TIBCO iProcess® Engine

Administrator's Guide

Software Release 11.6
January 2016

TIBCO®

Important Information

Contents

Figures

Tables

Preface

This guide describes how to administer iProcess Engine.

Topics

Related Documentation

This section lists documentation resources you may find useful.

TIBCO iProcess Engine Documentation

The following documents form the TIBCO iProcess Engine documentation set:

- *TIBCO iProcess Engine Installation* Read this manual for instructions on site preparation and installation.

- *TIBCO iProcess Engine Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

- TIBCO iProcess Suite Documentation This documentation set contains all the manuals for TIBCO iProcess Engine and other TIBCO products in TIBCO iProcess® Suite. The manuals for TIBCO iProcess Engine are as follows:

  — *TIBCO iProcess Engine Architecture Guide*

  — TIBCO iProcess Engine Administrator's Guides:

    TIBCO iProcess Engine Administrator's Guide

    TIBCO iProcess Objects Director Administrator's Guide

    TIBCO iProcess Objects Server Administrator's Guide

  — TIBCO iProcess Engine Database Administrator's Guides:

    TIBCO iProcess Engine (DB2) Administrator's Guide

    TIBCO iProcess Engine (Oracle) Administrator's Guide

    TIBCO iProcess Engine (SQL) Administrator's Guide

  — *TIBCO iProcess swutil and swbatch Reference Guide*

  — *TIBCO iProcess Engine System Messages Guide*

  — *TIBCO iProcess User Validation API User's Guide*

  — *LDAPCONF Utility User's Guide*

Other TIBCO Product Documentation

You may find it useful to read the documentation for the following TIBCO products:

- TIBCO ActiveMatrix BusinessWorks™

- TIBCO Business Studio™
- TIBCO Enterprise Message Service™
- TIBCO Hawk®
- TIBCO Rendezvous®

Typographical Conventions

TIBCO iProcess Engine can be run on both Microsoft Windows and UNIX/Linux platforms. In this manual, the Windows convention of a backslash (\) is used. The equivalent pathname on a UNIX or Linux system is the same, but using the forward slash (/) as a separator character.

UNIX or Linux pathnames are occasionally shown explicitly, using forward slashes as separators, where a UNIX or Linux-specific example or syntax is required.

Any references to UNIX in this manual also apply to Linux unless explicitly stated otherwise.

The following typographical conventions are used in this manual.

Table 1   General Typographical Conventions

Convention	Use
SWDIR	TIBCO iProcess Engine installs into a directory. This directory is referenced in documentation as *SWDIR*. The value of *SWDIR* depends on the operating system. For example, • on a Windows server (on the `C:` drive) if *SWDIR* is set to the `C:\swserver\staffw_nod1` directory, then the full path to the `swutil` command is in the `C:\swserver\staffw_nod1\bin\swutil` directory. • on a UNIX or Linux server if *SWDIR* is set to the `/swserver/staffw_nod1` directory, then the full path to the `swutil` command is in the `/swserver/staffw_nod1/bin/swutil` directory or the `$SWDIR/bin/swutil` directory. **Note**: On a UNIX or Linux system, the environment variable `$SWDIR` should be set to point to the iProcess system directory for the *root* and *swadmin* users.
`code font`	Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example: Use `MyCommand` to start the foo process.

Table 1   General Typographical Conventions (Cont'd)

Convention	Use
bold code font	Bold code font is used in the following ways: • In procedures, to indicate what a user types. For example: Type **admin**. • In large code samples, to indicate the parts of the sample that are of particular interest. • In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, MyCommand is enabled: MyCommand [**enable** | disable]
italic font	Italic font is used in the following ways: • To indicate a document title. For example: See *TIBCO ActiveMatrix BusinessWorks Concepts*. • To introduce new terms. For example: A portal page may contain several portlets. *Portlets* are mini-applications that run in a portal. • To indicate a variable in a command or code syntax that you must replace. For example: MyCommand *PathName*
Key combinations	Key name separated by a plus sign indicate keys pressed simultaneously. For example: Ctrl+C. Key names separated by a comma and space indicate keys pressed one after the other. For example: Esc, Ctrl+Q.
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Table 2   Syntax Typographical Conventions

Convention	Use
[ ]	An optional item in a command or code syntax. For example: MyCommand [optional_parameter] required_parameter

Table 2 Syntax Typographical Conventions (Cont'd)

Convention	Use
|	A logical OR that separates multiple items of which only one may be chosen. For example, you can select only one of the following parameters: `MyCommand param1 | param2 | param3`

{ }	A logical group of items in a command. Other syntax notations may appear within each logical group.
	
	For example, the following command requires two parameters, which can be either the pair `param1` and `param2`, or the pair `param3` and `param4`.
	
	`MyCommand {param1 param2} | {param3 param4}`
	
	In the next example, the command requires two parameters. The first parameter can be either `param1` or `param2` and the second can be either `param3` or `param4`:
	
	`MyCommand {param1 | param2} {param3 | param4}`
	
	In the next example, the command can accept either two or three parameters. The first parameter must be `param1`. You can optionally include `param2` as the second parameter. And the last parameter is either `param3` or `param4`.
	
	`MyCommand param1 [param2] {param3 | param4}`

Connecting with TIBCO Resources

How to Join TIBCO Community

TIBCO Community is an online destination for TIBCO customers, partners, and resident experts. It is a place to share and access the collective experience of the TIBCO community. TIBCO Community offers forums, blogs, and access to a variety of resources. To register, go to the following web address:

https://community.tibco.com

How to Access TIBCO Documentation

You can access TIBCO documentation here:

https://docs.tibco.com

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, contact TIBCO Support as follows:

• For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

   http://www.tibco.com/services/support

• If you already have a valid maintenance or support contract, visit this site:

   https://support.tibco.com

   Entry to this site requires a user name and password. If you do not have a user name, you can request one.

Chapter 1   # Controlling the iProcess Engine

This chapter describes basic operations for controlling iProcess Engine.

There are additional administration tasks that can be performed on TIBCO iProcess Workspace (Windows), such as case monitoring, managing users, and group and case administration. They are all described in *TIBCO iProcess Workspace (Windows) Manager's Guide.*

You can also control, start and stop iProcess Engine Process Sentinels and server processes using the swadm and swsvrmgr utilities located in the *SWDIR*\util directory. See Administering iProcess Engine Server Processes on page 105 for more information.

Topics

Starting iProcess Engine

The iProcess Engine server processes are controlled by Process Sentinels. Process Sentinels must be started first, they then control the start-up of the server processes. If you are using more than one server to host iProcess Engine (a node cluster), Process Sentinels must be started on each server.

Before you can start iProcess Engine, you must make sure that:

1. The iProcess database instance is running.

2. All required message queues are running.

3. The event manager is running. (This means that the event queues and agents are running.)

The following sections explain how to start the Windows (see Windows Version on page 2) and UNIX versions (see UNIX Version on page 4) of iProcess Engine.

Windows Version

In the Windows version, the iProcess Engine functions are provided by the iProcess *nodename* Process Sentinels service (where *nodename* is the name of your iProcess Engine installation).

By default, after having been started, Process Sentinel tests the event mechanism automatically and starts the iProcess Engine server auto-start processes.

> This behavior is controlled by the `PM_AUTO_BOOT` process attribute, only the value of the `PM_AUTO_BOOT` attribute is set to 1, the processes will be started automatically. See Administering Process Attributes on page 141 for more information.

You can start the Process Sentinels service in three different ways (unless you have installed iProcess Engine to a Windows cluster):

• At system startup - see An operating system group that gives you permissions to start a service, normally the Administrators group. on page 3.

• Manually, from the Windows Control Panel - see Manually Starting the Process Sentinels Service on page 4.

• Using the swstart.bat script located in the *SWDIR*\bin directory- see Using the swstart.bat Script on page 4.

To be able to start iProcess Engine, you must be logged in to either as an iProcess Engine Administrator user, or as a user who is a member of both:

- The iProcess Administrators local group (which gives you permissions on files and directories in *SWDIR*).

- An operating system group that gives you permissions to start a service, normally the Administrators group.

If you have installed iProcess Engine to a Windows cluster, use the Bring online service in the Microsoft Cluster Administrator to start iProcess Engine.

Do not attempt to start iProcess Engine on a Windows cluster by using Control Panel > Services or the `swstart` command located in the *SWDIR*\bin directory.

Configuring System Startup Behavior

When you install iProcess Engine, you choose whether or not the Process Sentinels service:

- Starts automatically on system startup (the default option).

- Needs to be started manually.

- Is disabled (cannot be started).

If you subsequently want to change this setting, do the following:

1. From the Start menu, select **Settings > Control Panel**.

2. Double-click the **Administration Tools** item.

3. Double-click the **Services** item. The Services dialog appears.

4. Select the **iProcess** *Nodename* **Process Sentinels** service item (where *Nodename* is the name of your iProcess Engine installation) and click the **Startup** button. The Service dialog appears.

5. Set the Startup Type item to:

   — Automatic, if you want the Process Sentinels service to start automatically on system startup.

   — Manual, if you want to manually start the Process Sentinels service (see the following).

   — Disabled, if you want to disable the Process Sentinels service.

Do not change any other options in the Service dialog. Doing so may cause iProcess Engine to fail.

Manually Starting the Process Sentinels Service

To manually start the Process Sentinels service:

1. From the Start menu, select **Settings > Control Panel**.

2. Double-click the **Administration Tools** item.

3. Double-click the **Services** item. The Services dialog appears.

4. Select the **iProcess** *Nodename* **Process Sentinels** service item (where *Nodename* is the name of your iProcess Engine installation).

5. Click the **Start** button. This will start the Process Sentinels service and the iProcess Engine server processes.

> You can use the Processes tab of the Windows Task Manager to view the processes as they start up. See Server Processes on page 107 for a list of processes that are started.

Using the swstart.bat Script

To start the Process Sentinels service using the `swstart.bat` script located in the *SWDIR*\bin directory:

1. Start the Process Sentinels by using the following command:

   ```
   swstart -p
   ```

2. Start the iProcess Engine server processes normally by using the following command:

   ```
   swstart
   ```

   Or, start the iProcess Engine server processes in quick start mode by using the following command:

   ```
   swstart -q
   ```

> When the iProcess Engine server is restarted or quick started, the iProcess Objects Server processes apply the RESTART_SPO_CACHE_PROC attribute to cache a specified number of procedure versions, and the WIS processes apply the RESTART_WIS_CACHE_THRESHOLD attribute to cache the work queues at a specified work item number. See RESTART_WIS_CACHE_THRESHOLD on page 215 and RESTART_SPO_CACHE_PROC on page 320 for more information.

UNIX Version

In the UNIX version, the iProcess Engine functions are provided by the "worker" and "watcher" Process Sentinel processes.

By default, after having been started, Process Sentinel tests the event mechanism automatically and starts the iProcess Engine server auto-start processes.

This behavior is controlled by the `PM_AUTO_BOOT` process attribute, only the value of the `PM_AUTO_BOOT` attribute is set to `1`, processes will be started automatically. See Administering Process Attributes on page 141 for more information.

Starting the Process Sentinels

You need to start Process Sentinels on each server in your iProcess Engine.

To start Process Sentinels on a server:

1. Log in to iProcess Engine as a background user.

2. Enter the command:

   SWDIR/bin/swstart -p

If you add this command to your UNIX start-up routine script the Process Sentinels will always be running on startup.

Starting the Server Processes

Use the `swstart` script located in the $*SWDIR*/bin directory to start all the required server processes.

If you are using a node cluster, you can run this script from any server that is part of the node cluster and it will start all the processes on all of the servers in iProcess Engine.

To start the iProcess Engine server processes, follow these steps:

1. Log in to iProcess Engine as a background user.

2. Run the following command to start iProcess Engine server processes normally:

   SWDIR/bin/swstart

   Or, run the following command to start iProcess Engine server processes in quick start mode:

   SWDIR/bin/swstart -q

As each server process is started, a start-up message is displayed.

When the iProcess Engine server is restarted or quick started, the iProcess Objects Server processes apply the RESTART_SPO_CACHE_PROC attribute to cache a specified number of procedure versions, and the WIS processes apply the RESTART_WIS_CACHE_THRESHOLD attribute to cache the work queues at a specified work item number. See RESTART_WIS_CACHE_THRESHOLD on page 215 and RESTART_SPO_CACHE_PROC on page 320 for more information.

Stopping iProcess Engine

The following sections explain how to stop the Windows (see below) and UNIX versions (see UNIX Version on page 8) of the iProcess Engine.

Windows Version

The iProcess Engine functions are provided by the iProcess *nodename* Process Sentinels service (where *nodename* is the name of your iProcess Engine installation).

You can stop the Process Sentinels service either:

- Manually, from the Services dialog. For more information, see below.

- Using the `swstop.bat` script located in the *SWDIR*\bin directory. For more information, see Using the swstop.bat Script on page 8.

⚠️ If you have installed iProcess Engine to a Windows cluster, use the Take offline service in the Microsoft Cluster Administrator to stop the Process Sentinels service.

Do not attempt to stop the Process Sentinels service on a Windows cluster by using Control Panel > Services or the `swstop` script located in the *SWDIR*\bin directory.

Manually Stopping the iProcess Engine

To stop iProcess Engine:

1. Make sure that all iProcess Workspace users are logged out from iProcess Engine.

2. From the Start menu, select **Settings > Control Panel**.

3. Double-click the **Administration Tools** item.

4. Double-click the **Services** item.The Services dialog appears.

5. Select the **iProcess *Nodename* Process Sentinels** service item (where *Nodename* is the name of your iProcess Engine installation).

6. Click the **Stop** button. This will stop the Process Sentinels service and the iProcess Engine server processes.

Using the swstop.bat Script

To stop iProcess Engine using the `swstop.bat` script located in the *SWDIR*\bin directory:

1. Make sure that all iProcess Workspace users are logged out from iProcess Engine.

> If you cannot or do not want to do this for any reason, you can force iProcess Engine to shut down even if users are still logged in. See Forcing the iProcess Engine to Shutdown on page 10 for more information.

2. Stop the iProcess Engine server processes using the following command:
   swstop

3. Stop the Process Sentinels using the following command:
   swstop -p

UNIX Version

To stop iProcess Engine you must:

1. Stop the server processes.

2. Stop Process Sentinels.

Stopping the Server Processes

Use the `swstop` script, which is located in the $*SWDIR*/bin directory to stop all the required server processes.

> If you are using a node cluster, you can run this script from any server that is part of the node cluster and it will stop all the processes on all of the servers in iProcess Engine.

To stop the iProcess Engine server processes:

1. Log in to iProcess Engine as a background user.

2. Make sure that all TIBCO iProcess Workspace users are logged out from iProcess Engine.

> If you cannot or do not want to do this for any reason, you can force iProcess Engine to shut down even if users are still logged in. See Forcing the iProcess Engine to Shutdown on page 10 for more information.

3. Locate the *SWDIR*\bin directory, and enter the following command:

swstop

A summary of the shutdown process is displayed as the processes are stopped. The example is shown below.

```
Attempting to stop 17 processes
Machine ID Proc Name      Proc   Status          Comment
                          Inst
1          BG             1      SHUTTING DOWN   Normal Shutdown
1          BG             2      SHUTTING DOWN   Normal Shutdown
1          BG             3      SHUTTING DOWN   Normal Shutdown
1          BG             4      SHUTTING DOWN   Normal Shutdown
1          BGPREDICT      1      SHUTTING DOWN   Normal Shutdown
1          DIRECTOR       1      SHUTTING DOWN   Normal Shutdown
1          DLMGR          1      SHUTTING DOWN   main calling shutdown
1          IAPJMS         1      SHUTTING DOWN   IAPJMS Process Shutdown
1          RPCBG          1      SHUTTING DOWN   Normal Shutdown
1          RPC_TCP_LI     1      SHUTTING DOWN   RPC server shutdown
1          RPC_UDP_LI     1      SHUTTING DOWN   RPC server shutdown
1          SPO            1      SHUTTING DOWN   Normal Shutdown
1          WIS            1      SHUTTING DOWN   Normal Shutdown
1          WIS            2      SHUTTING DOWN   Normal Shutdown
1          WISMBD         1      SHUTTING DOWN   WISMBD normal shutdown
1          WISMBD         2      SHUTTING DOWN   WISMBD normal shutdown
1          WQS            1      SHUTTING DOWN   WQS Normal shutdown
Current System Status : 'STOPPED'
```

Stopping Process Sentinels

You can also use the swstop script, which is located in the *SWDIR*\bin directory, to stop Process Sentinels.

If you are using a node cluster, you can run this script from any server that is part of the node cluster and it will stop Process Sentinels on all of the servers in iProcess Engine.

To stop the Process Sentinels:

1. Log in to iProcess Engine as a background user.

2. Make sure that all TIBCO iProcess Workspace users are logged out from iProcess Engine.

   If you cannot or do not want to do this for any reason, you can force iProcess Engine to shut down even if users are still logged in. See Forcing the iProcess Engine to Shutdown on page 10 for more information.

3. Locate the *SWDIR*\bin directory, and enter the following command:

   swstop -p

   which displays the following message:

   ```
   Please wait, stopping process sentinels.
   ```

Forcing the iProcess Engine to Shutdown

Normally, when you want to shut down iProcess Engine, you must first get all users to log out of iProcess Suite.

However, you can force iProcess Engine to shut down, even if there are users logged in. There are two ways you can do this:

• Using the swstop command from a command prompt. See below.

• Using the swstop command from the Services dialog. For more information, see Enable Forced Shutdown from the Services Dialog on page 11.

Using the swstop Command

You can use the following command to force the iProcess Engine to shut down:

SWDIR\bin\swstop [-f [*timeout*]]

where:

• -f issues a forced shutdown event to shutdown the iProcess Engine processes, whether or not there are users logged in.

• *timeout* is the period, in seconds, to wait before shutting down iProcess Engine. If *timeout* is omitted, a default timeout value of 300 (5 minutes) is used. If a subsequent swstop -f *timeout* command is issued before the first *timeout* value has expired, the *timeout* will be reset to the new value if the new *timeout* value is smaller. You cannot increase the timeout period - a larger timeout value will be ignored.

Note that:

• On a UNIX system, you must be logged in to iProcess Engine as a background user to use this command.

- When the forced shutdown command is issued, a message is sent to all users informing them that the system will be stopped in *timeout* seconds.

- Make sure you save any changes to procedure definitions before enabling the forced shutdown otherwise any such changes will be lost.

- If any released work items have not been processed by the time the background processes shut down, these changes are queued and processed when iProcess Engine restarts.

For example:

- The following command causes iProcess Engine to shut down after the default delay of 300 seconds.

  ```
  swstop -f
  ```

- The following command causes iProcess Engine to shut down after a delay of 3 minutes.

  ```
  swstop -f 180
  ```

- If this command is issued 1 minute after the previous example, the delay before shutdown will be reset to 30 seconds.

  ```
  swstop -f 30
  ```

Enable Forced Shutdown from the Services Dialog

To force iProcess Engine to shut down from the Services dialog, you must:

Create a new string value called SERVICE_STOP_PARAMS in the Windows Registry and enter the swstop command as the string value data. For more information, see Creating the SERVICE_STOP_PARAMS String Value on page 11.

Once you have created the SERVICE_STOP_PARAMS string value, when you stop Process Sentinels from the Services dialog, the Process Sentinels are shut down using the swstop command with the parameters you specified. For more information, see Manually Stopping the iProcess Engine on page 7 for more information.

To disable the forced shutdown from the Services dialog, either:

- Delete the SERVICE_STOP_PARAMS string value from the Windows Registry, or

- Delete the value data from the SERVICE_STOP_PARAMS string value in the Windows Registry.

Creating the SERVICE_STOP_PARAMS String Value

To create the SERVICE_STOP_PARAMS string value:

1. From the Start menu, click the **Run** button. The Run dialog appears.

2. In the Open: field, type **regedit**, and click the **OK** button. The Registry Editor window appears.

3. Navigate to the registry list where the SERVICE_STOP_PARAMS string value is to be located, at:

   RegistryLocation\Staffware plc\Staffware Server\Nodes*nodename*

   where:

   — *RegistryLocation* is either \HKEY_LOCAL_MACHINE\SOFTWARE, if you are running the iProcess Engine on a 32-bit machine, or \HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node, if you are using a 64-bit machine.

   — *nodename* is the name of the iProcess Engine installation.

4. From the Edit menu, select **New > String Value**. A new value named New Value #1 is created.

5. Right-click the **New Value #1** item and click the **Rename** button. Rename the New Value #1 item to the SERVICE_STOP_PARAMS item.

6. Right-click the **SERVICE_STOP_PARAMS** item and click the **Modify** button. The Edit String dialog appears.

7. Enter the following value in the Value Data: box:

   swstop *[-f [timeout]] [-n retries]*

   where:

   — -f issues a forced shutdown event to shutdown the iProcess Engine processes, whether or not there are users logged in.

   — *timeout* (optional) is the period, in seconds, to wait before shutting down the iProcess Engine. If *timeout* is omitted, a default timeout value of 300 seconds (5 minutes) is used. The *timeout* value can be a numeric value between 0 - 7200. If a value less than 0 is entered, the default value of 300 seconds (5 minutes) is used. If a value greater than 7200 is entered, the value of 7200 seconds is used.

   — -n *retries* (optional) is the maximum number of times the forced shutdown command will be retried, if required. The *retries* value can be a numeric value of 0 or greater. The re-issue of the forced shutdown command occurs if any of the processes have not shutdown. This overcomes the problem of

an event being lost in the event system and the process not receiving the shutdown message.

If all the processes have still not completely shutdown after the number of retries then a final forced shutdown is issued.

If *retries* is omitted, (or if a value of less than 0 is entered), a default value of 0 is used. This means that a forced shutdown is issued after the timeout period and is not re-tried. Any processes that have not shutdown are forced to shutdown.

If all the processes have still not completely shutdown after the final forced shutdown is issued because, for example, a process has hung, then these processes will have to be shutdown manually through the Task Manager or by restarting the machine that is hosting iProcess Engine.

Note that:

- When the forced shutdown command is issued, a message is sent to all users informing them that the system will be stopped in *timeout* seconds.

- After 2 minutes, Microsoft Windows issues the following message:

```
Could not stop the iProcess nodename Process Sentinels service
on Local Computer. Error 1053: The service did not respond to
the start or control request in a timely fashion
```

where *nodename* is the name of your iProcess Engine installation. This is a warning only. Click the **OK** button, Process Sentinels continue to shutdown.

- Make sure you save any changes to procedure definitions before enabling the forced shutdown otherwise any such changes will be lost.

- If any released work items have not been processed by the time the background processes shut down, these changes are queued and processed when iProcess Engine restarts.

Configuring iProcess Engine Events Server

This section is only relevant if you are running iProcess Engine on a Windows platform.

The iProcess Engine uses a publish/subscribe event mechanism to handle the following inter-process tasks:

- Notifying processes to update caches.
- Synchronization of process startup and shutdown.

Events are handled by the iProcess Events COM+ application. All processes that want to subscribe to events register with the COM+ application.

The iProcess Events COM+ application is installed on the same machine as iProcess Engine. If you are using a node cluster architecture, the event server is set to be the machine on which you installed the master server. The event server name is stored in the following registry key:

> *RegistryLocation*\Staffware plc\Staffware Server\Nodes*nodename*\
> IEL_EVENT_SERVER

where:

- *RegistryLocation* is either \HKEY_LOCAL_MACHINE\SOFTWARE, if you are running iProcess Engine on a 32-bit machine, or \HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node, if you are using a 64-bit machine.
- *nodename* is the name of iProcess Engine installation.

If performance becomes an issue, TIBCO recommends that you dedicate one machine in the cluster, which is not running any iProcess Engine processes, to host the iProcess events. This reduces the load on the machine. To do this, you will need to edit the registry entry and change the *nodename* to the name of the new iProcess events server. You will need to restart the machine after doing this.

Incorrectly editing the registry can severely damage your system. Make sure that you edit ONLY the indicated registry entry.

Make sure that the iProcess Events application starts successfully on the new iProcess events server before starting iProcess Engine, otherwise the system will not function correctly.

Configuring iProcess Engine Time Zone

An iProcess Engine that is installed on a server operating in one time zone may be accessed by TIBCO iProcess Workspace users who are operating in different time zones. For example, a company's office in California (Pacific Standard Time, GMT-08:00) may want to run cases of procedures that are hosted on a server running in the company's administrative centre in Washington D.C. (Eastern Standard Time, GMT-05:00).

This will lead to a disparity between time stamps created by the server (which will use its local time) and their subsequent interpretation by the computers hosting the TIBCO iProcess Workspaces. This disparity will affect:

- work item time stamps
- audit trail time stamps
- deadline time stamps and processing
- priority escalation of work items
- date/time settings for participation and redirection.

To avoid this disparity, you can configure the iProcess Engine processes to operate in the same time zone as the clients. Note that:

- The time zone is set for ALL processes generated by iProcess Engine. Different processes on the same iProcess Engine cannot use different time zones, even if they are running on different servers. To continue the example above, if iProcess Engine is configured to run in Pacific Standard Time it can only administer sites in that time zone without discrepancy.

- If multiple iProcess Engines are running on the same physical hardware (which can be either a single node or a node cluster), each iProcess Engine can operate in its own designated time zone.

Setting the Time Zone

The time zone used by iProcess Engine is stored using the TIMEZONE process attribute. Its value must be a valid time zone recognized by the operating system. See TIMEZONE on page 175.

By default, the TIMEZONE attribute is not set, and iProcess Engine uses the host server's local time.

You can set the value of TIMEZONE using the *SWDIR*\util\swadm utility. If you want to:

- See what time zone iProcess Engine is currently operating in, use the SHOW_ALL_ATTRIBUTES command. See Display All Process Attributes on page 143.

- Configure iProcess Engine to operate in a different time zone, use the SET_ATTRIBUTE command. See Set a Process Attribute on page 144.

- Reset iProcess Engine to use the host server's local time, use the DELETE_ATTRIBUTE command. See Delete a Process Attribute on page 145.

Using the *SWDIR*\util\swadm utility to change the time zone triggers an event informing the server processes that the time zone has changed. iProcess Engine does not need to be restarted for the change to take effect.

Error Handling

Most errors encountered by TIBCO iProcess Engine are reported directly to the user when they occur. Where this is not possible:

- A suitable error message is written to the sw_warn or sw_error file located in the *SWDIR*\logs directory. See iProcess Engine Log Files on page 439 for more information.

- A "System Information" message is sent to iProcess Engine Administrator, informing them that the file has been created.

See *TIBCO iProcess Engine System Messages Guide* for detailed information about the system error and warning messages that can be returned by iProcess Engine in the sw_warn or sw_error file located in the *SWDIR*\logs directory.

Each computer in a node cluster creates its own error files so you have to make sure to check each server for the sw_error and sw_warn files. See iProcess Engine Log Files on page 439 for more information.

In all cases where a resolution cannot be achieved on site, contact TIBCO Support for further assistance.

Chapter 2 **Using the iProcess Engine Configuration Files**

This chapter describes various TIBCO iProcess Engine configuration files.

See Tuning the iProcess Engine Using SWDIR\etc\staffcfg Parameters on page 45 for information about using the `staffcfg` file, which is located in the *SWDIR*\etc directroy to configure your iProcess Engine.

Topics

SWDIR\swdefs

The swdefs file, which is located in the *SWDIR* directory, is the main system configuration file.

The contents of this file are determined at installation time, and in general should not be changed.

The following table describes the contents of the swdefs file.

Table 3   Contents in the swdefs File

Line	Example	Description
1	i16.0-x(0.0)	TIBCO iProcess Engine version
2	pro	Background (iProcess Engine background) user
3	swadmin	System administrator (iProcess Engine Administrator)
4	D:\swbkp (Windows) or \usr\swbkp (UNIX)	Path to backup directory. **Note**: This is not used by TIBCO iProcess Engine.
5	NULL	Not used
6	swattach	Users' attachments subdirectory. **Note**: This is not used by TIBCO iProcess Engine.
7	NULL	Reserved. Do not change this entry
8	staffw_nod1	Nodename of this TIBCO iProcess Engine
9	English	System default language
10	391870	Server\Server RPC service number **Note**: This is not used by TIBCO iProcess Engine.
11	391875	Client\Server RPC service number
12	3.0	Server\Server RPC version

SWDIR\etc\language.lng\staffico

The staffico file, which is located in the *SWDIR*\etc\language.lng directory, specifies which tools are available to an iProcess user, depending on the value of the MENUNAME attribute.

Tools are displayed in the Work Queue Manager, as items on the Tools menu and as toolbar buttons.

If you want to modify the staffico file, which is located in the *SWDIR*\etc\language.lng directory:

1. Log in as a user who (on Windows) is a member of the iProcess Administrators group or, (on UNIX) as a user root.

2. Edit the file as required.

The changes take effect when a user next logs in. (Users who are already logged in will need to log out and log back in again.)

File Format

The file contains one section per defined MENUNAME attribute. Each section contains one entry per tool available for that MENUNAME.

Lines that begin with a ';' (semi-colon) character are treated as comments. Blank lines are ignored.

Section heading   Tools entry   The help and wqupdate entries are
                                         ignored in Work Queue Manager.

```
[admin]
casestart=1,1,&Case Start
audittrail=2,1,Case &Administration
help=3,1,iProcess Help
wqupdate=3,2,Update Queues
procdefn=1,2,Process &Definer
EXE,mainmenu.exe,NORM=2,2,Process Ad&ministrator
```

Tools are displayed in Work Queue Manager
in the order that they are listed in the section.

Tools Entry Format

Each tools entry has the following format:

Tool_Definition=xpos,ypos,description

where:

• *Tool_Definition* is one of the following:

Tool_Definition	Tool Description
CaseStart	Displays the Case Start dialog so that the user can start a case.
AuditTrail	Displays the Case Administration dialog, so that the user can perform administration tasks such as closing or purging cases and viewing audit trails of cases.
ProcDefn	Starts TIBCO iProcess Modeler.
EXE,*program*, NORM	Starts an executable *program*.exe. **Note**: By default, an EXE entry is provided to start TIBCO iProcess Administrator (mainmenu.exe).
RS,*procname*, *stepname*	Runs a caseless form for procedure *procname* and step *stepname*.
SWEIS[,*procname*, *EISobject*]	Runs an EIS report for procedure *procname* and report *EISobject*. If the *procname* and *EISobject* parameters are omitted the Run EIS Report dialog is displayed, from which the user can choose an EIS report to run.
SWIP	Starts TIBCO iProcess Monitoring.

The Help and WQUpdate entries are no longer used.

- *xpos*, *ypos* defines the horizontal (column) and vertical (line) position of the icon in the Tools window. (1,1 is the top, left hand side of the window.) Coordinates outside the range 1-10 are ignored.

  The xpos and ypos parameters are ignored because the Tools window is no longer supported. Tools are listed in the Tools menu and button bar in the order that they are listed in the section.

- *description* is the text that appears in the Tools menu and as button help in Work Queue Manager.

  description can be up to 40 characters long. Any text beyond this is truncated. The ampersand character (&) can be used to define a shortcut key for the tool. The character that follows the ampersand will appear underlined in the Tools menu. If you want to insert an actual ampersand character in the description, you must precede it with another ampersand character (&&).

SWDIR\etc\staffpms

The staffpms file, which is located in the *SWDIR*\etc directory, specifies a number of different configuration options.

⚠️ The contents of this file are determined at installation time, and should not be changed other than as described in this section.

To modify the staffpms file:

1. Log in as a user who (on Windows) is a member of the iProcess Administrators group or, (on UNIX) as a user root.

2. Edit the file as required.

3. Ask all users to log out of iProcess Suite, then stop and restart iProcess Engine.

Specifying if Client Passwords are Required on Login

Character 4 of line 4 specifies whether or not iProcess users need to give their password to log into this TIBCO iProcess Engine node.

Y0N**N**5YNY??0AY

If this character is:

- Y, iProcess users must supply their password when they log in to this TIBCO iProcess Engine node.

- N, passwords are not required on login.

Enabling Multiple Logins

Character 13 of line 4 specifies whether or not multiple logins to this TIBCO iProcess Engine are enabled.

Y0NN5YNY??0A**Y**

⚠️ This character must be set to Y. Multiple logins must be enabled for iProcess Engine to operate.

Specifying the Working Week

By default, all date calculations in iProcess use a 5-day working week of Monday to Friday. However, if a procedure has the Use working days flag un-set, a 7-day working week is used instead for cases of that procedure.

> The Use working days flag is set in the Procedure Manager, on the Status tab of the Properties dialog. For more information see "Use Working Days Flag" in *TIBCO iProcess Modeler Procedure Management*.

Line 5 ends with a 7-character string that defines the working week. There is one character for each day of the week, running from Sunday (on the left) to Saturday (on the right). Y indicates that the day is a working day, N indicates that it is a non-working day. The default entry specifies a working week of Monday to Friday, as shown below.

```
%2d\%2d\%4d\/\%s%s %s, %s\dmy\wdmy\%2d:%2d\:\ AM\ PM\Week\NYYYYYN
```

You can edit this string to change the specification of the working week that iProcess uses when calculating dates (for procedures which have the Use working days flag set). For example, to specify a 5 day working week of Sunday to Thursday, with Friday and Saturday being non-working days, change line 5 to:

```
%2d\%2d\%4d\/\%s%s %s, %s\dmy\wdmy\%2d:%2d\:\ AM\ PM\Week\YYYYYNN
```

To specify a six day working week of Monday to Saturday, with Sunday being a non-working day, change line 5 to read:

```
%2d\%2d\%4d\/\%s%s %s, %s\dmy\wdmy\%2d:%2d\:\ AM\ PM\Week\NYYYYYY
```

Changing the Date Format Using the staffpms File

Line 5 of the `staffpms` file determines how the date is displayed in iProcess Engine. (The following example is for a iProcess Engine for Windows).

```
%2d/%2d/%4d\/\%s%s %s, %s\dmy\wdmy\%2d:%2d\:\ AM\ PM\Week\NYYYYYN
```

Individual entries are separated by a backslash character (\). The following table describes the meaning of each entry.

Table 4   The Meaning of Each Entry for Changing the Date Format Using the staffpms File

Position	Example	Description
1	%2d/%2d/%4d	The number of characters used to specify each component of the date. For example, 2 characters for day, 2 for month and 4 for year.

Table 4   The Meaning of Each Entry for Changing the Date Format Using the staffpms File (Cont'd)

Position	Example	Description
2	/	The date delimiter.
3	%s%s %s, %s	Not used.
4	dmy	The order of the date format.
5	wdmy	Not used.
6	%2d:%2d	The time format. The default is 24 hour format, for example: 15:12.
7	:	The time delimiter.
8	AM	Not used.
9	PM	Not used.
10	Week	Not used.
11	NYYYYYN	The definition of the working week, running Sunday to Saturday. Y indicates a working day, N a non-working day. For example, Monday to Friday. For more information, see Specifying the Working Week on page 26.

Changing the Order of the Date Format

To change the format, for example, to yyyy/mm/dd:

1.   Amend the date order entry (position 4) to be ymd.

2.   Amend the number of characters entry (position 1) to be %4d/%2d/%2d.

%4d/%2d/%2d\/\%s%s %s, %s**ymd**\wdmy\%2d:%2d\:\ AM\ PM\Week\NYYYYYN

Changing the Date Delimiter

To change the date delimiter, for example to a hyphen character, amend the number of characters (position 1) and date delimiter (position 2) entries as shown.

%2d-%2d-%4d\-\%s%s %s, %s\dmy\wdmy\%2d:%2d\:\ AM\ PM\Week\NYYYYYN

Setting Database Connection Options

Line 9 contains the settings that iProcess Engine uses to connect to the database. (The following example is for a TIBCO iProcess Engine for Windows).

```
3\swpro\swuser\swpro\\sw-servers\0
```

Individual entries are separated by a backslash character (\). The following table describes the meaning of each entry.

Table 5   Setting Database Connection Options in the staffpms File

Position	Example	Description	Notes
1	3	iProcess Engine type	This field must always be 3 for a database version.
2	swpro	iProcess Engine database background user	The name of the database login (for SQL Server or Oracle) or UNIX account (for DB2) that iProcess Engine uses for background access to the iProcess Engine database schema. **Note**: This login/account must be the same login/account as the iProcess Engine database schema owner (specified in position 4).
3	swuser	iProcess Engine database user	The name of the database login (for SQL Server or Oracle) or UNIX account (for DB2) that iProcess Engine uses for other access to the iProcess Engine database schema.
4	swpro	iProcess Engine database schema owner	The name of the database login (for SQL Server or Oracle) or UNIX account (for DB2) that owns the iProcess Engine database schema.
5	null	Oracle database TNS identifier	The Oracle TNS identifier that iProcess Engine uses to connect to the Oracle instance holding the iProcess Engine database tables. This is the appropriate SERVICE_NAME entry in the tnsnames.ora file, which is located in the *ORACLE_HOME*\network\admin directory). If a TNS identifier is not defined, iProcess Engine will attempt to connect to the local Oracle instance, and Oracle Transparent Application Failover (TAF) will not be supported. **Note**: This entry is only used if iProcess Engine uses an Oracle database.

Table 5   Setting Database Connection Options in the staffpms File (Cont'd)

Position	Example	Description	Notes
6	`sw-servers`	ODBC Data Source (SQL) or Database Alias (DB2)	The name of the ODBC data source (for SQL Server) or Database Alias (for DB2) that iProcess Engine uses to connect to the database. **Note**: This entry is not used if iProcess Engine uses an Oracle database.
7	`0`	Reserved	This field is reserved for future use by iProcess Engine.

For more information about connecting to databases, see the appropriate iProcess Engine installation guide.

Changing Database Connection Passwords

If you need to change the passwords that the *iProcess Engine database schema owner* or *iProcess Engine database user use* to connect to the database, follow this procedure:

1. Log in to iProcess Engine as an Administrator.

2. Stop iProcess Engine.

3. Enter the following command:

   SWDIR**util****swconfig -u**

   The following prompt is displayed:

```
================================================================
              TIBCO(R) iProcess Suite  - Configuration Utility

                  Copyright (c) 2001-2012, TIBCO Software Inc.

================================================================
Please enter a new Background User Password, ('Q' to quit)
```

4. Enter the new password for a iProcess Engine database schema owner (the login/account defined in position 4 of line 9 of the `staffpms` file).

   The following prompt is displayed.

```
Please enter a new Foreground User Password, ('Q' to quit) :
```

5. Enter the new password for a iProcess Engine database user (the login/account defined in position 3 of line 9 of the `staffpms` file).

    The `swconfig` utility terminates and displays the following message.

    ```
    Now log onto the Database and change the passwords
    ```

6. Change the corresponding passwords for these users in the database (for Oracle or SQL Server) or UNIX (for DB2). See your database/UNIX documentation for more information about how to do this.

7. Restart iProcess Engine.

Controlling Access to the iProcess Engine (for UNIX)

This section only applies to the UNIX version of the iProcess Engine. It is not relevant to the Windows version.

Line 12 contains three settings (at the end of the line) that control access to iProcess Engine.

1\GROUPNAME\0\666**swuser****staffwar****7**

Individual entries are separated by a backslash character (\). The following table describes the meaning of each entry.

Position	Example	Description
1	1	Reserved for internal use - do not change.
2	GROUPNAME	Reserved for internal use - do not change.
3	1	Reserved for internal use - do not change.
4	666	Reserved for internal use - do not change.
5	swuser	The iProcess RPC Server account name. The default value is swuser.
6	staffwar	The iProcess group name. The default value is staffwar.
7	7	The iProcess security umask value, which controls "world" access to iProcess files in and under *SWDIR*. "World" permissions on each file installed by or created by the iProcess Suite are set to the iProcess group name permissions for the file, modified by this umask value. For example, if this value is: • 7 for high security. "World" has no access to iProcess files in and under *SWDIR*. This is the default. • 0 for low security. "World" has the same access to each file in and under *SWDIR* as the staffwar group.

To change the iProcess RPC Server account name, iProcess group name, or iProcess security umask value at any time after installation, do the following:

1. Log in to iProcess Engine as a background user.

2. Stop iProcess Engine (if it is running).

3. Change the appropriate value on line 12 of the `staffpms` file.

4. Run `fixperms`,which is located in the *SWDIR*\bin directory, to reset the ownership and permissions information on all files in and under *SWDIR*.

5. Restart iProcess Engine.

The implications of these security values in `staffpms` are:

- You must be logged in to iProcess Engine as a background user to start or stop the iProcess Engine. See Starting iProcess Engine on page 2.

- All iProcess processes run with the UID of an iProcess Engine background user, even if the process is started by root. The only exceptions are the `runcmd` utility, which is located in the *SWDIR*\util directory, and the `RPC_UDP_LI` process, which run as root.

- All iProcess files and directories (that is, all files in and under *SWDIR*) are owned by either root or the iProcess Engine background user. Their group ID is set to the iProcess group (staffwar).

- "World" access to iProcess files and directories is restricted. On a new installation, world has no access (security umask is set to `7`).

- All iProcess users who need access to iProcess files and directories must be members of the iProcess group (staffwar). For example, users who need to run `swutil`, or to use the `SERVERRUN` commands that access files under *SWDIR*.

Specifying How iProcess Validates Users

Lines 15 and 16 of the `staffpms` file define whether and how iProcess uses the integral User Validation API provided with the `LDAPCONF` utility.

Specifying an External User Validation Package

Line 15 defines whether the iProcess Suite validates users against O/S user accounts (the default), or against an external validation package developed using the TIBCO iProcess User Validation API.

This line is optional. If you are using the default method of validating users against O/S accounts, line 15 should be blank.

If you want to validate users against an external validation package, line 15 must contain the full pathname of the user validation package (a DLL file on Windows, a shared library on UNIX). Note that:

- The pathname must contain a leading drive letter and UNIX style separators (/).

- Variables such as *SWDIR* are not supported in this parameter.

The following example (for a iProcess Engine for Windows) specifies that user validation will be performed against the `swuvamod.dll` file in the `D:/iProcess/staff200/lib` directory.

```
d:/iProcess/staff200/lib/swuvamod.dll
```

The following example specifies that user validation will be performed against the `UVAPI.dll` file in the `C:/Tibco/iprocess_nod1/util` directory, and that the proxy users are supplied by swadmin.

```
C:/Tibco/iprocess_nod1/util/UVAPI.dll\N\swadmin\\\
```

For more information about how to:

- Develop an external validation package, see *TIBCO iProcess User Validation API User's Guide*.

- Install an external validation package, see installation guide for *TIBCO iProcess User Validation API*.

- Use the integral user validation API provided with LDAPCONF, see *LDAPCONF Utility User's Guide*.

Specifying a Proxy User

Line 16 defines which proxy operating system user ID is used by the LDAP integral user validation API.

This line applies only to UNIX and Linux systems.

This line is optional. If you are not using the LDAP integral user validation API, line 16 should be blank.

Configuring Signals That Threads are Blocked From Receiving

Line 17 allows you to specify threads to be blocked from receiving signals.

For example, to add blocking for the SIGALRM (14) signal the last few lines of the file will look like this:

```
QPARAM1\QPARAM2\QPARAM3\QPARAM4\CP_V...
        <blank line - unless UVAPI is configured>
        <blank line>
        14
```

If you wish to add more signals to the list they will need to be separated by the '\' character:

```
QPARAM1\QPARAM2\QPARAM3\QPARAM4\CP_V...

        <blank line - unless UVAPI is configured>

        <blank line>

        14\13\12\\
```

If the line is blank or there are no signal IDs set then the default behaviour is not to block the threads from receiving signals.

Configuring Log Files Directory

To make the log files directory configurable, specify the directory in line 19.

Ensure that the log files directory you specify is available and has the same permissions to access it as the original log files directory.

To configure the log files directory:

1. Log in to iProcess Engine as Administrator.

2. Stop iProcess Engine (if it is running).

3. Create a folder where you want to save the log files, and give the same permissions to access it as the original log files folder.

4. Open the staffpms file located in the *SWDIR*\etc directory. Then copy the absolute directory of the folder you created in step 3 to line 19 of the file.

5. Restart iProcess Engine.

If line 19 is blank, then iProcess Engine will use the *SWDIR*\logs directory as the default log files directory.

SWDIR\etc\sqloptim

The `sqloptim` file, which is located in the *SWDIR*/etc directory, specifies database hints or optimizers in SQL statements for database operations.

This configuration file only supports the following sqlIDs:

20, 21, 101, 139, 380, 456, 457, 461, 490, 502, 632.

SWDIR\etc\language.lng\audit.mes

This file contains the system-defined audit trail messages. These are added to the audit trail by the system each time an action of some sort is performed on the step in the case. These messages are pre-defined in the `audit.mes` file, which is located in the *SWDIR*\etc\language.lng directory. Each message has a three-digit number that is the message ID of the audit trail message. The system reserves Message IDs 000-255 for system use.

See Understanding Audit Trails on page 449 for an explanation of the system-defined messages and what they mean.

SWDIR\etc\language.lng\auditusr.mes

This file contains the user-defined audit trail messages. You must predefine these messages in the `auditusr.mes` file, which is located in the *SWDIR*`\etc\language.lng` directory. Once, you have predefined the audit trail messages, they can be added to the audit trail of a live case. You can use the `AUDIT` command under the *SWDIR*`\bin\swutil` directory to add a message to an audit trail of a live case. For information about adding user-defined audit entries, see "Audit Trails" in *TIBCO iProcess swutil and swbatch Reference Guide*.

SWDIR\etc\language.lng\stafferr.mes

This file contains the messages used by the $SYSTEM procedure.

The $SYSTEM procedure sends a work item to the iProcess Engine Administrator's work queue when the sw_warn or sw_error files, which are located in the *SWDIR*\logs directory, have been generated, warning the system administrator that an error has occurred. See iProcess Engine Log Files on page 439 for more information.

SWDIR\etc\language.lng\staffw.mes

This file contains some configurable messages that affect how the long date is displayed in TIBCO iProcess Workspace.

Changing the Long Date Format

For information on using the long date and time format in an iProcess step definition, see "Using Embedded and Ampersanded Fields" in *TIBCO iProcess Modeler Basic Design*.

The `staffw.mes` file, which is located in the *SWDIR*`\etc\language.lng` directory, determines how the long date is displayed by iProcess Workspace. The long date information is returned from the `staffw.mes` file, which is located in the *SWDIR*`\etc\language.lng` directory, instead of the `staffpms` file, which is located in the *SWDIR*`\etc\language.lng` directory, because it enables different users on the same system to have different long date displays depending on their `LANGUAGE` attribute.

See "Setting Pre-defined Attributes" in *TIBCO iProcess Workspace (Windows): Manager's Guide* for more information about how to set a user's `LANGUAGE` attribute.

To ensure the date is displayed consistently in both TIBCO iProcess Workspace and iProcess Engine, the information in the `staffw.mes` and `staff.mes` files, which are located in the *SWDIR*`\etc\language.lng` directory, must be the same. This means that any changes must be made in both files.

The following example is an extract from the `staffw.mes` file, which is located in the *SWDIR*`\etc\language.lng` directory:

```
0004:W:\\%s %s, %s\\dmy\\\ AM\ PM\Week
0013:W:Sunday\Monday\Tuesday\Wednesday\Thursday\Friday\Saturday$
0014:W:January\February\March\April\May\June\July\August\September
\October\November\December
```

The file is divided into one message per line. The messages that determine how the long date is displayed are:

- `0004` specifies each component of the long date.
- `0013` specifies the days of the week.
- `0014` specifies the months of the year.

Each message is in the format:

number:type:data

where:

- *number* is the identifier for this message. For example, `0004`.

- *type* is either `W`, indicating that the message is used by TIBCO iProcess Workspace, or blank, indicating that the message is used by iProcess Engine.

- *data* is one or more data entries associated with this message. If there are multiple data entries, each entry is separated by a backslash (\) character.

For example, the following table describes the data entries for message `0004`.

Position	Data	Description
1		Not used.
2		Not used.
3	`%s %s, %s`	The number of components used to specify each part of the long date format. Each component represents the date, month and year. For example, `10 March, 2004`.
4		Not used.
5	`dmy`	The order of the date format.
6		Not used.
7		Not used.
8	`AM`	Used for 12 hr time format. For example, `09:10 AM`.
9	`PM`	Used for 12 hr time format. For example, `03:12 PM`.
10	`Week`	Not used.

To change the long date format, for example, to `Wednesday 12 Dec, 2012`:

1. Edit message `0004` of the `staff.mes` file located in the
   SWDIR\etc\language.lng directory as follows:

   a. Add `%s` to position 3 to represent the day of the week, as shown below.

   b. Add *w* to position 5 to represent the day of the week, as shown below.

   c. Edit message `0014` to use short month names rather than long ones. For
      example, `Dec` instead of `December`.

```
0004:W:\\%s %s %s, %s\\wdmy\\\ AM\ PM\Week
0013:W:Sunday\Monday\Tuesday\Wednesday\Thursday\Friday\Saturday$
0014:W:Jan\Feb\Mar\Apr\May\Jun\Jul\Aug\Sept\Oct\Nov\Dec
```

---

2. Replicate the changes made in the `staffw.mes` and the `staff.mes` file, which
   are located in the *SWDIR*\etc\language.lng directory.

SWDIR\etc\language.lng\staff.mes

This file contains some configurable messages and options that are used by some of the iProcess Engine programs, for example, the iProcess Background.

Changing the Long Date Format

To ensure the date is displayed consistently in both TIBCO iProcess Workspace and iProcess Engine, the information in the `staffw.mes` and the `staff.mes` file must be the same. This means that any changes must be made in both files.

The format of the `staff.mes` file is divided into messages in the same way as the `staffw.mes` file.

To see how to amend the format of the long date in the `staff.mes`, see .

SWDIR\etc\swerwarn.mes

This file contains the templates for the messages that are written to the sw_warn and sw_error files, which are locatedin the *SWDIR*\logs directory. See iProcess Engine Log Files on page 439 for more information.

Chapter 3 **Tuning the iProcess Engine Using SWDIR\etc\staffcfg Parameters**

This chapter describes all of the parameters that you can use in the iProcess Engine `staffcfg` configuration file to optimize iProcess's performance for your particular requirements. The parameters all relate to memory and process configuration information.

Topics

Editing the SWDIR\etc\staffcfg File

⚠️ The default file contains a number of parameters most of which define the limits within which iProcess Suite is initially setup to work. There are also a number of other parameters which, when tuned, can give significant improvements in both performance and response.

You should be extremely careful when editing the staffcfg file. Careless changes can have a serious impact on system operation or performance. If you are in any doubt about whether or not to edit a specific parameter, please contact TIBCO Support for assistance.

If you want to add, remove or update parameters in the staffcfg file, which is under the *SWDIR*\etc directory:

1. Log in to as a user who (on Windows) is a member of the iProcess Administrators group or, (on UNIX) as user root.

2. Edit the file as required.

3. If necessary, ask all users to log out of iProcess Suite, then stop and restart the server.

SWDIR\etc\staffcfg File Format

The staffcfg file is an ASCII file containing a number of lines, divided into functional sections:

• Each section is headed by the section name at the start of a line, followed by a number of configuration lines.

• Each configuration line starts with a TAB character followed by the configuration name (e.g. MAXCASES), followed by a comma (,) followed by the configuration value.

• Anything from a semicolon (;) to the end of the line is treated as a comment and ignored.

Using Multiple Copies of SWDIR\etc\staffcfg

You can use different copies of the staffcfg file to optimize performance. For example, you can create one version which is optimized for batch processing, to be used at night, and another version which is optimized for user interaction, to be used during the day. You can then change the iProcess Suite's configuration by using batch files to:

1. Stop the server using the *SWDIR*\bin\swstop command. See Stopping iProcess Engine on page 7 for more informaiton.

You do not need to shut down Process Sentinels.

2. Copy the appropriate version of the staffcfg file to the *SWDIR*\etc directory.

3. Restart the server. See Starting iProcess Engine on page 2 for more information.

SWDIR\etc\staffcfg Parameters

The remaining sections in this chapter describe all of the staffcfg parameters, which are located in the *SWDIR*\etc directory. Each section of the staffcfg parameter has a corresponding section in this chapter, as follows:

- WQS Section
- FORM Section
- STAFFPRO Section
- STAFF Section
- DBSIZES Section
- DBPOOL Section
- CDQP Section

WQS Section

This section is used to configure the behavior of the work queue services. The following parameters are available:

- WQS_DEFAULTPRIORITY
- WQS_URGENTPRIORITY
- WQS_ROUND_ROBIN
- WIS_MAXFILEDESC
- WQS_QUEUE_WEIGHTING
- WQS_SHARED_MEMORY_QUEUES
- WIS_AGE_USE_WORKING_DAYS

WQS_DEFAULTPRIORITY

Section	WQS
Initial Value	50
Units	N/A
Range	0 to 32767
Description	Sets the default priority level for a new work item, if not already set. For more information about this parameter, see "Using Work Item Priorities and Escalation" in *TIBCO iProcess Modeler Advanced Design*.
Tuning	Work items can have priorities so that they can be sorted/filtered, etc. by priority level. You need to decide how your system will use priority levels and then decide upon a sensible default.
Related Parameters	None.

WQS_URGENTPRIORITY

Section	WQS
Initial Value	10
Units	N/A
Range	0 to 32767
Description	Sets the default Urgent Priority level for a new work item, if not already set. For more information about this parameter, see "Using Work Item Priorities and Escalation" in *TIBCO iProcess Modeler Advanced Design*.
Tuning	N/A.

WQS_ROUND_ROBIN

Section	WQS
Default Value	0
Units	N/A
Range	0 (use on-demand) or 1 (use round-robin)
Description	The Work Queue Server is responsible for the assignment of work queues to WIS processes. There are two methods it can use, either round-robin or on-demand.
Tuning	This parameter configures which of the methods is used for the queue allocation. See Configuring the Assignment of Queues to WIS Processes on page 405 for more information about the use of each method.

WIS_MAXFILEDESC

Section	WQS
Initial Value	0
Units	N/A
Range	>0
Description	The work item server process uses the select system call when waiting for client requests. It passes this the NOFILE/MAXFILES kernel parameter to receive as many clients as possible. This can cause a problem if this number is greater than FD_SETSIZE. If this happens, WIS_MAXFILEDESC can be set to a number greater than 0 but less than FD_SETSIZE.

WQS_QUEUE_WEIGHTING

Section	WQS
Initial Value	5
Units	N/A
Range	>0
Description	When using the on-demand queue allocation method, queues are allocated to Work Item Server (WIS) processes based on the cost of the work queue. The WQS_QUEUE_WEIGHTING parameter determines the cost of the work queues. See Configuring the Assignment of Queues to WIS Processes on page 405 for more information about the on-demand queue allocation method.
Tuning	This setting enables you to have some control about how work queues are allocated to WIS processes. For example, the larger the value, the more that the number of work queues rather than the number of work items in the work queues determines whether a work queue is allocated to a WIS process. Therefore, if you have lots of work queues with an even amount of work items in each, you may want to increase the value of the WQS_QUEUE_WEIGHTING parameter. If you only have a few work queues that contain large amounts of work items, you may want to lower the value.

WQS_SHARED_MEMORY_QUEUES

Section	WQS
Initial Value	1000
Units	N/A
Range	>0
Description	Specifies the *minimum* amount of shared memory to be allocated when the WQS process starts up.
Tuning	Because shared memory cannot be resized, the WQS process must allocate a fixed amount of shared memory when it starts up, it allocates shared memory equal to twice whichever of the following values is greater:

- the WQS_SHARED_MEMORY_QUEUES value.

- the number of user and group queues defined on the system.

> You must ensure that your system has enough shared memory configured for the WQS process to allocate. If it does not, the WQS process will be unable to start.

Depending on the number of queues you have defined, this value will therefore be at least:

```
(WQS_SHARED_MEMORY_QUEUES * 2 )* 1.2K
```

For example, if WQS_SHARED_MEMORY_QUEUES is 1000, and the number of queues defined on the system is 1250, then the WQS process will allocate 3000K of shared memory (1250*2*1.2). Your system must have at least 3000K of shared memory available for the WQS process.

> Please see your operating system documentation for information about how to configure shared memory on your system.

WIS_AGE_USE_WORKING_DAYS

Section	WQS
Initial Value	0
Units	N/A
Range	0 or 1
Description	Defines whether or not iProcess will escalate a work item's priority when its increment period expires. If the value is:

- 0 specifies a work item's priority will always escalate when its increment period expires, whether the current date/time is a working day or a non-working day (as defined in the staffpms file - see SWDIR\etc\staffpms on page 25).

- 1 specifies a work item's priority will only escalate if the current date/time is defined as a working day (in the staffpms file).

> If the value of this parameter is set to 1, it only affects procedures that have the Use Working Days flag set in TIBCO iProcess Modeler.

For example, suppose that:

- on a Friday morning, a work item has a priority value of 10.

- its increment period is 1 day, and this period expires at 5pm each day.

- the working week is defined in the staffpms file as Monday to Friday.

On the following Monday morning, the work item's priority value will therefore be:

- 9 specifies if WIS_AGE_USE_WORKING_DAYS is set to 1 and the procedure's Use Working Days flag is set. (The priority value is incremented when the increment period expires on Friday, but is not incremented when it expires on Saturday and Sunday.)

- 7 specifies for any other combination of these settings. (The priority value is incremented when the increment period expires on Friday, Saturday and Sunday.)

FORM Section

This section enables you to configure TIBCO iProcess Engine form parameters. Changes made to this section take effect after you log out of iProcess and then back in again.

The following parameters are available:

- MAX_SCRIPT_CALL_DEPTH
- MAXVLD

MAX_SCRIPT_CALL_DEPTH

Section	FORM
Initial Value	10
Units	N/A
Range	>0
Description	Defines the maximum recursive depth for calling scripts from scripts. The default is 10 which means that you can call out recursively up to 10 scripts. Therefore, if you have 10 scripts (script1, script2 etc) you can use the CALL expression in script1 to call script2 and script2 can call script3 and so on up to script10.
	See "Creating Scripts" in *TIBCO iProcess Modeler Advanced Design* for more information about using scripts.
Tuning	N/A.

MAXVLD

Section	FORM
Initial Value	50
Units	N/A
Range	>0
Description	The maximum number of validations that are added to a validations list with the VLDFILE or VLDQUERY functions.
Tuning	N/A.

STAFFPRO Section

This section enables you to configure TIBCO iProcess Engine server processes and performance parameters. You need to stop and restart the server before any changes are applied.

The following parameters are available:

- LDAP_DIT
- LDAP_POOL_SIZE
- MODTIME_PERM
- PROCDEF_CACHESIZE
- RESEND_ORIGINAL_TIMESTAMP
- LAST_MODIFIED_TIME

LDAP_DIT

In previous iProcess Engine versions this parameter was called X500_DIT. If you upgrade from a pre-Version 10.2.0 iProcess Engine, the X500_DIT parameter is left in the staffcfg file, and can be manually deleted if required.

Section	STAFFPRO
Initial Value	0
Units	N/A
Range	0 or 1
Description	Defines whether or not to obtain iProcess user data from an LDAP Directory information Tree (DIT):

- 0 - iProcess user data is held internally.

- 1 - obtain iProcess user data from an LDAP DIT.

LDAP_POOL_SIZE

Section	STAFFPRO
Initial Value	10
Units	Connections
Range	>0
Description	This parameter specifies the LDAP connection pool size. It only applies if LDAP_DIT is set to 1.
Tuning	N/A.

MODTIME_PERM

Section	STAFFPRO
Initial Value	0
Units	N/A
Range	0 or 1
Description	When you use LDAPCONF with Active Directory, the modified timestamp is returned with either a 'Z' or '0Z' at the end of the string depending on the version of Active Directory. A value of:

- 0 means use a 'Z' terminator for search.

- 1 means use a '.0Z' terminator for search.

PROCDEF_CACHESIZE

Section	STAFFPRO
Initial Value	5
Units	NA
Range	2-1000
Description	The number of procedure definitions to cache on the server computer.
Tuning	This value does not need to be larger than the number of procedures on your system.

LID_CLIENT_TIMEOUT

Section	STAFFPRO
Initial Value	60
Units	Seconds
Range	>0
Description	The time that the iProcess Workspace is not allowed to update the `sww.uid` file before being assumed to have logged out. This is to allow users to log back in from iProcess Workspaces after an abnormal iProcess Workspace shutdown.
Tuning	N/A.
Related Parameters	UIDCRPERIOD

RESEND_ORIGINAL_TIMESTAMP

Section	STAFFPRO
Initial Value	0
Units	N/A
Range	0 or 1
Description	Sets the timestamp to be used for the Arrival Time of a work item when a resend is performed on a client queue:

- 0 means that the current timestamp (of the RESEND) is used.

- 1 means that the original timestamp (when the item was added to the queue) is used.

If this parameter is not present, the system defaults to the current timestamp (0).

Tuning	N/A.

LAST_MODIFIED_TIME

Section	STAFFPRO
Initial Value	0
Units	N/A
Range	0 or 1

Description   By default, when LDAPCONF performs a partial synchronization, it checks the LDAP ModifyTimeStamp attribute to determine whether an entry has been modified since the last update (and so needs to be downloaded to iProcess). However, some LDAP Admin applications modify this attribute when handling user logons and authentication, which means that LDAPCONF cannot use it in this way. You can therefore use the LDAP lastModifiedTime attribute instead, with LDAP servers that require it.

The LAST_MODIFIED_TIME parameter defines which LDAP attribute LDAPCONF should check when performing a partial synchronization:

- 0 means that LDAPCONF checks the LDAP ModifyTimeStamp attribute to determine whether an entry has been modified since the last update.

- 1 means that LDAPCONF checks the LDAP lastModifiedTime attribute to determine whether an entry has been modified since the last update.

This parameter is not present by default. You must add it if required. If this parameter is not present, the system defaults to using the LDAP ModifyTimeStamp attribute (0).

Tuning   N/A.

STAFF Section

This section enables you to configure the behavior of TIBCO iProcess Workspace. You have to stop and restart the server before any changes will take effect.

The following parameters are available:

- UIDCRPERIOD
- RPCSVR_TIMEOUT
- PWD_PERIOD
- START_TX_RX
- RPCXFRSIZE
- MAX_USERS_PER_PROCESS
- PRE_LOAD_POOL_SERVERS
- USER_LOAD_ALLOCATION
- WQ_SORT_ITEM
- DYNDEADPRED
- IAPSCHEMA
- IAPSCHEMA2_1

UIDCRPERIOD

Section	STAFF
Initial Value	30
Units	Seconds
Range	>0
Description	Defines the amount of time between a windows foreground login refresh.
Related Parameters	LID_CLIENT_TIMEOUT.

RPCSVR_TIMEOUT

Section	STAFF
Initial Value	600
Units	Seconds
Range	>0
Description	This parameter defines the period of time an RPC server connection exists without being used.
	TIBCO iProcess Workspace will poll the RPC server (swrpcsvr) on a regular basis to keep it's connection alive. If the connection is lost for any reason, such as abnormal termination of the client, then the RPC server will wait for RPCSVR_TIMEOUT seconds without receiving a request before shutting down.
Tuning	There is generally no need to change this parameter as there should not be any need for it to come into effect.
	The downside of having it set to a long period (such as an hour) is that if a single machine is switched off with iProcess running, then the RPC server will not shutdown until after that period.

PWD_PERIOD

Section	STAFF
Initial Value	15
Units	Minutes
Range	>0
Description	Defines the time interval between passwords being cached on clients.
Tuning	Reducing this value means that iProcess can detect changes in users' passwords made outside of iProcess more quickly. However, it can mean that iProcess checks for user password changes more frequently causing a degradation in performance.

START_TX_RX

Section	STAFF
Initial Value	0
Units	N/A
Range	1 or 0
Description	Defines whether or not to start (1) server-to-server processes.
Tuning	None.

RPCXFRSIZE

Section	STAFF
Initial Value	4096
Units	Bytes
Range	512, 1024, 2048, 4096
Description	This setting determines the maximum buffer size used for client/server communication of stream data. This setting is primarily used when reading text files, forms or memos from the server or for copying files down to the client.
	As a significant amount of data needs to be read at login time increasing the size of this parameter can have benefits to login time on large systems, particularly over WANs.
	Adjusting this value enables you to tune the size and number of packets sent over the network.
Tuning	When considering network performance, particularly over a WAN, it is important to consider the number and size of requests being made over the network.
	Any tuning of this parameter needs to take into account the characteristics of the network, in general increasing the size of this parameter to 4096 will reduce the number of network requests and therefore reduce the latency inherent in waiting for a request to be responded to. There may be circumstances on a busy WAN where sending large packets is blocking other requests and therefore causing poor response for other users.
	In most cases, network performance problems in iProcess are not caused by the amount of data being transferred but the number of packets being sent. Therefore by increasing the value of RPCXFRSIZE many RPC calls can pass more data than before and therefore less calls are made. Even on a LAN a single RPC round trip can take 25ms irrespective of the size of the packet, i.e. 20 bytes or 4K, therefore 200 RPC calls are likely to take 5 seconds. If by increasing the packet size only 50 RPC calls are made then the total time come down to 1.25 secs.

Example

A procedure does a FileCopy from server to client of a 2Mb file.

Results

- With RPCXFRSIZE=1024 time to copy 2Mb to Client = 15 Sec.

- With RPCXFRSIZE=4095 time to copy 2Mb to Client = 8.5 Sec.

While this is a large file and not necessarily a typical operation, you can see there are some benefits.

MAX_USERS_PER_PROCESS

Section	STAFF
Initial Value	20
Units	Users
Range	>1
Description	Defines the number of users allocated to each RPC pool server. The iProcess Suite allocates users to the RPC pool servers, which have been started (or pre-loaded if you use PRE_LOAD_POOL_SERVERS), on a round robin basis by default. A new RPC server is started when there are no more allocated slots in the RPC servers currently running.
	For example, where there are 8 RPC pool servers pre-loaded and 8 users logged on, you could have each person connected to a different RPC pool server.
Tuning	N/A.

PRE_LOAD_POOL_SERVERS

Section	STAFF
Initial Value	0
Units	RPC pool servers
Range	-1, 0 or any positive integer
Description	Defines the number of RPC pool servers that you want to pre-load during the iProcess startup process.
Tuning	Setting this to a positive value results in that number of pool servers being started.
	If you set the value to -1, the RPC server calculates the number of RPC pool servers to start up. The RPC server calculates this number using the MAX_USERS_PER_PROCESS value and the number of users held in the iProcess Engine. For example, if there are 800 users and MAX_USERS_PER_PROCESS is set to 40, then 20 RPC pool servers will be started.
	If the value is set to 0, pool servers are started up on demand as users log in. This can slow the login process because users have to wait for the processes to be started.
	Each client login will be assigned to one of the RPC pool servers.
Related Parameters	MAX_USERS_PER_PROCESS.

USER_LOAD_ALLOCATION

Section	STAFF
Initial Value	0
Units	N/A
Range	0 or 1
Description	Defines the process by which client connections are allocated to RPC pool servers.
Tuning	When set to the default value of 0, client login requests are allocated using a round robin method where each client login is allocated to the next RPC pool server.
	When set to 1, client requests are allocated to RPC pool servers by finding the pool server that has the least number of client connections. If all pool servers are full, a new process is created for the client request.

WQ_SORT_ITEM

Section	STAFF
Initial Value	0
Units	N/A
Range	0 or 1
Description	Defines whether the folders in the work queues list of the Work Queue Manager are sorted by Queue Name or Queue Description.
Tuning	When set to the default value of 0, or when not present in the `staffcfg` file, the list of work queues are sorted by Queue Name.

When set to 1, the list of work queues is sorted by Queue Description. Note that upper-case letters appear first after sorting, so the following descriptions:

- Manager1
- allenb
- Administrator
- richardH
- paulap

would appear sorted as follows:

- Administrator
- Manager1
- allenb
- paulap
- richardH

DYNDEADPRED

Section	STAFF
Initial Value	1
Units	N/A
Range	0 or 1
Description	Defines how the predicted step duration is to be calculated.

Tuning
- If the value is 1, then:
  — If the 'Use Deadline for Step Duration' flag is set, then use the deadline as the predicted duration.
  — If the 'Use Deadline for Step Duration' flag is not set, and the deadline is processed, then use the deadline as the predicted duration.
  — If the deadline is processed, and the duration is set, then use the deadline as the predicted duration.
- If the value is 0, then:
  — If the 'Use Deadline for Step Duration' flag is set, then use the deadline as the predicted duration.
  — If the 'Use Deadline for Step Duration' flag is not set, the deadline is not processed, and the duration is set, then use the configured duration for the predicted duration.
  — If the deadline is processed, and the duration is set, then use the configured duration for the predicted duration.

IAPSCHEMA

Section	STAFF
Initial Value	0
Units	N/A
Range	0 or 1
Description	This parameter controls whether messages generated by the IAPJMS process should be produced in the basic format or in the extended format that includes information on the audit user and addressee of the step, and the main procedure of a sub-case:

- 0 means that messages should be in basic format
- 1 means that messages should be in extended format

See Activity Monitoring and Work Queue Delta Configuration on page 274 and "Monitoring Activities" in *TIBCO iProcess Engine Architecture Guide* for more information about IAPJMS.

Tuning	N/A.

IAPSCHEMA2_1

Section	STAFF
Initial Value	0
Units	N/A
Range	0 or 1
Description	This parameter controls whether messages generated by the IAPJMS process should be produced in basic format or in extended format that includes Mail ID and extended step description. A value of:

- 0 means messages should be generated in basic format.

- 1 means messages should be generated with Mail ID and extended step description. The MailID and ExtDescription elements are created in the messages.

    — The MailID element can be used to build a tag for each work item.

The extended step description, along with the step description, is published with a minimum length of 48 bytes for the messages (with the activity ID 001).

Tuning	N/A.

DBSIZES Section

This section enables you to specify the size of certain items in the database. Changes are applied only after stopping and restarting the server.

The following parameter is the only one available: MEMOATTMAX

MEMOATTMAX

Section	DBSIZES
Initial Value	64000
Units	Bytes
Range	NA
Description	Maximum size of Memos and Attachments.
Tuning	N/A.

DBPOOL Section

This section enables you to configure database connection pool parameters. You have to stop and restart an iProcess process before any changes take effect in that process.

The following parameters are available:

- POOLSIZE
- POOLGROWSIZE
- MAXPOOLSIZE
- POOLCONNTIMEOUT

POOLSIZE

Section	DBPOOL
Initial Value	1
Units	Database connections
Range	>0
Description	Defines the initial size of the database connection pool.
Tuning	N/A.

POOLGROWSIZE

Section	DBPOOL
Initial Value	2
Units	Database connections
Range	>0
Description	The size by which to grow the database connection pool.
Tuning	N/A.

MAXPOOLSIZE

Section	DBPOOL
Initial Value	10
Units	Database connections
Range	>0 ; > POOLSIZE
Description	Defines the maximum size of the database connection pool. This value is used to calculate the maximum concurrent user connections needed on the database server by any iProcess process.
Tuning	N/A.

POOLCONNTIMEOUT

Section	DBPOOL
Initial Value	600
Units	Seconds
Range	>0
Description	Defines the timeout value for database connections. This value is checked whenever a new database connection is requested, and any existing connections that have been inactive for longer than this value are terminated. This ensures that the database connection pool is not increased unless all existing connections are actually in use.
Tuning	N/A.

CDQP Section

This section allows you to configure the use of Case Data Queue Parameters (CDQPs) on the server.

Changes to parameters in this section take effect when CDQP configuration is next imported, using `swutil QINFO`. For more information about CDQPs, see "Case Data Queue Parameters" in *TIBCO iProcess swutil and swbatch Reference Guide*.

The following parameters are available:

- CDQPMAXGLOBAL
- CDQPMAXQUEUE

CDQPMAXGLOBAL

Section	CDQP
Initial Value	60
Units	NA
Range	0 - 32767
Description	Defines the maximum number of CDQPs that can be defined on this server.
	To disable the use of CDQP parameters, either set this parameter to 0, or delete it.
Tuning	N/A.
Related Parameters	CDQPMAXQUEUE

CDQPMAXQUEUE

Section	CDQP
Initial Value	40
Units	NA
Range	0 - 32767
Description	Define the maximum number of CDQPs that can be mapped to a particular queue (including the default user and default group queues). If this value is higher than the CDQPMAXGLOBAL value, the CDQPMAXGLOBAL value will be used instead.
	To disable the use of CDQP parameters, either set this parameter to 0, or delete it.
Tuning	N/A.
Related Parameters	CDQPMAXGLOBAL

NETWORK Section

This section allows you to configure the network information.

The following parameter is the only one available: IPPROTOCOL

IPPROTOCOL

Section	NETWORK
Initial Value	4
Units	NA
Range	4 or 6
Description	Configure the Internet Protocol, Internet Protocol Version 4 (IPv4) or Internet Protocol Version 6 (IPv6). The values 4 and 6 indicate IPv4 and IPv6 respectively.
Tuning	N/A.

Obsolete Parameters

Obsolete
The following `staffcfg` parameters are not used in this version of iProcess Engine.

When you upgrade, some of these parameters may be removed from the `staffcfg` file, others may remain. Those that do remain are, however, ignored by iProcess Engine.

Table 6   Obsolete staffcfg Parameters

Parameter	Section	Notes
FGLITO	STAFF	No longer needed because the login daemon process that uses it no longer exists.
RNGMODE	STAFF	These parameters are no longer needed because port range configuration is now stored in the database, and can be configured by using the `swadm` utility. See Administering Firewall Port Ranges on page 363 for more information.
RNGBLOCKED	STAFF	
RNGTHRESHOLD	STAFF	
PORTSTART	STAFF	
RPCSTART	STAFF	
ALLOCRPCTIMEOUT	STAFF	
IS_ACTIVEDIRECTORY	STAFFPRO	Obsolete because the `LDAPCONF` setup process now prompts for this data.
QUEUEPROCTIME	STAFFPRO	
RUNPROCTIME	STAFFPRO	
SYSPROCS	STAFFPRO	
URDSLEEP	STAFFPRO	
USERPROCS	STAFFPRO	
CMSDELAY	STAFFCMS	
CREATIME	STAFFCMS	

Table 6   Obsolete staffcfg Parameters (Cont'd)

Parameter	Section	Notes
CRXSIZE	STAFFCMS	
RPCTIME	STAFFCMS	
RXSLEEP	STAFFCMS	
TXSLEEP	STAFFCMS	
WIS_NEW_QUEUE_POLL_PERIOD	WQS	
WIS_CLIENT_IDLE_PERIOD	WQS	These parameters are no longer needed because the `WIS` process is now multi-threaded, and so can concurrently perform updates on queues and process RPC requests. See Overview on page 402 for more information.
WIS_MBOX_WORK_LIMIT	WQS	
WIS_RPC_SERVICE_PERIOD	WQS	
WIS_TOUT_GRANULARITY	WQS	
WQS_UPDATE_PERIOD	WQS	
WIS_WRITELOCKS	WQS	

Chapter 4   **Administering Servers**

This chapter explains how to use the *SWDIR*\util\swadm server configuration utility to administer the server(s) hosting your iProcess Engine.

To use this utility, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as a background or root user.

If you are using a node cluster architecture, you can run this utility from any server within the cluster (as long as that server has a connection to the TIBCO iProcess Engine database instance).

These commands read and update data in the node_cluster database table.

Topics

Show all Server Details

To display a list of the servers in your iProcess Engine, enter the following command:

```
swadm show_servers
```

Examples

1. This example shows the output from this command for an iProcess Engine that is installed as a single node, on server despina.

```
# swadm show_servers
```

Machine ID	Machine Name	Master	Check Error Files	Machine Comment
1	DESPINA	Y	Y	despina

2. This example shows the output from this command for an iProcess Engine that is installed as a node cluster, on servers despina and hades. The master Process Sentinels are running on despina and both servers are set to check for iProcess error files.

```
# swadm show_servers
```

Machine ID	Machine Name	Master	Check Error Files	Machine Comment
1	DESPINA	Y	Y	despina
2	HADES	N	Y	hades (slave)

Update Server Details

To update the settings of a server in your iProcess Engine, such as the description of the server and whether it will check for the sw_error or sw_warn files, which are located in the *SWDIR*\log directory, you can use the following command:

swadm update_server *machine_id* | *machine_name* *check_error_files* *machine_comment*

where:

- *machine_id* is the server identifier (such as 1, 2 or 3).

- *machine_name* is the physical name of the server (such as pluto or hercules).

- *check_error_files* is used to define if the server checks for iProcess error files (the sw_error and sw_warn files). Replace *check_error_files* with one of the following values:

  — Y specifies Process Sentinels check for error files.

  — N specifies no error checking is performed.

- *machine_comment* is used to provide any notes for the server. This can be used to describe the function of the server such as background_1 if it runs the background processes.

Example   If you want server hades in your iProcess Engine to start checking for error log files and have the description of BG_processor_2, you can change the setting of the server using the following command.

swadm update_server hades Y BG_processor_2

Add a Server

You can add servers to your iProcess Engine at any time. For example, you can increase the amount of case processing by adding a server and starting more background processes.

To add a server to your iProcess Engine, use the following command:

```
swadm add_server machine_name master check_error_files machine_comment
```

where:

- *machine_name* is the physical name of the server you want to add.

- *master* is the parameter that specifies if you want the server to host the master Process Sentinels. Replace *master* with either:

    — Y specifies master server

    — N specifies slave server.

    — See "Process Management" in *TIBCO iProcess Engine Architecture Guide* for more information about the Process Sentinels architecture.

- *check_error_files* specifies whether Process Sentinels on this server check for the creation of the sw_error and sw_warn files.

    — Y specifies Process Sentinels check for errors.

    — N specifies no checking is performed.

- *machine_comment* is the text description added to identify the server.

Example   The following example adds server pluto to iProcess Engine. It:

- Specifies that pluto will run as a slave server and will check for iProcess error files.

- Sets its comment as BG_processor_3, indicating that it is the third server (in a node cluster) that runs background processes.

    ```
    # swadm add_server pluto N Y BG_processor_3
    ```

Remove a Server

If you need to remove a server from your iProcess Engine, for example, to take a server offline and upgrade it, you can use the following command:

```
swadm delete_server machine_id | machine_name
```

where:

- *machine_id* is the server identifier (such as 1, 2 or 3) for the server you want to remove from iProcess Engine.

- *machine_name* is the physical name of the server (such as pluto).

Example   If you have four servers in your iProcess Engine (in a node cluster), and you need to take the server called pluto offline to perform some kernel changes and upgrades, you can remove the server from the node cluster using the following command:

```
# swadm delete_server pluto
```

Alternatively, before removing the server from the cluster you can move the processes that currently run on the server to another server in the cluster using the move_server command on . You can only move background processes individually.

If you remove a server that is running only background processes, users may notice a reduction in the performance of case processing. However, if you remove a server that is running foreground processes such as a WIS, all of the clients need to log out of and then log back in.

Find a Server's Details

To find out the configuration of a specific server in your iProcess Engine, use the following command:

    swadm find_server *machine_id* | *machine_name*

where:

- *machine_id* is the server identifier (such as 1, 2 or 3) for the server you want to see the properties of.

- *machine_name* is the physical name of the server.

Example The following example displays the configuration details for a server despina. The master Process Sentinels are running on despina and the server is set to check for iProcess error files.

```
# swadm find_server
Machine ID     Machine Name     Master     Check Error Files     Machine Comment
1              DESPINA          Y          Y                     despina
```

Find the Master Server

To find out which server is currently hosting the master Process Sentinels, enter the following command:

```
swadm find_master
```

Example The following example shows that the server called despina is currently configured to run the master Process Sentinels.

```
# swadm find_master
Machine ID     Machine Name     Master     Check Error Files     Machine Comment
1              DESPINA          Y          Y                     despina
```

Define a Server as the Master Server

Process Sentinels operate on each server involved in hosting your iProcess Engine, but one server has to be configured to host the master Process Sentinels. See "Process Management" in *TIBCO iProcess Engine Architecture Guide* for more information about Process Sentinels.

If the master process fails or needs to be shutdown, such as when upgrading the server, you can assign a new server to host the master Process Sentinels using the following command:

```
swadm set_master machine_id | machine_name
```

where:

- *machine_id* is the server identifier (such as 1, 2 or 3) on which you want the master Process Sentinels to run.

- *machine_name* is the physical name of the server on which you want the master Process Sentinels to run.

Example To set the master Process Sentinels to run on the server called hades, you would enter the following command.

```
# swadm set_master hades
The master machine has been set to machine hades.
```

Move Processes From One Server to Another

You can move all processes assigned to operate on one server to another server. You might want to do this if one server has failed or you need to take it offline to upgrade it.

Processes need to be stopped before you can move them. See for more information.

Use the following command to move processes to another server:

```
swadm move_server machine_id machine_name
```

where:

- *machine_id* is the server identifier (such as 1, 2 or 3) of the source server.

- *machine_name* is the physical name of the destination server (such as pluto).

Example   If you have two servers in your iProcess Engine (despina that has a unique ID of 1, and hades), you can move all the iProcess Engine server processes running on despina to hades using the following command.

```
# swadm move_server 1 hades
```

Chapter 5 **Administering iProcess Engine Server Processes**

This chapter explains how to use the `swadm` server configuration utility to administer iProcess Engine server processes.

Topics

Introduction

There are three utilities that you can use to administer iProcess Engine server processes:

- The `swadm` utility, which you can use to directly administer server processes. See Using SWDIR\util\swadm to Administer Server Processes on page 110 for more information.

- The `swsvrmgr` utility, which you can use to administer server processes using Process Sentinels. See Using SWDIR\util\swsvrmgr to Administer Server Processes on page 118 for more information.

The `swadm` utility directly updates the `process_config` database table, so any changes you make will still apply if iProcess Engine is restarted. By contrast, any changes you make using the `swsvrmgr` utility will be lost if Process Sentinels fail or are restarted.

- the iProcess Server Manager, which provides a graphical view of server processes. You can use it to administer single processes, processes on individual machines, or processes in a node cluster. See Using the iProcess Server Manager to Administer Server Processes on page 130 for more information.

Server Processes

The following table shows the server processes that are initially set up when iProcess Engine is installed. The details of each process are stored in the process_config table. Note that:

- **Process Sentinels** are responsible for controlling all the TIBCO iProcess Engine processes. If a node cluster architecture is used, then Process Sentinels will exist on each server to manage the processes running on that server.

- **Foreground processes** are responsible for communicating with TIBCO iProcess Workspaces and for passing any TIBCO iProcess Workspace requests such as released work items to the background area for processing.

All foreground processes must run on the master server.

- **Background processes** are responsible for processing message instructions received from the clients such as releasing a step or forwarding a step. They also monitor and process any deadlines that have been set up in the procedure and manage case prediction.

Table 7   Server Processes

Process Name	Process Description	Number of Processes	Name Shown in Task Manager[1]
Process Sentinels			
PROCMGR	Process Sentinel (worker)	1	procmgr.exe
PROCMGR	Process Sentinel (watcher)	1	procmgr.exe
Foreground Processes			
RPC_POOL[2]	RPC pool server	1-*n*	SWRPCSVR.EXE
RPC_TCP_LI	RPC TCP listener	1	SWRPCSVR.EXE
RPC_UDP_LI	RPC UDP listener	1	swrpcudp.exe
WIS	Work Item Server	2	WISRPC.EXE
WISMBD	Work Item Server Mbox daemon	2	wismbd.exe
WQS	Work Queue Server	1	WQSRPC.EXE

Table 7   Server Processes (Cont'd)

Process Name	Process Description	Number of Processes	Name Shown in Task Manager[1]
Background Processes			
BG	Background Mbox daemon and Case Instruction processor	4	swbgmd.exe
BGPREDICT[3]	Background case prediction server	1	swbgmd.exe
DBQD[3]	Database Queue Daemon	1	n/a
DIRECTOR[3]	TIBCO iProcess Objects Director	1	SPODirector.exe
DLMGR	Deadline Manager	1	dlmgr.exe
IAPJMS[4]	IAPJMS process	1	iapjms.exe
RPCBG	RPC Background process	1	staffrpcbg.exe
SPO	TIBCO iProcess Objects Server	1	SWEntObjSv.exe

1. The Windows Task Manager. Not applicable on UNIX.

2. This process does not get listed by `swadm show_processes` or `swsvrmgr status -v`.

3. Only present on the DB2 version of iProcess Engine.

4. This process is disabled unless you have chosen to enable it when installing iProcess Engine.

Sequence Number Caching

The iProcess Engine server processes use sequence numbers extensively in doing their work. A sequence number is simply a unique identifier for an object, such as a case number, wait ID or request ID.

Sequence numbers are generated on an "as required" basis by calling a stored database procedure that accesses the `sequence` table. (This table contains an identity column. The procedure inserts a row into the table, returns the value of the identity column, then deletes the row.)

However, getting sequence numbers directly from the database in this way can create a performance bottleneck, because while one process is requesting a number it must block any other process from attempting to do so.

To minimize the effect of this bottleneck, you can assign caches of sequence numbers to a process, using process attributes. The process will get a sequence number from its cache when it needs one, and will only need to access the database to refresh the cache when it has run out of numbers.

The following table shows:

• The different sequence numbers that can be cached, and the process attributes that you need to set to cache them. (See Administering Process Attributes on page 141 for more information about process attributes and how to set them.)

• The different processes that use each sequence number.

Sequence number (Process Attribute)	Process Name			
	BG	RPC_POOL	SWBATCH	WIS
Case number (CNUM_SEQ_CACHE)	No, unless the system makes heavy use of sub-procedures.	Yes	Yes	Yes - used when starting new cases from TIBCO iProcess Workspace.
REQ ID (REQID_SEQ_CACHE)	Yes - A REQ ID is needed for each work item that is sent out.	Yes	Yes	Yes - used when starting new cases from TIBCO iProcess Workspace.
Wait ID (WIS_INDEX_REFRESH)	Yes, if waits are used in procedures.	No	No	No

Note that if you use sequence number caching:

• Gaps may appear in the value of the sequence numbers. For example, if the BG process caches 50 REQ IDs when it starts up, processes one NEWCASE instruction and then shuts down, the unused REQ IDs (2 to 50) will be lost.

• It is possible for a lower case number to be started after a higher case number. For example, suppose that a WIS has 50 case numbers (1 to 50) cached, and a user uses SWUTIL CSTART, which are located in the *SWDIR*\bin directory to start a case. The case will have case number 51 - the next available number in the sequence. However, if a user then starts a case through the WIS, that case will have case number 1 - the next available number in the cached sequence. Thus, the start date/time for case number 1 will be later than the start date/time for case number 51.

Using SWDIR\util\swadm to Administer Server Processes

You can use the swadm utility to view, run, delete and disable server processes. Note that:

• To use this utility, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as a iProcess Engine background user or root user.

• If you are using a node cluster architecture, you can run this utility from any server within the cluster (as long as that server has a connection to the TIBCO iProcess Engine database instance).

The following table summarizes the commands you can use to administer process attributes.

The following table summarizes the swadm commands you can use to administer server processes.

Command	Task
swadm show_processes	Show Server Processes
swadm add_process	Run a New Process
swadm disable_process	Disable a Process
swadm enable_process	Enable a Process
swadm delete_process	Delete a Process
swadm evloopback	Test Events

These commands read and update the process_config database table.

Show Server Processes

To display a list of the iProcess Engine server processes currently defined on your iProcess Engine, use the following command:

```
swadm show_processes –mmachine_id [–pprocess_name [–iprocess_instance]]
```

where:

- *machine_id* is the unique identifier for the server, assigned when the server is added to iProcess Engine. You can find a server's identifier using the swadm show_servers command.

- *process_name* is the process name of the server process.

- *process_instance* is the specific instance of the process.

The command lists the following information for each process:

- Machine ID is the unique identifier for the server, assigned when the server is added to iProcess Engine.

- Process Name is the process name of the server process.

- Process Inst is the specific instance of the process.

- Enabled is Y if the process is currently enabled, N if it is not.

- Persistent is Y if the process will be automatically restarted if iProcess Engine restarts, and N if it will not.

> Whether or not a process restarts automatically is defined by the PM_AUTO_BOOT process attribute.

- Last Status is the last known status of the process - either starting, running, paused, shutting down or stopped.

- Status Comment is a descriptive comment associated with the Last Status.

Example   The following command shows the processes currently defined on a server.

```
# swadm show_processes -m1
```

Machine ID	Process Name	Process Inst	Enabled	Persistent	Last Status	Status Comment
1	BG	1	Y	Y	Running	BG process started
1	BG	2	Y	Y	Running	BG process started
1	BG	3	Y	Y	Running	BG process started
1	BG	4	Y	Y	Running	BG process started
1	BGPREDICT	1	Y	Y	Running	BG process started
1	DIRECTOR	1	Y	Y	Running	DIRECTOR process started
1	DLMGR	1	Y	Y	Running	DLMGR process started
1	IAPJMS	1	Y	Y	Running	IAPJMS process started
1	RPCBG	1	Y	Y	Running	RPCBG process started
1	RPC_TCP_LI	1	Y	Y	Running	RPC listener process started
1	RPC_UDP_LI	1	Y	Y	Running	RPC listener process started
1	SPO	1	Y	Y	Running	SPO Server process started
1	WIS	1	Y	Y	Running	WIS process started
1	WIS	2	Y	Y	Running	WIS process started
1	WIS	3	Y	Y	Running	WIS process started
1	WIS	4	Y	Y	Running	WIS process started
1	WISMBD	1	Y	Y	Running	WISMBD process started
1	WISMBD	2	Y	Y	Running	WISMB process started
1	WQS	1	Y	Y	Running	WQS process started

Run a New Process

To start a new process running on a server, use the following command:

    swadm add_process *machine_id* | *machine_name* *process_name* *enabled*

where:

- *machine_id* is the unique identifier for the server.

- *machine_name* is the descriptive name of the server.

- *process_name* is the process name of the server process you want to run.

  If *process_name* is BG, BGPREDICT, DBQD, DLMGR, SPO, or DIRECTOR the process starts as soon as Process Sentinels re-cache the changes to the process_config table. If *process_name* is any other process (i.e. a foreground process), the process does not start until iProcess Engine is restarted.

- *enabled* is used to specify if you want the process to run immediately (Y) or whether it will be added to the process_config table but will be currently disabled (N).

Notes   All foreground processes (see Server Processes on page 107) must run on the master server.

Example   To start a new instance of the Background Mbox Daemon process on server2 so that it runs immediately, enter the following command:

    # swadm add_process server2 bg Y

Disable a Process

You can temporarily disable a server process so that Process Sentinels will not start it. To prevent the process running without removing the entry and configuration settings for it from the database tables, use the following command:

```
swadm disable_process machine_id process_name process_instance
```

where:

- *machine_id* is the unique identifier for the server on which the process is configured to run.

- *process_name* is the process name of the server process you want to disable.

- *process_instance* is the number of the process instance which you want to disable.

Example   To disable the second instance of the WIS process on the server with an ID of 3, you would enter the following command:

```
# swadm disable_process 3 WIS 2
```

Enable a Process

You can re-enable a process so that Process Sentinels can start it again using the following command.

```
swadm enable_process machine_id process_name process_instance
```

where:

- *machine_id* is the unique identifier for the server on which you want to enable the process.

- *process_name* is the process name of the server process you want to enable.

- *process_instance* is the number of the process instance which you want to enable.

You need to use the `wsvrmgr` utility to start the process. See Using SWDIR\util\swsvrmgr to Administer Server Processes on page 118.

Notes   All foreground processes (see Server Processes on page 107) must run on the master server.

Example   To enable the second instance of the `Background Mbox Daemon` process on the server with an ID of 3, you would enter the following command:

```
# swadm enable_process 3 WISMBD 2
```

Delete a Process

To remove a process from a server, use the following command:

    swadm delete_process *machine_id* *process_name* *process_instance*

where:

- *machine_id* is the unique identifier for the server that you want to remove the process from.

- *process_name* is the process name of the server process you want to delete.

  If *process_name* is BG, BGPREDICT, DBQD, DLMGR, IAPJMS, SPO or DIRECTOR, the process is removed as soon as Process Sentinels re-cache the changes to the process_config table. If *process_name* is any other process (i.e. a foreground process), the process is not removed until iProcess Engine is restarted.

- *process_instance* is the number of the process instance which you want to delete.

Example If you want to remove an instance of a Background Mbox Daemon process so that instead of having four running instances of the process, you will only have three, enter the following command:

    # swadm delete_process 2 bg 4

This command specifies that on the server with an ID of 2, the fourth instance of the Background Mbox Daemon (BG) process will be removed.

Test Events

The iProcess Engine uses an event mechanism to handle the inter-process communication. If the event mechanism does not work normally, the functions based on event communication are disabled. For example,

• The processes cannot be managed.

• The change of process attributes cannot be updated in real time.

The iProcess Engine allows you to test the event mechanism manually by using the following command:

```
swadm evloopback
```

If the event mechanism works correctly, a loopback message will be received. Otherwise, an error message is displayed.

When Process Sentinel starts, a START event is issued, or a SHUTDOWN event is issued, the event mechanism is tested automatically. For more information, see .

Example The following message is displayed if the events work correctly.

```
Events working correctly. Received loopback message in 1 second(s).
```

Otherwise, the error message similar to the following is displayed:

```
WARNING: Failed to receive the message in 10 seconds.
```

Using SWDIR\util\swsvrmgr to Administer Server Processes

The swsvrmgr utility is used to administer server processes using Process Sentinels. The Process Sentinels operate by subscribing to published internal events such as START a process or PAUSE a process. You can use swsvrmgr to trigger the event types that you want Process Sentinels to subscribe to and then implement.

See "Process Management" in *TIBCO iProcess Engine Architecture Guide* for more information about the concepts of how Process Sentinels work.

To use this utility, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as a background user or root user.

The following table summarizes the swsvrmgr commands you can use to administer server processes.

Command	Task
swsvrmgr STATUS	View Process Status
swsvrmgr START	Issue a Start-up Event
swsvrmgr START_NEW	Issue a Start New Event
swsvrmgr RESTART	Issue a Restart Event
swsvrmgr SHUTDOWN	Issue a Shutdown Event
swsvrmgr PAUSE | UNPAUSE	Issue a Pause or Unpause Event
swsvrmgr DUMPLOG	Write a Shared Memory Debug Log File to Disk
swsvrmgr RESYNCTIME	Resynchronize Timestamps with Windows Time

View Process Status

To view the current state of the system and, optionally, all processes on the system, you can issue a STATUS event to list a status report on the screen using the following command line:

```
swsvrmgr STATUS [-v] [-T timeout]
```

where:

- -v displays the status of all processes on the system

- *timeout* is the optional timeout period that can be used to specify the time after which the command will terminate. If this is not specified, the default is 60 seconds.

The command lists the following information for each process:

- Machine ID is the unique identifier for the server, assigned when the server is added to the iProcess Engine.

- Proc Name is the process name of the server process.

- Proc Inst is the specific instance of the process.

- Status is the current status of the process - either starting, running, paused, shutting down or stopped.

- Comment is a descriptive comment associated with the Status.

Example   The example on the following page displays the system status and the status of all processes.

```
swsvrmgr STATUS -v
Machine ID  Proc Name        Proc       Status        Comment
                             Inst
1           BG               1          RUNNING       BG process started
1           BG               2          RUNNING       BG process started
1           BG               3          RUNNING       BG process started
1           BG               4          RUNNING       BG process started
1           BGPREDICT        1          RUNNING       BG process started
1           DIRECTOR         1          RUNNING       DIRECTOR process started
1           DLMGR            1          RUNNING       DLMGR process started
1           IAPJMS           1          RUNNING       IAPJMS process started
1           RPCBG            1          RUNNING       RPCBG process started
1           RPC_TCP_LI       1          RUNNING       RPC listener process started
1           RPC_UDP_LI       1          RUNNING       RPC listener process started
1           SPO              1          RUNNING       SPO Server process started
1           WIS              1          RUNNING       WIS process started
1           WIS              2          RUNNING       WIS process started
1           WISMBD           1          RUNNING       WISMBD process started
1           WISMBD           2          RUNNING       WISMBD process started
1           WQS              1          RUNNING       WQS process started


Current System Status : 'RUNNING'
```

Issue a Start-up Event

To start the entire iProcess Engine node or start individual processes, you can issue a START event so that Process Sentinels receive the published event and start the required processes.

To issue a START event, use the following command:

```
swsvrmgr START [machine_name|machine_id [process_name [process_instance]]]
[-T timeout]
```

where:

- *machine_name* is the name of the server.

- *machine_id* is the unique identifier of the server. You can find a server's identifier using the swsvrmgr status command.

- *process_name* is the process name of the server process you want to start, and must be one of: BG, BGPREDICT, DBQD, DLMGR, IAPJMS, SPO or DIRECTOR. If any other process name is specified, the command fails and an error message is displayed.

- *process_instance* is the instance of the process to start.

- *timeout* is the optional timeout period that can be used to specify the time after which the command will terminate. If this is not specified, the default is 60 seconds.

When you issue a START event, the swsvrmgr utility will test events automatically. If the events do not work, Process Sentinels will not start processes. See Test Events on page 117 for more information.

Notes All foreground processes (see Server Processes on page 107) must run on the master server.

Example To start the third instance of the background process (BG) that is operating on the computer called hercules using the default timeout, you would issue the following command:

```
swsvrmgr START hercules bg 3
BG 3 STARTED
Process(es) successfully started
```

Issue a Start New Event

You can issue a START_NEW event to start a number of temporary instances of a process. These instances will not be restarted if iProcess Engine is restarted. For example, you may want to start a new background process to cope with a short peak in demand.

To issue a START_NEW event, use the following command:

    swsvrmgr START_NEW [*machine_name*|*machine_id* [*process_name* [*instances*]]]
    [-T *timeout*]

where:

- *machine_name* is the name of the server.

- *machine_id* is the unique identifier of the server. You can find a server's identifier using the swsvrmgr status command.

- *process_name* is the process name of the server process you want to start, and must be one of: BG, BGPREDICT, DBQD, DLMGR, IAPJMS, SPO or DIRECTOR. If any other process name is specified, the command fails and an error message is displayed.

- *instances* is the number of instances of the process to start.

- *timeout* is the optional timeout period that can be used to specify the time after which the command will terminate. If this is not specified, the default is 60 seconds.

Notes All foreground processes (see Server Processes on page 107) must run on the master server.

Example To start an additional instance of the background process (BG) that is operating on the computer called hercules using the default timeout, you would issue the following command:

```
swsvrmgr START_NEW hercules bg 1
BG 5 STARTED
Process(es) successfully started
```

Issue a Restart Event

You can issue a RESTART event to manually restart a suspended process (one that has stopped and not been automatically restarted).

To issue a RESTART event, use the following command:

swsvrmgr RESTART [*machine_name* | *machine_id* [*process_name* [*instance*]]] [-T *timeout*]

where:

- *machine_name* is the name of the server.

- *machine_id* is the unique identifier of the server. You can find a server's identifier using the swsvrmgr status command.

- *process_name* is the process name of the server process.

- *instance* is the instance of the process to restart.

- *timeout* is the optional timeout period that can be used to specify the time after which the command will terminate. If this is not specified, the default is 60 seconds.

Example To restart the third instance of the background process (BG) that is operating on the computer called hercules using the default timeout, you would issue the following command:

```
swsvrmgr RESTART hercules bg 3
BG 3 STARTED
Process(es) successfully started
```

Issue a Shutdown Event

You can issue a SHUTDOWN event to shut down:

- The complete system i.e. all processes are stopped.
- A particular server (in a node cluster).
- Specific types of processes.
- Individual instances of processes.

To issue a SHUTDOWN event, use the following command:

```
swsvrmgr SHUTDOWN [machine_name | machine_id [process_name [instance]]]
[-T timeout]
```

where:

- *machine_name* is the name of the server.
- *machine_id* is the unique identifier of the server. You can find a server's identifier using the swsvrmgr status command.
- *process_name* is the process name of the server process you want to stop, and must be one of: BG, BGPREDICT, DBQD, DLMGR, IAPJMS, SPO or DIRECTOR. If any other process name is specified, the command fails and an error message is displayed.
- *instance* is the instance of the process to stop.
- *timeout* is the optional timeout period that can be used to specify the time after which the command will terminate. If this is not specified, the default is 60 seconds.

> When you issue a SHUTDOWN event, the swsvrmgr utility will test events automatically. If the events do not work, Process Sentinels will not stop processes. See Test Events on page 117 for more information.

The result of the attempt to shutdown each process is displayed on the screen and a final status is displayed at the end.

Example   The following command shuts down the third instance of a background process on the server with Machine ID 1.

```
swsvrmgr shutdown 1 BG 3

Attempting to stop 1 processes
```

Machine ID	Proc Name	Proc Inst	Status	Comment
1	BG	3	SHUTTING DOWN	Normal Shutdown

Issue a Pause or Unpause Event

You can issue a PAUSE event to pause:

- The complete iProcess system.

- A server and all the processes running on it.

- Process types.

- Individual instances of processes.

The following server processes can be paused:

- WIS Mbox Daemon (WISMBD)
- Work Item Server (WIS)
- Background (BG)
- Case Prediction Server (BGPREDICT)
- Deadline Manager (DLMGR)
- Database Queue Daemon (DBQD)

Similarly, you can issue an UNPAUSE event to restart any previously PAUSED process.

To issue a PAUSE or UNPAUSE event, use the following command:

> swsvrmgr PAUSE | UNPAUSE [*machine_name* | *machine_id* [*process_name* [*instance*]]] [-T *timeout*]

where:

- *machine_name* is the name of the server.

- *machine_id* is the unique identifier of the server. You can find a server's identifier using the swsvrmgr status command.

- *process_name* is the process name of the server process you want to stop.

- *instance* is the instance of the process to stop.

- *timeout* is the optional timeout period that can be used to specify the time after which the command will terminate. If this is not specified, the default is 60 seconds.

Example   The following command pauses the third instance of the background process on server hercules.

```
swsvrmgr PAUSE hercules BG 3
BG 3 PAUSED
Process(es) successfully paused
```

The following command restarts the same background process.

```
swsvrmgr UNPAUSE hercules BG 3
```

Write a Shared Memory Debug Log File to Disk

You should only use this command when explicitly requested to do so by TIBCO Support.

To write a shared memory debug log file for a process, use the following command:

```
swsvrmgr DUMPLOG [machine_name | machine_id [process_name [instance]]]
```

where:

- *machine_name* is the name of the server.

- *machine_id* is the unique identifier of the server. You can find a server's identifier using the `swsvrmgr` status command.

- *process_name* is the process name of the server process you want to create a debug log file for.

- *instance* is the instance of the process you want to create a debug log file for.

When this command is issued, all debug in the process' debug shared memory segment is written to the following file, which is located in the *SWDIR*\logs directory:

ProcessName_TimeStamp_ProcessID.dmp

Resynchronize Timestamps with Windows Time

This command is only relevant if you are running iProcess Engine on a Windows system.

To manually force iProcess Engine to resynchronize its timestamps with Windows system time, use the following command:

```
swsvrmgr RESYNCTIME [machine_id]
```

where `machine_id` is the unique identifier of the server that you want to resynchronize. If you omit this parameter, time will be resynchronized on all servers in the iProcess Engine node.

For more information about keeping iProcess Engine timestamps and Windows time synchronized, see the description of the WINTIME_RESYNC_PERIOD process attribute on .

Using the iProcess Server Manager to Administer Server Processes

The iProcess Server Manager is a JSP web client application that utilizes TIBCO Hawk to provide a graphical view of the server processes on a machine or a node cluster. You can do the following from the iProcess Server Manager:

- Start and stop processes (specifically BG, BGPREDICT, DLMGR, IAPJMS, SPO, and DIRECTOR)

- Restart suspended processes

- Start and stop all processes on a selected node or node cluster.

> You can also use the iProcess Server Manager to administer message queues. See Using the iProcess Server Manager to Administer Message Queues on page 350.

Hawk Requirements

If you are planning to use the iProcess Server Manager, you must have TIBCO Hawk installed on:

- The machine hosting iProcess Engine. For more information about how to install TIBCO Hawk, see *TIBCO Hawk Installation and Configuration Guide*.

> TIBCO Hawk is not included in the iProcess engine installation, you have to install it separately.

Enabling the iProcess Server Manager

Windows

On Windows, iProcess Engine uses the iProcess nodename Web Server service (where nodename is the node name of iProcess Engine) to communicate with TIBCO Hawk. This service runs a Tomcat JSP web server that is installed as part of iProcess Engine (in *SWDIR*\tomcat directory.)

You can install the iProcess nodename Web Server service when you install iProcess Engine. See *TIBCO iProcess Engine Installation* for more information.

Alternatively, you can use the following command to install, remove, start or stop the service:

    SWDIR\bin\smserv.bat *action TIBCO_ROOT*

where:

- *action* is either: install, uninstall, start or stop.

- *TIBCO_ROOT* is the base directory for TIBCO software installations on this computer (by default `c:\tibco`).

For example, to install and start the iProcess nodename Web Server service, use the following commands:

```
smserv.bat install C:\tibco
smserv.bat start C:\tibco
```

UNIX

On UNIX, iProcess Engine communicates with TIBCO Hawk using the Tomcat JSP web server that is installed in *SWDIR*/tomcat.) You must start the Tomcat JSP web server by running the following script, which is located in the *SWDIR*/bin directory:

```
smstart
```

If you have installed the TIBCO Hawk software in a location other than the default (/opt/tibco), you must edit the smstart file, which is located in the *SWDIR*\bin directory, to modify the paths specified for *HAWK_ROOT* and *RV_ROOT* to reflect this.

Similarly, to stop the Tomcat JSP web server, run the following script, which is located in the *SWDIR*/bin directory:

```
smstop
```

Configuring Tomcat JSP Web Server to Support HTTPS

If you want to access iProcess Server Manager by using HTTPS, you need to complete the following steps to configure the Tomcat JSP web server:

1. Get a keystore file and specify a password for it by using the `keytool` utility.

```
JAVA_HOME\bin\keytool -genkey -alias alias_name -keyalg RSA
-keystore keystore file_path
```

You are prompted to specify some personal information.

2. Add the HTTPS connector to the server.xml file of Tomcat.

For example, add the following code in the server.xml file, which is located in the *TOMCAT_HOME*\conf directory:

```
<!-- Define a SSL HTTP/1.1 Connector on port 8443 -->
<Connector port="8443" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true"
acceptCount="100" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile="keystore file_path"
keystorePass="keystore file_password" />
```

3. Restart Tomcat JSP web server.

See the Tomcat documentation for more information.

Configuring TIBCO Hawk Security for iProcess Server Manager

The *TIBCO Hawk Installation and Configuration Guide* describes how TIBCO Hawk implements its security policy. TIBCO iProcess Server Manager can be configured to use this TIBCO Hawk policy as follows:

- TIBCO Hawk Agents can be configured to use a specific Java class to implement secure access for console applications such as iProcess Server Manager. Once a Hawk Agent has been configured with a string specifying that security class, this same string must be specified by all console applications that want to connect to this Agent and its microagents.

  Therefore, the iProcess Server Manager provides a file, iprocesshawk.properties, that contains an example of the security string HawkSecurityString. If you edit this string to the value required by the security model of the appropriate Hawk Agent, the Hawk Agent will read the correct value from this file and allow access.

  For example, if your Hawk Agent uses a security class name MySecurity.class, edit the file to read:

```
HawkSecurityString=MySecurity.class
```

The file can be found at:

SWDIR\tomcat\webapps\ipsvrmgr\WEB-INF\classes
\iprocesshawk.properties

With Trusted or TrustedWithDomain (see *TIBCO Hawk Installation and Configuration Guide* for details), you can use the two Hawk default security models:

- The Hawk bin directory (for example, c:\tibco\hawk\bin) needs to be added to your system path. This is because the Java files use Java Native Interface (JNI) to call a Windows DLL (HawkTrustedUserID.dll). If this directory is not on the path, the iProcess Server Manager web application will not work with either of these two security models.

- You must also add the appropriate account to the appropriate access control file used by the TIBCO Hawk Agent's security policy, on all machines that will run the iProcess Server Manager. On Windows, this account is the SYSTEM account. On other platforms, this account is the account that is used to run the SM Start shell script.

Starting the iProcess Server Manager

To start the iProcess Server Manager:

1. If you are running iProcess Engine on:

   — Windows: Make sure the iProcess nodename Web Server and TIBCO Hawk Agent services have been started (see).

   — UNIX/Linux: Make sure you have run the smstart script, which are located in the *SWDIR*/bin directory (see).

2. Enter a URL that has the following format:

   http://*machine*:*port*/

   where

   — *machine* is the machine where the iProcess Server Manager is installed.

   — *port* is the port number of the machine where your iProcess Server Manager is listening to requests. The default is 8080.

   For example:

   http://titan:8080/

   When you first start the iProcess Server Manager, it opens with the Configuration pane displayed:

Figure 1   iProcess Server Manager Configuration Panel

Configuring the iProcess Server Manager

When you first start the iProcess Server Manager, it displays the Configuration pane. Configure the iProcess Server Manager for use in your environment as follows:

1. Enter the name of your Hawk Domain. By default the Hawk Domain is blank, but if you configured a different domain name, enter it here. The name specified for Hawk Domain on the master machine must also be specified on all machines in a clustered environment.

2. If when you installed TIBCO Hawk, you used the defaults for the following TIBCO Rendezvous configuration parameters, continue with the next step:

TIBCO Rendezvous Configuration Parameter	Default Value
Daemon	7474
Network	;
Service	7474

However, if when you installed TIBCO Hawk, you changed TIBCO Rendezvous configuration parameters Daemon, Network or Service from the defaults, you must change the following process attributes in iProcess Engine to reflect this.

— RV_DAEMON

— RV_NETWORK

— RV_SERVICE

3. In the Search for nodes field, enter the name of iProcess Engine node that you want to administer and click the **Search** button.

4. When the iProcess Server Manager locates the node, it displays information about it as follows:

Node Name zjh1141a4s

GUID	DB Type	DB Host	DB Connect String	DB Admin
A7D08C12-A5BD-11E3-A1AD-F04DA2297B50	SQL_SERVER	ZHANGJINHONG-DT	zjh1141a4s	swpro

5. To find TIBCO Hawk Agents associated with the selected node, click the **Browse for Agents** button. If you have already browsed for agents, you can click the **Load Known Agents** button, which is faster than browsing. Also, if the Tomcat software times out, you need to click the **Load Known Agents** button.

Browsing for TIBCO Hawk Agents can take several minutes.

6. The iProcess Server Manager displays the Process Control pane. Continue with the following section to learn more about controlling processes.

Controlling Processes

To view the Process Control pane, expand **iProcess Management** > **Control**.

The iProcess Management page shows information for the server you have selected. The view is hierarchical, so expand a server or a node in a cluster to show individual processes running on each. For example:

Process Control...

▽ zjh1141a4s ▽ △
　　▼ ZHANGJINHONG-DT (Processes = 12) ▽ △
　　　　▶ BG (Instances = 4)
　　　　▶ BGPREDICT (Instances = 1)
　　　　▶ DLMGR (Instances = 1)
　　　　▶ IAPJMS (Instances = 1)

Expanding a process shows the instances of that process:

Using the buttons at the bottom of the page, you can do the following:

Button	Description
Start	Starts the selected instance, all instances of a process, all processes on the selected server, or all processes in the node cluster.
Start Temp	Starts the specified number of temporary instances of the selected process. Specify the number of instances in the text box to the left of the Start Temp button. These instances will not be restarted if the iProcess Engine is restarted. For example, you may want to start a new background process to cope with a short peak in demand.
Stop	Stops the selected instance, all instances of a process, all processes on the selected server, or all processes in the node cluster. You can also force stop processes by selecting the Force stop check box and specifying the number of seconds after which Process Sentinels will stop waiting for processes to shut down cleanly and perform a forced stop (the default is 300 seconds).
Restart	Restarts a process that is in a SUSPENDED state (one that has stopped and not been automatically restarted).

Example

To stop all processes on node staffw_103a6, highlight the node and click the **Stop** button. The message "Requesting Process(es) stop" appears at the bottom of the window and a red square next to each instance indicates that the instance is shutting down or has shut down:

```
▽ staffw_103a6 ▽ △
        ▼ UK-TOMPLAP (Processes = 11) ▽ △
                ▼ BG (Instances = 4)
                        ■ Instance1
                        ■ Instance2
                        ■ Instance3
                        ■ Instance4
```

Viewing Process Statuses

To view the status of all processes, click the **iProcess Management - View** button. The Process View pane lists the currently configured processes, their status, number of instances and so on. For example:

Process View...

MachineID	ProcessName	ProcessInstance	LastKnownStatus	StatusComment	Persistent
1	BG	1	RUNNING	BG process started	true
1	BG	2	RUNNING	BG process started	true
1	BG	3	RUNNING	BG process started	true
1	BG	4	RUNNING	BG process started	true
1	BGPREDICT	1	RUNNING	BG process started	true
1	DLMGR	1	RUNNING	DLMGR process started	true
1	IAPJMS	1	NOT RUNNING		true
1	RPC_TCP_LI	1	RUNNING	RPC listener process started	true
1	RPC_UDP_LI	1	RUNNING	RPC listener process started	true
1	WQS	1	RUNNING	WQS process started	true
1	RPCBG	1	RUNNING	RPCBG process started	true
1	WIS	1	RUNNING	WIS process started	true
1	WIS	2	RUNNING	WIS process started	true
1	WISMBD	1	RUNNING	WISMBD process started	true
1	WISMBD	2	RUNNING	WISMBD process started	true
1	SPO	1	RUNNING	SPO Server process started	true
1	DIRECTOR	1	RUNNING	DIRECTOR process started	true

Note that the IAPJMS process is disabled, and therefore appears "greyed out." The display is refreshed every 5 seconds. You can configure the refresh interval as described in .

View the Process Summary

To view summary information about processes, click the **iProcess Management** - **Summary** button. The Process Summary pane lists the process name, number of processes, and the parent/child relationships. For example:

Process Summary...

ProcessName	NumInstances	NumRunning	ParentType	NumChildren	ChildrenPaused
BG	4	4		0	false
BGPREDICT	1	1		0	false
DLMGR	1	1		0	false
IAPJMS	1	0		0	false
RPC_TCP_LI	1	1		2	false
RPC_UDP_LI	1	1	RPC_TCP_LI	0	false
WQS	1	1	RPC_TCP_LI	4	false
RPCBG	1	1	WQS	0	false
WIS	2	2	WQS	1	false
SPO	1	1	WQS	0	false
DIRECTOR	1	1	WQS	0	false
WISMBD	2	2	WIS	0	false

Viewing the iProcess Engine Log Files

To view log files related to iProcess Engine, click the **Log Viewer** button. The Log Viewer pane is displayed:

Log Viewer...

| Server | ZHANGJINHONG-DT ▼ | | Log file filter | * | | Get Logs |

To view a log file, do the following:

1. From the Server list, select the server that contains the log files you want to view.

2. Use a wildcard, if desired in the Log file filter field.

3. Click the **Get Logs** button.

4. From the Log Files list, select the log file you want to view (for example, `sw_error`). The list contains all the log files found in the *SWDIR*`\logs` directory that matched the criteria you entered in the Log file filter field.

5. Enter the number of lines of the log file that you want to display and click either the **From Top of Log** button or the **From End of Log** button, depending on which part of the log file you want to view. You can also wrap lines by selecting the Wrap Lines check box.

6. The requested portion of the log file is displayed. For example:


```
2005/09/14 14:28 wqsrpc: normal shutdown
2005/09/15 12:35 wqsrpc: normal shutdown
2005/09/15 13:34 wqsrpc: normal shutdown
```

You can use the following buttons for navigation:

— Next - displays the next portion of the log file.

— Previous - displays the previous portion of the log file.

Customizing the iProcess Server Manager

There are several configuration options that you can change by editing the
configuration.xml file, which is located in the
SWDIR\tomcat\webapps\ipsvrmgr directory. For example, to change the default
refresh period (5 seconds), edit the following entry:

```
<!-- page refresh interval in seconds -->
<refresh>5</refresh>
```

Connecting to a Different Server

By default, the iProcess Server Manager displays the node cluster you are part of
or the individual server that you are using (if you are not part of a node cluster).
You can connect to other servers as follows:

1. Click the **Configuration** button.

2. Enter the node name of the server you are looking for in the Search for nodes
field and click the **Search** button.

After a short delay, the details of the requested node should be displayed in
the Configuration pane.

Chapter 6　**Administering Process Attributes**

This chapter describes how to use the `swadm` server configuration utility, which is located in the *SWDIR*\`util` directory, to administer iProcess Engine process attributes.

Each iProcess Engine server process can have associated attributes to specify how the process operates. Process attributes and their values are stored in the `process_attributes` database table.

Topics

Using SWDIR\util\swadm to Administer Process Attributes

You can use the swadm utility to view, set and delete process attributes. Note that:

- To use this utility, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as a background user or root user.

- If you are using a node cluster architecture, you can run this utility from any server within the cluster (as long as that server has a connection to the TIBCO iProcess Engine database instance).

The following table summarizes the commands you can use to administer process attributes.

Command	Task
swadm show_all_attributes	Display All Process Attributes
swadm set_attribute	Set a Process Attribute
swadm delete_attribute	Delete a Process Attribute

Display All Process Attributes

You can display a list of all process attributes and their values that are currently defined on iProcess Engine.

The following command enables you to set a filter for attribute names so that you can either display all attributes on all servers or display all attributes of a certain name on all servers:

```
swadm SHOW_ALL_ATTRIBUTES [attribute_name]
```

where *attribute_name* is the (optional) name of the process attribute that you want to restrict the search by. For a list of valid process attribute names see Alphabetical List of Process Attributes on page 146.

Set a Process Attribute

You can set up a new attribute for a specific server process or update an existing entry using the following command:

swadm SET_ATTRIBUTE *machine_id* *process_name* *process_instance* *attribute_name* *attribute_value*

where:

- *machine_id* is the unique identifier for the server. If you specify a value of 0, the command will apply to all servers in iProcess Engine.

- *process_name* is the name of the iProcess Engine process. If you specify a value of ALL, the command will apply to all process types.

- *process_instance* is the instance number of the process. If you specify a value of 0, the command will apply to all instances of the process.

- *attribute_name* is the name of the attribute to be set.

- *attribute_value* is the value for the specified process attribute.

Example A company's office in California (Pacific Standard Time, GMT-08:00) wants to run cases of procedures that are hosted on a node running on a machine in the company's administrative centre in Washington D.C. (Eastern Standard Time, GMT-05:00).

To configure iProcess Engine to use Pacific Standard Time, use the following command:

swadm set_attribute 0 ALL 0 TIMEZONE "PST8"

Delete a Process Attribute

You can remove a process attribute from a server process so that the attribute no longer effects the process and is removed from the process_attributes table. Use the following command:

    swadm DELETE_ATTRIBUTE *machine_id* *process_name* *process_instance*
    attribute_name

where:

- *machine_id* is the unique identifier for the server. If you specify a value of 0, the command will apply to all servers in iProcess Engine.

- *process_name* is the name of the iProcess Engine process. If you specify a value of ALL, the command will apply to all process types.

- *process_instance* is the instance number of the process. If you specify a value of 0, the command will apply to all instances of the process.

- *attribute_name* is the name of the attribute to be deleted.

Example If the third instance of the BG process keeps failing but it has been set up to automatically restart, you can stop it restarting while you investigate the reason why it keeps failing. Enter the following command:

    swadm delete_attribute 1 bg 3 process_auto_restarts

Alphabetical List of Process Attributes

The following table describes the available process attributes.

Process attributes that are used by the DIRECTOR process are not listed in this table. See *TIBCO iProcess Objects Director Administrator's Guide* for more information about attributes that are used by the DIRECTOR process.

Attribute	Description
AUDIT_CASEDATA_CHANGED	Defines whether or not to log the case data changes in the audit trail when these changes are made by iProcess Insight, iProcess Workspace (Browser), or the setCaseData TIBCO iProcess Server Objects interface, rather than by normal step processing.
AUDIT_OPENKEEP	Controls whether the Open Work Item and Keep Work Item audit trail entries are enabled.
AUTO_PURGE_DELAY	Defines the number of days to delay the auto-purge operation.
BG_LEAVER_DESTINATION	Defines the destination queue to which work items are redirected when the owner of the work items has been deleted.
BG_LEAVER_SUPER_AS_DEST	Defines to the redirect location of work items for a currently deleted user.
BG_MAX_ACTIONS_PER_TRANS	Defines the limit of actions per workflow transaction.
CCOUNT_CACHE_REFRESH	Enables you to define the refresh period for updating the cached list of cases currently on the system.
CHECK_EAIWITHDRAW_ONPURGE	Defines whether or not iProcess checks if any outstanding delayed release EAI steps have been successfully withdrawn before committing the purge transaction.
CHECKFREQ	Defines the number of loops to process before the background process checks for sw_error.log files, which are located in the *SWDIR*\logs directory and available disk space.

Attribute	Description
CNUM_SEQ_CACHE	Defines the number of case numbers to be cached.
CSTART_AUTO_REFRESH	Defines whether or not the list of available procedures in the TIBCO iProcess Workspace's Case Start dialog is automatically refreshed.
DBGMEMSIZE_KB	Defines the size of shared memory segment (in Kb) that should be allocated for shared memory debug logs.
DBQD_MAX_CACHED_MESSAGES	Defines the number of messages that are cached by the DBQD process when it requests a block of messages from a database message queue.
DBQD_MAX_FIL_SESSIONS	Defines the number of concurrent threads that the DBQD process uses to process RPC requests for messages from its cache from BG or WISMBD processes.
DEF_MAJOR_VERS	Defines the default major version number that TIBCO iProcess Modeler will use when a new procedure is saved.
DEF_MINOR_VERS	Defines the default minor version number that TIBCO iProcess Modeler will use when a new procedure is saved.
DEPLOY_XSL_OUT_ENCODING	Defines the character set to be used for output encoding.
DISABLE_CASE_COUNTING	Defines whether case counts are displayed for procedures in the Live (Dead) Cases column of the Case Administrator dialog, when a user starts iProcess Administrator from iProcess Workspace (Windows).
DISABLE_USER_CHECK	Defines whether or not a new user name is validated as an O/S user account when you add an iProcess user from the User Manager tool of TIBCO iProcess Administrator.
DISABLE_USER_LIST	Defines whether or not the Possible iProcess User List button is displayed in the User Manager tool of TIBCO iProcess Administrator.

Attribute	Description
DMD_PROCESS_INTERVAL	Defines the times during the day when the Deadline Manager checks the iProcess database for expired deadlines.
EAI_NEEDS_MSDTC	Defines the EAI server plug-ins that need to use the Microsoft Distributed Transaction Coordinator (MSDTC).
EAI_STEP_TIMEOUT	Defines a time threshold (in milliseconds) for the duration of processing an EAI step.
EMPTYMBOXSLEEP	Defines how long the Mbox Daemons will "sleep" when all Mbox queues in the Mbox set are empty.
EMPTYMBOXSLEEP_INC	Defines the number of seconds to increment the EMPTYMBOXSLEEP value by when a BG or WISMBD process requests a message from an empty Mbox.
EMPTYMBOXSLEEP_MAX	Defines the maximum value (in seconds) that EMPTYMBOXSLEEP can be set to.
ENABLE_CASE_PREDICTION	Defines whether or not background case prediction is enabled on the node. **Note**: This attribute has no effect on live case prediction or case simulation.
EXACT_OPERATION_TIMESTAMP	Defines whether the audit_date timestamp, which is written in the audit_trail table, records when the user operates a specified instruction or when the BG processes a specified instruction.
FIL_PROCDEF_CACHE_SIZE	Defines the maximum number of procedure definitions that can be cached in memory by the BG, WIS and SPO processes.
FORCE_DEADLINE_PRIORITY	Defines a priority value for case deadlines that override all other priorities set for case deadlines.
FREE_WIS_SESSION	Defines whether the WIS session, which is used to retrieve work items, should be freed up in the WIS memory or not after the lock, keep, forward, release, undo, or unlock operation.

Attribute	Description
IAPJMS_LANGUAGE	Defines the character set to be used for output encoding for messages published to the IAPJMS process from the BG or WIS process.
IAPJMS_PORTNO	Defines the port number that is used for message communications between the BG process and the IAPJMS library.
IAPJMS_PUBLISH	Defines whether or not the BG process is enabled to publish audit activities to the IAPJMS process.
IAPJMS_ROLLBACK	Defines whether or not failed message transactions should be rolled back.
IAPJMS_SIMPLETOPIC	Defines whether or not the JMS topic name is static or dynamically configured at run-time.
IAPJMS_SYNCHRONOUS	Defines whether message delivery is synchronous or asynchronous.
IAPJMS_TIMEOUT	Defines how long the IAPJMS process should wait before it times out if there is a network error.
IAPJMS_TOPICNAME	Defines the topic name for the JMS destination if activity monitoring is enabled.
IGNORE_PACK_CHANGED	Defines whether users may keep or release work items even if pack data has changed.
IQL_RETRY_COUNT	Defines how many times a failed message in a message queue is retried before being moved to the exception queue.
IQL_RETRY_DELAY	Defines the delay (in seconds) between each retry attempt for a failed message in a message queue, before the message is moved to the exception queue.
JVMPROPS	Defines the JVM attributes that should be specified for the Java Virtual Machine when it is started.
LOGON_OS_LOCATION	Defines the default location where passwords should be validated when a user attempts to log in to this iProcess Engine node.

Attribute	Description
MAINCASE_START_AUTOCOMMIT	Defines whether or not to insert a Transaction Control step automatically as the first step of all main cases.
MAX_AGE_BEFORE_RESETPOST	Defines the time limit in seconds before the Deadline Manager will re-post unprocessed deadline messages.
MAX_PREDICTION_LOOPS	Defines the maximum number of times to loop during the prediction process.
MAX_SUB_PROCEDURE_DEPTH	Determines the maximum number of nested sub-procedures supported by the server.
MBSET_DEQUEUE_TIMEOUT	Defines a time threshold (in milliseconds) for the duration of dequeuing a message from an Mbox set.
MBSET_READ_BG	Defines the unique identifier of the Mbox set to be used by a BG process when dequeuing messages received from a WISMBD process.
MBSET_READ_PREDICT	Defines the unique identifier of the Mbox set to be used by a BGPREDICT process when posting case changes messages to a BG process.
MBSET_READ_WIS	Defines the unique identifier of the Mbox set to be used by a WISMBD process when dequeuing messages received from a BG process.
MBSET_WRITE_BG	Defines the unique identifier of the Mbox set to be used by a process when writing to a BG process.
MBSET_WRITE_PREDICT	Defines the unique identifier of the Mbox set to be used by a BGPREDICT process when posting case changes messages to a BG process.
MBSET_WRITE_WIS	Defines the unique identifier of the Mbox set (as defined in the mbox_set table) to be used by the BG process when writing to a WISMBD process.
MIGRATE_LIVE_CASES	Defines whether or not to migrate only live cases from previous versions to a new version when deploying from TIBCO Business Studio to TIBCO iProcess Engine.

Attribute	Description
MINFREEKB	Defines the amount of disk space (in Kilobytes) required for the background process to run.
MN_UNLOCKER	Defines who has permission to unlock the work items.
NORMALISE_CASE_DATA	Defines whether or not case data normalization is enabled.
OS_USER_LOCATIONS	Defines where iProcess Engine should obtain the list of users when it populates the Possible iProcess User List in the User Manager tool of TIBCO iProcess Administrator.
PM_AUTO_BOOT	Defines whether or not the Process Sentinels automatically start the server processes after the Process Sentinels have started.
PROC_VER_COMMENT	Defines whether or not, in TIBCO iProcess Modeler, a user has to enter a comment whenever they save a procedure.
PROC_VER_INC	Defines whether or not, in TIBCO iProcess Modeler, a procedure's version number will be incremented whenever it is saved.
PROC_VER_NUM_INSTANCES	Defines the maximum number of instances of a procedure version.
PROCESS_AUTO_DUMPLOG	Defines whether or not, if a process fails, the Process Sentinels automatically write to disk the contents of that process' debug shared memory segment.
PROCESS_AUTO_RESTARTS	Defines whether or not a server process will automatically restart after a failure.
PROCESS_MAX_RESTARTS	Defines the maximum number of times the Process Sentinels will attempt to restart a failed process.
PROCESS_MIN_RESTART_PERIOD	Defines a period of time (in seconds) during which Process Sentinels suspends or restarts a failed process.

Attribute	Description
PROCESS_SLEEP	Defines the amount of time the Process Sentinels will sleep for.
PUBLISH_SYS_EVENT_METHOD	Defines in which way you want to publish system events.
REQID_SEQ_CACHE	Defines the number of REQ IDs to be cached.
RESTART_SPO_CACHE_PROC	Determines how many of the latest versions of the procedure definition to cache when the iProcess Objects Server process restarts or quick starts.
RESTART_WIS_CACHE_THRESHOLD	Defines the number of items that must exist in a work queue for it to be cached when the WIS process restarts or quick starts.
RETRY_OFF_FOR_RAC	Determines whether to enable the retry function that retrieves work items from the staffo database table for the configured Oracle RAC.
RPC_BLOCK	Defines whether or not iProcess Workspace (Windows) is able to access iProcess Engine.
RPC_SVR_CONTROL	Allows the batching of RPC calls to reduce the overhead in processing RPC calls individually.
RPC_SVR_NUM_THREADS	Defines the maximum number of threads that the WIS and WQS processes can use to process RPC requests from client applications.
RPC_TIMEOUT	Defines the timeout period (in seconds) that an RPC request waits for a response from the iProcess Engine process.
RV_DAEMON	Configures the iProcess Server Manager with the daemon used to handle session communication in TIBCO Rendezvous.
RV_NETWORK	Configures the iProcess Server Manager with the network used to handle outbound session communication in TIBCO Rendezvous.

Attribute	Description
RV_SERVICE	Configures the iProcess Server Manager with the User Datagram Protocol (UDP) service group used to handle session communication in TIBCO Rendezvous.
SE_WORKER_PORTNO	Defines the port number that is used for message communications between the "worker" process (Process Sentinel) and the processes that audit system events if you want to published system events using TIBCO Hawk Agent.
SEJMS_TOPICNAME	Defines the JMS topic name for the JMS destination that is used to publish system events, if publishing system events is enabled.
SHMKEY_ID	Defines the UNIX shared memory key that is allocated (using the ftok system call) when the WQS process is started.
SHUTDOWN_MIGRATION_MESSAGE	Defines whether or not to contain the audit trail message (with message ID 34), which writes the information about procedure migration, in the published Monitor Event Detail message.
SPO_CACHE_PROC	Determines how many of the latest versions of the procedure definition to cache when the iProcess Objects Server process starts normally.
SPO_USERMUTEX_WAITTIME	Defines the maximum amount of time (in milliseconds) that an iProcess Objects Server process should take to acquire for a mutual exclusion lock (mutex) before processing a message if multiple SSO clients use the same user name to log on TIBCO iProcess Engine. If the time for acquiring a user's mutex is over the time specified by the SPO_USERMUTEX_WAITTIME attribute, the duration will be recorded and later reported in the sw_warn log.
SUBCASE_START_AUTOCOMMIT	Defines whether or not to insert a Transaction Control step automatically as the first step of all sub-cases.
SWLIB_PATH	Defines the directory where the IAPJMS process will look for the Java libraries that it needs.

Attribute	Description
SYSTEM_EVENT_LOGGING	Defines whether you want to audit system events after installing or upgrading to TIBCO iProcess Engine 11.6.0.
TIMEZONE	Defines the time zone that this node will operate in.
UNPROCESSED_DL_POST_LIMIT	Sets a limit on the number of unprocessed deadline messages that are posted by the Deadline Manager.
USE_NEXT_MESSAGE_TO_DEQUEUE	Defines the ways to purge vast quantities of cases in the Oracle AQs.
USE_PRIORITY_DEADLINES	Defines whether to set the priority for the case deadlines by using the value of the SW_CP_VALUE field if the value of the FORCE_DEADLINE_PRIORITY attribute is set to the default value, 0.
WAITID_SEQ_CACHE	Defines the number of Wait IDs to be cached.
WARN_ERROR_LOG_SIZE	Defines the maximum size (in bytes) of the sw_warn and sw_error log files before the files stop logging error messages.
WINTIME_RESYNC_NOTICE	Defines the notice period (in seconds) that iProcess Engine processes are given before a resynchronization takes place.
WINTIME_RESYNC_PERIOD	Defines the interval (in seconds) at which iProcess Engine checks to see if its timestamps are in step with Windows system time.
WINTIME_RESYNC_TOLERANCE	Defines the interval (in seconds) at which iProcess Engine checks to see if its timestamps are in step with Windows system time.
WIS_CACHE_POOL_SIZE	Defines the size (in threads) of the pool of threads that is used to perform caching of work queues.
WIS_CACHE_THRESHOLD	Defines the number of items that must exist in a work queue for it to be cached when the WIS process starts normally.

Attribute	Description
WIS_CACHE_WAIT_TIME	Defines the maximum amount of time (in seconds) that an RPC processing thread in the WIS process waits for a work queue to be cached.
WIS_CDQP_DATA_RECACHE_BATCH	Defines the number of work items that the CDQP update thread will update in a single operation when updating CDQP field values for a WIS process' queues.
WIS_CHECK_STAFFO_RETRY_COUNT	Defines the number of times the WIS process attempts to retrieve the unfound work items from the staffo database table because a delay occurred between two Oracle RAC instances. **Note**: This attribute is only used on configured Oracle RAC.
WIS_FILTER_THREAD_BOUNDARIES	Defines the count boundary at which a work queue will be split into multiple blocks of work for filtering purposes, based on the number of work items in the queue
WIS_FILTER_THREAD_POOL_SIZE	Defines the number of threads in the queue filtering thread pool, used to process additional blocks of filtering work
WIS_INDEX_REFRESH	Defines the interval (in seconds) after which an index on a queue will be refreshed by a WIS process.
WIS_LOCK_POOL_SIZES	Defines the number of locks in the internal lock pool used by the WIS process
WIS_NEW_ITEM_BATCH_SIZE	Defines the number of new item requests to be batched together.
WIS_QCHANGE_EXTENDED_CHECK	Defines whether or not a change in the lock status of a work item is counted as a change to the work item.
WIS_SESSION_TIMEOUT	Defines the timeout period (in seconds) after which a WIS process will automatically shut down, starting from the time at which it was last accessed (by a TIBCO iProcess Workspace, SAL application or TIBCO iProcess Objects Server).

Attribute	Description
WIS_SESSION_TIMEOUT_SHUTDOWN	Defines the timeout period (in seconds) after which a `WIS` process will automatically shut down, starting from the time at which iProcess Engine was shut down.
WIS_UNCACHE_PERIOD	Defines the time period (in minutes) after which the cached work queue that waits to be handled by the `WIS` process is cleared from the cache, starting from the time at which the client who last accessed this work queue logs off. **Note**: This attribute only applies to the work queues that contain a number of work items less than the number you defined in the `WIS_CACHE_THESHOLD` attribute.
WIS_UPDATE_LENGTH	Defines the maximum amount of time (in seconds) that the queue update thread in the `WIS` process performs updates for before going back to sleep
WIS_UPDATE_PERIOD	Defines how often the queue update thread in the `WIS` process wakes up and updates the queues handled by the `WIS` process.
WIS_USE_PRIORITY_ESCALATION	Allows the use of priority escalation in the `WIS` process to be disabled.
WQDJMS_PORTNO	Defines the port number that is used for work queue delta messages between the `WIS` process and the `IAPJMS` process.
WQDJMS_TOPICNAME	Defines the default topic name for the JMS destination used by the `WIS` process for work queue delta publication.
WQS_GATHER_RPC_STATS	Allows the gathering of RPC call stats within the `WQS` process to be configurable.
WQS_NUM_SEARCH_SLOTS	Defines the maximum number of slots available in the SWRPCMTS multi-threaded RPC server shared library for threads to perform queue searching.

Attribute	Description
WQS_PERSIST_SHMEM	Defines how often (in seconds) the contents of the WQS/WIS shared memory are written to the `wqs_index` table in the database.
WQS_WIS_USER_COUNT	Defines the number of `WIS` processes that should be dedicated to handling user queues and group queues respectively.
XPC_READ_UNCOMMITTED	Allows configuration of uncommitted reads during an XPC SELECT.

General iProcess Engine Configuration

The following process attributes allow you to configure general aspects of iProcess Engine behavior.

Attribute	Description
DBGMEMSIZE_KB	Defines the size of shared memory segment (in Kb) that should be allocated for shared memory debug logs.
DEPLOY_XSL_OUT_ENCODING	Defines the character set to be used for output encoding.
EAI_NEEDS_MSDTC	Defines the EAI server plug-ins that need to use the Microsoft Distributed Transaction Coordinator (MSDTC).
EXACT_OPERATION_TIMESTAMP	Defines whether the audit_date timestamp, which is written in the audit_trail table, records when the user operates a specified instruction or when the BG processes a specified instruction.
FREE_WIS_SESSION	Defines whether the WIS session, which is used to retrieve work items, should be freed up in the WIS memory or not after the lock, keep, forward, release, undo, or unlock operation.
LDAP_UV	This attribute turns on and off the LDAP integral User Validation API.
LOGON_OS_LOCATION	Defines the default location where passwords should be validated when a user attempts to log in to this iProcess Engine node.
MIGRATE_LIVE_CASES	Defines whether or not to migrate only live cases from previous versions to a new version when deploying from TIBCO Business Studio to TIBCO iProcess Engine.
NORMALISE_CASE_DATA	Defines whether or not case data normalization is enabled.

Attribute	Description
RPC_TIMEOUT	Defines the timeout period (in seconds) that an RPC request waits for a response from the iProcess Engine process.
SPO_USERMUTEX_WAITTIME	Defines the maximum amount of time (in milliseconds) that an iProcess Objects Server process should take to acquire for a mutual exclusion lock (mutex) before processing a message if multiple SSO clients use the same user name to log on TIBCO iProcess Engine. If the time for acquiring a user's mutex is over the time specified by the SPO_USERMUTEX_WAITTIME attribute, the duration will be recorded and later reported in the sw_warn log.
TIMEZONE	Defines the time zone that this node will operate in.
WARN_ERROR_LOG_SIZE	Defines the maximum size (in bytes) of the sw_warn and sw_error log files before the files stop logging error messages.
WINTIME_RESYNC_NOTICE	Defines the notice period (in seconds) that iProcess Engine processes are given before a resynchronization takes place.
WINTIME_RESYNC_PERIOD	Defines the interval (in seconds) at which iProcess Engine checks to see if its timestamps are in step with Windows system time.
WINTIME_RESYNC_TOLERANCE	Defines the interval (in seconds) at which iProcess Engine checks to see if its timestamps are in step with Windows system time.
XPC_READ_UNCOMMITTED	Allows configuration of uncommitted reads during an XPC SELECT.

DBGMEMSIZE_KB

General iProcess Engine Configuration

Summary   This attribute specifies the size of shared memory segment (in Kb) that should be allocated for shared memory debug logs created either by the TIBCO iProcess Objects Server, or by using the `swsvrmgr DUMPLOG` command under the *SWDIR*\util directory.

Applies To   The attribute must be set for ALL processes.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	256

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   Setting this attribute allows the TIBCO iProcess Objects Server to size the shared memory segment that it uses to create shared memory debugging, without having to set a SAL debug string.

DEPLOY_XSL_OUT_ENCODING

General iProcess Engine Configuration

Summary	This attribute defines the character set to be used for output encoding for procedures imported to iProcess from TIBCO Business Studio. Specifying the correct character set ensures that the Description and Extended Description fields are displayed correctly.
Applies To	The attribute must be set for ALL processes.
Default Value	This attribute is not defined on a newly installed iProcess Engine. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	iso-8859-1

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes	The value of this attribute overrides any setting in the xpdl2xfr.xslt file.

EAI_NEEDS_MSDTC

General iProcess Engine Configuration

This attribute is only relevant to the Windows version of iProcess Engine. It has no effect on the UNIX version.

Summary   This attribute defines the EAI server plug-ins that need to use the Microsoft Distributed Transaction Coordinator (MSDTC).

Applies To   The attribute can be set for the BG, BGPRDICT and RPCBG processes.

Permissible   The attribute value must be a comma-delimited list of EAI step names. The name
Values   used should be the same name used to register the EAI server plug-in.

Default Value   The attribute is assigned the following default values when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	BG	0	EAICOM
0	BGPREDICT	0	EAICOM
0	RPCBG	0	EAICOM

Notes   You should set this attribute for any EAI server plug-ins that you develop that require the use of the MSDTC. If you don't do so, EAI steps using the plug-in may not function correctly or in a fully transactional manner.

Currently, the only TIBCO iProcess server plug-in that requires the use of the MSDTC is the TIBCO iProcess COM Server Plug-in. The default value for this attribute is therefore set to EAICOM.

When a BG process loads an EAI server plug-in, it will check to see if the plug-in's name is specified in the EAI_NEEDS_MSDTC value. If it is, it turns on the use of the MSDTC. For more information about:

- the MSDTC, see "What is MSDTC" in *TIBCO iProcess Engine Architecture Guide*.

- EAI server plug-ins, see Managing EAI Step Server Plug-ins on page 427, and "Using Enterprise Application Integration (EAI) Steps" in *TIBCO iProcess Modeler Integration Techniques*.

- the EAI COM server plug-in, see *TIBCO iProcess COM Plug-in User's Guide*.

EXACT_OPERATION_TIMESTAMP

General iProcess Engine Configuration

Summary	This attribute defines whether the audit_date timestamp, which is written in the audit_trail table, records when the user does a specified operation or when BG or RPCBG processes an instruction.
Applies To	The attribute can be set for the BG, RPCBG, or ALL processes.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
0	The time when the BG processes an instruction is recorded in audit_date in the audit_trail table.
1	The time when the user does a specified operation is recorded in audit_date in the audit_trail table.

Default Value   This attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

Notes   If the value of this attribute is set to 1, the timestamp is recorded in the audit_trail table only when the user does specified operations. The following table shows all the specified operations and their corresponding audit trail messages. See Understanding Audit Trails on page 449 for a complete listing of audit trail messages and their corresponding Message IDs.

Operation	Activity (Message ID)
Release	*StepDescription* released by to *UserName* (002)
Start a new case	Case started by *UserName* (000)
Redirect	*StepDescription* redirected to *UserName* (021)
Forward	*StepDescription* forwarded to *UserName* (004)
	StepDescription forwarded by *UserName* (041)
Event	*StepDescription* event issued by *UserName* (015)

Operation	Activity (Message ID)
Open	*stepdescription* opened by *username* (059)
Keep	*stepdescription* kept by *username* (060)
Custom audit message	user defined audit messages (message ID >255)
Suspend	Case Suspended by *UserName* (022)
Resume	Case Resumed by *UserName* (023)
Jump to	*StepDescription* Case Jump by *UserName* (024)
Release the first step after starting a case	Case started by *UserName* (000)

See Also None.

FREE_WIS_SESSION

General iProcess Engine Configuration

Summary   This attribute defines whether the WIS session, which is used to retrieve work items, should be freed up in the WIS memory or not after the lock, keep, forward, release, undo, or unlock operation.

> This attribute is only used for doing the lock, keep, forward, release, undo, and unlock operations.

Applies to   This attribute should be set for the SPO processes.

Permissible Values   The attribute value must be one of the following.

Value	Meaning
0	The WIS session, which is used to retrieve work items by TIBCO iProcess Objects Server, is saved in the WIS memory after the lock, keep, forward, release, undo, or unlock operation.
1	The WIS session, which is used to retrieve work items, is freed up in the WIS memory after the lock, keep, forward, release, undo, or unlock operation.

Default Value   This attribute is not defined automatically when you install or upgrade iProcess Engine. The WIS session that is used to retrieve work items by TIBCO iProcess Objects Server is saved in the WIS memory by default.

Notes   Details of how to use this attribute are as follows:

- If the WIS session that is used for retrieving work items is saved in the WIS memory, the TIBCO iProcess Objects Server can reuse the existing WIS session to retrieve work items. Therefore, the average time to retrieve work items is reduced. However, saving WIS sessions in the WIS memory may take up a lot of memory space.

- If the WIS session that is used to retrieve work items is freed after each operation, the WIS session needs to be re-created when another operation is performed. So the average time to retrieve work items is increased. However, cleaning up the WIS session in the WIS memory may save a lot of memory space.

See Also   None.

LDAP_UV

General iProcess Engine Configuration

Summary	This attribute turns on and off the LDAP User Validation API. See *LDAPCONF Utility User's Guide* for further details of user validation using LDAP.
Applies To	The attribute must be set for ALL processes.
Permissible Values	The attribute value must be one of the following.

Value	Meaning
0	LDAP user validation is disabled.
1	LDAP user validation is enabled.

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

LOGON_OS_LOCATION

General iProcess Engine Configuration

Summary   This attribute defines the default location where passwords should be validated when a user attempts to logon to this iProcess Engine node.

> This attribute is only used on the Windows variant of iProcess Engine. It has no effect if it is set on a UNIX system.

Applies To   The attribute must be set for ALL processes.

Permissible Values   The attribute value must be a text string containing a single valid machine name or domain name.

Default Value   This attribute is not defined automatically when you install or upgrade iProcess Engine. To use this attribute, you must explicitly assign a value to it using the SET_ATTRIBUTE command.

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command.

Notes   If the iProcess Engine is running on a machine that is a domain member or domain controller, the user account could exist in multiple places. The iProcess Engine node therefore uses the following search path to find the location it should use to validate the user's password:

1. the value of the user's SW_DOMAIN user attribute (if defined). This attribute specifies a single valid machine name or domain name that should be used to validate a particular user's password. (See *TIBCO iProcess Windows (Workspace) Manager's Guide* for more information about this attribute and how to set it.)

2. the LOGON_OS_LOCATION value (if defined).

3. the search path provided by the Windows LookupAccountName function (which iProcess Engine uses to find the user's account name). This path is:

   a. well-known Windows security identifiers. (A security identifier (SID) is a unique value that identifies a security principal or security group in

> Windows operating systems. Well-known SIDs are a group of SIDs that identify generic users or generic groups.)

  b. built-in and administratively defined local accounts.

  c. the primary domain.

  d. trusted domains.

Note that:

- If both attributes are set, the SW_DOMAIN value takes precedence over the LOGON_OS_LOCATION value.

- If iProcess Engine is running on a standalone machine, passwords are always validated against local machine accounts. The SW_DOMAIN and LOGON_OS_LOCATION attributes are ignored even if they are set.

If the SW_DOMAIN or LOGON_OS_LOCATION attribute is defined, iProcess Engine checks to see if the user account exists in that location. If the account does not exist there, or if the password does not match the one defined, password validation fails. An error is also written to the sw_warn file indicating that a mismatch has occurred. For example:

```
2006/11/30
13:23:16(BENCHTST:1968:1968:0:aduser1:filosuvm.c:1.18:373):
1631-WARNING: <LogoniProcessUser (): LookupAccountName(ssfsf)
failed: No mapping between account names and security IDs was
done.> <> <> <>
```

```
2006/11/30
13:23:16(BENCHTST:1968:1968:0:aduser1:filosuvm.c:1.18:373):
1631-WARNING: <LogoniProcessUser (): LogonUser(auser1@UK-BONDIC)
failed: Logon failure: unknown user name or bad password.> <> <> <>
```

You should define LOGON_OS_LOCATION (or the SW_DOMAIN user attribute) if user accounts with the same name exist in two or more trusted domains, because you cannot guarantee which domain the LookupAccountName function will check first, and so pick the account information from. Consequently, a logon attempt may fail because it is validated against the wrong domain.

If you use a UVAPI package to perform password validation, you should note that using the LOGON_OS_LOCATION and/or SW_DOMAIN attributes requires that you use extended (_ex) versions of some UVAPI interfaces. The extended interfaces support the passing in and out of user location information from the SW_DOMAIN user attribute and/or LOGON_OS_LOCATION process attribute. (The old interfaces are still supported, but if you use them the location of the user is not passed down from LOGON_OS_LOCATION or SW_DOMAIN attributes.)

See *TIBCO iProcess User Validation API User's Guide* for more information.

MIGRATE_LIVE_CASES

General iProcess Engine Configuration

Summary   This attribute defines whether or not to migrate only live cases from previous versions to a new version when deploying from TIBCO Business Studio to TIBCO iProcess Engine.

Applies To   The attribute must be set for the `RPC_TCP_LI` processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	Migrate all cases from previous versions to a new version when deploying from TIBCO Business Studio to TIBCO iProcess Engine.
1	Migrate only live cases from previous versions to a new version when deploying from TIBCO Business Studio to TIBCO iProcess Engine.

Default Value   The attribute is assigned the following default value when an iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	RPC_TCP_LI	0	0

Notes   Case migration is not supported by TIBCO Business Studio currently.

See Also   None.

NORMALISE_CASE_DATA

General iProcess Engine Configuration

Summary	This attribute defines whether or not case data normalization is enabled.
Applies To	The attribute must be set for ALL processes.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
0	Case data normalization is disabled.
1	Case data normalization is enabled.

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0 or 1

The default value is chosen by the user when they install or upgrade the iProcess Engine node.

Notes This attribute can be set during an installation/upgrade, or by using the swadm command.

See Administering Case Data Normalization on page 421 for more information.

RPC_TIMEOUT

General iProcess Engine Configuration

Summary This attribute defines the timeout period (in seconds) that an RPC request waits for a response from the iProcess Engine process.

Applies To The attribute must be set for ALL processes.

Permissible Values The attribute value must be one of the following:

Value	Meaning
n	The timeout period, where *n* is any integer value.

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	25

Notes None.

See Also None.

SPO_USERMUTEX_WAITTIME

General iProcess Engine Configuration

Summary   This attribute defines the maximum amount of time (in milliseconds) that an iProcess Objects Server process should take to acquire for a mutual exclusion lock (mutex) before processing a message if multiple SSO clients use the same user name to log on TIBCO iProcess Engine. If the time for acquiring a user's mutex is over the time specified by the SPO_USERMUTEX_WAITTIME attribute, the duration will be recorded and later reported in the sw_warn log.

Before TIBCO iProcess Objects Server processes a client message, it will check whether the corresponding user ID is used. If the user ID is used, TIBCO iProcess Objects Server process will wait until the user ID is not used. When the user ID is not used, the TIBCO iProcess Objects Server process acquires the user's mutex and then begin processing the related message. Therefore it may take a long time for TIBCO iProcess Objects Server processesto wait for acuiring a user's mutex.

The attribute is used to identify possible performance problems with applications sharing the same user ID.

Applies To   The attribute can be set for SPO processes.

Permissible Values   The attribute value must be an integer in the range 100 to 20000.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	SPO	0	200

Notes   TIBCO iProcess Objects Server process needs to acquire the user's mutex before processing a message.

An error is returned if the time for acquiring a user's mutex by the iProcess Objects Server process is longer than the time you set for the process attribute. The following error message is logged in sw_warn file to record the errors:

- If this is the first error message for the shared user name:

  The user *user_name* was in-use. Please check that multiple clients are not sharing the same username *user_name* as this can have performance impact.

- If this is not the first error message for the shared user name and the time interval between the error messages is equal to or later than one hour:

```
The user user_name was in-use and has been in use at time of
acquisition number_times time(s), in that last number_time seconds.
Please check that multiple clients are not sharing the same
username (user_name) as this can have performance impact.
```

If this is not the first error message for the shared user name and the time interval between the log messages is less than an hour, no error message is logged in the sw_warn file.

where:

- *user_name* is the shared user name.

- *number_times* is the number of times errors are returned during the interval.

- *number_time* is the time interval between the returned error messages.

TIMEZONE

General iProcess Engine Configuration

Summary	This attribute defines the time zone that this iProcess Engine node will operate in.
Applies To	The attribute must be set for ALL processes.
Permissible Values	The TIMEZONE value must be a valid time zone recognized by the operating system. It should be specified as a string in the following format:

tzn[+|-]*hh*[:*mm*[:*ss*]][*dzn*]

where:

*   *tzn* is a 3-letter name that identifies the time zone, such as GMT or EST. Any meaningful name can be used.

*   [+|-]*hh*[:*mm*[:*ss*] defines the number of hours (and, optionally, minutes and seconds) that the time zone is ahead of or behind GMT. This number represents an offset i.e. the figure to be subtracted from GMT, so timezones that are:

    — *behind* GMT should be specified as a *positive* value.

    — *ahead of* GMT should be specified as a *negative* value.

*   *dzn* is a 3-letter name that identifies a daylight-saving time zone, such as BST. If *dzn* is set daylight saving is enabled and the date and time are adjusted accordingly. Any meaningful name can be used.

Examples	Any of the following strings can be used to define the TIMEZONE value for Washington D.C. (Eastern Standard Time, GMT-05:00):

```
5
EST5
EST+5
EST05:00
```

Any of the following strings can be used to define the TIMEZONE value for Sydney, Australia (Western Standard Tim, GMT+10:00):

```
-10
GMT-10
GMT-10:00:00
```

Default Value	This attribute is not defined on a newly installed iProcess Engine. If required, it must be explicitly set up using the SET_ATTRIBUTE command. By default, iProcess Engine will use the host computer's local time.

Notes This attribute should be set if an iProcess Engine node installed on a computer operating in one time zone is being accessed by TIBCO iProcess Workspace instances that are operating in different time zones, to avoid discrepancies between the server and client timestamps. See Configuring iProcess Engine Time Zone on page 15 for more information.

WARN_ERROR_LOG_SIZE

General iProcess Engine Configuration

Summary	This attribute defines the maximum size (in bytes) of the `sw_warn` and `sw_error` log files before the files stop logging error messages.
Applies To	This attribute can be set for ALL processes.
Permissible Values	An integer that is greater than 0.
Default Value	The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	2147483648 (2GB)

Notes	TIBCO iProcess Engine might continually log errors, which causes the `sw_warn` or `sw_error` log file to fill up quickly. To avoid losing error messages, you should rename the `sw_warn` or `sw_error` log file before the file reaches its maximum size and stops logging.

WINTIME_RESYNC_NOTICE

General iProcess Engine Configuration

Summary This attribute defines the notice period (in seconds) that iProcess Engine processes are given before a resynchronization takes place.

Applies To The attribute must be set for ALL processes.

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	60

Notes See WINTIME_RESYNC_PERIOD on page 179 for more information about the use of this attribute.

See Also WINTIME_RESYNC_PERIOD, WINTIME_RESYNC_TOLERANCE

WINTIME_RESYNC_PERIOD

General iProcess Engine Configuration

Summary This attribute defines the interval (in seconds) at which iProcess Engine checks to see if its timestamps are in step with Windows system time.

Applies To The attribute must be set for ALL processes.

Permissible Values An integer that is greater than or equal to 0. If this attribute is set to 0 then no checks are performed.

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	300

Notes The iProcess Engine node records audit trail timestamps to microsecond precision, and sorts the audit trail based on the timestamp.

Because the Windows system timer only returns time to millisecond accuracy, the iProcess Engine node uses two system timers to generate its audit trail timestamps - the system timer (GetSystemTime function) and a high-resolution performance counter (QueryPerformanceCounter function) which can be used to provide extra precision. However, it has been found that these timers do not keep in step with each other, and can diverge by up to several seconds over a period of days. This can result in two problems:

- iProcess Engine timestamps do not correspond to the current Windows time when they are generated.

- If BG processes are started at different times, any timestamps they generate will be out of synchronization with each other. This can result in audit trail entries appearing out of order.

To deal with these problems, you can use the WINTIME_RESYNC_* process attributes to configure how iProcess Engine synchronizes its timestamps with Windows system time.

Every `WINTIME_RESYNC_PERIOD` seconds iProcess Engine checks to see if its timestamps are in step with Windows system time. If the timestamps differ by more than `WINTIME_RESYNC_TOLERANCE` milliseconds iProcess Engine resynchronizes its timers with Windows system time. iProcess Engine processes are given `WINTIME_RESYNC_NOTICE` seconds notice before the resynchronization takes place.

You can also manually force iProcess Engine to resynchronize its timestamps with Windows system time by using the `swsvrmgr RESYNCTIME` command. See Resynchronize Timestamps with Windows Time on page 129 for more information.

See Also   WINTIME_RESYNC_NOTICE, WINTIME_RESYNC_TOLERANCE

WINTIME_RESYNC_TOLERANCE

General iProcess Engine Configuration

Summary	This attribute defines the interval (in seconds) at which iProcess Engine checks to see if its timestamps are in step with Windows system time.
Applies To	The attribute must be set for ALL processes.
Permissible Values	This value must be an integer that is greater than or equal to 20 (as Windows system time is only accurate to within 15.625ms). Lower values cannot be specified.
Default Value	The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	50

Notes	The tolerance (in milliseconds) by which the TIBCO timestamp and Windows system time can differ. If this value is exceeded, iProcess Engine resynchronizes its timers with Windows system time.
	See WINTIME_RESYNC_PERIOD on page 179 for more information about the use of this attribute.
See Also	WINTIME_RESYNC_NOTICE, WINTIME_RESYNC_PERIOD

XPC_READ_UNCOMMITTED

General iProcess Engine Configuration

Summary	This attribute applies to SQL Server only.
	Allows configuration of uncommitted reads during an XPC SELECT.
Applies To	The attribute must be set for ALL processes.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
0	Uncommitted reads cannot be used.
1	XPC is enabled to use uncommitted reads.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

Notes	None.
See Also	None.

Process Management Configuration

The following process attributes allow you to configure the behavior of the iProcess Engine Process Sentinels and server processes.

A further set of process attributes allow you to configure specific WIS and WQS behavior - see WIS and WQS Process Configuration on page 207 for more information.

Attribute	Description
BG_LEAVER_DESTINATION	Defines the destination queue to which work items are redirected when the owner of the work items has been deleted.
BG_LEAVER_SUPER_AS_DEST	Defines to the redirect location of work items for a currently deleted user.
CHECKFREQ	Defines the number of loops to process before the Process Sentinels check for `sw_error.log` files and available disk space.
DMD_PROCESS_INTERVAL	Defines the times during the day when the Deadline Manager checks the iProcess database for expired deadlines.
FORCE_DEADLINE_PRIORITY	Defines a priority value for case deadlines that override all other priorities set for case deadlines.
MAINCASE_START_AUTOCOMMIT	Defines whether or not to insert a Transaction Control step automatically as the first step of all main cases.
MAX_AGE_BEFORE_RESETPOST	Defines the time limit in seconds before the Deadline Manager will re-post unprocessed deadline messages.
MINFREEKB	Defines the amount of disk space (in Kilobytes) required for the `BG` process to run.
MN_UNLOCKER	Defines who has permission to unlock the work items.
PM_AUTO_BOOT	Defines whether or not the Process Sentinels automatically start the server processes after the Process Sentinels have started.

Attribute	Description
PROCESS_AUTO_DUMPLOG	Defines whether or not, if a process fails, the Process Sentinels automatically write to disk the contents of that process' debug shared memory segment.
PROCESS_AUTO_RESTARTS	Defines whether or not a server process will automatically restart after a failure.
PROCESS_MAX_RESTARTS	Defines the maximum number of times the Process Sentinels will attempt to restart a failed process.
PROCESS_MIN_RESTART_PERIOD	Defines a period of time (in seconds) during which Process Sentinels suspends or restarts a failed process.
PROCESS_SLEEP	Defines the amount of time the Process Sentinels will sleep for.
SUBCASE_START_AUTOCOMMIT	Defines whether or not to insert a Transaction Control step automatically as the first step of all sub-cases.
UNPROCESSED_DL_POST_LIMIT	Sets a limit on the number of unprocessed deadline messages that are posted by the Deadline Manager.
USE_PRIORITY_DEADLINES	Defines whether to set the priority for the case deadlines by using the value of the SW_CP_VALUE field if the value of the FORCE_DEADLINE_PRIORITY attribute is set to the default value, 0.

BG_LEAVER_DESTINATION

Process Management Configuration

Summary	This attribute specifies the destination queue to which work items are redirected when the owner of the work items has been deleted.
Applies To	The attribute must be set for the BG processes.
Permissible Values	This attribute value must be a valid name of the user-specified destination queue or the admin user's queue.
Default Value	The attribute is assigned the following default value when an iProcess Engine node is installed.

Machine ID	Process	Instance	Value
1	BG	0	1

Notes If a user is deleted, the work items that are currently in the user's queue can be redirected to a user-specified destination or to the admin user's queue. This attribute is used with the BG_LEAVER_SUPER_AS_DEST attribute. Details of how to use these attributes are as follows:

- To redirect the work items to a user-specified queue when a user has just been deleted, follow these steps:

  a. Set the value of the BG_LEAVER_SUPER_AS_DEST attribute to 0.

  b. Set the value of the BG_LEAVER_DESTINATION attribute to a user-specified destination queue.

- To redirect the work items to the admin user's queue when a user is deleted, set the value of the BG_LEAVER_SUPER_AS_DEST attribute to 0.

The deleted user's work items are redirected to the admin user's queue by default, if the BG_LEAVER_DESTINATION attribute is not set. If the BG_LEAVER_DESTINATION attribute has been set, use the swadm delete_attribute command to delete this setting. For more information about this command, see Delete a Process Attribute on page 145.

- To redirect the work items to the first supervisor's queue when a user is deleted, set the value of the BG_LEAVER_SUPER_AS_DEST attribute to 1.

  For more information about configuring the BG_LEAVER_SUPER_AS_DEST attribute, see BG_LEAVER_SUPER_AS_DEST on page 187.

If the name of the destination queue, which is specified in setting the BG_LEAVER_DESTINATION attribute, is currently removed, the work item will be redirected to the admin user's queue.

When a user is deleted, an entry is inserted into the leavers table, and a corresponding entry is also inserted into the redir_defn table. These records will be removed automatically when the following conditions are met:

- All work items of the deleted user are redirected.

- The period specified by the WQS_LEAVER_PERIOD attribute expires.

  The WQS_LEAVER_PERIOD attribute is set by minutes, and its default value is 60.

- Perform a Movesysinfo request or iProcess Engine shuts down.

> When iProcess Engine starts up, the timestamp column for all records in the leavers table are updated to the current time.

If the records in the leavers table and the redir_defin table are not removed, they will be removed when the deleted user is added into iProcess Engine again.

See Also   BG_LEAVER_SUPER_AS_DEST

BG_LEAVER_SUPER_AS_DEST

Process Management Configuration

Summary   This attribute defines the redirect location of work items for a currently deleted user.

Applies To   The attribute must be set for the BG processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	The work items, which arrive at a currently deleted user, are redirected to the specified destination queue or to the admin user's queue.
1	The work items, which arrive at a currently deleted user, are redirected to the first supervisor's queue.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
1	BG	0	1

Notes   When a user is deleted, the work items, which are currently in the deleted user's queue can be redirected to the first supervisor's queue, the user-specified destination, and the admin user's queue. Details on how to redirect the work items are as follows:

- To redirect the work items to the first supervisor's queue, set the value of the BG_LEAVER_SUPER_AS_DEST attribute to 1.

- To redirect the work items to the specified destination queue, use the BG_LEAVER_DESTINATION attribute.

    For more information about how to use this attribute, see BG_LEAVER_DESTINATION on page 185.

See Also   BG_LEAVER_DESTINATION

CHECKFREQ

Process Management Configuration

Summary   This attribute defines the number of processing loops that the Process Sentinels will cycle through before checking:

- for sw_error files, which are located in the *SWDIR*\logs directory. See iProcess Engine Log Files on page 439 for more information.

- that the system has sufficient available disk space.

Applies To   The attribute must be set for ALL processes.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	50

Notes   The actual time between these checks will therefore be CHECKFREQ * PROCESS_SLEEP seconds.

See Also   PROCESS_SLEEP

DMD_PROCESS_INTERVAL

Process Management Configuration

Summary This attribute defines the times during the day when the Deadline Manager checks the iProcess database for expired deadlines.

Applies To The attribute can be set for the DLMGR process.

Permissible Values The attribute value must be an integer in the range -1439 to +720, representing a processing interval, in minutes, calculated relative to midnight local time on the server where the DLMGR process is running.

If this value is:

- zero or less than zero, the processing interval is interpreted as an absolute interval. An absolute interval is used to process deadlines once per day at a set time. A value of zero means exactly midnight.

- greater than zero, the processing interval is interpreted as a repeating interval. A repeating interval is used to process deadlines at regular intervals and at set times throughout the day, on each day. If an interval crosses the midnight boundary, the calculation is reset to start from midnight again (so that deadlines are processed at the same times each day).

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	1

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   The following table shows some example settings and the intervals they
represent.

Value	Type	Deadlines will be processed at
-720	Absolute	12 noon every day.
60	Repeating	1am, 2am, 3am...and every hour thereafter.
0	Absolute	Midnight every day.
360	Repeating	6am, 12pm, 6pm, 12am every day
300	Repeating	5am, 10am, 3pm, 8pm every day. **Note**: Processing on the second day does NOT start at 1am (8pm + 5 hrs)

See Also   MAX_AGE_BEFORE_RESETPOST, UNPROCESSED_DL_POST_LIMIT

FORCE_DEADLINE_PRIORITY

Process Management Configuration

Summary This attribute defines a priority value for case deadlines that override all other priorities set for case deadlines.

The FORCE_DEADLINE_PRIORITY attribute only applies to the deadline message being sent, therefore it will effect the overall priority that deadlines are processing in. It will not have any subsequent effect on the priority of the case that the deadline is expiring.

Applies To This attribute can be set for the DLMGR processes.

Permissible Values The attribute value must be an integer in the range 0 to 999,

where:

- 0 means the priority for the case deadlines is not set.

- 1 means set the case deadlines to the highest priority.

- 999 means set the case deadlines to the lowest priority.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	DLMGR	0	0

See Also USE_PRIORITY_DEADLINES

MAINCASE_START_AUTOCOMMIT

Process Management Configuration

Summary   This attribute defines whether or not to insert a Transaction Control step automatically as the first step of all main cases.

Applies To   This attribute must be set for the BG processes.

Permissible Values   The attribute value must be one of the following.

Value	Meaning
0	The steps are processed based on the procedure defined, and no autocommits are inserted in main cases.
1	A Transaction Control step is added automatically before processing the first step of the procedure. This Transaction Control step is defined as the Commit and Concede type.
2	A Transaction Control step is added automatically before processing the first step of the procedure. This Transaction Control step is defined as the Commit and Continue type.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	BG	0	0

Notes   If the value of the MAINCASE_START_AUTOCOMMIT attribute is set to 1 or 2, all main cases run the Transaction Control step as the first step of the procedure before processing the predefined first step of the procedure. The meaning of each type of the Transaction Control step are listed as follows:

- **Commit and Continue** allows you to commit the current transaction and start a new transaction for subsequent steps using the same Background process. The advantage of this option is that it is faster, since it uses the previous process to start the new transaction.

- **Commit and Concede** allows you to commit the current transaction and start a new transaction for subsequent steps using a different Background process. It uses a different Background process to start the second transaction. The advantage of this option is that it enables load balancing.

For more information about Transaction Control step, see *TIBCO iProcess Modeler Integration Techniques*.

See Also   SUBCASE_START_AUTOCOMMIT

MAX_AGE_BEFORE_RESETPOST

Process Management Configuration

Summary This attribute defines the time limit in seconds before the Deadline Manager will re-post unprocessed deadline messages. This specifies the time period before the Deadline Manager resets its internal marker of the last deadline it has processed to 0 (beginning of time).

Applies To The attribute can be set for the DLMGR process.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	DLMGR	0	3600

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

See Also DMD_PROCESS_INTERVAL, UNPROCESSED_DL_POST_LIMIT

MINFREEKB

Process Management Configuration

Summary   This attribute defines the amount of disk space required for a BG process to run.

Applies To   The attribute can be set for the PROCMGR process.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	10000

MN_UNLOCKER

Process Management Configuration

Summary	This attribute defines who has permission to unlock the work items.
Applies To	This attribute can be set for the SPO process.
Permissible Values	This attribute value must be a valid iProcess user name for the MENUNAME attribute.
Default Value	This attribute is not defined automatically when you install or upgrade iProcess Engine. To use this attribute, you must specifically assign a value to it.
Notes	You can check the valid iProcess user names in the following line of the STAFF.MES file, which is located in the *SWDIR*/etc/english.lng directory:

```
0444::ADMIN\USER\PRODEF\MANAGER\TEAMLEADER\SUPERVISOR\MS1\MS2\MS3
```

You can add an iProcess user name in this line. After adding the user, you have to set the attribute by using swadmin in *SWDIR*/util. For example, ./swadm set_attribute 1 SPO 1 MN_UNLOCKER MANAGER:TEAMLEADER:SUPERVISOR.

For more information about the MENUNAME attribute, see "Setting Pre-defined Attributes" in *TIBCO iProcess Workspace (Windows) Manager's Guide*.

For more information about unlocking work items, see *TIBCO iProcess Workspace (Windows) User's Guide*.

PM_AUTO_BOOT

Process Management Configuration

Summary   This attribute defines whether or not the Process Sentinels automatically start the server processes after the Process Sentinels have started.

Applies To   The attribute can be set for the PROCMGR process.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	The Process Sentinels will not automatically start the server processes.
1	The Process Sentinels will automatically start the server processes.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	PROCMGR	0	1

This value is the default for a UNIX system.

On a Windows system, the value is set by the user when they install or upgrade iProcess Engine.

PROCESS_AUTO_DUMPLOG

Process Management Configuration

⚠️ You should only use this attribute when explicitly requested to do so by TIBCO Support.

Summary

This attribute defines whether or not, if a process fails, the Process Sentinels automatically write to disk the contents of that process' debug shared memory segment.

Applies To

The attribute can be set for any process.

Permissible Values

The attribute must be assigned one of the following values.

Value	Meaning
0	No debug is written to disk if the process fails.
1	All debug in the process' debug shared memory segment is written to disk if the process fails.

Default Value

This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	1

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

PROCESS_AUTO_RESTARTS

Process Management Configuration

Summary	This attribute defines whether or not a server process will automatically restart after a failure.
Applies To	The attribute can be set for any process.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
0	The process will not automatically restart after a failure.
1	The process will automatically restart after a failure.

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	1

See Also PROCESS_MAX_RESTARTS, PROCESS_MIN_RESTART_PERIOD

PROCESS_MAX_RESTARTS

Process Management Configuration

Summary This attribute defines the maximum number of times the Process Sentinels will attempt to restart a failed process.

Applies To The attribute can be set for any process.

Permissible Values The attribute value must be one of the following:

Value	Meaning
0	The Process Sentinels will keep attempting to restart the failed process.
n	The Process Sentinels will attempt to restart the failed process n times (where n is a positive integer).

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	5

See Also PROCESS_AUTO_RESTARTS, PROCESS_MIN_RESTART_PERIOD

PROCESS_MIN_RESTART_PERIOD

Process Management Configuration

Summary	This attribute defines a period of time (in seconds) during which Process Sentinels suspends or restarts a failed process.
	If the time that the process crashed between this time and the last time, is less than the time interval you defined in this process, the crashed process will be suspended, otherwise the crashed process will be restarted.
Applies To	This attribute can be set for any process.
Default Value	This attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	120

See Also	PROCESS_AUTO_RESTARTS, PROCESS_MAX_RESTARTS

PROCESS_SLEEP

Process Management Configuration

Summary   This attribute defines the amount of time (in seconds) the Process Sentinels will sleep for.

Applies To   The attribute can be set for the PROCMGR process.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	5

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   The Process Sentinels go into a sleep/process loop once they have done their initial job of starting all processes. This means that they will sleep for a configurable amount of time in between actively monitoring processes.

See Also   CHECKFREQ

SUBCASE_START_AUTOCOMMIT

Process Management Configuration

Summary	This attribute defines whether or not to insert a Transaction Control step automatically as the first step of all sub-cases.
Applies To	This attribute must be set for the BG processes.
Permissible Values	The attribute value must be one of the following.

Value	Meaning
0	The steps are processed based on the procedure defined, and no autocommits are inserted in sub-cases.
1	A Transaction Control step is added automatically before processing the first step of the procedure. This Transaction Control step is defined as the Commit and Concede type.
2	A Transaction Control step is added automatically before processing the first step of the procedure. This Transaction Control step is defined as the Commit and Continue type.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	BG	0	0

Notes   If the value of the SUBCASE_START_AUTOCOMMIT attribute is set to 1 or 2, all sub-cases run the Transaction Control step as the first step of the procedure before processing the predefined first step of the procedure. The meanings of each type of Transaction Control step are listed as follows:

- **Commit and Continue** allows you to commit the current transaction and start a new transaction for subsequent steps using the same Background process. The advantage of this option is that it is faster, since it uses the previous process to start the new transaction.

- **Commit and Concede** allows you to commit the current transaction and start a new transaction for subsequent steps using a different Background process. It uses a different Background process to start the second transaction. The advantage of this option is that it enables load balancing.

For more information about Transaction Control step, see *TIBCO iProcess Modeler Integration Techniques*.

See Also   MAINCASE_START_AUTOCOMMIT

UNPROCESSED_DL_POST_LIMIT

Process Management Configuration

Summary   This attribute is used to set a limit on the number of unprocessed deadline messages that are posted by the Deadline Manager.

Applies To   This attribute can be set for the DLMGR process.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	10000

This attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   When the UNPROCESSED_DL_POST_LIMIT value is exceeded, the Deadline Manager stops sending deadline messages until the number of deadline messages in the Mbox queue drops below the value that is currently set for this process attribute.

If both the UNPROCESSED_DL_POST_LIMIT and MAX_AGE_BEFORE_RESETPOST are reached at the same time, then all the deadline messages are resent.

See Also   MAX_AGE_BEFORE_RESETPOST

USE_PRIORITY_DEADLINES

Process Management Configuration

Summary   This attribute defines whether to set the priority for the case deadlines by using the value of the SW_CP_VALUE field if the value of the FORCE_DEADLINE_PRIORITY attribute is set to the default value, 0.

Applies To   This attribute can be set for the DLMGR process.

Permissible Values   The attribute value must be one of the following.

Value	Meaning
0	Disable the priority setting for the case deadlines if the value of the FORCE_DEADLINE_PRIORITY attribute is set to the default value, 0.
1	Set the value of the SW_CP_VALUE field as the priority value for the case deadlines if the value of the FORCE_DEADLINE_PRIORITY attribute is set to the default value, 0. If the value of the SW_CP_VALUE field is not set, the value of the WQS_DEFAULTPRIORITY parameter will be set as the priority value for the case deadlines.
	For more information about the SW_CP_VALUE field, see *TIBCO iProcess Modeler Advanced Design*. For more information about the WQS_DEFAULTPRIORITY parameter, see WQS_DEFAULTPRIORITY on page 49.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	DLMGR	0	0

See Also   FORCE_DEADLINE_PRIORITY

WIS and WQS Process Configuration

The following process attributes allow you to configure the behavior of the WQS and WIS processes.

Attribute	Description
AUDIT_OPENKEEP	Controls whether the Open Work Item and Keep Work Item audit trail entries are enabled.
CCOUNT_CACHE_REFRESH	Enables you to define the refresh period for updating the cached list of cases currently on the system.
IGNORE_PACK_CHANGED	Defines whether users may keep or release work items even if pack data has changed.
RESTART_WIS_CACHE_THRESHOLD	Defines the number of items that must exist in a work queue for it to be cached when the WIS process restarts or quick starts.
RETRY_OFF_FOR_RAC	Determines whether to enable the retry function that retrieves work items from the staffo database table for the configured Oracle RAC.
RPC_SVR_CONTROL	Allows the batching of RPC calls to reduce the overhead in processing RPC calls individually.
RPC_SVR_NUM_THREADS	Defines the maximum number of threads that the WIS and WQS processes can use to process RPC requests from client applications.
SHMKEY_ID	Defines the UNIX shared memory key that is allocated (using the ftok system call) when the WQS process is started.
WIS_CACHE_POOL_SIZE	Defines the size (in threads) of the pool of threads that is used to perform caching of work queues.
WIS_CACHE_THRESHOLD	Defines the number of items that must exist in a work queue for it to be cached when the WIS process starts normally.

Attribute	Description
WIS_CACHE_WAIT_TIME	Defines the maximum amount of time (in seconds) that an RPC processing thread in the WIS process waits for a work queue to be cached.
WIS_CDQP_DATA_RECACHE_BATCH	Defines the number of work items that the CDQP update thread will update in a single operation when updating CDQP field values for a WIS process' queues.
WIS_CHECK_STAFFO_RETRY_COUNT	Defines the number of times the WIS process attempts to retrieve the unfound work items from the staffo database table because a delay occurred between two Oracle RAC instances. **Note**: This attribute is only used on configured Oracle RAC.
WIS_FILTER_THREAD_BOUNDARIES	Defines the count boundary at which a work queue will be split into multiple blocks of work for filtering purposes, based on the number of work items in the queue
WIS_FILTER_THREAD_POOL_SIZE	Defines the number of threads in the queue filtering thread pool, used to process additional blocks of filtering work
WIS_INDEX_REFRESH	Defines the interval (in seconds) after which an index on a queue will be refreshed by a WIS process.
WIS_LOCK_POOL_SIZES	Defines the number of locks in the internal lock pool used by the WIS process
WIS_NEW_ITEM_BATCH_SIZE	Defines the number of new item requests to be batched together.
WIS_QCHANGE_EXTENDED_CHECK	Defines whether or not a change in the lock status of a work item is counted as a change to the work item.

Attribute	Description
WIS_SESSION_TIMEOUT	Defines the timeout period (in seconds) after which a WIS process will automatically shut down, starting from the time at which it was last accessed (by a TIBCO iProcess Workspace, SAL application or TIBCO iProcess Objects Server).
WIS_SESSION_TIMEOUT_SHUTDOWN	Defines the timeout period (in seconds) after which a WIS process will automatically shut down, starting from the time at which iProcess Engine was shut down.
WIS_UNCACHE_PERIOD	Defines the time period (in minutes) after which the cached work queue that waits to be handled by the WIS process is cleared from the cache, starting from the time at which the client who last accessed this work queue logs off. **Note**: This attribute only applies to the work queues that contain a number of work items less than the number you defined in the WIS_CACHE_THESHOLD attribute.
WIS_UPDATE_LENGTH	Defines the maximum amount of time (in seconds) that the queue update thread in the WIS process performs updates for before going back to sleep.
WIS_UPDATE_PERIOD	Defines how often the queue update thread in the WIS process wakes up and updates the queues handled by the WIS process.
WIS_USE_PRIORITY_ESCALATION	Allows the use of priority escalation in the WIS process to be disabled.
WQS_GATHER_RPC_STATS	Allows the gathering of RPC call stats within the WQS process to be configurable.
WQS_NUM_SEARCH_SLOTS	Defines the maximum number of slots available in the SWRPCMTS multi-threaded RPC server shared library for threads to perform queue searching.

Attribute	Description
WQS_PERSIST_SHMEM	Defines how often (in seconds) the contents of the WQS/WIS shared memory are written to the `wqs_index` database table.
WQS_WIS_USER_COUNT	Defines the number of `WIS` processes that should be dedicated to handling user queues and group queues respectively.

AUDIT_OPENKEEP

WIS and WQS Process Configuration

Summary	This attribute determines if opening or keeping a work item generates an audit trail entry. The default behavior is not to produce audit trail entries when a work item is opened or kept. Enabling this option may cause opening and keeping activities to be marginally slower, and could significantly increase the size of an audit trail.
Applies to	This attribute should be set for ALL processes.
Permissible Values	The attribute must be assigned one of the following values.

Value	Meaning
0	Open and Keep audit messages are not posted.
1	Open and Keep audit messages are posted.

Default Value The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

Notes If this attribute is set to 1, the WIS posts an audit message to the BG process whenever an Open or a Keep operation is performed on a work item. See messages 059 and 060 in Understanding Audit Trails on page 449.

CCOUNT_CACHE_REFRESH

WIS and WQS Process Configuration

Summary   This attribute enables you to define the refresh period for updating the cached list of cases currently on the system. The Background process can retrieve a case count list from the database by looking at which procedures have cases running. The case count list is used by utilities such as Audit Trail and Case Administration. These utilities refer to the cached list so that they do not have to retrieve a list of cases from the database every time, thereby improving performance.

Applies to   This attribute should be set for ALL processes.

Permissible Values   The attribute must be assigned one of the following values.

Value	Meaning
n	The number of seconds between refreshes of the cached list.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	RPC_POOL	0	30

Notes   If the value of the attribute is set to 0, the cached list of cases currently on the system will not be refreshed.

AUDIT_OPENKEEP

WIS and WQS Process Configuration

Summary   This attribute determines if opening or keeping a work item generates an audit trail entry. The default behavior is not to produce audit trail entries when a work item is opened or kept. Enabling this option may cause opening and keeping activities to be marginally slower, and could significantly increase the size of an audit trail.

Applies to   This attribute should be set for ALL processes.

Permissible Values   The attribute must be assigned one of the following values.

Value	Meaning
0	Open and Keep audit messages are not posted.
1	Open and Keep audit messages are posted.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

Notes   If this attribute is set to 1, the WIS posts an audit message to the BG process whenever an Open or a Keep operation is performed on a work item. See messages 059 and 060 in Understanding Audit Trails on page 449

IGNORE_PACK_CHANGED

WIS and WQS Process Configuration

Summary   This attribute defines whether users can Keep or Release work items even if the item's pack data has changed since they opened it.

Applies to   This attribute can be set for the WIS process (only).

Permissible   The attribute must be assigned one of the following values.
Values

Value	Meaning
0	Pack data changes lock work items. A user cannot Keep or Release a work item that has had its pack data updated since the user opened it.
1	Pack data changes are ignored. A user may Keep or Release a work item that has had its pack data updated since the user opened it. If any of the user's changes to the work item conflict with the changed pack data, the user's changes overwrite them.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	0

This attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   In earlier iProcess Engine versions, if pack data is updated for a work item while a user has that work item open (for example, via a swutil EVENT -p command, which is under the *SWDIR*\bin directory), the WIS process locks the work item and does not allow the user to Keep or Release it. The following error is displayed to the user when they try to Keep or Release the work item:

```
Error case data updated elsewhere since item opened. Please open
item and edit it again.
```

Setting IGNORE_PACK_CHANGED to 1 allows users to Keep or Release work items even if the item's pack data has changed since they opened it.

RESTART_WIS_CACHE_THRESHOLD

WIS and WQS Process Configuration

Summary This attribute defines the number of items that must exist in a work queue for it to be cached when the WIS process restarts or quick starts.

Applies To This attribute can only be used when a WIS process restarts or quick starts.

Permissible Values This attribute must be an integer in the range 0 to 500000.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	1000

This attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes A queue is usually cached in the following two situations:

- When the WIS process first handles the queue (either on startup or after a MoveSysInfo operation).

- When the queue is first accessed by a client application.

This attribute is only used when the WIS process restarts or quick starts in conjunction with the WISCACHE queue attribute to control whether a queue is cached:

- If the value of WISCACHE is set to YES, the WIS process caches the queue (irrespective of how many work items there are in the queue).

- If WISCACHE is not created or set, the WIS process caches the queue when the number of work items in the queue equals or exceeds the value of the RESTART_WIS_CACHE_THRESHOLD attribute.

See Configuring When WIS Processes Cache Their Queues on page 416 for more information.

See Also WIS_CACHE_THRESHOLD, WIS_CACHE_POOL_SIZE, WIS_CACHE_WAIT_TIME

RETRY_OFF_FOR_RAC

WIS and WQS Process Configuration

This attribute is only applied on configured Oracle RAC.

Summary   This attribute determines whether to enable the retry function that retrieves work items from the `staffo` database table for configured Oracle RAC.

Applies To   This attribute can be set for `WIS` and `WISMBD` process.

Permissible Values   This attribute must be one of the following:

Value	Meaning
0	Switches on the retry function.
1	Switches off the retry function.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

This attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   • With configured Oracle RAC, a commit transaction is divided into two actions: dequeue the REQUEST instruction by `WISMBD` and insert the new work items into the `staffo` table. These two actions are on two Oracle RAC instances. If the dequeue action completes before the committed the new work items to the table is replicated to the second node, the `WIS` process may not be able to read the work items from the `staffo` table.

If the retry function is enabled, the `WISMBD` will retry to send message. Otherwise, the `WIS` will treat it as a withdrawn message and ignores the request. The message similar to the following will be displayed in the `WIS` log:

```
1631-WARNING: <no matching entry found in the STAFFO table when
the REQUEST message is processed>
```

In this case, you can set this attribute to 1 to switch on the retry function.

- By default, the retry function is enabled. If you need to create many work items that are quickly withdrawn either by the withdraw action or closing case, the work items cannot be retrieved from the `staffo` database table. To expedite the WIS Mbox processing, you can configure this attribute to switch off the retry function.

See Also   WIS_CHECK_STAFFO_RETRY_COUNT

RPC_SVR_CONTROL

WIS and WQS Process Configuration

Summary	This attribute allows the batching of RPC calls to reduce the overhead of processing RPC calls individually.
Applies To	This attribute should be set for ALL processes.
Permissible Values	This attribute must be in the form B[,batch size] where batch size specifies either 0 to turn off batching or the number of items to be batched. The default value is 20.

For example,

  B   = Set batch size to default (20)

  B,0   = Turn off batching of RPC calls

  B,10 = Set batch size to 10

Default Value	The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	B

Notes	Using this attribute reduces the overhead in processing RPC calls individually. Any slight overhead in waiting for 20 (or the number specified) to be batched should not be noticeable.
See Also	N/A.

RPC_SVR_NUM_THREADS

WIS and WQS Process Configuration

Summary	This attribute defines the maximum number of threads that the WIS and WQS processes can use to process RPC requests from client applications.
Applies To	This attribute should be set for ALL processes.
Permissible Values	This attribute must be an integer in the range 1 to 100 (but see the Notes below).
Default Value	The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	ALL	0	5

Notes   To process RPC requests, both the WIS and WQS processes access a pool of "worker" threads that is provided by a multi-threaded RPC server shared library (SWRPCMTS). This attribute defines the number of threads that are available in the SWRPCMTS library to process RPC requests.

⚠️ The maximum RPC_SVR_NUM_THREADS value is also limited by the value of the WQS_NUM_SEARCH_SLOTS process attribute.

If you want to increase the RPC_SVR_NUM_THREADS value beyond the WQS_NUM_SEARCH_SLOTS value, you must stop iProcess Engine, change the RPC_SVR_NUM_THREADS value and then restart iProcess Engine.

If you try to increase RPC_SVR_NUM_THREADS beyond WQS_NUM_SEARCH_SLOTS without stopping the iProcess Engine, the RPC_SVR_NUM_THREADS value will instead be set to the WQS_NUM_SEARCH_SLOTS value.

You can adjust the value of this process attribute to optimize the WQS and WIS process' response times when processing RPC requests against available CPU capacity. Increasing the number of threads will improve the throughput of client RPC requests, but at the cost of increased CPU usage.

See Also   WIS_FILTER_THREAD_BOUNDARIES, WIS_FILTER_THREAD_POOL_SIZE, WQS_NUM_SEARCH_SLOTS

SHMKEY_ID

WIS and WQS Process Configuration

⚠️   TIBCO recommends that you do not change the value of this attribute unless you are instructed to do so by TIBCO Support, or you are fully familiar with the use of UNIX shared memory and the operation of the ftok system call.

Summary   This attribute defines the UNIX shared memory key that is allocated (using the ftok system call) when the WQS process is started

Applies To   This attribute must be set for ALL processes.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	"x"

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   The default value should work correctly in most situations. However, it is possible for a shared memory conflict to occur - for example, if iProcess Engine is restarted, another application may allocate to itself the shared memory key that iProcess expects to use when it restarts. If this happens, the WQS process will fail to start, and the following error message is written to the sw_error file:

```
WQS initialise failed, connected to shared memory for nodename
```

where *nodename* is either a valid nodename, or blank.

If such a shared memory conflict does occur you can change the SHMKEY_ID value to resolve it.

WIS_CACHE_POOL_SIZE

WIS and WQS Process Configuration

Summary	This attribute defines the size (in threads) of the pool of threads that is used to perform caching of work queues.
Applies To	This attribute can be set for a WIS process (only).
Permissible Values	This attribute must be an integer in the range 1 to 100.
Default Value	The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	4

Notes	You may want to increase the WIS_CACHE_POOL_SIZE value if there are a large number of work queues that need caching at one time. When all the work queues have been cached you may want to reduce the value again, as the threads in this pool will not be used until a new queue is first handled by a WIS process.
	See Configuring When WIS Processes Cache Their Queues on page 416 for more information.
See Also	WIS_CACHE_THRESHOLD, WIS_CACHE_WAIT_TIME

WIS_CACHE_THRESHOLD

WIS and WQS Process Configuration

Summary   This attribute defines the number of items that must exist in a work queue for it to be cached when the WIS process starts normally.

Applies To   This attribute can only be used when a WIS process starts normally.

Permissible Values   This attribute must be an integer in the range 0 to 500000.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	1000

This attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   A queue is usually cached in the following two situations:

- When the WIS process first handles the queue (either on startup or after a MoveSysInfo operation).

- When the queue is first accessed by a client application.

This attribute is only used when the WIS process starts normally in conjunction with the WISCACHE queue attribute to control whether a queue is cached:

- If the value of WISCACHE is set to YES, the WIS process caches the queue (irrespective of how many work items there are in the queue).

- If WISCACHE is not created or set, the WIS process caches the queue if the number of work items in the queue equals or exceeds the value of the WIS_CACHE_ITEM attribute.

See Configuring When WIS Processes Cache Their Queues on page 416 for more information.

See Also   RESTART_WIS_CACHE_THRESHOLD, WIS_CACHE_POOL_SIZE, WIS_CACHE_WAIT_TIME, WQS_PERSIST_SHMEM

WIS_CACHE_WAIT_TIME

WIS and WQS Process Configuration

Summary   This attribute defines the maximum amount of time (in seconds) that an RPC processing thread in the `WIS` process waits for a work queue to be cached.

Applies To   This attribute can be set for a `WIS` process (only).

Permissible Values   This attribute must be an integer in the range 0 to unlimited.

⚠️   This value must be set to a value less than:

- The iProcess Workspace RPC Timeout period (the default is 25 seconds). See *TIBCO iProcess Workspace Managers's Guide* for information.

- The iProcess Objects SAL RPC Timeout (the default is 25 seconds). See *TIBCO iProcess Objects Programmer's Guide* for information.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	5

Notes   When a client application makes an RPC call to a work queue that has not already been cached, the `WIS` process immediately begins caching it. If the `WIS_CACHE_WAIT_TIME` value is reached and the work queue has still not been cached, the `WIS` process returns an `ER_CACHING` error to the client application.

See Configuring When WIS Processes Cache Their Queues on page 416 for more information.

See Also   WIS_CACHE_POOL_SIZE, WIS_CACHE_THRESHOLD

WIS_CDQP_DATA_RECACHE_BATCH

WIS and WQS Process Configuration

Summary   This attribute defines the number of work items that the CDQP update thread will update in a single operation when updating CDQP field values for a WIS process' queues.

Applies To   This attribute should be set for a WIS process (only).

Permissible Values   This attribute must be an integer in the range 1000 to 500000.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	5000

Notes   The WIS process' CDQP update thread is used to update CDQP field values for work items in its queues following a swutil QINFO PUBLISH command, which is under the *SWDIR*\bin directory. The CDQP update thread updates each work item in each queue handled by the WIS process, updating WIS_CDQP_DATA_RECACHE_BATCH items at the same time.

The CDQP update thread obtains the updated CDQP field values from the pack_data database table, which prevents other processes from updating or deleting any rows in the table that the CDQP update thread is accessing.

If you find that performance is impacted after the swutil QINFO PUBLISH command, you should reduce the WIS_CDQP_DATA_RECACHE_BATCH value.

See for more information.

WIS_CHECK_STAFFO_RETRY_COUNT

WIS and WQS Process Configuration

This attribute is only applied on configured Oracle RAC.

Summary   This attribute defines the number of times the `WIS` process attempts to retrieve the unfound work items from the `staffo` database table because a delay occurred between two Oracle RAC instances.

Applies To   This attribute can be set for the `WIS` process.

Permissible Values   This attribute must be an integer.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	3

The default value means the `WIS` process will retry 3 times if it failed to find the work items from the `staffo` database table.

Notes   With configured Oracle RAC, a commit transaction is divided into two actions: dequeue the REQUEST instruction by `WISMBD` and insert the new work items into the `staffo` table. These two actions are on two Oracle RAC instances. If the dequeue action completes before the committed the new work items to the table is replicated to the second node, the `WIS` process may not be able to read the work items from the `staffo` table. To find the new records in the table, you can configure this attribute to define the maximum number of retry times to retrieve the newly added work items.

See Also   RETRY_OFF_FOR_RAC.

WIS_FILTER_THREAD_BOUNDARIES

WIS and WQS Process Configuration

Summary

This attribute defines the count boundary at which a work queue will be split into multiple blocks of work for filtering purposes, based on the number of work items in the queue.

Applies To

This attribute can be set for the WIS process (only).

Permissible Values

This attribute must be a string in the following format:

"*Threshold1*[:*Threshold2*[:*Threshold3*[:*Threshold4*]]]"

where the four *Threshold* parameters are numeric values indicating the number of work items in a work queue at which an additional block of filtering work will be created. Each subsequent value, if used, must be greater than the preceding value.

Default Value

The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	100,000

Notes

By default, the WIS process uses the thread that is processing an RPC request to perform any work queue filtering required by that RPC request. When the number of items in a work queue reaches one of the threshold values defined in this attribute, the queue is split into equal blocks of filtering work. The first block is still handled by the RPC processing thread. Subsequent blocks are handled by threads from the queue filtering thread pool (the number of which is defined by the WIS_FILTER_THREAD_POOL_SIZE attribute).

Modifying this attribute can therefore reduce the time taken by the WIS process to filter work queues, particularly when queues are large or use complex filter criteria involving expressions or CDQPs.

See Configuring How Work Queues are Filtered on page 414 for more information.

Examples

The following example means that the queue will be split into two blocks of work for filtering purposes when the number of work items in the queue reaches 100000. The queue is split into two equal blocks of 50000 work items. The first block is handled by the original RPC processing thread and the second is handled by one of the queue filtering threads.

```
100000
```

The following example means that the queue will be split into two filtering blocks (each of 50000 work items) when the number of work items in the queue reaches 100000, and into three blocks (each of 60000 work items) when the number of items reaches 180000. The first block is handled by the original RPC processing thread. The second and third blocks are handled by the queue filtering threads.

```
100000:180000
```

See Also   RPC_SVR_CONTROL, WIS_FILTER_THREAD_POOL_SIZE

WIS_FILTER_THREAD_POOL_SIZE

WIS and WQS Process Configuration

Summary   This attribute defines the number of threads in the queue filtering thread pool, used to process additional blocks of filtering work.

Applies To   This attribute can be set for the WIS process (only).

Permissible Values   This attribute must be an integer that is greater than or equal to 1.

Default Value   The attribute is assigned the following default value when the iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	8

Notes   By default, the WIS process uses the thread that is processing an RPC request to perform any work queue filtering required by that RPC request. When the number of items in a work queue reaches one of the threshold values defined in the WIS_FILTER_THREAD_BOUNDARIES attribute, the queue is split into equal blocks of filtering work. The first block is still handled by the RPC processing thread. Subsequent blocks are handled by threads from the queue filtering thread pool (the number of which is defined by this attribute).

Modifying this attribute can therefore reduce the time taken by the WIS process to filter work queues, particularly when queues are large or use complex filter criteria involving expressions or CDQPs.

See Configuring How Work Queues are Filtered on page 414 for more information.

See Also   RPC_SVR_CONTROL, WIS_FILTER_THREAD_BOUNDARIES

WIS_INDEX_REFRESH

WIS and WQS Process Configuration

Summary This attribute value defines the interval (in seconds) after which an index on a work queue will be refreshed by a `WIS` process. You can set this attribute to fine tune the memory footprint of a `WIS` process.

Applies To The attribute can be set for a `WIS` process.

Permissible Values The attribute value must be an integer, with a minimum value of 10.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	300

The attribute only appears in the output of the `SHOW_ALL_ATTRIBUTES` command if you have explicitly assigned a value to it using the `SET_ATTRIBUTE` command. The default setting is not displayed

Notes When a user or process accesses a work queue, the `WIS` process creates an index in memory for subsequent use with that "view" of the queue. The `WIS` process holds a copy of all work item data for the queue in memory, referenced by the index, until the data is no longer needed. Refreshing the index clears out any information that is no longer needed for that "view", thus reducing the memory footprint of the `WIS` process.

If users or processes have indexes onto a busy queue and these indexes are not refreshed, the `WIS` memory footprint grows (because old records are not released and new memory is required for new items entering the queue). For example, if a user leaves a TIBCO iProcess Workspace session logged in on a queue and does not refresh that queue, any items removed from the queue (through purging, forwarding or releasing) will still be held in memory, causing the WIS memory footprint to grow.

WIS_LOCK_POOL_SIZES

WIS and WQS Process Configuration

Summary This attribute defines the number of locks in the internal lock pool used by the `WIS` process.

⚠️ Do not change the value of this process attribute unless you are advised to do by TIBCO Support.

Applies To The attribute can be set for a `WIS` process (only).

Permissible Values The attribute value must be one of the following:

Value	Meaning
TINY	Sets the size of the internal lock pool. (The actual numbers represented by these values are set internally by iProcess Engine.)
SMALL	
MEDIUM	
LARGE	
HUGE	
GIGANTIC	
VAST	

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	MEDIUM

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   The `WIS` process uses pools of locks to reduce its resource usage when handling large numbers of queues and work items. Because these locks are in pools, the resources required for locking do not increase as the number of work queues and work items increases. This attribute is only read when iProcess Engine starts up. Any changes that are made when iProcess Engine is running are ignored.

WIS_NEW_ITEM_BATCH_SIZE

WIS and WQS Process Configuration

Summary	This attribute value defines the number of new item requests to be batched together so more can be processed in a single write lock.
Applies To	The attribute can be set for a `WIS` process.
Permissible Values	The attribute value must be an integer between 0 and 500000 where 0 means that batching is not used.
Default Value	This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	0

Notes	If you have batching turned on and less than the batch size value number of items come into the queue, then the update thread, when it next wakes up, will process any items batched up. Or, if another RPC request comes in to add a new item that takes the batch size over the configured value, then all items will be processed.
	Using this attribute means that incoming items from the `WISMBD` are batched up before the index is updated (similar to pre 10.3 versions). This reduces the load on the system, but does mean that new items take longer to appear in the queue (by at most the WIS_INDEX_REFRESH period of time).

WIS_QCHANGE_EXTENDED_CHECK

WIS and WQS Process Configuration

Summary   This attribute changes the behavior of a WIS process as to whether it counts changes to the lock status of work items as changes to the work items and work queues.

Applies To   The attribute can be set for a WIS process.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	A change in the lock status of a work item is not counted as a change to the work item.
1	A change to the lock status of a work item is counted as a change to the work item.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   If this attribute is not set, changes to the lock status of work items are not counted as changes to queues for the purpose of monitoring changes in queues. This means that if a user just locks or keeps an item without making any other changes to a queue, then iProcess Objects or the SAL does not flag that any changes have been made to the queue, and even if the queue is refreshed, no changes are apparent.

If this attribute is set then the lock status is changed (the QPAR version number is updated and the OREC version number is incremented) in the following situations:

- when a work item is opened

- when the first work item that is not locked is opened

- when a work item is kept

This allows iProcess Engine to detect these changes in the queue.

WIS_SESSION_TIMEOUT

WIS and WQS Process Configuration

Summary This attribute defines the timeout period (in seconds) after which a `WIS` process will automatically shut down, starting from the time at which it was last accessed (by TIBCO iProcess Workspace, SAL application or iProcess Objects Server).

Applies To The attribute can be set for a `WIS` process.

Permissible Values The attribute value must be one of the following:

Value	Meaning
0	Do not timeout `WIS` processes.
n	The timeout period, where *n* is any integer value equal to or greater than 60.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	28800

The default value gives a timeout period of 8 hours.

The attribute only appears in the output of the `SHOW_ALL_ATTRIBUTES` command if you have explicitly assigned a value to it using the `SET_ATTRIBUTE` command. The default setting is not displayed.

See Also WIS_SESSION_TIMEOUT_SHUTDOWN

WIS_SESSION_TIMEOUT_SHUTDOWN

WIS and WQS Process Configuration

Summary This attribute defines the timeout period (in seconds) after which a WIS process will automatically shut down, starting from the time at which iProcess Engine was shut down.

Applies To The attribute can be set for a WIS process.

Permissible Values The attribute value must be an integer, with a minimum value of 60.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	WIS	0	300

The default value gives a timeout period of 5 minutes.

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

See Also WIS_SESSION_TIMEOUT

WIS_UNCACHE_PERIOD

WIS and WQS Process Configuration

Summary   This attribute defines the time period (in minutes) after which the cached work queue that waits to be handled by the WIS process is cleared from the cache, starting from the time at which the client who last accessed this work queue logs off.

> This attribute only applies to the work queues that contain the number of work items less than the number you defined in the WIS_CACHE_THESHOLD attribute.

Applies To   This attribute can be set for the WIS processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
n	The timeout period, where *n* is any integer value in minutes.

Default Value   The attribute is assigned the following default value when iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	0

The default value means never un-cache the work queue. It performs like previous versions of TIBCO iProcess Engine that did not support this attribute.

Notes   This attribute is used in conjunction with the WIS_CACHE_THRESHOLD attribute. If the cached work queue contains a number of work items that equals or exceeds the value of the WIS_CACHE_THRESHOLD attribute, it cannot be cleared from the cache by using the WIS_UNCACHE_PERIOD attribute.

See Also   WIS_CACHE_THRESHOLD

WIS_UPDATE_LENGTH

WIS and WQS Process Configuration

Summary   This attribute defines the maximum amount of time (in seconds) that the queue update thread in the WIS process performs updates for before going back to sleep.

Applies To   This attribute can be set for the WIS process (only).

Permissible Values   This attribute must be an integer with a minimum value of 5. There is no maximum value.

Default Value   The attribute is assigned the following default value when iProcess Engine node is installed.

Machine ID	Process	Instance	Value
0	WIS	0	120

Notes   The queue update thread wakes up every WIS_UPDATE_PERIOD seconds. It updates work queues for WIS_UPDATE_LENGTH seconds, then goes back to sleep. If it has updated all the queues before the WIS_UPDATE_LENGTH period has expired, it goes back to sleep immediately.

You should decrease the WIS_UPDATE_LENGTH value if you find that the update thread in the WIS process is using too much CPU.

See Configuring Queue Updates on page 415 for more information.

See Also   WIS_UPDATE_PERIOD

WIS_UPDATE_PERIOD

WIS and WQS Process Configuration

Summary	This attribute defines how often the queue update thread in the WIS process wakes up and updates the queues handled by the WIS process.
Applies To	This attribute can be set for the WIS process (only).
Permissible Values	This attribute must be an integer in the range 1 to 3600.
Default Value	The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	WIS	0	20

Notes	The queue update thread wakes up every WIS_UPDATE_PERIOD seconds. It updates work queues for WIS_UPDATE_LENGTH seconds, then goes back to sleep. If it has updated all the queues before the WIS_UPDATE_LENGTH period has expired, it goes back to sleep immediately.
	See Configuring Queue Updates on page 415 for more information.
See Also	WIS_UPDATE_LENGTH

WIS_USE_PRIORITY_ESCALATION

WIS and WQS Process Configuration

Summary   This attribute allows the use of priority escalation in the WIS process to be disabled.

Applies To   This attribute can be set for the WIS process (only).

Permissible   The attribute value must be one of the following:
Values

Value	Meaning
0	Priority ageing will no longer be processed. Every item that is displayed will use the default priority as set in the staffcfg file. Changes to priority fields will not affect the current priority value of a work item.
1	Priority ageing will still function.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	WIS	0	1

Notes   This attribute can be turned off for a small performance and CPU benefit. However, you should only do this if you are not using the Priority escalation feature as switching it off will prevent the priority value being automatically decremented. Be careful to ensure you are not using this feature before disabling it.

See Also   n/a

WQS_GATHER_RPC_STATS

WIS and WQS Process Configuration

Summary   This attribute allows the gathering of RPC call stats within the WQS process to be configurable.

Applies To   This attribute should be set for the WQS process.

Permissible   The attribute value must be one of the following:
Values

Value	Meaning
0	RPC stats gathering is turned off.
1	RPC stats gathering is turned on.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	WQS	0	0

Notes   When the attribute is turned off and then on again, any previously gathered stats are not retained.

To use the attribute, turn on R=2 debug on the WQS process, set the process attribute and the stats will be seen in the debug log.

TIBCO recommends that this attribute be turned off unless you specifically require these stats.

See Also   N/A

WQS_NUM_SEARCH_SLOTS

WIS and WQS Process Configuration

Summary This attribute defines the maximum number of slots available in the SWRPCMTS multi-threaded RPC server shared library for threads to perform queue searching. This is an internal design feature that limits the number of "worker" threads available for the WQS process to use to process RPC requests.

⚠️ Do not change the value of this process attribute unless you are advised to do so by TIBCO Support.

Applies To This attribute should be set for ALL processes.

Permissible Values This attribute must be an integer that is greater than or equal to the value of the RPC_SVR_CONTROL process attribute.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	50

Notes This attribute cannot be set when the WQS process is running. You must stop iProcess Engine if you want to change the value of this attribute.

When the iProcess Engine starts up the WQS process checks the value of the RPC_SVR_CONTROL process attribute. If it is:

- less than or equal to the WQS_NUM_SEARCH_SLOTS value, the WQS_NUM_SEARCH_SLOTS value is left unchanged.

- greater than the WQS_NUM_SEARCH_SLOTS value, WQS_NUM_SEARCH_SLOTS is reset to 2 * RPC_SVR_NUM_THREADS.

See Also RPC_SVR_CONTROL

WQS_PERSIST_SHMEM

WIS and WQS Process Configuration

Summary	This attribute defines how often (in seconds) the contents of the WQS/WIS shared memory are written to the wqs_index table in the database.
Applies To	This attribute can be set for the WQS process (only).
Permissible Values	The attribute value must be an integer in the range 1 to 3600.
Default Value	The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	WQS	0	300

Notes	When the WIS process starts up, it uses the total_items column in the wqs_index table to determine the number of work items in each work queue. It compares this value to the WIS_CACHE_THRESHOLD value to determine whether to cache the work queue.
	If TIBCO iProcess Engine is started, cases are loaded into a work queue, then the system is shut down again within the WQS_PERSIST_SHMEM value, the item counts in the wqs_index table will not match the actual item counts.
	See Configuring When WIS Processes Cache Their Queues on page 416 for more information.
See Also	WIS_CACHE_THRESHOLD

WQS_WIS_USER_COUNT

WIS and WQS Process Configuration

Summary This attribute defines the number of WIS processes that should be dedicated to handling user queues and group queues respectively.

Applies To The attribute can be set for the WQS process.

Permissible Values The attribute value must be a string, and can be either:

- a number, indicating the number of WIS processes that should be dedicated to handling user queues. For example:

  "2"

- a percentage in the range 1% to 99%, indicating the percentage of WIS processes that should be dedicated to handling user queues. For example:

  "20%"

Default Value This attribute is not defined on a newly installed iProcess Engine. In this case (or if the attribute is defined incorrectly), queues are allocated to WIS processes alphabetically, irrespective of whether they are user or group queues (either by round robin or on-demand allocation - see WQS_ROUND_ROBIN on page 51).

Notes The remaining WIS processes will be dedicated to handling group queues. Note that:

- There must always be at least one WIS available to handle user queues and one WIS to handle group queues if the attribute is defined. The attribute value should be set accordingly.

- If a percentage value is used, iProcess will round this figure down, subject to there being at least one WIS available to handle user queues. For example, the following table shows how different WQS_WIS_USER_COUNT values are interpreted, depending on the number of available WIS processes.

Value	Number of WIS processes	Resulting allocation for:	
		User queues	Group queues
"20%"	5	1	4
"50%"	5	2	3
"50%"	6	3	3
"90%"	5	4	1

| Value | Number of WIS processes | Resulting allocation for: | |
		User queues	Group queues
"90%"	20	18	2
"10%"	5	1	4

- If there are not enough WIS processes configured to create the specified allocation, the WQS_WIS_USER_COUNT value is ignored, default queue allocation is used, and one of the following messages is written to the sw_warn file:

    WQS_WIS_USER_COUNT ignored - too big

or

    WQS_WIS_USER_COUNT ignored - percentage too big

For example, if there are 5 WIS processes configured, the following WQS_WIS_USER_COUNT values would all generate an error as described:

"0" "0%" "5" "6" "100%" "150%"

- WIS processes can also be dedicated to handling explicitly specified queues - see Assigning a Queue Explicitly to a WIS Process on page 407. Dedicated queues are not considered when calculating the allocation of WIS processes to user queue or group queue pools.

The following table shows how the allocations described in the example above would be affected if one of the WIS processes was subsequently dedicated to handling a specific queue. (The values in bold font show the changes.)

| Value | Number of non-dedicated WIS processes | Resulting allocation for: | |
		User queues	Group queues
"20%"	4	1	**3**
"50%"	4	2	**2**
"50%"	5	**2**	3
"90%"	4	**3**	1
"90%"	19	**17**	2
"10%"	4	1	**3**

Message and Mbox Processing Configuration

The following process attributes allow you to configure how iProcess Engine processes messages.

Attribute	Description
DBQD_MAX_CACHED_MESSAGES	Defines the number of messages that are cached by the DBQD process when it requests a block of messages from a database message queue.
DBQD_MAX_FIL_SESSIONS	Defines the number of concurrent threads that the DBQD process uses to process RPC requests for messages from its cache from BG or WISMBD processes.
EMPTYMBOXSLEEP	Defines how long the Mbox Daemons will "sleep" when all Mbox queues in the Mbox set are empty.
EMPTYMBOXSLEEP_INC	Defines the number of seconds to increment the EMPTYMBOXSLEEP value by when a BG or WISMBD process requests a message from an empty Mbox.
EMPTYMBOXSLEEP_MAX	Defines the maximum value (in seconds) that EMPTYMBOXSLEEP can be set to.
IQL_RETRY_COUNT	Defines how many times a failed message in a message queue is retried before being moved to the exception queue.
IQL_RETRY_DELAY	Defines the delay (in seconds) between each retry attempt for a failed message in a message queue, before the message is moved to the exception queue.
MBSET_DEQUEUE_TIMEOUT	Defines a time threshold (in milliseconds) for the duration of dequeuing a message from an Mbox set.
MBSET_READ_BG	Defines the unique identifier of the Mbox set to be used by a BG process when dequeuing messages received from a WISMBD process.
MBSET_READ_PREDICT	Defines the unique identifier of the Mbox set to be used by a BGPREDICT process when posting case changes messages to a BG process.

Attribute	Description
MBSET_READ_WIS	Defines the unique identifier of the Mbox set to be used by a `WISMBD` process when dequeuing messages received from a `BG` process.
MBSET_WRITE_BG	Defines the unique identifier of the Mbox set to be used by a process when writing to a `BG` process.
MBSET_WRITE_PREDICT	Defines the unique identifier of the Mbox set to be used by a `BGPREDICT` process when posting case changes messages to a `BG` process.
MBSET_WRITE_WIS	Defines the unique identifier of the Mbox set (as defined in the `mbox_set` table) to be used by the `BG` process when writing to a `WISMBD` process.
THRESHOLD_FAIL_TIMES_TO_REPORT_IQL_MESSAGE	Defines how many times a failed message in a message queue is retried before sending a message to the Process Sentinels.
USE_NEXT_MESSAGE_TO_DEQUEUE	Defines the ways to purge vast quantities of cases in the Oracle AQs.

DBQD_MAX_CACHED_MESSAGES

Message and Mbox Processing Configuration

This attribute is currently only used on the DB2 version of iProcess Engine. It has no effect on the Oracle or SQL Server versions.

Summary This attribute defines the number of messages that are cached by the DBQD process when it requests a block of messages from a database message queue.

Applies To This attribute can be set for the DBQD process (only).

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	DBQD	0	1000

Notes Each BG and WISMBD process requests a message from one of its allocated message queues whenever it is not either already processing a message or sleeping. The DBQD process receives this request and returns a message from its cache for the specified queue. If the cache is empty, the DBQD process requests another block of DBQD_MAX_CACHED_MESSAGES messages from the database message queue to refill the cache.

The rate at which messages are processed from the cache depends upon the number of BG and WISMBD processes that are running, and the type of procedure being processed. For example, procedures involving significant use of deadlines or EAI steps would take longer to process than those involving normal steps.

Increasing the DBQD_MAX_CACHED_MESSAGES value increases the amount of memory used by the DBQD process and the time required to perform the caching operation. Decreasing this value means that the process needs to access the database to refill its cache more often.

See Also DBQD_MAX_FIL_SESSIONS, EMPTYMBOXSLEEP

DBQD_MAX_FIL_SESSIONS

Message and Mbox Processing Configuration

> This attribute is currently only used on the DB2 version of iProcess Engine. It has no effect on the Oracle or SQL Server versions.

Summary   This attribute defines the number of concurrent threads that the DBQD process uses to process RPC requests for messages from its cache from BG or WISMBD processes. You may need to alter this value according to the number of BG and WISMBD processes you have configured on the system.

Applies To   This attribute can be set for the DBQD process (only).

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	DBQD	0	5

See Also   DBQD_MAX_CACHED_MESSAGES

EMPTYMBOXSLEEP

Message and Mbox Processing Configuration

Summary   This attribute defines the number of milliseconds or seconds that a BG or WISMBD process sleeps when all Mbox queues in its Mbox set are empty.

Applies To   This attribute can be set for the BG, WISMBD or ALL processes.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	2

Notes   The value of the EMPTYMBOXSLEEP process attribute can be treated as the sleep time either in seconds or in milliseconds:

- If the value of this attribute is up to (including) 60, then it is treated as a measure of the time in seconds.

- If the value of this attribute is greater than 60, then it is treated as a measure of the time in milliseconds.

Whenever a BG or WISMBD process requests a message from an empty Mbox, the EMPTYMBOXSLEEP value is incremented by the EMPTYMBOXSLEEP_INC value until either:

- the EMPTYMBOXSLEEP_MAX value is reached, or

- a message is returned from the Mbox, in which case EMPTYMBOXSLEEP is reset to its configured value.

By tailoring the values of these three attributes to your particular system configuration, you can avoid unnecessary system overhead resulting from polling for messages on empty queues.

You may notice a delay in processing messages if the system is very quiet and the EMPTYMBOXSLEEP value has increased to its maximum. For example:

- A user releases a work item just after the BG process has polled the Mbox. The message remains in the Mbox until the sleep period has expired.

- The BG processes the release instruction and sends out the next work item. That message arrives in its Mbox just after the WISMBD process has polled it, and so remains there until the next sleep period has expired.

In this way, there could be a delay between the work item being released and the next work item arriving of approximately twice the EMPTYMBOXSLEEP_MAX value, even though the system is otherwise idle.

See Also   EMPTYMBOXSLEEP_INC, EMPTYMBOXSLEEP_MAX

EMPTYMBOXSLEEP_INC

Message and Mbox Processing Configuration

Summary	This attribute defines the number of milliseconds or seconds to increment the EMPTYMBOXSLEEP value by when a BG or WISMBD process requests a message from an empty Mbox.
Applies To	This attribute can be set for the BG, WISMBD or ALL processes.
Permissible Values	The attribute value must be a numeric value in the range 0 to EMPTYMBOXSLEEP_MAX.
Default Value	The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	2

See Also	EMPTYMBOXSLEEP, EMPTYMBOXSLEEP_MAX

EMPTYMBOXSLEEP_MAX

Message and Mbox Processing Configuration

Summary	This attribute defines the maximum value (in milliseconds or seconds) that EMPTYMBOXSLEEP can be set to.
Applies To	This attribute can be set for the BG, WISMBD or ALL processes.
Default Value	The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	2

See Also	EMPTYMBOXSLEEP, EMPTYMBOXSLEEP_INC

IQL_RETRY_COUNT

Message and Mbox Processing Configuration

This attribute value is only used on the SQL Server and DB2 iProcess Engine variants. On the Oracle variant this value is set using Oracle AQ parameters.

Summary This attribute defines how many times a failed message in a message queue is retried before being moved to the exception queue.

Applies To This attribute can be set for the BG process.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	12

Notes If the IQL_RETRY_COUNT limit is exceeded, the message is moved to the exception queue (also known as the dead queue or poison queue), and manual intervention by a system administrator will be necessary to resolve the problem and progress the case that the message belongs to.

See Also IQL_RETRY_DELAY, DEPLOY_XSL_OUT_ENCODING.

IQL_RETRY_DELAY

Message and Mbox Processing Configuration

	This attribute value is only used on the SQL Server and DB2 iProcess Engine variants. On the Oracle variant this value is set using Oracle AQ parameters.

Summary This attribute defines the delay (in seconds) between each retry attempt for a failed message in a message queue, before the message is moved to the exception queue.

Applies To This attribute can be set for the BG process.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	300

Notes A failed message is retried a number of times up to the IQL_RETRY_COUNT limit. If that limit is exceeded the message is moved to the exception queue (also known as the dead queue or poison queue), and manual intervention by a system administrator will be necessary to resolve the problem and progress the case that the message belongs to.

See Also IQL_RETRY_COUNT, DEPLOY_XSL_OUT_ENCODING.

MBSET_DEQUEUE_TIMEOUT

Message and Mbox Processing Configuration

Summary This attribute defines a time threshold (in milliseconds) for the duration of dequeuing a message from an Mbox set.

If the time for dequeuing the message is equal to or over the defined threshold, the dequeue operation keeps processing, and the time period of dequeuing the message will be logged in the sw_warn file. If the time for dequeuing the message is less than the threshold, the dequeue operation will be processed and no log is written in the sw_warn file.

Applies To This attribute can be set for the BG process.

Permissible Values The attribute value must be one of the following:

Value	Meaning
n	The timeout threshold, where *n* is any integer value that is 0 - 60000.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	200

Notes If you set the value of the attribute to 0, the time period of dequeuing a message will be logged in the sw_warn file each time after dequeuing the message from an Mbox set. This may lead to an extremely lage log file, since the log file records each time period of dequeuing messages, which may be problematic.

MBSET_READ_BG

Message and Mbox Processing Configuration

Summary	This attribute defines the unique identifier of the Mbox set (as defined in the mbox_set table) to be used by a BG process when dequeuing messages received from a process.
Applies To	This attribute can be set for the BG or ALL processes.
Default Value	The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value	Notes
0	BG	1	3	for Mbox set WISBGMBSET1
0	BG	2	3	for Mbox set WISBGMBSET1
0	BG	3	4	for Mbox set WISBGMBSET2
0	BG	4	4	for Mbox set WISBGMBSET2
0	ALL	0	1	for all other processes (TIBCO iProcess Objects, swbatch etc.)

Notes	See Default Message Handling Configuration on page 351 for more information about how these default values are used.

MBSET_READ_PREDICT

Message and Mbox Processing Configuration

Summary	This attribute defines the unique identifier of the Mbox set (as defined in the mbox_set table) to be used by a BGPREDICT process when dequeuing case change messages received from a BG process.
Applies To	This attribute can be set for the BGPREDICT or ALL processes.
Default Value	The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	3

MBSET_READ_WIS

Message and Mbox Processing Configuration

Summary This attribute defines the unique identifier of the Mbox set (as defined in the `mbox_set` table) to be used by a `WISMBD` process when dequeuing messages received from a `BG` process.

Applies To This attribute can be set for the `WISMBD` or ALL processes.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	2

MBSET_WRITE_BG

Message and Mbox Processing Configuration

Summary   This attribute defines the unique identifier of the Mbox set (as defined in the `mbox_set` table) to be used by a process when posting messages to a `BG` process.

Applies To   This attribute can be set for the `BG`, `WIS`, `SPO`, `RPC_POOL` or ALL processes.

Default Value   The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value	Notes
0	WIS	1	3	for Mbox set WISBGMBSET1
0	WIS	2	3	for Mbox set WISBGMBSET1
0	WIS	3	3	for Mbox set WISBGMBSET1
0	WIS	4	4	for Mbox set WISBGMBSET2
0	WIS	4	4	for Mbox set WISBGMBSET2
0	WIS	4	4	for Mbox set WISBGMBSET2
0	ALL	0	1	for all other processes (TIBCO iProcess Objects, swbatch etc.)

Notes   See Default Message Handling Configuration on page 351 for more information about how these default values are used.

MBSET_WRITE_PREDICT

Message and Mbox Processing Configuration

Summary	This attribute defines the unique identifier of the Mbox set (as defined in the `mbox_set` table) to be used by a `BG` process when posting case change messages to a `BGPREDICT` process.
Applies To	This attribute can be set for the `BG` or ALL processes.
Default Value	The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	3

MBSET_WRITE_WIS

Message and Mbox Processing Configuration

Summary   This attribute defines the unique identifier of the Mbox set (as defined in the mbox_set table) to be used by a BG process when posting messages to a WISMBD process.

Applies To   This attribute can be set for the BG, RPCBG or ALL processes.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	2

THRESHOLD_FAIL_TIMES_TO_REPORT_IQL_MESSAGE

Message and Mbox Processing Configuration

Summary   When a process fails to send an IQL message, it will try again for the number of times defined by IQL_RETRY_COUNT. During this retry procedure, if the number of retries exceeds the value of this attribute (but does not reach the value of IQL_RETRY_COUNT), the process sends a warning message to the Process Sentinels.

Applies To   This attribute can be set for the BG process.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	3

See Also   IQL_RETRY_COUNT, IQL_RETRY_DELAY.

USE_NEXT_MESSAGE_TO_DEQUEUE

Message and Mbox Processing Configuration

This attribute value is only of benefit when there is a large buildup of messages in the Oracle AQs, for example, when a large number of messages are injected as part of a batch. If there is no buildup of messages in the Oracle AQs, there is no benefit to this setting.

Summary This attribute defines the ways to purge vast quantities of cases in the Oracle AQs.

Applies To This attribute can be set for the BG process.

Permissible Values The attribute value must be one of the following:

Value	Meaning
0	To purge a large quantity of cases, the subsequent dequeue operations need to perform new queries for each message rather than to fetch the messages that are currently in the queue.
1	A large quantity of cases would be purged effectively by dequeuing messages, which are currently in the queue.

Default Value This attribute is not defined automatically when you install or upgrade iProcess Engine. To use this attribute, you must explicitly assign a value to it.

Notes When the value of the USE_NEXT_MESSAGE_TO_DEQUEUE attribute is set to 1, the dequeuing process loads all messages that are currently in the queue, and iterates through them as a single result set. In this way, it avoids performing any new queries for each message.

Any other messages outside the result set, even with higher internal message queue priorities, will not be picked up until all the messages in the result set have been dequeued.

See Also None.

Sequence Numbering Configuration

The following process attributes allow you to configure how iProcess Engine caches sequence numbers. For more information about sequence numbers see Sequence Number Caching on page 108.

Attribute	Description
CNUM_SEQ_CACHE	Defines the number of case numbers to be cached.
REQID_SEQ_CACHE	Defines the number of REQ IDs to be cached.
WIS_INDEX_REFRESH	Defines the number of Wait IDs to be cached.

CNUM_SEQ_CACHE

Sequence Numbering Configuration

Summary This attribute defines the number of case numbers to be cached.

Applies To This attribute can be set for the BG, SWBATCH, WIS, SPO, SSOLITE or ALL processes.

Default Value The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	50
0	SWBATCH	0	5
0	WIS	0	50
0	SPO	0	5
0	SSOLITE	0	5 (Oracle) or 10 (SQL and DB2)

Notes Case number caching can provide a performance benefit when applied to the BG, WIS , RPC_POOL, and SWBATCH processes. It should not be used with other processes. For more information see Sequence Number Caching on page 108.

If you use case number caching, you should note that it is possible for a lower case number to be started after a higher case number.

For example, suppose that a WIS process has 50 case numbers (1 to 50) cached, and a user uses the SWUTIL CSTART command to start a case. The case will have case number 51 - the next available number obtained from the cnum_sequence table.

However, if a user then starts a case through the WIS, that case will have case number 1 - the next available number in the cached sequence.

Thus, the start date/time for case number 1 will be later than the start date/time for case number 51.

See Also REQID_SEQ_CACHE, WIS_INDEX_REFRESH.

REQID_SEQ_CACHE

Sequence Numbering Configuration

Summary   This attribute defines the number of REQ IDs to be cached.

Applies To   This attribute can be set for the BG, SWBATCH, WIS, SPO, SSOLITE or ALL processes.

Default Value   The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	50
0	SWBATCH	0	5
0	WIS	0	50
0	SPO	0	5
0	SSOLITE	0	5 (Oracle) or 10 (SQL and DB2)

Notes   REQ ID caching can provide a performance benefit when applied to the BG, WIS, RPC_POOL and SWBATCH processes. It should not be used with other processes. For more information see Sequence Number Caching on page 108.

See Also   CNUM_SEQ_CACHE, WIS_INDEX_REFRESH

WAITID_SEQ_CACHE

Sequence Numbering Configuration

Summary   This attribute defines the number of Wait IDs to be cached.

Applies To   This attribute can be set for the BG or ALL processes.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   Case number caching can provide a performance benefit when applied to the BG process (if waits are used in procedures being processed by the BG process). It should not be used with other processes. For more information see Sequence Number Caching on page 108.

See Also   CNUM_SEQ_CACHE, REQID_SEQ_CACHE

Transaction Control Configuration

The following process attributes allow you to configure how iProcess Engine handles transactions.

Attribute	Description
BG_MAX_ACTIONS_PER_TRANS	Defines the limit of actions per workflow transaction.
CHECK_EAIWITHDRAW_ONPURGE	Defines whether or not iProcess checks if any outstanding delayed release EAI steps have been successfully withdrawn before committing the purge transaction.
EAI_STEP_TIMEOUT	Defines a time threshold (in milliseconds) for the duration of processing an EAI step.

BG_MAX_ACTIONS_PER_TRANS

Transaction Control Configuration

Summary This attribute limits the number of steps sent or withdrawn during the processing of a single workflow transaction (i.e. the number of EAI steps that can be processed in one transaction without any other step types in between).

Applies To This attribute can be defined for the BG, RPCBG and BGPREDICT processes.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	1000

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes When this limit is reached the workflow transaction is aborted and an appropriate message is logged to the sw_warn log file. See for more information.

CHECK_EAIWITHDRAW_ONPURGE

Transaction Control Configuration

Summary   When you purge a case that contains an outstanding delayed release EAI step, the BG process attempts to withdraw the EAI step (sending an instruction to the external system to remove any data associated with that step). By default, iProcess checks if any outstanding delayed release EAI steps have been successfully withdrawn before committing the purge transaction.

The CHECK_EAIWITHDRAW_ONPURGE process attribute allows you to configure this behavior to suit your requirements.

If the withdrawal fails, the data is left in the external system even though the case is purged. The external system and iProcess case are thus out of synchronization with each other. Therefore, TIBCO recommends that the default setting (1) is used instead.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	iProcess assumes that the EAI step is successfully withdrawn, commits the transaction and purges the case. If the value is set to 0, iProcess assumes that the withdrawal succeeds, commits the transaction and purges the case.
1	iProcess checks whether the EAI step is successfully withdrawn or not. If the withdraw: • succeeds, iProcess commits the transaction and purges the case. • fails, iProcess rolls back the transaction and does not purge the case. This is the default value.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	1

Notes   If CHECK_EAIWITHDRAW_ONPURGE is set to 1 you should note the following
implications:

- If you are using a custom shell EAI Server Plug-in (developed using the EAI
  SDK), and you want to use delayed release EAI steps, you must implement
  the EAIRun_Withdraw function. The iProcess Suite uses the return value from
  this function to determine whether it should commit (EAI_SUCCESS) or
  rollback (any return value other than EAI_SUCCESS) the purge transaction.

- If the purge transaction fails, it will be automatically re-queued and retried a
  number of times, as determined by the values of the IQL_RETRY_COUNT and
  IQL_RETRY_DELAY process attributes.

- The external system is responsible for handling failed withdraws, and
  ensuring that the withdraw attempt ultimately succeeds. Otherwise, cases
  will be left in iProcess that cannot be purged.

- If you use the TIBCO iProcess Workspace's Case Administration tool to purge
  cases (by selecting a case and clicking the Purge Case(s) button), if the purge
  transaction fails the case will still be visible when you click the **Refresh**
  button.

EAI_STEP_TIMEOUT

Transaction Control Configuration

Summary	This attribute defines a time threshold (in milliseconds) for the duration of processing an EAI step.

If the time for processing an EAI step is equal to or over the defined threshold, the EAI step keeps processing, and the time period of processing the EAI step will be logged in the sw_warn file. If the time for processing an EAI step is less than the defined threshold, the EAI step will be processed and no log is written in the sw_warn file.

Applies To   This attribute can be set for the BG process.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
n	The timeout threshold, where *n* is any integer value that is 0 - 60000.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	2000

Notes   If you set the value of the attribute to 0, the time period of processing an EAI step will be logged in the sw_warn file each time after processing the EAI step. This may lead to an extremely large log file, since the log file records each time period of processing EAI steps, which may be problematic.

Activity Monitoring and Work Queue Delta Configuration

The following process attributes allow you to configure how iProcess Engine performs activity monitoring and Work Queue Delta publication.

For more information about:

- administering activity monitoring and Work Queue Delta publication, see .

- configuring activity monitoring, see "Configuring Activity Monitoring" in *TIBCO iProcess Modeler Integration Techniques*.

Attribute	Description
AUDIT_CASEDATA_CHANGED	Defines whether or not to log the case data changes in the audit trail when these changes are made by iProcess Insight, iProcess Workspace (Browser), or the `setCaseData` TIBCO iProcess Server Objects interface, rather than by normal step processing.
IAPJMS_LANGUAGE	Defines the character set to be used for output encoding for messages published to the `IAPJMS` process from the `BG` or `WIS` process.
IAPJMS_PORTNO	Defines the port number that is used for message communications between the `BG` process and the IAPJMS library.
IAPJMS_PUBLISH	Defines whether or not the `BG` process is enabled to publish audit activities to the `IAPJMS` process.
IAPJMS_ROLLBACK	Defines whether or not failed message transactions should be rolled back.
IAPJMS_SIMPLETOPIC	Defines whether or not the JMS topic name is static or dynamically configured at run-time.
IAPJMS_SYNCHRONOUS	Defines whether message delivery is synchronous or asynchronous.
IAPJMS_TIMEOUT	Defines how long the `IAPJMS` process should wait before it times out if there is a network error.

Attribute	Description
IAPJMS_TOPICNAME	Defines the topic name for the JMS destination if activity monitoring is enabled.
JVMPROPS	Defines the JVM attributes that should be specified for the Java Virtual Machine when it is started.
PUBLISH_SYS_EVENT_METHOD	Defines in which way you want to publish system events.
SE_WORKER_PORTNO	Defines the port number that is used for message communications between the "worker" process (Process Sentinel) and the processes that audit system events if you want to published system events using TIBCO Hawk Agent.
SEJMS_TOPICNAME	Defines the JMS topic name for the JMS destination that is used to publish system events, if publishing system events is enabled.
SHUTDOWN_MIGRATION_MESSAGE	Defines whether or not to contain in the published Monitor Event Detail message the audit trail message (with message ID 34), which writes the information about procedure migration.
SWLIB_PATH	Defines the directory where the IAPJMS process will look for the Java libraries that it needs.
SYSTEM_EVENT_LOGGING	Defines whether you want to audit system events after installing or upgrading to TIBCO iProcess Engine 11.6.0.
WQDJMS_PORTNO	Defines the port number that is used for work queue delta messages between the WIS process and the IAPJMS process.
WQDJMS_TOPICNAME	Defines the default topic name for the JMS destination used by the WIS process for work queue delta publication.

AUDIT_CASEDATA_CHANGED

Activity Monitoring and Work Queue Delta Configuration

Summary	This attribute defines whether or not to log the case data changes in the audit trail when these changes are made by iProcess Insight, iProcess Workspace (Browser), or the setCaseData TIBCO iProcess Server Objects interface, rather than by normal step processing.
Applies To	This attribute applies to the SPO process.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
0	Case data changes made by iProcess Insight, iProcess Workspace (Browser), or the setCaseData TIBCO iProcess Server Objects interface are not logged in the audit trail.
1	Case data changes made by iProcess Insight, iProcess Workspace (Browser), or the setCaseData TIBCO iProcess Server Objects interface are logged in the audit trail.

Default Value	This attribute is not defined automatically when you install or upgrade iProcess Engine. To use this attribute, you must explicitly assign a value to it using the swadm set_attribute command. For more information about this command, see Set a Process Attribute on page 144.
Notes	If you change the value of this attribute, you do not need to stop and restart iProcess Engine to make the change take effect.
	The changes of the case data made by iProcess Insight, iProcess Workspace (Browser), or the setCaseData TIBCO iProcess Server Objects interface are recorded as an audit message "133: Case data changed by *username*" in the audit.mes file located in the *SWDIR*\etc\language.lng directory.
See Also	None.

IAPJMS_LANGUAGE

Activity Monitoring and Work Queue Delta Configuration

Summary	This attribute defines the character set to be used for output encoding for messages published to the IAPJMS process from the BG or WIS process.
Applies To	The attribute should be set for BG, WIS, or ALL processes.
Permissible Values	This attribute value must be a valid character encoding name.
Default Value	The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	UTF-8

Notes	None.
See Also	IAPJMS_PUBLISH, IAPJMS_SYNCHRONOUS, IAPJMS_ROLLBACK, IAPJMS_TIMEOUT, IAPJMS_TOPICNAME, IAPJMS_SIMPLETOPIC, JVMPROPS, SWLIB_PATH

IAPJMS_PORTNO

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines the port number that is used for message communications between the BG process and the IAPJMS process.

Applies To   The attribute should be set for ALL processes.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	9071

Notes   If you change the value of this attribute, the change does not take effect until you stop and restart iProcess Engine.

See Also   IAPJMS_PUBLISH, IAPJMS_SYNCHRONOUS, IAPJMS_ROLLBACK, IAPJMS_TIMEOUT, IAPJMS_TOPICNAME, IAPJMS_SIMPLETOPIC, JVMPROPS, SWLIB_PATH

IAPJMS_PUBLISH

Activity Monitoring and Work Queue Delta Configuration

Summary	This attribute defines whether or not the BG process is enabled to publish monitored activities to the IAPJMS process.
Applies To	The attribute must be set for ALL processes.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
0	Activity monitoring is disabled.
1	Activity monitoring is enabled.

Default Value   The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

Notes   If activity monitoring is enabled then activity information about auditable objects (for example, procedures and steps) can be published to an external application. This enables real-time monitoring of auditable objects so that mission critical or important business events can be easily monitored.

See Also   IAPJMS_PORTNO, IAPJMS_SYNCHRONOUS, IAPJMS_ROLLBACK, IAPJMS_TOPICNAME, IAPJMS_SIMPLETOPIC, JVMPROPS, SWLIB_PATH

IAPJMS_ROLLBACK

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines whether or not failed message transactions should be rolled back.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	The iProcess Engine transaction succeeds and is committed even if the message fails. This means that failed JMS messages cause an error to be written to the sw_error file but the failed message transaction is not rolled back.
1	Any error causes the BG process to fail the current instruction and roll back any outstanding iProcess Engine transactions.

Default Value   The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	1

Notes   To ensure reliable message delivery, TIBCO recommends that the value of this attribute be set to 1. This means that failed JMS messages cause an error to be written to the sw_error file and are rolled back. See iProcess Engine Log Files on page 439 for more information about log files.

See Also   IAPJMS_PUBLISH, IAPJMS_PORTNO, IAPJMS_SYNCHRONOUS, IAPJMS_TOPICNAME, IAPJMS_SIMPLETOPIC, JVMPROPS, SWLIB_PATH

IAPJMS_SYNCHRONOUS

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines the JMS message delivery method. There are two delivery methods, synchronous or asynchronous.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	The JMS message delivery method is asynchronous. The message is assumed to have been processed correctly if the message was sent successfully to the IAPJMS process.
1	The JMS message delivery method is synchronous. When the message is sent, a receipt is requested. The BG process waits until the IAPJMS process has confirmed the message has been published. If the message is not published, an error is written to the sw_error file.

Default Value   The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	1

Notes   If you chose the synchronous message delivery method, there will be an impact on the performance of your iProcess Engine.

See Also   IAPJMS_PUBLISH, IAPJMS_PORTNO, IAPJMS_ROLLBACK, IAPJMS_TIMEOUT, IAPJMS_TOPICNAME, IAPJMS_SIMPLETOPIC, JVMPROPS, SWLIB_PATH

IAPJMS_TIMEOUT

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines the amount of time (in seconds) before the IAPJMS process should timeout, for example, if there is a network error.

Applies To   This attribute must be set for ALL processes.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	30

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   If you change the value of this attribute, the change does not take effect until you stop and restart iProcess Engine.

See Also   IAPJMS_PUBLISH, IAPJMS_PORTNO, IAPJMS_SYNCHRONOUS, IAPJMS_TOPICNAME, IAPJMS_SIMPLETOPIC, JVMPROPS, SWLIB_PATH

IAPJMS_TOPICNAME

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines the JMS topic name for the JMS destination that is used to audit an individual case of a procedure and system events, if activity monitoring is enabled.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be a string. The JMS topic name format depends on your J2EE environment. See the documentation supplied with your J2EE Application Server for more information about how you should format your JMS topic name for your J2EE environment. However, iProcess Engine forces a maximum length of 511 characters for the length of the process attribute.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	IATOPIC

Notes   If activity monitoring is enabled, the BG process sends JMS messages to a JMS topic name that you can specify using this attribute. The JMS topic name can be static or dynamically configured at run-time.

This attribute is used with the IAPJMS_SIMPLETOPIC process attribute:

- If the value of IAPJMS_SIMPLETOPIC is 1, the JMS topic name specified in the IAPJMS_TOPICNAME process attribute is static.

- If the value of IAPJMS_SIMPLETOPIC is 0, the JMS topic name specified in the IAPJMS_TOPICNAME process attribute is dynamically configured at run-time to include the iProcess procedure name and step name.

  For example, if the IAPJMS_TOPICNAME is IAPTOPIC and IAPJMS_SIMPLETOPIC is 0, then all messages are addressed to one of the following JMS topic names, depending on the activity being audited:

  — IAPTopic.*procedurename*.START

  — IAPTopic.*procedurename*.*stepname*.START

  — IAPTopic.*procedurename*.*stepname*.END

  — IAPTopic.*procedurename*.END

where:

— *procedurename* is the name of the iProcess procedure

— *stepname* is the name of the step in the iProcess procedure.

Some applications demand that the JMS topic name be configured this way. However, you may want to configure the JMS topic name this way if you want to use lots of small topics as opposed to one single large topic.

The following table shows which audit trail messages are logged to which topics. (See Appendix D on page 449 for a complete listing of audit trail messages and their corresponding Message IDs).

JNDI Name	Activity (Message ID)
IAPTopic.*procedurename*.START	Case started by *UserName* (000)
IAPTopic.*procedurename.stepname*.START	StepDescription processed to *UserName* (001)
	StepDescription forwarded to *UserName* (004)
	Sub-Case started from *StepDescription* (016)
IAPTopic.*procedurename.stepname*.END	All activities not covered by any of the other listed topics.
IAPTopic.*procedurename*.END	Case terminated normally (009)
	Case terminated prematurely by *UserName* (008)
	Case terminated abnormally (007)
	Any other activity that has a blank stepname.

See Also IAPJMS_PUBLISH, IAPJMS_PORTNO, IAPJMS_ROLLBACK, IAPJMS_SYNCHRONOUS, IAPJMS_SIMPLETOPIC, IAPJMS_TIMEOUT, JVMPROPS, SWLIB_PATH

IAPJMS_SIMPLETOPIC

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines whether or not the JMS topic is static or dynamically configured at run-time.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
1	The JMS topic name is static.
0	The JMS topic name is dynamically configured at run-time.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	1

Notes   This attribute is used with the `IAPJMS_TOPICNAME` process attribute.

See Also   IAPJMS_PUBLISH, IAPJMS_PORTNO, IAPJMS_SYNCHRONOUS, IAPJMS_ROLLBACK, IAPJMS_TIMEOUT, IAPJMS_TOPICNAME, JVMPROPS, SWLIB_PATH

JVMPROPS

Activity Monitoring and Work Queue Delta Configuration

Summary This attribute defines the JVM attributes that should be specified for the Java Virtual Machine when it is started.

Applies To This attribute can be set for ALL processes.

Permissible Values The attribute value must be a string. See the documentation supplied with your J2DK application for more information about how you should format the JVMPROPS attribute for your J2DK environment.

Default Value The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	NULL

Notes If activity monitoring is enabled, you can use this process attribute to configure any JVM attributes, for example debug values, that should be specified for the Java Virtual Machine when it is started.

If you change the value of this attribute, the change does not take effect until you stop and restart the process that you have changed the attribute value for.

See Also IAPJMS_PUBLISH, IAPJMS_PORTNO, IAPJMS_ROLLBACK, IAPJMS_SIMPLETOPIC, IAPJMS_SYNCHRONOUS, IAPJMS_TIMEOUT, IAPJMS_TOPICNAME, SWLIB_PATH

PUBLISH_SYS_EVENT_METHOD

Activity Monitoring and Work Queue Delta Configuration

Summary	This attribute defines in which way you want to publish system events.
Applies To	This attribute can be set for ALL processes.
Permissible Values	This attribute value must be one of the following:

Value	Meaning
0	System events are not published.
1	System events are published using Windows Event Viewer. **Note**: This function is only available for Microsoft Windows system.
2	System events are published as TIBCO Hawk alerts on TIBCO Hawk Event Service. For more information about TIBCO Hawk Event Service, see *TIBCO Hawk Installation, Configuration, and Administration Guide*.
3	System events are published using the IAPJMS process. You can check the published XML file in the iapjms_java.log file, which is located in the *SWDIR*/logs directory.

Default Value The PUBLISH_SYS_EVENT_METHOD attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

Note If you want to publish system events, you must audit them first. Set the value of the SYSTEM_EVENT_LOGGING process attribute to 1 to audit system events.

If the system events are published using TIBCO Hawk Agent, the processes that audit the system events, send the event details to the "worker" process (Process Sentinel) through a socket. Then the "worker" process sends the event details to TIBCO Hawk Agent. To define a socket number in the connection between the "worker" process and processes, which audit system events, set the SE_WORKER_PORTNO process attribute.

If you use the IAPJMS process to publish system events, the topic name on which the events are published depends on the following conditions:

- If the value of the SEJMS_TOPICNAME process attribute is set, the events will be published under the topic name that is set in the SEJMS_TOPICNAME process attribute.

- Otherwise, the events will be published under the topic name that is set in the IAPJMS_TOPICNAME process attribute.

See Also SYSTEM_EVENT_LOGGING, SEJMS_TOPICNAME, IAPJMS_TOPICNAME, SE_WORKER_PORTNO

SE_WORKER_PORTNO

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines the port number that is used for message communications between the "worker" process (Process Sentinel) and the processes that audit system events if you want to publish system events using TIBCO Hawk Agent.

Applies To   This attribute can be set for ALL processes.

Default Value   This attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	9085

Note   If you change the value of this attribute, the change does not take effect until you stop and restart iProcess Engine.

The port number must be unique for each installation of iProcess Engine on the same server even if system events are not configured to be published. Otherwise, the second subsequent instances will fail.

If the system events are published using TIBCO Hawk Agent, the processes that audit the system events, send the event details to the "worker" process (Process Sentinel) through a socket. Then the "worker" process sends the event details to TIBCO Hawk Agent. This process attribute is used to define a socket number for the connection between the "worker" process and processes, which audit system events.

See Also   SYSTEM_EVENT_LOGGING, PUBLISH_SYS_EVENT_METHOD

SEJMS_TOPICNAME

Activity Monitoring and Work Queue Delta Configuration

Summary	This attribute defines the JMS topic name for the JMS destination that is used to publish system events, if publishing system events is enabled.
Applies To	This attribute can be set for ALL processes.
Permissible Values	This attribute value must be a string. The format of JMS topic name depends on your J2EE environment. See the documentation supplied with your J2EE Application Server for more information about how to format the JMS topic name for the J2EE environment. The maximum length of this process attribute is 511 characters.
Default Value	This attribute is not defined automatically when you install or upgrade TIBCO iProcess Engine. To use this attribute, you must sepcifically assign a value to it by using the swadm set_attribute command. For more information about this command, see Set a Process Attribute on page 144.
Note	This attribute is used with the PUBLISH_SYS_EVENT_METHOD process attribute.

SHUTDOWN_MIGRATION_MESSAGE

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines whether or not to contain in the published Monitor Event Detail message the audit trail message (with message ID 34), which writes the information about procedure migration.

Applies To   This attribute can be set for the BG processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
1	The information about procedure migration is not contained in the published Monitor Event Detail message.
0	The audit trail message (with message ID 34), which writes the information about procedure migration is contained in the Monitor Event Detail message.

Default Value   The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
1	BG	0	0

Notes   When releasing a new version of a procedure and choosing to migrate existing cases of that procedure to the new version, the following audit trail entry (message ID 34) is generated for each case:

`Case migrated from Procedure` *StepName* `to` *StepDescription* `by` *UserName*`.`

If TIBCO iProcess Engine is configured to publish activity information, and the value of the SHUTDOWN_MIGRATION_MESSAGE attribute is set to 0, then a corresponding Monitor Event Detail message is published for each case, containing this audit message (with ActivityID=34).

For more information about the audit trail message, see Understanding Audit Trails on page 449.

See Also   None.

SWLIB_PATH

Activity Monitoring and Work Queue Delta Configuration

Summary This attribute defines the directory where the IAPJMS process will look for the Java libraries that it needs.

Applies To This attribute can be set for ALL processes, but is currently only used by the IAPJMS, BG, BGPREDICT and RPCBG process.

Permissible Values The attribute value must be a fully qualified pathname to a directory that contains a full Java Runtime Environment (JRE).

Default Value The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	See the Notes following.

Notes When a process that uses this attribute starts up, it searches the system's shared library/command path for the Java libraries that it needs.

When SWLIB_PATH is set its value is prefixed to the system's shared library/command path. The default value points to the Java libraries that are distributed with iProcess Engine, as shown in the following table.

Platform	Default SWLIB_PATH Value...	...is prefixed to the environment variable
HP-UX (HP)	On HP-9000: • *SWDIR*/java/lib/PA_RISC2.0/server:*SWDIR*/java/lib/PA_RISC2.0 On HP-Itanium: • *SWDIR*/java/lib/IA64N/server:*SWDIR*/java/lib/IA64N	SHLIB_PATH
AIX	*SWDIR*/java/bin/classic:*SWDIR*/java/bin	LIBPATH
SunOS	*SWDIR*/java/lib/sparc/server:*SWDIR*/java/lib/sparc	LD_LIBRARY_PATH

Platform	Default SWLIB_PATH Value...	...is prefixed to the environment variable
Linux	*SWDIR*/java/lib/i386/server: *SWDIR*/java/lib/i386	LD_LIBRARY_PATH
Windows	%*SWDIR*%java\bin\client	PATH

⚠️ You should only change SWLIB_PATH if you have a specific requirement to use different Java libraries from the default versions distributed with iProcess Engine. If you specify a directory that does not contain the necessary Java libraries, the process using the attribute will fail.

⚠️ On AIX, the IAPJMS process is linked to the libjvm.a Java library. Some Java 1.5 builds, however, supply a libjvm.so library either in addition to, or instead of, the libjvm.a library.

If the particular build of Java 1.5 that you wish to use in SWLIB_PATH only includes a libjvm.so library, you must either:

- copy it to libjvm.a, or
- create a symbolic link called libjvm.a, which links to the provided libjvm.so.

See Also IAPJMS_PUBLISH, IAPJMS_PORTNO, IAPJMS_ROLLBACK, IAPJMS_SIMPLETOPIC, IAPJMS_SYNCHRONOUS, IAPJMS_TIMEOUT, IAPJMS_TOPICNAME

SYSTEM_EVENT_LOGGING

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines whether you want to audit system events after installing or upgrading to TIBCO iProcess Engine version 11.6.0.

Applies To   This attribute can be set for ALL processes.

Permissible Values   This attribute value must be one of the following:

Value	Meaning
0	System events are not audited after installing or upgrading to TIBCO iProcess Engine version 11.6.0.
1	System events are audited after installing or upgrading to TIBCO iProcess Engine version 11.6.0.

Default Value   After installing TIBCO iProcess Engine version 11.6.0, the system events are audited by default. This attribute is assigned the following default value after installing TIBCO iProcess Engine version 11.6.0.

Machine ID	Process	Instance	Value
0	ALL	0	1

After upgrading to TIBCO iProcess Engine version 11.6.0, the system events are not audited by default. This attribute is assigned the following default value after upgrading to TIBCO iProcess Engine version 11.6.0.

Machine ID	Process	Instance	Value
0	ALL	0	0

Note   To publish system events after auditing the events, you have to set the PUBLISH_SYS_EVENT_METHOD process attribute after you have audited system events.

See Also   PUBLISH_SYS_EVENT_METHOD

SE_WORKER_PORTNO

WQDJMS_PORTNO

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines the port number that is used for work queue delta message communications between the WIS processes and the IAPJMS process. It is read when iProcess Engine starts up.

Applies To   The attribute should be set for ALL processes.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	9075

Notes   If you change the value of this attribute, the change does not take effect until you stop and restart iProcess Engine.

See Also   WQDJMS_TOPICNAME

WQDJMS_TOPICNAME

Activity Monitoring and Work Queue Delta Configuration

Summary   This attribute defines the default JMS topic name for the JMS destination used for work queue delta messages, if Work Queue Delta Publication via JMS is in use. This default can be overridden for an individual subscription, if that subscription supplies a different topic name.

Applies To   By default this attribute applies to ALL processes.

Permissible Values   The attribute value must be a string. The JMS topic name format depends on your J2EE environment. See the documentation supplied with your J2EE Application Server for more information about how you should format your JMS topic name for your J2EE environment. However, iProcess Engine forces a maximum length of 511 characters for the length of the process attribute.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	WQDTOPIC

Notes   If work queue delta monitoring is enabled, the `WIS` process sends JMS messages to a JMS topic name that you can specify using this attribute. The JMS topic name can be static or dynamically configured at run-time.

See Also   WQDJMS_PORTNO

TIBCO Rendezvous Configuration

The following process attributes allow you to configure how iProcess Engine communicates with TIBCO Rendezvous.

Attribute	Description
RV_DAEMON	Configures the iProcess Server Manager with the daemon used to handle session communication in TIBCO Rendezvous.
RV_NETWORK	Configures the iProcess Server Manager with the network used to handle outbound session communication in TIBCO Rendezvous.
RV_SERVICE	Configures the iProcess Server Manager with the User Datagram Protocol (UDP) service group used to handle session communication in TIBCO Rendezvous.

RV_DAEMON

TIBCO Rendezvous Configuration

Summary   This attribute is used to configure the iProcess Server Manager with the daemon used to handle session communication in TIBCO Rendezvous.

Applies To   This attribute must be set for ALL processes.

Permissible Values   If you are using the iProcess Server Manager, the setting of this process attribute must correspond to the daemon configuration parameter in TIBCO Rendezvous.

> By default, TIBCO Rendezvous uses the local daemon with the TCP socket number 7474. You do not need to change this attribute if your configuration uses this default port number.

If your TIBCO Rendezvous configuration does not use the default port number you must specify the TIBCO Rendezvous daemon being used. For more information about the daemon configuration parameter, see *TIBCO Hawk Installation and Configuration*.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	tcp:7474

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

See Also   RV_NETWORK, RV_SERVICE and Configuring the iProcess Server Manager on page 134

RV_NETWORK

TIBCO Rendezvous Configuration

Summary   This attribute is used to configure the iProcess Server Manager with the network used for outbound session communications in TIBCO Rendezvous.

Applies To   This attribute must be set for ALL processes.

Permissible Values   If you are using the iProcess Server Manager, the setting of this process attribute must correspond to the network configuration parameter in TIBCO Rendezvous.

> By default, TIBCO Rendezvous uses a null value for this parameter (indicated by a semi-colon or white space). You do not need to change this attribute if your configuration uses this default.

> If your TIBCO Rendezvous installation does not use the default configuration, you must ensure that the setting of this attribute matches the setting of the network configuration parameter in TIBCO Rendezvous. For more information about the network configuration parameter, see *TIBCO Hawk Installation and Configuration*.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	Null

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

See Also   RV_DAEMON, RV_SERVICE and Configuring the iProcess Server Manager on page 134

RV_SERVICE

TIBCO Rendezvous Configuration

Summary   This attribute is used to configure the iProcess Server Manager with the User Datagram Protocol (UDP) service group used for session communications in TIBCO Rendezvous.

Applies To   This attribute must be set for ALL processes.

Permissible Values   If you are using the iProcess Server Manager, the setting of this process attribute must correspond to the service configuration parameter in TIBCO Rendezvous.

> By default, TIBCO Rendezvous uses the service port number 7474. You do not need to change this attribute if your configuration uses this default port number.

If your TIBCO Rendezvous configuration does not use the default port number you must specify the service being used, either by its name or its port number. For more information about the service configuration parameter, see *TIBCO Hawk Installation and Configuration*.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	7474

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

See Also   RV_DAEMON, RV_NETWORK and Configuring the iProcess Server Manager on page 134

Case Prediction Configuration

The following process attributes allow you to configure the use of case prediction on iProcess Engine.

Attribute	Description
ENABLE_CASE_PREDICTION	Defines whether or not background case prediction is enabled on the node. **Note**: This attribute has no effect on live case prediction or case simulation.
MAX_PREDICTION_LOOPS	Defines the maximum number of times to loop during the prediction process.

ENABLE_CASE_PREDICTION

Case Prediction Configuration

Summary   This attribute defines whether or not the case prediction server process (BGPREDICT) is enabled for the iProcess system.

Applies To   This attribute can be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	BGPREDICT is disabled.
1	BGPREDICT is enabled.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	0

Notes   This attribute only affects background case prediction. It has no effect on live case prediction or case simulation.

For more information about the use of case prediction, see "Using Case Prediction to Forecast Outstanding Work Items" in *TIBCO iProcess Modeler Advanced Design*.

MAX_PREDICTION_LOOPS

Case Prediction Configuration

Summary   This attribute defines the maximum number of times to loop during the prediction process. An error is reported if this value is exceeded - this prevents infinite loops occurring as a result of loops in the procedure.

Applies To   This attribute applies to the BGPREDICT process.

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BGPREDICT	0	500

TIBCO iProcess Workspace (Windows) Configuration

The following process attributes allow you to configure aspects of TIBCO iProcess Workspace (Windows) behavior.

Attribute	Description
CSTART_AUTO_REFRESH	Defines whether or not the list of available procedures in the TIBCO iProcess Workspace's Case Start dialog is automatically refreshed.
DISABLE_CASE_COUNTING	Defines whether case counts are displayed for procedures in the Live (Dead) Cases column of the Case Administrator dialog, when a user starts iProcess Administrator from iProcess Workspace (Windows)
DISABLE_USER_CHECK	Defines whether or not a new user name is validated as an O/S user account when you add an iProcess user from the User Manager tool of TIBCO iProcess Administrator.
DISABLE_USER_LIST	Defines whether or not the Possible iProcess User List button is displayed in the User Manager tool of TIBCO iProcess Administrator.
OS_USER_LOCATIONS	Defines where iProcess Engine should obtain the list of users when it populates the Possible iProcess User List in the User Manager tool of TIBCO iProcess Administrator.
RPC_BLOCK	Defines whether or not iProcess Workspace (Windows) is able to access iProcess Engine.

CSTART_AUTO_REFRESH

TIBCO iProcess Workspace (Windows) Configuration

Summary This attribute defines whether or not the list of available procedures in the TIBCO iProcess Workspace's Case Start dialog is automatically refreshed.

Applies To This attribute can be set for ALL processes.

Permissible Values The attribute value must be one of the following:

Value	Meaning
0	The procedure list in the Case Start dialog is not automatically refreshed when the dialog is opened. The user must click the **Refresh** button to update the procedure list.
1	The procedure list in the Case Start dialog is automatically refreshed when the dialog is opened.

Default Value This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	1

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes When automatic refresh is enabled, the dialog is refreshed when it is opened. This ensures that the list of available procedures and versions shown to the user is accurate.

However, you can disable automatic refresh if you wish. You may want to do this if you have very large numbers of procedures, so that the refresh takes a noticeable time.

DISABLE_CASE_COUNTING

TIBCO iProcess Workspace (Windows) Configuration

Summary   This attribute defines whether case counts are displayed for procedures in the Live (Dead) Cases column of the Case Administrator dialog, when a user starts iProcess Administrator from iProcess Workspace (Windows).

Applies To   This attribute can be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	The Live (Dead) Cases column is populated when the Case Administrator dialog loads.
1	The Live (Dead) Cases column is not populated when the Case Administrator dialog loads. This improves the dialog's loading time.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   Normally, procedures are only displayed in the Case Administrator dialog if they have Case Administration access, have started cases, and the user is logged in to as either the procedure owner or an Administrator. However, when DISABLE_CASE_COUNTING=1, all procedures on the system are displayed in the Case Administrator dialog.

DISABLE_USER_CHECK

TIBCO iProcess Workspace (Windows) Configuration

> This attribute has no effect if you are validating iProcess users against an external validation package rather than against the O/S. See Specifying How iProcess Validates Users on page 32.

Summary   This attribute defines whether or not a new user name is validated as an O/S user account when you add an iProcess user (from the User Manager tool of TIBCO iProcess Administrator).

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	When you use User Manager to add a new user, the iProcess Suite checks if the username is a valid O/S user account. If it is not, the user is not created and an "Invalid User" error is displayed.
1	When you use User Manager to add a new user, the iProcess Suite does not check if the username is a valid O/S user account. The user is created even if it is not a valid O/S user account.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

See Also   DISABLE_USER_LIST

DISABLE_USER_LIST

TIBCO iProcess Workspace (Windows) Configuration

Summary   This attribute defines whether or not the Possible iProcess User List button is displayed in the User Manager tool of TIBCO iProcess Administrator.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	The Possible iProcess User List button is displayed in User Manager.
1	The Possible iProcess User List button is not displayed in User Manager. You should use this setting if you want to prevent users from accessing the list of valid O/S users.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   By default, the iProcess Suite requires that an iProcess user is also a valid O/S user account. When you add a user in the User Manager, click the **Possible iProcess User List** button to display a list of valid O/S accounts, and thus choose a user name that you know will be valid as an iProcess user name.

However, if this model does not meet your security requirements, you can use the TIBCO iProcess User Validation API to create your own user validation method that matches your business requirements. You may, for example, want to maintain the list of users (and their passwords) in a separate database, separating them entirely from O/S accounts. In this case, there is no requirement to display a list of O/S accounts in the User Manager. Indeed, for security reasons, you can choose not to display the list.

See Also   DISABLE_USER_CHECK

OS_USER_LOCATIONS

TIBCO iProcess Workspace (Windows) Configuration

Summary This attribute defines where iProcess Engine should obtain the list of users when it populates the Possible iProcess User List in the User Manager tool of TIBCO iProcess Administrator.

> This attribute is only used on the Windows variant of iProcess Engine. It has no effect if it is set on a UNIX system.

Applies To This attribute must be set for ALL processes.

Permissible Values The attribute value must be a text string of 1024 characters or less, which consists of a comma-delimited list of machine and/or domain names in the following format:

> "*machine*[,*machine*[,M:*machine*][,D:*domain*]...]"

Each name in the list can be explicitly identified as either:

- a *machine*, by using the M: prefix.

- a *domain*, by using the D: prefix. A domain name can be specified either as a simple name (for example, EMEA), or as a fully qualified domain name (for example, xyzCorp.dev.EMEA).

A name is treated as a machine name in the absence of either prefix.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	See below

The default value is:

> *machine*, *user*

where:

- *machine* is the name of the computer hosting this iProcess Engine node installation.

- *user* is the location of the user that ran the Setup program to install this iProcess Engine node. (If *machine* is a standalone computer, rather than a member of a domain, *user* is blank.)

For example, if iProcess Engine was installed on a computer called SERVER_TIB3 using the domain user account \\EMEA\AJones, the default value for this attribute would be:

```
SERVER_TIB3,D:EMEA
```

If SERVER_TIB3 was a standalone computer and AJones a local account, the default value would be:

```
SERVER_TIB3
```

The existing attribute value is preserved when you upgrade iProcess Engine.

Notes   When a user clicks the **Possible iProcess User List** button in the User Manager tool of TIBCO iProcess Administrator, the iProcess Engine node populates the displayed list of operating system (OS) logins with all the user names found in each location specified in this attribute value. User names are displayed in the format:

location\user

where *location* is the machine or domain specified in the OS_USER_LOCATIONS attribute, and *name* is the user name found in that location.

> You can use the `plist -U` command to display the list of OS users that will be generated by the current setting of the OS_USERS_LOCATION value.

If iProcess Engine is unable to contact a specified machine or domain for any reason, it writes an appropriate error message (with message ID 1631) to the sw_warn file. For example:

```
2006/11/30 14:07:47(plist:2784:2784:0:stevec:filosuvm.c:1.18:341):
1631-WARNING: <osuv_get_nxt_user ():
NetQueryDisplayInformation(dev1) failed: Access is denied.><> <> <>

2006/11/30 14:07:54(plist:2784:2784:0:stevec:filosuvm.c:1.18:341):
1631-WARNING: <osuv_get_nxt_user ():
NetQueryDisplayInformation(invalid) failed: The RPC server is
unavailable.> <> <> <>

2006/11/30 14:08:06(plist:2784:2784:0:stevec:filosuvm.c:1.18:341):
1631-WARNING: <priv_GetLocationMachineName ():
GetDomainController(ff) failed: The specified domain either does
not exist or could not be contacted.> <> <> <>
```

RPC_BLOCK

TIBCO iProcess Workspace (Windows) Configuration

Summary   This attribute defines whether or not iProcess Workspace (Windows) is able to access iProcess Engine.

Applies To   This attribute must be set for the RPC_TCP_LI processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	iProcess Workspace (Windows) is prevented from accessing iProcess Engine.
1	iProcess Workspace (Windows) is able to access iProcess Engine.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	RPC_TCP_LI	0	0

Notes   If the value of the RPC_BLOCK attribute is set to 1, the system will show you the following error message when you try to log in to TIBCO iProcess Workspace (Windows) or TIBCO iProcess Administrator:

```
L070: You are not allowed to access iProcess Engine with TIBCO
iProcess Workspace Windows. Please contact your iProcess Engine
Administrator.
```

See Also   None.

Procedure Configuration

The following process attributes allow you to configure how iProcess Engine handles iProcess procedures.

Attribute	Description
AUTO_PURGE_DELAY	Defines the number of days to delay the auto-purge operation.
DEF_MAJOR_VERS	Defines the default major version number that TIBCO iProcess Modeler will use when a new procedure is saved.
DEF_MINOR_VERS	Defines the default minor version number that TIBCO iProcess Modeler will use when a new procedure is saved.
FIL_PROCDEF_CACHE_SIZE	Defines the maximum number of procedure definitions that can be cached in memory by the BG, WIS and SPO processes.
MAX_SUB_PROCEDURE_DEPTH	Determines the maximum number of nested sub-procedures supported by the server.
RESTART_SPO_CACHE_PROC	Determines how many of the latest versions of the procedure definition to cache when the iProcess Objects Server process restarts or quick starts.
PROC_VER_COMMENT	Defines whether or not, in TIBCO iProcess Modeler, a user has to enter a comment whenever they save a procedure.
PROC_VER_INC	Defines whether or not, in TIBCO iProcess Modeler, a procedure's version number will be incremented whenever it is saved.
PROC_VER_NUM_INSTANCES	Defines the maximum number of instances of a procedure version.
SPO_CACHE_PROC	Determines how many of the latest versions of the procedure definition to cache when the iProcess Objects Server process starts normally.

AUTO_PURGE_DELAY

Procedure Configuration

Summary   This attribute defines the number of days to delay the auto-purge operation.

> This attribute is used when the Purge On field in the Deadline tab of the Properties dialog is set as a date field, and the value of the date field is not given.

Applies To   This attribute must be set for the BG processes.

Permissible Values   The attribute value must be a numeric value greater than or equal to 0.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	90

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   For more information about setting auto-purge delay, see "Setting Auto-Purge and Deadlines" in *TIBCO iProcess Modeler Procedure Management.*

DEF_MAJOR_VERS

Procedure Configuration

Summary	This attribute defines the default major version number that TIBCO iProcess Modeler will use when a new procedure is saved.
Applies To	This attribute must be set for ALL processes.
Permissible Values	The attribute value must be a numeric value greater than or equal to 0.
Default Value	This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes	For more information about the use of version numbering with procedures, see "Using Version Control" in *TIBCO iProcess Modeler Procedure Management.*
See Also	DEF_MINOR_VERS, PROC_VER_COMMENT, PROC_VER_INC

DEF_MINOR_VERS

Procedure Configuration

Summary   This attribute defines the default minor version number that TIBCO iProcess Modeler will use when a new procedure is saved.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be a numeric value greater than or equal to 0.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed.

Notes   For more information about the use of version numbering with procedures, see "Using Version Control" in *TIBCO iProcess Modeler Procedure Management.*

See Also   DEF_MAJOR_VERS, PROC_VER_COMMENT, PROC_VER_INC

FIL_PROCDEF_CACHE_SIZE

Procedure Configuration

Summary This attribute defines the maximum number of procedure definitions that can be cached in memory by the BG and WIS processes.

Applies To This attribute can be set for the WIS, BG, SPO or ALL processes.

Permissible Values The attribute value must be a numeric value greater than or equal to 1.

Default Value The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	ALL	0	64

Notes The in-memory procedure definition cache is used by the BG, WIS and SPO processes for rapid access to recently-used procedure definitions. When a BG, WIS or SPO process first accesses a procedure definition, the definition is fetched from the database and written to the cache. Subsequent accesses will use the definition from the cache rather than from the database, and so will be faster.

The BG process uses the procedure definition cache for all procedures that it processes. The WIS and SPO processes use it to filter queues that contain CDQP definitions.

This attribute defines the maximum number of procedure definitions that can be cached by the specified process. Increasing this value:

- increases the number of procedure definitions that can be rapidly accessed from the cache, but also increases the memory footprint of the process.

- can speed up work item filtering on large queues by the WIS or SPO processes.

Once the FIL_PROCDEF_CACHE_SIZE limit is reached for a process, if a new procedure definition needs to be added to the cache, the oldest procedure definition is removed. When this happens, the following message (with ID 1631) is written to the sw_warn file:

proc_name `has been bumped from the FIL procedure definition cache`

where *proc_name* is the name of the procedure definition that has been deleted from the cache. If this occurs you may want to increase the FIL_PROCDEF_CACHE_SIZE value.

The SPO process caches every procedure version of every procedure. This means that if your iProcess Engine has many procedures each of which has many procedure versions, the `FIL_PROCDEF_CACHE_SIZE` limit may easily be reached, causing a sw_warn file to be generated. To avoid this, you should reset the value of the `FIL_PROCDEF_CACHE _SIZE` to be (number of procedures) * (number of procedure versions).

The `sw_warn` file that is generated contains messages like the example below:

```
2007/04/05 08:42:39(SPO:1:2180:0:swadmin:\filpdcc.c::1253):
1631-WARNING: <'$EMAIL' has been bumped from the FIL procedure
definition cache> <> <> <>
```

The number and frequency of these messages indicates whether you need to amend the `FIL_PROCDEF_CACHE_SIZE` limit. For example, a couple of messages generated over a few minutes means there is no need to alter the `FIL_PROCDEF_CACHE_SIZE` value. However, lots of messages generated in a short space of time means the `FIL_PROCDEF_CACHE_SIZE` has been greatly exceeded.

You should be aware that if you do increase the `FIL_PROCDEF_CACHE_SIZE` value, the process uses more memory so you may reach the Operating System memory limit sooner.

MAX_SUB_PROCEDURE_DEPTH

Procedure Configuration

Summary   This attribute defines the maximum number of nested sub-procedures supported by iProcess Engine.

Applies To   This attribute can be set for the BG, RPCBG and BGPREDICT processes.

Default Value   The attribute is assigned the following default values when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	BG	0	100
0	BGPREDICT	0	100

RESTART_SPO_CACHE_PROC

Procedure Configuration

Summary   This attribute determines how many of the latest versions of the procedure definition to cache when the iProcess Objects Server process restarts or quick starts.

Applies To   This attribute can only be set for an iProcess Objects Server (SPO) process.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
-1	Caches all versions of each procedure.
n	Caches the latest *n* versions of each procedure (where *n* is an integer in the range 1 to 60000).

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	iProcess Objects Server (SPO)	0	1

See Also   SPO_CACHE_PROC

PROC_VER_COMMENT

Procedure Configuration

Summary   This attribute defines whether or not, in TIBCO iProcess Modeler, a user has to enter a comment whenever they save a procedure.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	Not supported. The comment field is not displayed in the Procedure > Save dialog(s) in TIBCO iProcess Modeler.
1	Optional. The comment field is displayed in the Procedure > Save dialog(s) in TIBCO iProcess Modeler. The user can leave it blank if desired.
2	Required. The comment field is displayed in the Procedure > Save dialog(s) in TIBCO iProcess Modeler. The user must fill it in before they can save the procedure.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	1

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   For more information about the use of version numbering with procedures, see "Using Version Control" in *TIBCO iProcess Modeler Procedure Management*.

See Also   DEF_MAJOR_VERS, DEF_MINOR_VERS, PROC_VER_INC, PROC_VER_NUM_INSTANCES

PROC_VER_INC

Procedure Configuration

Summary   This attribute defines whether or not, in TIBCO iProcess Modeler, a procedure's version number will be incremented whenever it is saved.

Applies To   This attribute must be set for ALL processes.

Permissible Values   The attribute value must be one of the following:

Value	Meaning
0	The version number will be incremented only when a new version of the procedure is explicitly created.
1	The version number will be incremented every time the procedure is saved.

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   For more information about the use of version numbering with procedures, see "Using Version Control" in *TIBCO iProcess Modeler Procedure Management*.

See Also   DEF_MAJOR_VERS, DEF_MINOR_VERS, PROC_VER_COMMENT, PROC_VER_NUM_INSTANCES

PROC_VER_NUM_INSTANCES

Procedure Configuration

Summary	This attribute defines how many old instances of a procedure are kept in the iProcess database. The most recent instance of a procedure version is always kept.
Applies To	This attribute must be set for ALL processes.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
0	There is no limit to the number of instances of a procedure that are kept. This is the default value.
n	*n* number of instances of a procedure will be kept in the iProcess database (where *n* is a positive integer).

Default Value   This attribute is not defined on a newly installed iProcess Engine node. In this case, the default value is:

Machine ID	Process	Instance	Value
0	ALL	0	0

The attribute only appears in the output of the SHOW_ALL_ATTRIBUTES command if you have explicitly assigned a value to it using the SET_ATTRIBUTE command. The default setting is not displayed

Notes   Every time you edit and save a version of a procedure, iProcess creates a new instance of that procedure version.

The PROC_VER_NUM_INSTANCES attribute applies to all old instances of a procedure. Each procedure instance is allocated an instance identifier. Each time a new instance is created the instance identifier is incremented by one.

The instances of a procedure are tidied up as when a procedure is saved. This is because a tidy operation is performed each time a procedure is saved which tidies up the number of instances according to the attribute value you have set.

The first time you set the attribute you can run the swadm tidy_instances command to force a tidy operation to tidy up the number of instances of some or all of your procedures, depending on your requirements. To do this you need to run the swadm tidy_instances command. See Tidy Instances of Procedures on page 362 for more information.

For more information about the use of version numbering with procedures, see "Using Version Control" in *TIBCO iProcess Modeler Procedure Management*.

See Also   DEF_MAJOR_VERS, DEF_MINOR_VERS, PROC_VER_COMMENT, PROC_VER_INC

SPO_CACHE_PROC

Procedure Configuration

Summary	This attribute determines how many of the latest versions of the procedure definition to cache when the iProcess Objects Server process starts normally.
Applies To	This attribute can only be set for an iProcess Objects Server (SPO) process.
Permissible Values	The attribute value must be one of the following:

Value	Meaning
-1	Caches all versions of each procedure.
n	Caches the latest *n* versions of each procedure (where *n* is an integer in the range 1 to 60000).

Default Value   The attribute is assigned the following default value when iProcess Engine is installed.

Machine ID	Process	Instance	Value
0	iProcess Objects Server (SPO)	0	-1

See Also   RESTART_SPO_CACHE_PROC

iProcess Objects Director

Process attributes that are used by the DIRECTOR process are not documented in this guide. For more information about attributes that are used by the DIRECTOR process, see *TIBCO iProcess Objects Director Administrator's Guide*.

Chapter 7 **Administering Message Queues and Mbox Sets**

This chapter describes how to use the swadm server configuration utility to administer Mbox sets, message queues and message instructions.

See "iProcess Mbox Sets" in *TIBCO iProcess Engine Architecture Guide* for more information about how iProcess Engine uses Mbox sets, message queues and messages.

Topics

Introduction

There are two utilities that you can use to administer iProcess message queues:

- the swadm utility, which you can use to directly administer Mbox sets, queues and messages. See Using SWDIR\util\swadm to Administer Mbox Sets, Message Queues and Messages on page 329 for more information.

- the iProcess Server Manager, which provides a graphical view of message queues. You can use it to administer single messages or queues. See Using the iProcess Server Manager to Administer Message Queues on page 350 for more information.

You can administer the cases or work items' priorities for internal message queues when passing messages between iProcess processes such as from the background process and the WISes, or from SSOLite to the BG processes. See "Where can Priorities be Used" in *TIBCO iProcess Modeler Advanced Design* for more information.

Using SWDIR\util\swadm to Administer Mbox Sets, Message Queues and Messages

You can use the swadm utility, which is located in the *SWDIR*\util directory, to administer (view, add, delete and modify) Mbox sets, queues and messages. Note that:

- To use this utility, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as a background use or root user.

- If you are using a node cluster architecture, you can run this utility from any server within the cluster (as long as that server has a connection to the TIBCO iProcess Engine database instance).

The following table summarizes the commands you can use to administer Mbox sets, message queues and messages.

Area	Task	swadm Command
Mbox sets	Show Mbox Sets	show_mboxsets
	Add an Mbox Set	add_mboxset
	Add a Message Queue to an Mbox Set	add_queue_to_mboxset
	Delete a Message Queue From an Mbox Set	delete_queue_from_mboxset
	Rename an Mbox Set	update_mboxset
	Delete an Mbox Set	delete_mboxset
Queues	Show Message Queues	show_queues
	Add a Message Queue	add_queue
	Update a Message Queue	update_queue
	Delete a Message Queue	delete_queue
Messages	Show Messages in a Queue	show_messages
	Count Messages in a Queue	count_messages
	Show Details of a Message	detail_message
	Restore Dead Messages to a Queue	restore_dead_messages
	Delete Messages From a Queue	delete_messages

These commands read and update the mbox_set, mbox_set_group, and iql_queues database tables.

Show Mbox Sets

To display a list of current Mbox sets defined on iProcess Engine, use the following command:

```
swadm show_mboxsets [v]
```

The command lists the following information for each Mbox set:

- `Mboxset ID` is the unique identifier for the Mboxset, assigned when the Mbox set is created.

- `Mboxset Name` is the descriptive name of the Mbox set.

- `Queue Type` identifies the type of messages held in queues in the Mbox set. This will be Local (for local messages).

If the `v` option has been specified, the following information is also displayed:

- Queues in MBOX Set lists the queues that belong to the Mbox set. Queues are listed by their unique queue identifier. (You can use the `show_queues` command to find out more about each queue.)

The following example shows the use of the basic `show_mboxsets` command.

```
swadm show_mboxsets
```

Mboxset ID	Mboxset Name	Queue Type
1	BGMBSET	Local
2	WMDMBSET	Local
3	PREDICTMBSET	Local

The following example shows the use of the `show_mboxsets v` command.

```
swadm show_mboxsets v
```

Mboxset ID	Mboxset Name	Queue Type	Queues in MBOX Set
1	BGMBSET	Local	1,2
2	WMDMBSET	Local	3,4
3	PREDICTMBSET	Local	6,7

Add an Mbox Set

To add a new Mbox set to iProcess Engine, use the following command:

```
swadm add_mboxset mboxset_name message_type
```

where:

- *mboxset_name* is the name of the new Mbox set (up to a maximum of 32 characters). You can use this to identify what the Mbox set is used for, for example, processing Background messages.

- *message_type* is used to identify the message type. This should be Local (for local messages).

The following example adds a new Mbox set BGMBSET2 to iProcess Engine.

```
swadm add_mboxset BGMBSET2 Local
```

Add a Message Queue to an Mbox Set

You can add or remove message queues to Mbox sets at any time to alter the Mbox set configuration. For example, you might want to increase the number of queues in an Mbox to handle a larger volume of messages.

You must explicitly create a queue before you can add it to an Mbox set. You can do this using the `add_queue` command.

To add a queue to an Mbox set, use the following command:

```
swadm add_queue_to_mboxset mboxset_id queue_id
```

where:

- *mboxset_id* is the unique identifier for the Mbox set. You can find an Mbox set's identifier using the `show_mboxsets` command.

- *queue_id* is the unique identifier of the queue you want to add. You can find a queue's identifier using the `show_queues` command.

The following example adds the queue `BGMBOX3` to the `BGMBSET` Mbox set. (The `show_mboxsets` command is used first to identify the `BGMBSET` Mbox set's *mboxset_id*, which is 1.)

```
swadm show_mboxsets
Mboxset ID              Mboxset Name            Queue Type
1                       BGMBSET                 Local
2                       WMDMBSET                Local
3                       PREDICTMBSET            Local


swadm add_queue_to_mboxset 1 BGMBOX3
```

Delete a Message Queue From an Mbox Set

To remove a queue from an Mbox set, use the following command:

    swadm delete_queue_from_mboxset *mboxset_id queue_id*

where:

- *mboxset_id* is the unique identifier for the Mbox set. You can find an Mbox set's identifier using the `show_mboxsets` command.

- *queue_id* is the unique identifier for the queue you want to delete. You can find a queue's identifier using the `show_queues` command.

The following example deletes the queue BGMBOX3 from the BGMBSET Mbox set. (The `show_mboxsets` command is used first to identify the BGMBSET Mbox set's *mboxset_id*, which is 1.)

```
swadm show_mboxsets
Mboxset ID              Mboxset Name              Queue Type
1                       BGMBSET                   Local
2                       WMDMBSET                  Local
3                       PREDICTMBSET              Local


swadm delete_queue_from_mboxset 1 BGMBOX3
```

Rename an Mbox Set

To change the name of an Mbox set, use the following command:

swadm update_mboxset *mboxset_id* *new_name*

where:

- *mboxset_id* is the unique identifier for the Mbox set. You can find an Mbox set's identifier using the show_mboxsets command.

- *new_name* is the new name for this Mbox set (up to a maximum of 32 characters).

The following example renames the BGMBSET2 Mbox set as BGMBSET3. (The show_mboxsets command is used first to identify the BGMBSET2 Mbox set's *mboxset_id*, which is 4.)

```
swadm show_mboxsets
Mboxset ID              Mboxset Name            Queue Type
1                       BGMBSET                 Local
2                       WMDMBSET                Local
3                       PREDICTMBSET            Local
4                       BGMBSET2                Local


swadm update_mboxset 4 BGMBSET3
```

Delete an Mbox Set

To delete an Mbox set, use the following command:

swadm delete_mboxset *mboxset_id*

where *mboxset_id* is the unique identifier of the Mbox set. You can find an Mbox set's identifier using the show_mboxsets command.

Any queues contained in the Mbox set are not affected by this command. If you also want to delete the queues you must use the delete_queue command after you have deleted the Mbox set.

The following example deletes the PREDICTMBSET Mbox set. (The show_mboxsets command is used first to identify the PREDICTMBSET Mbox set's *mboxset_id*, which is 3.)

```
swadm show_mboxsets
Mboxset ID              Mboxset Name            Queue Type
1                       BGMBSET                 Local
2                       WMDMBSET                Local
3                       PREDICTMBSET            Local


swadm delete_mboxset 3
```

Show Message Queues

To display a list of all the message queues currently set up on your system and view their queue names and identifiers, use the following command:

    swadm show_queues [*queue_name*]

where *queue_name* is the optional name of a queue, which you can use to only display queues matching this name.

The command lists the following information for each queue:

- `Queue ID` is the unique identifier for the queue, assigned when the queue is created.

- `Queue Name` is the descriptive name of the queue.

- `Queue Type` identifies the type of messages held in the queue. This will be Local (for local messages).

- `Queues Desc` specifies the physical database table that is used to hold the queue. See the `add_queue` command for a full description of the format of this entry.

The following example lists all the queues currently defined on iProcess Engine (Windows version).

```
swadm show_queues
Queue ID    Queue Name     Queue Type     Queue Desc

1           BGMBOX1        Local          0003:swpro.sw_db_bgqueue_1
2           BGMBOX2        Local          0003:swpro.sw_db_bgqueue_2
3           WISMBOX1       Local          0003:swpro.sw_db_wisqueue_1
4           WISMBOX2       Local          0003:swpro.sw_db_wisqueue_2
5           DEADQUEUE      Local          0003:swpro.sw_db_deadqueue
6           PREDICTMBOX1   Local          0003:swpro.sw_db_predictqueue_1
7           PREDICTMBOX2   Local          0003:swpro.sw_db_predictqueue_2
```

Add a Message Queue

When adding queues, you have to add:

1. a queue

2. an Mbox set

3. the queue to the Mbox set.

To set up a new queue on your system (so that you can then add it to an Mbox set), use the following command:

```
swadm add_queue queue_name message_type queue_description
```

where:

- *queue_name* is a descriptive alphanumeric name for the queue.

- *message_type* is used to identify the message type. This should be Local (for local messages).

- *queue_description* specifies the physical database table that is used to hold the queue, in the following format:

  version: *table*

  where:

  — *version* is an internal number used by iProcess Suite to identify the physical syntax of the string that follows it. This should be either:

    - 0001, for Oracle AQ.

    - 0003, for queues that are held in the iProcess database.

  — *table* is the name of the database table that holds the queue, and must be specified in the format needed to access the table (e.g. MS-SQL, DB2 or Oracle AQ).

    The database table used to hold the queue must already exist, and must conform to the appropriate format. If it does not, messages will not be able to be added to or read from the queue and iProcess Engine will not function correctly. For more information, see:

  — "Oracle AQ Queue Tables and Queues" in *TIBCO iProcess Engine (Oracle) Administrator's Guide* (for Oracle AQ tables).

  — "iProcess DB2 Database Queues" in *TIBCO iProcess Engine (DB2)s Administrator's Guide* (for iProcess database tables in a DB2 database).

- "iProcess SQL Server Database Queues" in *TIBCO iProcess Engine (SQL): Administrator's Guide* (for iProcess database tables in a SQL Server database).

If you are using queues held in the iProcess database (*version = 0003*), you should note that:

- Each individual queue used by iProcess Engine must be held in its own database table. These tables must be held in either:

  — the database being used by iProcess Suite (the default option).

  — a different database on the same database server.

- Wherever the tables are held, the following permissions must be set up:

  — The iProcess Engine database schema owner (default `swpro`) must have at least insert, select and delete permissions on the database table used to hold the queue.

  — The iProcess Engine database user (default `swuser`) must have at least insert permissions on the database table used to hold the queue.

Examples

1. This example (for Windows/SQL Server) adds a queue called BGMBOX3. This queue is the physical queue `sw_db_wisqueue3`, owned by `swpro`, in the current iProcess database.

   `swadm add_queue BGMBOX3 Local 0003:swpro.sw_db_wisqueue3`

2. This example (for Windows/SQL Server) adds a queue called BGMBOX4. This queue is the physical queue sw_db_bgqueue4 owned by user bart, in the sw database (on the SQL server hosting the iProcess database).

   `swadm add_queue BGMBOX4 Local 0003:sw.bart.sw_db_bgqueue4`

Update a Message Queue

You can change the queue name, message type and/or queue description using the following command:

```
swadm update_queue queue_id | queue_name new_name message_type
queue_description
```

where:

- *queue_id* is the unique identifier for the queue. You can find a queue's identifier using the show_queues command.

- *queue_name* is the descriptive alphanumeric name for the queue.

- *new_name* is the new name to be used for this queue. If you want to leave the existing name unchanged, use a hyphen '-'.

- *message_type* is used to identify the message type. This value must be either Local (for local messages), or a hyphen '-' (to leave the value unchanged).

- *queue_description* specifies the physical database table that holds the queue. If you want to leave the existing name unchanged, use a hyphen '-'.

  See the description of this parameter under the add_queue command for a full description of the syntax and requirements for this parameter.

Examples

1. This example (for Windows/SQL Server) points the queue BGMBOX3 to use a different physical queue, sw_db_bgqueue5 owned by swpro (in the current database used by iProcess Engine). The queue's current name and message type are left unchanged.
   ```
   swadm update_queue BGMBOX3 - - 0003:swpro.sw_db_bgqueue5
   ```

2. This example renames the queue BGMBOX3 to BGMBOX5. The queue will continue to use its existing message type and physical queue.
   ```
   swadm update_queue BGMBOX3 BGMBOX5 - -
   ```

Delete a Message Queue

Before deleting a queue you should remove it from the Mbox set, using the delete_queue_from_mboxset command.

To delete a queue, use the following command:

```
delete_queue queue_id | queue_name
```

where:

- *queue_id* is the unique identifier for the queue. You can find a queue's identifier using the `show_queues` command.

- *queue_name* is the descriptive alphanumeric name for the queue.

`swadm` displays a warning message if you have not already removed the queue from the Mbox set.

The following example deletes the BGMBOX3 queue.

```
swadm delete_queue BGMBOX3
```

Show Messages in a Queue

To display a summary list of all the iProcess messages that are currently in a queue, use the following command:

```
swadm show_messages queue_id|ALL [EXP] [-x]
```

where:

- *queue_id* is the unique identifier of the queue you want to view messages for. You can find a queue's identifier using the `show_queues` command. Enter ALL to show the messages in all queues.

- EXP is an optional parameter used with Oracle databases only. Use it to show all the Oracle exception messages. If this parameter is not specified, show_messages when used on an Oracle system will only list normal messages, with other types of database it will show all the messages.

- *-x* is an optional parameter that changes the format in which messages are displayed. If this parameter is specified, one message is displayed per line and fields in the message are separated by '|' characters.

The following example (for Windows/SQL Server) lists all the messages in the DEADQUEUE queue. (The show_queues command is used first to identify the DEADQUEUE's *queue_id*, which is 5.) In this case the DEADQUEUE contains just a single RELEASE instruction that has failed to be processed.

```
swadm_show_queues

Queue ID    Queue Name      Queue Type      Queue Desc


1           BGMBOX1         Local           0003:swpro.sw_db_bgqueue_1
2           BGMBOX2         Local           0003:swpro.sw_db_bgqueue_2
3           WISMBOX1        Local           0003:swpro.sw_db_wisqueue_1
4           WISMBOX2        Local           0003:swpro.sw_db_wisqueue_2
5           DEADQUEUE       Local           0003:swpro.sw_db_deadqueue


swadm show_messages 5

Queue ID    5

            Message ID:     291E84BA-A898-4D6A-A812-A76BE108B21D
            Instruction:    RELEASE
            Addressee:      pro
            Procedure:      TESTBW2
            Step Name:      STEP1
            Case Number:    1253
            Req ID:         2504
            Failed Count:   0
            Failed By:      BG
```

The following command displays the same messages in the alternative format:

```
swadm              5|291E84BA-A898-4D6A-A812-A76BE108B21D|RELEASE|pro|TESTBW2|STEP1|
show_messages 5    1253|2504|0|BG
-x
```

Count Messages in a Queue

To display the total count of all the iProcess messages that are currently in a given queue, use the following command:

```
swadm count_messages queue_id|ALL [EXP]
```

where:

- *queue_id* is the unique identifier of the queue you want to count messages for. You can find a queue's identifier using the show_queues command. Enter ALL to count the messages in all queues.

- EXP is an optional parameter used with Oracle databases only. Use it to count all the Oracle exception messages. If this parameter is not specified, count _messages when used on an Oracle system will only include normal messages, with other types of database it will count all the messages.

The following example counts the messages in queue 5:

```
swadm count_messages 5
The total count of the messages in the queue 5: 6
```

The following example counts the messages in all queues:

```
swadm count_messages all
The total count of the messages in the queue 1: 0
The total count of the messages in the queue 2: 0
The total count of the messages in the queue 3: 0
The total count of the messages in the queue 4: 0
The total count of the messages in the queue 5: 6
The total count of the messages in the queue 6: 0
The total count of the messages in the queue 7: 0
```

Show Details of a Message

To display the header and body information of a message, use the following command:

```
swadm detail_message queue_id message_id [-x]
```

where

- *queue_id* is the unique identifier of the queue containing the message for which you want to display details. You can find a queue's identifier using the `show_queues` command.

- *message_id* is the identifier of a specific message in the queue *queue_id*. This ID is part of the information displayed by swadm show_messages - see .

- *-x* is an optional parameter that changes the format in which messages are displayed. If this parameter is specified, the message is displayed on one line and fields in the message are separated by '|' characters.

The following example shows the detail of one message in queue 5:

```
swadm detail_message 5 291E84BA-A898-4D6A-A812-A76BE108B21D

Queue ID    5

            Message ID:    291E84BA-A898-4D6A-A812-A76BE108B21D
            Instruction:   RELEASE
            Procedure      16:0:0
            Number:        1253
            Case Number:   swpro
            User Name:     STEP1
            Step Name:
            Step Desc:     13/03/2008
            Release Date:  09:07
            Release Time:  swpro
            Addressee      RELEASE^staffw_107^16:0.0^1^staffw_107,swpro,
            User:          2504,staffw_107^1253^1253^staffw_107^swpro^ST
            Message        EP1^^3^13/03/2008^09:07^2504^staffw_107^swpro
                           ^1^
```

The following example shows the same message using the alternative display option:

```
swadm detail_message 5 291E84BA-A898-4D6A-A812-A76BE108B21D -x
5|291E84BA-A898-4D6A-A812-A76BE108B21D|RELEASE|16:0.0|1253|swpro|S
TEP1||13/03/2008|09:07|swpro|RELEASE^staffw_107^16:0.0^1^staffw_10
7,swpro,2504,staffw_107^1253^1253^staffw_107^swpro^STEP1^^3^13/03/
2008^09:07^2504^staffw_107^swpro^1^
```

Restore Dead Messages to a Queue

To restore one or more dead iProcess messages to the queue from which they came, and make the messages live again, use the following command:

```
swadm restore_dead_messages to_queue_id ALL
[BG|WISMBD|BGPREDICT]|message_id|-f file_name
```

where:

- *to_queue_id* is the queue identifier of the destination queue, the queue to which you want to restore messages.

- The usage of the parameter `ALL [BG|WISMBD|BGPREDICT]` varies according to database type:

  - In SQL Server, there is a specific dead queue that holds all the dead messages that have been failed from any of the live queues. Messages in this dead queue have a unique additional field, called FAILED_BY, which identifies the type of queue from which the dead message comes. This field has one of three values:

  - `BG`: the dead message comes from a BG queue

  - `WISMBD`: the dead message comes from a WISMBD queue

  - `BGPREDICT`: the dead message comes from a BGPREDICT queue

    Because there is more than one BG, WIS or BGPREDICT queue in the system, you must tell this command what type of dead message you want to move from the dead queue (one of BG, WIS or BGPREDICT), and the number of the live queue to which the message is to be restored. For example, the command:

    `RESTORE_DEAD_MESSAGES 1 ALL BG`

    moves all dead messages failed from the BG queue from the dead queue to queue 1.

  - In DB2 usage is similar because the DB2 version has the same database structure as SQL Server, but there is one difference. The FAILED_BY field contains the name of the actual queue from which the dead message was failed, instead of a queue type. Neither this queue name nor the queue type needs to be specified in the command line. For example, the command:

    `RESTORE_DEAD_MESSAGES 1 ALL`

    moves all the dead messages failed by queue 1 from the dead queue to queue 1.

  - In Oracle, there is no specific dead queue. If a message in any given queue is failed, it is stored in that same queue and is just marked as a dead

message. This command therefore does not actually move messages from one queue to another, it merely changes them from dead messages to live messages within the same queue. It is therefore only necessary to specify in the command which queue you want to restore. For example, the command:

```
RESTORE_DEAD_MESSAGES 1 ALL
```

restores all the dead messages in queue 1 to being live messages in queue 1.

- *message_id* specifies the ID of the message that you want to restore. The following example restores an exception message to queue 1:

```
swadm restore_dead_messages 1
291E84BA-A898-4D6A-A812-A76BE108B21D
```

- -f *file_name* identifies a text file that contains one or more message IDs, separated from each other by a return character. If the beginning of a line has a semi-colon (;) character, that line is treated as a comment and all the content in the line is omitted.

Delete Messages From a Queue

To delete one or more iProcess messages from a queue, use the following command:

```
swadm delete_messages queue_id ALL [EXP]| message_id|-f file_name
```

where

- *queue_id* is the unique identifier of the queue containing the messages which you wish to delete. You can find a queue's identifier using the `show_queues` command. Enter ALL to delete the messages in all queues.

- EXP is an optional parameter used with Oracle databases only. Use it to delete all the Oracle exception messages in the queue. If this parameter is not specified, only normal messages will be deleted on an Oracle system, with other types of database, all types of message are deleted.

- *message_id* specifies the ID of the message that you want to delete. The following example deletes an exception message from queue 1:

  ```
  swadm delete_messages 1 291E84BA-A898-4D6A-A812-A76BE108B21D
  ```

- `-f` *file_name* identifies a text file that contains one or more message IDs for deletion. The file must be located in the *SWDIR*/util directory. Message IDs are separated from each other by a return character. If the beginning of a line has a semi-colon (;) character, that line is treated as a comment and all the content in the line is omitted.

Example   The following command deletes the specified message from queue 5:

```
swadm delete_messages 5 AB87DAEF-CEAD-4EC2-A44B-6F5DF716E4D6
```

Using the iProcess Server Manager to Administer Message Queues

The iProcess Server Manager is a JSP web client application that utilizes TIBCO Hawk to provide a graphical view of the messages and message queues on a machine or a node cluster.

> You can also use the iProcess Server Manager to administer server processes. See Using the iProcess Server Manager to Administer Server Processes on page 130 for details of TIBCO Hawk requirements, and of how to set up and start the iProcess Server Manager.

Controlling Message Queues

To view the Message Queue Management pane, expand **iProcess Management > Queue Management**.

The iProcess Management page shows information for the server you selected on the Configuration pane - see Configuring the iProcess Server Manager on page 134. The view is hierarchical, so expand a server or a node in a cluster to show individual message queues running on each.

Alerts

The iProcess Server Manager sends warning messages when:

- A message is retried a configurable number of times, or more. This value defaults to three times.

- Any queue contains more than 50 messages.

- A message is put in the exception queue.

Default Message Handling Configuration

This section describes the message handling configuration that is used on a default iProcess Engine installation.

Default Mbox Sets

The following table shows the default Mbox sets that are created when iProcess Engine is installed. See Show Mbox Sets on page 331 for an explanation of the Mboxset ID, Mboxset Name and Queues in Mboxset columns.

Mboxset ID	Mboxset Name	Queues in Mboxset
1	BGMBSET	BGMBOX1, BGMBOX2
2	WMDMBSET	WISMBOX1, WISMBOX2
3	WISBGMBSET1	BGMBOX1
4	WISBGMBSET2	BGMBOX2
5	PREDICTMBSET	PREDICTMBOX1, PREDICTMBOX2

Default Message Queues

The following sections describe the default message queues that are created when iProcess Engine is installed on a Windows or UNIX system. See Add a Message Queue on page 338 for an explanation of the Queue Name and Queue Description columns.
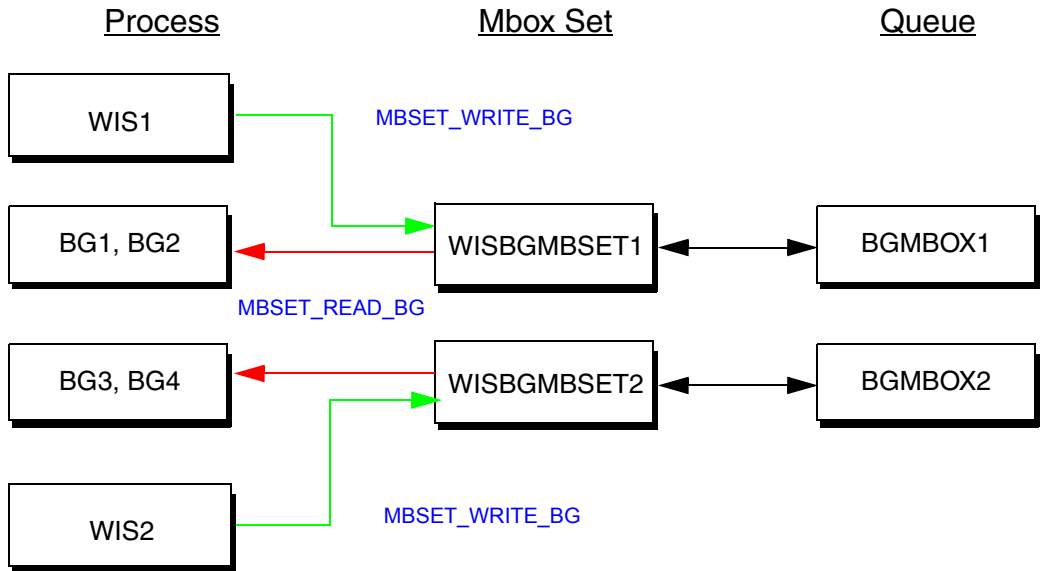
Windows/SQL Server or UNIX/DB2

The following tables shows the default message queues that are created when iProcess Engine is installed on a Windows/SQL Server or UNIX/DB2 system.

Queue Name	Queue Description
BGMBOX1	0003:swpro.sw_db_bgqueue_1
BGMBOX2	0003:swpro.sw_db_bgqueue_2
WISMBOX1	0003:swpro.sw_db_wisqueue_1
WISMBOX2	0003:swpro.sw_db_wisqueue_2
DEADQUEUE	0003:swpro.sw_db_deadqueue
PREDICTMBOX1	0003:swpro.sw_db_predictqueue_1
PREDICTMBOX2	0003:swpro.sw_db_predictqueue_2

Each individual queue used by iProcess Engine must be held in its own database table. These tables exist by default in the same database as the other iProcess tables, but they do not have to be held there. See Add a Message Queue on page 338 for more information.

UNIX/Oracle or Windows/Oracle

The following table shows the default Oracle AQ message queues that are created when iProcess Engine is installed on a UNIX/Oracle or Windows/Oracle system.

Queue Name	Parameters
BGMBOX1	0001::bgmboxtable1:bgmboxqueue1
BGMBOX2	0001::bgmboxtable2:bgmboxqueue2
WISMBOX1	0001::wismboxtable1:wismboxqueue1
WISMBOX2	0001::wismboxtable2:wismboxqueue2
PREDICTMBOX1	0001::predictmboxtable1:predictmboxqueue1
PREDICTMBOX2	0001::predictmboxtable2:predictmboxqueue2

How WIS Processes Send Messages to BG Processes

The following diagram shows how the WIS processes send messages to the BG processes using the default configuration.

1. One WIS process is configured to write messages to each WISBGMBSET Mbox set.

2. Each WISBGMBSET Mbox set contains a single message queue, BGMBOX.

3. Two BG processes are configured to read messages from each WISBGMBSET Mbox set.
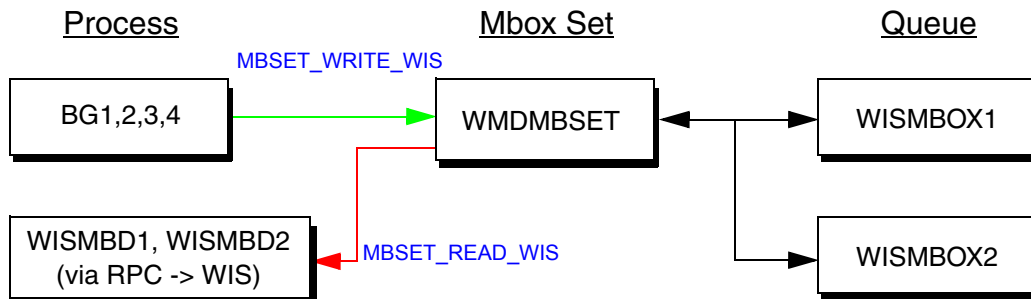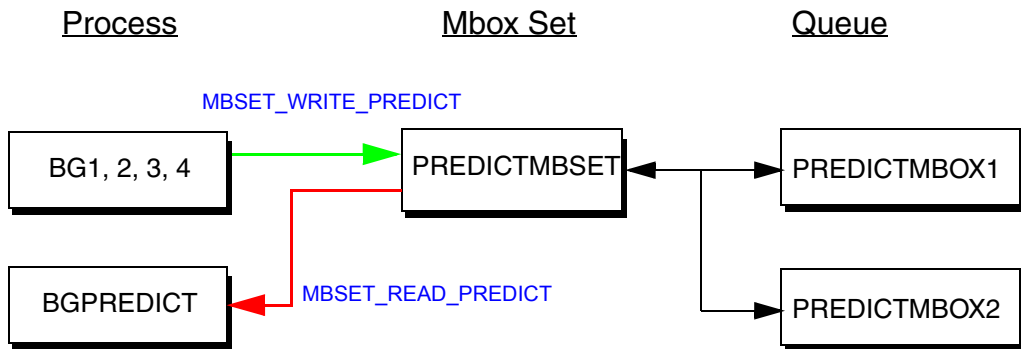
How non-WIS Processes Send Messages to BG Processes

The following diagram shows how non-WIS processes send messages to the BG processes using the default configuration.

1. All non-WIS processes (such as swbatch) are configured to write messages to the BGMBSET Mbox set.

2. The BGMBSET Mbox set contains two message queues, BGMBOX1 and BGMBOX2.

3. All BG processes are configured to read messages from the BGMBSET Mbox set.

How BG Processes Send Messages to WIS Processes

The following diagram shows how the BG processes send messages to the WIS processes using the default configuration.

1. All four BG processes are configured to write messages to a single Mbox set, WMDMBSET.

2. The WMDMBSET Mbox set contains two message queues, WISMBOX1 and WISMBOX2.

3. Both WISMBD processes are configured to read messages from the WMDMBSET Mbox set. (Each WISMBD process then forwards each message to the appropriate WIS process via RPC.)

How BG Processes Send Messages to the Prediction Process

The following diagram shows how the BG processes send messages to the BGPREDICT process using the default configuration.

1. All four BG processes write messages to the PREDICTMBSET Mbox set.

2. The PREDICTMBSET Mbox set contains two Mboxes, PREDICTMBOX1 and PREDICTMBOX2.

3. The BGPREDICT process reads messages from the PREDICTMBSET Mbox set.

Chapter 8 **Administering Procedure Objects**

This chapter explains how to use the swadm server configuration utility, which is under the *SWDIR*\util directory to administer the procedures (including sub-procedures and sub-procedure parameter templates) and libraries that are defined on this iProcess Engine node.

To use this utility, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as a background user or root user.

If you are using a node cluster architecture, you can run this utility from any server within the cluster (as long as that server has a connection to the TIBCO iProcess Engine database instance).

These commands read and update data in the pm_objects, proc_index, proc_version, proc_instance, proc_audit, proc_defn, and proc_mgt_hierarchy database.

Topics

Show Procedures and Libraries

TIBCO recommends that you run this command if you have problems when importing procedures or procedure libraries.

To display a list of procedures and libraries that are defined on this iProcess Engine node, enter the following command:

```
swadm show_procedures [fix]
```

where `fix` is an optional parameter that you can use to fix any errors that are reported - for example, if a database record in the `pm_objects` table has become corrupt. (See Errors on page 359.)

Output

The list of procedures and libraries (objects) is displayed. The following information is displayed about each object:

(*type*) *ObjectName* – *ObjectGUID*

where:

- *type* is one of the following single characters that indicates what the object is:
    — `F` is a library.
    — `P` is a procedure.
    — `S` is a sub-procedure.
    — `T` is a sub-procedure parameter template.
- *ObjectName* is the name of the procedure or library.
- *ObjectGUID* is the unique identifier for this procedure or library.

The contents of libraries are indented to indicate their hierarchical relationship.

To identify a list of sub-procedure parameter templates, you can also use the *SWDIR*\util\plist -p command. In the Flag column returned by the command:

- the flag `T` indicates that the procedure is a sub-procedure parameter template.
- the flag `I` indicates that the procedure is a sub-procedure.

For more information about the `plist -p` command, see "Procedure Information" in Using the plist Utility to Obtain Information About TIBCO iProcess Engine Facilities on page 481.

Errors

ERROR messages are displayed if any errors are detected. If the `fix` parameter has been specified, and the error is one that can be fixed, a FIX message is displayed immediately after the ERROR message, indicating what has been done.

The following list shows the errors that can occur and be fixed, and the fixes that are applied if you specify the `fix` parameter:

```
ERROR: PM_OBJECTS ObjectName - ObjectGUID refers to proc_id: proc_id which does not exist
FIX: delete PM_OBJECTS record ObjectName - ObjectGUID

ERROR: PM_OBJECTS ObjectName - ObjectGUID refers to proc_id: proc_id which is the proc_id
for procedure proc_name
FIX: delete PM_OBJECTS record ObjectName - ObjectGUID

ERROR: PROC_MGT_HIERARCHY record refers to non existent object ObjectGUID
FIX: delete PROC_MGT_HIERARCHY record ObjectGUID

ERROR: PROC_MGT_HIERARCHY record refers to non existent folder ParentGUID
FIX: change PROC_MGT_HIERARCHY record ObjectGUID to point to Root

ERROR: PM_OBJECTS ObjectName - ObjectGUID not in hierarchy
FIX: add PROC_MGT_HIERARCHY record ObjectGUID to point to Root
```

The following list shows the errors that can occur but that cannot currently be fixed by specifying the `fix` parameter. If any of these errors occur you should contact TIBCO Support for further assistance.

```
ERROR: PM_OBJECTS ObjectName - ObjectGUID refers to proc_id: proc_id which is too small

ERROR: PM_OBJECTS ObjectName - ObjectGUID refers to proc_id: proc_id which is too
big(>pcount)

ERROR: PM_OBJECTS ObjectName - ObjectGUID refers to proc_id: proc_id which is referred to
by other record(s)

ERROR: PM_OBJECTS ObjectName - ObjectGUID (PROC) refer to proc_id: proc_id which is a
SUBPROC

ERROR: PM_OBJECTS ObjectName - ObjectGUID (SUBPROC) refers to proc_id: proc_id which is a
PROC

ERROR: PROC_INDEX p_ix - proc_name is not referred to by any PM_OBJECTS records
```

Examples

1. This example shows the output from the `swadm show_servers` command.
    The root library contains the CARPOOL, HIRING and QUOTA procedures and two
    libraries - Purchasing and Admin, each of which contains further procedures.

    A corrupt TEST3 record, which references a procedure that does not exist, has
    also been found.

```
# swadm show_procedures
```
```
ERROR: PM_OBJECTS TEST3 - DA22EA30-FE69-11D7-A619-0050DAC9102A refers to proc_id:
-1 which does not exist
(F) ROOT_LIBRARY - ROOT_LIBRARY_GUID
    (F) Purchasing - A14E77B0-D268-11D7-BE25-0050DAC9102A
        (P) PROC1 - 96EFB7C0-F5D0-11D7-BAB5-0050DAC9102A
        (P) TEST1 - ACABECB0-D268-11D7-9833-0050DAC9102A
        (P) TEST3 - DA22EA30-FE69-11D7-A619-0050DAC9102A
        (S) DOCHECK - 99C651A0-E3C8-11D7-911C-0050DAC9102A
        (S) SUB1 - 306F0B50-DFD9-11D7-A8AC-0050DAC9102A
        (T) TEMPL1 - 3F956EB0-DFDA-11D7-A683-0050DAC9102A
    (F) Admin - 66D85000-E321-11D7-B184-0050DAC9102A
        (P) DYNAMIC1 - 272CA750-E3C7-11D7-A96A-0050DAC9102A
        (P) TEST2 - 43F72230-F507-11D7-BFCF-0050DAC9102A
        (P) WAIT1 - C88236B0-E329-11D7-BCB9-0050DAC9102A
        (S) DOCHECK - 99C651A0-E3C8-11D7-911C-0050DAC9102A
        (T) TEMPL1 - 3F956EB0-DFDA-11D7-A683-0050DAC9102A
    (P) CARPOOL - 9E697DC0-D4F7-11D7-B115-0050DAC9102A
    (P) HIRING - 75A4BB20-D4F7-11D7-9E50-0050DAC9102A
    (P) QUOTA - 94A58F00-D4F7-11D7-88D0-0050DAC9102A
```

2. This example shows the output when the `swadm show_servers fix` command is used to correct the problem found in the previous example. The corrupt TEST3 record is deleted.

```
# swadm show_procedures fix
```
```
ERROR: PM_OBJECTS TEST3 - DA22EA30-FE69-11D7-A619-0050DAC9102A refers to proc_id:
-1 which does not exist
FIX: delete PM_OBJECTS record TEST3 - DA22EA30-FE69-11D7-A619-0050DAC9102A
(F) ROOT_LIBRARY - ROOT_LIBRARY_GUID
    (F) Purchasing - A14E77B0-D268-11D7-BE25-0050DAC9102A
        (P) PROC1 - 96EFB7C0-F5D0-11D7-BAB5-0050DAC9102A
        (P) TEST1 - ACABECB0-D268-11D7-9833-0050DAC9102A
        (S) DOCHECK - 99C651A0-E3C8-11D7-911C-0050DAC9102A
        (S) SUB1 - 306F0B50-DFD9-11D7-A8AC-0050DAC9102A
        (T) TEMPL1 - 3F956EB0-DFDA-11D7-A683-0050DAC9102A
    (F) Admin - 66D85000-E321-11D7-B184-0050DAC9102A
        (P) DYNAMIC1 - 272CA750-E3C7-11D7-A96A-0050DAC9102A
        (P) TEST2 - 43F72230-F507-11D7-BFCF-0050DAC9102A
        (P) WAIT1 - C88236B0-E329-11D7-BCB9-0050DAC9102A
        (S) DOCHECK - 99C651A0-E3C8-11D7-911C-0050DAC9102A
        (T) TEMPL1 - 3F956EB0-DFDA-11D7-A683-0050DAC9102A
    (P) CARPOOL - 9E697DC0-D4F7-11D7-B115-0050DAC9102A
    (P) HIRING - 75A4BB20-D4F7-11D7-9E50-0050DAC9102A
    (P) QUOTA - 94A58F00-D4F7-11D7-88D0-0050DAC9102A
```

Tidy Instances of Procedures

Each time you edit and save a version of a procedure, a new instance of the procedure version is created. By default, all the instances of a procedure are kept. If you want to limit the amount of old instances that are kept for each procedure (for example, in order to save space in the iProcess database), you need to set the PROC_VER_NUM_INSTANCES attribute.

Even if you have set the PROC_VER_NUM_INSTANCES attribute, the most recent instance of a procedure version is always kept.

The first time you set the PROC_VER_NUM_INSTANCES attribute, you can use the swadm tidy_instances command to tidy up the old instances of a procedure so that they match the value you have set for the PROC_VER_NUM_INSTANCES attribute. Otherwise, the instances of a procedure are tidied up as and when a procedure is saved. This is because a tidy operation is performed each time a procedure is saved.

To tidy up the instances of a procedure defined on iProcess Engine, use the following command:

```
swadm tidy_instances machine_ID proc_ID
```

where:

- *machine_ID* is the unique identifier for the server. If you specify a value of 0, the command will apply to all servers in iProcess Engine.

- *proc_ID* is the unique identifier for the procedure. If you specify a value of 0, the command will apply to all procedures in iProcess Engine.

Chapter 9 **Administering Firewall Port Ranges**

This chapter explains how to use the swadm server configuration utility, which is
under the *SWDIR*\util directory, to set up and use port ranges for iProcess
Engine, for use with firewall filters when iProcess Engine is being used in a
firewalled environment.

To use this utility, you must be logged in to iProcess Engine as an Administrator
or (on UNIX) as a background user or root user.

If you are using a node cluster architecture, you can run this utility from any
server within the cluster (as long as that server has a connection to the TIBCO
iProcess Engine database instance).

Topics

Overview

If you are using iProcess Engine in a firewalled environment, you can define specific port ranges which the firewall administrator can add to the network firewall filter.

A port range is a specific range of either port numbers, RPC numbers or both. Once you have defined a port range, you can place the iProcess Engine node behind it. iProcess Engine processes will then only accept incoming RPC requests from within that port range.

> For more information about how iProcess Engine works in a firewalled environment, see "Using iProcess Engine in a Firewalled Environment" in *TIBCO iProcess Engine Architecture Guide*.

Using Port Ranges with a Node Cluster

If your iProcess Engine uses a node cluster configuration, each server in the iProcess Engine node can sit behind the same port range, sit behind a different port range, or not sit behind a port range at all, according to your network configuration requirements.

How Port Range Information is Stored

Port range information is stored in the following tables in the iProcess Engine database:

- `port_range` - contains the firewall data about individual port/RPC numbers that lie within each port range defined on this iProcess Engine node.
- `port_range_active` - lists what port/RPC numbers are being actively used to provide RPC services by iProcess Engine processes.
- `port_range_conf` - lists the port ranges currently defined for this iProcess Engine node.
- `port_range_nodes` - lists which port range configurations are being used by which machines in the iProcess Engine node.

See "Firewall Port Ranges" in the appropriate database guide for more information about these tables.

How to Set up and use a Port Range

To set up and use a port range on iProcess Engine:

1. Use the ADD_RANGE command to define the port range.

2. Use the SET_RANGE command to place the required iProcess Engine server(s) behind the defined port range.

3. Pass the details of the port range to the firewall administrator, to include in the network firewall filter.

> If iProcess Engine servers are configured to run behind port ranges, a log file detailing the resource allocation is stored in the rpcport.log file, which is located in the *SWDIR*\logs directory. See iProcess Engine Log Files on page 439 for more information.

Port Range swadm Commands

The following table summarizes all the swadm commands that you can use to administer port ranges. Each command is fully described in the following sections.

To do this...	Use this command...
Define a new port range.	ADD_RANGE
Delete an existing port range.	DEL_RANGE
Modify an existing port range (for example, to change the number range or operating mode).	MOD_RANGE
Place an iProcess Engine server behind a defined port range, or remove an iProcess Engine server from behind a defined port range.	SET_RANGE
Show how the ports for a particular port range are currently allocated.	SHOW_PORTS
Show the details of all defined port ranges and the iProcess Engine servers that are sitting behind them.	SHOW_RANGES

ADD_RANGE

swadm command

Syntax   swadm ADD_RANGE [-m *Range_mode*] [-p *Port_range_start*] [-r *RPC_range_start*]
                        [-s *Range_size*]

Description   This command defines a new port range for use with this iProcess Engine node.
You can then use the SET_RANGE command to place a server behind this port
range.

The port range is stored as a record in the port_range_conf table.

Options

Option	Description
-m *Range_mode*	Defines how servers that use this port range configuration should allocate ports. Specify one of the following values:
	0 - Do not use port or RPC ranges. A process can use any port number and RPC number (as assigned by the operating system).
	1 - Use port ranges. A process must use a port number allocated from within the defined range, but can use any RPC number.
	2 - Use RPC ranges. A process must use an RPC number allocated from within the defined range, but can use any port number.
	3 - Use port ranges and RPC ranges. A process must use both a port number and an RPC number allocated from within the defined ranges.
	If this value is omitted the range mode defaults to 3.
-p *Port_range_start*	The port number that the range should start from. (The range will therefore end at *Port_range_start* + *Range_size*.)
	If this value is omitted the port range start defaults to 10000.
-r *RPC_range_start*	The RPC number that the range should start from. (The range will therefore end at *RPC_range_start* + *Range_size*.)
	If this value is omitted the RPC range start defaults to 400000.
-s *Range_size*	The number of slots in the defined port and/or RPC number ranges.
	If this value is omitted the range size defaults to 20.

Errors   The following error messages may be returned by this command.

Message	Description
`Unable to access the port_range_conf table`	`swadm` cannot update the iProcess Engine database. Examine the `sw_error` and `sw_warn` files, which is located in the *SWDIR*\logs directory, for more information about the cause of the error. See iProcess Engine Log Files on page 439 for more information.

See Also   DEL_RANGE, MOD_RANGE, SET_RANGE, SHOW_PORTS, SHOW_RANGES

DEL_RANGE

swadm command

Syntax swadm DEL_RANGE *Port_range_ID*

Description This command deletes an existing port range from iProcess Engine. The port range is deleted from the `port_range_conf` table.

When you run this command, if any servers are currently configured to run behind this port range the following prompt is displayed:

```
Deleting this port range will cause the following servers to be
removed from the port_range_nodes table:
server_ids, ...
Are you sure you want to do this (Y/N)?
```

where *server_ids* is a comma-separated list of server identifiers and names for the servers that are currently configured to run behind this port range. If you answer:

- Y, the port range is deleted. The indicated servers are no longer running behind a port range. (The appropriate entries are deleted from the `port_range_nodes` table.)

- N, the port range is not deleted. The indicated servers are still running behind it.

Options

Option	Description
Port_range_ID	The ID of the port range that you want to delete. You can use the SHOW_RANGES command to find out what port range IDs are defined.

Errors The following error messages may be returned by this command.

Message	Description
The specified port_range_ID paramater Port_range_ID is invalid.	You have used a *Port_range_ID* value that does not exist. Re-run the command using the correct *Port_range_ID* value.
Unable to access the database table.	swadm cannot update the iProcess Engine database. Examine the `sw_error` and `sw_warn` files, which is located in the *SWDIR*\logs directory, for more information about the cause of the error.

See Also   ADD_RANGE, MOD_RANGE, SET_RANGE, SHOW_PORTS, SHOW_RANGES

MOD_RANGE

swadm command

Syntax   swadm MOD_RANGE *Port_range_ID* [-m *Range_mode*] [-p *Port_range_start*]
                 [-r *RPC_range_start*] [-s *Range_size*]

Description This command modifies one or more values for an existing port range. The port
                 range record in the port_range_conf table is updated.

> You cannot use this command if the port range you want to modify is currently in
> use. For example, if any of the ports within the range are currently allocated to
> iProcess Engine processes. You can check this using the SHOW_PORTS command.

Options

Option	Description
Port_range_ID	The ID of the port range that you want to modify.
	You can use the SHOW_RANGES command to find out what port range IDs are defined.
-m *Range_mode* -p *Port_range_start* -r *RPC_range_start* -s *Range_size*	Specify an allowed value as defined for the same parameter in the ADD_RANGE command. If one or more of these parameters is omitted the current value is left unchanged.

Errors   The following error messages may be returned by this command.

Message	Description
There are currently n records allocated from this port range configuration.	You cannot update the *Port_range_ID* port range because it is currently in use.
Unable to access the port_range_conf table.	swadm cannot update the iProcess Engine database. Examine the sw_error and sw_warn files, which is located in the *SWDIR*\logs directory, for more information about the cause of the error.

See Also   ADD_RANGE, DEL_RANGE, SET_RANGE, SHOW_PORTS, SHOW_RANGES

SET_RANGE

swadm command

Syntax swadm SET_RANGE *Machine_ID* [*Port_range_ID*]

Description This command can be used to:

- place an iProcess Engine server behind a defined port range.
- remove an iProcess Engine server from behind a defined port range.

This information is updated in the `port_range_nodes` table.

Options

Option	Description
Machine_ID	The server ID of the machine that you want to add to or remove from a port range.
	You can use the SHOW_SERVERS command to find out the server IDs of servers in this iProcess Engine node.
Port_range_ID	If you want to:
	add the specified *Machine_ID* to a port range, specify the ID of the port range that you want to place this server behind.
	You can use the SHOW_RANGES command to find out what port range IDs are defined.
	remove the specified *Machine_ID* from the port range that it is currently placed behind, you should omit this parameter.

Errors The following error messages may be returned by this command.

Message	Description
Machine_ID is not a valid logical machine ID. Use 'swadm SHOW_SERVERS' to see the correct list.	You have used a *Machine_ID* that does not exist. Re-run the command using the correct server ID.
The specified Port Range ID parameter is invalid.	You have used a *Port_range_ID* value that does not exist. Re-run the command using the correct *Port_range_ID* value.

Message	Description
`Unable to add the specified iPE machines to the port range configuration. Check sw_error/sw_warn for more details.`	`swadm` cannot update the iProcess Engine database. Examine the `sw_error` and `sw_warn` files, which is located in the *SWDIR*`\logs` directory, for more information about the cause of the error.

See Also   ADD_RANGE, DEL_RANGE, MOD_RANGE, SHOW_PORTS, SHOW_RANGES

SHOW_PORTS

swadm command

Syntax   `swadm SHOW_PORTS [-m ` *Machine_ID* `] [-p ` *Process_name* `]`

Description   This command displays information about which ports are currently being used by processes on this iProcess Engine node. This information is read from the `port_range_active` table.

Options

Option	Description
-m *Machine_ID*	The server ID of the machine that you want to show details for.
	You can use the SHOW_SERVERS command to find out the server IDs of servers in this iProcess Engine node.
	If this parameter is omitted the local machine is used.
-p *Process_name*	The logical process name (in full) that you want to show details for.
	You can use the SHOW_PROCESSES command to find out the different logical process names.
	If this parameter is omitted all iProcess Engine processes that currently have port/RPC numbers allocated are shown.

Output   The command displays the current port number and RPC number allocations for the specified parameters. For example:

```
---------------------------------------------------------------------------
Machine ID   Process Name Process Instance Port Number RPC Number Process ID
---------------------------------------------------------------------------
1            RPCBG        1                1147        1073745660 3836
1            RPC_POOL     1                1196        1073746828 5004
1            RPC_TCP_LI   1                1121        391875     3784
1            WIS          1                1145        1073745652 3828
1            WIS          2                1138        1073745636 3812
1            WQS          1                1131        1073744748 2924
```

Errors   The following error messages may be returned by this command.

Message	Description
`Unable to access the` `port_range table.`	swadm cannot read the information from the iProcess Engine database. Examine the `sw_error` and `sw_warn` files, which is located in the *SWDIR*`\logs` directory, for more information about the cause of the error.

See Also   ADD_RANGE, DEL_RANGE, MOD_RANGE, SET_RANGE, SHOW_RANGES

SHOW_RANGES

swadm command

Syntax   swadm SHOW_RANGES

Description   This command shows the port ranges that are currently defined on this iProcess
Engine node, and the servers that are currently running behind each of them. This
information is read from the `port_range_conf` and `port_range_nodes` tables.

Output   The command displays the following information about the port ranges (values
shown are examples):

```
--------------------------------------------------------------------
Port Range ID Range Mode Range Size Port Start RPC Start  Server ID's
--------------------------------------------------------------------
1             0          20         10000      400000
2             2          50         11000      410000
3             1          20         15000      400000
```

where:

- `Range ID` is the ID of this port range

- `Range Mode`, `Range Size`, `Port Start` and `RPC Start` are the configuration
  values for this port range. See the ADD_RANGE command for a full description
  of these values.

- `Server ID's` is a comma-delimited list of server IDs of the servers that are
  currently running behind this port range. You can use the SHOW_SERVERS
  command to find out the details of each server ID.

See Also   ADD_RANGE, DEL_RANGE, MOD_RANGE, SET_RANGE, SHOW_PORTS

ADD_AQ_PORT_RANGE

swadm command

Syntax swadm ADD_AQ_PORT_RANGE {*machine_ID*|ALL|0} *start_port* *count*

Description Applies to TIBCO iProcess Engine for Oracle only.

This command adds a new port range in the aq_port_range_conf table.

Options

Option	Description
machine_ID	The ID of machine to which the port range is added. If this field is input as "ALL" or "0", the port range will be added to all machines.
start_port	The port at which the port range starts.
count	The range size for the port range.

Output This command outputs a *range_ID* which identifies this port range.

See Also MOD_AQ_PORT_RANGE, DEL_AQ_PORT_RANGE

MOD_AQ_PORT_RANGE

swadm command

Syntax   swadm MOD_AQ_PORT_RANGE *machine_ID range_ID start_port count*

Description   Applies to TIBCO iProcess Engine for Oracle only.

This command modifies a port range in the aq_port_range_conf table.

Options

Option	Description
machine_ID	The ID of the machine to which the port range identified by the *range_ID* argument applies.
range_ID	The range identifier of the port range you wish to modify.
start_port	The port at which the modified port range now starts.
count	The new range size for the modified port range.

See Also   ADD_AQ_PORT_RANGE, DEL_AQ_PORT_RANGE

DEL_AQ_PORT_RANGE

swadm command

Syntax	swadm DEL_AQ_PORT_RANGE *range_ID*
Description	Applies to TIBCO iProcess Engine for Oracle only.
	This command deletes a port range from the aq_port_range_conf table.

Options

Option	Description
range_ID	The range_ID of the port range you wish to delete.

See Also	ADD_AQ_PORT_RANGE, MOD_AQ_PORT_RANGE

Chapter 10 **Administering Activity Monitoring and Work Queue Delta Publication**

This chapter explains how to configure iProcess Engine to publish iProcess Engine activity information and Work Queue Deltas to external applications.

Topics

Overview

The TIBCO iProcess Engine node can be enabled to publish both iProcess Engine activity information and iProcess Engine Work Queue Deltas to external applications, using JMS queues.

An activity is any instruction in iProcess Engine that creates an audit trail entry, for example, Case started or Event Issued. You can configure any combination of step and/or activity to be monitored. This enables an external application to monitor important business events during the processing of cases.

A BG process can identify if a step is being processed and if activity monitoring has been configured for it. The BG process then sends details of the configured activities in XML format to the IAPJMS process.

Work Queue Delta publication enables an external application (iProcess Server Objects (Java) or iProcess Server Objects (.NET)) to monitor a queue and to retrieve only those work items in a given work queue that have changed since the subscription started. The WIS process sends messages reporting Work Queue changes to the IAPJMS process.

For both activity monitoring and Work Queue Delta publication, the IAPJMS process sends the XML message to a specified JMS topic, from which an external application (for example, iProcess Objects, iProcess Analytics or an external application that you have written yourself) can receive the JMS messages.

Note that this Work Queue Delta publication via JMS functionality is independent of the ability to obtain Work Queue Delta items via the iProcess Server Objects interface, as described in *TIBCO iProcess Server Objects (Java) Programmer's Guide.*

Enabling Activity Monitoring

Activity monitoring and Work Queue Delta publication can be configured when iProcess Engine is installed. If this has not been done, you can subsequently enable them manually. To do this, you need to complete the following steps:

1. Make sure that the SWLIB_PATH process attribute points to the directory containing the Java libraries that you want the IAPJMS process to use. See SWLIB_PATH on page 292.

   > By default, SWLIB_PATH points to the Java libraries distributed with iProcess Engine. You do not need to change this value unless you have a specific requirement for the IAPJMS process to use a different version of these libraries.

2. Enable activity monitoring on your iProcess Engine node by configuring the IAPJMS_PUBLISH process attribute. See IAPJMS_PUBLISH on page 279.

3. Specify the JMS message delivery method by configuring the IAPJMS_SYNCHRONOUS on page 281 and WQDJMS_TOPICNAME process attributes. See IAPJMS_SYNCHRONOUS on page 281 and WQDJMS_TOPICNAME on page 297.

4. For activity monitoring, configure the port number that is used for message communications between the BG process and IAPJMS process by configuring the IAPJMS_PORTNO process attribute. See IAPJMS_PORTNO on page 278.

5. For Work Queue Delta publication, configure the port number that is used for message communications between the WIS process and IAPJMS process by configuring the WQDJMS_PORTNO process attribute. See WQDJMS_PORTNO on page 296.

6. Configure the JNDI name for the JMS topic and whether it should be static or dynamic by configuring the IAPJMS_TOPICNAME, IAPJMS_SIMPLETOPIC on page 285 and WQDJMS_TOPICNAME process attributes. See IAPJMS_TOPICNAME on page 283, IAPJMS_SIMPLETOPIC on page 285 and WQDJMS_TOPICNAME on page 297.

   > If you are using WebLogic as your JMS provider, you must ensure that the WebLogic JNDI name and topic name are the same as the topic name specified in the IAPJMS_TOPICNAME and the WQDJMS_TOPICNAME attributes. (See WebLogic documentation for more information.)

7. Specify whether you wish the IAPJMS process to generate messages in a basic or extended format. See IAPSCHEMA on page 79.

8. Configure the JMS message error handling by configuring the `IAPJMS_ROLLBACK` process attribute. See IAPJMS_ROLLBACK on page 280.

9. Configure the JVM Attributes that should be specified when the Java Virtual Machine is started by configuring the `JVMPROPS` process attribute. See JVMPROPS on page 286.

10. Depending on your requirements, you can filter MER messages and Work Queue Delta messages using the message properties. See Filtering Message Event Request (MER) Messages on page 383.

11. Configure the IAP JMS configuration files - see Configuring the iProcess Activity Publication (IAP) Configuration Files on page 384

12. Update the IAP security principle and credentials - see Updating the IAP Security Principle and Credentials on page 388

Filtering Message Event Request (MER) Messages

Every MER message sent to the iProcess database to update the activity monitoring configuration information consists of XML requesting the events to monitor. The MER XML format is defined by the SWMonitorList.xsd schema.

The table below describes the properties of the MER message:.

Property	Description
IAPMessageType	The message type is MER (Monitor Event Request)
IAPProcedureName	The iProcess Engine procedure name
IAPNodeName	The name of iProcess Engine.
IAPComputerName	The name of the machine where iProcess Engine is installed.

You can filter the MER messages using these properties. Refer to the information supplied with your J2EE Application Server for more information on filtering messages.

Configuring the iProcess Activity Publication (IAP) Configuration Files

If you want to enable IAP, there are two configuration files that you can configure. If necessary consult the administrator for your JMS provider software. The configuration files are found in *SWDIR*\etc directory:

- `iapjms.properties` - contains all the configuration information for the IAPJMS process.

- `iapjms_classpath.properties` - contains a list of the required JAR files for each of the supported application servers.

Configuring the IAP JMS Properties File

The `iapjms.properties` file contains all the configuration information for the IAPJMS process. The `iapjms.properties` file enables you to configure the following settings:

Table 8   IAP JMS Properties File (Sheet 1 of 3)

Property	Description
IAPJMSConnect.InitialContextFactory	Defines the J2EE initial context factory to be used for all J2EE connections within the application.
IAPJMSConnect.InitialURL	Defines the initial context URL, if required.
IAPJMSConnect.SecurityPrinciple	Defines the username if security is set in the InitialContextFactory. See Updating the IAP Security Principle and Credentials on page 388 for more information.
IAPJMSConnect.SecurityCredentials	Defines the password if security is set in the InitialContextFactory. See Updating the IAP Security Principle and Credentials on page 388 for more information.

Table 8   IAP JMS Properties File (Sheet 2 of 3)

Property	Description
IAPJMSConnect.SecurityEncryption	Defines the encryption method used for the `IAPJMSConnect.SecurityCredentials` parameter. Valid values are: • `PLAIN` - Plain text format (default) • `IPE` - iPE proprietary encryption (this mechanism is used by the `swconfig` utility when writing the password) See Updating the IAP Security Principle and Credentials on page 388 for more information.
IAPJMSConnect.TopicConnectionFactory	Defines where the JMS topic details are configured. If a topic cannot be looked up then the topic is dynamically created by the `IAPJMS` process, if possible.
IAPJMSConnect.TimeToLive	Defines the maximum time to live for the JMS messages in millisceonds. If the property is set to `0` the messages never time out. For more information, see the documentation supplied with your J2EE Application Server.
IAPJMSConnect.Priority	Defines the priority of the JMS message in the system. For more information, see the documentation supplied with your J2EE Application Server.
WQDJMSConnect.ListenerThreads	Defines the number of socket listener threads that are created when the `IAPJMS` process starts up. **Note**: This property is not present in the `iapjms.properties` file by default. It should only be set on instructions from TIBCO Support.

Table 8   IAP JMS Properties File (Sheet 3 of 3)

Property	Description
WQDJMSConnect.ConnectionPoolSize	Each work queue is required to publish its messages on the same JMS connection to ensure that messages appear in the correct order. Therefore the IAPJMS process keeps a pool of JMS connection caches and ensures that all messages from each unique work queue are sent using the same connection. The connection pool is created when the IAPJMS process starts up.
	This property therefore defines the connection pool size, which is the maximum number of connections available for JMS publication. Each connection will always be used for all items for a single work queue.
	Note: This property is not present in the iapjms.properties file by default. It should only be set on instructions from TIBCO Support.

Configuring the IAPJMS Classpath File

The `iapjms_classpath.properties` file contains:

- A list of the IAPJMS internal libraries, as shown below:

```
########################################################################
# Internal libraries
########################################################################
#
#
# The following entries are required by the IAPJMS process and should NOT be
modified
#
classpath.internal.log4j=thirdparty/log4j-1.2.8.jar
classpath.internal.common=common_swprocess_library.jar,common_bootstrap_library.jar
,common_utils_library.jar
classpath.internal.socket=socketproxy_socketproxy_library.jar
classpath.internal.iapjms=iapjms_iapjms_library.jar
```

The internal libraries are required by the IAPJMS process and should not be modified.

- The required JAR files for each of the supported application servers. Shown below is an extract of the `iapjms.classpath` file that describes the JAR files for Websphere.

```
########################################################################
#WebSphere 5.1
########################################################################
#classpath.basedir.WAS=c:/program files/WebSphere/AppServer/lib
#classpath.WAS.1=bootstrap.jar,iwsorb.jar,j2ee.jar,wsexception.jar
#classpath.WAS.2=ffdc.jar,namingClient.jar,ras.jar,utils.jar,idl.jar
#classpath.WAS.3=messagingClient.jar,ecutils.jar,naming.jar
#classpath.WAS.MQ=com.ibm.mq.jar,com.ibm.mqjms.jar
#classpath.WAS.ext=ibmext.jar,ibmorb.jar
```

You must configure this file for the application server you are using. You must uncomment the lines of the file that apply to the application server you are using. For example, if you are using Websphere, you should uncomment the paths to the JAR files as shown below:

```
########################################################################
#WebSphere 5.1
########################################################################
classpath.basedir.WAS=c:/program files/WebSphere/AppServer/lib
classpath.WAS.1=bootstrap.jar,iwsorb.jar,j2ee.jar,wsexception.jar
classpath.WAS.2=ffdc.jar,namingClient.jar,ras.jar,utils.jar,idl.jar
classpath.WAS.3=messagingClient.jar,ecutils.jar,naming.jar
classpath.WAS.MQ=com.ibm.mq.jar,com.ibm.mqjms.jar
classpath.WAS.ext=ibmext.jar,ibmorb.jar
```

Updating the IAP Security Principle and Credentials

If you enabled IAP, default values for the JNDI/JMS user name and password are contained in the `iapjms.properties` file, which is located in the *SWDIR*\etc directory (in the properties `SecurityPrinciple` and `SecurityCredentials` respectively). For security reasons, you can change the user name/password using the `swconfig` utility as described below.

When you use the `swconfig` utility to modify the `iapjms.properties` file, a backup file (`iapjms.properties.bak`) is created, preserving the previous settings.

Resetting the User Name and Password

If you want to update the username and password without encrypting the password, you can directly edit the `iapjms.properties` file to add the new user name and password; otherwise use the procedure described below.

1. From the *SWDIR*\util directory, enter the following command:

   `swconfig -i`

2. The `swconfig` utility displays the current user name and prompts you to enter a new one.

3. The `swconfig` utility echoes the current password and prompts you to enter a new one.

4. The password is encrypted and the properties `IAPJMSConnect.SecurityPrinciple` and `IAPJMSConnect.SecurityCredentials` are updated accordingly.

Deleting the User Name and Password

If you do not want to use security, you can delete the current user name and password as follows:

1. Enter the following command:

   `swconfig -i -x`

2. The properties `IAPJMSConnect.SecurityPrinciple` and `IAPJMSConnect.SecurityCredentials` are deleted from the file `iapjms.properties` file.

Testing the Password

1. Enter the following command:

   ```
   swconfig -i -t
   ```

2. The swconfig utility prompts you to enter the user name and password.

   The password held in IAPJMSConnect.SecurityCredentials is decrypted and compared against the password you supplied.

3. The swconfig utility indicates whether the password is valid.

Using SWDIR\util\swadm to Administer Work Queue Delta Publication

You can use the `swadm` utility to administer (view and delete) subscriptions to work queues. Note that:

- To use this utility, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as background user or root user.

- If you are using a node cluster architecture, you can run this utility from any server within the cluster (as long as that server has a connection to the TIBCO iProcess Engine database instance).

The following table summarizes the commands you can use to administer work queue subscriptions.

Area	Task	swadm Command
Publication	List Subscriptions	SHOW_ALL_SUBSCRIPTIONS
	Clear Subscriptions	CLEAR_SUBSCRIPTION

List Subscriptions

To display a list of all publications of Work Queue Deltas currently enabled, use the following command:

```
swadm SHOW_ALL_SUBSCRIPTIONS
```

The command lists the following information for each publication:

- `WIS No` is the identifying number of the `WIS` process.

- `Work Queue` is the unique identifier for the queue subscribed to.

- `WQDID` is the Work Queue Delta ID.

- `JMS Topic` is the topic that is used for Work Queue Delta messages. By default it is the topic specified by the WQDJMS_TOPICNAME process attribute but a different topic can be specified by the subscribing application.

Example This example shows the output from the swadm SHOW_ALL_SUBSCRIPTIONS command.

WIS No	Work Queue	WQDID	JMS Topic
1	user002@swnod10 2	16D3B33A-D305-11DC-8FE2-0017A49 9ABAD	WQD TOPIC
1	user002@swnod10 2	2DB0E050-D305-11DC-8029-0017A49 9ABAD	WQD.TOPIC.USER002
1	user002@swnod10 2	E2AC427A-D304-11DC-AB61-0017A49 9ABAD	WQD TOPIC

Clear Subscriptions

To clear a work queue delta subscription, use the following command:

    swadm clear_subscription *queue_name* *WQD_ID*

where:

- *queue_name* is the descriptive alphanumeric name for the queue.

- *WQD_ID* is the Work Queue Delta ID.

Chapter 11 **Administering System Events**

This chapter explains how to audit, publish, review, or remove information about system events and how to import and export configuration of system events.

Topics

-
-

Configuring System Events Information

To configure system event information, you might have to complete the following tasks:

- Task A, Audit System Events

- Task B, Publish System Events

- Task C, Import and Export System Event Configurations

- Task D, Display System Event Information

- Task E, Remove System Event Information

Task A   Audit System Events

Set the value of the SYSTEM_EVENT_LOGGING process attribute to 1 to audit system events. See SYSTEM_EVENT_LOGGING on page 294.

Task B   Publish System Events

After auditing system events, set the PUBLISH_SYS_EVENT_METHOD process attribute to decide in which way to publish system events. Three ways to publish system events:

- Windows Event Viewer

  Set the PUBLISH_SYS_EVENT_METHOD process attribute to 1. This function is only available for Microsoft Windows system.

The following is an example of a published system event shown in Windows Event Viewer.

- TIBCO Hawk Agent

  Set the `PUBLISH_SYS_EVENT_METHOD` process attribute to 2. The following is the format of the alter message about published system events in TIBCO Hawk:

  Logical Machine ID-Logical Process Name-Logical Process Instance: *Details in DB*:*User*

  The following is an example of a published system events in TIBCO Hawk.

- The IAPJMS process

  Set the PUBLISH_SYS_EVENT_METHOD process attribute to 3. You can check the published system events in the XML format in the iapjms_java.log file, which is located in the *SWDIR*/logs directory.

  The following in an example of a published system event in the iapjms_java.log file:

```
<SystemEvent xmlns="http://bpm.tibco.com/2014/IAPSE/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://bpm.tibco.com/2014/IAPSE/1.0
SWSEMessage.xsd">

 <ActivityID>1</ActivityID>

 <EventDesc>Successful login</EventDesc>

 <DetailInfo>
  <MachineName>WANGJUAN-LT</MachineName>
  <ProcessName>RPC_POOL</ProcessName>
 </DetailInfo>

 <AuditUser>administrator</AuditUser>

 <AuditDate
 Microseconds="611534">2014-05-15T14:35:41</AuditDate>

</SystemEvent>
```

If the value of the SEJMS_TOPICNAME process attribute is set, the events are published on the topic name that is set in the SEJMS_TOPICNAME process attribute. Otherwise, the events are published on the topic name that is set in the IAPJMS_TOPICNAME process attribute.

See PUBLISH_SYS_EVENT_METHOD on page 287 for more information.

Task C   Import and Export System Event Configurations

You can specify publishing and auditing system events in a configuration file. To import and export a configuration file, use the swutil IMPEVENTCONF [filename] and swutil EXPEVENTCONF commands. The sysevents.cfg configuration file is used by default, which is located in the *SWDIR*/etc/english.lng directory.

See "System Event Monitoring" in *TIBCO iProcess swutil and swbatch Reference Guide* for more information.

Task D   Display System Event Information

If you want to review the information about system events, use the plist -e command.

See System Event Information on page 524 for more information.

Task E   Remove System Event Information

If you want to delete system event information from database, use the swadm

`delete_system_event` command.

See for more information.

Removing System Events Information from the iProcess Database

To clean system event information or to delete information about system events from the iProcess database before a specific date, enter the following command:

```
swadm delete_system_event
<typeid[,typeid…]]|ALL[,!typeid[,!typeid…]]> [DD/MM/YYYY]
```

where:

- `typeid` specifies the ID of a system event.

  To review events and their corresponding ID, see the `sysevents.cfg` file in the *SWDIR*/etc/english.lng directory.

- `ALL` specifies all the system events.

- `!typeid` specifies the ID of a system event that will not be removed from the iProcess database. Except for this system event, all other events will be removed from the iProcess database.

  Note: The `!typeid` parameter must be used together with the `ALL` parameter.

- `DD/MM/YYYY` specifies all the system events published in the iProcess database befored this date will be removed.

For example:

- To delete information about the system event where the type ID is equal to `1`:

  ```
  swadm delete_system_events 1
  ```

- To delete information about the system events where the type IDs are equal to `1`, `2`, and `3`:

  ```
  swadm delete_system_events 1,2,3
  ```

- To delete the system event information except the events with the type IDs equal to `5` and `6`:

  ```
  swadm delete_system_events ALL,!5,!6
  ```

- To delete the system event information that was published before `23/10/2013`:

  ```
  swadm delete_system_events ALL 23/10/2013
  ```

Chapter 12 **Administering the Work Queue Server and Work Item Server Processes**

This chapter describes how you can configure the `Work Queue Server (WQS)` and `Work Item Server (WIS)` processes for optimum performance.

Topics

Overview

The iProcess work queues, which contain all the iProcess users' work items, are managed by the following processes:

- `Work Queue Server (WQS)`, which handles the listing of queues. This process is run by `wqsrpc`, which is located in the *SWDIR*\etc directory. There is only a single `wqsrpc` process running at any time. See The WQS Process on page 403 for more information.

- `Work Item Server (WIS)`, which handles the listing of work items in the queues. This process is run by `wisrpc`, which is located in the *SWDIR*\etc directory. The number of `wisrpc` processes running is controlled by the Process Sentinels (the `process_config` table). See The WIS Process on page 409 for more information.

The `WQS` process handles what is displayed in the left hand pane of the Work Queue Manager (the queue list) and the `WIS` process handles the contents of the right hand pane (the work items list).
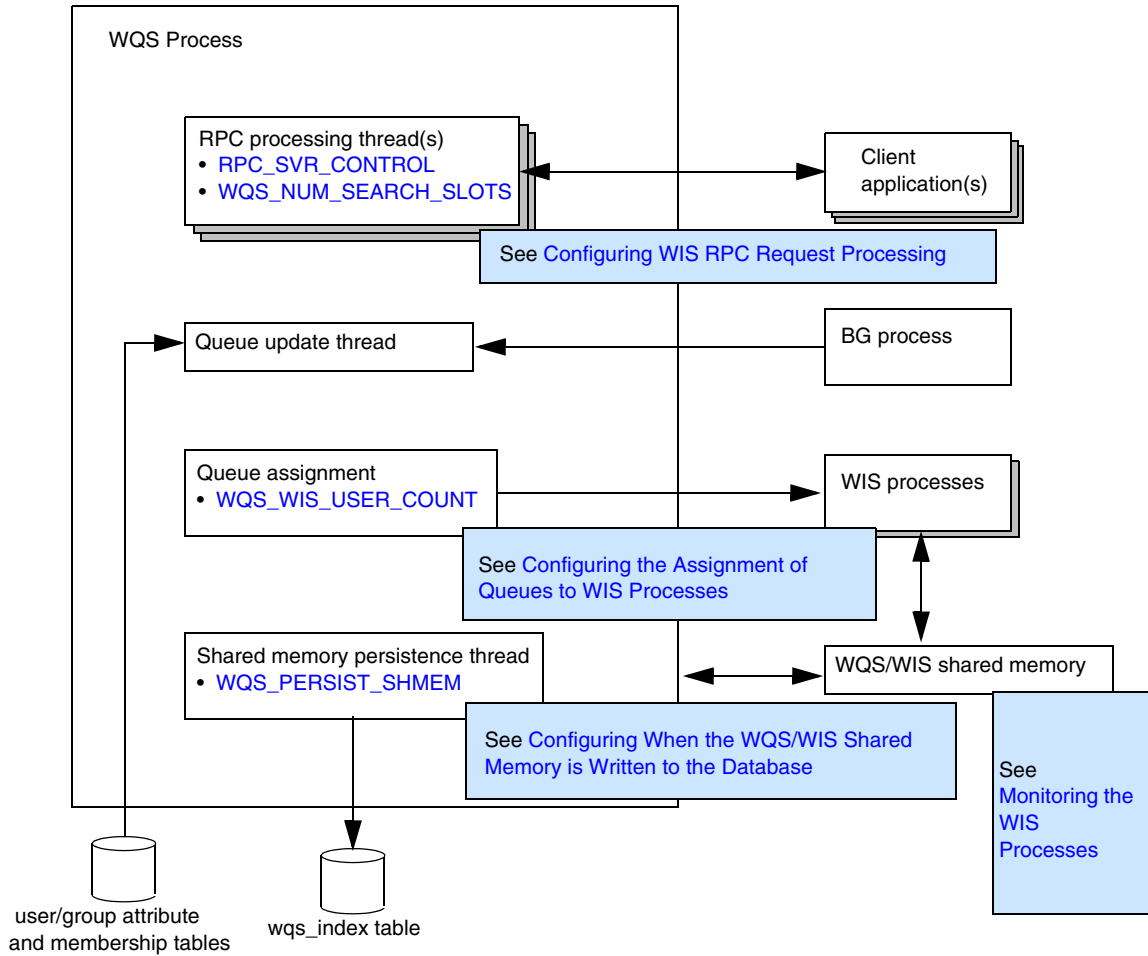
The WQS Process

The `Work Queue Server (WQS)` process handles the listing of work queues. The `WQS` process allocates one or more queues to each `WIS` process and responds to client RPC requests to access these queues.

The `WQS` process is multi-threaded, allowing it to perform multiple tasks simultaneously. Different threads are used to:

• process RPC requests from client applications.

• update work queues following a `MOVESYSINFO` event.

• persist the contents of the WQS/WIS shared memory to the database.

The following diagram shows:

• the different aspects of the `WQS` process' behavior that you can configure.

• the process attributes that you can use to do this.

• a reference for more information on how to configure each aspect of the `WQS` process' behavior.

Configuring WQS RPC Request Processing

To process RPC requests, both the `WQS` and `WIS` processes access a pool of "worker" threads that is provided by a multi-threaded RPC server shared library (SWRPCMTS). You can use the RPC_SVR_CONTROL process attribute to define the number of threads that are available in the SWRPCMTS library to process RPC requests.

You can adjust the value of this process attribute to optimize the `WQS` and `WIS` process' response times when processing RPC requests against available CPU capacity. Increasing the number of threads will improve the throughput of client RPC requests, but at the cost of increased CPU usage.

The RPC processing threads perform their work independently of and concurrently with the queue update thread. In pre-10.4 versions of iProcess Engine, where the `WQS` process was single-threaded, the `WQS` process had to switch between processing RPC requests and updating work queues.

Configuring the Assignment of Queues to WIS Processes

When iProcess Engine starts up, the `WQS` process is responsible for assigning all the work queues to `WIS` processes.

By default, queues are assigned to `WIS` processes dynamically, using either the `round-robin` or `on-demand` method (as determined by the `WQS_ROUND_ROBIN` parameter in the `staffcfg` file, which is located in the *SWDIR*\etc directory- see WQS_ROUND_ROBIN on page 51):

- `Round-robin`. This method assigns a work queue to each `WIS` process alphabetically, cycling round until all work queues are assigned. For example, if a system has 5 `WIS` processes and 20 work queues A to O then:

  — queues A, F, K are allocated to `WIS` process 1,

  — queues B, G, L are allocated to `WIS` process 2,

  — queues C, H, M are allocated to `WIS` process 3, etc.

  The `round-robin` method takes no account of queue size. It is best used when the messages are fairly evenly distributed between the majority of queues and user access is also evenly spread.

- `On-demand`. This method assigns work queues to `WIS` processes based on cost. All work queues have a weighting (determined by the `WQS_QUEUE_WEIGHTING` parameter) that determines the cost of the work queue to the `WIS` process. Queues are assigned to the `WIS` process with the lowest overall cost. The more work queues that are allocated to a `WIS` process, the higher the cost of the `WIS` process so the less new work queues are allocated to it. The cost calculation is as follows:

$$cost = wicount + (\text{WQS\_QUEUE\_WEIGHTING} * qcount)$$

where:

— *wicount* is the number of work items the WIS process is currently processing.

— *qcount* is the number of work queues the WIS process is currently processing.

The number of items in a work queue is taken from data that has been persisted to the wqs_index database table. If, for example, a new queue has been added to iProcess Engine after it has been started, it means the allocation of the work queues may not reflect the actual count of work items in the work queue. To overcome this, restart iProcess Engine. This results in the work queues being re-allocated according to the latest work item count.

To control how work queues are allocated to WIS processes, you can adjust the WQS_QUEUE_WEIGHTING parameter. This parameter changes the cost of a work queue to a WIS process. For example, the larger the value, the more that the number of work queues rather than the number of work items in the work queues determines whether a work queue is allocated to a WIS process. Therefore, if you have lots of work queues with an even amount of work items in each, you may want to increase the value of the WQS_QUEUE_WEIGHTING parameter. If you only have a few work queues that contain large amounts of work items, you may want to lower the value.

The effect of on-demand assignment is that work queues are distributed across WIS processes based on their cost so a more even distribution of work queues is achieved.

However, there are two additional methods you can use to customize the assignment process to better reflect your system requirements, and so optimize performance.

The following sections describe these methods.

Using Different WIS Processes to Handle User and Group Queues

User queues and group queues frequently have different characteristics, in terms of the amount of load they carry.

For example, if group queues are far more active than user queues on your system, you may want to give them higher priority for WIS process allocation.

You can do this by specifying the WQS_WIS_USER_COUNT process attribute for the WQS process. This attribute defines the number of WIS processes that should be dedicated to handling user queues and group queues respectively (either as a fixed number or as a percentage of the available processes). See WQS_WIS_USER_COUNT on page 244 for more information.

Assigning a Queue Explicitly to a WIS Process

If you have certain queues that are very large or very busy, you may find it useful to dedicate specific WIS processes to handling only those queues (leaving the remaining queues to be dynamically assigned to the remaining WIS processes).

To dedicate a specific WIS process to handling a specific queue:

1. Start the Process Administrator, and then start the User Manager. (See "Using TIBCO iProcess Administrator" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.)

2. To make it possible to allocate queues to specific WIS processes, define a new attribute called SW_WISINST. This should have a Type of Numeric, with a Decimal value of 0.

   See "Adding a New Attribute" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

3. To assign a queue to a specific WIS process, assign the WIS instance number that you want the queue to use as the value of the SW_WISINST attribute for that queue. (You can use the swadm show_processes command to list the available WIS instances - see Show Server Processes on page 111.)

   See "Setting User Values for an Attribute" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

4. Save your changes, exit from User Manager and perform a MoveSysInfo event to register your changes on iProcess Engine.

   See "Moving System Information" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

5. If the queue is already in use (and therefore already allocated to a WIS process), you will need to stop and restart iProcess Engine before the change takes effect.

Once a WIS process has been dedicated to handling a specific queue or queues, it will handle only those queues. It is no longer available for dynamic queue allocation.

There is one exception to this: if all the available WIS processes are dedicated to handling specific queues, and a new queue is added, the queues are no longer treated as dedicated. This means that:

- the new queue will be dynamically assigned to the appropriate WIS process, according to the current dynamic allocation rules. All dedicated WIS processes are considered to be available to handle the queue. See Using Different WIS Processes to Handle User and Group Queues on page 406.

- the dedicated WIS processes continue to handle their assigned queues (but they may also have to handle the newly assigned queue as well).

An Example of How to Use These Assignment Methods

By using the methods described above, you can configure your system to operate more efficiently under load. For example, consider a system that has 6 WIS processes (WIS 1-6), 8 group queues (GQ1-8) and 500 users (UQ1-500). Queue characteristics are:

- GQ1 has 100K items and is a holding queue (sometimes searched).

- GQ2 has 50K items and is the most active queue.

- GQ3-8 are all fairly busy with up 10K items in each.

- User queues are not used extensively.

The system is now configured as follows:

- GQ1 is assigned to WIS 1.

- GQ2 is assigned to WIS 2.

- WQS_WIS_USER_COUNT is set to 2

This means that:

- The two biggest queues, GQ1 and GQ2, are each handled by their own dedicated WIS process, WIS 1 and WIS 2.

- The remaining 6 group queues, GQ3 to GQ8, are handled by 2 of the 4 remaining WIS processes. The queues are dynamically assigned to WIS processes.

- The remaining 2 WIS processes handle the 500 user queues. The queues are dynamically assigned to WIS processes.

Configuring When the WQS/WIS Shared Memory is Written to the Database

The WQS/WIS shared memory cache holds summary information about work queues, such as which WIS process is handling a queue, how many work items it contains, how many new items, items with deadlines and so on. This information is constantly updated by the WQS and WIS processes.

The shared memory persistence thread wakes up every WQS_PERSIST_SHMEM seconds and writes the contents of the WQS/WIS shared memory to the wqs_index database table.

When the WIS process starts up, it needs to know how many work items are in each queue that it is handling, so that it can determine whether or not to cache the queue immediately (see Configuring When WIS Processes Cache Their Queues on page 416). The WIS process can therefore read this information from the total_items column in the wqs_index database table.

The WIS Process

The `Work Item Server (WIS)` process handles the listing of work items in user and group queues. Each `WIS` process is allocated one or more queues to handle by the `WQS` process and responds to client RPC requests to process work items held in these queues.
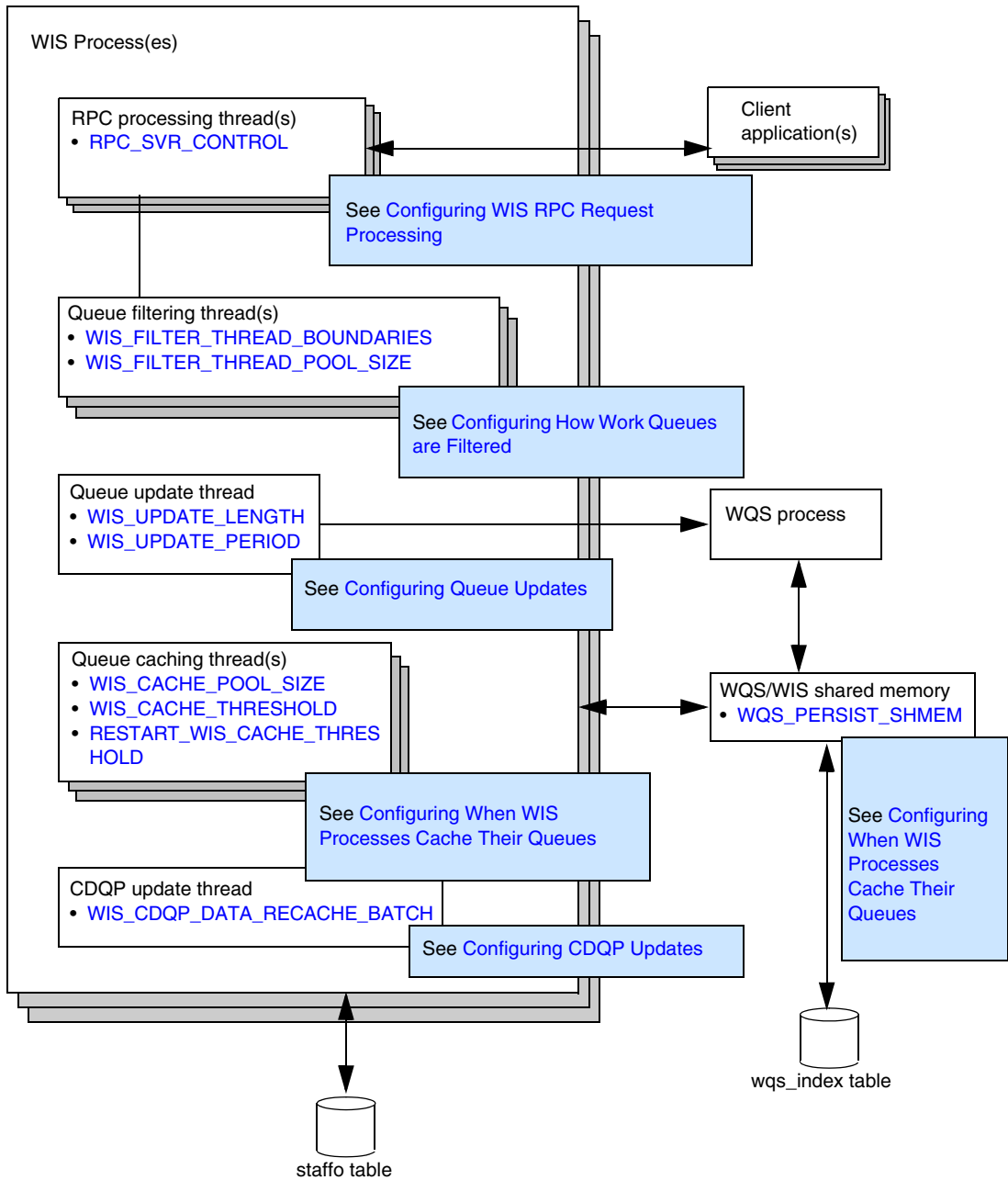
You can use the `swadm add_process` and `delete_process` commands to change the number of `WIS` processes on your system according to your requirements. See Using SWDIR\util\swadm to Administer Server Processes on page 110 for more information about how to use these commands.

The `WIS` process is multi-threaded, allowing it to perform multiple tasks simultaneously. Different threads are used to:

- process RPC requests from client applications.

- filter work queues - for example, only show work items started by a particular user.

- update each queue being handled for example, checking for expired deadlines, priority escalations, or for new queues to be handled.

- cache the information that the `WIS` process maintains about each work queue that it is handling, allowing the `WIS` processes to respond quickly to RPC requests from client applications.

- dynamically update CDQP definitions for work items.

The following diagram shows:

- the different threads that are used by the `WIS` process.

- the process attributes that you can use to control each type of thread.

- a reference for more information on how to configure this aspect of the `WIS` process' behavior.

Monitoring the WIS Processes

You can use the *SWDIR*\util\plist -w command to monitor the operation of the WIS processes. TIBCO recommends you do this regularly, particularly in the following circumstances:

- On initial configuration of your system. The default values can be used but when cases, users or groups are added, you will need to monitor and perhaps configure the system.

- After a number of new queues have been added.

- After a significant increase in the number of cases in the system. If there are only a small number of queues, for example, less than 10, monitor the system after you add more users or group queues so you can monitor the load balancing of the WIS processes.

The format of the *SWDIR*\util\plist -w command is:

```
plist -w[V][v] [WIS]
```

where:

- V can be used to display additional information (the LastCacheTime and CDQPVer columns)

- v can be used to display additional information (the Version, NewVers, DelVers, ExpVers, UrgVers, and QParamV columns)

- *WIS* is the number of a specific WIS process, and can be used to display details only for that WIS process. If this parameter is omitted, the command displays details for all the WIS processes.

Use the plist -w command to view detailed information about the WIS processes such as the number of items in the queue, whether the queue is disabled, and the number of new items in each WIS process.

Use the plist -wVv command to view all the additional information that is returned by the plist -wV and the plist -wv commands.

For example (using plist -wV):

WIS	QueueName	Flags	#Items	#Newp	#Dead	#Urgent	*LastCacheTime(ms)*	CDQPVer
1	sblanch	-----NM	3000	3000	0	0	*766*	-1
1	steveb	-------	0	0	0	0	*11*	-1
1	swadmin	-----NM	2	2	0	0	*29*	-1
1	swgrp0000	--G----	0	0	0	0	*12*	-1
1	swgrp0001	--G----	0	0	0	0	*11*	-1
1	swgrp0002	--G----	0	0	0	0	*11*	-1
1	swgrp0003	--G----	0	0	0	0	*-1*	-1

The `plist -w[V][v] [WIS]` command displays the following information.

Table 9   Information Returned by the plist -w[V][v] [WIS] Command

Column	Description
WIS	The number of this `WIS` process instance.
QueueName	The name of the work queue allocated to this WIS instance.
Flags	Any combination of the following, in order. A "-" in place of the indicated letter means that the corresponding flag is not set: • `D` = The queue is disabled (this would normally be when the system has just been started and the queues have not yet been allocated to a WIS. • `U` = There are urgent items in this queue. • `G` = This is a group queue. • `T` = This is a test queue. • `D` = There are items in this queue with deadlines set. • `N` = There is new mail in this queue. • `M` = There is mail in this queue (i.e. it is not empty).
#Items	The total number of work items in this work queue.
#NewP	The total number of new (unread) work items in this work queue.
#Dead	The total number of work items in this work queue that have deadlines.
#Urgent	The total number of urgent work items in this work queue.
LastCacheTime	Displayed if the `-V` option is used. The number of milliseconds that the `WIS` process took to cache this work queue. Note that: • The time shown is the time taken when the queue was last cached (which could be either when the `WIS` process was started or when the queue was first accessed). The number of items in the queue at that time may have been different from the number of items currently in the queue as shown in the `#Items` column. • A value of `-1` indicates that the queue has not been cached yet.

Table 9   Information Returned by the plist -w[V][v] [WIS] Command (Cont'd)

Column	Description
CDQPVer	Displayed if the –V option is used.
	The current CDQP definition version for this work queue. (This should match the version number of the cdqp row in the version database table.)
#New	Displayed if the –v option is used.
	The total number of new (unread) work items in the work queue.
Version	Displayed if the –v option is used.
	The total number of work items in the work queue that are processed from the time when the queue was last cached.
NewVers	Displayed if the –v option is used.
	The total number of new (unread) work items in the work queue from the time when the queue was last cached.
DelVers	Displayed if the –v option is used.
	The total number of work items in the work queue that are deleted from the time when the queue was last cached.
ExpVers	Displayed if the –v option is used.
	The total number of work items in the work queue that are expired from the time when the queue was last cached.
UrgVers	Displayed if the –v option is used.
	The total number of urgent work items in the work queue from the time when the queue was last cached.
QParamV	Displayed if the –v option is used.
	The current QParam definition version for the work queue.

Configuring WIS RPC Request Processing

To process RPC requests, both the WIS and WQS processes access a pool of "worker" threads that is provided by a multi-threaded RPC server shared library (SWRPCMTS). You can use the RPC_SVR_NUM_THREADS process attribute to define the number of threads that are available in the SWRPCMTS library to process RPC requests.

You can adjust the value of this process attribute to optimize the WQS and WIS process' response times when processing RPC requests against available CPU capacity. Increasing the number of threads will improve the throughput of client RPC requests, but at the cost of increased CPU usage.

> The RPC processing threads perform their work independently of and concurrently with the queue update thread. In pre-10.4 versions of iProcess Engine, where the WIS process was single-threaded, the WIS process had to switch between processing RPC requests and updating work queues.

Configuring How Work Queues are Filtered

When filter criteria are applied to a work queue - for example, only show work items started by a particular user - the WIS process has to filter the work queue to find the correct items to display.

By default, the WIS process uses the thread that is processing an RPC request to perform any work queue filtering required by that RPC request. This is perfectly adequate if the queues are small and the filter criteria are simple. However, the time taken to filter a queue can increase significantly as the number of work items in the queue grows and/or the complexity of the filter criteria increases. This can result in a perceptible delay for the user viewing the work queue.

For example, filtering a queue that contains over 100000 work items using filter criteria that includes CDQPs can take over 6 seconds. (Obviously, CPU availability on the machine is also a factor in determining how long the filtering operation takes.)

To cope with this situation, the WIS process contains a pool of queue filtering threads that can be used to filter work queues more quickly. The following process attributes allow you to configure how and when these threads are used:

- WIS_FILTER_THREAD_BOUNDARIES allows you to define when a work queue should be split into multiple "blocks" of work for filtering purposes. You can define up to 4 threshold values for the number of work items in a queue. As each threshold is passed, an additional block of filtering work is created, which will be handled by the first available queue filtering thread.

- WIS_FILTER_THREAD_POOL_SIZE allows you to define the number of queue filtering threads in the pool. These threads are used to process all additional filtering blocks generated by the WIS_FILTER_THREAD_BOUNDARIES thresholds. Increasing the number of threads in this pool allows more blocks of filtering work to be processed in parallel, but at the cost of increasing the CPU usage of the WIS process.

For example, consider the following scenario:

- A work queue contains 180000 work items.

- `WIS_FILTER_THREAD_BOUNDARIES` has been set to create additional filtering blocks when a queue contains 100000 and 150000 work items.

- The `WIS` process receives 5 RPC requests to filter the queue.

Each RPC request on the queue generates 2 additional filtering blocks (each of 60000 work items). The first filtering block is still handled by the RPC processing thread that is handling the RPC request.

The 5 RPC requests therefore generate 10 blocks of additional filtering work to be processed by the queue filtering threads. If `WIS_FILTER_THREAD_POOL_SIZE` is set to:

- 10 or more, each block is immediately filtered by one of the queue filtering threads.

- less than 10, some blocks will have to be queued until a queue filtering thread is available to process them.

> When altering the `WIS_FILTER_THREAD_BOUNDARIES`, `WIS_FILTER_THREAD_POOL_SIZE` or `RPC_SVR_CONTROL` process attributes, you should bear in mind that the more RPC processing threads there are and the larger the number of work items in a queue, the more threads in the queue filtering thread pool will be used by a single RPC request to filter a queue.

Configuring Queue Updates

The queue update thread performs two functions:

- It goes through all the queues handled by the `WIS` process and checks for expired deadlines, priority escalations, redirection work, new or purged work items and so on.

- It calls the `WQS` process for a new queue to handle when required (i.e. when the `WQS` process has processed a `MOVESYSINFO` event and sent out an `SE_WQSQUEUE_ADDED` event to the `WIS` process).

The queue update thread performs updates for `WIS_UPDATE_LENGTH` seconds or until all queues have been processed, at which point it will go to sleep for `WIS_UPDATE_PERIOD` seconds. If the thread hasn't gone through all the queues within the WIS_UPDATE_LENGTH time then it will start from the point it finished at on its previous update.

> The queue update thread performs its work independently of and concurrently with the RPC processing threads. In pre-10.4 versions of iProcess Engine, where the `WIS` process was single-threaded, the `WIS` process had to switch between processing RPC requests and updating work queues.

Configuring When WIS Processes Cache Their Queues

The WQS/WIS processes maintain an in-memory cache of the information that each WIS process contains about each work queue that it is handling. Caching this information allows the WIS processes to respond quickly to RPC requests from client applications.

However, the amount of time that a WIS process takes to start up is heavily influenced by the number of queues that it has to cache, the number of work items in the queue, the number of CDQPs defined in the queue, and the general load on the machine.

You can monitor how long a WIS process is taking to start up using the plist -wV command, which is under the *SWDIR*\util directory (see Monitoring the WIS Processes on page 411). The LastCacheTime column shows the number of milliseconds that the WIS process took to cache each queue when it was last cached.

You can tailor this behavior to suit your particular requirements by configuring work queues to be cached either:

• when they are first handled by a WIS process. This will be either when iProcess Engine starts up, or for queues that are added when the system is running, after a MoveSysInfo event request.

    or

• when they are first accessed by a client application.

You control which queues are cached when they are first handled by a WIS process by using a combination of the WISCACHE queue attribute and the WIS_CACHE_THRESHOLD or RESTART_WIS_CACHE_THRESHOLD process attributes. When the WIS process first handles a queue, it checks the value of the queue's WISCACHE attribute:

• If WISCACHE is set to YES, the WIS process caches the queue (irrespective of how many work items the queue contains).

• If WISCACHE has not been created, or has not been set, the WIS process only caches the queue if the queue contains a number of work items that equal or exceed the value of the WIS_CACHE_THRESHOLD or RESTART_WIS_CACHE_THRESHOLD process attributes.

When the WIS process starts up, it reads the number of work items in each work queue from the total_items column in the wqs_index database table. This table is populated from the contents of the WQS/WIS shared memory, which is written to the database every WQS_PERSIST_SHMEM seconds.

Any queue that is not cached now will be cached when it is first accessed by a client application.

Note that:

- Queues are cached by a pool of threads in the WIS process. You can configure the number of threads in this pool by using the WIS_CACHE_POOL_SIZE

- When an RPC client application makes an RPC call to a work queue that has not already been cached, the WIS process immediately begins caching it. If the value of the WIS_CACHE_WAIT_TIME process attribute is reached and the work queue has still not been cached, the WIS process returns an ER_CACHING error to the client application.

  If the RPC client application is a TIBCO iProcess Workspace (Windows) session, the user will see the following message in the right-hand pane of the Work Queue Manager, instead of the expected list of work items:

  ```
  The Work Item Server (WIS) is fetching the work items for this
  queue. Please wait...
  ```

- The WISMBD process also makes RPC calls to WIS processes to pass instructions from the BG processes. If the WISMBD process receives an ER_CACHING error from the WIS process it retries the connection a number of times. If the attempt still fails, it requeues the message and writes a message (with ID 1984) to the sw_warn file, which is located in the *SWDIR*\logs directory.

  See *TIBCO iProcess Engine System Messages Guide* for more information about this message.

- Configuring more work queues to be cached when they are first accessed obviously improves the startup time for the WIS processes, but the potential cost is that users may have to wait to access their queues while they are being cached.

Setting the WISCACHE Attribute for a Queue

The WISCACHE queue attribute does not exist by default. If you wish to use it, you must first create it and then assign a value for it to any queues that you want to use it. To do this:

1. Start the Process Administrator, and then start the User Manager. (See "Using TIBCO iProcess Administrator" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.)

2. Define a new attribute called WISCACHE. This should have a Type of Text, with a Length of 4.

   See "Adding a New Attribute" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

3. Assign a value of YES to WISCACHE for each queue that you want to be cached when the WIS process first handles it (irrespective of how many work items the queue contains).

   All other queues (for which WISCACHE is not set) will be cached either when the WIS process first handles it or when they are first accessed by a client application, depending on the value of the WIS_CACHE_THRESHOLD process attribute.

   See "Setting User Values for an Attribute" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

4. Save your changes, exit from User Manager and perform a MoveSysInfo to register your changes on iProcess Engine.

   See "Moving System Information" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

Configuring CDQP Updates

CDQPs allow values from case data to be used by client applications to sort, display and filter work items lists, and to find specific work items.

When the WIS process starts up it caches all the CDQP definitions that are used by the queues it is handling, and uses the cached values when displaying CDQPs in its work queues.

> The WIS process obtains the field values of fields that are defined as CDQPs from the pack_data database table.

You can change existing CDQP definitions or create new ones by using the swutil QINFO command. By default, you then have to restart iProcess Engine to allow the WIS process to pick up the changed definitions and update its work queues with them.

However, you can dynamically pick up changes to CDQP definitions without having to restart iProcess Engine, by using the PUBLISH parameter with the QINFO command. This publishes an event that signals that updated CDQP definitions are available.

When the WIS process detects this event its CDQP update thread wakes up and updates the CDQP definitions for all work items in its queues. Work items are updated in batches, the size of which is determined by the value of the WIS_CDQP_DATA_RECACHE_BATCH process attribute.

See "Case Data Queue Parameters" in *TIBCO iProcess swutil and swbatch Reference Guide* for more information about CDQPs and the QINFO command.

Troubleshooting Work Queues

This section provides troubleshooting information for when users have problems accessing work queues.

When experiencing problems with the WIS processes, there are three common error messages that appear in the Work Queue Manager:

`Failed to Open Work Item List for Queue`

or

`Work Queue Servers Not Responding`

when moving between queues in Work Queue Manager, and

`That Facility is Not Available`

when attempting to start a case.

In these examples, the client is unable to contact the WIS or WQS process to find out what queues or work items exist. The problem is that users are unable to access their work items in the queues because the work queues are grayed out in Work Queue Manager.

To resolve the problem, try one of the following:

• Use `plist -w` to check the status of each WIS process.

• Check to make sure that the WQS and WIS processes are running:

— On Windows, use the Processes tab of the Task Manager.

— On UNIX, run the `ps -fe` command.

The processes are named `wisrpc` and `wqsrpc`.

• Use the `swsvrmgr` Process Sentinels command line utility, which is under the *SWDIR*\util directory, to report the status of the processes. See View Process Status on page 119.

• Check the `sw_warn` and `sw_error` files, which is located in the *SWDIR*\logs directory, for any error messages to see if any problems have been logged. See iProcess Engine Log Files on page 439 for more information.

• If you cannot resolve your work queue problem, contact TIBCO Support.

Chapter 13 **Administering Case Data Normalization**

This chapter describes case data normalization and how to administer it on iProcess Engine.

Topics

- Overview, page 422
- Enabling Case Data Normalization, page 423

Overview

Case data normalization makes case data searching more efficient and therefore faster by populating the column `field_value_N` in the `case_data` table with data from the `field_value` column. Some previous versions of iProcess Engine did not support case data normalization, so when you install/upgrade iProcess Engine, you are prompted to enable this feature.

⚠️ If you are using TIBCO iProcess Objects to perform case searches, TIBCO recommends that you enable case data normalization. If you do not, although you will be able to view and start procedures, you will not be able to see the cases until you normalize the data.

Case data normalization is controlled by the following:

- the global process attribute `NORMALISE_CASE_DATA` (which enables case data normalization system-wide).

- the `normalise_data` column on the `proc_index` table (which indicates whether case data normalization is enabled for a specific procedure). This is controlled by the Case Data Normalization flag on the Status tab of the Properties dialog (see "Setting and Viewing Status Information" in *TIBCO iProcess Modeler Procedure Management*) or by the Case Data Normalization Utility (see).

Enabling Case Data Normalization

To enable/disable case data normalization by either:

- Responding to the prompt during an installation or upgrade:

    — If you enable this feature, the process attribute `NORMALISE_CASE_DATA` is set to 1 and all existing case data is normalized. Future cases of all procedures are also normalized.

    — If you disable the feature, the process attribute `NORMALISE_CASE_DATA` is set to 0 and existing and future case data is not normalized.

- Setting the process attribute `NORMALISE_CASE_DATA` using the `swadm` utility, which is under the *SWDIR*\util directory (see NORMALISE_CASE_DATA on page 171).

When you have enabled case data normalization, you can normalize case data by either:

- Using the Case Data Normalization Utility as described in the following section. This utility changes the setting of the `normalise_data` column on the `proc_index` table. Using this utility you can normalize case data either:

    — system wide, or

    — on a per-procedure basis.

- Selecting the "Normalise Case Data" check box in the Status tab of the Properties dialog to enable the feature for a specific procedure. This check box is only enabled if `NORMALISE_CASE_DATA` is set to 1 and the procedure has no cases. For more information, see "Setting and Viewing Status Information" in *TIBCO iProcess Modeler Procedure Management*.

Using the Case Data Normalization Utility

The Case Data Normalization Utility allows you to normalize existing case data; either system-wide or on a per-procedure basis. For example, you may have disabled case data during an upgrade because of the large amount of case data involved. After the upgrade you can use the Case Data Normalization Utility to convert the case data during off-peak hours.

You can also disable or enable case data normalization on a per-procedure basis with the "Normalise Case Data" check box on the Status tab of the Procedure Properties dialog.

Before using the Case Data Normalization Utility, ensure that the global process attribute NORMALISE_CASE_DATA is set to 1, using the swadm utility if necessary (see Using SWDIR\util\swadm to Administer Process Attributes on page 142). This enables case data normalization and allows you to use the Case Data Normalization Utility.

The Case Data Normalization Utility is located in the following directory:

SWDIR\util

The command you enter to use the utility has the following format:

swnormcd [/U] [/T *nnn*] */A | procedure_list | /F control_file*

where:

- /U indicates that you want to disable case data normalization. Note that disabling case data normalization does not delete the data held in the field_value_N column in the case_data table. New cases of procedures will not use case data normalization and if you are using TIBCO iProcess Objects, new cases will not appear in case data searches.

- /T *nnn* specifies the number (*nnn*) of concurrent threads for case data normalization. The default is 10. Use this parameter to improve performance when normalizing large amounts of data.

- /A indicates that existing case data should be normalized for all procedures. The normalise_data column on the proc_index table is set to 1 and new cases of procedures are normalized.

⚠️ Normalizing large amounts of case data can take a significant amount of time.

- *procedure_list* is either the name of a procedure, or a list of procedures separated by white space.

- */F control_file* specifies the name of a file that contains procedure names separated by white space.

Examples

This command disables case data normalization for the hiring procedure. Any new cases of this procedure will not use case data normalization and will not appear in searches using TIBCO iProcess Objects.

swnormcd /U hiring

This command enables case data normalization for all procedures and normalizes existing case data.

swnormcd /A

This command enables case data normalization for the procedures listed in the file `proclist.txt` and converts any existing case data.

```
swnormcd /F proclist.txt
```

Chapter 14 **Managing EAI Step Server Plug-ins**

This chapter explains how to use the `sweaireg` command line utility, which is under the *SWDIR*\util directory, to manage the EAI step server plug-ins.

Topics

Overview

To function correctly, each EAI step type in TIBCO iProcess Modeler requires an associated EAI server plug-in to be installed and registered on every server in the TIBCO iProcess Engine node cluster that runs background processes.

The following plug-ins are automatically installed when you install iProcess Engine:

- TIBCO iProcess COM Server Plug-in (on Windows only; during installation you choose whether or not to register the Plug-in.)

- TIBCO iProcess Script Server Plug-in

- TIBCO iProcess Database Server Plug-in

- TIBCO iProcess EMail Server Plug-in

- TIBCO iProcess Plug-in SDK

For installation of these plug-ins, see iProcess Engine installation guide for your platform/database. For other EAI plug-ins, refer to the specific EAI server plug-in installation guide for installation information.

See "Using Enterprise Application Integration (EAI) Steps" in *TIBCO iProcess Modeler Integration Techniques* for information about how to use EAI steps in your procedures.

You can, however, design procedures using a EAI client plug-in for which you have not installed the corresponding EAI server plug-in. This is useful if you want to prepare for porting a procedure to a different platform in the future. If you use an EAI client plug-in without having the relevant EAI server plug-in installed, the EAI step that you create is not processed at run time and an error message is displayed. The error informs you that the step is incompatible with the connected database, and so will not be processed.

Although the installation, upgrading, and registration of most EAI plug-ins is handled automatically by iProcess Engine installation, you can use this utility to:

- Register or re-register an EAI server plug-in - see Register/Re-register (upgrade) an EAI Plug-In on page 430.

- Unregister an EAI server plug-in - see Unregister (Remove) an EAI Plug-In on page 432.

- Modify parts of an existing EAI server plug-in's registry entry - see Modify an Existing EAI Plug-In Entry on page 433.

- list EAI server plug-in registry entries - see List Existing EAI Plug-In Registry Entries on page 434

- manually request iProcess Engines to reload EAI server plug-ins - see Reload an EAI Plug-in on page 436

- get the release version of an EAI server plug-in - see Get Release Version Stored in EAI Plug-In on page 437.

See Possible Errors When Using sweaireg on page 438 for information about solving possible errors you might encounter when using sweaireg.

To run sweaireg, you must be logged in to iProcess Engine as an Administrator or (on UNIX) as a background user or root user.

Register/Re-register (upgrade) an EAI Plug-In

The REG command installs or upgrades an EAI server plug-in. This command is automatically used by the EAI server plug-in's installation script. Therefore, you only need to use this command if you need to install a plug-in for a given operating system in a shared location. You would then use the REG command to register the plug-in on all your servers.

This command automatically detects if this is the first registration of the plug-in or an upgrade for a given EAI step type.

This command does not install the plug-in file. Follow the installation procedure described in the specific EAI plug-in documentation.

Before using this command, you must ensure that the run-time loading requirements are met because the library is loaded when this command is used. For example, if the library uses other system shared libraries, they must be defined in the server's shared library path.

Syntax

```
sweaireg REG eai_type_name [-m machine_name] -l library [-i init_params]
[-y]
```

where:

- *eai_type_name* is the short name of the EAI step type handled by the server plug-in. This can be a text string up to 20 characters.

- *machine_name* is the optional name of the server in the iProcess Engine node cluster on which the plug-in is to be registered. If this value is omitted, the default is the server on which the command is being run. This can be a text string up to 256 characters.

- *library* is the path and file name for the plug-in. This is a text string of up to 256 characters.

- *init_params* is an optional value that can be used for any plug-in specific initialization parameters. Refer to the documentation for each plug-in to see what values can be used. If this is omitted and you do a re-registration, the existing parameters will be preserved. This can be a text string up to 1024 characters.

- -y can be used to automatically answer yes to all the sweaireg command prompts so the command is run immediately without displaying the prompts.

For a first registration, the values are written to the EAI run-time plug-in registry (the `eai_run_plugins` table). For a re-registration, the following message is displayed:

Re-register *eai_step_name* run time plug-in *version current_release_version* with *version install_set_release_version*? (y/n)

If you enter Y, the EAI plug-in registry is updated. If you enter N, no changes will be made.

After you have registered the plug-in, you must set the EAI_NEEDS_MSDTC process attribute if the plug-in needs to use the Microsoft Distributed Transaction Coordinator (MSDTC). If you don't do so, EAI steps using the plug-in may not function correctly or in a fully transactional manner.

For more information about process attributes and how to set them, see Administering Process Attributes on page 141.

Example   To register the EAI server plug-in for eaidb on the server called hercules, enter the following:

```
sweaireg REG eaidb -m hercules -l SWDIR\lib\eaidb -y
```

Before exiting, the following status is displayed:

```
EAI Run-Time Plug-in Registration Successful:
EAI Type: EAIDB
Machine: Hercules
Version: 1.0
Library:$SWDIR\lib\eaidb
Init Params:
```

Unregister (Remove) an EAI Plug-In

Use the UNREG command to remove an EAI step type entry from the plug-in registry. This results in the EAI step type being unregistered from the server so the server will not be able to process any EAI steps that use this server plug-in.

Syntax   sweaireg UNREG *eai_type_name* [-m *machine_name*] [-y]

where:

- *eai_type_name* is the short name of the EAI step type handled by the plug-in. This can be a text string up to 20 characters.

- *machine_name* is the optional name of the server in the TIBCO iProcess Engine node cluster on which the plug-in is registered. If this value is omitted, the default is the server on which the command is being run. This can be a text string up to 256 characters.

- -y can be used to automatically answer yes to all the sweaireg command prompts so the command is run immediately without displaying the prompts.

After running the command, the following prompt is displayed:

```
Unregister EAI Run-Time Plug-In
EAI Type EAI Type Name
Machine machine name ID:xx
Version Release Version
Library: library path and name
Init Params: Initialisation parameters

OK to unregister? (y/n)
```

If you choose Y, the plug-in's registry entry is removed. The following message is displayed:

```
EAI Run-Time Plug-In Registration successfully removed
```

Example   To unregister the eaidb plug-in from the server called hercules (the computer on which you are running this command), enter the following:

sweaireg UNREG eaidb

When prompted, enter Y to proceed with un-registering the server plug-in.

Modify an Existing EAI Plug-In Entry

Use this command to modify the server plug-in path or initialization parameters in the EAI plug-in's registry entry.

Syntax

```
sweaireg MOD eai_type_name [-m machine_name] [-l library]
[-i init_params][-y]
```

where:

- *eai_type_name* is the short name of the EAI step type handled by the plug-in. This can be a text string up to 20 characters.

- *machine_name* is the optional name of the server in the iProcess Engine node cluster on which the plug-in is registered. If this value is omitted, the default is the server on which the command is being run. This can be a text string up to 256 characters.

- *library* is the path and file name for the server plug-in. This is a text string up to 256 characters.

- *init_params* is an optional value that can be used for any plug-in specific initialization parameters. Refer to the documentation for your specific plug-in to see what values can be used. If this is omitted and you do a reregistration, the existing parameters will be preserved. This can be a text string up to 1024 characters.

- -y can be used to automatically answer yes to all the sweaireg command prompts so the command is run immediately without displaying the prompts.

Example If you move the plug-in files to a different directory (from *SWDIR*\lib to *SWDIR*\lib\version1), you can update the path to point to the new location by entering:

sweaireg MOD eaidb -l SWDIR\libpath\version1\eaidb

This will make the change for the computer on which you are running this command. You need to do this for any other servers using this server plug-in.

List Existing EAI Plug-In Registry Entries

Use this command to list all of the EAI plug-in registry entries.

Syntax   sweaireg LIST [*eai_type_name*] [-m machine *name*] [-x]

where:

- *eai_type_name* is the short name of the EAI step type handled by the plug-in. This can be a text string up to 20 characters.

- *machine_name* is the optional name of the server in the iProcess Engine node cluster on which the plug-in is to be registered. This can be a text string up to 256 characters.

- -x is used to output the listing in a format suitable for script processing (a ; separated list of parameters on a single line). This is optional, and if omitted, the results are provided in a user-friendly format.

The entries listed are determined by the EAI type name and machine name:

Parameters Used	Result
Neither *eai_type_name* or *machine_name* are specified.	All registry entries are listed.
If both are specified.	The single registry entry for that EAI type on the given computer is listed.
If only *eai_type_name* is specified.	The registry entry for the given EAI type is listed for each machine on which it is registered.
If only *machine_name* is specified.	The registry entries for all EAI types registered on the given machine are listed.

The iProcess BusinessWorks Plug-in is not displayed in the list of the EAI plug-in registry entries.

Example   To list the EAI plug-in registry entries on the server called hercules, enter the following:

```
sweaireg LIST -m hercules
```

The following is a sample output:

```
EAI Type: eaidb On Machine: Hercules
Version: 1.0
Library: $SWDIR\lib\eaidb
Init Params:
```

Reload an EAI Plug-in

When an EAI plug-in entry is re-registered or modified, iProcess Engine automatically reloads the plug-in. However, you might want to manually reload an EAI server plug-in using this command if:

- the EAI server plug-in is failing

- the initialization parameters specify a configuration file and the contents of that file has changed

Syntax   `sweaireg RELOAD` *eai_type_name* [`-m` *machine_name*]

where:

- *eai_type_name* is the short name of the EAI step type handled by the plug-in. This can be a text string up to 20 characters.

- *machine_name* is the optional name of the server in the iProcess node cluster on which the plug-in is to be registered. If this value is omitted, the default is the server on which the command is being run. This can be a text string up to 256 characters.

Example   To reload the eaidb plug-in on the server called `hercules`, enter the following:

`sweaireg RELOAD eaidb -m hercules`

If the command is successful, the following message is displayed:

```
Background reload and re-initialisation requested for eaidb plug-in
on machine hercules
```

Get Release Version Stored in EAI Plug-In

Use the GETRELVERS command to output the release version in the given EAI server plug-in. This is provided so that the plug-in installation script can display the release version of the plug-in before it installs it. This enables version upgrades to be performed.

Before using this command, you must ensure that the run-time loading requirements are met because the plug-in library is loaded when this command is used. For example, if the plug-in uses other system shared libraries, they must be defined in the server's shared library path.

Syntax   sweaireg GETRELVERS -l *library*

where *library* is the path and file name for the server plug-in. This is a text string up to 256 characters.

Example   To extract the release version from the EAI Database library called eaidb in the *SWDIR*\eai directory, you would enter the following command:

sweaireg GETRELVERS -l \eai\eaidb

Possible Errors When Using sweaireg

This section details some of the typical errors you might get when using the `sweaireg` utility.

FORMAT:sweaireg REG *eai_type_name* [-m *machine_name*] -l *library* [-i *init_params*]

You have entered an invalid command line or there are missing parameters or options. Re-enter the command making sure you include all the required parameters and options.

Invalid Parameter: *parameter_name*

The parameter you have entered is incorrect. Re-enter the command line with a valid parameter.

Error connecting to the iProcess Engine

Your iProcess Engine `node` environment variables are not set up correctly i.e. check *SWDIR* and any other environments required for the system are set up correctly and that Oracle is running.

Error accessing the EAI run-time plug-in registry

There is an error accessing or updating the plug-in registry. For example, the database might not be accessible. An error may also be logged to the `sw_warn` file, which is under the *SWDIR*\`logs` directory. See iProcess Engine Log Files on page 439 for more information.

Unexpected Error

An internal system error has occurred. Contact TIBCO Support for help.

Failed to load library: *system defined error message*
Failed to load EAIRun_GetReleaseVers() from library: *library_path*

You need to make sure that the given library path is correct and any related run-time libraries have been installed and set up correctly.

Appendix A   **iProcess Engine Log Files**

The iProcess Engine node automatically produces the following log files in the *SWDIR*\logs directory.

Table 10   iProcess Engine Log Files

Log File	Description
sw_error.log	This file is created if a serious error occurs that needs to be investigated immediately, and the error occurs at a different date than the date the last error is logged. In the mean time, the previous log file will be archived as sw_error*timestamp*.log, where the *timestamp* variable is the date when that log is generated. See *TIBCO iProcess Engine System Messages Guide* for detailed information about the system errors and warning messages that can be returned by iProcess Engine.
sw_error*timestamp*.log	This file is the archived sw_error.log, where the *timestamp* variable is the date when the log is generated.
sw_warn.log	This file is created if an error occurs that needs to be dealt with, but is not serious enough to prevent iProcess from being used, and the error occurs at a different date than the date the last error is logged. In the mean time, the previous log file will be archived as sw_warn*timestamp*.log, where the *timestamp* variable is the date when that log is generated. See *TIBCO iProcess Engine System Messages Guide* for detailed information about the system error and warning messages that can be returned by iProcess Engine.
sw_warn*timestamp*.log	This file is the archived sw_warn.log, where the *timestamp* variable is the date when the log is generated.
iapjms_java.log	This file is created by the IAPJMS process (if enabled). By default any warning or error messages produced by the IAPJMS process are written to this file.
userinfo.log	An entry is added to this file whenever user information is updated on the system. For example: `staffusr updated by swadmin - Tue Dec 7 17:27:15 2001`

Table 10   iProcess Engine Log Files (Cont'd)

Log File	Description
`roleinfo.log`	An entry is added to this file whenever role information is updated on the system. For example: `staffrol updated by swadmin - Tue Dec 7 17:27:36 2001`
`swjmx_java.log`	This file is created by the JMX engine (which is part of the `RPC_TCP_LI` process). By default any warning or error messages produced by the JMX engine are written to this file.
`wiswarn.log`	An entry is added to this file whenever the server shuts down. For example: `2001/12/ 7 17:58 wisrpc : normal shutdown`
`wqswarn.log`	An entry is added to this file whenever the server shuts down. For example: `2001/12/ 7 17:54 wqsrpc: normal shutdown`
`rpcport.log`	This text file is only used when port and/or RPC number ranging is enabled (see Administering Firewall Port Ranges on page 363). The file contains entries that show the resource allocation for the ports and RPC numbers used. It records the following events: • Startup of the port/RPC resource allocation service • Shutdown of the port/RPC resource allocation service • Allocation of a port/RPC number • Release of a port/RPC number • Failure to re-bind a released port • Successful re-binding of a previously failed port • Errors in the allocation/release of a port/RPC number

Appendix B   **System Backup Guidelines**

This appendix provides guidelines for the safe backup and recovery of iProcess workflow data.

A system backup consists of:

- backing up your SQL/Oracle database. The iProcess database instance contains all the iProcess case data.

- backing up configuration files on iProcess Engine and client. This will prevent you having to record what configuration changes you have made.

Backup and Recovery of iProcess Case Data

Because all iProcess case data is stored in the SQL/Oracle database, you need to make sure that your database administrator makes regular backups. If the database gets corrupted or the system goes down, the database administrator can use the database recovery tools to recover the iProcess case data.

Backup and Recovery of iProcess Engine Configuration Files

TIBCO recommends that you also backup the following:

- any configuration files that you change, for example `staffcfg`, which is under the *SWDIR*\etc directory.

- any "use" files in *SWDIR*\nodename.n\use.

Appendix C **iProcess Engine Directory Structure**

This appendix describes the physical location of iProcess Engine's programs and data on the computer hosting the server.

The directories are described relative to the iProcess System directory *SWDIR*. If there are multiple iProcess Engine installations on the computer, each must have a unique *SWDIR*. Each computer in a node cluster will have iProcess Engine directories and files.

Table 11   iProcess Engine Directory (Sheet 1 of 3)

Directory	Description
SWDIR\bin	Contains system executables and the swutil utility program.
SWDIR\cms	Contains failed mail items for remote nodes. **NOTE**: This directory is not currently used by iProcess Engine.
SWDIR\cms.rx	CMS receive folder. **NOTE**: This directory is not currently used by iProcess Engine.
SWDIR\cms.tx	CMS transmit folder. **NOTE**: This directory is not currently used by iProcess Engine.
$*SWDIR*/eaidist	(UNIX only) Contains TIBCO iProcess Engine Server Plug-ins.
SWDIR\etc	Contains iProcess executables, message files and configuration files. It also contains the *language*.lng sub-directory, which contains language dependent message, and configuration files, where *language* is the language for this installation. There is one directory per installed language.
SWDIR\examples	Contains the EAI step procedure examples. This directory only exists if you have installed the examples for the TIBCO iProcess Engine Server Plug-ins.

Table 11   iProcess Engine Directory (Sheet 2 of 3)

Directory	Description
SWDIR\jar	Contains JAR files required by the IAPJMS process.
SWDIR\java	Contains the Java JRE distributed with iProcess Engine.
SWDIR\lib	(Windows only) Contains shared libraries such as fil.so and TIBCO iProcess Engine Server Plug-in software.
$*SWDIR*/libs	(UNIX only) Contains shared libraries such as fil.so and TIBCO iProcess Engine Server Plug-in software.
SWDIR\logs	Contains system log files.
SWDIR\mscluster	(Windows only) Contains the mscluster tool used to add iProcess Engine components to secondary machines in a Windows cluster environment.
SWDIR\pro\sww.uid	Contains one file per user currently logged in.
SWDIR\queues	Contains a *username* directory for each user defined on this installation. *username* is the iProcess work queues directory for the user (or group) *username*.
SWDIR\rpc	(Windows only) Contains RPC executables.
SWDIR\schema	Contains XML schema definitions.
SWDIR\sdks	Contains the following iProcess Engine Software Development Kit (SDK) sub-directories: deploysdk - for internal use only. eaisdk - the TIBCO iProcess Plug-in SDK. See *TIBCO iProcess Plug-in SDK User's Guide* for more information about this SDK. salsdk - the TIBCO Application Layer SDK (also known as the Staffware Application Layer SDK). See the saldsk\docs directory for more information about this SDK. uvapisdk - the TIBCO iProcess User Validation API. See *TIBCO iProcess User Validation API User's Guide* for more information about this SDK.
$*SWDIR*/seo	(UNIX only) Contains iProcess Objects Server configuration files.

Table 11   iProcess Engine Directory (Sheet 3 of 3)

Directory	Description
SWDIR*nodename*.n\use	Contains Use files defined on this node.
SWDIR\sysinfo	**NOTE**: This directory is not currently used by iProcess Engine.
SWDIR\tomcat	Contains the Apache Tomcat application server distributed with iProcess Engine.
SWDIR\tsys	Temporary editing area.
SWDIR\uninstll	(Windows only) Uninstall directory.
SWDIR\util	Contains utility programs and XFR procedure files.

Appendix D **Understanding Audit Trails**

An audit trail is a predefined iProcess report that provides a detailed log of all transactions for an individual case of a procedure.

There are two types of audit trail message:

- System-defined. The table below describes the system-defined messages.

- User-defined. See SWDIR\etc\language.lng\auditusr.mes on page 37 for more information about using this file to define user-defined audit trail messages.

Audit trail messages can be used in two ways:

- You can view a detailed audit trail for any iProcess case to see how a case is progressing or has progressed using the Case Administration tool. See "Administering Cases" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

- You can configure iProcess Engine to publish audit trail messages to an external application. This enables an external application to monitor important business events during the processing of cases. See "Configuring Activity Monitoring" in *TIBCO iProcess Modeler Integration Techniques* for more information and Administering Activity Monitoring and Work Queue Delta Publication on page 379 for more information.

The following table describes the system-defined messages that can be displayed in your audit trails and what they mean.

Table 12   Audit Trail Messages (Sheet 1 of 12)

Message ID	Message	Description
000	`Case started by` *UserName*	The case of a procedure has been started where *UserName* is the name of the iProcess user who has started the case. See "Starting Cases" in *TIBCO iProcess Workspace (Windows) User's Guide* for more information.
001	*StepDescription* `processed to` *UserName*	The *StepDescription* work item has been processed to the *UserName* user. See "Opening and Processing a Work Item" in *TIBCO iProcess Workspace (Windows) User's Guide* for more information.

Table 12   Audit Trail Messages (Sheet 2 of 12)

Message ID	Message	Description
002	*StepDescription* `released by` *UserName*	The *StepDescription* work item has been released by the *UserName* user. See "Opening and Processing a Work Item" in *TIBCO iProcess Workspace (Windows) User's Guide* for more information.
003	`Deadline for` *StepDescription* `expired for` *UserName*	The deadline set for the *StepDescription* work item has expired for the *UserName* user. If the deadline has expired, then the deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
004	*StepDescription* `forwarded to` *UserName*	An iProcess user has forwarded the *StepDescription* work item from their work queue to another iProcess user's work queue. The *UserName* is the name of the iProcess user who has received the work item in their work queue.
		See "Enabling Steps to be Forwarded" in *TIBCO iProcess Modeler Basic Design* for more information.
006	`Error -` *StepDescription* `not found`	The *StepDescription* work item cannot be found. You may see this message if, for example, the case has been purged and so the work item no longer exists.
		Check the `sw_warn` or `sw_error` log files to see if any error messages were logged. See *TIBCO iProcess Engine System Messages Guide* for more information.
007	`Case terminated abnormally`	The case has terminated abnormally. You may see this message if there has been a system error that has caused the case to terminate abnormally.
		Check the `sw_warn` or `sw_error` log files to see if any error messages were logged. See *TIBCO iProcess Engine System Messages Guide* for more information.
008	`Case terminated prematurely by` *UserName*	The case of a procedure has been terminated prematurely by the *UserName* user. This means that not all the steps in the case have been completed because the case was terminated prematurely.
		See "Closing Cases" in *TIBCO iProcess Workspace (Windows) Manager's Guide* for more information.

Table 12   Audit Trail Messages (Sheet 3 of 12)

Message ID	Message	Description
009	Case terminated normally	The case has completed processing all its steps and, therefore, it has terminated normally.
011	*StepDescription* released from queue by *UserName*	This message is obsolete. If this message appears in an audit trail, contact TIBCO Support for further assistance.
012	There is no message defined for this number.	
013	*StepDescription* withdrawn from *UserName*	The *StepDescription* work item has been withdrawn from the *UserName* queue because the deadline expired or as the result of a withdraw action. If the step has been withdrawn because a deadline has expired, the deadline actions will be processed. See "Withdrawing Steps from the Procedure" in *TIBCO iProcess Modeler Basic Design* for more information.
014	*StepDescription* resent to *UserName*	The *StepDescription* work item has been resent to the *UserName* user. See "Resending work items" in *TIBCO iProcess swutil and swbatch Reference Guide* for more information.
015	*StepDescription* event issued by *UserName*	The *StepDescription* event step has been issued by the *UserName* user. See "Using Events" in *TIBCO iProcess Modeler Integrating Techniques* for more information.
016	Sub-Case started from *StepDescription*	A case of a sub-procedure has been started from the *StepDescription* step. See "Defining and Using Sub-procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.
017	Sub-case started from *StepDescription* completed	A case of a sub-procedure that was started from the *StepDescription* step has terminated normally. See "Defining and Using Sub-procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.

Table 12   Audit Trail Messages (Sheet 4 of 12)

Message ID	Message	Description
018	Sub-case started from *StepDescription* terminated abnormally	A case of a sub-procedure has terminated abnormally where *StepDescription* is the description of the step. You may see this message if a system error has caused the sub-case to terminate abnormally.
		Check the sw_warn or sw_error log files to see if any error messages were logged. See *TIBCO iProcess Engine System Messages Guide* for information.
		See "Defining and Using Sub-procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.
019	Deadline for sub-case started from *StepDescription* expired	The deadline set for the *StepDescription* step that is calling the sub-case has expired. This causes the sub-case started from this step to be closed. This means that the deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
020	Sub-case started from *StepDescription* closed	The *StepDescription* step that called the sub-case has been withdrawn because the deadline has expired. This causes the sub-case started from this step to be closed. This means that the deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
021	*StepDescription* redirected to *UserName*	The *StepDescription* work item has been redirected to another user's work queue. *UserName* is the name of the iProcess user who has received the work item in their work queue. See "Redirecting Work Items" in *TIBCO iProcess Workspace (Windows) User's Guide* for more information.
022	Case Suspended by *UserName*	The case has been suspended by the *UserName* user. See "Suspending the Flow of a Case" in *TIBCO iProcess Modeler Integration Techniques* for more information.
023	Case Resumed by *UserName*	The case has been resumed by the *UserName* user. See "Suspending the Flow of a Case" in *TIBCO iProcess Modeler Integration Techniques* for more information.

Table 12   Audit Trail Messages (Sheet 5 of 12)

Message ID	Message	Description
024	*StepDescription* `Case Jump` `by` *UserName*	The *UserName* user has caused the case to jump to this *StepDescription* step. See "Using GOTOSTEP to Simplify Procedure Routing" in *TIBCO iProcess Modeler Basic Design* for more information.
025	*SubProcedureDescription* `Sub-Case started` `(using array element` *StepName*`)`	A case of a *SubProcedureDescription* sub-procedure has been started by a *StepName* array element step. See "Using Array Fields" in *TIBCO iProcess Modeler Advanced Design* for more information.
026	`Task count` *StepName* `received for` *Status:StepName*	The external application has informed the iProcess Engine of all the processes that need to be completed before the graft step can complete, where:
		StepName is the name of the graft step
		Status:StepName is the current status of the graft step and the graft step name.
		See "Graft Step Task Count" in *TIBCO iProcess Modeler Integration Techniques* for more information.
027	`Task count` `decremented for` *Status:StepName*	One of the processes grafted to this *StepName* step has completed. *Status* is the current status of the graft step.
		See "Graft Step Task Count" in *TIBCO iProcess Modeler Integration Techniques* for more information.
028	`Sub-Case grafted to` *StepDescription*	The sub-case has been grafted to the *StepDescription* graft step. See "Using Graft Steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
029	`External process` *ExternalProcessName* `grafted to` *StepDescription*`.`	The external process has been grafted to the *StepDescription* graft step. *ExternalProcessName* is the name of the external process. See "Using Graft Steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
030	*StepDescription* `initiated`	The *StepDescription* graft step has been initiated by the external system. See "Using Graft Steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.

Table 12   Audit Trail Messages (Sheet 6 of 12)

Message ID	Message	Description
031	External process *ExternalProcessName* released	The external process has completed. *ExternalProcessName* is the name of the external process. See "Using Graft Steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
032	*StepDescription* released, all tasks complete	The *StepDescription* graft step has been released because all the tasks grafted to the graft step are complete. See "Using Graft Steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
033	*StepDescription* released, all sub-cases complete	The *StepDescription* dynamic sub-procedure step has been released. This is because all the sub-cases started from the step are complete. See "Defining a Dynamic Call to Multiple Sub-Procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.
034	Case migrated from Procedure *StepName* to *StepDescription* by *UserName*	The case from the procedure has migrated to a new procedure with a new version number, where: *StepName* is the name of the step. *StepDescription* is the name of the form which is displayed when you open this work item. *UserName* is the name of the iProcess user who has received the work item in their work queue. See "Using Version Control" in *TIBCO iProcess Modeler Procedure Management* for more information. See "Release a Procedure Version" in *TIBCO iProcess swutil and swbatch Reference Guide* for more information about migrating cases to new procedure versions.
035	Sub-cases, grafted to *StepDescription*, closed	The sub-cases grafted to the *StepDescription* graft step have been closed. This is because the graft step has been withdrawn because a deadline expired. This means that the deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.

Table 12   Audit Trail Messages (Sheet 7 of 12)

Message ID	Message	Description
036	`Deadline for` *StepDescription* `expired`	The deadline set for the *StepDescription* graft step has expired. If the deadline has expired, then the deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
037	`Sub-cases, started from` *StepDescription*`,` `closed`	The deadline set on the *StepDescription* dynamic sub-procedure step has expired so the dynamic sub-procedure step has been withdrawn. This has caused the sub-cases started from the dynamic sub-procedure step to close. This means that the deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
038	*StepDescription* `withdrawn,` `outstanding items not` `deleted`	The *StepDescription* step has been withdrawn because a deadline has expired. However, the outstanding items have not been deleted. If the deadline has expired, then the deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
039	`No addressees defined for step` *StepDescription* `– automatically released`	The *StepDescription* step has no addressees defined for it so it has been automatically released. See "Defining a Step" in *TIBCO iProcess Modeler Getting Started* for more information.
040	`No sub-procedures defined for step` *StepDescription* `– automatically released`	The *StepDescription* step has no sub-procedures defined for it so it has been automatically released. See "Defining and Using Sub-procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.
041	*StepDescription* `forwarded by` *UserName*	A *StepDescription* work item is forwarded by an iProcess user. The *UserName* is the name of the iProcess user who forwards the work item.
042-049	There are no messages defined for these numbers.	

Table 12   Audit Trail Messages (Sheet 8 of 12)

Message ID	Message	Description
050	*StepDescription* `EAI call-out initiated` (*UserName*)	The *StepDescription* step has initiated an EAI call-out to an external system on behalf of a *UserName* user. The iProcess Suite cannot continue processing the case until the EAI call-out has completed. See "Using EAI steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
051	*StepDescription* `EAI call-out completed` (*UserName*)	The EAI call-out initiated by the *StepDescription* step has completed. *UserName* is the name of the iProcess user on whose behalf the call-out was made. See "Using EAI steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
052	`Deadline for EAI Step` *StepDescription* `expired`	The deadline for the *StepDescription* EAI step has expired. The deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
053	`EAI Step` *StepDescription* `withdrawn`	The *StepDescription* EAI step has been withdrawn because the deadline has expired. The deadline actions will be processed. See "Using Deadlines in Procedures" in *TIBCO iProcess Modeler Basic Design* for more information.
054	`Commit Point` *StepDescription* `reached`	The procedure has reached a *StepDescription* transaction control step. This step is configured to commit the current data at the current point in the business process. See "Using Transaction Control steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
055	`New Transaction started from` *StepDescription*	The procedure has started a new transaction from the *StepDescription* transaction control step. See "Using Transaction Control steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
056	`New Transaction start retried from` *StepDescription*	The *StepDescription* step has retried the new transaction. See "Using Transaction Control steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.

Table 12   Audit Trail Messages (Sheet 9 of 12)

Message ID	Message	Description
057	Case purged	The case has been purged. For example, the iProcess Administrator may purge cases if they are dead or if a new version of a procedure is produced and cases for the existing version should no longer be processed.
058	*Reason* Case data modified by *UserName*	Case data has been modified by user *UserName*. *Reason* gives a description of the reason for the change, as specified in the SW_MODIFY_CASEDATA statement. See *TIBCO iProcess Engine Database Administrator's Guide* for your database for details.
059	*stepdescription* opened by *username*	The *StepDescription* work item has been opened by the user *UserName*. See AUDIT_OPENKEEP on page 211 for more information.
060	*stepdescription* kept by *username*	The *StepDescription* work item has been kept by the user *UserName*. See AUDIT_OPENKEEP on page 211 for more information.
061-079	There are no messages defined for these numbers.	
080	*StepDescription* EAI call-out failed (*UserName*)	The EAI call-out initiated from the *StepDescription* EAI step on behalf of the *UserName*. Check the sw_warn or sw_error log files, which is located in the *SWDIR*\logs directory, to see if any error messages were logged. See *TIBCO iProcess Engine System Messages Guide* for more information. See "Using EAI steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.

Table 12   Audit Trail Messages (Sheet 10 of 12)

Message ID	Message	Description
081	Workflow may have an infinite loop (at *StepDescription*) – reached max actions per transaction (*UserName*)	You can limit the number of steps sent or withdrawn during the processing of a single workflow transaction (i.e. the number of EAI steps that can be processed in one transaction without any other step types in between). You receive this message if this limit is reached. If this limit is reached, the workflow transaction is aborted and an appropriate message is logged to the sw_warn log file, which is located in the *SWDIR*\logs directory. See *TIBCO iProcess Engine System Messages Guide* for more information.
082	Error, workflow transaction aborted because of a system failure - check sw_warn/sw_error logs	The workflow transaction has been aborted because of an internal system failure. Appropriate messages should be logged to the sw_warn or sw_error log files. See *TIBCO iProcess Engine System Messages Guide* for more information.
083	The run-time plug-in for EAI Type *UserName* (used by step *StepDescription* is not registered on all servers or failed to load/initialize correctly.	Some EAI plug-ins need to be registered before you can use them. You may receive this message if your EAI plug-in has not been registered or if it has not been installed correctly, where: *UserName* is the name of the iProcess user on whose behalf the EAI step is running. *StepDescription* is the description of the EAI step. See "Using EAI steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
084	Invalid sub-procedure *UserName* specified for *StepDescription* - check sw_warn/sw_error logs	The *UserName* specified for the *StepDescription* sub-procedure step (on whose behalf the sub-procedure is being called) is invalid. You need to fix the step so that it uses the correct name. Check the sw_warn or sw_error log files to see if any error messages were logged. See *TIBCO iProcess Engine System Messages Guide* for more information. See "Defining and Using Sub-procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.

Table 12   Audit Trail Messages (Sheet 11 of 12)

Message ID	Message	Description
085	*StepDescription* and sub-procedure *UserName* are not based on the same parameter template - check sw_warn/sw_error logs	The *StepDescription* step is trying to call a sub-procedure whose parameter template is not the same as the main procedure. *UserName* is the name of the iProcess user on whose behalf the sub-procedure is being called. Check the sw_warn or sw_error log files to see if any error messages were logged. See *TIBCO iProcess Engine System Messages Guide* for more information. See "Defining and Using Sub-procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.
086	*StepDescription* and sub-procedure *UserName* are not based on the same version of parameter template - check sw_warn/sw_errlogs	The *StepDescription* step is trying to call a sub-procedure whose parameter template is not the same version as the main procedure. *UserName* is the name of the iProcess user on whose behalf the sub-procedure is being called. Check the sw_warn or sw_error log files, which is located in the *SWDIR*\logs directory, to see if any error messages were logged. See *TIBCO iProcess Engine System Error Messages Guide* for more information. See "Defining and Using Sub-procedures" in *TIBCO iProcess Modeler Advanced Design* for more information.
087	Transaction Aborted at *StepDescription*	The procedure has found an error and has reached a *StepDescription* transaction control step that has caused the transaction to abort. Check the sw_warn or sw_error log files to see if any error messages were logged. See *TIBCO iProcess Engine System Messages Guide* for more information. See "Using Transaction Control Steps" in *TIBCO iProcess Modeler Integration Techniques* for more information.
088	*StepDescription* EAI delayed-release failed (*error code (failure_count, BG_action) : error message*)	The transaction of delayed release *StepDescription* EAI step failed. The *error code* and *error message* variables specify the error. The *failure_count* variable specifies the number of times the delayed release EAI step has failed. The *BG_action* variable specifies what the BG process does when an error occurred.

Table 12   Audit Trail Messages (Sheet 12 of 12)

Message ID	Message	Description
089-127	There are no messages defined for these numbers.	
128	*stepdescription* `delivered to Exchange recipient` *username*	This message is obsolete.
129	*stepdescription* `release received from Exchange recipient` *username*	This message is obsolete.
130	*stepdescription* `withdrawn from Exchange recipient` *username*	This message is obsolete.
131	`BusinessWorks Activity Audit` *description* `processed by` *username*	The action *description* has been carried out by the user *UserName* within BusinessWorks. This message is generated by, and the *description* text is provided by, the BusinessWorks iProcess Audit activity. It is used to audit BusinessWorks activities in iProcess Engine. See the *TIBCO iProcess Connector for ActiveMatrix BusinessWorks User's Guide* for more information about the iProcess Audit activity.
132	There are no messages defined for this number.	
133	`Case data changed by` *username*	The changes *UserName* makes to the case data by using iProcess Insight, iProcess Workspace (Browser), or the `setCaseData` TIBCO iProcess Server Objects interface, rather than by normal step processing. For more information about changing case data in iProcess Insight, see *TIBCO iProcess Insight User's Guide*. For more information about changing case data in iProcess Workspace (Browser), see *TIBCO iProcess Workspace (Browser) User's Guide*.
134-255	There are no messages defined for these numbers.	

Appendix E   **iProcess Server Manager Interfaces**

This appendix describes the programming interfaces provided by the Process Sentinels for integration with TIBCO Hawk. Using these interfaces, you can create your own TIBCO Hawk console application for use with iProcess Engine. For more information, see the TIBCO Hawk documentation set.

Topics

getNodeDetails()

Method

Purpose	This method returns the Globally Unique Identifier (GUID) associated with a iProcess Engine node.
Type	Synchronous, IMPACT_INFO.
Arguments	None.
Returns	

Name	Type	Description
NodeGUID	String	Globally Unique Identifier of the node to which this process belongs.
DatabaseInfo	String	Description of the database host, type and schema to which this micro agent belongs.

getProcessDetails()

Method

Purpose This method returns the details of a iProcess Engine process, including the logical machine ID, process name and instance (as configured in the process_config database table.

Type Synchronous, IMPACT_INFO.

Arguments None.

Returns

Name	Type	Description
MachineID	Integer	Logical machine ID of the server on which the process is running.
ProcessName	String	Logical process name of the process.
ProcessInstance	Integer	Logical process instance of the process.

getProcessSummary()

Method

Purpose This method returns a summary of the current processes for a server (regardless of whether they are configured to run).

Type Synchronous, IMPACT_INFO.

Arguments None.

Returns

Name	Type	Description
ProcessType	String	Logical process name of this process type.
NumInstances	Integer	Number of configured processes of this type.
NumRunning	Integer	Number of running processes of this type.
ParentType	String	Logical process name of the process's parent type.
NumChildren	Integer	Number of child processes the process type has.
ChildrenPaused	Boolean	Whether the child processes are paused.

Index ProcessType

getProcessStatus()

Method

Purpose	This method returns detailed process information for a server (only for those processes configured to run on the server).
Type	Synchronous, IMPACT_INFO.
Arguments	None.
Returns	

Name	Type	Description
MachineID	Integer	Logical machine ID of the server on which the process is running.
ProcessName	String	Logical process name of the process.
ProcessInstance	Integer	Logical process instance of the process.
Enabled	Boolean	Whether the process is enabled for startup.
Persistent	Boolean	Whether the process is persistent.
LastKnownStatus	String	Last known status of the process.
StatusComment	String	Comment associated with the last known status.

Indexes   MachineID, ProcessName, ProcessInstance

doStartProcesses()

Method

Purpose   This method starts one or more processes on the specified server. If no parameters are passed, all processes on the current server are started. Use the `ProcessType` parameter to start processes of a specified type. If you use this parameter you can also specify a specific process instance with the `ProcessInstance` parameter.

Type   Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
ProcessType	String	Logical process name of the type of process to start. The process type must be one that can start independently of the other iProcess Engine processes (see getIsTypeDynamic() on page 471).
ProcessInstance	Integer	Logical process instance of the process to start. Must be used in conjunction with the `ProcessType` parameter. A value of 0 means that all instances of the indicated process type are started.

Errors

Error Code	Error Message
ERR_PMAMI_PROCTYPE	*ProcessType* is not a valid process type.
ERR_PMAMI_TYPEDYNAMIC	Processes of type *ProcessType* cannot be started independently.
ERR_PMAMI_PROCINST	ProcessInstance must be >= 0.
ERR_PMAMI_PARAM	ProcessInstance cannot be specified without ProcessType.
ERR_PMAMI_MALLOC	Insufficient memory to start processes.

See Also   getIsTypeDynamic()

doStartTemporaryProcess()

Method

Purpose   This method starts one or more temporary processes of the specified type on the current server. These instances will not be restarted if iProcess Engine is restarted. Both arguments are mandatory.

Type   Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
ProcessType	String	Logical process name of the type of temporary process to start. The process type must be one that can start independently of the other iProcess Engine processes (see getIsTypeDynamic() on page 471).
ProcessInstance	Integer	Logical process instance of the temporary process to start.

Errors

Error Code	Error Message
ERR_PMAMI_NOPROCTYPE	You must specify a process type to start.
ERR_PMAMI_PROCTYPE	*ProcessType* is not a valid process type.
ERR_PMAMI_TYPEDYNAMIC	Processes of type *ProcessType* cannot be started independently.
ERR_PMAMI_PROCINST	ProcessInstance must be > 0.
ERR_PMAMI_PARAM	ProcessInstance cannot be specified without ProcessType.
ERR_PMAMI_MALLOC	Insufficient memory to start processes.

See Also   getIsTypeDynamic()

doRestartProcess()

Method

Purpose   This method restarts a iProcess Engine process that has failed and been placed in a suspended state by its controlling process sentinel.

Type   Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
ProcessName	String	Logical process name of the process to restart.
ProcessInstance	Integer	Logical process instance of the process to restart.

Errors

Error Code	Error Message
ERR_PMAMI_NOPROCTYPE	You must specify the process type of the process to restart.
ERR_PMAMI_PROCTYPE	*ProcessType* is not a valid process type.
ERR_PMAMI_PROCINST	ProcessInstance must be > 0.
ERR_PMAMI_NOTSUSPENDED	The process *MachineID*, *ProcessName*, *ProcessInstance* is not suspended.
ERR_PMAMI_NOSUCHPROC	The process *MachineID*, *ProcessName*, *ProcessInstance* does not exist.
ERR_PMAMI_MALLOC	Insufficient memory to start processes.

doStopProcesses()

Method

Purpose This method stops one ore more iProcess Engine processes on the current server. If no parameters are passed, all processes on the current server are stopped. Use the ProcessType parameter to stop processes of a specified type. If you use this parameter you can also specify a specific process instance with the ProcessInstance parameter. You can also specify optional arguments to perform a forced shutdown, which stops processes regardless of any active user sessions.

Type Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
ProcessType	String	Logical process name of the type of process to stop. The process type must be one that can be stopped independently of the other iProcess Engine processes (see getIsTypeDynamic() on page 471).
ProcessInstance	Integer	Logical process instance of the process to stop. Must be used in conjunction with the ProcessType parameter. A value of 0 means that all instances of the indicated process type are stopped.
ForcedShutdown	Boolean	Whether the shutdown is forced. If 1, users are given 300 seconds before the forced shutdown begins.
ForcedTimeout	Integer	Number of seconds before the forced shutdown begins. Must be used in conjunction with the ForcedShutdown parameter.

Errors

Error Code	Error Message
ERR_PMAMI_PROCTYPE	*ProcessType* is not a valid process type.

Error Code	Error Message
ERR_PMAMI_TYPEDYNAMIC	Processes of type *ProcessType* cannot be stopped independently.
ERR_PMAMI_PROCINST	ProcessInstance must be >= 0.
ERR_PMAMI_PARAM	ProcessInstance cannot be specified without ProcessType.
ERR_PMAMI_MALLOC	Insufficient memory to start processes.

See Also   getIsTypeDynamic()

getIsTypeDynamic()

Method

Purpose This method queries the Process Sentinels to determine if a specific process type can be started or stopped independently of the other iProcess Engine processes.

Type Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
ProcessType	String	Logical process name of the type of process to check.

Returns

Name	Type	Description
IsDynamic	Boolean	Whether the process type can be started or stopped independently of the other iProcess Engine processes. TRUE means it can; FALSE means it cannot.

Errors

Error Code	Error Message
ERR_PMAMI_NOPROCTYPE	You must specify the process type of the process to check.
ERR_PMAMI_PROCTYPE	*ProcessType* is not a valid process type.

getLogFileLines()

Method

Purpose   This method returns a portion of the contents of the specified log file in *SWDIR*/logs. You control which portion of the log file is displayed by the arguments you pass.

Type   Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
LogFile	String	Name of the log file in *SWDIR*/logs that you want to open.
StartPos	Integer	Line from which the Process Sentinels should start returning data. This parameter can take one of the following values: • 0 - starts returning data from the start of the file • n (where n is a line number greater than zero) - returns data starting with the specified line number • -1 - starts returning data from the end of the file
NumLines	Integer	Number of lines from the end of the log file that should be returned. Defaults to 10.

Returns

Name	Type	Description
ErrorMessage	String	A line from the specified log file.

getAllQueues()

Method

Purpose This method can be used to return identifying information on all the message queues defined in iProcess.

Type Synchronous, IMPACT_INFO.

Arguments None.

Returns

Name	Type	Description
QueueID	Integer	Message queue ID.
QueueName	String	Message queue name.
QueueType	String	Message queue type.
QueueDesc	String	Message queue description.
For Oracle only:		
NormalMessageNumber	Integer	The number of normal messages in this queue.
DeadMessageNumber	Integer	The number of dead messages in this queue.
For SQL and DB2 only:		
MessageNumber	Integer	The number of messages in this queue.

getMessageHeader()

Method

Purpose	This method can be used to get the message summary information for the messages in a given queue.
Type	Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
QueueID	Integer	The ID of the message queue from which you want message summary information.
For Oracle only:		
ShowExceptionMessage	Boolean	Set this parameter as `true` if you want to show exception message items. Otherwise set it as `false`.

Returns

Name	Type	Description
MessageID	String	The message ID of a message in the queue. This and the other items listed are returned for each message in the queue.
Instruction	String	The instruction type of the message (for example KEEP, RELEASE, NEWCASE).
Addressee	String	The addressee of the message.
Procedure	String	The name of the procedure which generated the message.
StepName	String	The name of the step within the procedure which generated the message.
CaseNumber	String	The case number of the case being processed.

Name	Type	Description
ReqID	String	The request ID for the message. A ReqID is needed for each work item.
For SQL Server Only:		
Failedby	String	This parameter is only applicable to messages in the dead queue. It indicates the process from which the dead message was created, such as WIS or BG.
For DB2 only:		
Failedby	String	This parameter is only applicable to messages in the dead queue. It is the database table where the message was located before it became dead.

Errors

Error Code	Error Message
ERR_PMAMI_PARAM	If you enter an invalid QueueID, the interface returns an error similar to the following: `Failed to open queue ` *queue id*`:`*iProcess error code*

getMessageDetail()

Method

Purpose This method can be used to get detailed information for a given message.

Type Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
QueueID	Integer	The ID of the message queue from which you want message detail information.
MessageID	String	The message ID of the message from which you want detail information. Specify All to get detail information on all messages.
For Oracle only:		
ShowExceptionMessage	Boolean	Set this parameter as true if you want to show exception message items. Otherwise set it as false.

Returns

Name	Type	Description
MessageID	String	The message ID of a message in the queue.This and the other items are returned for each message in the queue.
Instruction	String	The instruction type of the message (for example KEEP, RELEASE, NEWCASE).
Addressee	String	The addressee of the message.
Procedure	String	The name of the procedure which generated the message.
StepName	String	The name of the step within the procedure which generated the message.

Name	Type	Description
Case Number	String	The case number of the case being processed.
ReqID	String	The request ID for the message. A ReqID is needed for each work item.
FailCount	Integer	The fail count.
ParsedMessageData	String	Parsed Message Data. A sequence of *field name:field value*; pairs. If the raw data cannot be parsed successfully, this argument will be `N/A:N/A;`.
RawMessageData	String	The raw message data - the same data as in the previous field, but unparsed.
For SQL Server only:		
Failedby	String	This parameter is only applicable to messages in the dead queue. It indicates the process from which the dead message was created, such as WIS or BG.
For DB2 only:		
Failedby	String	This parameter is only applicable to messages in the dead queue. It is the database table where the message was located before it became dead.

Errors

Error Code	Error Message
ERR_PMAMI_PARAM	If you enter an invalid QueueID, the interface returns an error similar to the following: `Failed to open queue queue id:iProcess error code`

deleteMessage()

Method

Purpose   This method can be used to delete a given message.

Type   Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
QueueID	Integer	The ID of the message queue from which you want message detail information.
MessageID	String	The message ID of the message you want to delete. Specify All to delete all messages.
For Oracle only:		
ExceptionQueue	Boolean	Set this parameter as true if you want to delete exception messages. Otherwise set it as false.

Returns

Name	Type	Description
AffectedNum	Integer	The number of messages that have been deleted successfully.

Errors

Error Code	Error Message
ERR_PMAMI_PARAM	If you enter an invalid QueueID, the interface returns an error similar to the following: Failed to open queue *queue id*:*iProcess error code*

restoreDeadMessage()

Method

Purpose   This method can be used to restore a dead message to the queue from which it came.

Type   Synchronous, IMPACT_INFO.

Arguments

Name	Type	Description
MessageID	String	The message ID of the message you want to restore. Specify All to restore all dead messages.
For Oracle and DB2 only:		
QueueID	String	The ID of the queue to which dead messages will be restored.
For SQL Server Only:		
ProcessName		A logical process name. This parameter specifies the resource process from which the dead messages came.
ToQueueID		The Destination Queue ID - the ID of the queue to which you wish to restore the message. This queue must be hosted by the process specified by ProcessName.

Returns

Name	Type	Description
AffectedNum	Integer	The number of dead messages that have been restored successfully.

Errors

Error Code	Error Message
ERR_PMAMI_PARAM	If you enter an invalid QueueID, the interface returns an error similar to the following: `Failed to open queue` *queue id*:*iProcess error code*

Overview of the plist Utility

The *SWDIR*/util/plist utility provides the ability to list information about a range of TIBCO iProcess Engine facilities. In this section, it introduces the list of options that can be used with the *SWDIR*/util/plist utility.

General Usage of the *SWDIR*/util/plist Utility

To find the list of options, which can be used together with the *SWDIR*/util/plist utility, use the following command:

- On Microsoft Windows:
  SWDIR/util/plist /?

- On UNIX:
  SWDIR/util/plist -?

Functions for the *SWDIR*/util/plist Command Options

The *SWDIR*/util/plist utility is used with an option command. The options and the functions of each option command are listed in the following table.

Table 13 plist Command Options (Sheet 1 of 3)

Option	Function	Command Line
-a	list audit trail	plist -a [nodename] <procedure> ALL|<casenum>
-A	list attribute value	plist -A [nodename] <group>|<user>
-c	list cases for procedure	plist -c [nodename] <procname>
-C	list sub-cases status for procedure	plist -C [nodename] <procname>
-d	list steps in procedure	plist -d [nodename] <procname>
-D	list step status	plist -D [nodename] <procname>
-e	list system events	plist -e <typeid[,typeid...]|ALL> [username[,username...]|ALL [startdate [enddate]]]
-f	list fields in procedure	plist -f [nodename] <procname>

Table 13   plist Command Options (Sheet 2 of 3)

Option	Function	Command Line
-G	list group memberships	`plist -G [nodename]` `<group>|<user>`
-l	list long term locked items	`plist -l [username]`
-m	list mail items for queue	`plist -m <queuename> |` `ALLQUEUES`
-mW	List mail items for queue including Work Queue Parameters and CDQPs	`plist -mW` `<queuename>|ALLQUEUES`
-M	list Mbox information	plist -M <MBSetName[,MBSetName[,MBSetName]...][interval]>, page 521
-n	list nodes	`plist -n`
-o	list outstanding mail	plist -o <procedure> ALL|<casenum> ALL|<queue> ALL|<node>
-p	list procedures	`plist -p [nodename]`
-P	display procedure version information	plist -P [ALL [version|ALLVERS]|<procname> [version|ALLVERS]]
-Q	list queue participation information	plist -Q
-r	list roles	plist -r [nodename]
-R	list queue redirection information	plist -R
-s	list installed tables	`plist -s [nodename]`
-t	list uninstalled tables	`plist -t [nodename]`
-T	get performance data from pool servers	plist -T[g] <pool svr id|ALL> [interval [count]]
-u	list users	`plist -u [nodename]`
-U	list the OS or UVAPI users	`plist -U`

Table 13   plist Command Options (Sheet 3 of 3)

Option	Function	Command Line
-w	list WIS queue information	plist -w [WIS Number]

The following options are used in combination with the options, which are shown in the previous table:

- `-v` Verbose (list extra information), for example: `plist -nv` (verbose nodes)

- `-x` Format output for scripts

- `-X` TIBCO iProcess Engine 2000 compatibility mode

The options are case sensitive.

Node Information

Command   The following commands output the node and any slave node information:

- To return the information about any node or slave nodes:
  ```
  plist -n
  ```
- To return additional information about the node, like Flags and Step Name:
  ```
  plist -nv
  ```

Returns   The `plist -n` and `plist -nv` commands return the following information.

Column	Description
Flags	Bit field is used to represent single bit flags. The meaning for each of the single bits is as follows: • 1 is record used. • 2 acknowledge connection. • 4 specifies a slave node. • 8 specifies the node is awaiting removal by the background process.

Example   The following examples show the output node information of the commands.

- Input the *SWDIR*/util/plist -n command. The output is shown as follows.

```
Name      Type   Cert Mail     SW Ver  RPC Ver   Mail Address
----------------------------------------------------------------
ipc_nod1   ?                    11.3    3.0
```

- Input the *SWDIR*/util/plist -nv command. The output is shown as follows.

```
Name      Flags  Type   Step Name     Cert Mail    Mail Address
----------------------------------------------------------------
ipc_nod1    3     ?     ipc_nod1.n
```

Procedure Information

Command   The following commands output the procedure information:

- To return the procedure information:

  ```
  plist -p [nodename]
  ```

- To return additional information about the total number of cases for each procedure:

  ```
  plist -pX
  ```

- To return additional procedure information about Version, StartStep, and Directory:

  ```
  plist -pv
  ```

- To return additional procedure information about NumCase, CurCases, and DedCases:

  ```
  plist -pvX
  ```

Returns   The `plist -p`, `plist -pX`, `plsit -pvX` and `plist -pv` commands return the following information.

Column	Description
Owner	The owner of the procedure.
Number	The information in this column is displayed in the *reference_number*:*time_number* format, where: • *reference_number* is the unique reference number. • *time_number* is the number of times the record has been used.

Column	Description
Flags	The meaning for each flags is: • `N` networks. • `a` admin functions. • `s` start cases. • `I` this procedure is a Sub-procedure. • `H` this procedure has a Sub-procedure. • `m` management reports. • `M` Management reports exist. • `P` Auto purge ON. • `W` Working days. • `h` hidden case description. • `o` optional case description. • `0` field addressee NA. • `T` this procedure is a template. • `:WTH` this procedure is withdrawn. • `:REL` this procedure is released. • `:UNR` this procedure is unreleased. • `:INC` this procedure is incomplete. • `:MOD` this procedure is a model. • `:WIC` WITHDRAWN_INC. • `:DEL` this procedure is deleted.
Name	The name of the procedure.
Description	The description of the procedure.
NumCases	Displayed if the `plist -pX` command is used. The latest started case number.
Version	Displayed if the `plist -pv` or `plist -pvX` command is used. This column displays the procedure version number in *X:Y* format, where: • *X* is the major version number. • *Y* is the minor version number.

Column	Description
StartStep	Displayed if the `plist -pv` or `plist -pvX` command is used.
	The name of the start step for the procedure.
CurCases	Displayed if the `plist -pvX` command is used.
	The total number of the cases in a procedure.
DedCases	Displayed if the `plist -pvX` command is used.
	The total number of dead cases in a procedure.

Example   The following examples show the output procedure information of the commands.

- Input the *SWDIR*`/util/plist -p` command. The output is shown as follows.

```
Owner     Number Flags                Name      Description
-----------------------------------------------------------------
Administ  1:1    -----------P-:REL $EMAIL   Electronic Mail
Administ  3:1    -H---------W:REL TESTFUL   Automated Test
Framework
Administ  4:1    --I--------W:REL TESTPRC1 Automated Test Proc
```

- Input the *SWDIR*`/util/plist -pX` command. The output is shown as follows.

```
Owner     Number Flags            Name      Description                          NumCases
-------------------------------------------------------------------------------
Administ  1:1    -----------P-:REL $EMAIL   Electronic Mail                      0
Administ  2:1    -----------P-:REL $SYSTEM  System Info                          0
Administ  3:1    -----------W:INC TP1                                            0
Administ  4:1    -----------W:INC TEST                                           0
```

- Input the *SWDIR*`/util/plist -pv` command. The output is shown as follows.

```
Name      Flags            Version  StartStep Directory
------------------------------------------------------------
$EMAIL    -----------P-:REL  0:0    MESSAGE1 $email.p
$SYSTEM   -----------P-:REL  0:0    SYSINFO  $system.p
TESTFUL   -H---------W:REL   0:0    FULSCR01 testful.p
TESTPRC1  --I--------W:REL   0:1    START    testprc1.p
```

- Input the *SWDIR*`/util/plist -pvX` command. The output is shown as follows.

Name	Flags	Version	StartStep	Directory	NumCase	CurCases	DedCases
$EMAIL	-----------P-:REL	0:0	MESSAGE1	$email.p	0	0	0
$SYSTEM	-----------P-:REL	0:0	SYSINFO	$system.p	0	0	0
TP1	------------W:INC	0:0		tp1.p	0	0	0
TEST	------------W:INC	0:0		test.p	0	0	0

Installed and Uninstalled Table Information

This section lists the plist commands that return the installed or uninstalled tables information.

Installed Table Information

Command   The following commands output the installed table information:

- To return the information about installed table items:

  ```
  plist -s [nodename]
  ```

- To return additional information about Subfield, Type, Length, and Decimal:
  ```
  plist -sv
  ```

Example   The following examples show the output node information of the commands.

- Input the *SWDIR*/util/plist -s command. The output is shown as follows.

```
Name
----
HARDWARE
SOFTWARE
CUSTOMERS
PROSPECTS
```

- Input the *SWDIR*/util/plist -sv command. The output is shown as follows.

```
Name        Subfield   Type    Length    Decimal
----------------------------------------------------------------
PROSPECTS
            CONAME     A       20        0
            STREET     A       20        0
            TOWN       A       20        0
            COUNTY     A       20        0
            POST       A       20        0
            COUNTRY    A       20        0
```

Uninstalled Table Information

Command   The following commands output the uninstalled table information:

- To return the information about uninstalled table items:
  ```
  plist -t [nodename]
  ```

- To return additional information about uninstalled table items:

  ```
  plist -tv
  ```

Returns   The output contains the same column names as the output of the *SWDIR*/util/plist -s command. For more information, see Example in .

Role Information

Command   The following command returns the information about user's roles:

```
plist -r [nodename]
```

Returns   The `plist -r` command returns the following information.

Column	Description
Name	Name of the role.
User	Name of the (user or group) queue that the role is assigned to.

Example   Input the *SWDIR*/util/plist -r command. The output is shown as follows.

```
Name            User
-------------------------------
LOAN MANAGER    Dev1@ipc_nod1
```

Long Term Locked Item Information

Command   The following commands output the long term locked item information:

- To return the information about long term locked items:

  ```
  plist -l [username]
  ```

- To return Locker, UserName, and RQID information:

  ```
  plist -lv
  ```

Returns   The `plist -l`, and `plist -lv` commands return the following information.

Column	Description
Locker	User who has the work item locked (for example, when in group mailbox).
Procnum	The procedure number. The number of times the record is used.
RQID	The unique reference assigned internally.

Example   The following examples show the output node information of the commands.

- Input the *SWDIR*/util/`plist -l` command. The output is shown as follows.

```
Q    Node Procname Case# Stepname Flags      Case Description
------------------------------------------------------------------
```

- Input the *SWDIR*/util/`plist -lv` command. The output is shown as follows.

```
Q Node Flags Locker UserName Host Procnum Procname Casenum Dirname  RQID
Deadline-time
---------------------------------------------------------------------------------
```

Mail Item Information

Command   The following commands output the mail item information:

- To return the information about mail items:

  `plist -m <queuename> | ALLQUEUES`

- To return work queue parameters:

  `plist -mW <queuename>|ALLQUEUES`

- To return Locker, Username, Host, Procnum, Stepname, RQID, and Deadline-time:

  `plist -mv <queuename> | ALLQUEUES`

- To return Case-Description, Case-Starter, Step-Description, Procedure-Description, Step-Flags, Proc-Flags, Arrival-time, and PVer:

  `plist -mvv <queuename> | ALLQUEUES`

Returns   The `plist -m`, `plist -mv`, `plist -mvv`, and `plist -mW` commands return the following information.

Column	Description
QName	The name of the queue.
RQID	The unique reference assigned internally.
Case#	The case number. It can be the CID or CCR. The CID is the case ID, a unique reference used internally by TIBCO iProcess Engine. The CCR is referred to as a case control record number.

Column	Description
Flags	The meaning for each flag is:
	• `s` mail suspended.
	• `m` item has no memos (for WIS only).
	• `r` item awaiting forwarding (for WIS only).
	• `T` test mail.
	• `G` item is locked for a long term.
	• `f` not available for opening or forwarding.
	• `U` invalid addressee.
	• `S` start step.
	• `X` the deadline.
	• `D` this mail has a deadline.
	• `P` this mail has been purged.
	• `L` item is open and locked by the user.
	• `N` new piece of mail.
	• `F` free record.
Locker	Displayed if the `plist -mv` or `plist -mvv` command is used.
	User who has the work item locked (for example, when in group mailbox).
UserName	Displayed if the `plist -mv` or `plist -mvv` command is used.
	The user who owns this queue, or the group who owns this work item.
Host	Displayed if the `plist -mv` or `plist -mvv` command is used.
	The node ID, which is associated with the work item.
Procnum	Displayed if the `plist -mv` or `plist -mvv` command is used.
	The procedure number. The number of times the record is used.

Column	Description
Procname	Displayed if the `plist -mv` or `plist -mvv` command is used.
	The name of the procedure.
Casenum	Displayed if the `plist -mv` or `plist -mvv` command is used.
	The Case number.
Stepname	The name of the step.
Pno	Displayed if the `plist -mv` or `plist -mvv` command is used.
	The place number. The place is a unique number allocated to each step.
Deadline-time	Displayed if the `plist -mv` or `plist -mvv` command is used.
	The date and time when the deadline (if defined) expires on the work item.
Case-Description	Displayed if the `plist -mvv` command is used.
	The description of the case, which the work item belongs to.
Case-Starter	Displayed if the `plist -mvv` command is used.
	The user who started the case, which the work item belongs to.
Step-Description	Displayed if the `plist -mvv` command is used.
	The description of the step.
Procedure-Description	Displayed if the `plist -mvv` command is used.
	The description of the procedure, which generated the work item.

Column	Description
Step-Flags	Displayed if the `plist -mvv` command is used.
	The meaning for each flag is:
	• `W` means withdraw.
	• `V` means view.
	• `R` means rebuild.
	• `O` means outonly.
	• `F` means forward.
	• `E` means edit.
	• `S` means status.
	• `d` means deadline.
	• `a` means action.
	• `D` means document.
	• `A` means address.

Column	Description
Proc-Flags	Displayed if the `plist -mvv` command is used.
	The meaning for each flags is:
	• `N` networks.
	• `a` admin functions.
	• `s` start cases.
	• `I` this procedure is a Sub-procedure.
	• `H` this procedure has a Sub-procedure.
	• `m` management reports.
	• `M` Management reports exist.
	• `P` Auto purge ON.
	• `W` Working days.
	• `h` hidden case description.
	• `o` optional case description.
	• `0` field addressee NA.
	• `T` this procedure is a template.
	• `:WTH` this procedure is withdrawn.
	• `:REL` this procedure is released.
	• `:UNR` this procedure is unreleased.
	• `:INC` this procedure is incomplete.
	• `:MOD` this procedure is a model.
	• `:WIC` WITHDRAWN_INC.
	• `:DEL` this procedure is deleted.
Arrival-time	Displayed if the `plist -mvv` command is used.
	The timestamp when the work item is delivered to the queue.
PVer	Displayed if the `plist -mvv` command is used.
	The version number of the procedure, which generated the work item.

Example The following examples show the output node information of the commands.

- Input the *SWDIR*/util/plist -m queue command where queue is the name of the queue. The output is shown as follows.

```
QName     ReqID     PName     Case#     Stepname Flags Case Description
------------------------------------------------------------------
queue     301       MORT01    201       TAKEAPP  ----S------ Rain
queue     951       MORT01    601       REVAPP   ---------N- test
queue     952       MORT01    551       REVAPP   ---------N- test
queue     1051      MORT01    701       TAKEAPP  ----S------ t
queue     1201      MORT01    801       REVAPP   ---------N- qq
queue     1251      MORT01    651       REVAPP   ---------N- xxx
```

- Input the *SWDIR*/util/plist -mW queue command where queue is the name of the queue. The output is shown as follows.

```
QName     ReqID     PName     Case#     Stepname Flags       Case Description
------------------------------------------------------------------
queue     301       MORT01    201       TAKEAPP  ----S------ Rain
          WQParam1 =
          WQParam2 =
          WQParam3 =
          WQParam4 =
queue     951       MORT01    601       REVAPP   ---------N- test
          WQParam1 =
          WQParam2 =
          WQParam3 =
          WQParam4 =
```

- Input the *SWDIR*/util/plist -mv queue command where queue is the name of the queue. The output is shown as follows.

```
QName    Flags Locker   UserName         Host     Procnum Procname Casenum    Stepname
Pno RQID    Deadline-time
------------------------------------------------------------------------------
queue  ----S------            swadmin@ipe113 ipe113    3:1    MORT01    201:201 TAKEAPP
3   301       3000/365 23:15
queue  ---------N-                ipe113    3:1   MORT01   601:601 REVAPP  4    951
3000/365 23:15
queue  ---------N-                ipe113    3:1   MORT01   551:551 REVAPP  4    952
3000/365 23:15
queue  ----S------            swadmin@ipe113 ipe113    3:1    MORT01    701:701 TAKEAPP
3   1051      3000/365 23:15
queue  ---------N-                ipe113    3:1   MORT01    801:801 REVAPP  4
1201      3000/365 23:15
queue  ---------N-                ipe113    3:1   MORT01    651:651 REVAPP  4
1251      3000/365 23:15
```

- Input the *SWDIR*/util/plist -mvv ALLQUEUES command. The output is shown as follows.

```
Flags           Locker   UserName          Host       Procnum Procname  Casenum   Stepname
Pno RQID    Deadline-time     Case-Description             Case-Starter
Step-Description          Procedure-Description    Step-Flags  Proc-Flags
Arrival-time    PVer
-----------------------------------------------------------------------------------
-------S------             administrator@sql_11_3_0 sql_11_3_0   3:1    SS1        51:51
STEP1    3   51        3000/12/31 23:15 dd1
administrator@sql_11_3_0                                         --R---S-aDA
------W:UNR 2012/06/26 10:14 0.0
-------S------             administrator@sql_11_3_0 sql_11_3_0   3:1    SS1        101:101
STEP1    3   101       3000/12/31 23:15 tt2
administrator@sql_11_3_0                                         --R---S-aDA
------W:UNR 2012/06/26 15:40 0.0
```

User Information

Command   The following commands output the user information:

- To return the information about users:

  ```
  plist -u [nodename]
  ```

- To return additional information about the directory:

  ```
  plist -uvx
  ```

Example   The following examples show the output node information of the commands.

- Input the *SWDIR*/util/plist -u command. The output is shown as follows.

```
Name
-------
Administrator
Agent
COMUsers
Dev1
Dev2
Dev3
Dev4
```

- Input the *SWDIR*/util/plist -uvx command. The output is shown as follows.

```
Administrator:C:\swserver\ipc_nod1\queues
Agent:C:\swserver\ipc_nod1\queues
COMUsers:C:\swserver\ipc_nod1\queues
Dev1:C:\swserver\ipc_nod1\queues
Dev2:C:\swserver\ipc_nod1\queues
Dev3:C:\swserver\ipc_nod1\queues
Dev4:C:\swserver\ipc_nod1\queues
FranMgr:C:\swserver\ipc_nod1\queues
```

WIS Queue Information

Command The following command returns the information about WIS queue:

```
plist -w [WIS Number]
```

where `WIS number` is an option to only display the details for a specific Work Item Server.

Returns The `plist -w` command returns the following information.

Column	Description
WIS	The number of the WIS (starting from 0). For example, if you have 5 WISes, the number of each WIS should be 0, 1, 2, 3, and 4.
QueueName	The name of the queue allocated to WIS.
Flags	The meaning for each flag is as follows: • `D` specifies the queue is disabled (this would normally be when the system has just been started and the queues have not yet been allocated to a WIS). • `U` specifies there are urgent items in this queue. • `G` specifies this is a group queue. • `T` specifies this is a test queue. • `D` specifies there are items in this queue with deadlines set. • `N` specifies there is new mail in this queue. • `M` specifies there is mail in this queue (i.e. it is not empty).
#Items	The number of items in the queue.
#NewP	The number of new items in the queue.
#Dead	The number of items in the queue with deadlines.
#Urgent	The number of urgent items in the queue.

Example Input the *SWDIR*/util/plist -w command. The output is shown as follows.

```
WIS QueueName              Flags     #Items #Newp  #Dead  #Urgent
-----------------------------------------------------------------
1    Administrator         -----NM   6      6      0      0
2    Agent                 --G--NM   8      8      0      0
1    COMUsers              --G----   0      0      0      0
2    Dev1                  ------M   1      0      0      0
Shared memory slots in use: 1.3%
```

Note   For more information about how to use this command, see Monitoring the WIS Processes on page 411.

Case Information

Command The following commands output the case information:

- To return the information about cases:

  ```
  plist -c [nodename] <procname>
  ```

  or

  ```
  plist -cv [nodename] <procname>
  ```

- To return Mail:

  ```
  plist -cX [nodename] <procname>
  ```

  or

  ```
  plist -cvX [nodename] <procname>
  ```

- To return PVer and PPrc:

  ```
  plist -cvv [nodename] <procname>
  ```

Returns The `plist -c`, `plist -cv`, `plist -cX`, `plist -cvv`, and `plist -cvX` commands return the following information.

Column	Description
CNum	The unique case number.
CID	The case ID, which is a unique reference used internally by TIBCO iProcess Engine. It is sometimes referred to as a case control record number or CCR.
Sts	The case status, LIVE or DEAD.
yyyy/ddd hh:mm	If the case status is Dead, the column displays the date and time when starting the case, otherwise it displays nothing.
Case Description	Information about the case.
Starter	User who started the case.
Mail	Displayed if the -X option is used. The number of the outstanding addressees.
PVer	Displayed if the -plist -cvv command is used. The version number of the procedure.

Column	Description
PPrc	Displayed if the `-plist -cvv` command is used.
	The procedure precedence.
	Three characters are displayed in the output, the order of the characters represents the procedure precedence.
	The meaning for each character is:
	• `u` specifies Unreleased procedure.
	• `m` specifies Model procedure.
	• `r` specifies Released procedure.

Example   The following examples show the output node information of the commands.

• Input the *SWDIR*/util/plist -c maintem command where maintem is the procedure name. The output is shown as follows.

```
CNum   CID   Sts   Case Description           Starter            yyyy/ddd hh:mm
--------------------------------------------------------------------------------
1452   1452  DEAD  ex1                        Dev1@ipc_nod1      2012/097 12:30
1453   1453  LIVE  con1                       Dev1@ipc_nod1      2012/097 14:31
1454   1454  DEAD  con2                       Dev1@ipc_nod1      2012/097 14:33
1455   1455  DEAD  con3                       Dev1@ipc_nod1      2012/097 14:34
1456   1456  LIVE  lab3.2.7                   Dev1@ipc_nod1      2012/097 15:25
1502   1502  DEAD  ex2                        Dev1@ipc_nod1      2012/097 12:18
1503   1503  DEAD  3.2.4                      Dev1@ipc_nod1      2012/097 18:21
1504   1504  LIVE  3.2.1.2                    Dev1@ipc_nod1      2012/097 15:48
1505   1505  DEAD  3.2.a                      Dev1@ipc_nod1      2012/097 16:53
1552   1552  DEAD  ex3                        Dev1@ipc_nod1      2012/097 12:29
```

• Input the *SWDIR*/util/plist -cX ss1 command where ss1 is the procedure name. The output is shown as follows.

```
CNum   CID   Sts   Case Description           Starter         Mail yyyy/ddd hh:mm
--------------------------------------------------------------------------------
1      1     DEAD  aa1                        administrator@sql 0   2012/178 10:12
51     51    LIVE  dd1                        administrator@sql 1   3000/365 23:15
101    101   LIVE  tt2                        administrator@sql 1   3000/365 23:15
```

• Input the *SWDIR*/util/plist -cvv ss1 command where ss1 is the procedure name. The output is shown as follows.

```
CNum  CID   Sts  Case Description         Starter            yyyy/ddd hh:mm
PVer  PPrc
------------------------------------------------------------------------------
----
1     1     DEAD aa1                      administrator@sql_11_3_0 2012/178 10:1
2 0.0  umr
51    51    LIVE dd1                      administrator@sql_11_3_0
0.0  umr
101   101   LIVE tt2                      administrator@sql_11_3_0
0.0  umr
```

Subcase Information

Command   The following command returns the information about sub-cases:

```
plist -C [nodename] <procname>
```

Returns   The `plist -C` command returns the following information.

Column	Description
CNum	Case number.
CID	The case ID, which is a unique reference used internally by TIBCO iProcess Engine. It is sometimes referred to as a case control record number or CCR.
yyyy/ddd hh:mm	If the case status is Dead, the column displays the date and time when starting the case, otherwise it displays nothing.
Case Description	Information about the case.
Starter	User who started the case.

Example   Input the *SWDIR*/util/plist -C maintem command where maintem is the procedure name. The output is shown as follows.

```
CNum   CID    Sts   Case Description   Starter           yyyy/ddd hh:mm
-----------------------------------------------------------------
1      1      (null) -D-    case1     swadmin@swnod012    2012/097 12:30
101    101    (null) -D-    test2     swadmin@swnod012    2012/097 14:31
201    201    (null) ---    test3     swadmin@swnod012    2012/097 14:33
```

Step Information

Command   The following command returns the information about steps:

```
plist -d [nodename] <procname>
```

Returns   The `plist -d` command returns the following information.

Column	Description
Flags	The meaning for each flag is as follows:
	• `W` auto-withdraw on deadline expire.
	• `V` view.
	• `R` dynamic rebuilding.
	• `O` output only.
	• `F` forwarding.
	• `E` edit.
	• `S` status.
	• `d` deadlines.
	• `a` actions.
	• `D` document.
	• `A` addressee.

Column	Description
Types	The meaning for each type is as follows:
	• 0 step.
	• 1 management report.
	• 2 script.
	• 3 event.
	• 4 EIS.
	• 5 SUBPROC.
	• 6 NULLSTPE.
	• 7 EAI STEP.
	• 8 INTERNAL.
	• 9 MULTISUBPROC.
	• 10 GRAFTSTEP.
	• 11 TCS.
Place	The unique number that is allocated to each step.
ExFlags	The meaning for each flag is as follows:
	• I specifies the step ignores the Case Suspend Function.
	• S specifies TIBCO iProcess Engine form.
	• P specifies the step ignores the Case Suspend Function.

Example   Input the *SWDIR*/util/plist -d maintem command where maintem is the procedure name. The output is shown as follows.

```
Flags        Type Name     Description                     Place   ExFlags   Lines
----------------------------------------------------------------------------------
------S-aDA   0    $PANIC1   Exception notification          1       0         4096
------S-aDA   0    $PANIC2   Exception resolution            2       0         4096
--R---SdaDA   0    TAKEAPP   Take Application                3       0         45
--R---SdaDA   0    REVAPP    Review Application              4       0         5
--R---S-aDA   0    NOTIFY    Manager Notification            5       0         4
--R---SdaDA   0    APPRAISE  Property Appraisal              6       0         10
--R---SdaDA   0    ACCTREV   Account Review                  8       0         13
--R---S-aDA   0    DECLINED  Application Declined            9       0         9
```

Field Information

Command The following command returns the information about fields:

```
plist -f [nodename] <procname>
```

Returns The `plist -f` command returns the following information.

Column	Description
Offset	The position in the `pack` file that the case data is kept.
Name	The field name.
Typ	The meaning for each type is as follows: • A ASC Ⅱ. • R real. • D date. • M management report field. • F memo. • C composite. • T time. • X attachment.
MID	The memo ID.
Marks	Not in use.
Flags	Not in use.

Example Input the *SWDIR*/util/plist -f maintem command where maintem is the procedure name. The output is shown as follows.

```
Offset Name            Typ  Len  DP/MID  Marks  Flags    Dbase
----------------------------------------------------------------
0        $PANICDIR      A   8    0       0      0000h
10       $PANICFLD      A   15   0       0      0000h
27       $PANICRSP      A   1    0       0      0000h
```

Step Status Information

Command The following command returns the information about step status:

```
plist -D [nodename] <procname>
```

Returns The `plist -D` command returns the following information.

Column	Description
Options	The meaning for each option is as follows: • `W` withdraw. • `O` outstanding. • `R` released. • `.` not processed.

Example Input the *SWDIR*/util/plist -D maintem command where maintem is the procedure name. The output is shown as follows.

```
Step Status Listing for procedure: TEM2051
Case  : $$TRNAADAR RARRARUE
        PPAEOPCECE EWEECEPD
        AAKVTPCCCM TACMCSDI
        NNEAIRTLEI RIEIRUAT
        IIAPFARIPN YTINELT
        CCPPYIENTD   VDVTE
        12P  SVEEE   E  S
             E DDR
========================
1452  : ..RRR..... ........
1453  : ..RRWRR.R. ........
1454  : ..RRWR.R.. ........
1455  : ..RRWRRR.. ........
1456  : ..RRWRR.R. ........
1502  : ..RRW..... ........
1503  : ..RRWRR.RR ........
1504  : ..RRWRR.R. ........
1505  : ..RRWR.R.. R.......
1552  : ..RRW..... ........
```

Audit Trail Information

Command   The following command returns the information about audit trail:

```
plist -a [nodename] <procedure> ALL|<casenum>
```

Returns   The `plist -a` command returns the following information.

Column	Description
Step Description	The description of the steps.
Event	The description of the event.
User	The name of the user who performed this audit event.
yyyy/ddd hh:mm	The date and time when the audit event occurred.

Example   Input the *SWDIR*/util/`plist -a maintem all` command where `maintem` is the procedure name. The output is shown as follows.

```
Step Description        Event          User             yyyy/mm/dd hh:mm
-----------------------------------------------------------------------
                        Start          Dev1@ipc_nod1    2012/04/06 12:10
Take Application        Processed To   Dev1@ipc_nod1    2012/04/06 12:10
Take Application        Released By    Dev1@ipc_nod1    2012/04/06 12:11
Review Application      Processed To   Agent@ipc_nod1   2012/04/06 12:11
Review Application      Expired        Agent@ipc_nod1   2012/04/06 12:11
Manager Notification    Processed To   Dev1@ipc_nod1    2012/04/06 12:11
Review Application      Released By    Dev1@ipc_nod1    2012/04/06 12:12
Manager Notification    Released By    Dev1@ipc_nod1    2012/04/06 12:30
                        Terminated                      2012/04/06 12:30
```

Group Membership Information

Command   The following command returns the information about group membership:

```
plist -G [nodename] <group>|<user>
```

Returns   The output varies according to whether a username or a group name is specified in the command.

Example   Input the *SWDIR*/util/plist -G super1 command where super1 is a group name. The output is shown as follows.

```
Groups for user: super1
-----------------------
OFFICER

Members of Group: officer
-------------------------
Administrator
Super2
Super3
Super4
Super99
Super1
```

Attribute Value Information

Command   The following command returns the information about attribute values:

```
plist -A [nodename] <group>|<user>
```

Returns   The `plist -A` command returns the following information.

Column	Description
Attribute	An item of information about a user.
Value	The specific informaion of the attribute for the user .

Example   Input the *SWDIR*/util/plist -A swadmin command where awadmin is a user name. The output is shown as follows.

```
Attribute          Value
-------------------------------------------
DESCRIPTION        Dev1
LANGUAGE           English
MENUNAME           PRODEF
QSUPERVISORS       Super1
SORTMAIL           PROCEDURE
USERFLAGS
```

Outstanding Mail Information

Command The following command returns the information about outstanding mail:

```
plist -o <procedure> ALL|<casenum> ALL|<queue> ALL|<node>
```

Returns The `plist -o` command returns the following information.

Column	Description
Casenum	Unique case number for this case.
Case Description	Cases description supplied when the case starts.
Stepname	The name of the outstanding step.
Addressee	The name of the queue that the outstanding step has been sent to.
CaseID	The unique ID of the work item.

Example Input the *SWDIR*`/util/plist -o maintem 1254 all all` command where `maintem` is the procedure, `1254` is the case number. The output is shown as follows.

```
Casenum   Case Description         Stepname    Addressee              CaseID
------------------------------------------------------------------------------
1254      mkw requesting vehicle   AUTH_OCS    user27@mkw2000            3629
```

Procedure Version Information

Command   The following command returns the information about procedure version:

```
plist -P [ALL [version|ALLVERS]|<procname> [version|ALLVERS]]
```

Returns   The `plist -P` command returns the following information.

Column	Description
Procedure	The name of the procedure.
Version	This column displays the procedure version number in *X.Y* format where: • *X* is the major version number. • *Y* is the minor version number.
User	User who created the procedure in this version.
Status	This column displays the status of the procedure: • Released • Incoplete • Unreleased • Model • Withdrawn
Comment	The unique ID of the work item.

Example   Input the *SWDIR*/`util/plist -P` command. The output is shown as follows.

```
Procedure:$EMAIL
Version User                          Status    Comment
-------------------------------------------------
 0.0    swadmin                       RELEASED  New Procedure Imported

Procedure:$SYSTEM
Version User                          Status    Comment
-------------------------------------------------
 0.0    swadmin                       RELEASED  New Procedure Imported
```

Queue Participation Information

Command   The following command returns the information about queue participation:

```
plist -Q
```

Returns   The `plist -Q` command returns the following information.

Column	Description
Queue Name	The name of the queue that this participation record allows users to participate in.
Days	Days of the week that users can participate in the specified Queue Name column. For example, `-TWT-SS` indicates every day except Monday or Friday.
Date Range	Date and time when participation starts and ends.
User	User who is allowed to participate.

Example   Input the *SWDIR*`/util/plist -Q` command. The output is shown as follows.

```
Queue Name              Days    Date Range                              User
-----------------------------------------------------------------------------
*swadmin                MTWTFSS                                         martynw
martynw                 MTWTF--2000/03/09 09:00 2000/03/10 18:00        swuser1
```

Queue Redirection Information

Command   The following command returns the information about queue redirection:

```
plist -R
```

Returns   The `plist -R` command returns the following information.

Column	Description
Queue Name	The name of the queue from which work items are redirected.
Destination	The name of the queue to which work items are redirected.
Start	The date and time when the queue redirection starts.
End	The date and time when the queue redirection ends.

Example   Input the *SWDIR*/util/plist -R command. The output is shown as follows.

```
Queue Name Destination     Start           End
----------------------------------------------------------------
martynw swadmin 2000/03/09 18:00 2000/03/10 09:00 2000/03/10 09:00
```

Pool Server Performance Information

Command   The following command returns the information about pool server performance:

```
plist -T[g] <pool svr id|ALL> [interval [count]]
```

Returns   The `plist -T` command returns the following information.

Column	Description
Pool Svr ID	The RPC Pool Server ID.
Svr Up Time	(Not in use.) Cases description supplied when the case starts.
Svr Work Time	(Not in use.) The RPC Pool Server work time.
Num Users	The total numbers of users who connected to the RPC Pool Server.
Date/Time	The date and time when the command was run.

Example   Input the *SWDIR*/util/plist -T ALL 60 60 command. The command example provides performance data for all RPC pool servers every 60 seconds for the next hour. The output is shown as follows.

```
Pool Svr ID  Svr Up Time        Svr Work Time      Num Users  Date / Time
-----------------------------------------------------------------------------
10              0                  0                  10         2012/04/25 11:53
```

Mbox Information

Command The following command returns the number of messages and the corresponding queue names that belong to the specified Mbox set:

```
plist -M <MBSetName[,MBSetName[,MBSetName]...][interval]>
```

where:

- `MBSetName` is the descriptive name of the Mbox set.
- `interval` is the time interval (in seconds) that the command returns an output.

The command keeps returning an output until you stop it. To stop running the command, press Ctrl+C.

Returns The `plist -M` command displays the following information.

Column	Description
Timestamp	The time when returning an output.
queue_name (the first column after the Timestamp row)	The descriptive name of the queue that belongs to the Mbox set you specified in the input. See Administering Message Queues and Mbox Sets on page 327 for more information about Mbox sets, message queues, and messages.
number_of_messages (the second column after the Timestamp row)	The total number of messages in the queue.

Example   Input the *SWDIR*`/util/plist -M BGMBSET, WMDMBSET 5` command where
`BGMBSET` and `WMDMBSET` are the name of the Mbox set, and `5` is the time interval.
The output is shown as follows.

```
Timestamp        2012-04-13_14:06:53
BGMBOX1          0
BGMBOX2          0
WISMBOX1         0
WISMBOX2         0
Timestamp        2012-04-13_14:06:58
BGMBOX1          0
BGMBOX2          0
WISMBOX1         0
WISMBOX2         0
```

Information about OS or UVAPI Users

Command   The following command returns the information about OS or TIBCO iProcess
User Validation API users:

```
plist -U
```

The command output depends on your setting in the *SWDIR*\etc\staffpms file.
If you specified the pathname of the user validation package on line 15 in the
staffpms file, then the output displays the validation system user accounts.
Otherwise, it displays the operating system user accounts.

For more information about TIBCO iProcess User Validation API, see *TIBCO
iProcess User Validation API User's Guide*. For more information about configuring
the staffpms file, see Specifying an External User Validation Package on page 32.

Returns   The plist -U command returns the following information.

Column	Description
Name	The name of the user.
Description	User description.

Example   Input the *SWDIR*/util/plist -U command. The output is shown as follows.

```
name                            description
----------------------------------------
Administrator\Administrator
ClientName\Dev1                 Dev1
ClientName\Dev2                 Dev2
ClientName\Dev3                 Dev3
ClientName\Dev4                 Dev4
```

System Event Information

Command The following command returns information about system events:

```
plist -e <typeid[,typeid...]|ALL> [username[,username...]|ALL
[startdate [enddate]]]
```

where:

- `typeid` is the ID of a system event.

  To review system events and their corresponding ID, see the `sysevents.cfg` file in the *SWDIR*/etc/english.lng directory.

- `username` is the iProcess user who performed the system event.

- `startdate` is the date from which the information about system events is returned. The format of this parameter is *DD/MM/YYYY*.

- `enddate` is the date after which the information about system events is not returned. The format of this parameter is *DD/MM/YYYY*.

Returns The `plist -e` command returns the following information:

Column	Description
Event	The ID of a system event.
User	The name of the iProcess user who performed this event.
yyyy/mm/dd	The date when this event occurred.
hh:mm:ss	The time when this event occurred.
Details	The description of this event.

Example   Input the *SWDIR*/util/plist -e ALL ALL command. The output is shown as follows.

```
Event User yyyy/mm/dd hh:mm:ssDetails
-------------------------------------------------------------------------------
48    pro 2014/06/13 12:48:46  Shutdown process (Machine ID = 1, Process Name = BG,
Process Instance = 4)
48    pro 2014/06/13 12:48:46  Shutdown process (Machine ID = 1, Process Name = BG,
Process Instance = 3)
40    pro 2014/06/13 14:51:56  Shutdown Sentinel (Machine ID = 1, Machine Name =
DESPINA)
```

Index

X