

Agile

Version e6.1

ORACLE®

Oracle® Agile

Engineering Data Management

Agile e6.1.2.2 Batch Client

Part No. E27832-01

April 2012

Copyright and Trademarks

Copyright © 1995, 2012, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third party content, products and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third party content, products or services.

CONTENTS

Copyright and Trademarks	iii
Preface	vi
Introduction.....	1
Hardware and Software Requirements	3
Installing and Configuring the Batch Client	5
Installing the Agile e6.1.2.2 Batch Client	5
Configuring the Agile e6.1.2.2 Batch Client	6
Batch Client Settings	7
Batch Client Solution	7
Java Wrapper Settings	7
Agile e6.1.2.2 Batch Client Properties	8
Secure User Credentials	9
Shutdown Batch Client Properties	9
Enable the Old Batch Solution	10
Configuring the Batch Client for Executing a Simple LogiView Procedure	10
System Architecture	13
Use Cases.....	15
Scenario Mode.....	15
Run-time Process (Batch Job Loop).....	18
Shutdown Agile e6.1.2.2 Batch Client	20
ECI Server Mode	21
ECI Server Settings	23
References	25
Office Suite PDF Service	25
AutoVue Offline Metafile Cache Service	25
Why to check-in a Metafile into the Vault?.....	25
In Which Vault will the Metafile be Checked-In?.....	25
Requirements and Assumptions	25
Order Queue.....	26
Order Positions.....	26
Creating an Offline Metafile Cache Order	27

Adding All Files of a Document	27
Adding a Specific File	28
Batch Process	29
Installing the Offline Metafile Cache Service	29
Runtime	32

Preface

The Oracle documentation set includes Adobe® Acrobat™ PDF files. The [Oracle Technology Network \(OTN\) Web site](http://www.oracle.com/technology/documentation/agile.html) (<http://www.oracle.com/technology/documentation/agile.html>) contains the latest versions of the Oracle Agile EDM PDF files. You can view or download these manuals from the Web site, or you can ask your Agile administrator if there is an Oracle Documentation folder available on your network from which you can access the documentation (PDF) files.

Note To read the PDF files, you must use the free Adobe Acrobat Reader™ version 7.0 or later. This program can be downloaded from the [Adobe Web site](http://www.adobe.com) (<http://www.adobe.com>).

Note Before calling Agile Support about a problem with an Oracle Agile EDM manual, please have the full part number ready, which is located on the title page.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

Readme

Any last-minute information about Oracle Agile EDM can be found in the Release Notes file on the [Oracle Technology Network \(OTN\) Web site](http://www.oracle.com/technology/documentation/agile_eseries.html) (http://www.oracle.com/technology/documentation/agile_eseries.html)

Agile Training Aids

Go to the [Oracle University Web page](http://www.oracle.com/education/chooser/selectcountry_new.html) (http://www.oracle.com/education/chooser/selectcountry_new.html) for more information on Agile Training offerings.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Introduction

Note The "old" EDB_BATCH mechanism of the PLM server is no more recommended and the Batch Client replaces the "old" batch solution of the PLM server.

The Agile e6.1.2.2 Batch Client allows running the batch scenarios implemented in [Groovy](#) to support the typical batch use cases.

The Agile e6.1.2.2 Batch Client supports two different modes:

1. **Scenario Mode** – In the scenario mode the Batch Client starts a defined scenario which interacts with the PLM Server to execute the batch use case.
2. **ECI Server Mode** – In this mode the Batch Client provides an ECI tunnel to the PLM Server and standard features like file access to the File Server.

For detailed information refer to the corresponding use case chapters.

Note The "old" batch solution is still supported but you must configure the security level (see Settings) to enable it.

Chapter 2

Hardware and Software Requirements

The Agile e6.1.2.2 Batch Client is available for all server platforms of the Agile e6.1.2.2 release.

The Agile e6.1.2.2 Batch Client has no Installer software. A manual installation and configuration is necessary.

Installing and Configuring the Batch Client

Installing the Agile e6.1.2.2 Batch Client

The Agile e6.1.2.2 Batch Client is delivered as a ZIP archived file.

The ZIP archive is contained in the *batchclient.zip* installation package located in the packages folder of the installation medium.

1. Copy the Batch Client installation package (*batchclient.zip*) from the packages folder of the installation medium into a temporary directory

Example:

```
copy y:\packages\batchclient.zip c:\temp
```

2. Extract the installation package; this unpacks the Agile e6.1.2.2 Batch Client bundle ZIP archive into the axalant/bin/java sub directory of your temporary directory.

Example:

```
unzip c:\temp\batchclient.zip -d c:\temp
```

3. Extract the Agile e6.1.2.2 Batch Client bundle into the installation directory (Example: C:\Program Files). The Agile e6.1.2.2 Batch Client files are unpacked into the batchclient directory which is automatically created.

Example:

```
unzip c:\temp\axalant\bin\java\batchclient.zip -d "c:\Program Files\Agile_e6"
```

The Agile e6.1.2.2 Batch Client bundle contains the following directories:

Batch Client Root	(C:\Program Files\Agile_e6\batchclient)
axalant	
bin	
ia64-hp-hpux11.31	(HPUX FMS-Client binary)
ia32-linux-sles11	(Linux FMS-Client binary)
intel-ms-nt5.1	(Windows 32 bit FMS-Client binaries)
java	(Java archives)
rs6000-ibm-aix6.1	(IBM FMS-Client binary)

sparc-sun-solaris10	(Sun FMS-Client binary
cmd	(Windows scripts directory)
dmp	(Example loader files)
ini	(Configuration files)
scripts	(UNIX scripts directory)
examples	(Example batch files)
ext	
bin	
ia64-hp-hpux11.31	(external binaries)
ia32-linux-sles11	(external binaries)
intel-ms-nt5.1	(external binaries)
java	(external Java archives)
rs6000-ibm-aix6.1	(external binaries)
sparc-sun-solaris10	(external binaries)
tmp	(Logging directory)

Configuring the Agile e6.1.2.2 Batch Client

- The Java environment has to be set to a Java 5 or higher version. For the MS Windows Batch Client adapt the BatchClient.cmd shell script.

Example:

```
set JAVA_HOME="C:\Program Files\Java\jre1.5.0_16"
```

- For the UNIX environment adapt the Batch Client shell script.

Example:

```
JAVA_HOME=/usr/local/java/jre1.5.0_15
```

Chapter 4

Batch Client Settings

This chapter describes the several settings related to the Java Wrapper framework of the Agile e6.1.2.2 Batch Client in the Scenario mode and ECI mode.

Batch Client Solution

The Agile e6.1.2.2 Batch Client uses the Java Wrapper as its run-time frame work.

Java Wrapper Settings

The Java Wrapper needs a configuration file which is described here.

The Java Wrapper supports two different types of shutdown modes.

1. **Simple type:** The Java application controlled by the Java Wrapper will be configured and simply killed on shutdown, this mode is used for the Java Daemon
2. **Start / Stop type:** This mode allows to define which Java application should be controlled and which Java application should be used to shut down the controlled application, this mode is used for a TOMCAT for instance.

The Agile e6.1.2.2 Batch Client uses the Start / Stop mode to shut down the Agile e6.1.2.2 Batch Client and the connected PLM server.

Configurations

Here is a short description of the settings which may be adopted to implement a custom batch solution.

Java Application Settings

This section in the configuration defines the controlled Java application and its command line parameters.

```
# Application parameters. Add parameters as needed starting from 1.
wrapper.app.parameter.1=com.agile.testclient.BatchClientApp
wrapper.app.parameter.2=3
wrapper.app.parameter.3=-p%EP_ROOT%/examples/LgvLoop.properties
wrapper.app.parameter.4=-i%EP_ROOT%/examples/LgvLoop.groovy
wrapper.app.parameter.5=-T%EP_ROOT%/axalant/ini/testclient.properties
```

The next section in the configuration defines the Java application and its command line parameters to shutdown the controller application.

```
# Application parameters to stop the batch client
wrapper.app.parameter.6=com.agile.testclient.BatchClientApp
wrapper.app.parameter.7=true
wrapper.app.parameter.8=3
wrapper.app.parameter.9=-p%EP_ROOT%/examples/LgvLoopStop.properties
wrapper.app.parameter.10=-i%EP_ROOT%/examples/LgvLoopStop.groovy
wrapper.app.parameter.11=-T%EP_ROOT%/axalant/ini/testclient.properties
```

MS Windows Service Settings

This section within the configuration file defines the MS Windows Service settings. Here, the name of the server can be defined and some extended settings for services.

```
*****
# Wrapper NT Service Properties
*****
# WARNING - Do not modify any of these properties when an application
using this configuration file has been installed as a service.
# Please uninstall the service before modifying this section. The
service can then be reinstalled.

# Name of the service
wrapper.ntservice.name=AgilePLM_Batch_Service
# Display name of the service
wrapper.ntservice.displayname=AgilePLM_Batch_Service
# Description of the service
wrapper.ntservice.description=Batch Service for Agile PLM
# Service dependencies. Add dependencies as needed starting from 1
wrapper.ntservice.dependency.1=
# Mode in which the service is installed. AUTO_START or DEMAND_START
wrapper.ntservice.starttype=AUTO_START
# Allow the service to interact with the desktop.
wrapper.ntservice.interactive=false
```

Agile e6.1.2.2 Batch Client Properties

The used Agile e6.1.2.2 Batch Client scenario and the corresponding property file are defined as command line parameters in the Java Wrapper configuration file as you can see above.

This configuration file has to be adapted to your environment.

Note The Batch Client bundle provides an example which has to be adapted to your environment.

```
# ECI connection
```



```
host=<JAVA-DAEMON-HOSTNAME>
port=<JAVA_DAEMON-PORT>
env=<PLM-APPLICATION>
scope=BATCH

# Directories
varenv.ep_root=<ROOT_DIRECTORY_OF_THE_BATCH_CLIENT>
varenv.axalant_root=<AXALANT_ROOT_DIRECTORY_OF_THE_BATCH_CLIENT>

# Platform
varenv.EP_MACH=<MACHINE_PLATFORM_NAME>

# Security settings
KeyStoreFile=cwallet.sso
KeyAlias=C=DE,ST=Baden,L=Karlsruhe,O=Oracle,OU=Agile PLM,CN=PLM
TicketKey=RSA-PUBLIC-
BASE64:D8VyVSBxMfgXcZ8AXhOSZMI6Agh4IVQdU49RgszulDGm+z7dQbSIBYRWbpdfsYgP
s4GmjQL//tVYLDttGvLw6n2uN4/iwLFjGO93PtzGuX7TGqWZQkgXR4pGw7M2KjXMDNN/nIL
9rwlWyRKYL0zHZka1ZMgaopEuwPRmsqoQ2lU=

# PLM Client
client1=<USER_NAME>,<PASSWORD>,<SCENARIO_CLASS_NAME>
```

Secure User Credentials

The User credentials are stored in the Agile e6.1.2.2 Batch Client properties file. To protect the user password it can be encrypted by the epkeytool application.

```
epkeytool -encryptpwd -keyStore cwallet.sso -keyAlias
"C=DE,ST=Baden,L=Karlsruhe,O=Oracle,OU=Agile PLM,CN=PLM" -pass
USER_PASSWORD
```

The output is:

```
RSA-PUBLIC-
BASE64:P9joyIBH7ATx6pW5Gxw9RGHxeh4mUWLzEHCmd/o3xhVVmje1mf091cTF+Pum71Px
0da7OTyPkt0M1ubjdL8zdgjaiHx6RczOA+grCgVnNnpAdc=
```

Copy the encrypted password into the scenario properties file as user password.

Example:

```
# PLM Client
client1=USER_NAME,RSA-PUBLIC-
BASE64:P9joyIBH7ATx6pW5Gxw9RGHxeh4mUWLzEHCmd/o3xhVVmje1mf091cTF+Pum71Px
0da7OTyPkt0M1ubjdL8zdgjaiHx6RczOA+grCgVnNnpAdc=,com.agile.LGVCall
```

Shutdown Batch Client Properties

The shutdown properties are the same as the controlled Agile e6.1.2.2 Batch Client properties

for the batch use case. There is only another scenario scripts used.

Enable the Old Batch Solution

The "old" batch solution of the PLM server is still available, but due to security risks this solution is not recommended.

To enable the EDB_BATCH mechanism, the security level has to be decreased to unrestricted.

The security level can be configured in *the <environment>.xml file located in the <ep_root>/init directory.*

Example:

```
<IPC
AbsEciUrl="eci://localhost:19997" SecurityLevel="unrestricted"
TicketTimeout="600">
</IPC>
```

Note The security level setting controls the security settings of the ECI connection also. A client needs no authentication to connect to an unrestricted ECI server. So ensure that this setting is not effective for PLM servers which allow to be contacted via ECI.

Configuring the Batch Client for Executing a Simple LogiView Procedure

The standard Agile e6.1.2.2 Batch Client bundle is pre-configured for executing a LogiView Start/Stop scenario and cannot be used out of the box for a starting a simple LogiView scenario. Especially, the wrapper configuration file needs a different wrapper java *MainClass* and different wrapper application parameters.

The Agile e6.1.2.2 Batch Client requires the following configuration files / start scripts:

- <ep_root>\axalant\ini\batch_wrapper.conf (can be renamed)
- <ep_root>\axalant\ini\testclient.properties (can be renamed)
- <ep_root>\examples\LGVCall.properties (can be renamed)
- <ep_root>\examples\LGVCall.groovy (can be renamed)
- <ep_root>\axalant\cmd\batchclient.cmd (Windows) (can be renamed)

or

- <ep_root>/axalant/scripts/batchclient (Unix) (can be renamed)

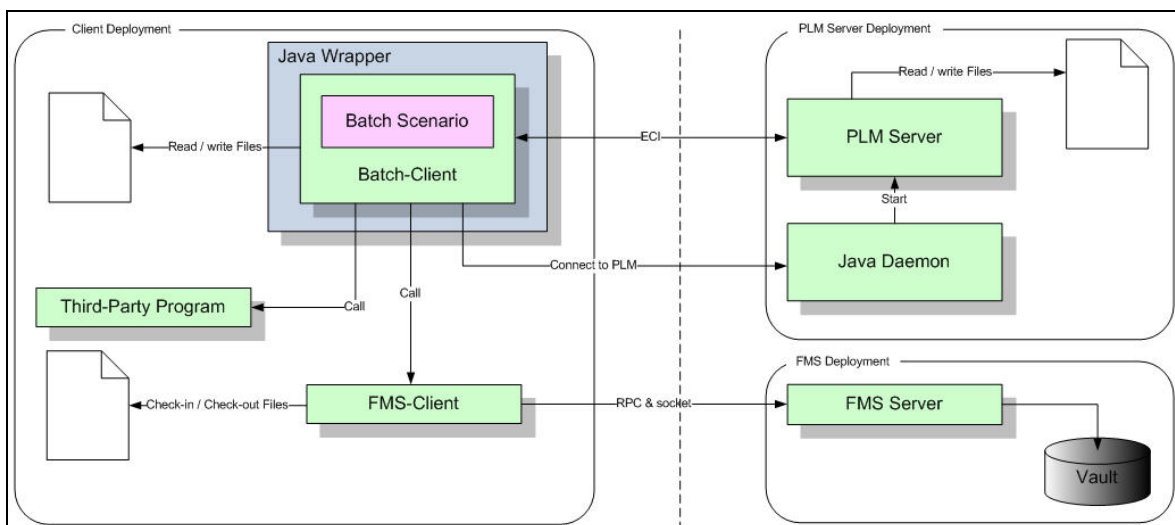
To enable the batch client for executing a simple LogiView procedure, the Agile e6.1.2.2 Batch Client bundle is delivered with pre-configured examples which call a simple LogiView (LGV) procedure. The example files are located in the installation folder as follows:

Batch command script	axalant/cmd/Simple.cmd
Wrapper configuration	axalant/ini/batch_simple_wrapper.conf
Batch scenario	axalant/examples/LGVCall.groovy
Batch configuration	axalant/examples/LGVCall.properties

Chapter 5

System Architecture

The Agile e6.1.2.2 Batch Client is a special version of the Java-Client without any UI. The Batch Client uses the Java Daemon to start the PLM Server and communicates via ECI with the PLM Server. The Agile e6.1.2.2 Batch Client supports client side callables which are used by the PLM Server to start client side activities like the startup of an external program or the FMS Client to check-in / check-out files from the File Server. The standard features to exchange files with the PLM Server are as well supported.



The Agile e6.1.2.2 Batch Client runs a batch scenario which is implemented in Groovy to control the batch use case. The most batch use cases are designed as a loop which uses a job table to determine if a job is present or if the process should wait for a new job.

In "old" batch designs that loop is implemented in LogiView and sometimes there are external programs used to implement the sleep functionality to wait some seconds for a new batch job.

That batch job loop has to be moved into the batch control scenario on client side for some reasons.

- The Agile e6.1.2.2 Batch Client provides the "sleep" functionality to wait for a new batch job.
- The Agile e6.1.2.2 Batch Client communicates via ECI with the PLM Server and checks for a new job to help keep the connection alive.
- The PLM Server is started by the Java Daemon and the Java Daemon communicates with the PLM Server for controlling purposes. If the PLM Server is in a LogiView loop the communication with the Java Daemon is not possible.

The Agile e6.1.2.2 Batch Client uses the **Java Wrapper** like the Java Daemon to support being installed as a service.

Chapter 6

Use Cases

The Agile e6.1.2.2 Batch Client supports two modes:

1. **Scenario Mode:** to implement the Batch Use Cases.
2. **ECI Server Mode:** to provide an ECI tunnel and client access like file server access.

Scenario Mode

The scenario mode runs a Groovy script which implements the batch process. This chapter explains how to design and write a script to implement a Batch use case.

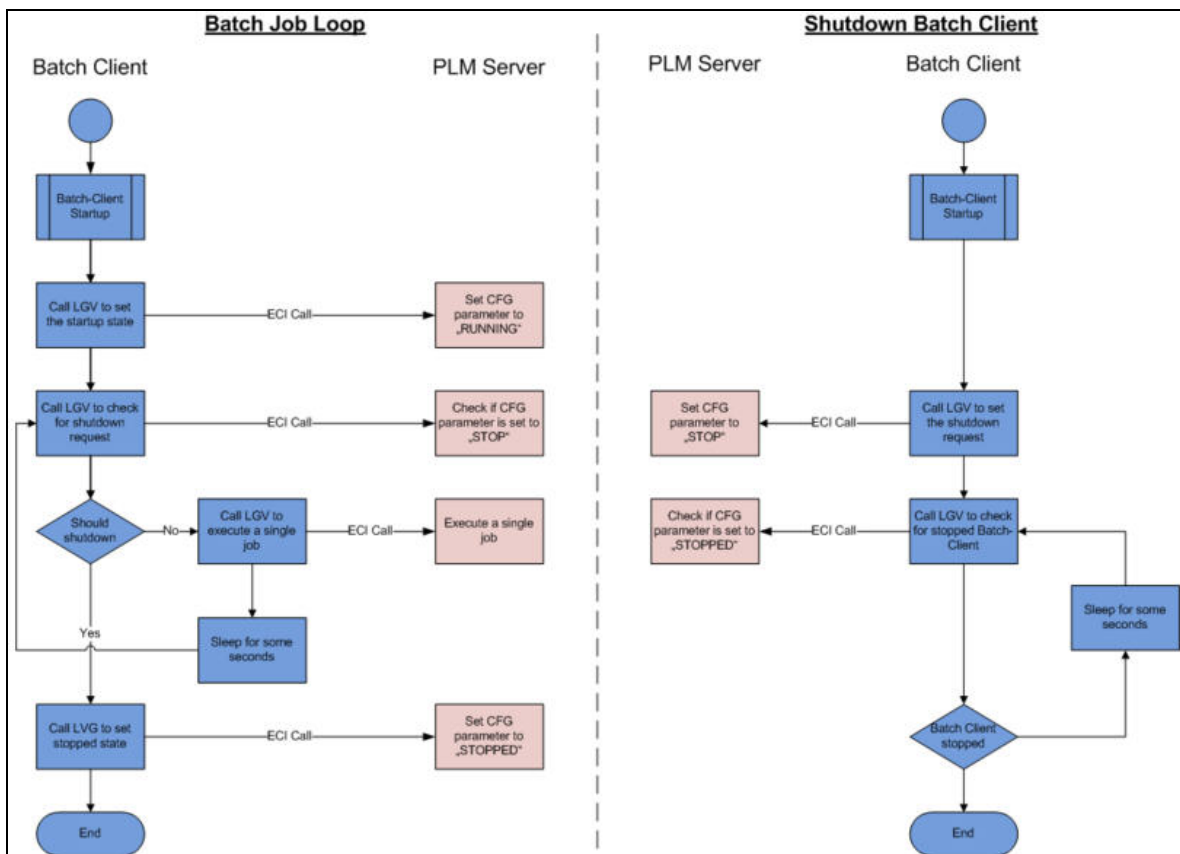
Recommended Design of a Batch Control

Note With the Agile e6.1.2.2 Batch Client bundle, a set of batch control scripts are available in the **examples** folder. The LogiView procedures of the examples are available as loader files. In this example the synchronization of the run-time process and the shutdown process is carried out by using a PLM configuration parameter within the database. But you can use any information source which is accessible by both processes, for example, a shutdown job in the job table, a local file, a record in the database, and so on.

The Agile e6.1.2.2 Batch Client bundle comes with a set of batch control scripts. These are in the *examples* folder.

The *LgvLoop.groovy* script can be used as a template for each job table based batch use case. The script uses a set of LogiView procedures to check, control, and execute the batch functionality. Like the TOMCAT Server or the Oracle Application Server, the Agile e6.1.2.2 Batch Client provides a run-time process and a shutdown process. In the provided template, the Agile e6.1.2.2 Batch Client LogiView backbone uses a configuration parameter to report the current state of the batch process and to control the batch process.

The following flowchart shows a command flow of the run-time and the shutdown process.



The Java wrapper executes the run-time process during start-up and calls the shutdown process to shutdown the run-time process. The run-time process sets the current state of the batch job by using the corresponding LogiView procedures and executes the batch job with the central LogiView procedure. To detect a shutdown request the run-time progress checks if a shutdown request is available by calling a special LogiView procedure.

In this example, there are several LogiView procedures to report the current state of the run-time process:

- **setRunning** called in the start-up phase to report that the batch job is running
- **setStopped** called in the shutdown phase to report that the batch job has been stopped
- **shouldShutdown** to detect a shutdown request
- **executeOperation** to do the real work of the batch job

First the run-time process calls **setRunning** during the start-up (you may use another LogiView procedure like **isRunning** to detect if there is already a batch job running to prevent two batch jobs at the same time). Then the run-time process checks if there is a shutdown request available by calling **shouldShutdown**. If not, then the **executeOperation** LogiView procedure is called to execute a batch job. If a shutdown request is available, the run-time process sets the "stopped" state by calling **setStopped** and ends the run-time scenario.

The shutdown process uses the following LogiView procedures:

- **shutdown** to create a shutdown request
- **isStopped** to detect that the run-time process has been stopped

The shutdown process waits until the run-time process has been stopped before the shutdown scenario ends.

The Java wrapper waits until the shutdown process ends or until the shutdown timeout has been reached. The shutdown timeout can be configured in the Java wrapper configuration file.

Time-Out Value Information

wrapper.jvm_exit.timeout

Allowed time, in seconds, between the time JVM reports that the JVM is stopped and the time that the JVM process actually terminates.

- Default is set to 15 seconds.
- 0 seconds means no time out.

In a normal operation, the Java side of the Wrapper executes the file `System.exit` when it has completed its JVM shutdown cycle and is ready to exit. When this timeout is triggered, a message like the following is logged.

```
wrapper | Shutdown failed: Timed out waiting for the JVM to terminate.  
wrapper | Java Virtual Machine did not exit on request, terminated
```

If the application has registered its own shutdown hook, which takes some time to complete, you could experience timeouts waiting for the JVM process to terminate. To avoid this problem, it may be necessary to extend the timeout to give the application's shutdown hook time to execute to completion. Be aware that, as a rule, shutdown hooks should always complete almost instantly.

Example: `wrapper.jvm_exit.timeout=5`

Note For further information about the `wrapper.jvm` please refer to:
<http://wrapper.tanukisoftware.org/doc/english/prop-jvm-exit-timeout.html>

Run-time Process (Batch Job Loop)

The following is the Groovy script of the *Batch Job Loop*.

```
/**
 * Batchclient example script to demonstrate the basic implementation of
 * a batch service.
 * $Id: LgvLoop.groovy,v 34.2 2009/06/21 15:51:21 schwera1 Exp $
 */

package com.agile.LgvLoop;
import com.agile.testclient.*;

/**
 * Each scenario needs to extend the Scenario class.
 * The Scenario class provides the execution environment.
 */

public class LgvLoop extends Scenario {

    /**
     * The run() method is called by the framework to start the scenario.
     */

    public void run() {
        System.out.println("Starting BatchExample/LgvProd");

        getTestClient().callNativeUsx("lgv_nosel_run", "BatchExample/setRunning"
        );
        while(!shouldShutdown()) {
            getTestClient().callNativeUsx("lgv_nosel_run", "BatchExample/executeOper
            ation");
            sleepSeconds(10);
        }
        getTestClient().callNativeUsx("lgv_nosel_run", "BatchExample/setStopped"
        );
        System.out.println("Done BatchExample/LgvProd");
    }

    /**
     * Checks if the batch client should shutdown.
     * The method checks the DTV default "EDB-BATCH-CONTROL" to receive a
     * shutdown request.
     * @return true if the batch client should shutdown else false
     */
    private boolean shouldShutdown() {
        int result =
        getTestClient().callNativeUsx("lgv_nosel_run", "BatchExample/shouldShutd
        own");
    }
}
```

```
        return result == 1;
    }
}
```

The script uses the following methods to implement the run-time process:

- **callNativeUsx** to call LogiView procedures
- **sleepSeconds** to sleep for some seconds to wait
- **println** to print information into the log file

Therefore the script calls 4 LogiView procedures to set the current process state, to execute the batch job, and to check for shutdown requests. Here are the LogiView procedures used by the batch use case:

BatchExample/setRunning

This LogiView procedure sets the batch process control variable (here the configuration parameter *EDB-BATCH-EXAMPLE-CONTROL*) to the value "RUNNING".

```
EDB_BATCH_VALUE = "RUNNING"
update(EDB_BATCH_VALUE)
where("T_CFG_DAT.EDB_ID" = "EDB-BATCH-EXAMPLE-CONTROL")
exec_update()
```

The configuration parameter *EDB-BATCH-EXAMPLE-CONTROL* can be used to check if the Agile e6.1.2.2 Batch Client is online or not. If your batch process does not allow running more than one Agile e6.1.2.2 Batch Client you can add another LogiView call to check the current state of the batch process. The LogiView procedure **BatchExample/isRunning** checks if there is already another batch process active so you can prevent starting a new Agile e6.1.2.2 Batch Client by checking the current state and implement your batch script to exit in that case.

BatchExample/shouldShutdown

The LogiView procedure that checks the configuration parameter *EDB-BATCH-EXAMPLE-CONTROL* for a shutdown request (you can also use different variables to reflect the current state and to control the batch process). LogiView reports a shutdown request if the configuration parameter *EDB-BATCH-EXAMPLE-CONTROL* is set to "STOP" or if the configuration parameter is not present in the system.

```
select(EDB_BATCH_VALUE)
where("T_CFG_DAT.EDB_ID" = "EDB-BATCH-EXAMPLE-CONTROL")
RES = exec_select(1)
if (RES == 1
    if (EDB_BATCH_VALUE == "STOP")
        exit()
    endif
endif
```

```
else
    exit()
endif
```

BatchExample/executeOperation

This is the LogiView procedure that does the real work. The procedure checks if there is a batch job available and executes that job or it returns the message that the Agile e6.1.2.2 Batch Client can sleep for some seconds and try again later. You can also use these messages to trace the actions of the batch job in the client trace.

BatchExample/setStopped

The batch process checks in every loop if a shutdown request is pending. Before the Agile e6.1.2.2 Batch Client shutdown, the LogiView procedure *BatchExample/setStopped* is called to set the current state of the batch process.

All sample LogiView procedures are available as loader files within the Agile e6.1.2.2 Batch Client Bundle.

Shutdown Agile e6.1.2.2 Batch Client

This is the shutdown script used by the Java Wrapper to shutdown the Agile e6.1.2.2 Batch Client.

```
/**
 * Batchclient example script to demonstrate the shutdown process.
 *
 * $Id: LgvLoopStop.groovy,v 34.2 2009/06/21 15:51:21 schwera1 Exp $
 */

package com.agile.LgvLoop;
import com.agile.testclient.*;

/**
 * Each scenario needs to extend the Scenario class.
 * The Scenario class provides the execution environment.
 */
public class LgvLoop extends Scenario {

    /**
     * The run() method is called by the framework to start the
     * scenario.
     */
    public void run() {
        System.out.println("Send shutdown request");

        getTestClient().callNativeUsx("lgv_nosel_run", "BatchExample/shutdown");
    }
}
```

```
while(!isStopped()) {
    System.out.println("Wait for stopped batch client");
    sleepSeconds(1);
}

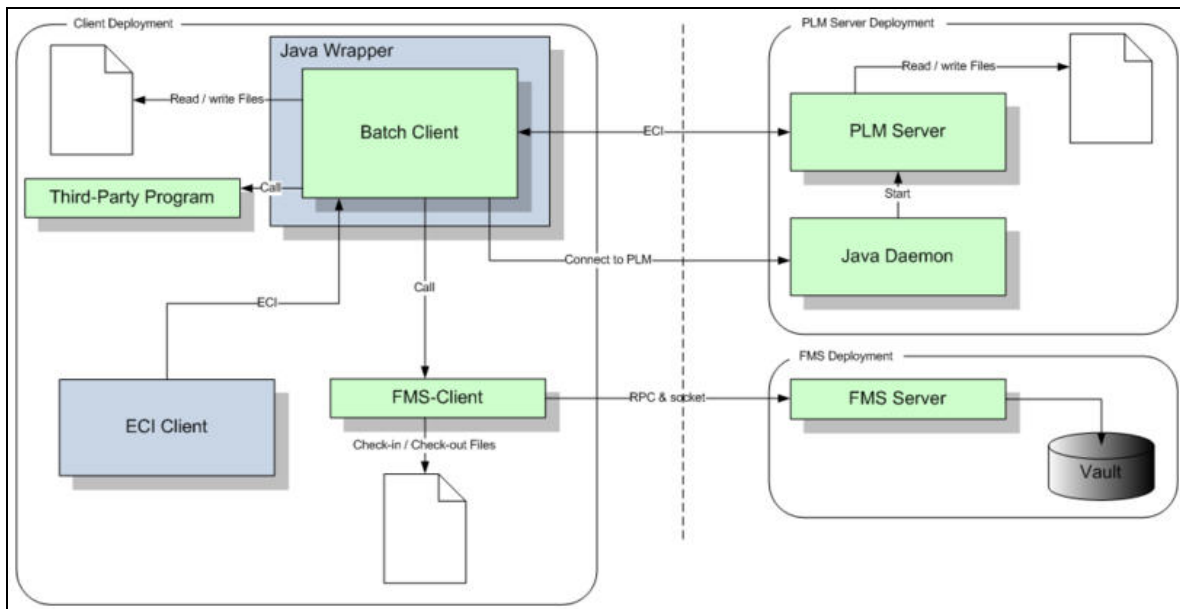
System.out.println("Done");
}
/**
 * Checks if the batch client has stopped.
 * The method checks the DTV default "EDB-BATCH-CONTROL".
 *
 * @return true if the batch client has stopped else false
 */
private boolean isStopped() {
    int result =
getTestClient().callNativeUsx("lgv_nosel_run", "BatchExample/isStopped")
;
    return result == 1;
}
}
```

ECI Server Mode

The Agile e6.1.2.2 Batch Client can be used as an ECI Server which tunnels the ECI calls of an ECI client to the PLM Server. Additionally to that tunnel functionality, the Agile e6.1.2.2 Batch Client provides some callables to start external programs or to interact with the FMS Client for file operations.

So a "simple" ECI client can be enabled to check-in or check-out files from the file vault without an interactive Java Client.

The following picture shows an overview of a Agile e6.1.2.2 Batch Client in ECI server mode.



The Agile e6.1.2.2 Batch Client in ECI Server mode uses the Java Wrapper to enable the service features, as it does in the Scenario mode.

The external ECI client can be, for example, a CAD integration which normally uses a Java Client to gain access to the file within a vault.

The Agile e6.1.2.2 Batch Client allows multi connections from several ECI clients.

To shutdown the Agile e6.1.2.2 Batch Client, the ECI client can call the *eci_stb_edb* ECI function.

Wrapper Configuration Settings

To start the Agile e6.1.2.2 Batch Client in ECI Server mode, some wrapper configuration settings have to be changed as follows:

Java Environment

- The Java environment has to be set to a Java 5 or higher version. In the MS Windows, adapt the **ECIServer.cmd** shell script for the Agile e6.1.2.2 Batch Client in ECIServer mode.

Example: `set JAVA_HOME="C:\Program Files\Java\jre1.5.0_16"`

- For a UNIX environment adapt the ECI Server shell script.

Example: `JAVA_HOME=/usr/local/java/jre1.5.0_15`

ECI Server Settings

The configuration of the Agile e6.1.2.2 Batch Client in ECI server mode is similar to the Agile e6.1.2.2 Batch Client in scenario mode with some small differences which are described here.

The ECI server mode does not need a scenario script but the ECI server settings have to be configured.

Here are the additional settings within the Agile e6.1.2.2 Batch Client properties file:

```
#
# ECI connection
#
host=<JAVA-DAEMON-HOSTNAME>
port=<JAVA_DAEMON-PORT>
env=<PLM-APPLICATION>
scope=BATCH

# ECI Tunnel
#
varenv.EciServer.Port=44444
varenv.EciServer.Encoding=UTF-8
varenv.EciServer.Secure=true
varenv.EciServer.StartupImmediately=true
varenv.EciServer.ConnectionDelay=100
varenv.EciServer.allowHost.1=localhost
```

- The **Port** setting defines the port of the ECI server for client connections
- The **Encoding** should be set to UTF-8
- The **Secure** setting decides if the ECI should be secured or not. The value **true** is recommended, it ensures that ECI client must authenticate with user/password to access the ECI server.
- The **StartupImmediately** setting defines if the PLM server should started immediately with the start-up of the ECI server or on demand.
- The **ConnectionDelay** setting defines the minimum delay between two subsequent connection requests to prevent deny of service attacks.
- The **allowHost** list contains all allowed client host names which are accepted for ECI connections.

Additional settings:

```
varenv.EciServer.denyHost.1=<hostname_1>
varenv.EciServer.denyHost.2=<hostname_2>
varenv.EciServer.CallablePackage.1=<custom_callable_package>
```

- The **denyHost** list allows defining client host names which are explicitly declined when they try to open an ECI connection.
- The **CallablePackage** list can be used to add custom callable packages

Once the necessary wrapper configuration settings have been modified, you can use **EciServer.cmd – c** to start the Agile e6.1.2.2 Batch Client in console mode and check with the command **eci_test** if the Agile e6.1.2.2 Batch Client works properly.

References

Office Suite PDF Service

For complete information on the Office Suite PDF Service, please refer to the Administration Manual > *Office Suite – PDF Generator Installation*.

AutoVue Offline Metafile Cache Service

For complete information on the AutoVue Offline Metafile Cache Service, please refer to the Administration Manual.

Why to check-in a Metafile into the Vault?

The local cache on the AutoVue Server may be cleaned or crashed. In this case, the AutoVue server can download all native files and the metafile from the EDM system and does not need to create a new metafile.

Note The conversion process from the native format to the AutoVue metafile format is a time consuming process.

In Which Vault will the Metafile be Checked-In?

The system uses the DFM feature to specify the target vault for metafiles.

The integration sets the following properties:

- File Type: *META*
- File Format: Same as the base file or *NAT* if not set
- Creation System: *AUTOVUE*

Requirements and Assumptions

- Batch-Client as service
- JVue bean as control interface of the AutoVue server

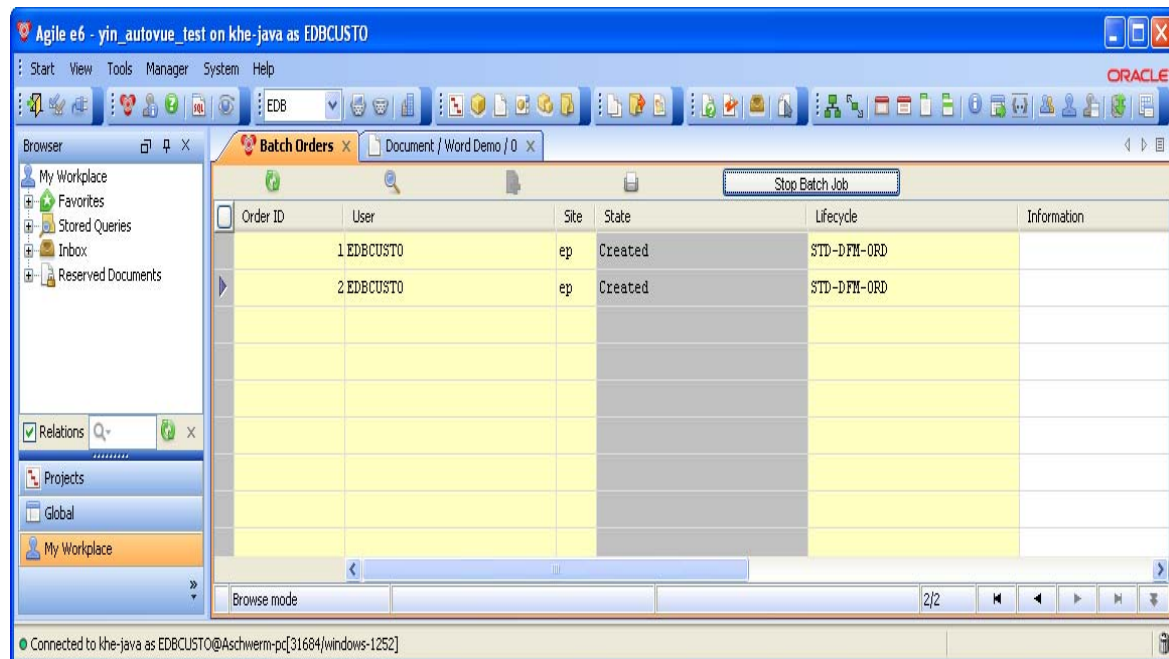
- VueLink DMS as data source and used to check-in the meta file
- AutoVue-Batch-Engine implemented in Java (JVue bean control)
- DFM replication is not supported; all files have to be replicated with the standard replication feature before the file can be cached on remote locations. The system checks if all files are replicated and sets an error code if not all files are available on the remote location.

Order Queue

The Offline Metafile Cache feature uses an order queue to send the native files and the checked-in metafile (if available) to the AutoVue server (System->AutoVue->Batch Orders)

An order has one or more order elements or order positions which contain the necessary document and file data.

The state shows the current working state of the order.

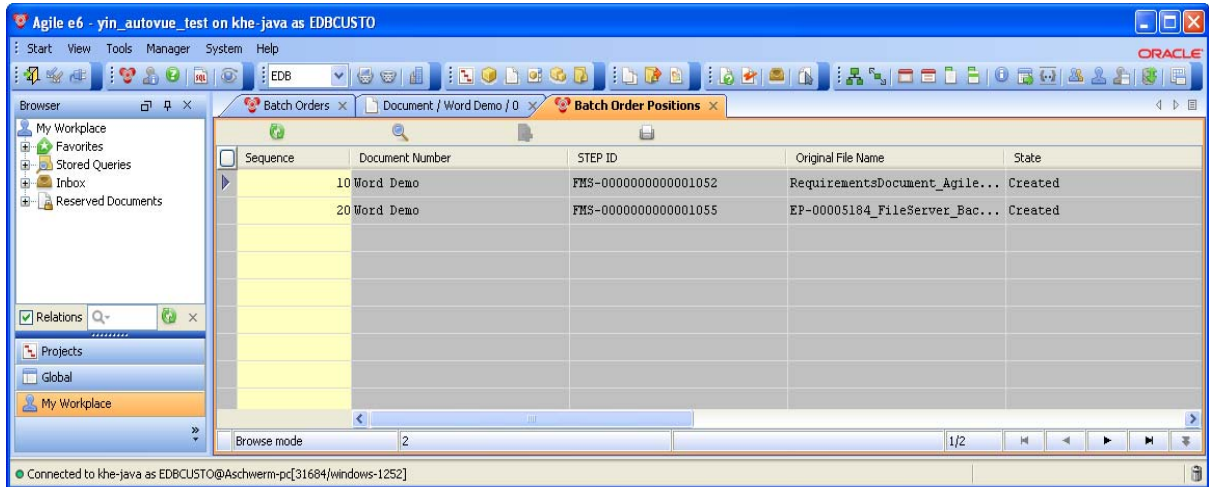


Order Positions

The order positions are assigned to the order and specify which files should be cached on the AutoVue server.

It is only necessary to add the base file of a 3D assembly. The batch job caches all related native files on the AutoVue server to pre-cache the assembly.

The state shows the current working state of the order position.



Creating an Offline Metafile Cache Order

The system provides two userexits to add documents and files to the Offline Metafile Cache:

- **xpvm_bat_doc** - creates a new order and adds all assigned files which are viewable with AutoVue to an order position.
- **xpvm_bat_fil** - creates a new order and adds the specified file to an order position.

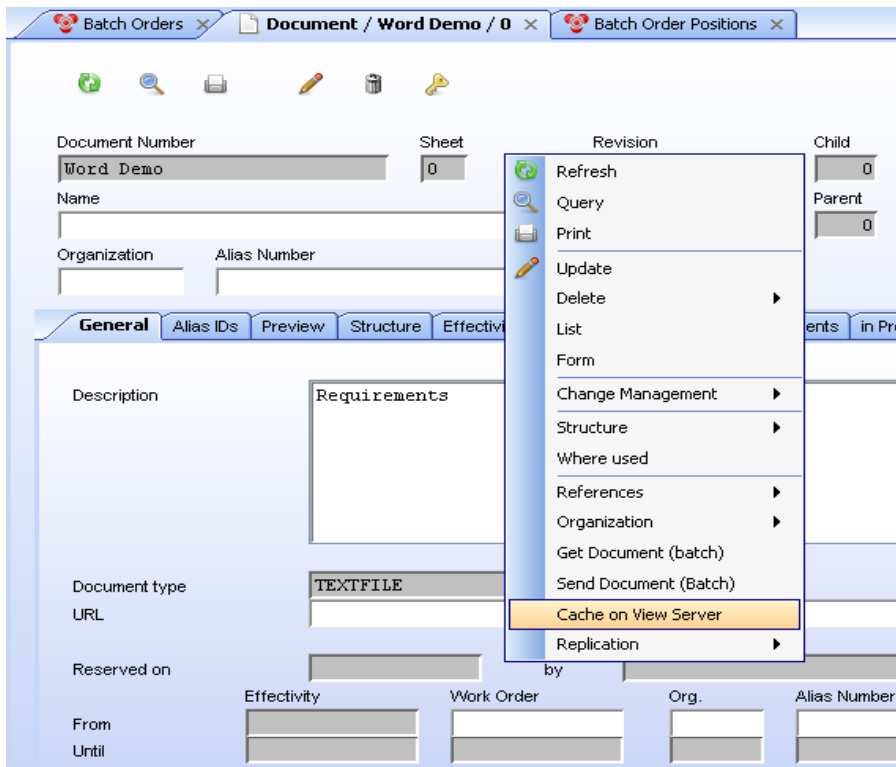
Both the userexits are select-menu userexits.

xpvm_bat_doc can be used on document masks and **xpvm_bat_fil** on Document -File relation masks.

It is also possible to create orders and elements via LogiView by creating records in the order and order position masks.

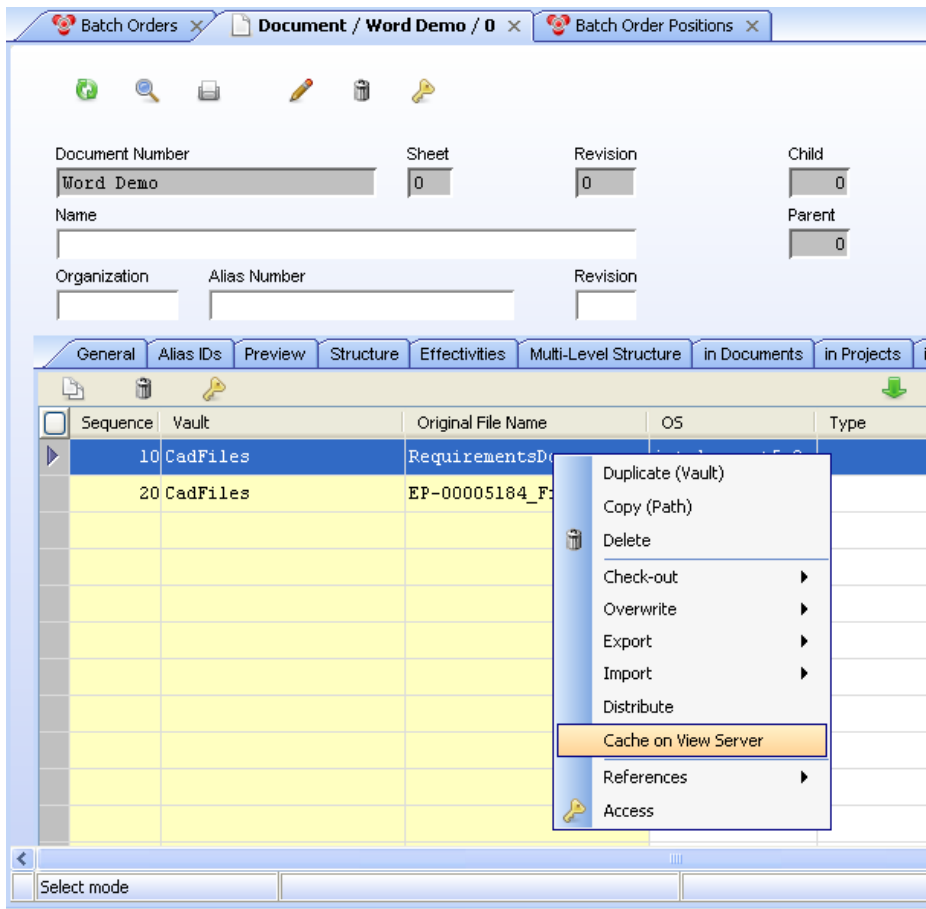
Adding All Files of a Document

In the standard installation, the userexit **xpvm_bat_doc** is available on the document master form.



Adding a Specific File

In the standard installation, the userexit **xpvm_bat_fil** is available on the document file sub list.



Batch Process

- The Offline Metafile Cache feature uses the Batch-Client to control the caching process.
- The main loop is implemented on the Batch-Client to reduce the load on the PLM server machine.
- The sleep interval length can be configured in the Batch-Client settings.
- The batch process is available on Windows only and should be installed on a machine near to the AutoVue Server to reduce the network traffic.
- The batch process can be installed as a Windows Service or run as a console program.

Installing the Offline Metafile Cache Service

This section describes the manual installation of the Offline Metafile Cache Service.

Extract the Installation Package

The installation package of the Offline Metafile Cache Service is located on the **package** directory and named **AutoVueBatch.zip**.

Extract the package to your installation directory.

In this installation example, the installation path is "C:\Program Files\Agile_e6\AutoVueBatch" (the *AutoVueBatch* path is the base path in the package)

```
unzip z:\package\AutoVueBatch.zip -d "c:\Program Files\Agile_e6"
```

The installation package contains the following directories:

```
-> PDF Service Root      (C:\Program Files\Agile_e6\AutoVueBatch)
-> axalant
  -> cmd                  (scripts directory)
  -> batch                (Batch service files)
  -> bin
    -> intel-ms-nt5.1     (binaries like the FMS-Client)
    -> java               (e6.1 Java archives)
-> ext
  -> bin
    -> intel-ms-nt5.1     (external binaries)
    -> java               (external Java archives)
-> tmp                   (Logging directory)
```

Adapting the Installation

You need to adapt the start-up script to setup the Java-Runtime and the installation path of the Offline Metafile Cache Service.

The script is located at the **axalant/cmd** sub directory of the installation.

The **vuelink_batch.bat** script contains the following basic configuration settings:

```
set JAVA_HOME=<JAVA_HOME>
set ep_root=<ROOT DIRECTORY OF THE OFFLINE METAFIELD CACHE SERVICE>
```

Example:

```
set JAVA_HOME=C:\Program Files\Java\jdk1.6
set ep_root=C:\Program Files\Agile_e6\AutoVueBatch
```

Adapting the Service Settings

The **vuelink.properties** file is located at the **axalant/batch** sub directory of the installation. This file sets the environment variables needed by the Offline Metafile Cache Service.

The following properties must be adapted (the other properties should not be changed):

```
#
# ECI connection
#
host=<HOSTNAME OF THE ECI DAEMON>
port=<PORT OF THE ECI DAEMON>
env=<PLM APPLICATION ENVIRONMENT>
scope=BATCH
#
# Directories
#
varenv.ep_root=<ROOT DIRECTORY OF THE PDF SERVICE>
varenv.axalant_root=<AXALANT DIRECTORY OF THE PDF SERVICE>
varenv.$TMP=<PDF WORK DIRECTORY OF THE PDF SERVICE>
#
# Environment
#
varenv.EP_DDM_SITE=<DFM SITE>
#
# PLM Client
#
client1=<PLM USER>,<PASSWORD>,com.agile.autovue.VueLinkBatch
```

Example :

```
#
# ECI connection
#
host=khe-plm
port=20001
env=plm_ref
scope=BATCH
#
# Directories
#
varenv.ep_root=C:/Program Files/Agile_e6/AutoVueBatch
varenv.axalant_root=C:/Program Files/Agile_e6/AutoVueBatch/axalant
varenv.$TMP=C:/Program Files/Agile_e6/AutoVueBatch/tmp
#
# Environment
#
varenv.EP_DDM_SITE=ep
#
# PLM Client
#
client1=DEMOEP_M,not4test,com.agile.autovue.VueLinkBatch
```

Runtime

The Offline Metafile Cache Service uses the same mechanism as the Java-Daemon to install, remove, start, and stop the service.

Install as Service

The configuration of the Windows Service registration can be found in the vuelink_wrapper.conf file which is located in the axalant/pdf sub directory of the installation.

```
***** #
Wrapper NT Service Properties
*****
# WARNING - Do not modify any of these properties when an application
using this configuration file has been installed as a service.
# Please uninstall the service before modifying this section. The
service can then be reinstalled.
# Name of the service
wrapper.ntservice.name=AgilePLM_AutoVue_Batch_Service

# Display name of the service
wrapper.ntservice.displayname=AgilePLM_AutoVue_Batch_Service

# Description of the service
wrapper.ntservice.description=Offline Metafile Cache for Agile PLM

# Service dependencies. Add dependencies as needed starting from 1
wrapper.ntservice.dependency.1=

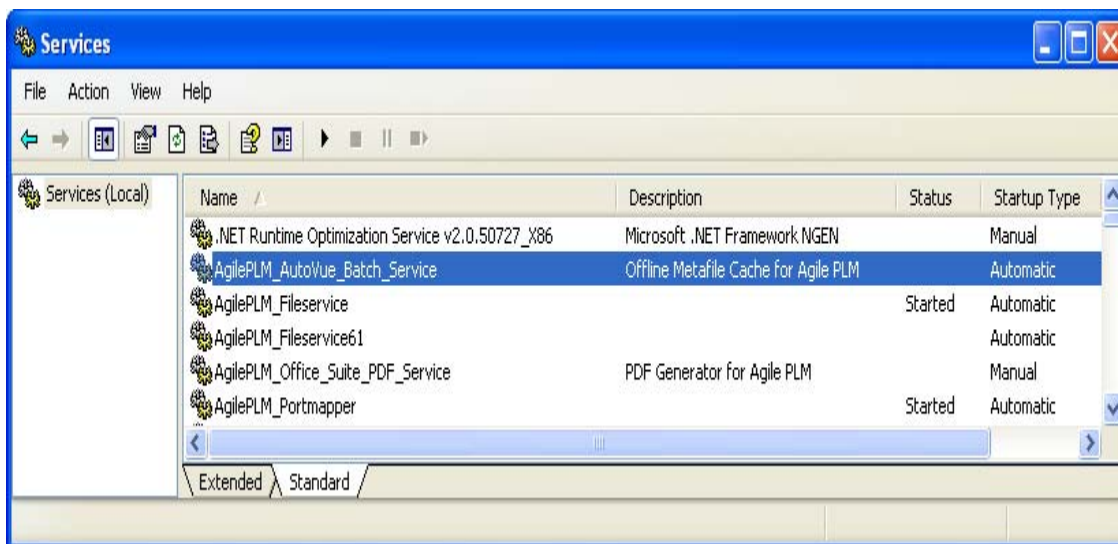
# Mode in which the service is installed. AUTO_START or DEMAND_START
wrapper.ntservice.starttype=AUTO_START

# Allow the service to interact with the desktop.
wrapper.ntservice.interactive=false

wrapper.ntservice.account=.\axalantrt
wrapper.ntservice.password=*****
```

To install the Offline Metafile Cache Service as Windows Service use the vuelink_batch.bat command script located in the axalant/cmd sub directory of the installation.

```
vuelink_batch.bat -i
```

Removing a Service

To remove the Offline Metafile Cache Service as Windows Service, use the vuelink_batch.bat command script located in the axalant\cmd sub directory of the installation.

```
vuelink_batch.bat -r
```

Running as a Console Application

To run the Offline Metafile Cache Service as a console application, use the vuelink_batch.bat command script located in the axalant\cmd sub directory of the installation.

```
vuelink_batch.bat -c
```

