

CA BEST PRACTICES

CA Process Automation

Web Services Utilities User Guide

Rev. 1.1



LEGAL NOTICE

This publication is based on current information and resource allocations as of its date of publication and is subject to change or withdrawal by CA at any time without notice. The information in this publication could include typographical errors or technical inaccuracies. CA may make modifications to any CA product, software program, method or procedure described in this publication at any time without notice.

Any reference in this publication to non-CA products and non-CA websites are provided for convenience only and shall not serve as CA's endorsement of such products or websites. Your use of such products, websites, and any information regarding such products or any materials provided with such products or at such websites shall be at your own risk.

Notwithstanding anything in this publication to the contrary, this publication shall not (i) constitute product documentation or specifications under any existing or future written license agreement or services agreement relating to any CA software product, or be subject to any warranty set forth in any such written agreement; (ii) serve to affect the rights and/or obligations of CA or its licensees under any existing or future written license agreement or services agreement relating to any CA software product; or (iii) serve to amend any product documentation or specifications for any CA software product. The development, release and timing of any features or functionality described in this publication remain at CA's sole discretion.

The information in this publication is based upon CA's experiences with the referenced software products in a variety of development and customer environments. Past performance of the software products in such development and customer environments is not indicative of the future performance of such software products in identical, similar or different environments. CA does not warrant that the software products will operate as specifically set forth in this publication. CA will support only the referenced products in accordance with (i) the documentation and specifications provided with the referenced product, and (ii) CA's then-current maintenance and support policy for the referenced product.

Certain information in this publication may outline CA's general product direction. All information in this publication is for your informational purposes only and may not be incorporated into any contract. CA assumes no responsibility for the accuracy or completeness of the information. To the extent permitted by applicable law, CA provides this document "AS IS" without warranty of any kind, including, without limitation, any implied warranties of merchantability, fitness for a particular purpose, or non-infringement. In no event will CA be liable for any loss or damage, direct or indirect, from the use of this document, including, without limitation, lost profits, lost investment, business interruption, goodwill or lost data, even if CA is expressly advised of the possibility of such damages.

COPYRIGHT LICENSE AND NOTICE:

This publication may contain sample application programming code and/or language which illustrate programming techniques on various operating systems. Notwithstanding anything to the contrary contained in this publication, such sample code does not constitute licensed products or software under any CA license or services agreement. You may copy, modify and use this sample code for the purposes of performing the installation methods and routines described in this document. These samples have not been tested. CA does not make, and you may not rely on, any promise, express or implied, of reliability, serviceability or function of the sample code.

Copyright © 2013 CA. All rights reserved. All trademarks, trade names, service marks and logos referenced herein belong to their respective companies. Microsoft product screen shots reprinted with permission from Microsoft Corporation.

TITLE AND PUBLICATION DATE:

CA Process Automation Web Services Utilities User Guide
Publication Date: September 30, 2013

CA PRODUCT REFERENCES

This document references the following CA products:

- CA Process Automation (PAM)

FEEDBACK

Please email us at impcdfedback@ca.com to share your feedback on this publication. Please include the title of this publication in the subject of your email response. For technical assistance with a CA product, please contact CA Technical Support at <http://ca.com/support>. For assistance with support specific to Japanese operating systems, please contact CA at <http://www.casupport.jp>.

Contents

Chapter 1: CA Process Automation Utilities	7
Before You Begin	7
Command Line Interface	8
Setup and Configuration	8
Syntax and General Usage	8
Supported Actions	11
abortProcess	12
approvePendingInteraction	12
controlInstance	13
deleteArchivedInstances	14
deleteAttachments	15
exportObject	15
generateEvent	17
getAttachments	18
getMatchingEvents	19
getPendingInteraction	19
getPendingInteractions	20
getProcessLogs	21
getProcessParameters	22
getProcessPendingInteractions	23
getProcessStatus	23
getServerStatus	25
getServiceConfig	25
getStartRequestForms	26
getStartRequestProcessReferenced	26
getStartRequestStatus	27
importObject	28
rejectPendingInteraction	29
resumeProcess	30
setServiceConfig	31
startProcess	31
startRequest	33
suspendProcess	34
Action Files	35
Using Action Files	35
Argument Super-Session	37
Troubleshooting	38
Web Service Client	39
Sample Eclipse Project	39
Https access using Truststore	41
Changes from previous version	42
Known Limitations	43
Summary	43

Chapter 1: CA Process Automation Utilities

The CA Process Automation Web Service Utilities are field developed utilities designed to simplify interaction and integration with CA Process Automation. Interaction is supported through the following two levels:

- **Command line interface (CLI)** that can be invoked interactively or programmatically
- **Java web service client wrapper** that eliminates the need to build a web service stub from the “out-of-the-box” WSDL and that works with more complex structures that can be included as a library in any Java application.

In a relatively short amount of time you should be able to initiate, control and track the progress of processes defined to CA Process Automation from the command line or external application.

Before You Begin

Before attempting use the utility you should:

- Be familiar with the CA Process Automation architecture and management interfaces
- Have a working knowledge the terms “library”, “object” and “operator” in the context of CA Process Automation
- Have some experience defining, launching and troubleshooting CA Process Automation processes
- Understand the concepts of “datasets”, “attachments”, and “parameters” as applied to CA Process Automation processes and datasets
- Verify that CA Process Automation is installed and available. To do this launch a URL similar to the following from your browser:

`http(s)://<Server Name or IP>:<Port>/itpam/soap?wsdl`

The CA Process Automation WSDL should be returned and displayed. If the service fails to respond, confirm the service address and port and verify that the CA Process Automation services have been started on the server. If the service still fails to respond, review the product documentation and other published troubleshooting guides to resolve the issue. If, after exploring these methods, you are still unable to communicate with the web service, contact CA Support for further assistance.

Note: If your orchestrator only supports secure https connections, please refer to the Https access using Truststore section for additional details.

As with any new software component, you should test and explore the utility in a lab environment prior to moving it into a production environment.

Command Line Interface

The command line interface (CLI) can be useful in a variety of scenarios. For example:

- Providing simple scripts that end users can run to initiate self-service processes
- Enabling simple integration with external applications that may only be capable of executing a command and accepting a return code
- Scripting unit and regression testing of newly defined or modified processes

The CLI provides a command driven, text based interface that is simple to use while still supporting the capabilities of the CA Process Automation web service.

Setup and Configuration

It is not necessary to install any CA Process Automation components locally on system(s) where the command line interface will be used - only a Java Runtime Environment (JRE) version 6 or 7. The Web Service Utilities package includes the necessary binary and Java library files. To setup and configure the utility do the following:

1. Extract the contents of the package to any directory (for example, C:\Program Files\CA\CA IT PAM Utilities)
2. Open the "caitpamwsucli.cmd" file with any text editor
3. Edit the following line to set the location of the installed JRE

```
set JRE_PATH=<Path to Installed Java JRE 6 or Java 7 JRE (ex.
C:\Program Files\Java\jre7)>
```
4. Save the file

The utility is ready for use.

Syntax and General Usage

As with most command line interfaces, the CLI is invoked by calling or executing the executable file followed by the appropriate switches and arguments. For convenience, help with syntax and usage can also be output to the console or redirected to a file by executing the following command:

```
caitpamwscli.cmd [/? or /h or /help]
```


To interact with the CA Process Automation web service in order to initiate a process or request information, you will need to include arguments that specify the “action” to perform as well as qualifying parameters. Arguments are specified as name/value pairs joined with an equal sign (“name=value”). An argument value that contains spaces must be enclosed in double quotes. For example:

```
name="value with spaces"
```

Multiple argument name value pairs must be separated with spaces. For example:

```
name1=value1 name2=value2
```

Spaces between an argument name and value are not allowed. For example, the following is valid:

```
name=value
```

While the following will be rejected:

```
name = value
```

Commands may be run silently (typically when invoked in scripts or triggered by an external application) or interactively in a console session.

Silent Execution

To support integration and scripting, commands run silently by default. Commands that produce output will write to STDOUT when successful and provide no output when an error occurs. Error messages will be written to the log file only. The environment variable "ERRORLEVEL" will be set to one of the following success/error codes and the success/error code will be returned:

Code	Meaning
0	Completed successfully
9001	Attempt to initialize CLI utilities failed
9002	Unexpected Java or web service exception
9003	CA Process Automation entity not found or inaccessible
9004	Method not supported for specified action/entities
9005	Required argument or parameter not specified
9006	Unsupported date format specified
9007	Host specified not found or inaccessible
9008	Insufficient user permissions
9009	Unexpected web service exception thrown by web service host
9999	Generic failure indicator, check CLI log file under 'logs' subdirectory directly inside the CLI folder for additional information

Important: When “wrapping” the command in Windows batch scripts remember to “call” the command instead of simply executing it to capture the “ERRORLEVEL”. For example:

```
call caitpamwscli.cmd <Switches and arguments>
```

Interactive Execution

Commands can also be executed interactively by specifying the “/i” switch as the first argument. For example:

```
caitpamwscli.cmd /i <Additional switches and arguments...>
```

When invoked interactively, all output, including error messages and usage tips, will be displayed on the console.

Service Configuration and Authentication

Actions that communicate with the CA Process Automation web service will require users to specify the following values:

- user=<CA Process Automation user name>
- password=<CA Process Automation user’s password>
- uri=<CA Process Automation web service URI (for example, “http(s)://<SERVER>:<PORT>/itpam/soap”)>

For security reasons, the user name and password are not stored and must be specified *each time* a command is invoked.

If the specified URI begins with “https”, you will also have to specify the trust store and trust store password. For full details on setting up the trust store see the [Https access using Truststore](#) section.

The “setServiceConfig” action has been provided as a convenient way to store the “uri”, “version” and “truststore” values eliminating the need to specify the arguments each time.

```
caitpamwscli.cmd /i user=<User> password=<Password>  
action=setServiceConfig uri=<URI> version=<Version>
```

You can retrieve the current saved values by using the “action” argument “getServiceConfig”.

```
caitpamwscli.cmd /i user=<User> password=<Password>  
action=getServiceConfig
```

Saved values are only recognized by the instance of "caitpamwscli.cmd" executing the command. For example, if the utility is installed on "MachineA" and "MachineB", executing the command on "MachineA" would have no impact on the configuration of "MachineB" and vice versa. Likewise, if two instances of the utility are installed on "MachineA" - once in "<Drive:\

The saved values may be superseded by specifying the arguments when executing a command.

The user may also always supply "version" as a parameter.

version=<CA Process Automation version (for example, 3.1 or 4.1)

This is optional, since as part of its processing the utility will directly obtain the actual version for the PAM instance specified by the "uri".

Note: If "version" is explicitly provided as a parameter and the version retrieved by the utility is less than the version specified by the user, the utility will exit with an appropriate error message; otherwise the "version" parameter supersedes the version retrieved by the utility.

Supported Actions

An "action" argument must be specified for each command. Additional required and optional arguments vary dependent upon the action specified. For convenience, help with syntax and usage for supported actions can also be output to the console or redirected to a file by executing the following command:

```
caitpamwscli.cmd </? or /h or /help> action=<Supported Action>
```

A current list of supported actions is included in the output of the general usage command (see Syntax and General Usage).

abortProcess

This action aborts processing of the specified process.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=abortProcess processId=<Process ID>
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soapaction=abortProcess  
processId=1 truststore="C:\mykeystore.jks"  
truststorepassword="pamadmin"
```

The command outputs a message indicating that it completed successfully.

approvePendingInteraction

This action flags pending interaction specified as approved to continue.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=approvePendingInteraction interactionId=<Interaction ID>
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soap  
action=approvePendingInteraction interactionId=7  
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

The command outputs a message indicating that it completed successfully.

controlInstance

This action allows user to set the archiving flag of a process or start request form instance (and all its child instances that were not triggered in detached mode). In other words, this can be used to make a Process or Start Request Form (and descendants) subject to archival policies. This can be useful if isAutoArchive =false had been specified when initiating the Process or Start Request Form, and you now want the item to be archivable.

This action was introduced with the current version of the utilities, and may only be run against CA Process Automation versions 4.0 or later.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=controlInstance [rootUuid=<Root UUID>]  
[instanceId=<Instance ID>] [tagId=<Unique ID>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details)

rootUuid: this option specifies the rootUUID of the process or start request form

instanceId: this option specifies the instance ID of the process or start request form

tagId: this option specifies the tagId of the process or start request form

You must specify the instance of the process or start request form with either rootUuid, instanceId, or tagId. If more than one of these is specified, rootUuid supersedes instanceId and tagId and instanceId supersedes tagId.

This new action requires IT PAM version 4.0 or higher, and fails when running against earlier versions.

Example: user calls controlInstance for a process and specifies the process instanceId.

```
caitpamwscli.cmd /i LogLevel=DEBUG user=someuser  
password=somepass uri=http://myserver.edu:8080/itpam/soap  
action=controlInstance instanceId=6998
```

If the parameters are correct, and a process with ROID 6998 exists the following is returned.

The archive action was successfully executed.

deleteArchivedInstances

This action purges archived process instances in the specified date range.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=deleteArchivedInstances startDate=<Date> endDate=<Date>
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details)

Supported date formats include:

- yyyy-MM-dd
- yyyy/MM/dd
- MM-dd-yyyy
- MM/dd/yyyy

For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soap  
action=deleteArchivedInstances startDate=2012-12-13 endDate=2012-  
12-14 truststore="C:\mykeystore.jks"  
truststorepassword="pamadmin"
```

The command outputs a message indicating that it completed successfully. For example:

7 archived instances were deleted successfully.
Request completed successfully

deleteAttachments

This action deletes the specified attachments.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=deleteAttachments attachmentId=<Attachment ID  
1>...attachmentId=<Attachment ID N>
```

Arguments "uri" and "version" are optional only if a default web service URI and version has been set (see [Service Configuration and Authentication](#) for more details).

At least one "attachmentId" argument must be specified. Any number of "attachmentId" arguments may be specified. For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soap action=deleteAttachments  
attachmentId=3 truststore="C:\mykeystore.jks"  
truststorepassword="pamadmin"
```

The command does not generate output.

exportObject

This action exports objects from CA Process Automation library to a specified file.

```
caitpamwsu.cmd    [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>][version=<Web Service Version>]  
action=exportObject sourceLibraryPath=<Source ITPAM library path  
to search for content to be exported (include object name to  
export a specific object)> targetPath=<Target file system path  
for resulting object export (Note: Content may exported to  
existing directory on the IT PAM server or an accessible UNC  
path)> targetFileName=<Target file name for resulting object  
export file> [targetPathIsLocal=<Flag set to true if target  
path/file name specified is local(relative to the client) or  
false if relative to the server (default is false)]  
[levelsToSearch=<Integer set levels relative to source library path to  
search/include for content to be exported (default is 0)]  
[absolutePath=<Flag set to true if library paths in resulting export  
file should be absolute or false if paths should be relative (default  
is false)>] [overwriteTargetFile=<Flag set to true if any existing
```

export file with the same name in the same location should be overwritten (default is false)>] [objectType=<String set to a valid object type applied as a filter that determines which objects in the source path to include in the exported content (Note: Multiple name/value pairs may be specified) [outputFormat=<Properties (name/value pairs) (default) or XML>]

Supported object type filter strings:

- Agenda (now known as Schedule since PAM 4.0)
- Calendar
- Commander
- CustomIcon
- CustomOperator
- CustomSensor
- Dataset
- InteractionRequestForm
- LogViewer (deprecated)
- Package
- Process
- ProcessWatch
- Resources
- StatePolicy (deprecated)
- System(deprecated)

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=exportObject
sourceLibraryPath=/TEST targetPath="C:\\\"
targetFileName=exporttestfile.xml targetPathIsLocal=false
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

This command does not return output to the calling application or console. The CA Process Automation export file is written directly to the specified file system location.

generateEvent

This action generates an event consumable by the specified waiting CA Process Automation process operator(s)

```
caitpamwsu.cmd    [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=generateEvent eventName=<Event Name> [eventType=<Event
Type>] [eventDestination=<Event Destination>] [eventSource=<Event
Source>] [eventExpirationDuration=<Event Duration in Seconds
(default is 0, expires immediately)>] [eventSingleSubscriber=<True
or False to flag event as "consumable" by a single listener
only>] [eventPayload=<Event Pay Load (content to be used by
waiting process operator(s) - creates a simple parameter in the
waiting process operator(s) dataset named "payload" set to the
value specified)>] [param.<Parameter Name1>=<Parameter
Value1>...param.<Parameter NameN>=<Parameter ValueN>] creates an
value map parameter in the waiting process operator(s) dataset
named "payload" and adds elements for each parameter specified
(see IT PAM for information on referencing individual member
elements of value maps)]
```

Important: Event payload ("eventPayload") will be superseded (ignored) if a parameter or parameters are specified.

Argument "uri" is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=generateEvent
eventName=TestEvent eventExpirationDuration=300
eventSingleSubscriber=false truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs a listing of event data for events generated in either "Properties" format (name/value pairs (default)) or "XML" format. For example:

```
eventName=TestEvent
eventCreationTime=12/13/12 5:08 PM
eventExpirationTime=12/13/12 5:13 PM
eventDestination=
eventExpression=
eventSingleSubscriber=false
eventSource=
eventType=
eventUser=pamadmin
```

getAttachments

This action returns properties of the specified attachments.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password> [uri=<Web Service URI>] [version=<Web Service Version>] action=getAttachments  
contentId=<Content ID (Attachment Name)> contentType=<Content Type>  
[outputFormat=<Properties (Name/Value Pairs) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soap action=getAttachments  
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

Following is an example of a specific attachment:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soap action=getAttachments  
contentId=333 truststore="C:\mykeystore.jks"  
truststorepassword="pamadmin"
```

The command outputs a listing of attachments that match the specified criteria in either “Properties” format (name/value pairs (default)) or “XML” format. For example, the output of the first example above would be:

```
1.attachmentId=1  
1.contentId=111  
1.contentType=text/xml  
1.attachmentName=books.txt  
  
2.attachmentId=2  
2.contentId=111  
2.contentType=text/xml  
2.attachmentName=  
  
4.attachmentId=4  
4.contentId=333  
4.contentType=text/xml  
4.attachmentName=books.txt
```

While the output of the second might be:

```
4.attachmentId=4  
4.contentId=333  
4.contentType=text/xml  
4.attachmentName=books.txt
```

getMatchingEvents

This action returns event data for events matching the parameters specified.

```
caitpamwsu.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getMatchingEvents eventName=<Event Name>
[eventType=<Event Type>] [eventDestination=<Event Destination>]
[eventSource=<Event Source>] [eventExpression=<Expression that
will be compared to event "payload" parameter(s) (ex.
payload.paramName=="paramValue")>] [allowPartialMatch=<Specify
true to return data for events that partially match the specified
criteria (default is false)>] [enablePatternMatch=<Specify true
if criteria specified should be interpreted as regex-type
patterns (default is false)>] [outputFormat=<Properties
(name/value pairs) (default) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=getMatchingEvents
eventName=TestEvent truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs a listing of event data for events matching the specified criteria in either “Properties” format (name/value pairs (default)) or “XML” format. For example:

```
event2045652012.eventName=TestEvent
event2045652012.eventCreationTime=12/13/12 5:08 PM
event2045652012.eventExpirationTime=12/13/12 5:13 PM
event2045652012.eventDestination=
event2045652012.eventExpression=
event2045652012.eventSingleSubscriber=false
event2045652012.eventSource=
event2045652012.eventType=
event2045652012.eventUser=pamadmin
```

getPendingInteraction

This action returns properties of the specified pending interaction.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getPendingInteractions [outputFormat=<Properties
(Name/Value Pairs) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=getPendingInteraction
interactionId=33 truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs the properties of the specified interaction in either “Properties” format (name/value pairs (default)) or “XML” format. For example:

```
interactionId=33
processId=27
description=Test
```

getPendingInteractions

This action returns all pending interactions and related properties.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getPendingInteractions [outputFormat=<Properties
(Name/Value Pairs) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap
action=getPendingInteractions truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs a listing of all pending interactions and related properties in either “Properties” format (name/value pairs (default)) or “XML” format.

For example:

```
interactionId=33
processId=27
description=Test
interactionId=40
processId=34
description=Test
interactionId=47
processId=41
description=Test
```

getProcessLogs

This action returns process log entries for the specified process filtered by specified optional criteria.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getProcessLogs processId=<Process ID> [level=<Minimum
Severity Level of Log Entries> [category=<Log Category
1>...category=<Log Category N>] [outputFormat=<Properties
(Name/Value Pairs) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=getProcessLogs
processId=41 level=Normal category=Process category=Operator
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

All log messages for the process specified are returned by default. For example:

```
2012-12-13 02:30:11:546, NOTICE, Process: 'Process_41' instance
was created.
```

```
2012-12-13 02:30:11:546, NOTICE, Process: 'Process_41' is in
'Queued' state.
```

```
2012-12-13 02:30:11:671, NOTICE, Process: Process started at
'12/13/2012 14:30:11' by 'pamadmin'.
```

```
2012-12-13 02:30:11:687, NOTICE, Operator: 'Assign_User_Task_1'
is enabled following 'Start_1'.
```

```
2012-12-13 02:30:11:687, NOTICE, Operator: 'Start_1' is 'Completed' on
'Current Server'.
```

```
2012-12-13 02:30:11:703, NOTICE, Operator: A service request was sent
for 'Assign_User_Task_1'.
```

```
2012-12-13 02:30:11:703, NOTICE, Operator: 'Assign_User_Task_1' is
'Running' on 'Current Server'.
```

```
2012-12-13 02:30:11:718, NOTICE, Process: The process is in Waiting
state.
```

Specifying a valid value for “level” will filter messages and return only messages with the severity level specified and higher. Valid levels are “Error”, “Warning”, “Notice” and “Normal”. Invalid values for “level” will be ignored.

Specifying one or more “category” arguments will filter messages and return only messages associated with the category/categories specified. Valid categories are “Process”, “Operator”, “Handler”, “Response”, “Other” and “Custom”. Invalid values for “category” arguments will be ignored.

The command outputs a listing of log entries for the specified process matching the optional level and category filters specified in either “Properties” format (name/value pairs (default)) or “XML” format.

getProcessParameters

This action returns parameter names and values for process specified.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getProcessParameters processId=<Process ID>
[outputFormat=<Properties (Name/Value Pairs) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=getProcessParameters
processId=146 truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs a listing of parameters names and values for the specified process in either “Properties” format (name/value pairs (default)) or “XML” format.

For example:

```
146.param.RootProcessROID=
146.param.rootUUID=abdb7d81-b25b-49ed-adf8-cbd7c0fe118a
146.param.StartTime=2012-12-13 14:45:30
146.param.effectiveUser=pamadmin
146.param.touchpoint=Orchestrator
146.param.Progress=100.0
146.param.EndDate=2012-12-13 14:45:35
146.param.TouchpointName=Orchestrator
146.param.Result=1
146.param CallerUser=pamadmin
146.param.InstanceUUID=abdb7d81-b25b-49ed-adf8-cbd7c0fe118a
146.param.ScheduledStartTime=2012-12-13 14:45:30
146.param.ServerId=12a21e33-a82c-48b2-9274-be0c2c7d7e07
146.param.InstanceName=CreatingPrerequisites_146
146.param.ServerName=server01.ca.com
146.param.EndTime=2012-12-13 14:45:35
```

```
146.param.StartDate=2012-12-13 14:45:30
146.param.RuntimeROID=146
146.param.DisplayName=CreatingPrerequisites
146.param.ParentProcessROID=
```

getProcessPendingInteractions

This action returns pending interactions and related properties for process specified.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getProcessPendingInteractions processId=<Process ID>
[outputFormat=<Properties (Name/Value Pairs) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap
action=getProcessPendingInteractions processId=163
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

The command outputs a listing of pending interactions and related properties for the specified process in either “Properties” format (name/value pairs (default)) or “XML” format.

For example:

```
interactionId=171
processId=163
description=Test
interactionId=172
processId=163
description=Test
interactionId=173
processId=163
description=Test
```

getProcessStatus

This action returns current status and related properties for process specified.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getProcessStatus processId=<Process ID>
[tagID=<UniqueID>] [outputFormat=<Properties (Name/Value Pairs) or
XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=getProcessStatus
processId=41 truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs the current status and a listing of related properties for the specified process in either “Properties” format (name/value pairs (default)) or “XML” format. For example:

```
connectionId=none
message=Flow state received
messageStatus=
processDisplayName=Process
processId=41
processInstanceName=Process_41
processLibraryPathName=
processState=Inactive
senderAddress=none
submissionTime=2012-12-13T14:50:22.437+05:30

41.param.RootProcessROID=
41.param.rootUUID=e7ad393f-54dd-4021-ae89-2acae505732d
41.param.StartTime=2012-12-13 14:30:11
41.param.effectiveUser=pamadmin
41.param.Progress=0
41.param.EndDate=
41.param.TouchpointName=Orchestrator
41.param.Result=
41.param CallerUser=pamadmin
41.param.InstanceUUID=e7ad393f-54dd-4021-ae89-2acae505732d
41.param.ScheduledStartTime=2012-12-13 14:30:11
41.param.ServerId=12a21e33-a82c-48b2-9274-be0c2c7d7e07
41.param.InstanceName=Process_41
41.param.ServerName=server01.ca.com
41.param.EndTime=
41.param.StartDate=2012-12-13 14:30:11
41.param.RuntimeROID=41
41.param.DisplayName=Process
41.param.ParentProcessROID=
```

Starting with the current version of the utilities, GetProcessStatus supports a new input: tagId. If a tagId was specified when starting the Process, you may refer to the process instance either by processId or tagId in subsequent requests.

This new input field requires CA Process Automation version 4.0 or higher. It is ignored when running against earlier versions of PAM.

Example: user specifies a tagId with value 08b6d113-0243-4aby-8cre-6581c25747c9

```
caitpamwscli.cmd /i LogLevel=DEBUG user=someuser password=somepw  
uri=http://someserver.com:8080/itpam/soap action=getProcessStatus  
tagid=08b6d113-0243-4aby-8cre-6581c25747c9
```

If parameters are otherwise correct, output similar to that for a call without “tagId” specified is returned.

getServerStatus

This action returns pending interactions and related properties for process specified.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=getServerStatus
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soap action=getServerStatus  
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

The command outputs a string indicating current server status. For example:

```
Server status ok. (https://server01:8443/itpam/soap version 4.1)
```

getServiceConfig

This action returns current stored values for IT PAM web service URI and version.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=getServiceConfig
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin  
uri=https://server01:8443/itpam/soap action=getServiceConfig  
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

The command outputs the current stored values as name/value pairs. For example:

```
uri=https://server01:8443/itpam/soap
truststore=C:\WSCMDUtilities\Nov16\Distribution\Distribution\myke
ystore.jks
```

getStartRequestForms

This action returns start request forms and related properties located in the specified search path.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getStartRequestForms [searchPath=<Library Path to Search
(Default is root ("\"/"))>] [recursive=<True or False (Default is
False)>] [outputFormat=<Properties (Name/Value Pairs) or XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=getStartRequestForms
searchPath=/SOAPModule truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs a listing of start request forms and related properties located in the search path specified in either “Properties” format (name/value pairs (default)) or “XML” format. For example:

```
/SOAPModule/SRF1.description=
/SOAPModule/SRF1.startRequestName=SRF1
/SOAPModule/SRF1.startRequestPath=/SOAPModule/

/SOAPModule/SRF2.description=
/SOAPModule/SRF2.startRequestName=SRF2
/SOAPModule/SRF2.startRequestPath=/SOAPModule/
```

getStartRequestProcessReferenced

This action returns the fully qualified process name referenced by the specified start request.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getStartRequestProcessReferenced startRequestName=<Start
Request Name> startRequestPath=<Start Request Library Path>
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap
action=getStartRequestProcessReferenced startRequestName=SRF1
startRequestPath=/SOAPModule truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs the fully qualified process name referenced by the specified start request. For example:

```
/Process_2
```

getStartRequestStatus

This action returns current status and related properties for the specified start request.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=getStartRequestStatus startRequestId=<Start Request ID> |
tagID=<uniqueID> [outputFormat=<Properties (Name/Value Pairs) or
XML>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=getStartRequestStatus
startRequestId=180 truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command outputs the current status and related properties for the specified start request in either “Properties” format (name/value pairs (default)) or “XML” format. For example:

```
earliestStartDate=Thu Dec 13 15:24:16 IST 2012
endDate=Thu Dec 13 15:24:17 IST 2012
interactionId=180
processId=181
processName=/Process_2
startDate=Thu Dec 13 15:24:16 IST 2012
startRequestState=Completed
```

Starting with this version of the utilities, GetStartRequestStatus supports a new input: tagId. If a tagID was specified when starting the start request form, you now may now refer to the Start Request Form instance by startRequestId or tagId.

This new input field requires IT PAM version 4.0 or higher. It is ignored if running against earlier versions of PAM.

Example: user specifies a tagId that was set in StartRequest, tagId=06a4d113-0333-4aba-8cce-8781c18647c8

```
caitpamwscli.cmd /i LogLevel=DEBUG user=someuser password=someuserspw
uri=http://someserver.someorg.com:8080/itpam/soap
action=getStartRequestStatus logLevel=Debug tagid=06a4d113-0333-4aba-
8cce-8781c18647c8
```

If tagID is matched, the command is successful and as before current status and related properties for the specified start request are returned.

```
earliestStartDate=Mon Sep 23 17:19:28 EDT 2013
endDate=Mon Sep 23 17:19:29 EDT 2013
interactionId=6970
processId=6971
processName=/CLI_Tests/newDataset
startDate=Mon Sep 23 17:19:28 EDT 2013
startRequestState=Completed
```

importObject

This action creates/updates a CA Process Automation object defined by the source file in the CA Process Automation library relative to the specified target path and folder. Note that, if an existing object with same name in the same location exists, it will be overwritten.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=importObject importSourcePathFileName=<Source Path and
File Name of Object Definition XML File>
importTargetLibraryPathName=<Target Library Path and Folder Name>
[importOverwriteAction=incrementObjectVersion|replaceObject|
skipImport] importMakeAvailable=True|False
importMakeCurrent=True|False
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details).

Starting with this version of the utilities, importObject supports a new input “importOverwriteAction” , which can be set to one of the following values:

- incrementObjectVersion - Import as a new version and keep the existing object. (this is the default behavior)
- replaceObject - Import and replace the existing object.
- skipImport - Do not import objects with the same name as an existing object.

This new parameter requires PAM version 4.0 or higher, and is ignored if running against an earlier version.

For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=importObject
importSourcePathFileName=C:\\SDM.xml
importTargetLibraryPathName=/TEST importMakeAvailable=true
importOverwriteAction=replaceObject importMakeCurrent=true
truststore="C:\\mykeystore.jks" truststorepassword="pamadmin"
```

This command generates a message indicating that it completed successfully.

rejectPendingInteraction

This action flags pending user interactions as rejected.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=rejectPendingInteraction interactionId=<Interaction ID>
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details).

Note: Beginning with CA Process Automation v2.2, attempts to reject a pending interaction are blocked if the "Show approval page" option for the "User Interaction" is set and an appropriate message will result.

The command does not generate output.

resumeProcess

This action resumes processing of previously suspended process specified.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>  
[uri=<Web Service URI>] [version=<Web Service Version>]  
action=resumeProcess processed=<Process ID>
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details).

For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=resumeProcess
processId=15 truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command does not generate output.

setServiceConfig

This action sets the default web service URI and version to eliminate the need to specify “uri” and “version” arguments for every command issues (see [Service Configuration and Authentication](#) for more details).

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=setServiceConfig uri=<IT PAM Web Service URI> version=<IT
PAM Web Service Version>
```

For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=setServiceConfig
uri=https://server01:8443/itpam/soap
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
```

The command generates a message indicating that it completed successfully.

startProcess

This action starts the specified process.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=startProcess processName=<Fully Qualified Process Name>
[startDate=<Date>] [startTime=<Time>] [tagId=<UniqueID>]
[isAutoArchive=true|false] [param..<Parameter Name 1>=<Parameter
Value 1>...param..<Parameter Name N>=<Parameter Value N>]
[attachmentParameterName=<Attachment Parameter Name>
attach.<Attachment Name 1>=<Attachment Location
1>...attach.<Attachment Name N>=<Attachment Location N>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details).

Default start date and time is “now”. Start date is required if start time is specified. Supported date formats are:

- yyyy-MM-dd
- yyyy/MM/dd
- MM-dd-yyyy
- MM/dd/yyyy

Supported time formats are:

- hh:mm
- hh:mm:ss

Parameter ("param..") arguments are optional. Any number of parameter arguments may be specified.

Attachment ("attach.") arguments are optional. Any number of attachments may be specified. If attachment arguments ("attach.") are specified an attachment parameter name ("attachmentParameterName") and only one attachment parameter name must be specified.

For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=start Process
processName=/Process_2 truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The output of this command is the ROID of the process instance. For example:

186

Starting with this version of the utilities, StartProcess supports new inputs: "tagId" and "isAutoArchive".

tagId allows a user specified ID to be associated with the process instance. This ID should be unique.

isAutoArchive can be set to false to skip process/start request from instances (along with their child instances) from being archived under the archival policy. Processes that are run in detached mode are not affected by this tag. The default value for <IsAutoArchive> is true.

These new input fields require CA Proacess Automation version 4.0 or higher. They are ignored if running against earlier versions of PAM.

Example: user specifies tagId= 08b6d113-0243-4aby-8cre-6581c25747c9 and isAutoArchive=false


```
caitpamwscli.cmd /i LogLevel=DEBUG user=someuser password=somepw
uri=http://someserver.someorg.com:8080/itpam/soap
action=startProcess processName=/somepath/someprocess
tagid=08b6d113-0243-4aby-8cre-6581c25747c9 isAutoArchive=false

6992
```

Process can now be referenced by either the tag value specified or the returned ROID.

startRequest

This action initiates the specified start request.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=startRequest startRequestName=<Start Request Name>
startRequestPath=<Start Request Library Path> [startDate=<Date>]
[startTime=<Time>] [tagId=<UniqueID>] [isAutoArchive=true|false]
[param..<Parameter Name 1>=<Parameter Value 1>...param..<Parameter
Name N>=<Parameter Value N>] [attachmentParameterName=<Attachment
Parameter Name> attach.<Attachment Name 1>=<Attachment Location
1>...attach.<Attachment Name N>=<Attachment Location N>]
```

Argument “uri” is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details).

Default start date and time is “now”. Start date is required if start time is specified. Supported date formats are:

- yyyy-MM-dd
- yyyy/MM/dd
- MM-dd-yyyy
- MM/dd/yyyy

Supported time formats are:

- hh:mm
- hh:mm:ss

Parameter ("param..") arguments are optional. Any number of parameter arguments may be specified.

Attachment ("attach.") arguments are optional. Any number of attachments may be specified. If attachment arguments ("attach.") are specified an attachment parameter name ("attachmentParameterName") and only one attachment parameter name must be specified.

Starting with this version of the utilities, StartRequest supports new inputs: tagId and isAutoArchive

tagId allows a user specified ID to be associated with the start request form instance. This ID should be unique.

isAutoArchive can be set to false to skip process/start request form instances (along with their child instances) from archiving under the archival policy. Processes that are run in detached mode are not affected by this tag. The default value for "isAutoArchive" is true.

These new input fields require CA Process Automation version 4.0 or higher. They are ignored when running against earlier versions of PAM.

For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=startRequest
startRequestName=SRF1 startRequestPath=/SOAPModule
truststore="C:\mykeystore.jks" truststorepassword="pamadmin"
tagid=06a4d113-0333-4aba-8cce-8781c18647c8 isAutoArchive=false
```

The output of this command is the Instance ID of the new SRF. For example:

221

suspendProcess

This action suspends processing of specified process.

```
caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=suspendProcess processed=<Process ID>
```

Argument "uri" is optional only if a default web service URI has been set (see [Service Configuration and Authentication](#) for more details). For example:

```
C:\>caitpamwscli.cmd /i user=pamadmin password=pamadmin
uri=https://server01:8443/itpam/soap action=suspendProcess
processId=1 truststore="C:\mykeystore.jks"
truststorepassword="pamadmin"
```

The command does not generate output. Rather, the response generated by the proceeding example might be:

Request completed successfully

Action Files

Required and optional values may be stored in an “action file” to eliminate the need to specify command line arguments. By reducing the number of arguments that must be entered the effort to use the command line interface is significantly reduced while simultaneously increasing the reliability of the outcome. The benefits are compounded when “action files” are created and referenced for repetitive activities. Arguments values specified in “action files” may be supplemented or superseded by arguments passed when a command referencing an “action file” is executed.

Using Action Files

An “action file” is a simple text file similar to a Windows “INI” or Java “Properties” file. Each entry is a name/value pair corresponding to any name/value pair argument that would be specified for a supported action via the command line (see Supported Actions).

To illustrate, assume the following:

- CA Process Automation user is “user01”
- CA Process Automation user’s password is “pwd01”
- CA Process Automation web service URI is “http://server:8080/itpam/soap”
- CA Process Automation version is “4.1”
- Process to be started is the “Example 1” process in the “/Examples” folder in the library (or fully qualified name “/Examples/Example 1”
- Value for one parameter, “EmailAddress”, defined and used in the “Example 1” process should be set to user02@domain.com
- The process also references an attachment parameter, “EmailAttachments”, that accepts links uploaded attachments to the process. For the example, the file “C:\Temp\Instructions.txt” should be uploaded as an attachment.

Refer to CA Process Automation product documentation for more information on process parameters and attachments.

The first step is to refer to the syntax for the “action” to be performed to determine the supported argument names (since the above example requires that a process be started, see “startProcess” in the Supported Actions section of this document or execute the command “caitpamwscli.cmd /help action=startProcess”).

```

caitpamwscli.cmd [/i] user=<User Name> password=<Password>
[uri=<Web Service URI>] [version=<Web Service Version>]
action=startProcess processName=<Fully Qualified Process Name>
[startDate=<Date>] [startTime=<Time>] [param..<Parameter Name
1>=<Parameter Value 1>...param..<Parameter Name N>=<Parameter Value
N>] [attachmentParameterName=<Attachment Parameter Name>
attach.<Attachment Name 1>=<Attachment Location
1>...attach.<Attachment Name N>=<Attachment Location N>]

```

Simply add the argument names and corresponding values as name/value pairs, one per line to a text file:

```

user=user01
password=pwd01
uri=http://server:8080/itpam/soap
version=4.1
action=startProcess
processName=/Examples/Example 1
param..EmailAddress=user02@domain.com
attachmentParameterName=EmailAttachments
attach.EmailAttachment01=C:\Temp\Instruction.txt

```

Assuming the “action file” content was saved to “C:\Temp\Example.txt” the process could be started by executing the following command:

```
caitpamwscli.cmd actionFile=C:\Temp\Example.txt
```

Although using the “action file” made it possible to start the process passing required parameter values with a single argument, it should be clear that storing credentials in a simple text file would not be recommended. However, as noted earlier, command line arguments can be specified to supplement or supersede “action file” entries.

To avoid storing credentials, remove the entries from the “action file”. In the example the revised content of “C:\Temp\Example.txt” would be as follows:

```

uri=http://server:8080/itpam/soap

action=startProcess
processName=/Examples/Example 1
param..EmailAddress=user02@domain.com
attachmentParameterName=EmailAttachments
attach.EmailAttachment01=C:\Temp\Instruction.txt

```

The command to start the processing with the revised “action file” would be:

```
caitpamwscli.cmd actionFile=C:\Temp\Example.txt user=user01
password=pwd01
```

Finally, let's assume that, as recommended, the web service configuration arguments now included in the "action file" referenced a test environment. When moved to production, the file would need to be edited to reference the production URI and version.

To eliminate the need to reference to the web service URI and version in "action files" based on the environment (i.e., test versus production) use the "setServiceConfig" action to store the values globally for all commands issued using a specific instance of "caitpamwscli.cmd" (see

```
caitpamwscli.cmd action=setServiceConfig  
uri=http://server:8080/itpam/soap
```

The "action file" content would be revised as follows:

```
action=startProcess  
processName=/Examples/Example 1  
param.EmailAddress=user02@domain.com  
attachmentParameterName=EmailAttachments  
attach.EmailAttachment01=C:\Temp\Instruction.txt
```

The same "action file" and command:

```
caitpamwscli.cmd actionFile=C:\Temp\Example.txt user=user01  
password=pwd01
```

...could not be used in both the test and production environments (assuming "user01" was defined in both environments with the same password, otherwise the values for "user" and "password" would need to be changed).

Argument Super-Session

Argument values can be specified in several ways:

- As arguments when the "caitpamwscli.cmd" is issued (see Supported Actions)
- As entries in an "action file" (see Action File)
- As stored configuration values in the case of "uri" and "version" arguments (see Service Configuration and Authentication)
- Default values when applicable (see Supported Actions)

Argument value conflicts (such as multiple values for the same argument name specified in two or more ways) will be resolved in the following order:

- Value passed as an argument when the "caitpamwscli.cmd" is executed will supersede all other values
- Value specified as an entry in an action file will supersede store configuration or default values

- Stored configuration values or default values will be used only when applicable and not superseded by an “action file” entry or argument specified as argument when the “caitpamwscli.cmd” is issued

For example, assume the following:

- Web service URI was set to <http://Production:8080/itpam/soap> using the “setServiceConfig” action (see [Service Configuration and Authentication](#) for more details).
- Action file “C:\Temp\QA.txt” contains the entry “uri=<http://QA:8080/itpam/soap>” (see Action File)

Here is the command that is issued:

```
caitpamwscli.cmd actionFile=C:\Temp\QA.txt  
uri=http://TEST:8080/itpam/soap
```

Since the argument passed when the command is executed supersedes all other values, the targeted web service URI will be [“http://TEST:8080/itpam/soap”](http://TEST:8080/itpam/soap).

Change the command to:

```
caitpamwscli.cmd actionFile=C:\Temp\QA.txt
```

...and the targeted web service URI will be [“http://QA:8080/itpam/soap”](http://QA:8080/itpam/soap) since the “action file” entry supersedes stored configuration or default values.

Remove the entry “uri=<http://QA:8080/itpam/soap>” from the action file “C:\Temp\QA.txt” and issue the command:

```
caitpamwscli.cmd actionFile=C:\Temp\QA.txt
```

...and the target web service URI will be [“http://Production:8080/itpam/soap”](http://Production:8080/itpam/soap) (the stored value set using the “setServiceConfig” action).

Troubleshooting

Every effort was made to produce clear, specific exception messages. When executing interactively (see Interactive Execution), exception messages will be displayed on the console. In addition, all messages generated are logged to the “caitpamwscli.log” file in the “log” sub-directory relative to the location of the “caitpamwscli.cmd” executable file.

Typically, only informational, warning, error and fatal messages will be logged and, in most cases, these messages should be sufficient to identify the cause of the problem and make corrections. However, unexpected exceptions are always a possibility and it may be necessary to gather more verbose debug tracing information. To increase or decrease the minimum severity of messages logged add the following argument to any command or “action file”:

logLevel=<Minimum Severity of Messages to Logged>

...where severity is one of the following:

- FATAL
- ERROR
- WARN
- INFO
- DEBUG
- ALL

When severity “ALL” is specified, all messages will be written to the log file and displayed on the console. If no value or an invalid value is specified, the level will be set to the default (INFO).

The log file will rollover when the size reaches 2mb. The current and previous 9 log files will be retained.

Web Service Client

The CA Process Automation Utilities bundle also includes the Java libraries required to instantiate a web service client and to interact with the web service directly. To use the web service client in your Java project do the following:

1. Copy all the library (JAR) files from the CA Process Automation Utilities “lib” directory to a location that can be conveniently accessed and referenced from your project
2. Add all the library (JAR) files to the class path (build path) for your project
3. Include the following classes as needed:
 - com.ca.dso.itpam.webservice.client.ItpamWebServiceClientFactory
 - com.ca.dso.itpam.webservice.client.ItpamWebService Client

To assist those less familiar with Java, steps to create an example project follow using the popular Eclipse IDE.

Sample Eclipse Project

Possibly the best way to become familiar with the CA Process Automation web service client is to experiment with some sample code. The “doc” sub-directory of the CA Process Automation Utilities package contains a “ready to use” project for the popular Eclipse IDE. To successfully use the sample you should:

- Be familiar with the Eclipse development environment
- Have Eclipse 3.4 or higher installed
- Eclipse workspace you intend to use should be configured to use a Java 6 or Java 7 JRE

- Have a test environment with an instance of CA Process Automation installed and running (see Before You Begin)

Assuming the above...

1. Locate the “CA IT PAM Web Service Client Sample” folder in the “doc” sub-directory of the CA Process Automation Utilities package
2. Copy “CA IT PAM Web Service Client Sample” folder and contents to the Eclipse workspace folder you intend to use
3. Launch Eclipse and open the workspace
4. From the Eclipse menu select File, Import to launch the “Import” wizard
5. On the first panel of the wizard, expand the “General” node in the tree displayed, select “Existing Projects into Workspace” then click Next.
6. On the “Import Projects” panel click the Browse button associated with the “Select root directory” text box
7. Navigate to the workspace folder containing the expanded “CA IT PAM Web Service Sample” project folder

The “CA IT PAM Web Service Sample” project should appear in the “Projects” list.

8. Check the associated check box then click Finish.

The “CA IT PAM Web Service Sample” project should now appear in the Eclipse “Project Explorer” view.

The sample code will need to be modified slightly to set values specific to your CA Process Automation test environment. To do this:

1. Expand the “CA IT PAM Web Service Sample” node in the Eclipse “Project Explorer” tree
2. Expand the “src” folder node
3. Expand the “com.ca.dso.itpam.webservice.sample” package node
4. Double-click on the “Sample.java” node

The “Sample.java” class should appear in an Eclipse editor window.

5. Locate the lines listed below:

```
String strItpamUri = "http://<Server>:<Port>/itpam/soap";  
String strItpamVersion = "<Version>";  
String strItpamUser = "<User Name>";  
String strItpamPassword = "<User's Password>";
```

6. Replace the highlighted entries with the values that apply to your test environment then save the changes.

You are now ready to test.

1. From the Eclipse menu select Run, Run Configurations.
2. Expand the “Java Applications” node in the explorer tree on the left side of the “Run Configurations”

A run configuration called “Sample” should be listed.

3. Select the “Sample” run configuration node then click the Run button on the “Run Configurations” dialog box (bottom right).

The sample code should execute and the following message should appear in the Eclipse “Console” view (just below the editing pane).

```
Server status ok.
```

Assuming you were successful and are familiar with Java, JavaDoc is included for all exposed public methods. You may view the JavaDoc via the Eclipse IDE or, alternatively, as HTML, by using your browser to open the “index.html” in the “doc\CA IT PAM Web Service Client JavaDoc” sub-directory. Try your hand at invoking other web service methods by adding code to the sample project provided.

Https access using Truststore

If an Orchestrator has been configured to support only secure connections on https (usually on port 8443), the web services command line tools can still be used with that orchestrator. To do this, however, you will need to perform the following steps to create a truststore file that contains the orchestrator’s certificate.

1. Locate the orchestrator’s certificate

By default, the orchestrator certificate file is named “itpamcertificate.cer” and it is located in the following directory:

```
<install_directory>\server\c2o\.c2orepository
```

If, however, you are using your own certificate you will need to identify and locate it.

2. Use the java “keytool” to create a truststore file on the client system in an accessible path.

To do this, execute the Java “keytool” utility which is located in the <java_home>\bin directory. The syntax of the command is as follows:

```
Keytool -importcert  
-v -alias <alias>  
-file <Path & Name of Certificate file>  
-keypass <Password to Access the Keystore>
```

Where:

- <Alias> is the name of the certificate key to be added to the truststore. Use “keytool -list” to check the name of certificates that are already defined.
- <Keypass> is the password to be used to access the truststore file.

Note: When this command is executed you will be prompted for the keystore(certificate) password.

Once the trust store file has been created, and the trust store password is known (specified in the `-keypass` parameter above), these values can be provided to the web services command line tool. For example:

```
Caitpamwscli.cmd /i action=getServerStatus user=<orchestrator
username> password=<orchestrator password> action=<action name>
uri=https://<host>:8443/itpam/soap truststore=<path&name of
truststore file> truststorepassword=<truststorepassword>
```

The truststore *attribute* can be provided to the `setServiceConfig` action for storage in the config file, although *truststorepassword* will not be stored in the service config file. These parameters can also be stored in an action file if required.

If you specify URI that starts with “https”, the web service command line tool will generate an error if the truststore and truststorepasswords parameters have not been specified on the command line, config file or action file.

If you are using the web services tools from within your own java application or servlet, the truststore and truststore password have to be set using the following java code:

```
System.setProperty("javax.net.ssl.trustStore", ""<Path and Name
of Trust Store File>");
System.setProperty("javax.net.ssl.trustStorePassword", ""<Trust Store
Password>");
```

Changes from previous version

The following changes have been implemented since the previous original release.

1. Removed support of PAM 2.1 and 2.2

The web services tools no longer support CA Process Automation versions 2.1 and 2.2, as these of CA Process Automation have reached end of support.

2. *version* parameter is now optional. If *version* is not specified, the web services tools will obtain the version from CA Process Automation.
3. New input field in **ImportObject**

ImportObject supports a new input: *importOverwriteAction*.

4. New input fields in **StartRequest**:

StartRequest supports new inputs: *tagId* and *isAutoArchive*.

5. New input field in **GetStartRequestStatus**:

GetStartRequestStatus supports a new input: *tagId*.

6. New input fields in **StartProcess**

StartProcess supports new inputs: *tagId* and *isAutoArchive*.

7. New input field in **GetProcessStatus**

GetProcessStatus supports a new input: *tagId*. The user can now refer to a process instance by *processId* or *tagId*.

8. New action: **controlInstance**

This new action allows setting the archiving flag of a process or a start request form instance (and all its child instances that are not triggered in detached mode).

Known Limitations

CA Process Automation web services have and will continue to evolve. Since the CA Process Automation Utilities are designed to work with multiple releases, dependent upon the release you have, implementation of certain features may behave differently or not be supported. If present, consult the "Release Notes.mht" document distributed with the package for details.

Summary

When an integrated solution requires triggering and interacting with CA Process Automation processes you can use the CA Process Automation Utilities command line interface and/or the web services client to reduce the effort and time to value