workflowengine.log.xml

Runtime generated logs

Runtime generated logs can be viewed for errors, which help to troubleshoot if a problem is related to the WebSphere Product Center or internal support infrastructure.

The log files generated by WebSphere Product Center are stored in $TOP/logs/*.log.

Configure log files

The properties of the WebSphere Product Center log files can be edited as needed (i.e. location, max size, format.) The following sections describe the elements used to configure the logs and to provide a list of values that may be used when configuring a WebSphere Product Center log file.

Change location

Note: Applies only to File and Rolling appenders

To change the location of a generated log file, change the parameters of the specified log configuration files.

For example:

<param name="File" value="${TOP}/logs/webserver_db.log " />

Change file size

Note: Applies only to rolling appenders

The size of the log file can be set to a specified storage size before it starts to rotate and purge the top order of the output. To control when the file begins to truncate, change the log file size parameter value.

For example:

<param name="maxFileSize" value="10MB" />

Change file backup option

Note: Applies only to rolling appenders

The logger can be defined to keep a specified number of backups for a log file. Once the max value is reached, the oldest file is thrown out.

For example:

<param name="maxBackupIndex" value="2" />

Change conversion pattern

The layout configuration of the logs can be change by redefining the conversion pattern.

For example:

<param name="ConversionPattern" value=
"%d [%t] %-5p %c (%F:%L) %x- %m%n"/>

The conversion pattern is closely related to the conversion pattern of the printf function in C. A conversion pattern is composed of literal text and format control expressions called *conversion specifiers*.

Note: You are free to insert any literal text within the conversion pattern.

Conversion specifiers

Each conversion specifier starts with a percent sign "%" and is followed by optional *format modifiers* and a *conversion character.*

% (format modifiers)(conversion character)

For example,

%-5p [%t]: %m%n

- The format modifiers control such things as field width, padding, left and right justification
- The conversion character specifies the type of data (e.g. category, priority, date, and thread name.)

By default the relevant information is output as is. However, with the aid of format modifiers it is possible to change the minimum field width, the maximum field width and justification.

The optional format modifier is placed between the percent sign and the conversion character. In the example the conversion specifier %-5p means the priority of the logging event should be left justified to a width of five characters.

The first optional format modifier is the *left justification flag* which is just the minus (-) character. Then comes the optional *minimum field width* modifier. This is a decimal constant that represents the minimum number of characters to output. If the data item requires fewer characters, it is padded on either the left or the right until the minimum width is reached.

The default is to pad on the left (right justify) but you can specify right padding with the left justification flag. The padding character is space. If the data item is larger than the minimum field width, the field is expanded to accommodate the data. The value is never truncated.

This behavior can be changed using the *maximum field width* modifier, which is designated by a period followed by a decimal constant. If the data item is longer than the maximum field, then the extra characters are removed from the *beginning* of the data item and not from the end.

For example, if the maximum field width is eight and the data item is ten characters long, then the first two characters of the data item are dropped.

Note: This behavior deviates from the printf function in C where truncation is done from the end.

The following pages provide the values use to define the conversion specifiers.

Format modifiers

Below are various format modifier examples for the category conversion specifier.

Format modifier	Left justify	Min width	Max width	Comment
%20c	False	20	None	Left pad with spaces if the category name is less than 20 characters long.
%-20c	True	20	None	Right pad with spaces if the category name is less than 20 characters long.
%30c	NA	None	30	Truncate from the beginning if the category name is longer than 30 characters.
%20.30c	False	20	30	Left pad with spaces if the category name is shorter than 20 characters. However, if category name is longer than 30 characters, then truncate from the beginning.
%-20.30c	True	20	30	Right pad with spaces if the category name is shorter than 20 characters. However, if category name is longer than 30 characters, then truncate from the beginning.

Conversion characters

The following is a list of recognized conversion characters:

Conversion Character	Effect

c	Used to output the category of the logging event. A precision specifier can optionally follow the category conversion *specifier*, which is a decimal constant in brackets. If a precision specifier is given, then only the corresponding number of right most components of the category name will be printed. By default the category name is printed in full. For example, for the category name "a.b.c" the pattern %c{2} will output "b.c".
d	Used to output the date of the logging event. A date format specifier enclosed between braces may follow the date conversion specifier. For example, %d{HH:mm:ss,SSS} or %d{dd MMM yyyy HH:mm:ss,SSS}. If no date format specifier is given then ISO8601 format is assumed. The date format specifier admits the same syntax as the time pattern string of the SimpleDateFormat. Although part of the standard JDK, the performance of SimpleDateFormat is quite poor. For better results it is recommended to use the log4j date formatters. These can be specified using one of the strings "ABSOLUTE", "DATE" and "ISO8601" for specifying AbsoluteTimeDateFormat, DateTimeDateFormat and respectively ISO8601DateFormat. For example, %d{ISO8601} or %d{ABSOLUTE}. These dedicated date formatters perform significantly better than SimpleDateFormat.
m	Used to output the WebSphere Product Center supplied message associated with the logging event.
	Outputs the platform dependent line separator character

n	or characters. This conversion character offers practically the same performance as using non-portable line separator strings such as "\n", or "\r\n". Thus, it is the preferred way of specifying a line separator.
p	Used to output the priority of the logging event.
r	Used to output the number of milliseconds elapsed since the start of the WebSphere Product Center until the creation of the logging event.
t	Used to output the name of the thread that generated the logging event.
x	Used to output the NDC (nested diagnostic context) associated with the thread that generated the logging event.
%	The sequence %% outputs a single percent sign.

WebSphere Product Center logging setup files

The following examples demonstrate how WebSphere Product Center's log files are defined. The entries in bold set the configuration of the WebSphere Product Center log files.

```
<!-- basic ASYNC appender -->
<appender name="ASYNC" class="org.apache.log4j.AsyncAppender">
<appender-ref ref="DEFAULT"/>
</appender>



<!-- basic CONSOLE appender. This is the same as doing system.out-->
<appender name="STDOUT"
```

```
class="org.apache.log4j.ConsoleAppender">
<layout class="org.apache.log4j.PatternLayout">
<param name="ConversionPattern" value=

"[%t] %-5p %c (%F:%L) %x- %m%n"/>
</layout>
</appender>



<!-- simple FILE appender. The file will be opened and if append is true
-->
<!--                    it will not be truncated                    -->
<appender name="DEFAULT" class="org.apache.log4j.FileAppender">
  <param name="File"   value="${TOP}/logs/tomcat_default.log " />
  <param name="Append" value="true" />
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value=

"%d [%t] %-5p %c (%F:%L) %x- %m%n"/>
   </layout>
</appender>



<!-- Rolling FILE appender. The file will be opened and if append is true
-->
<!--                    it will not be truncated                    -->
<!--                    maxFileSize: How big before you rotate       -->
<!--                    maxBackupIndex: How many backups do you keep?
-->
  <appender name="DB" class="org.apache.log4j.RollingFileAppender">
    <param name="File"   value="${TOP}/logs/tomcat_db.log " />
    <param name="Append" value="true" />
<param name="maxFileSize" value="10MB" />
<param name="maxBackupIndex" value="2" />
<layout class="org.apache.log4j.PatternLayout">
 <param name="ConversionPattern" value=
```

```
"%d [%t] %-5p %c (%F:%L) %x- %m%n"/>
  </layout>
</appender>

<!-- For the austin.db category, you want to have only a few logs kept
hence -->
<!--  the rollingappender -->
<category name="austin.db" additivity=" false">
    <priority value="INFO" />
    <appender-ref ref="DB" />
</category>

<!-- ROOT CATEGORY -->
<!-- MUST ALWAYS BE LAST ENTRY AND HAVE AN
APPENDER-->
<!-- If a logging event is not caught by any other logger it will be
handled by this-->
<!-- rule. -->
<root>
    <priority value="error"/>
    <appender-ref ref="DEFAULT"/>
</root>

</log4j:configuration>
```

Ch 7 Troubleshooting

Tools

- Log file analysis
- netstat
- ps
- kill
- svrmgrl or sqlplus
- telnet
- jar
- tar
- gunzip
- superuser access to the web and application servers is often essential

Application server issues

Environment issues

The WebSphere Product Center pseudo-user on the application server must have the following environment variables configured prior to starting WebSphere Product Center:

- TOP: the top directory of the WebSphere Product Center installation
- DB2_HOME: necessary for DB2 client binaries
- JAVA_HOME: necessary for JDK
- PATH: Must include $DB2_HOME/bin and should include $JAVA_HOME/bin

Additionally, the shell script init_ccd_vars.sh must be sourced before WebSphere Product Center is started. This is usually done in the user's .bashrc file.

The CLASSPATH environment variable should not be modified after init_ccd_vars.sh is sourced.

Common incorrect configuration file settings

- common.properties

  The most common error is an incorrect database specifier in common.properties. An incorrectly configured database will have the following symptoms:

  appsvr, eventprocessor, queuemanager, scheduler, and workflowengine will not start

  Errors in log files logs/db_pool and logs/svc/

  smtp_address. Smtp_address should point to an SMTP relay, either sendmail on the localhost, or another system which is capable of sending email out of the organization.

- License file

  No services will start if the license file (WPC_license.xml) is missing or incorrect. This error will be reflected in the logs files under logs/svc

Application server unresponsive

Scenario

The application server becomes extremely unresponsive. Although it is possible to ping the server, users cannot log into environment and the administrator cannot log into the application server

Things to look for:

Check to see if a user recently launched an unusually large job. If the job was intentional, check the script used by the job.

Database issues

1. Character conversion during data exports / imports

2. Database space allocation problems

3. Data blocks corruption and index corruption problems

4. Import or export hangs with no change in the status bar after a long period of time

5. After killing a running job, the application becomes very slow

6. Redo log switch problems

7. WebSphere Product Center middleware hangs and the GUI is frozen

8. Analyze schema job hangs

9. SQL connection automatically restarts

---

1. Character conversion during data exports / imports

Issue

During the export/import of a database, to create test environments using a copy of the database, error messages regarding the character set used appears.

Symptoms

For example, if a database using character set US7ASCII is exported, the following error message appears in the export log:

Export done in US7ASCII character set and UTF8 NCHAR character set server uses UTF8 character set (possible charset conversion)

Resolution

Whenever exporting/importing the database, set the NLS_LANG parameter to use the character set american_america.utf8.

2. Database space allocation problems

Issue

Occasionally, import and export jobs fail because of insufficient space allocated for tables, indexes, rollback segments and temporary segments.

Symptom

If the rollback segment is full or the rollback segment tablespace is full then you will see the error message in the alert log file similar to following error message shown below:

ORA-1650: unable to extend rollback segment RBS8 by 512 in tablespace RBS

Failure to extend rollback segment 9 because of 1650 condition FULL status of rollback segment 9 set.

Resolution

- Make sure there is enough free space in the tablespaces. For larger jobs, more space may be needed in rollback and temporary segments.
- Check the alert log file of the database every day to see if there are any errors generated related to space issues in the database.

3. WebSphere Product Center slows once a running job is killed

Issue

Whenever a job is killed, like import or export, the database system has to rollback the complete transaction to bring the database to a consistent state. This rollback process utilizes maximum system resources like CPU

time and memory.

Symptoms

The WebSphere Product Center middleware slows once a running job is killed.

Resolution

Wait until the rollback completes and the system returns to a normal state. Do not kill a running job unless it is necessary.

4. Redo log switch problems

Issue

Inadequate number/size of log files can cause the database system to wait a long time for a log switch.

Symptoms

The database system is waiting for a very long time for a log switch and if all the redo log files are active.

Resolution

- Increase the number of log files
- Increase the size of the redo log files

5. WebSphere Product Center middleware hangs and the GUI is frozen

Issue

If errors appear when accessing the WebSphere Product Center middleware, it is possible that the connection to the database has been lost.

Symptoms

The WebSphere Product Center middleware freezes or is in a constant wait state. Errors appear when attempting to access the WebSphere Product Center middleware.

Resolution

- Check the status of listener process
- Check the status of the database whenever connection to the database or the WebSphere Product Center Middleware Screens cannot be established.

6. Analyze schema job hangs

Issue

It is suggested to analyze the schema once in a while when you load huge amount of data into the database or delete purge the tables in the database.

The WebSphere Product Center middleware must be stopped before running analyze schema. If the middleware is not stopped, the analyze schema job may hang because the tables are being used by the middleware.

Symptoms

WebSphere Product Center hangs when running analyze schema.

Resolution

If analyze schema hangs then kill the analyze job, stop the WebSphere Product Center Middleware, analyze schema again and start WebSphere Product Center.

Analyze the schema in regular intervals to collect the latest statistics about the data distribution in the database.

Monitor log files for errors

Monitoring and reviewing system log files can help diagnose and resolve many problems.

Note: This chapter will be extended in the next document version. More information on using log files and troubleshooting techniques will be provided.

Connectivity issues

HTTP post errors

When http post errors occur, consider the following:

1. Can the WebSphere Product Center box see the target destination?

    - Use a Linux/Unix http browser such as "Lynx" and type in the WebSphere Product Center middleware URL to see if the target is accessible.
    - If a browser is not available from the WebSphere Product Center server, try telnetting to port 80 on the destination. For example, if the destination URL is http://myserver/>urlname<, type "telnet myserver 80" (port 80 is the default http port on most web servers).

2. If WebSphere Product Center can see the destination, is the WebSphere Product Center Distributor working correctly?

    - Check for the existence of new files under $TOP/public_html/created_files/distributor. Check to see if any file has the approximate timestamp of when you tried to push the file through.

- It is possible that a runaway script had generated a bad output file. Check the file size. Does the file size correspond to what you were expecting? If the file is an XML or otherwise readable file, type it out. Does it contain the correct information that you were expecting?

3. If the file exists, is the transfer in progress?

    - You can use different tools to see if an actual transfer is in progress. At the minimum, you will need to use a combination of "netstat" and "snoop" (under Solaris) or "tcpdump" (under Linux).
    - Manage your expectations. If the file size is 300 MB, and it is posting to a URL through the Internet, the file can only go at the top speed of the Internet connection.

FTP Fetch Error

If WebSphere Product Center tried to login to a target FTP server and failed to find the specified directory, an error occurs, "Unable to change to remote directory."

There are a couple of reasons for this error:

- The target FTP address is not accessible from the WebSphere Product Center server. From the WebSphere Product Center server, try to FTP directly to the target FTP and verify the file transfer.
- The filename used might be wrong. Check for capitalization and spelling errors.

Test Java Connectivity

The JDBC URL is defined in the file common.properties. To test the Java connectivity from the WebSphere Product Center middleware to the

JDBC URL, use the following script to test for java connectivity.

$TOP/bin/test_java_db.sh

The script tries to connect to the database and run a simple 'select count(*) from dual'. If connection is established, the results from the test script appear.

Other Issues

Stopping and restarting WebSphere Product Center

An issue has been reported when using regular stop scripts under Linux/Solaris. Apparently, WebSphere Product Center does not stop properly or smoothly. If this is the case, stop and start WebSphere Product Center using the following steps:

1. Attempt to gently stop WebSphere Product Center by executing the following script:

    $TOP/bin/go/stop_local.sh

2. Wait for approximately one minute, then type the following command:

    ps –u (USERNAMEWITHOUT THE PARANTHESIS)

3. If there are any active java processes, a scheduled job may still be in progress. If desired, let the job complete, otherwise stop it manually using the following script:

    $TOP/bin/go/abort_local.sh

4. Wait for approximately thirty seconds, then type the following command:

    ps –u (USERNAMEWITHOUT THE PARANTHESIS)

5. If there continue to be active java processes, the JVM has most likely crashed. The java process must be killed manually using the following command:

```
kill `ps -u (USERNAMEWITHOUT THE PARANTHESIS)
| grep java | cut -b10-15`
```

Note: If any java processes still exist, the system may need to be restarted.

6. Once all java processes have been killed, restart the WebSphere Product Center using the following script:

```
$TOP/bin/go/start_local.sh
```

7. Wait approximately one minute and verify the WebSphere Product Center has been started correctly. Run the script $TOP/bin/go/rmi_status.sh or log into the WebSphere Product Center environment.

Ch 8 Integration Best Practices

The purpose of this chapter is to summarize Integration Best Practices in a WebSphere Product Center implementation. Using these best practices will help to achieve a high quality and dependable integration between systems. To cover all aspects of integration, this document has been developed to identify the best practices associated with each of these different aspects of integration.

Key Elements of Integration:

- Design principles
- Implementation
- Validation
- Visibility

Definitions and Acronyms

Integration Dimensions: We can use the dimensions listed below to understand the various kinds of implementations encountered in WebSphere Product Center implementations. The rest of the document will highlight which best practices or guidelines are applicable to which dimensions or kinds of implementations.

WebSphere Product Center as source or target system

The most obvious dimension is whether WebSphere Product Center is the source system or the destination system for the information being exchanged. A source system places its constraints on an integration, the most important of which are (a) the ability to do delta syndications, (b) the ability to initiate an integration, (c) the ability to receive a notification on the success/failure of the transmitted data and take appropriate action, and (d) the protocols and formats supported, as well as support of an EAI infrastructure.

Controlling system

We will define a controlling system as a system that takes action according to an internal trigger for an integration. An example would be WebSphere Product Center running a syndication on a scheduled basis as a job. Another example would be SAP triggering a WBI adaptor as the result of an item add. Whether WebSphere Product Center is the source or target system for an integration is completely independent of which system is the controlling system in an integration. Several cases are possible. When an intermediary, such as an FTP server or an EAI tool, is involved, both source and target systems could be controlling systems: a legacy system on a scheduled basis puts a file on an FTP server, while WebSphere Product Center on a scheduled basis picks up the file. An example of where WebSphere Product Center is a controlled destination system (i.e. it waits for something external to trigger an import of data) would be IBM WBI posting a message to WebSphere Product Center via the invoker with the message contents being an item update from Transora, let's say. An example of where WebSphere Product Center is a controlled source system (i.e. it waits for something external to trigger an export of data) is where IBM WBI polls a WebSphere Product Center queue periodically to see if there is a file ready to be picked up.

Protocol

There is a lot of confusion in WebSphere Product Center implementation teams and in customer resources between protocol, format and message vs. file-based integration. So, one goal of this document is to make sure that we've established a common nomenclature for these concepts. Examples of protocols are File Transfer Protocol (FTP), Hyper Text Transfer Protocol (HTTP), Simple Message Transfer Protocol (SMTP, i.e. email), Java Messaging Service (JMS) and IBM WebSphere Message Queuing (IBM WebSphere MQ). A protocol defines such things as envelopes, encoding of data such as numbers, and expected response, but have nothing to do with the contents being transmitted. In all integrations we should be quite clear on the protocols being used since there will always be at least one. In addition, the various stages of an integration might actually be using different protocols: The WBI adaptor in SAP might be transmitting data from

SAP to the WBI Inter Connection Server (ICS) via HTTP, which then hands over to an IBM MQ queue manager for that data to be transmitted to another queue manager to which WebSphere Product Center is connected as an MQ client.

Format

The format in which the data is laid out is independent of the protocol. Examples of formats are Comma Separated Values (CSV), pipe delimited, eXtensible Markup Language (XML), or just some predefined field and record structure such as for Electronic Data Interchage (EDI) messages. Each format defines fields either through location/length parameters or through tags. It is important to keep in mind the encoding that might be needed for a particular format. For example the fact that characters such as angle brackets ('<', '>') need to be escaped in XML or that content may actually contain commas in CSV is often forgotten in implementations.

Size of data

This dimension is very often confused with "message-based" or "file-based" communication so it is important to get this right. "Message-based" integration is typically assumed to involve smaller exchanges of data, and properties like the following:

- Data is being exchanged more frequently and in smaller chunks so that changes are being communicated within a much smaller time interval of occurrence than in traditional "batch" oriented systems where exports/imports might be happening on a weekly basis.
- The two systems (source and destination) are in contact with each other so that a message sent is being processed and acknowledged back, rather than a file being generated that might get FTP'ed around or sit on a file system for a week before being picked up for processing.

However, there is no clear line that can be drawn to distinguish a message-based from a file-based or batch integration, and it is important to define a clear set of dimensions and not confusing or

overlapping ones. Thus, the overall size of data should be a more important dimension to consider than whether it is labeled as "message-based" or "batch" integration.

Types of communication

Another dimension to consider is the type of communication that will be involved in the integration. Synchronous communication gives direct feedback to a user or system of the result of a particular action. Using HTTP to communicate, for instance, gives automatic feedback to the system or user after an action has been posted. Asynchronous communication, on the other hand, uses more of a "fire-and-forget" strategy. If integration involves depositing a file on an FTP server, which will then be picked up by a system, for instance, there is no automatic feedback to the system depositing the file of the result of its action.

Frequency

Along with the "size of data" dimension, this "how often" dimension captures the total amount of data that will need to be processed on a periodic basis.

Integration thread

This intermediate systems and infrastructure dimension captures whether an EAI infrastructure is being used or not. Also at times in integrations with legacy systems intermediate programs might be written to upload or extract data into the legacy system. It is important to understand and document these intermediate systems or programs since they are most often the weakest link in the integration chain.

Especially in complex integrations that might require multiple hops (WebSphere Product Center to WBI to destination system, for example), non-standard means (such as direct database updates), multiple protocols, or other communication challenges (such as communicating through a firewall), establish a working single thread or complete path of integration early. This will identify issues and give other parties (such as network admins, or teams working on IBM WBI) ample time to resolve these connectivity issues in parallel.

The dimensions of an integration listed above should become the standard terminology for describing integration in WebSphere Product Center implementations. Documentation provided by PS teams in analysis/design stages should use these dimensions clearly and consistently.

Acronyms

Acronym	Definition
EAI	Enterprise Application Integration
FTP	File Transfer Protocol
HTTP	Hyper Text Transfer Protocol
MQ	IBM's message queuing middleware. Often referred to IBM Websphere MQ since all connectivity solutions are now under the WebSphere brand.
ICS	IBM's WBI Inter Connect Server
WBI	IBM's WebSphere Business Integration suite, the EAI suite from IBM.

Design Principles

Re-usability

The overall foundation underlying the implementation methodology of integration is re-usability. As WebSphere Product Center grows and more client implementations are done, it is necessary that we be able to quickly scale and solve integrations with both previously un-integrated systems and those that have been integrated with in previous implementations. In order to solve this requirement, it is necessary that we approach all integration efforts with re-usability in mind such that

should we ever need to integrate with the same system for another client we will be able to do so with the utmost efficiency.

Re-usability is accomplished through: (a) Leveraging EAI tools such as IBM WBI and its model of generic business objects, (b) choosing formats that are independent of the data model, (c) writing libraries of WebSphere Product Center scripts (acknowledgement, polling, etc.) that are independent of the data model and can be reused in other implementations.

Information sharing

Communication as a means to integration

Conceptually, integration can be thought of simply as a series of events that can be triggered by communication between a controlling system WebSphere Product Center and controlled system(s). These events can be triggered through messages that are passed between the systems, automated processes that poll for content or files, or any other means of rudimentary communication. Communications would include, for instance, the type of change to be made (add, update, delete), a unique communication ID (for tracking / confirmation), and the relevant content for effecting the change either within WebSphere Product Center or within the integral system(s).

Reliability measures

In addition to passing information between systems in order to communicate changes, there should also be a means in place to communicate the success or failure of a particular transaction. Such hand-shaking communications can be implemented most intuitively with synchronous forms of communication, and allow the integrated systems to track whether a particular transaction may need to be re-sent due to failed reception at the other end and thus improve and ultimately ensure the integration's reliability.

Information formats

The specific format of these communications should be designed in a generic enough fashion that both the format and the processing

functionality surrounding it can be re-used across implementations.

When considering the general format to utilize for communications between systems, it is important to note the usability of a format in light of the following needs:

- International character sets and special characters (commas, quotation marks, angle brackets, etc.)
- Complex structures (i.e. hierarchies of content and relationships)
- Being able to handle multiple instances of a content or item placeholder with different values per instance

Information processing

While it is conceivable that the format of information that is being sent between systems should be somewhat generic, it is understandable that not all implementations will be able to fit into a pre-defined format, especially if we are to view integration as a direct link between WebSphere Product Center and the integral system(s). To avoid a need for re-tooling of formats and mappings between formats and WebSphere Product Center specs at every implementation due to differences such as data models, it is recommended that we make use of re-usable mapping functionality between XML formats and WebSphere Product Center specs.

Using EAI platforms

One manner of doing this is to make use of EAI platforms such as the WBI or webMethods suite, which will allow us to build re-usable connectors that will allow WebSphere Product Center to, for instance, communicate via a single, completely re-usable message format (i.e. a single XML DTD). Differences that do arise due to an implementation's particulars can then be translated by WBI rather than requiring a re-tooling of WebSphere Product Center functionality to process the information. With no re-tooling of WebSphere Product Center functionality being required, the same functionality can be used across implementations.

Other options

Another factor to consider, however, is that particular clients may have a need to re-use a format already in use with other systems across their enterprise. This would make it difficult for WebSphere Product Center to introduce a completely separate DTD that will then need to be translated for other systems in the enterprise to understand rather than for WebSphere Product Center to make use of the DTD that already exists. In such cases, we should make use of re-usable functionality for translating between specs within WebSphere Product Center and the DTD.

It may also be that even with an EAI platform in use, this approach will be more useful to a particular client from the perspective of understanding the mapping of WebSphere Product Center -managed information to their internal DTD for communicating information, and could thus be a more ideal approach.

Event handling

Ideally an automated process within WebSphere Product Center will handle events. For instance, queuing functionality that was introduced in the WebSphere Product Center release could be used to handle both sending of messages (outbound queues) and receiving both responses and incoming messages (inbound queues). Queue processing scripts could then be utilized to handle the actual processing of the messages and thus, in effect, actually carry out the events that are to be triggered as a result of a specific message.

Event handling need not be tied directly to specific functionality or specific versions of WebSphere Product Center, however. Other means of handling events can include scheduled jobs within WebSphere Product Center that poll an FTP server, scheduled jobs that check a local file system for a file (via the Document Store), having an invoker-based trigger script fire events based on posted information, or other means. The method chosen should ultimately depend on careful consideration of the size of data and frequency dimension requirements of a particular integration.

Change tracking

To be able to implement a full synchronization between systems, there will need to be a means within WebSphere Product Center of tracking changes made to content and items that can furthermore be effectively tagged as communicated to the integral system(s) or not. For instance, if an item is deleted within WebSphere Product Center (as a source system), we likely want to be able to not only trigger a message being sent to the target system that notifies the target system it should delete the same item, but also track the success or failure of that particular communication so that WebSphere Product Center can be aware of whether the item was actually deleted from the target system.

Re-usable connectors

Connectors repository

As implementations progress and our partnerships evolve, we will gradually be building a repository of re-usable connectors to a variety of systems. Whenever possible, we should make every effort to re-use these connectors, as the functionality around processing items and so on that flow through these connectors can then be re-used with little or no modification from the standpoint of a specific implementation. Of course, this will greatly speed up the execution time of the implementation as a whole and enhance the overall reliability and stability of the connectors and the implementations that use theose connectors as issues are found and resolved over time.

When integrating with systems which do not yet have connectors defined, an integration expert should be involved who can quickly build a re-usable connector that can then be used both for the integration on the specific implementation and can also be stored in the repository of connectors for later use should we ever need to integrate with the system on another implementation.

Connector usage

Connectors should be used such that any modifications that may need to be done are done via an EAI layer handling translations of any information that is being passed between systems. In other words, prior to rewriting any of the re-usable functionality within WebSphere Product Center for processing information passed through EAI, we should take advantage of the EAI platform's ability to do any necessary translations such that we do not need to re-write any in- WebSphere Product Center functionality.

Implementation

Scaling the Implementation

Mini-integrations

The large task of an overall integration should be broken into much smaller, more easily manageable tasks. This can be done, for instance, by breaking a single, complete integration into much smaller integration – from "separate" integrations for each item type (spec), to integrations for each container (catalog), all the way down to integrations for a group of attributes (if necessary). Once there is confidence that each of these "mini-integrations" works flawlessly, they can then be combined to form the single, complete integration.

Granularity of functionality

Careful attention should be paid to the levels at which integration between systems needs to occur. For instance, when sending changes to a target system one may want to be able to send all changes since a particular date, only those changes for a particular catalog since the last changes were sent, only changes that have occurred on a specific group of items, or any changes that have occurred on a specific attribute across all items. The specific requirements will be

implementation-dependent, but it is important to take the granularity required into consideration early in the design process of the implementation so that it can be catered for appropriately.

Performance Tuning

General performance comments

Do not leave performance issues as an afterthought. It will be easier to change and fix formats or other aspects of integration late in the game, but a performance bottleneck can require major redesign and sometimes engineering support. Put performance measurement hooks in the scripts in the due course of development.

Measuring performance

Assuming the mini-integration approach (as detailed in the Implementation section), performance should be measured at each step of the integration by measuring the total time required for each mini-integration task. Potential areas of poor performance can then be identified at an appropriately granular level and can thus be much more easily targeted for performance tweaking.

Tweaking performance

Once performance problem areas are identified, a detailed analysis should be done to determine the root cause of slow operation. Detailed analysis can be done by using tools like WebSphere Product Center's middleware profiling and the performance tab on the job detail screen. The analysis can then be applied to focus in on a particular area of a script or SQL query, and appropriate action can then be taken – modifying or rewriting the script, or involving Engineering to enhance a database query.

Validation

Stability

Advantages of mini-integrations

Implementing mini-integrations (as detailed in the Implementation section) should provide a much higher level of confidence that integration has been successful by providing a detailed listing of all areas where integration has been shown to be working. Without the visibility of mini-integrations, it is not only more difficult to provide proof of the details of integration working, but integration as a whole is also more likely to suffer from issues that are difficult to identify, diagnose and debug. Implementing mini-integrations will hence enhance the overall stability of integration.

Scalable Testing

Advantages of mini-integrations

Implementing mini-integrations (as detailed in the Implementation section) allows testing of integration to take place at a much finer level of detail so that any errors or problems that arise are not clouded by large amounts of (potentially irrelevant) complexity. Thus, as mentioned above, the processes of diagnosing, debugging, and resolving issues that are found should all be greatly sped up by this approach.

Representative vs. complete environments

Integration testing should be done on a representative environment with the same configuration as the final environment (same specs, validation rules, value rules, views), but with as few as possible representative entities (locales, catalogs, category trees, items, categories, organizations, users, roles). This should reduce the time it takes for tests to be run, screens to be loaded, and in general

should speed up the time testing takes versus testing in an environment that is fully populated. All testing and debugging should be done in this environment.

Only after testing is complete in the representative environment and everything there appears to be working should integration then be verified in a complete and fully populated environment. This step should still be carried out, however, in order to ensure that no edge scenarios were accidentally ignored in the representative environment as well as to test the production-level performance of the integration.

Scalable process testing

Any schedule-able job (i.e. imports, exports) should first be run only with a very small number of representative items – 10 or less. This number should be increased proportional to the level of confidence that is gained in the script that is actually processing these items. This approach will ensure that a massive job is not executed over a matter of hours only for the person running it to find at the end that something went wrong without causing the entire process to fail outright within the first few minutes of running.

Only after there is full confidence in the operation of the script associated with the job should a job be run involving a complete set of data. As with the complete environment recommendation, this step should still be carried out in order to ensure that no edge scenarios were accidentally ignored in the smaller job runs as well as to test the production-level performance of the job.

Visibility

Reporting

Advantages of mini-integrations

Implementing mini-integrations (as detailed in

the Implementation section) allows a more detailed level of reporting due to smaller and more quickly implement-able chunks of integration. Compared to reporting on the progress of implementation at the level of the entire integration, this finer level of reporting allows for a more concrete and quantitative tracking of the implementation.

The mini-integrations can be listed and their relationship to the larger picture of the complete integration detailed in a chart, and then an accurate picture of the overall progress of implementation can easily be drawn from reporting on the progress of the mini-integration tasks.

Ownership

Even when working with multiple teams, assign ownership to a single person across the integration. This person's job is to ensure that the single thread is established early, that the teams are working according to the guidelines in this document, and that the incremental build/test cycle across (a) mini-integrations, (b) scalable process testing and (c) representative vs. complete environments is in sync across the various teams.

Documentation

Clearly identify formats and approach

Decide on a clear path for execution and document clearly any formats that will be used when there are multiple teams working on integration. The most common example is where a WebSphere Product Center team is working on exporting data from WebSphere Product Center, and a customer or SI team is working on uploading that data into a

destination system. Do not begin work without specs for the common format, and keep this documentation up to date daily. This is an absolute requirement that the project manager must enforce.

This approach is not inconsistent with using representative environments and doing mini-integrations. Both teams should build and test incrementally to ensure steady, visible progress.

Top Ten Guidelines for WebSphere Product Center Integrations

Use clear and common terminology to describe integrations

All implementations should use the dimensions identified in the section Integration Dimensions.

Re-usability

The key to scaling will lie in learning from prior integrations and packaging the integrations keeping the re-usability principles described in the Re-usability section in mind.

Visibility

Establish an overall metric for reporting progress and provide a clear status update every few days at worst to the project manager.

Mini-integrations

Slice up the complexity of a large integration according to various dimensions (catalogs, attributes) that make sense for that integration. Focus on one mini-integration at a time and tie directly to the visibility metrics.

Representative vs. complete environments

Maintain a representative environment that is easy to debug and test with. Move to the complete environment only when there is confidence in the validity of scripts and specs. Tie this in with visibility metrics.

Scalable process testing

Test all jobs with small data sets to check correctness before rolling out to complete data sets. Tie this in with visibility metrics.

Performance

Without worrying about correctness of logic or formatting, run some performance tests early in the development cycle and regularly thereafter to identify issues.

Establish single thread early

Especially in complex integrations that might require multiple hops, multiple protocols, or non-standard means, establish a working single thread of integration early.

Design specs and documentation

Define and document a clear path for execution and clearly document any formats that will be used, especially when there are multiple teams working on integration.

Single owner

Even when working with multiple teams, assign ownership to a single person across the integration.

EAI Platform Integrations

Approach

Generic communications format

Whenever possible, a generic communication format should be designed or re-used from a previous project. The more general the format, the more systems can be involved in the integration without special re-working of formats required in order for all systems to communicate with each other. Of course, there can be trade-offs in performance the more general a format becomes and thus the right format for one project may not be the ideal choice for another. The Integration Dimensions should still be taken into account when determining a specific format to use.

Content mappings

As much as possible, mappings between the in- WebSphere Product Center content model and the model seen through the communication format should be done through dynamically-updateable means. Again, based on an investigation of the Integration Dimensions certain project needs may dictate that the creation of these mappings cannot be fully dynamically-updateable – due to a high priority placed on absolute maximum throughput of processing, for instance. One manner of doing this involves using category trees (representing an XML structure, for instance) with a single-node spec related to them which can indicate the spec node path of the attribute a particular node of the category tree maps to in the in- WebSphere Product Center content model. A recursive processing script can then be utilized to process the mapping of an item into an XML file based on this category tree and its defined mappings, and can even cater for nested multi-occurrences with a little effort.

Additional advantages

Information translation / transformation

Systems involved in integration should have no need to handle of

themselves the information or content restrictions and requirements of other systems in the integration. EAI platforms can be readily utilized to handle this content translation and transformation. For instance, while WebSphere Product Center stores a FLAG's value as "TRUE" or "FALSE", a system with which we integrate may store the value as "Y" or "N". EAI platforms can be used to make these translations such that WebSphere Product Center can always send TRUE/FALSE and assume that it will be sent TRUE/FALSE, while integrated systems can always send and assume to be sent Y/N. This ensures that down the line, if more systems are involved in integration, no re-coding will be required in light of these additional systems.

Client understanding

Because we can re-use a platform with which a client is likely to already be familiar, the client will gain the added confidence that the integration makes use of known functionality – i.e. the EAI platform. Additionally, if a client-specific communication format already exists and is re-used for the WebSphere Product Center integration, client-side developers need little to no additional training to understand the communication format that WebSphere Product Center will be mapping to.

Communications flexibility and reliability

Most EAI platforms have native functionality to allow communications to occur over a variety of protocols and also to ensure that communications are delivered by means of brokering. This allows WebSphere Product Center to focus on simply generating the necessary document to communicate and not have to worry about supporting potentially different means of communicating this document to a variety of systems nor needing to worry about tracking whether the document was received by each system or not – these concerns become concerns of the EAI layer and platform, and WebSphere Product Center need simply be aware of them from an overall integration thread perspective.

Notices

IBM may not offer the products, services, or features discussed in this document in all 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 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.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Burlingame Laboratory
Director IBM Burlingame Laboratory
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not necessarily 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 may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may 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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

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

IBM
the IBM logo
AIX
CrossWorlds
DB2
DB2 Universal Database
Domino
Lotus
Lotus Notes
MQIntegrator
MQSeries
Tivoli
WebSphere

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

MMX, Pentium, and ProShare are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

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

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

This product includes software (Axis, Jakarta Commons Collection, Jakarta Commons DBCP, Jakarta Commons Pool, Jakarta Commons Regular Expression, Log4J, Regexp, Xalan, Xerces, XML4J) developed by the Apache Software Foundation (http://www.apache.org/).

CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or

bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

a. You must give any other recipients of the Work or Derivative Works a copy of this License; and

b. You must cause any modified files to carry prominent notices stating that You changed the files; and

c. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those

notices that do not pertain to any part of the Derivative Works; and

d. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

CUP Parser Generator Copyright Notice, License, and Disclaimer
Copyright 1996-1999 by Scott Hudson, Frank Flannery, C. Scott Ananian

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both the copyright notice and this permission notice and warranty disclaimer appear in supporting documentation, and that the names of the authors or their employers not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

The authors and their employers disclaim all warranties with regard to this software, including all implied warranties of merchantability and fitness. In no event shall the authors or their employers be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

JLEX COPYRIGHT NOTICE, LICENSE AND DISCLAIMER.

Copyright 1996-2003 by Elliot Joel Berk and C. Scott Ananian

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both the copyright notice and this permission notice and warranty disclaimer appear in supporting documentation, and that the name of the authors or their employers not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

The authors and their employers disclaim all warranties with regard to this software, including all implied warranties of merchantability and fitness. In no event shall the authors or their employers be liable for any special, indirect or consequential damages or any damages

whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

Java is a trademark of Sun Microsystems, Inc. References to the Java programming language in relation to JLex are not meant to imply that Sun endorses this product.