TIBCO iProcess® Objects Server

Administrator's Guide

Software Release 11.6
January 2016

Two-Second Advantage®

Important Information

Contents

Preface

This guide provides an overview of TIBCO iProcess Objects Server, as well as information about starting, stopping and configuring a TIBCO iProcess Objects Server. It also includes information about using the logs produced by TIBCO iProcess Objects Server — TIBCO iProcess Objects Server log, audit log, and UNIX system log.

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*

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

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.
`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]`

Table 1 General Typographical Conventions (Cont'd)

Convention	Use
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`
`|`	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`

Table 2 Syntax Typographical Conventions (Cont'd)

Convention	Use

`{ }`	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 TIBCOmmunity

TIBCOmmunity 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. TIBCOmmunity offers forums, blogs, and access to a variety of resources. To register, go to http://www.tibcommunity.com.

How to Access TIBCO Documentation

Documentation for this and other TIBCO products is available on the TIBCO Documentation site:

https://docs.tibco.com

Documentation on the TIBCO Documentation site is updated more frequently than any documentation that might be included with the product. To ensure that you are accessing the latest available help topics, please visit us at 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 Introduction

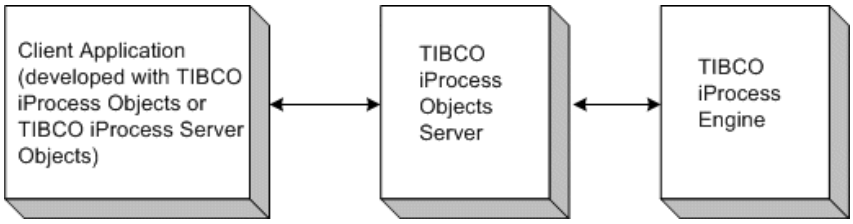
This chapter provides an overview of TIBCO iProcess Objects Server.

Topics

Overview

The TIBCO iProcess Objects Server receives requests for services or data from a client application developed with either TIBCO iProcess Objects (COM, Java, or C++) or TIBCO iProcess Server Objects (Java or .NET). It processes the request, then makes the appropriate call to a TIBCO iProcess Engine to initiate the desired service or obtain the desired information.

Therefore, the TIBCO iProcess Objects Server acts as a gateway between the client application and the TIBCO iProcess Engine, as follows.



The client application must establish a connection with a TIBCO iProcess Objects Server. This can be accomplished in a variety of ways — see *TIBCO iProcess Objects Programmer's Guide* or *TIBCO iProcess Server Objects Programmer's Guide* for information.

After communication is established between the client and TIBCO iProcess Objects Server, TIBCO iProcess Objects Server waits for request messages from the client. When TIBCO iProcess Objects Server receives a request message, it in turn makes calls to the TIBCO iProcess Engine to perform functions such as locking work items, moving work items to other work queues, writing data to the database, etc.

Message Timeout

As described above, client applications make requests to TIBCO iProcess Objects Server, then wait for a response. Because of this, you may have a desire to configure the client so that if a specified period of time elapses waiting for a response from the server, the client will timeout and generate an error. To configure this "message wait time," you must add a Registry key (Windows systems) or environment variable (UNIX systems), and set it to the number of milliseconds you would like the client to wait before timing out.

Registry key for TIBCO iProcess Objects:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Staffware plc\Staffware SEO Client\MessageWaitTime
```

Registry key for TIBCO iProcess Server Objects:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Staffware plc\Staffware SSO Client\MessageWaitTime
```

If the software is installed on a 64-bit machine, the Registry path will include "Wow6432Node", as follows:

`HKEY_LOCAL_MACHINE\SOFTWARE\`*Wow6432Node*`\Staffware plc\...`

Environment variable:

`MessageWaitTime`

If the number of milliseconds specified by MessageWaitTime is exceeded, the client will generate an swTimeoutErr error. If MessageWaitTime is set to 0 (zero), the client will not timeout. By default (i.e., if you do not set MessageWaitTime), Windows clients timeout in 30 seconds; UNIX clients timeout in 60 seconds.

TIBCO iProcess Objects Server Version

Prior to version 10.2.0, TIBCO iProcess Objects Servers were called "Tibco iProcess Objects (SPO) Servers". They had version numbers either with or without an "i". Servers with an "i" (e.g., i10.0(4.0)) were used with TIBCO iProcess Engines; Servers without an "i" (e.g., 9.3(5.0)) were used with TIBCO Process Engines.

From version 10.2.0 forward, these servers are called TIBCO iProcess Objects Servers and their version number will always be 3 digits, with no "i"; these servers will be used with TIBCO iProcess Engines.

You can determine the version of your TIBCO iProcess Objects Server by executing one of the following:

- *SWDIR*`\bin\swentobjsv -v` (Windows) or $*SWDIR*`/bin/swentobjsv -v` (UNIX)
- `what` $*SWDIR*`/bin/swentobjsv` (UNIX only) - The output of this command contains the same as the `swentobjsv -v` command. However, using this command does not require having the *SWDIR* or any other environment variables configured. The `what` command will also run even if there is a compatibility problem between the iProcess Objects Server and engine libraries in $*SWDIR*`/libs`, and will run even if the engine libraries are missing.

Note that internally the TIBCO iProcess Objects Server version number is still in the older format. Therefore, the commands above will show version 10.2.1 as version i10.2(1.0).

Starting/Stopping the TIBCO iProcess Objects Server

The TIBCO iProcess Objects Server runs as a process under the control of the Process Sentinels. The Process Sentinels can be configured to automatically start the TIBCO iProcess Objects Server process, or you can issue a request to start or stop the process using the `swsvrmgr` utility, as follows:

Windows systems:

```
SWDIR\util\swsvrmgr START <MachineID> SPO <ProcessInst>
SWDIR\util\swsvrmgr SHUTDOWN <MachineID> SPO <ProcessInst>
```

UNIX systems:

```
$SWDIR/util/swsvrmgr START <MachineID> SPO <ProcessInst>
$SWDIR/util/swsvrmgr SHUTDOWN <MachineID> SPO <ProcessInst>
```

Note that the TIBCO iProcess Objects Server process is also shut down if the Process Sentinels are stopped. For information about stopping the Process Sentinels, see *TIBCO iProcess Engine Administrator's Guide*.

Start / Stopping a UNIX TIBCO iProcess Objects Server as the Background User

To be able to start or stop a UNIX TIBCO iProcess Objects Server, you must be logged in as either the root user or the background user (by default, pro). If you want to be able to start/stop a UNIX TIBCO iProcess Objects Server when logged in as the background user, you must set the UNIX kernel parameter that defines the hard limit for the maximum number of file descriptors per process to a value of at least the number specified in the `NumFiles` parameter (see NumFiles (UNIX Only) on page 29). If you do not do this, you will need to be logged in as the root user. (The UNIX kernel parameter that defines the hard limit for the maximum number of file descriptors per process is platform-specific. See your UNIX documentation for more information about which parameter you need to set, how to set it, and the effect on your system of setting it.)

Note that if you want the background user to be able to start the Process Sentinels (which in turn starts the TIBCO iProcess Objects Server), you must set the hard limit as described in the previous paragraph.

Running Multiple Instances of the TIBCO iProcess Objects Server

You can run multiple instances of the TIBCO iProcess Objects Server. This is done in the following ways:

To be able to run multiple instances of TIBCO iProcess Objects Server, your server must have CR 10974 implemented.

- On Multiple Machines in a Node Cluster

  If you've saturated the resources of a single machine, you can add TIBCO iProcess Objects Servers to other machines in the cluster, allowing you to spread the load across multiple machines. All nodes in the cluster share the same database.

  To run the TIBCO iProcess Objects Server on multiple machines, it must be installed on each of the machines in the cluster on which it will be running. As an example, if the TIBCO iProcess Objects Server is installed on two machines in the cluster, the `process_config` table will appear as follows:

Machine ID	Process Name	Process Instance
1	SPO	1
2	SPO	1

  This shows that there is a single instance of the TIBCO iProcess Objects Server installed on each of two machines.

- On a Single Machine

  Multiple instances of the TIBCO iProcess Objects Server can be run on a single machine, resulting in all TIBCO iProcess Objects Servers running from the same **$***SWDIR* directory and using the same database. This allows you to run multiple TIBCO iProcess Objects Servers without requiring you to have a cluster. The reasons for running multiple instances of TIBCO iProcess Objects Server on the same machine are:

  — It increases throughput by reducing SAL threading contention. Each instance has its own copy of the SAL, so by spreading the same number of users over several instances (and SALs), contention can be reduced. (Threads in one process do not contend against threads in another process trying to enter the same critical section.)

  — It avoids the process-size limitation that is imposed in some operating systems. Since a large component of the size of a TIBCO iProcess Objects

Server is user-related data, a TIBCO iProcess Objects Server with fewer users will be smaller. Therefore, spreading a given user load over multiple servers yields smaller servers.

Adding and Deleting Instances of the TIBCO iProcess Objects Server

When the TIBCO iProcess Objects Server is initially installed on a machine, it becomes instance 1 by default. A new installation of a TIBCO iProcess Objects Server on a machine will cause an entry for that server to be automatically added to the `process_config` table, as follows:

Machine ID	Process Name	Process Instance
1	SPO	1

Once the initial installation is completed, additional instances of the TIBCO iProcess Objects Server can then be added to or deleted from the `process_config` table using the following `swadm` commands:

Windows systems:

SWDIR`\util\swadm add_process` *MachineID* `SPO Y`
SWDIR`\util\swadm delete_process` *MachineID* `SPO` <*ProcessInst*>

UNIX systems:

`$`*SWDIR*`/util/swadm add_process` *MachineID* `SPO Y`
`$`*SWDIR*`/util/swadm delete_process` *MachineID* `SPO` <*ProcessInst*>

For example, after adding a second instance to machine 1, the `process_config` table appears as follows:

Machine ID	Process Name	Process Instance
1	SPO	1
1	SPO	2

For additional information about using the `swadm` utility, see *iProcess Engine Administrator's Guide.*

Number of Instances

The TIBCO iProcess Engine does not impose a limitation on the number of instances of a process running on a machine in the node cluster. The maximum number of instances is really a function of the amount of resources available on the machine. In reality, it does not seem reasonable nor practical to run more than 16 TIBCO iProcess Objects Server instances on a machine. However, the TIBCO iProcess Objects Server is coded to limit the number of instances per machine to 99. You will still be able to configure more than 99 instances per machine in the Process Manager, but any TIBCO iProcess Objects Server instance greater than 99 will generate an error status to the Process Manager and exit gracefully. This error will be seen in the "Last Status" column when running “swadm show_processes”.

Implementation

To run multiple TIBCO iProcess Objects Servers against one TIBCO iProcess Engine, you must specify the TCP and UDP ports, as appropriate, for each instance of the TIBCO iProcess Objects Server.

For information about configuring TCP ports for multiple instances, see the `TCPServiceName` parameters on This identifies the port number on which the TIBCO iProcess Objects Server will listen for client connections. This can be specified in the following ways: on page 32; for information about configuring UDP ports for multiple instances, see the `UDPServiceName` parameter on UDPServiceName on page 39.

Starting/Stopping Multiple TIBCO iProcess Objects Servers

The Process Sentinels control the starting and stopping of multiple TIBCO iProcess Objects Servers that have been added to the `process_config` table.

By default, once the Process Sentinels have started, they automatically start the processes in the `process_config` table. You can force all processes in the `process_config` table to be started or stopped by using the following commands:

Windows systems:

SWDIR`\bin\swstart`
SWDIR`\bin\swstop`

UNIX systems:

`$`*SWDIR*`/bin\swstart`
`$`*SWDIR*`/bin\swstop`

Or, you can start or stop individual TIBCO iProcess Objects Server instances using the following commands:

Windows systems:

```
SWDIR\util\swsvrmgr START MachineID SPO <ProcessInst>
SWDIR\util\swsvrmgr SHUTDOWN MachineID SPO <ProcessInst>
```

UNIX systems:

```
$SWDIR/util/swsvrmgr START <MachineID> SPO <ProcessInst>
$SWDIR/util/swsvrmgr SHUTDOWN <MachineID> SPO <ProcessInst>
```

For more information about the `swsvrmgr` utility, see *iProcess Engine Administrator's Guide*.

Configuration Parameter Instances

All of the TIBCO iProcess Objects Server configuration parameters (except NumFiles, AnonymousUserName'X', AnonymousSWUserName'X', AnonymousPoolSize'X' and AnonymousPrivilege'X') can be designated for a particular instance, allowing you to configure each instance of the TIBCO iProcess Objects Server differently. The way in which this is done depends on whether you are using a UNIX or Windows TIBCO iProcess Objects Server, as follows:

- The UNIX TIBCO iProcess Objects Server includes a configuration file that specifies the values assigned to each TIBCO iProcess Objects Server configuration parameter. You can define a configuration parameter for each instance of the TIBCO iProcess Objects Server by appending the instance number to the parameter name (e.g., LogFileMaxSize03). For more information, see Configuring Multiple Instance Parameters on UNIX Systems on page 18.
- The Windows TIBCO iProcess Objects Server includes a configuration utility for administering configuration parameters. The configuration utility allows specification of parameters for each instance of the TIBCO iProcess Objects Server running on that machine. For more information, see Configuring Multiple Instance Parameters on Windows Systems on page 19.

Specifying TCP and UDP ports for multiple instances of the TIBCO iProcess Objects Server works a little differently than the other configuration parameters. See `TCPServiceName` on page 32 and `UDPServiceName` on page 39 for more information.

Log File Names

If you are using a TIBCO iProcess Objects Server that is multiple-instance capable, the name of the TIBCO iProcess Objects Server log, archive log, and audit log will include the instance number (*XX*) of the TIBCO iProcess Objects Server instance. The *timestamp* variable in the log file names:

- The audit log and TIBCO iProcess Objects Server archive log files: the date when the log is generated.
- TIBCO iProcess Objects Server log files: 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 `SWEntObjSv`*XX*`_timestamp.log`, where the *timestamp* variable is the date when that log is generated.

See Name and Location of the TIBCO iProcess Objects Server Log on page 75 for more information.

Log File	Windows	UNIX

TIBCO iProcess Objects Server log	`SWEntObjSv`*XX*`.log`
	or
	`SWEntObjSv`*XX*`_timestamp.log`[1]
		`SWEntObjSv`*XX*`.log`
		or
		`swentobjsv`*XX*`_timestamp.log`

Audit log	`SWEntObjUa`*XX*`_timestamp.log`	`swentobjua`*XX*`_timestamp.log`
TIBCO iProcess Objects Server archive log[2]	`SWEntObjSv`*XX*`_timestamp_archive_`*xxx*`.log`	`swentobjsv`*XX*`_timestamp_archive_`*xxx*`.log`

1. This file is the archived `SWEntObjSv`*XX*`.log`, where the *timestamp* variable is the date when the log is generated.
2. For more information about the archive log, see LogFileMaxArchives on page 48.

Accessing Multiple Instances of the TIBCO iProcess Objects Server

The way in which you access multiple instances of the server depends whether you are using TIBCO iProcess Objects or TIBCO iProcess Server Objects, as described in the following subsections.

Using TIBCO iProcess Objects

The following are methods you can use to access multiple instances of the TIBCO iProcess Objects Server using TIBCO iProcess Objects:

- UDP Broadcast - Each SWNodeInfo object that is returned from a UDP broadcast contains information about which instance of the TIBCO iProcess Objects Server the object represents. The SWNodeInfo object key includes the instance number as follows:

  `ComputerName|NodeName|IsDirector|InstanceNumber`

- Directed UDP Message - The AddNodeEx method is used to send a directed UDP message to a specific instance of the TIBCO iProcess Objects Server. This method contains an *InstanceNumber* parameter to specify the instance.
- Manually Create an SWNodeInfo Object- The MakeNodeInfoEx method is used to create an SWNodeInfo object for a specific instance of the TIBCO iProcess Objects Server. This method contains an *InstanceNumber* parameter to specify the instance.
- TIBCO iProcess Objects Director - The TIBCO iProcess Objects Director is a standalone program that chooses a TIBCO iProcess Objects Server for you. The method it uses to choose a server depends on how the TIBCO iProcess Objects Director is configured.

For more information about these methods of accessing multiple instances of the TIBCO iProcess Objects Server, see *TIBCO iProcess Objects Programmer's Guide*.

Using TIBCO iProcess Server Objects

The following are methods you can use to access multiple instances of the TIBCO iProcess Objects Server using TIBCO iProcess Server Objects:

- UDP Broadcast - Calling the `getNodes` method on sNodeManager causes a UDP broadcast to be issued on the LAN segment. The getNodes method returns an array of vNode objects, one for each TIBCO iProcess Objects Server that responded to the UDP broadcast. You can then call the `getInstance` method on the vNode object to determine the instance number of that TIBCO iProcess Objects Server.
- Directed UDP Message - Calling the `verifyNode` method on sNodeManager causes a directed UDP message to be sent to a specific instance of the TIBCO iProcess Objects Server. This method provides an `aInstance` parameter to specify the specific instance of the TIBCO iProcess Objects Server to receive the UDP message.
- Manually Create a vNodeId Object- You can construct a vNodeId object that represents the specific instance of the desired TIBCO iProcess Objects Server. Note that the constructor for the vNodeId object does not have an `Instance` parameter — you must use the `aTCPPort` parameter to uniquely identify the instance of the TIBCO iProcess Objects Server the vNodeId object is to represent.

- TIBCO iProcess Objects Director - The TIBCO iProcess Objects Director is a standalone program that chooses an TIBCO iProcess Objects Server for you. The method it uses to choose a server depends on how the TIBCO iProcess Objects Director is configured.

For more information about these methods of accessing multiple instances of the TIBCO iProcess Objects Server, see *TIBCO iProcess Server Objects Programmer's Guide*.

Multiple Instance Limitations

The following limitations apply when running multiple instances of TIBCO iProcess Objects Servers:

- The same user logging into multiple instances consumes one license per instance. This is because each instance must create a separate SAL session for the user.
- An SWXList (TIBCO iProcess Objects) or pageable list (TIBCO iProcess Server Objects) of work items or predicted work items is tied to a specific server instance. If an SWXList/pageable list of work items or predicted work items is created, that list can only be accessed on the server instance where it was created. This is not just limited to getting the work items on an SWXList/pageable list, but also to the method calls on work items obtained from an SWXList/pageable list. This is because it holds state to the Work Item Server. However, all other operations can be performed against any instance in the cluster (bearing in mind the licensing limitation above).
- Since each instance will cache the procedures, user, groups, roles, attributes, tables and lists, more memory will be required system-wide due to duplication of this information.

Chapter 2 Configuring the TIBCO iProcess Objects Server

This chapter describes the configuration parameters available to configure the TIBCO iProcess Objects Server.

Topics

Configuration Parameters

The TIBCO iProcess Objects Server is configured using configuration parameters. The operating system you are using will determine how you modify the parameters:

- Windows - In Windows, configuration parameters are modified using *TIBCO iProcess Objects Server Configuration Utility*. The configuration utility is available as a Control Panel applet (note that if the server is installed on a 64-bit machine, the utility applet is found under "View x86 Control Panel Icons" on the Control Panel) or by executing *SWDIR*`\bin\SWEntObjSvCfg.exe`.

  The TIBCO iProcess Objects Server stores all configuration parameters in the Windows Registry in the following key:

```
\HKEY_LOCAL_MACHINE\SOFTWARE\Staffware plc\Staffware EntObj Server\Nodes
```

If the software is installed on a 64-bit machine, the Registry path will include "Wow6432Node", as follows:

`HKEY_LOCAL_MACHINE\SOFTWARE\`*`Wow6432Node`*`\Staffware plc\...`

Each TIBCO iProcess Objects Server keeps its configuration under a key with the same name as the node name of the server. Note that it is highly recommended that you use the configuration utility to modify parameters rather than editing the registry directly.

If any configuration parameters are missing when the TIBCO iProcess Objects Server is started, they will be automatically created with default values. The TIBCO iProcess Objects Server will not start if any configuration parameters have invalid values.

- UNIX - When using the supported UNIX operating systems (AIX, HP-UX, etc.), the configuration parameters are stored in the text file $*SWDIR*`/seo/data/swentobjsv.cfg`. The parameters in this file have the format:

  ConfigKey = *Values*

  where:

  ConfigKey is the name of the configuration parameter.

  Value is the value of the parameter.

Lines beginning with a # are ignored and can be used for comments or for temporary changes. Blank lines are ignored and can be used to improve the readability of the file.

If any configuration parameters are missing when the TIBCO iProcess Objects Server is started, they will be automatically created with default values. If any of the configuration parameters contain invalid values, the TIBCO iProcess Objects Server will not start.

To make changes to the configuration file, you must first stop the UNIX TIBCO iProcess Objects Server, make your changes, then restart the server.

iProcess Objects Server Configuration File When Upgrading

[UNIX only] If you are upgrading TIBCO iProcess Objects Server from an earlier version on a UNIX system, the installation program will *not* overwrite your server configuration file, $*SWDIR*`/seo/data/swentobjsv.cfg`, if it exists. (Since this is an upgrade, the only reason the configuration file wouldn't already exist is if it has been manually moved/deleted). The installation program won't overwrite your existing configuration file because you may have customized it for your particular needs.

- If the `swentobjsv.cfg` file does not exist, the installation program will install both an `swentobjsv.cfg` file and a `sample.cfg` file in the $`SWDIR/seo/data` directory. These files will be exactly the same.
- If the `swentobjsv.cfg` file does exist, the installation program will install only a `sample.cfg` file in the $*SWDIR*`/seo/data` directory.

In both cases above, the newly installed configuration file (`swentobjsv.cfg` and/or `sample.cfg`) is the configuration file for the version of the server to which you are upgrading. It may contain new configuration parameters or other changes. Note, however, that all configuration parameters in the new configuration file(s) are commented out — if used as is, iProcess Objects Server will use the default values for all parameters.

If the new configuration file contains new/modified information, you can copy that information and paste it into your configuration file. (You can use the diff program (`diff swentobjsv.cfg sample.cfg`) to determine what is different between your configuration file and the new configuration file.)

Whether or not you make any modifications, the $*SWDIR*`/seo/data` directory must contain a `swentobjsv.cfg` file, which will be used to configure iProcess Objects Server when it is started.

Also, if any of the configuration parameters contain invalid values, iProcess Objects Server will not start. Note that since iProcess Objects Server log file is not opened until after the configuration file is read, if there is an error in the configuration file, it is not written to the log file. Instead, it is written to the $*SWDIR*`/logs/seo_error` file (this file is created when iProcess Objects Server starts, so the presence of the file does not mean there were errors).

Multiple Instances of Configuration Parameters

Although this section primarily pertains to setting configuration parameters when running multiple instances of the TIBCO iProcess Objects Server, some of it is also applicable when setting parameter values for non-multiple-instance TIBCO iProcess Objects Servers.

Multiple instances of the TIBCO iProcess Objects Server can be run on a single machine. See Running Multiple Instances of the TIBCO iProcess Objects Server on page 6 for more information.

If the multiple instance feature is being used, you can also specify multiple instances of the configuration parameters described in this chapter. This allows you to configure each instance of the TIBCO iProcess Objects Server differently. You can define multiple instances of all configuration parameters except NumFiles, AnonymousUserName'X', AnonymousSWUserName'X', AnonymousPoolSize'X' and AnonymousPrivilege'X'.

Configuring Multiple Instance Parameters on UNIX Systems

To specify a parameter for a particular instance, add "01" to the parameter name for the first instance, "02" for the second and so on. Any instance that does not have an individual parameter set will use the "common" parameter.

For example, if the following is the only "LogFileMaxSize" parameter specified:

```
LogFileMaxSize = 50
```

Then all instances will use a log size of 50MB.

If the following "LogFileMaxSize" parameters have been defined:

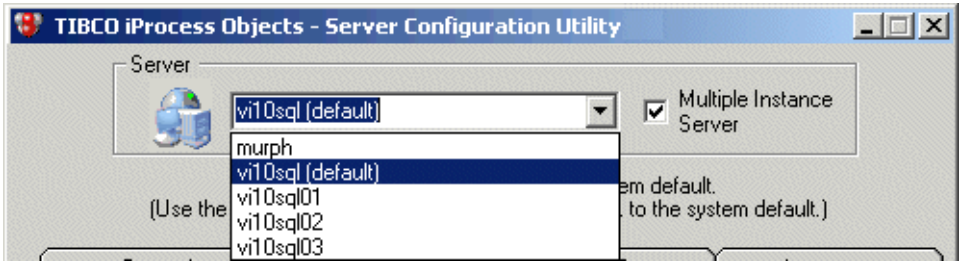
```
LogFileMaxSize = 50
LogFileMaxSize03 = 100
```

Then instance 3 of the TIBCO iProcess Objects Server will have a 100MB file size, while all other instances will have a 50MB file size.

The `TCPServiceName and UDPServiceName` parameters work a little differently. For information about how these parameters work with multiple instances of TIBCO iProcess Objects Servers, see This identifies the port number on which the TIBCO iProcess Objects Server will listen for client connections. This can be specified in the following ways: on page 32 and UDPServiceName on page 39.

Configuring Multiple Instance Parameters on Windows Systems

The TIBCO iProcess Objects Server Configuration Utility allows you to select each instance of the TIBCO iProcess Objects Server on the machine so that parameters can be configured for that instance:



If the TIBCO iProcess Objects Server is multiple-instance capable (i.e., it has CR 10974 implemented), the Multiple Instance Server check box will be checked when that server is selected in the server drop-down list. Also, for each TIBCO iProcess Objects Server that is multiple-instance capable, the server name will be shown in the server drop-down list with "(default)" to the right of the name. Then, each instance of that TIBCO iProcess Objects Server is shown in the list with a two-digit instance number appended to the server name. In the example shown above, the TIBCO iProcess Objects Server named "vi10sql" has three instances.

If the TIBCO iProcess Objects Server is multiple-instance capable, but you are not running multiple instances of that server, there will still be two entries for that server in the drop-down list: "`<ServerName>`(default)" and "`<ServerName>`01".

Setting Default Parameter Values When Using the Configuration Utility

There are two types of default values for the TIBCO iProcess Objects Server configuration parameters

- System defaults - These are the values to which all parameters are set when the TIBCO iProcess Objects Server is initially installed. These default values are established by the system.

- Multiple-instance defaults - These are values that you can establish as the default values for a TIBCO iProcess Objects Server that is multiple-instance capable. Initially, these default values are the same as the system default values. To establish the default values for a particular TIBCO iProcess Objects Server, select the "<ServerName>(default)" entry for the desired server from the server drop-down list, then change the desired values.

The TIBCO iProcess Objects Server Configuration Utility has been designed so that the color of the text indicates if a value is different from either the system default or the multiple-instance default.

- Values that are the same as the system default are displayed in black.
- Values that are different than the system default are displayed in blue. Note that this also applies to parameters for TIBCO iProcess Objects Servers that are non-multiple-instance servers.
- Values that are different than the multiple-instance default are displayed in red. This, of course, only applies to TIBCO iProcess Objects Servers that are multiple-instance capable.

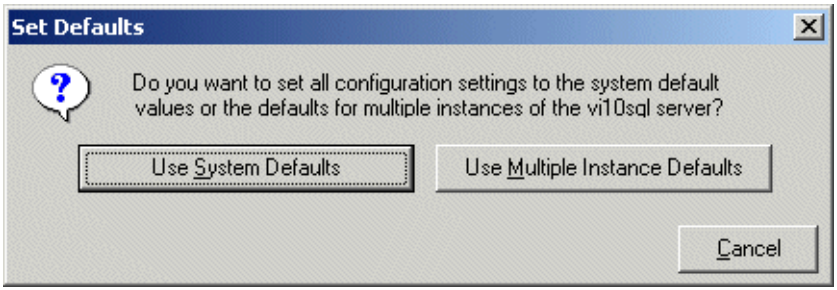
Setting Values to the Defaults

You can set parameter values back to the default values using the following methods:

- Set All Defaults button - This button, which is displayed on the bottom of the TIBCO iProcess Objects Server Configuration Utility window, allows you to set the parameter values to either the system defaults or the multiple-instance defaults, as follows:
  - — If a non-multiple-instance TIBCO iProcess Objects Server is selected in the server drop-down list, clicking Set All Defaults causes all parameters for that server to be set to the system defaults.
  - — If the "(default)" selection for a multiple-instance capable TIBCO iProcess Objects Server is selected in the server drop-down list, clicking Set All Defaults causes the multiple-instance defaults for that TIBCO iProcess Objects Server to be set to the system defaults. This also causes the parameters for instances of that TIBCO iProcess Objects Server to take on the system default values, but ONLY if they had not previously been modified. In other words, any parameter values that had been changed for a particular instance of the TIBCO iProcess Objects Server is not modified

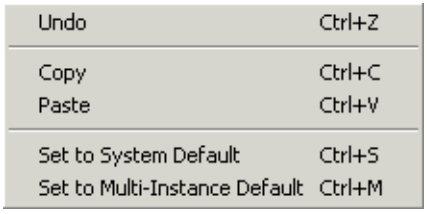
when you set the default parameters for that TIBCO iProcess Objects Server.

— If one of the instances of the TIBCO iProcess Objects Server is selected in the server drop-down list (e.g., vi10sql02 in the example shown earlier), clicking on Set All Defaults causes the following dialog to be displayed:



From this dialog, you can set all of the parameters for that instance of the TIBCO iProcess Objects Server to the system defaults or the multiple-instance defaults.

- Right-click drop-down menu - Right-clicking on an individual field or check box in the TIBCO iProcess Objects Server Configuration Utility displays the following menu.



This menu allows you to set that specific parameter to either the system default or multiple-instance default. Note that the Set to System Default selection is available only if the parameter value is currently different than the system default, and the Set to Multi-Instance Default selection is available only if the parameter value is currently different than the multiple-instance default.

This right-click drop-down menu also provides selections that allow you to perform standard Windows copy, paste, and undo functions.

Accessing Configuration Parameters Through the Object Model

Each of the configuration parameters can be accessed through TIBCO iProcess Objects or TIBCO iProcess Server Objects object models using the following properties/methods:

- TIBCO iProcess Objects (COM): ConfigInfos property on SWNodeInfoEx
- TIBCO iProcess Objects (Java and C++): `getConfigInfos` method on SWNodeInfoEx
- TIBCO iProcess Server Objects (.NET): ConfigInfos property on vANode
- TIBCO iProcess Server Objects (Java): `getConfigInfos` method on vANode

These properties/methods all return a list of objects, each representing one of the TIBCO iProcess Objects Server configuration parameters, as well as the value of each of the parameters. See the following pages for the names of the available parameters.

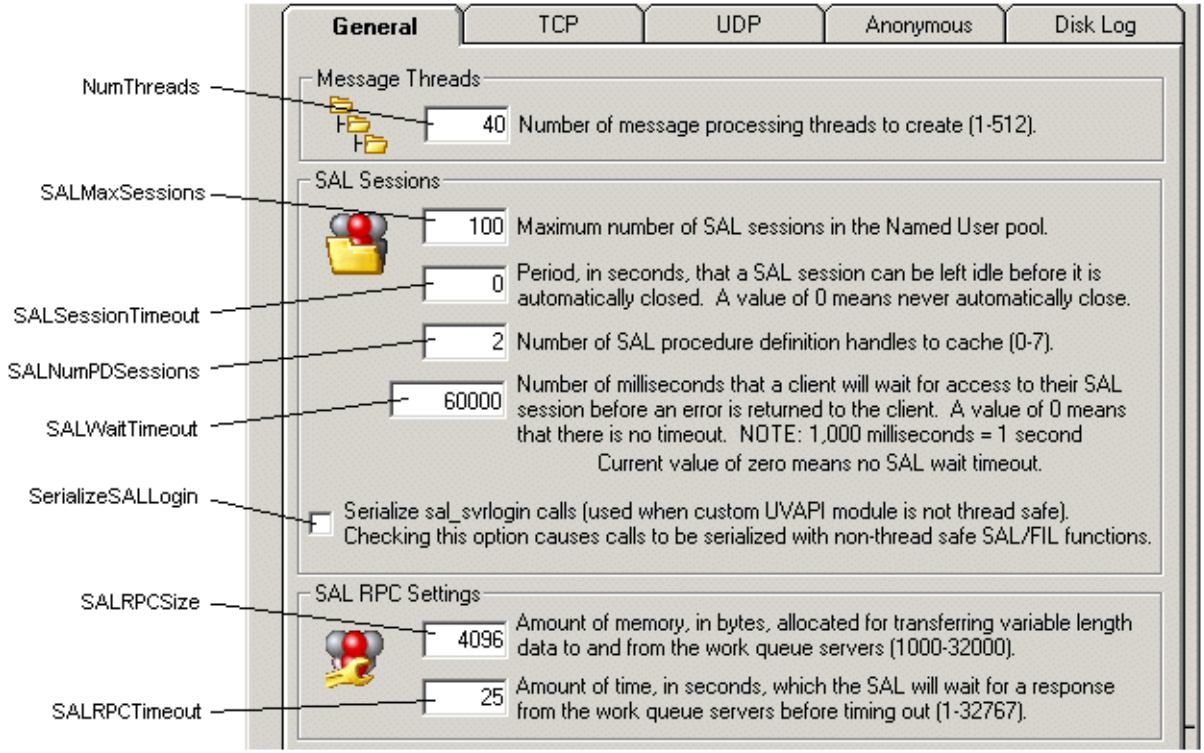
In addition to the TIBCO iProcess Objects Server configuration parameters, the following parameters are also returned by the ConfigInfos/getConfigInfos property/method. These additional parameters allow you to determine if the TIBCO iProcess Objects Server supports new features that have been added to the server:

- UVAPISupported - Indicates if the User Validation API is supported. This feature was added to the TIBCO iProcess Objects Server in CR 10355.
- QCountsAudMsgSupported - Indicates if the audit text message is available on the SWAuditStep object (in TIBCO iProcess Objects) or the vAuditStep object (in TIBCO iProcess Server Objects). This feature was added to the TIBCO iProcess Objects Server in CR 12407.
- MultipleInstanceSupported - Indicates if multiple instances of the TIBCO iProcess Objects Server can be run. This feature was added to the TIBCO iProcess Objects Server in CR 10974.
- XPCCaseFilteringSupported - Indicates whether or not all cases will be filtered by the database. This feature was added to the TIBCO iProcess Objects Server in CR 13182.
- F3WISFilteringSupported - Indicates whether or not all work items will be filtered by the Work Item Server (WIS). This feature was added to the TIBCO iProcess Objects Server in CR 12744.
- MemosSupported - Indicates if fields of type swMemo are supported. This feature was added to the TIBCO iProcess Objects Server in CR 8427.
- SmartHeapSupported - Indicates if SmartHeap was compiled and linked into TIBCO iProcess Objects Server.

- SmartHeapVersion - Contains the version number, in the format XX.XX.XX, if SmartHeap is supported.

These parameters (with the exception of SmartHeapVersion) will return a value of 1 if the TIBCO iProcess Objects Server supports that feature; they will return a value of 0 if the TIBCO iProcess Objects Server does not support the feature.

General Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

NumThreads

Number of message processing threads to create. This specifies the size of the pool of threads in the TIBCO iProcess Objects Server available for processing requests from clients. Note that this sets the TOTAL number of message processing threads, not the number per processor.

Thread information can be monitored using SWNodeInfoEx.ThreadInfos (TIBCO iProcess Objects), vANode.getThreadInfos (TIBCO iProcess Server Objects (Java)), or vANode.ThreadInfos (TIBCO iProcess Server Objects (.NET)). You may need to increase the number of threads if the threads are always busy, you've added more CPUs, you're running short transactions, or you've added more users.

Note that if you change the value of this parameter, you may also need to change the value of the MAXPOOLSIZE parameter, which is specified in the $*SWDIR*/etc/staffcfg (UNIX) or *SWDIR*\etc\staffcfg (Windows) file. This depends on whether or not your TIBCO iProcess Objects Server has CR 14735 implemented. If your TIBCO iProcess Objects Server contains CR 14735, you do not need to set MAXPOOLSIZE. If it does not contain CR 14735, you must set MAXPOOLSIZE according to the information below.

To ensure that you do not run out of database connections, you must make sure that the value of the MAXPOOLSIZE parameter in the staffcfg file is set to the proper value. Running out of database connections can result in the TIBCO iProcess Objects Server failing with an ER_SYSTEM error. The value to set MAXPOOLSIZE depends on the number of threads being used. Therefore, to determine its value, you must use the value of the NumThreads configuration parameter. Use the following formula to determine the value to set the MAXPOOLSIZE parameter:

```
((NumThreads + 3) * 2) + 5
```

Values:

— 1 to 512 (Windows)

— 1 to 32,767 (UNIX)

Default Value

The function for calculating the default number of message processing threads is as follows:

```
Default value = NumCPUs*DFLTMAXTHREADS*Multipler
Multipler = MAX (1, (2.4 - LOG (NumCPUs)))
```

where:

- NumCPUs is the number of processors on the system.
- DFLTMAXTHREADS is the default value, 5.

Run the following command to display the default number of message processing threads:

- On Microsoft Windows

  SWDIR\bin\swentobjsv -v

- On UNIX

  $*SWDIR*`/bin/swentobjsv -v`

The default number of message processing threads is prompted as follows:

```
# of CPU's   : Number_of_CPU
# of Threads: Number_of_Threads
```

The upper range of the allowable values for this parameter was changed a couple of times in previous versions of the TIBCO iProcess Objects Server. The upper range for pre-version 8.1 servers is 31; for versions from 8.1 to 9.0(0.5), it is 512; for version 9.0(0.6) and later, it is 32,767.

Be aware that if you are running multiple TIBCO iProcess Objects Servers on the same machine, and they are different versions that allow a different upper range for NumThreads, *TIBCO iProcess Objects Server Configuration Utility* will allow you to set the upper limit up to 32,767. Ensure that you specify only the number of threads allowed for the version of server you are configuring.

SALMaxSessions

The number of SAL sessions to pool in the general user pool. (Anonymous users use their own pool (see `Anonymous` parameters on Anonymous Parameters on page 41).)

Windows Systems

The value of this parameter should be at least as large as the maximum number of concurrent users you expect to have on the system. If your server has restrictions on amounts of available memory, you may want to set this value lower than the maximum number of concurrent users. However, this will cause the TIBCO iProcess Objects Server to context switch SAL sessions by logging users in and out of its SAL session cache (this context switching is transparent to clients). This context switching does not provide optimal performance, but conserves memory usage of the TIBCO iProcess Objects Server. Only set the value of this parameter to less than the maximum number of concurrent users when making the memory usage of the TIBCO iProcess Objects Server smaller is more important than response time.

You will get better performance by adding memory and increasing the value of this parameter.

Lower Bound: 1
Upper Bound: none
Default: 100

UNIX Systems

This parameter defaults to the value in TCPMaxClients, but will be dynamically changed by the TIBCO iProcess Objects Server as needed.

Lower Bound: 0
Upper Bound: 32,767
Default: Same as TCPMaxClients

SALSessionTimeout

Sets the amount of time, in seconds, that the TIBCO iProcess Objects Server will wait before it will close idle SAL sessions. A value of 0 means SAL sessions will not automatically close.

Lower Bound: 0 (disables timeouts)
Upper Bound: unlimited
Default: 0

SALWaitTimeout

Specifies, in milliseconds, the amount of time the server will wait for a SAL session to become free before returning an error to the client. (Each user has their own SAL session. While a particular user is running a transaction, any other users with the same name will have to wait to get exclusive access to their SAL session.)

If the server times out waiting for the SAL session to return, it returns an ER_ACQUIRE error (-158 - "error acquiring a user's mutex and SAL session") to the client. It also writes the following to the log file:

"*process_msg*: acquire_user (*user*) timed out (-158)"

where *process_msg* is the name of the function where the timeout occurred, and *user* is the name of the user that timed out trying to get a SAL session.

Lower Bound: 0 (disables timeouts)
Upper Bound: unlimited
Default: 60000 (60 seconds)

SALNumPDSessions

Specifies the number of PD (procedure definition) session handles to cache when a SAL session is started, i.e., whenever a user logs on for the first time. This setting should be changed only if instructed by TIBCO Support personnel.

Lower Bound: 0
Upper Bound: 7
Default: 2

SerializeSALLogin

Specifies whether or not the TIBCO iProcess Objects Server should serialize calls to the SAL login function. SAL login calls should be serialized if your UVAPI package is not thread safe. If the UVAPI package is not thread safe, concurrent calls to the SAL login function can cause the server to crash. Note that setting this parameter to 1 will have a negative impact on performance.

Values: 0 (No) or 1 (Yes)
Default: 0 (No)

On UNIX systems, this configuration parameter is not automatically written to your TIBCO iProcess Objects Server configuration file (`swentobjsv.cfg`) when you upgrade from an earlier TIBCO iProcess Objects Server. (It can be found in the "sample" configuration file (`sample.cfg`) that is written to your system when you upgrade.) To make use of this parameter, you must manually add it to your configuration file.

SALRPCSize

Sets the size of the RPC buffer used to communicate with the work queue servers.

Lower Bound: 1,000
Upper Bound: 32,000
Default: 4,096 bytes

SALRPCTimeout

Sets the amount of time, in seconds, that the RPC layer of the SAL will wait for a response to an RPC request before timing out.

Lower Bound: 1
Upper Bound: 32767
Default: 25 seconds

NumFiles (UNIX Only)

This parameter sets the maximum number of files that the UNIX TIBCO iProcess Objects Server can open.

TIBCO iProcess Engine must be started with a sufficient number of available file descriptors per process based upon the number of users. It is highly recommended that the file limit be as high as possible since the TIBCO iProcess Objects Server will immediately exit if there are no more file descriptors available. The following three methods are available to set this number:

- Set the NumFiles configuration parameter to the desired maximum number of files to open. If present, the TIBCO iProcess Objects Server will use this value to set the maximum number of open files. If NumFiles is NOT specified, the TIBCO iProcess Objects Server will use the `TCPMaxClients` parameter as described below to determine the maximum number of open files. (Note - By default, the `NumFiles` parameter is NOT specified.)
- Allow the TIBCO iProcess Objects Server to calculate the maximum number of open files using the value in the `TCPMaxClients` configuration parameter (see TCPMaxClients on page 35). The calculation shown below is used:

  ```
  (2 times value of TCPMaxClients) + 20
  ```

  The TIBCO iProcess Objects Server will automatically use the formula shown above to set the maximum number of files if the `NumFiles` parameter is not specified.
- Depending on the types of transactions being run, the number of file descriptors determined by the formula above may be insufficient on some systems. In these cases, you may want to set the number of files in one of the following ways:
  - Use the following formula as a guideline to determine the number of file descriptors you need:

    ```
    (12 * Number of users) + 100
    ```

    For a default configuration, the default number of users is 1024, which means the upper file limit should be 12388 ((12 * 1024) + 100).
  - Run the command `ulimit -n unlimited` to set the file limit to the operating system design limits; this command will only be successful if TIBCO iProcess Engine is started as the root user or if the system limits allow this.

Lower Bound: 148
Upper Bound: System limit
Default: There is no default. If this parameter is not specified, the `TCPMaxClients` parameter is used to calculate the number of files.

StackSize (UNIX Only)

The number of kilobytes to set the thread stack size, per thread.

The `StackSize` parameter is not included in the configuration file (*$SWDIR*`/seo/data/swentobjsv.cfg`) by default. If you want to set the stack size to a value different from the default, you must manually add this parameter to the configuration file.

When filtering cases, if the filter expression contains a large number of clauses, the SAL sal_xpc_list_filter_cases routine crashes, resulting in the TIBCO iProcess Objects Server crashing (a clause consists of a field being compared to a value, e.g., "SW_CASENUM = 7 AND AMOUNT > 10000" contains two clauses).

If you experience this type of error, increasing the thread stack size may resolve the problem. The default stack size is 2MB per thread. To increase this value, you must manually add the `StackSize` parameter to your configuration file. Note that increasing the stack size will increase the overall memory size of TIBCO iProcess Objects Server.

Lower Bound: `100 (Although the lower bound of 100K is enforced, this parameter should probably not be set to a value lower than the default.)`

Upper Bound: Limited by system thread stack size limits
Default: 2MB (AIX, HP-UX, Linux, and Solaris)

CacheProcEAIStep

This parameter specifies whether or not to cache the EAI step definition.

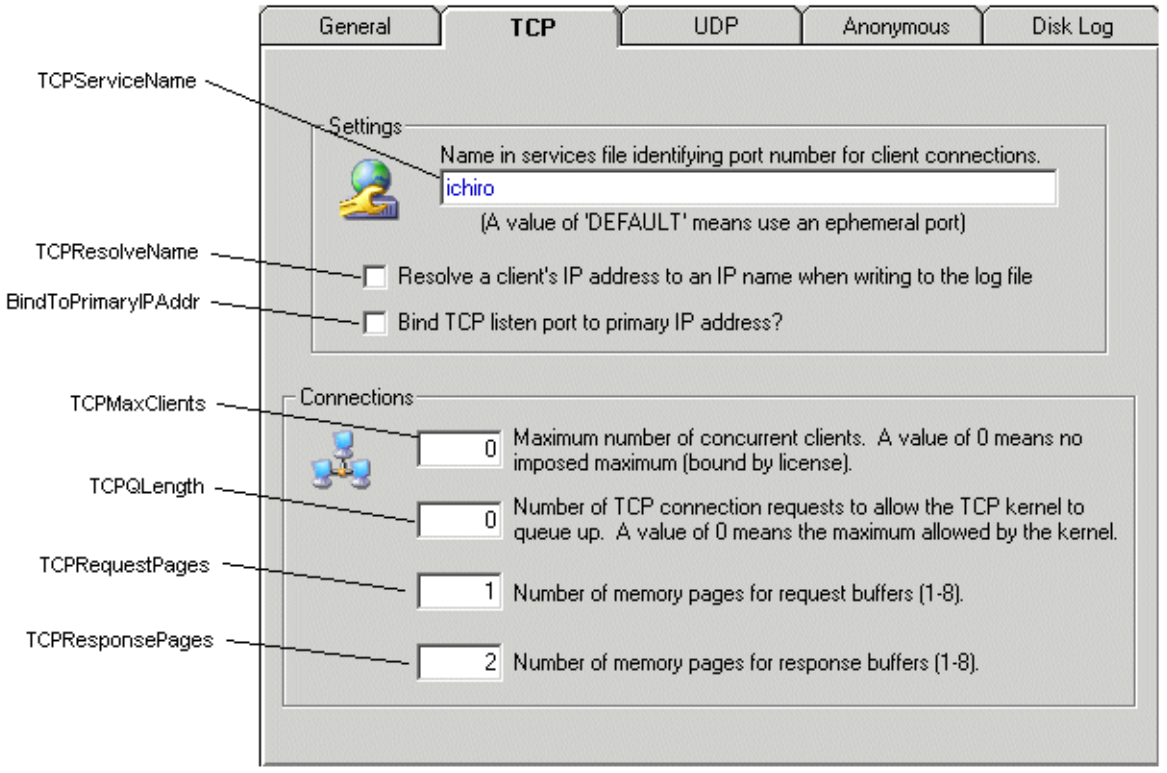
The value of this flag must be one of the following.

Value	Meaning
0	The EAI step definition is not cached. **Note**: Empty information is returned for both TIBCO iProcess Objects and the TIBCO iProcess Server Objects client. The functions that are used to return values for each client are as follows: • For TIBCO iProcess Server Objects client The `getExternalForm` function in the `sProcManager` and `xProcManager` objects. • For TIBCO iProcess Objects client The `getExtForm` function in the `SWStep` object.
1	The EAI step definition is cached.

The EAI step definition is not cached by default when you install or upgrade the TIBCO iProcess Objects Server.

The `CacheProcEAISep` parameter is not included in the `swentobjsv.cfg` configuration file by default when you install or upgrade TIBCO iProcess Objects Server. To cache the EAI step definition, you must manually add this parameter to the configuration file.

TCP Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

TCPServiceName

This identifies the port number on which the TIBCO iProcess Objects Server will listen for client connections. This can be specified in the following ways:

- Specify a value of “DEFAULT”. This means use a dynamic port, which causes the operating system to assign the port number when the TIBCO iProcess Objects Server starts. This designation is used if you are issuing a UDP broadcast to determine the available TIBCO iProcess Objects Servers, or you are sending a directed UDP message to a specific TIBCO iProcess Objects Server.

See *TIBCO iProcess Objects Programmer's Guide* or *TIBCO iProcess Server Objects Programmer's Guide* for information about issuing UDP broadcasts or sending directed UDP messages.

- Specify a value other than "DEFAULT". This means use a static port, which causes the TCP port number to be fixed for the TIBCO iProcess Objects Server you are configuring. This is used if you are manually creating the node object that represents the TIBCO iProcess Objects Server, which requires that you know the TCP port the TIBCO iProcess Objects Server is using. You must also configure the TIBCO iProcess Objects Server to use a static TCP port if you are using TIBCO iProcess Objects Director to choose a TIBCO iProcess Objects Server for you. (For information about manually creating a node object that represents the desired TIBCO iProcess Objects Server, see *TIBCO iProcess Objects Programmer's Guide* or *TIBCO iProcess Server Objects Programmer's Guide.* For information about using TIBCO iProcess Objects Director, see *TIBCO iProcess Director Administrator's Guide*.)

  To configure the TIBCO iProcess Objects Server to use a static TCP port, either specify the desired TCP port number in the `TCPServiceName` parameter, or specify a "service name" that will map to the TCP port number. If using a service name, you must add the service name to the `%SystemRoot%\System32\Drivers\Etc\Services` file (Windows) or `/etc/services` file (UNIX) that maps the service name to the TCP port on which you want the TIBCO iProcess Objects Server to listen for client connections. (If you specify a service name in this parameter, and that service name does not exist in the `services` file, the TIBCO iProcess Objects Server will not start.)

Multiple Instances of the TIBCO iProcess Objects Server (on UNIX Systems)

The following is the process a UNIX TIBCO iProcess Objects Server goes through to establish a TCP port when it starts up when you are running multiple instances of the TIBCO iProcess Objects Server:

1. The TIBCO iProcess Objects Server looks to see if an instance-specific TCPServiceName parameter is defined for the instance of the TIBCO iProcess Objects Server that is starting. For example, if instance 2 of the TIBCO iProcess Objects Server is starting, it looks for TCPServiceName02 (the instance number appended to the parameter name is always two digits and zero padded). If the instance-specific parameter exists, it uses the TCP port specified.
2. If an instance-specific TCPServiceName parameter has *not* been defined, the TIBCO iProcess Objects Server will look to see if the "generic" TCPServiceName parameter (without the instance number appended to the TCPServiceName parameter) has been defined. If it has been defined, it determines the TCP port number assigned to that parameter. It then considers

that the "base" TCP port number. Using the base TCP port number, it adds the TIBCO iProcess Objects Server's instance number (minus 1; because the base number is used by instance 1) to the base TCP port number to determine the TCP port for that instance of the TIBCO iProcess Objects Server (e.g., if the base TCP port number is 10000, instance 3 of the TIBCO iProcess Objects Server will use TCP port 10002).

3. If neither the instance-specific nor the "generic" TCPServiceName parameter has been defined, it defaults to dynamic, causing the operating system to assign the port number when the TIBCO iProcess Objects Server is started.

Multiple Instances of the TIBCO iProcess Objects Server (on Windows Systems)

The following is the process a Windows TIBCO iProcess Objects Server goes through to establish a TCP port when it starts up when you are running multiple instances of the TIBCO iProcess Objects Server:

1. The TIBCO iProcess Objects Server looks to see if you've assigned a TCP port to the specific instance of the TIBCO iProcess Objects Server that is starting (i.e., you've selected the specific instance in the TIBCO iProcess Objects Server Configuration Utility and specified a TCP port different than that specified for "<*ServerName*>(default)" in the utility). If an instance-specific assignment exists, it uses the TCP port specified.
2. If an instance-specific assignment does not exist (i.e., the TCP port assignment for the specific instance of the TIBCO iProcess Objects Server is the same as "<*ServerName*>(default)" in the TIBCO iProcess Objects Server Configuration Utility). This causes the TCP port assignment for "<*ServerName*>(default)" to become the "base" TCP port. Using the base TCP port number, it adds the TIBCO iProcess Objects Server's instance number (minus 1; because the base number is used by instance 1) to the base TCP port number to determine the TCP port for that instance of the TIBCO iProcess Objects Server (e.g., if the base TCP port number is 10000, instance 3 of the TIBCO iProcess Objects Server will use TCP port 10002).

Length: 40 characters maximum

Default: DEFAULT

TCPResolveName

This flag specifies if the server should employ TCP name resolution (DNS, host file, YP, etc.). Setting this value to 1 causes the TIBCO iProcess Objects Server to do name resolution of all client connection requests. If set to 0, the TIBCO iProcess Objects Server uses the client IP address and TCP connection port to identify clients.

This parameter is used for debugging purposes. When set to 1, the machine name of the client appears in the log file, instead of the client's IP address.

Setting this parameter to 1 may have a negative impact on performance. It is recommended that you normally leave this parameter set to 0 unless debugging.

Values: 0 (no) or 1 (yes)
Default: 0

BindToPrimaryIPAddr (Windows Only)

Specifies whether or not the TIBCO iProcess Objects Server should bind the TCP and UDP ports to the primary IP address of the machine. Checking this box (value of 1) causes the ports to be bound ONLY to the primary IP address of the machine (this is the behavior of previous releases of the TIBCO iProcess Objects Server). Not checking this box (value of 0) causes the TCP and UDP ports to be bound to ADDR_ANY. This allows TCP (client) connections to the TIBCO iProcess Objects Server over any of the network interfaces on the machine, rather than just the primary interface.

If this parameter is set to 1, the SWClientInfo.Name property returns the following information:

```
<Client IP Address>:<TCP Port #>
```

For example: 10.12.84.135:9288

If this parameter is set to 0 (the default), the SWClientInfo.Name property returns the following information:

```
<Client IP Address>:<TCP Port #> - <Server IP Address>:<TCP Port #>
```

For example: 10.12.84.135:9288 - 10.12.84.36:1520

Values: 0 or 1
Default: 0

TCPMaxClients

The maximum number of concurrent client connections.

Windows Systems

A value of 0 means no imposed maximum (still bound by the license, however).

Lower Bound: 0
Upper Bound: none
Default: 0

UNIX Systems

Increasing this value will require that the system supports an increased number of open files as well as raising the amount of memory that will be used (approx. 16K of additional memory will be used for each 1,000 that this parameter is increased). Under normal circumstances, this value should be set to slightly above either your licensed number of users or the maximum number of clients that you expect to connect to the TIBCO iProcess Objects Server.

If a client connects to the TIBCO iProcess Objects Server and that connection causes the total number of connections in the TIBCO iProcess Objects Server to exceed TCPMaxClients, the connection will be aborted and the client will get a "TIBCO iProcess Objects Server disconnected" error on the login. (The TIBCO iProcess Objects Server error log will be written to if this occurs.)

This parameter may also be used by TIBCO iProcess Engine startup script to determine the maximum number of open files — see the `NumFiles` parameter on NumFiles (UNIX Only) on page 29 for more information.

Lower Bound: 64
Upper Bound: 32,767
Default: 1,024

TCPQLength

Number of TCP connection requests to allow the TCP kernel to queue up. A value of 0 means the maximum allowed by the kernel. Under normal operating conditions, it is recommended that you do not change the value of this parameter.

Lower Bound: 0
Upper Bound: none
Default: 0

TCPRequestPages

Number of pages for request buffers. Each page is 2,048 bytes. It is not normally necessary to change the value of this parameter from its default value.

Values: 1 to 8
Default: 1 (2,048 bytes)

TCPResponsePages

Windows Systems

Number of pages for response buffers. Each page is 2,048 bytes. It is not normally necessary to change the value of this parameter from its default value. The value of this parameter would only need to be increased in an installation where the TIBCO iProcess Objects Server responses to a client request are very large.

Values: 1 to 8
Default: 2 (4096 bytes)

UNIX Systems

The size of the response buffer defined in number of pages, where each page is 2,048 bytes.

This value defines the maximum amount of data that the server will send to the client in one transmission; the server then waits for the client to acknowledge receipt of that transmission before sending the next buffer.

Depending on your network configuration, it may be beneficial to set this value to any number up to the maximum (8 pages, which is 16,384 bytes), particularly if large amounts of data are being sent back to the client.

Under normal circumstances, the buffer size should be set to your network buffer limit or the TIBCO iProcess Objects Server limit (16,384 bytes), whichever is lower.

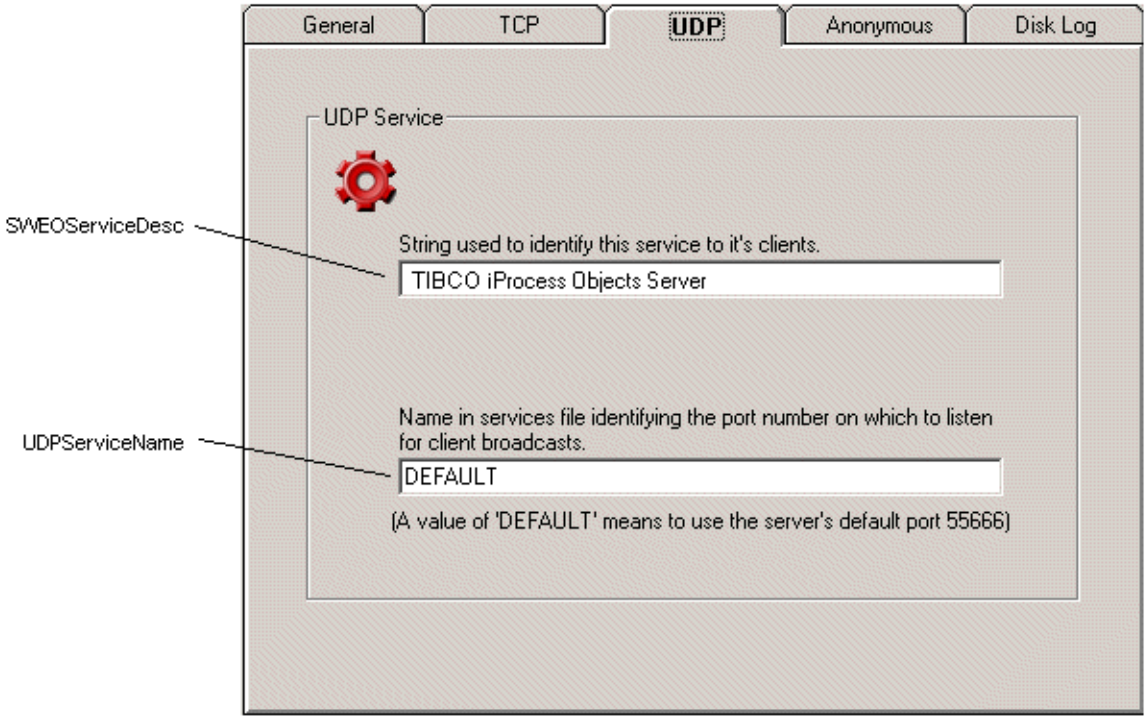
The maximum network buffer size can be displayed with the `ndd` command, as follows:

```
/usr/sbin/ndd /dev/tcp tcp_max_buf
```

The default size for this is 1,048,576 bytes, which allows the configuration parameter to be set to the maximum 8 pages.

Values: 1 to 8
Default: 2 (4096 bytes)

UDP Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

SWEOServiceDesc

String used to describe the TIBCO iProcess Objects Server to clients. The client has access to this value in the SWNodeInfo.SWEOSrvDesc property (TIBCO iProcess Objects), `vNode.getSEOSrvDesc` method (TIBCO iProcess Server Objects (Java)), or vNode.SEOSrvDesc property (TIBCO iProcess Server Objects (.NET)). The value of this parameter can be any string that is useful to the client for identifying the server.

Length: 64 chars max

Default: TIBCO iProcess Objects Server

UDPServiceName

This parameter identifies the port number on which the TIBCO iProcess Objects Server will listen for UDP messages/broadcasts. This can be specified in one of the following ways:

- Specify a value of "DEFAULT" (the default). This causes the TIBCO iProcess Objects Server to listen for UDP messages/broadcasts on port 55666.
- Specify the port number on which you want the TIBCO iProcess Objects Server to listen for UDP messages/broadcasts.
- Specify a "service name" that will map to the UDP port number. This requires that you add the service name to the `%SystemRoot%\System32\Drivers\Etc\Services` file (Windows) or **/etc/services** file (UNIX) that maps the service name to the UDP port on which you want the TIBCO iProcess Objects Server to listen for UDP messages/broadcasts. (If you specify a service name in this parameter, and that service name does not exist in the `services` file, the TIBCO iProcess Objects Server will not start.)
- Specify "None". This causes the TIBCO iProcess Objects Server to not open a UDP port, i.e., it will not respond to UDP messages/broadcasts.

For more information about issuing UDP broadcasts or sending directed UDP messages, see *TIBCO iProcess Objects Programmer's Guide* or *TIBCO iProcess Server Objects Programmer's Guide* .

Multiple Instances of the TIBCO iProcess Objects Server (on UNIX Systems)

The following is the process a UNIX TIBCO iProcess Objects Server goes through to establish a UDP port when it starts up when you are running multiple instances of the TIBCO iProcess Objects Server:

1. The TIBCO iProcess Objects Server looks to see if an instance-specific UDPServiceName parameter is defined for the instance of the TIBCO iProcess Objects Server that is starting. For example, if instance 2 of the TIBCO iProcess Objects Server is starting, it looks for UDPServiceName02 (the instance number appended to the parameter name is always two digits and zero padded). If the instance-specific parameter exists, it uses the UDP port specified (or if "None" is specified, it will not open a UDP port).
2. If an instance-specific UDPServiceName parameter has *not* been defined, the TIBCO iProcess Objects Server will look to see if the "generic" UDPServiceName parameter (without the instance number appended to the UDPServiceName parameter) has been defined. If it has been defined, it determines the UDP port number assigned to that parameter. It then considers that the "base" UDP port number. Using the base UDP port number, it adds the TIBCO iProcess Objects Server's instance number (minus 1; because the

base number is used by instance 1) to the base UDP port number to determine the UDP port for that instance of the TIBCO iProcess Objects Server (e.g., if the base UDP port number is 55670, instance 3 of the TIBCO iProcess Objects Server will use UDP port 55672).

3. If neither the instance-specific nor the "generic" UDPServiceName parameter has been defined, the first instance of the TIBCO iProcess Objects Server is assigned port 55666, instance 2 is assigned 55667, and so on.

Multiple Instances of the TIBCO iProcess Objects Server (on Windows Systems)

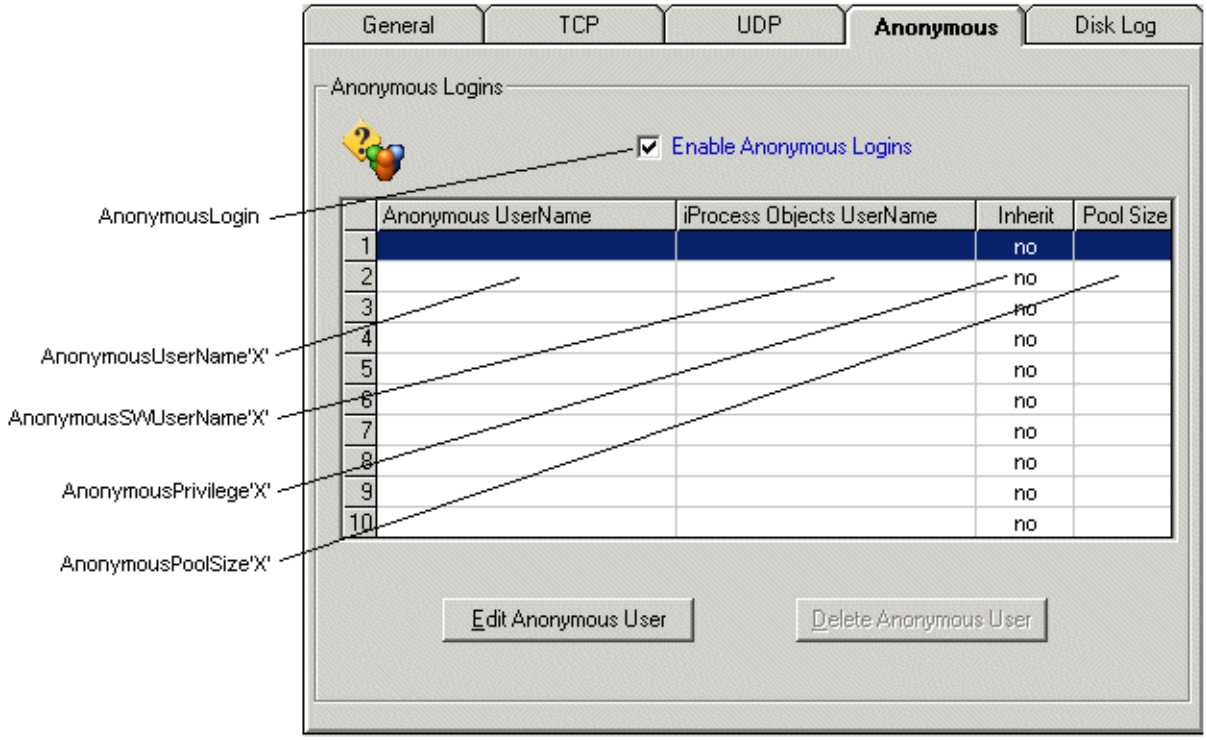
The following is the process a Windows TIBCO iProcess Objects Server goes through to establish a UDP port when it starts up when you are running multiple instances of the TIBCO iProcess Objects Server:

1. The TIBCO iProcess Objects Server looks to see if you've assigned a UDP port to the specific instance of the TIBCO iProcess Objects Server that is starting (i.e., you've selected the specific instance in the TIBCO iProcess Objects Server Configuration Utility and specified a UDP port different than that specified for "<*ServerName*>(default)" in the utility). If an instance-specific assignment exists, it uses the UDP port specified (or if "None" is specified, it will not open a UDP port).
2. If an instance-specific assignment does not exist (i.e., the UDP port assignment for the specific instance of the TIBCO iProcess Objects Server is the same as "<*ServerName*>(default)" in the TIBCO iProcess Objects Server Configuration Utility). This causes the UDP port assignment for "<*ServerName*>(default)" to become the "base" UDP port. Using the base UDP port number, it adds the TIBCO iProcess Objects Server's instance number (minus 1; because the base number is used by instance 1) to the base UDP port number to determine the UDP port for that instance of the TIBCO iProcess Objects Server (e.g., if the base UDP port number is 55670, instance 3 of the TIBCO iProcess Objects Server will use UDP port 55672).

Length: 40 chars max

Default: DEFAULT

Anonymous Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

AnonymousLogin

Flag (0 for no, 1 for yes) indicating if anonymous logins are enabled. If set to 0, all other Anonymous parameters are ignored.

Values: 0 or 1
Default: 0

AnonymousUserName'X'

Name of user to use for logging into the anonymous pool. This is the username that is specified in the `Login` method. This name does not correspond to any actual user name. It can be any name that you want that is not the name of an iProcess user.

'X' is replaced by an integer from 1 - 10. You can configure 10 separate anonymous login pools.

Length: 24 chars max

Default: *UNDEFINED*

AnonymousSWUserName'X'

Name of an iProcess user that AnonymousUserName'X' will be mapped to if anonymous logins are enabled. For example, if AnonymousUserName1 is anonuser and AnonymousSWUserName1 is appadmin, logging into the server with username anonuser will run the transactions as iProcess user appadmin.

X' is replaced by an integer from 1 - 10. You can configure 10 separate anonymous login pools.

Length: 24 chars max

Default: *UNDEFINED*

AnonymousPrivilege'X'

Flag (0 for no, 1 for yes) specifying if an anonymous user should inherit the MENUNAME attribute value of AnonymousUserSWName'X'. If set to 1, an anonymous access by a client will inherit the MENUNAME attribute value defined for AnonymousSWUserName'X'. If set to 0, an anonymous access by a client will be forced to a MENUNAME attribute value of USER.

'X' is replaced by an integer from 1 - 10. You can configure 10 separate anonymous login pools.

Values: 0 or 1
Default: 0

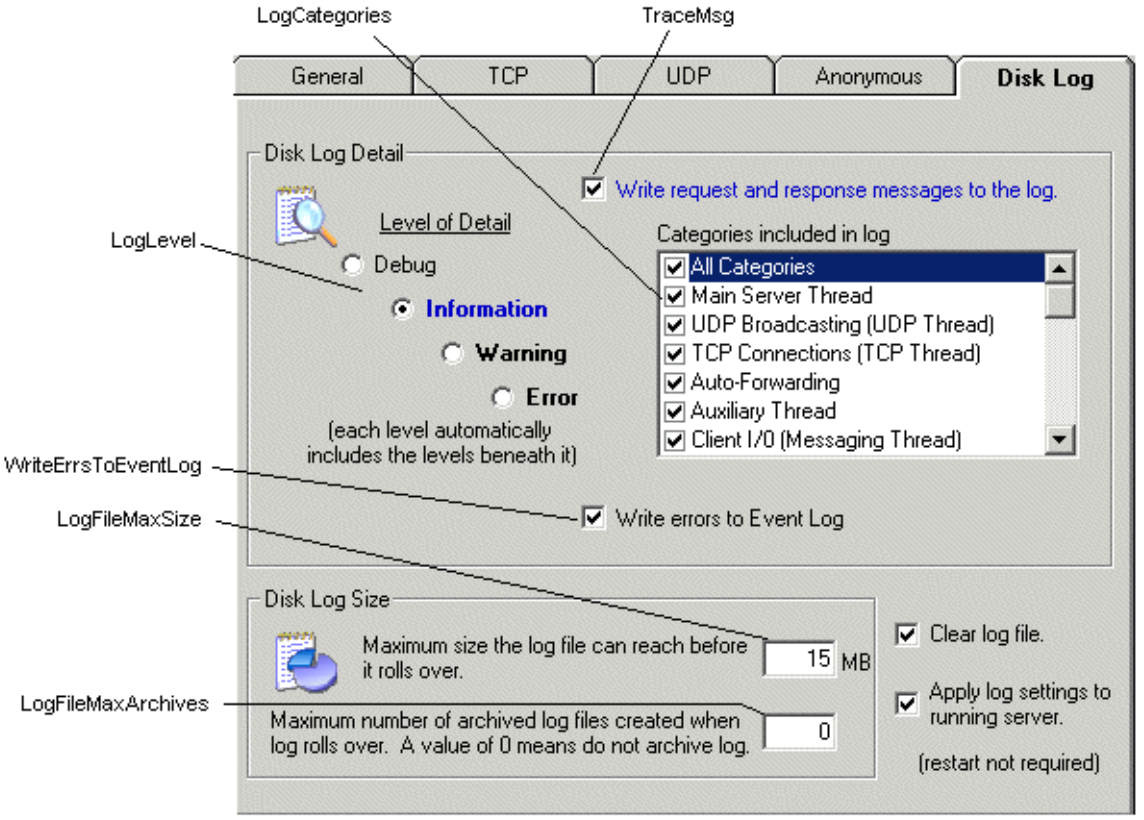
AnonymousPoolSize'X'

Number of SAL sessions to pool for this anonymous login. The server will create this many sessions to be shared among all anonymous clients. The value of this parameter should be set to the average number of concurrent anonymous logins to this pool.

'X' is replaced by an integer from 1 - 10. You can configure 10 separate anonymous login pools.

Lower Bound: 1
Upper Bound: 1,024
Default: 5

Disk Log Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

TraceMsg

Flag (0 for no, 1 for yes) specifying if client request and response messages should be traced to the log file. Note that even if this parameter is set to “0” (No), messages will still be traced if LogLevel (see LogLevel on page 47) is set to “4” (Debug). In addition, only transactions that are specified through the LogCategories (see below) will be traced.

Unless directed by TIBCO Support, it is highly recommended that TraceMsg be set to "0" (No) since setting this parameter to "1" could cause the log file to quickly fill, causing the possibility of critical error messages being overlooked or lost when the log reaches its maximum size and is truncated. Turning this parameter on will degrade the performance and response time of the TIBCO iProcess Objects Server.

Values: 0 or 1
Default: 0

LogCategories

Specifies which categories of information to write to the log.

Do not change this parameter unless you are advised to do so by TIBCO Support.

Values: Bit settings from 0x00000001 to 0xFFFFFFFF
Default: 0xFFFFFFFF (LOGCAT_ALL)

The following log categories are available (either alone or by combining the indicated hex values):

Category	Value	Description
LOGCAT_ALL	0xFFFFFFFF	All
LOGCAT_MAINTHD	0x00000001	Main Thread
LOGCAT_UDPTHD	0x00000002	UDP Thread
LOGCAT_TCPTHD	0x00000004	TCP Message Receive Thread
LOGCAT_AUTOFWDTHD	0x00000008	Auto Forward Thread
LOGCAT_AUXTHD	0x00000010	Auxiliary Thread
LOGCAT_MSGTHD	0x00000020	Message Processing Threads
LOGCAT_LOGIN	0x00000040	Login/logouts
LOGCAT_PASSWORD	0x00000080	Set Password
LOGCAT_USER	0x00000100	Add/Remove/Change/List Users

Category	Value	Description
LOGCAT_ATTRIBUTE	0x00000200	Add/Remove/Change/List Attributes, Change User
LOGCAT_ROLE	0x00000400	Add/Remove/List Role
LOGCAT_GROUP	0x00000800	Add/Remove Group, Add/Remove User From/To Groups
LOGCAT_PROC	0x00001000	Purge Case, List/Query Procedures
LOGCAT_PROCQUERY	0x00002000	Query Procedures
LOGCAT_PROCDEF	0x00004000	Extended list procs, extended marking info, aggregate case info
LOGCAT_QACCESS	0x00008000	Add/Remove/List View Queue and Autoforward Queues
LOGCAT_QQUERY	0x00010000	Queue query, get queue item, destroy view
LOGCAT_CASE	0x00020000	Close/Purge Case
LOGCAT_NODE	0x00040000	List node
LOGCAT_EVENT	0x00080000	Trigger event
LOGCAT_WORKITEM	0x00100000	Get/Keep/Release/Forward work item
LOGCAT_FORWARDING	0x00200000	Forward Item, Add/Remove/List Auto-Forward List
LOGCAT_INSTRUMENTATION	0x00400000	Get Instrumentation (GI) transaction
LOGCAT_MEMOATTACHMENT	0x00800000	(not currently used)
LOGCAT_SWLSTTBL	0x01000000	(not currently used)
LOGCAT_LOG	0x02000000	Set log level/categories
LOGCAT_TIMING	0x04000000	SAL SDK timing (for internal use)
LOGCAT_DIRECTOR	0x08000000	TIBCO iProcess Objects Director operations

LogLevel

Level of detail to write to log file. Each level includes the levels numerically smaller (e.g., Informational includes Warnings and Errors). The level is specified using the corresponding number. Note that if the log level is set to 4 (Debug), messages are automatically traced to the log (even if the `TraceMsg` parameter is set to 0 — see `TraceMsg` above).

Unless directed by TIBCO Support, LogLevel should be specified as either "1" or "2". Setting the LogLevel to "4" (Debug) will cause an extremely large number of messages to be written to the log file. This will cause the performance and response time of the TIBCO iProcess Objects Server to be degraded. It also causes the possibility that critical error messages will be lost if the log file fills up and rolls over.

Errors: 1
Warnings: 2
Informational: 3
Debug: 4
Default: 2

WriteErrsToEventLog (Windows Only)

Flag that specifies whether or not TIBCO iProcess Objects Server errors should be written to the Windows Event Log.

The Write errors to Event Log check box is used to set this configuration parameter. Checking this box (the default) causes TIBCO iProcess Objects Server errors to be written to the Event Log. Unchecking this box disables the writing of *most* TIBCO iProcess Objects Server errors to the Event Log — note that the following errors are still written to the Event Log regardless of the setting of this parameter: errors opening, writing to, or archiving log files; errors reading configuration values in the Registry; errors starting the NT service; errors because of invalid configuration parameters.

LogFileMaxSize

Maximum size in MB of the log file before it is truncated (rolls over). Note that the log file will roll over when the maximum size is reached regardless of the LogLevel setting.

Only the following log file is rolled over:

- On UNIX

  The `swentobjsv`*XX*`.log` file, which is located in the $*SWDIR*`/logs`.

- On Microsoft Windows

  The `SWEntObjSv`*XX*`.log` file, which is located in the *SWDIR*`\logs`.

The following audit log is never rolled over by TIBCO iProcess Objects Server:

- On UNIX

  The `swentobjua`*XX*`_`*timestamp*`.log` file, which is located in the $*SWDIR*`/logs`.

- On Microsoft Windows

  The `SWEntObjUa`*XX*`_`*timestamp*`.log` file, which is located in the *SWDIR*`\logs`.

It is the responsibility of the administrator at your site to back up and/or remove this file. See Log File Names on page 10 for more information.

Lower Bound: 1
Upper Bound: none
Default: 15 MB

LogFileMaxArchives

This specifies the number of archive log files that will be saved when the log rolls over as a result of reaching the maximum size limit (LogFileMaxSize). This can be useful if you want to collect many MB of debug log, but don't want to deal with a very large log file by setting LogFileMaxSize to a large value.

Windows Systems

The "non-archived" TIBCO iProcess Objects Server log file is named `SWEntObjSv`*XX*`.log` (see Name and Location of the TIBCO iProcess Objects Server Log on page 75). If the LogFileMaxArchives parameter is set to a value greater than 0, and the "non-archived" TIBCO iProcess Objects Server log file reaches the maximum size set by the LogFileMaxSize parameter, the log rolls over and "archived" log files are saved. The archived log files are named `SWEntObjSv`*XX*`_`*timestamp*`_archive_`*xxx*`.log`, where xxx is a counter that is

incremented each time the log rolls over (starting at 1). For example, if this parameter is set to 2, the first time the log rolls over, it will be saved as `SWEntObjSv`*XX*`_timestamp_archive_1.log`. The next time it rolls over, it will be saved as `SWEntObjSv`*XX*`_timestamp_archive_2.log`. If it rolls over again, it will be saved as `SWEntObjSv`*XX*`_timestamp_archive_3.log`, but `SWEntObjSv`*XX*`_timestamp_archive_1.log` will be deleted because it is only saving two archives.

UNIX Systems

The "non-archived" TIBCO iProcess Objects Server log file is named `swentobjsv`*XX*`.log` (see Name and Location of the TIBCO iProcess Objects Server Log on page 75). If the LogFileMaxArchives parameter is set to a value greater than 0, and the "non-archived" TIBCO iProcess Objects Server log file reaches the maximum size set by the LogFileMaxSize parameter, the log rolls over and "archived" log files are saved. The archived log files are named `swentobjsv`*XX*`_timestamp_archive_`*xxx*`.log`, where *xxx* is a counter that is incremented each time the log rolls over (starting at 1). For example, if this parameter is set to 2, the first time the log rolls over, it is saved as `swentobjsv`*XX*`_timestamp_archive_1.log`. The next time it rolls over, it is saved as `swentobjsv`*XX*`_timestamp_archive_2.log`. If it rolls over again, it is saved as `swentobjsv`*XX*`_timestamp_archive_3.log`, but `swentobjsv`*XX*`_timestamp_archive_1.log` will be deleted because it is only saving two archives.

Whenever the TIBCO iProcess Objects Server is restarted, the archive log file name starts back at `swentobjsvb`*XX*`_timestamp_archive_1.log`. Therefore, if the system is set up for three archive log files, and the log rolls over eight times, the archive log files will be ..._6.log, ..._7.log, and ..._8.log. However, if the server is now shut down and restarted, the next three archive log files will be ..._1.log, ..._2.log, and ..._3.log (the ..._6.log, ..._7.log, and ..._8.log files will still exist).

Lower Bound: 0 (do not archive log files)
Upper Bound: none
Default: 0

UseSysLog (UNIX Only)

Flag (0 for no, 1 for yes) indicating whether all messages other than debug messages that are written to the TIBCO iProcess Objects Server message and audit logs are also written to the UNIX system log.

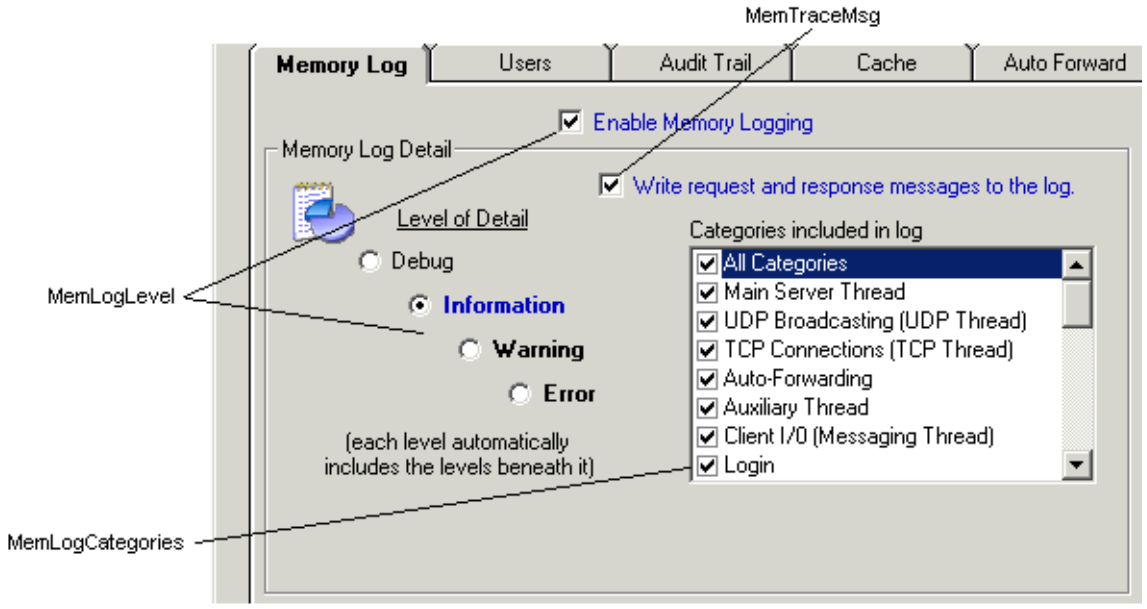
When the TIBCO iProcess Objects Server sends a message to the UNIX system log via the "syslogd" daemon, the facility or subsystem on the message is set to "local0". The syslogd priority levels are set as follows:

- info - Informational (and general messages)
- err - Errors
- warn - Warning messages
- notice - All audit functions

For more information about the UNIX System Log, see UNIX System Log on page 95

Values: 0 (No) or 1 (Yes)
Default: 0 (No)

Memory Log Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

MemLogLevel

This parameter is used to set the level of in-memory logging.

You can also use this parameter to enable or disable in-memory logging. In Windows, disable in-memory logging by unchecking the Enable Memory Logging check box. In UNIX, disable in-memory logging by setting the MemLogLevel parameter to 0. Note, however, these methods of enabling/disabling do not totally turn off in-memory logging. If you disable in-memory logging, this prevents server log messages (i.e., messages that would normally appear in the iProcess Objects Server log file) from being logged in the in-memory log. However, there will still be an in-memory log created, and if the

process dies or crashes, a memory dump will still be created. The in-memory log will contain a minimum amount of information, such as initialization parameters and SDK messages. If the SDK debug is set, any SDK log messages will be stored in the in-memory log.

In-memory logging allows you to specify that log messages be written to memory during normal server operation, rather than to disk; they are written to disk when the server crashes, or manually using the `swsvrmgr DUMPLOG` command.

In-memory logging allows for easier and faster detection and resolution of problems in the TIBCO iProcess Objects Server without degrading the performance as much as using the "standard" disk file logging. However, using in-memory logging will still impact performance to some degree and it should not be used unless TIBCO Support personnel recommend it. The standard disk file logging is still available (note that the equivalent standard disk file logging configuration parameter is LogLevel — see LogLevel on page 47).

For more information about using in-memory logging, see In-Memory Log File on page 73.

The available log levels are shown below (note that each level includes the numerically smaller level (e.g., level 2 also includes level 1 messages):

Disable In-Memory Logging: 0

Errors: 1 (least amount of information)
Warnings: 2
Informational: 3
Debug: 4 (greatest amount of information)
Default: 0 (off)

MemLogCategories

The categories of messages to include in the in-memory TIBCO iProcess Objects Server log. See the table for the `LogCategories` parameter (the equivalent "file" logging parameter) on LogCategories on page 45 for a list of the available log categories.

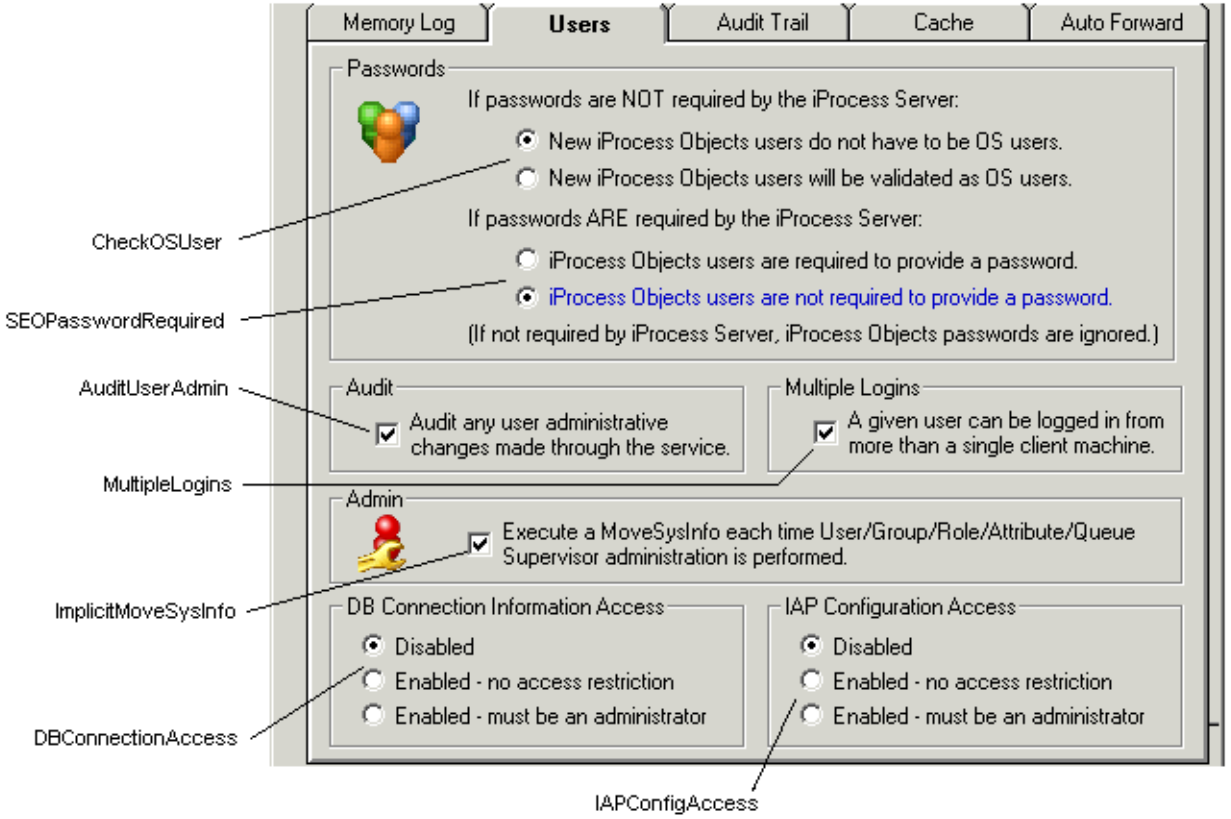
Values: Bit settings from 0x00000000 to 0xFFFFFFFF
Default: 0xFFFFFFFF (all categories)

MemTraceMsg

Flag specifying if request and response messages should be traced and written to the in-memory TIBCO iProcess Objects Server log. Note that even if this parameter is set to “0” (No), messages will still be traced if MemLogLevel is set to “4” (Debug).

Values: 0 (No) or 1 (Yes)
Default: 0 (No)

User Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration param7eter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

CheckOSUser

Flag indicating if all new TIBCO users must exist as O/S users.

This parameter is applicable only if TIBCO iProcess Engine does not require a password. This is specified in the `$`*SWDIR*`/etc/staffpms` (UNIX) or *SWDIR*`\etc\staffpms` (Windows) file (for information about configuring TIBCO iProcess Engine to check for passwords, see *TIBCO iProcess Objects Programmer's Guide* or *TIBCO iProcess Server Objects Programmer's Guide*).

TIBCO iProcess Engine contains a `DISABLE_USER_CHECK` process attribute that is checked before the `CheckOSUser` parameter is checked. If the `DISABLE_USER_CHECK` attribute is set to `1`, the new user does not have to be an existing O/S user, regardless of how the `CheckOSUser` parameter was set.

Values: 0 (No) or 1 (Yes)
Default: 0 (No)

SEOPasswordRequired

Flag indicating whether user passwords will be checked by TIBCO iProcess Objects Server.

This parameter is applicable only if TIBCO iProcess Engine requires a password. This is specified in the `$`*SWDIR*`/etc/staffpms` (UNIX) or *SWDIR*`\etc\staffpms` (Windows) file (for information about configuring TIBCO iProcess Engine to check for passwords, see *TIBCO iProcess Objects Programmer's Guide* or *TIBCO iProcess Server Objects Programmer's Guide*).

If TIBCO iProcess Engine is set to not require passwords, this parameter is forced to 0 (No) regardless of the setting in this configuration file.

Values: 0 (No) or 1 (Yes)
Default: 1 (Yes)

AuditUserAdmin

Flag specifying if user administrative changes should be written to the audit log. Audit messages are written to the `$`*SWDIR*`/logs/swentobjua`*XX*`_`*timestamp*`.log` (UNIX) or *SWDIR*`\logs\SWEntObjUa`*XX*`_`*timestamp*`.log` (Windows) file (for information about the audit log, see *TIBCO iProcess Objects Programmer's Guide* or *TIBCO iProcess Server Objects Programmer's Guide*).

Note that the audit log file is never rolled over by TIBCO iProcess Objects Server; it is the responsibility of administrator at your site to back up and remove this file.

Values: 0 (No) or 1 (Yes)
Default: 1 (Yes)

MultipleLogins

Flag specifying whether or not to allow simultaneous connections to the TIBCO iProcess Objects Server from the same user name.

If you are using a web-based client (for example, TIBCO iProcess Workspace (Browser)), this flag must be set to 1 (enable multiple logins). Disabling multiple logins only makes sense in thick-client applications.

Values: 0 (No) or 1 (Yes)
Default: 1 (Yes)

ImplicitMoveSysInfo

This configuration parameter allows you to control when the `MOVESYSINFO` function is performed.

If this parameter is set to 1 (the check box in the Windows *TIBCO iProcess Objects Server Configuration Utility* is checked), the TIBCO iProcess Objects Server will implicitly call `MOVESYSINFO` whenever an administrative function is performed, i.e., any function that affects a user, group, role, attribute, or queue supervisor definition. Note that this can tie up the background and WIS/WQS processes for long periods of time if there are lots of users. This is the default.

If this parameter is set to 0, the TIBCO iProcess Objects Server will *not* call the `MOVESYSINFO` function whenever an administrative function is performed. This allows the client application to use the `MoveSysInfo` method to explicitly call the `MOVESYSINFO` function. This allows the client application to control when the `MOVESYSINFO` function will occur. For information about the `MoveSysInfo` method, see TIBCO iProcess Objects or TIBCO iProcess Server Objects on-line help.

Values: 0 (No) or 1 (Yes)
Default: 1 (Yes)

DBConnectionAccess

This configuration parameter allows you to control user access to the database configuration information. This parameter can be set to the following states:

- Disabled - This is the default setting. The database configuration information is not accessible by any user.
- Enabled - no access restriction - The database configuration information is accessible by all users.
- Enabled - must be an administrator - The database configuration information is accessible only by system administrators (users whose MENUNAME attribute is ADMIN).

Values: 0 (Disabled), 1 (Enabled for all users), or 2 (Enabled only for system administrators)
Default: 0 (Disabled)

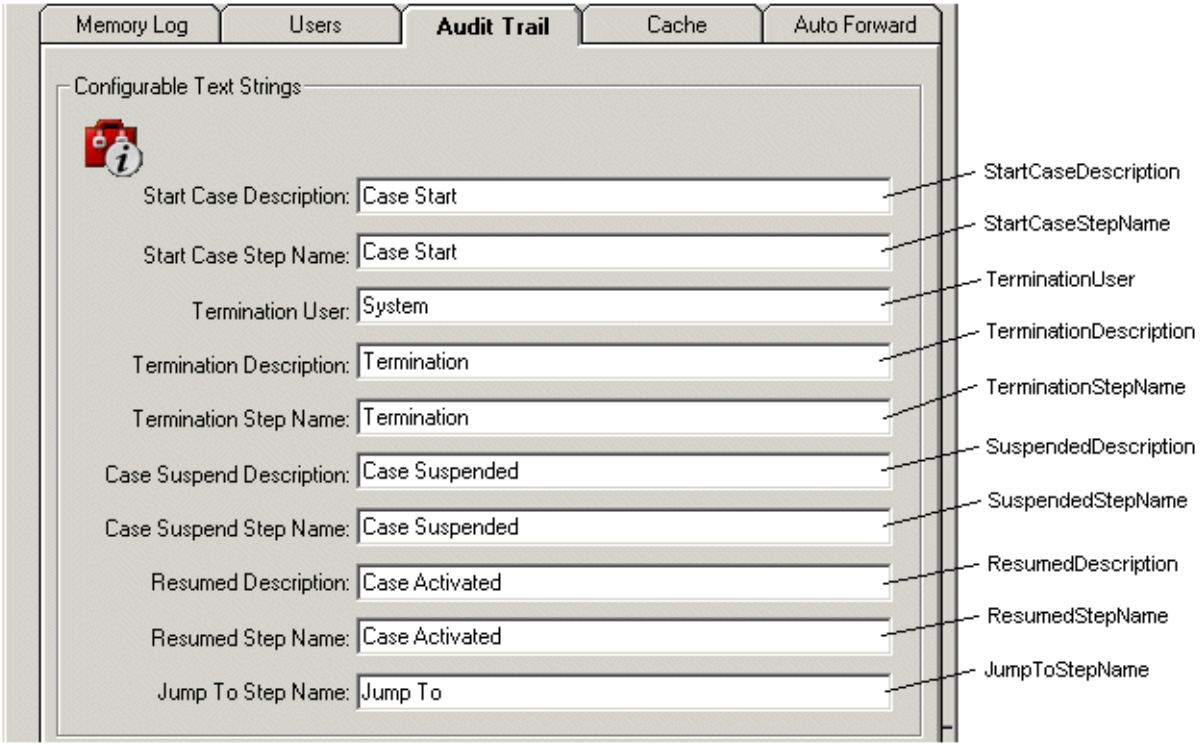
IAPConfigAccess

This configuration parameter allows you to control user access to TIBCO iProcess Engine's activity publication configuration. This parameter can be set to the following states:

- Disabled - This is the default setting. The activity publication configuration is not accessible by any user.
- Enabled - no access restriction - The activity publication configuration is accessible by all users.
- Enabled - must be an administrator - The activity publication configuration is accessible only by system administrators (users whose MENUNAME attribute is ADMIN).

Values: 0 (Disabled), 1 (Enabled for all users), or 2 (Enabled only for system administrators)
Default: 0 (Disabled)

Audit Trail Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

The configuration parameters on this dialog can be used to specify the value that will be returned by various properties/methods on SWAuditStep (TIBCO iProcess Objects) or vAuditStep (TIBCO iProcess Server Objects) when specific actions are processed (start case, close case, etc.). These are actions that typically don't have a corresponding step (resulting in an empty step name and description) or user (resulting in an empty user name). These strings are used in the audit trail for the stated actions. Note that you can specify a blank or empty field value for any of these parameters by assigning an empty string ("") to the parameter.

StartCaseDescription

String that is written to the audit trail (SWAuditStep.Description property in TIBCO iProcess Objects; `vAuditStep.getDescription` method in TIBCO iProcess Server Objects) when a case start event (swStartCase) is processed.

Default: Case Start

StartCaseStepName

String that is written to the audit trail (SWAuditStep.Name property in TIBCO iProcess Objects; `vAuditStep.getName` method in TIBCO iProcess Server Objects) when a case start event (swStartCase) is processed.

Default: Case Start

TerminationUser

String that is written to the audit trail (SWAuditStep.User property in TIBCO iProcess Objects; `vAuditStep.getUser` method in TIBCO iProcess Server Objects) for close case (termination) events where the case is not manually closed (swTermAbnormal and swTermNORMAL).

Default: System

TerminationDescription

String that is written to the audit trail (SWAuditStep.Description property in TIBCO iProcess Objects; `vAuditStep.getDescription` method in TIBCO iProcess Server Objects) for all close case (termination) events (swTermAbnormal, swTermPremature, and swTermNORMAL).

Default: Termination

TerminationStepName

String that is written to the audit trail (SWAuditStep.Name property in TIBCO iProcess Objects; `vAuditStep.getName` method in TIBCO iProcess Server Objects) for all close case (termination) events (swTermAbnormal, swTermPremature, and swTermNORMAL).

Default: Termination

SuspendedDescription

String that is written to the audit trail (SWAuditStep.Description property in TIBCO iProcess Objects; `vAuditStep.getDescription` method in TIBCO iProcess Server Objects) for case suspend events (swSuspendedBy).

Default: Case Suspended

SuspendedStepName

String that is written to the audit trail (SWAuditStep.Name property in TIBCO iProcess Objects; `vAuditStep.getName` method in TIBCO iProcess Server Objects) for case suspend events (swSuspendedBy).

Default: Case Suspended

ResumedDescription

String that is written to the audit trail (SWAuditStep.Description property in TIBCO iProcess Objects; `vAuditStep.getDescription` method in TIBCO iProcess Server Objects) for case resume events (swResumedBy).

Default: Case Activated

ResumedStepName

String that is written to the audit trail (SWAuditStep.Name property in TIBCO iProcess Objects; `vAuditStep.getName` method in TIBCO iProcess Server Objects) for case resume events (swResumedBy).

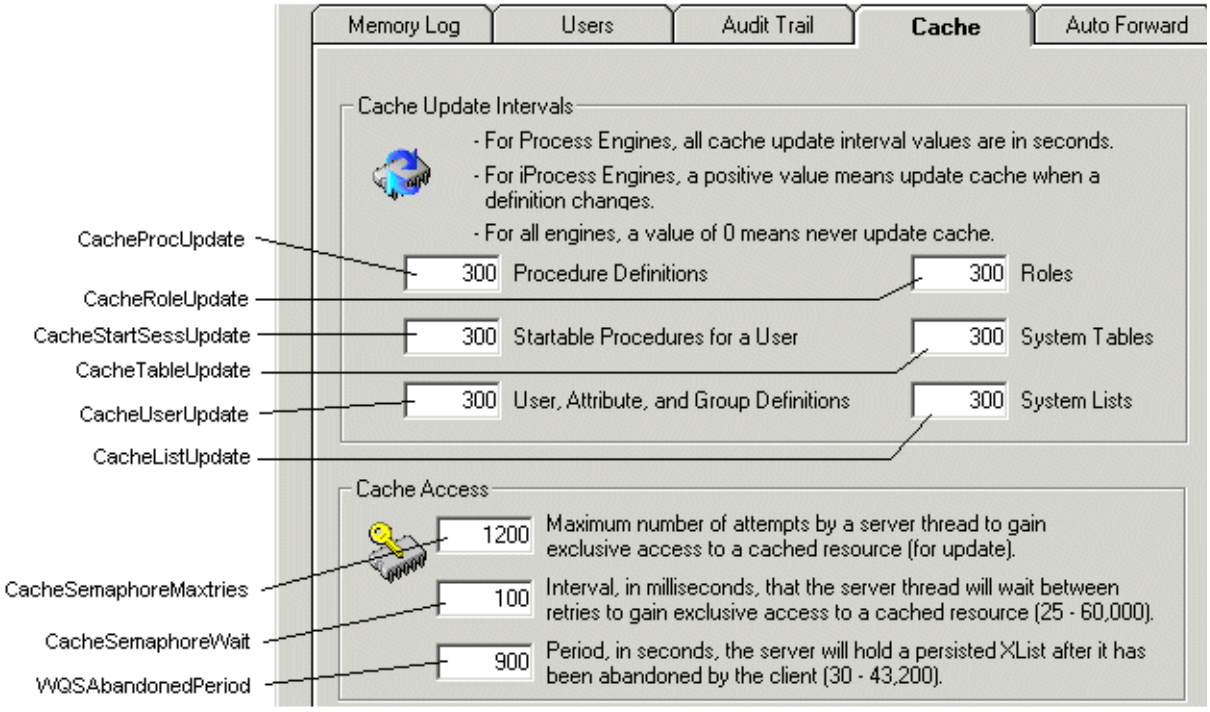
Default: Case Activated

JumpToStepName

String that is written to the audit trail (SWAuditStep.Name property in TIBCO iProcess Objects; `vAuditStep.getName` method in TIBCO iProcess Server Objects) for jump to events (swCaseJumpBy).

Default: Jump To

Cache Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

CacheProcUpdate

This specifies whether or not to update the cache of procedure definitions. Setting this to 0 (zero) causes the cache to never be updated (other than the initial load on startup). Setting this to a positive number causes the cache to be updated in the background immediately after a change to the procedure definitions.

If you don't change procedures while the TIBCO iProcess Objects Server is running, set this parameter to 0. If this is set to 0, and you make a change to a procedure definition, you will need to stop, then restart, the TIBCO iProcess Objects Server for the change to be recognized.

This parameter defaults to 300 in Windows because this same configuration utility may be used to configure multiple servers running on the same machine, and one of them may be an older version (in older TIBCO iProcess Objects Servers, this configuration parameter specified the number of seconds between cache updates, rather than working as a flag as it does now).

Lower Bound: 0 (Never update)
Default: 1 (UNIX), 300 (Windows)

CacheStartSessUpdate

Interval in seconds between updating the cache of startable procedures for a user.

If you don't change the list of procedures a user can start while the TIBCO iProcess Objects Server is running, set this parameter to 0. If this is set to 0, and you make a change to the list of startable procedures for a user, you will need to stop, then restart, the TIBCO iProcess Objects Server for the change to be recognized.

Lower Bound:0 (Never update)

Upper Bound: Unlimited

Default: 300 seconds

CacheUserUpdate

This specifies whether or not to update the cache of user, group, and attribute definitions. Setting this to 0 (zero) causes the cache to never be updated (other than the initial load on startup). Setting this to a positive number causes the cache to be updated in the background immediately after a change to the user, group, or attribute definitions.

If you don't change users, attributes, or groups while the TIBCO iProcess Objects Server is running, set this parameter to 0. If this is set to 0, and you make a change to a user, attribute, or group definition, you will need to stop, then restart, the TIBCO iProcess Objects Server for the change to be recognized.

This parameter defaults to 300 in Windows because this same configuration utility may be used to configure multiple servers running on the same machine, and one of them may be an older version (in older TIBCO iProcess Objects Servers, this configuration parameter specified the number of seconds between cache updates, rather than working as a flag as it does now).

Lower Bound: 0 (Never update)
Default: 1 (UNIX), 300 (Windows)

CacheRoleUpdate

This specifies whether or not to update the cache of role definitions. Setting this to 0 (zero) causes the cache to never be updated (other than the initial load on startup). Setting this to a positive number causes the cache to be updated in the background immediately after a change to the role definitions.

If you don't change role definitions while the TIBCO iProcess Objects Server is running, set this parameter to 0. If this is set to 0, and you make a change to a role definition, you will need to stop, then restart, the TIBCO iProcess Objects Server for the change to be recognized.

This parameter defaults to 300 in Windows because this same configuration utility may be used to configure multiple servers running on the same machine, and one of them may be an older version (in older TIBCO iProcess Objects Servers, this configuration parameter specified the number of seconds between cache updates, rather than working as a flag as it does now).

Lower Bound: 0 (Never update)
Default: 1 (UNIX), 300 (Windows)

CacheTableUpdate

This specifies whether or not to update the cache of table definitions. Setting this to 0 (zero) causes the cache to never be updated (other than the initial load on startup). Setting this to a positive number causes the cache to be updated in the background immediately after a change to the table definitions.

If you don't change table definitions while the TIBCO iProcess Objects Server is running, set this parameter to 0. If this is set to 0, and you make a change to a table definition, you will need to stop, then restart, the TIBCO iProcess Objects Server for the change to be recognized.

This parameter defaults to 300 in Windows because this same configuration utility may be used to configure multiple servers running on the same machine, and one of them may be an older version (in older TIBCO iProcess Objects Servers, this configuration parameter specified the number of seconds between cache updates, rather than working as a flag as it does now).

Lower Bound: 0 (Never update)
Default: 1 (UNIX), 300 (Windows)

CacheListUpdate

This specifies whether or not to update the cache of list definitions. Setting this to 0 (zero) causes the cache to never be updated (other than the initial load on startup). Setting this to a positive number causes the cache to be updated in the background immediately after a change to the list definitions.

If you don't change list definitions while the TIBCO iProcess Objects Server is running, set this parameter to 0. If this is set to 0, and you make a change to a list definition, you will need to stop, then restart, the TIBCO iProcess Objects Server for the change to be recognized.

This parameter defaults to 300 in Windows because this same configuration utility may be used to configure multiple servers running on the same machine, and one of them may be an older version (in older TIBCO iProcess Objects Servers, this configuration parameter specified the number of seconds between cache updates, rather than working as a flag as it does now).

Lower Bound: 0 (Never update)
Default: 1 (UNIX), 300 (Windows)

CacheSemaphoreMaxtries

Number of tries that are made to update the cache. (Also see the comments for CacheSemaphoreWait below.)

Do not change this parameter unless you are advised to do so by TIBCO Support.

Values: 1 - unlimited
Default: 1200 times

CacheSemaphoreWait

Number of milliseconds to wait to acquire all threads caching semaphores and update the needed cache. After this number of milliseconds, the caching semaphores will be released, allowing the other threads to process any pending transactions. Another attempt will then be made to acquire the caching semaphores; this will continue until the number of tries specified by CacheSemaphoreMaxTries are made.

Do not change this parameter unless you are advised to do so by TIBCO Support.

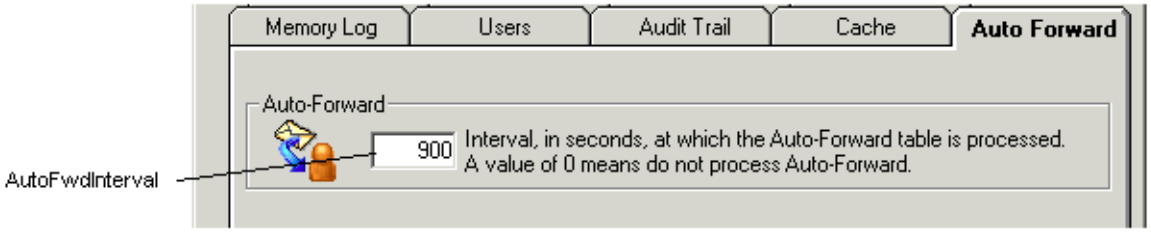
Values: 25 - 60000
Default: 100 milliseconds

WQSAbandonedPeriod

This parameter defines the number of seconds the TIBCO iProcess Objects Server will maintain a "persisted SWXList" (TIBCO iProcess Objects) or "held pageable list" (TIBCO iProcess Server Objects) that is being held on the TIBCO iProcess Objects Server (persisted SWXLists/held pageable lists can only contain work items or predicted work items). The TIBCO iProcess Objects Server will “abandon” (throw away) persisted SWXLists/held pageable lists in this number of seconds after you disconnect from the TIBCO iProcess Objects Server.

Values: 30- 43200
Default: 900 seconds

Auto Forward Parameters



This dialog is from the Windows *TIBCO iProcess Objects Server Configuration Utility*. The configuration parameter names shown in the callouts are from the UNIX TIBCO iProcess Objects Server configuration file. This illustration provides a cross-reference to determine which fields to change when using the configuration utility.

AutoFwdInterval

Number of seconds between each processing of the auto-forward table. A value of 0 means don't do auto forward processing.

This parameter is used only by older TIBCO iProcess Objects that did not have redirection functionality. It is not used by TIBCO iProcess Server Objects.

Lower Bound: 0
Upper Bound: none
Default: 900 (900 seconds = 15 minutes)

Configuring Auto-Forward and View-Only Queue Access

Beginning with version 10.5.0 of iProcess Objects Server, the auto-forward and view-only queue access functionality is turned off by default on Windows systems — this older functionality has been superseded by redirection and participation functionality, respectively.

This functionality is still automatically enabled on UNIX iProcess Objects Servers. This section is not applicable if you are using a UNIX iProcess Objects Server.

However, if you would still like to use this older functionality, you can turn it on by adding and setting a Registry entry, ensuring you have the Microsoft Access driver installed, then setting DCOM permissions. These are described in the following sections.

Adding Registry Entry

1. Create the following Registry string:

   ```
   \HKEY_LOCAL_MACHINE\SOFTWARE\Staffware plc\Staffware EntObj
   Server\Nodes\NodeName\DB_Enabled
   ```

   where *NodeName* is the name of your TIBCO iProcess Objects Server.

If the software is installed on a 64-bit machine, the Registry path will include "Wow6432Node", as follows:

`HKEY_LOCAL_MACHINE\SOFTWARE\`**`Wow6432Node`**`\Staffware plc\...`

2. Set the newly created DB_Enabled Registry entry to 1 to enable auto-forward and view-only queue access. This results in starting the `SWEntObjDB.exe` process when the server is started.

Setting Up the Microsoft Access Driver

You must have a Microsoft Access Driver (.mdb) ODBC driver set up for the IPEADMIN user to use auto-forward and view-only queue access functions.

The IPEADMIN user is designated when TIBCO iProcess Engine is installed. It defaults to the user installing iProcess Engine, but can be specified as any iProcess user.

To check if you have the Microsoft Access Driver installed, perform one of the following sets if steps, depending on the type of machine you are using:

32-Bit Machine

1. Log on to Windows as the IPEADMIN user.
2. From Administrative Tools, select **Data Sources (ODBC)**.
3. Click the **Drivers** tab. If you have the driver installed, there will be an entry for Microsoft Access Driver (.mdb) in the list.

   If you do not have the Microsoft Access Driver installed, you must download a copy of it from Microsoft's website and install it according to their instructions.

64-Bit Machine

Note - Following the steps above to access the ODBC Data Source Administrator does not allow you to view 32-bit drivers.

1. Log on to Windows as the IPEADMIN user.
2. Create a shortcut that points to "`%SystemRoot%\SysWOW64\odbcad32.exe`" with the "Start in" field set to "`%SystemRoot%\SysWOW64`".
3. Start the ODBC Data Source Administrator using the shortcut created in step 2.
4. Click the **Drivers** tab. If you have the driver installed, there will be an entry for Microsoft Access Driver (.mdb) in the list.

   If you do not have the Microsoft Access Driver installed, you must download a copy of it from Microsoft's website and install it according to their instructions.

Setting Up DCOM Permissions

You must also set up DCOM permissions to use the auto-forward and view-only queue access functions.

If DCOM is enabled on your machine, you must give the IPEADMIN user Access Permissions and Launch Permissions. And if you are running Windows, you must also give the IPEADMIN user Configuration Permissions (see the note above about the IPEADMIN user).

You may have DCOM enabled without having explicitly set it. For example, Microsoft Web Server enables it as part of its setup.

To Check if DCOM is Enabled

To check if DCOM is enabled, complete these steps:

1. Log on to Windows as an administrator.
2. Click the **Start** button, then select **Run**.
3. Enter **`dcomcnfg`** and press **OK**. This runs the DCOM Configuration Utility.
4. From the Component Services icon, drill down to the My Computer icon.
5. Right click on **My Computer**, then select **Properties**. The My Computer Properties dialog is displayed.
6. Click the **Default Properties** tab.

If the **Enable Distributed COM** check box on this computer is selected, DCOM is enabled and you must complete the steps in Adding Permissions for the IPEADMIN User. Otherwise, DCOM is not enabled — no further action is required.

Adding Permissions for the IPEADMIN User

To give the IPEADMIN user access and launch permissions, complete these steps:

1. In the My Computer Properties dialog, click the **COM Security** tab.
2. Complete the following steps to set up Access Permissions:
   a. In the Access Permissions frame, click **Edit Default**; then click **Add**.
   b. In the Select Users or Groups dialog, click **Locations**.
   c. In the Locations dialog, choose the appropriate computer or domain name where TIBCO iProcess Engine is installed; then click **OK**.
   d. In the Select Users or Groups dialog, click **Advanced**.
   e. Click **Find Now**.
   f. From the list of search results, select the IPEADMIN user; then click **OK**.
   g. In the Select Users or Groups dialog, click **OK**. The IPEADMIN user is now displayed in the list of users with access permissions.
   h. Click the IPEADMIN user and ensure the **Allow** check box is selected. If it is not select, select the **Allow** check box.
   i. Click **OK** to return to the My Computer Properties dialog.

3. Complete the following steps to set up Launch Permissions:
   a. In the Launch Permissions frame, click **Edit Default**; then click **Add**.
   b. In the Select Users or Groups dialog, click the **Locations** button.
   c. In the Locations dialog, choose the appropriate computer or domain name where TIBCO iProcess Engine is installed; then click **OK**.
   d. In the Select Users or Groups dialog, click **Advanced**.
   e. Click **Find Now**.
   f. From the list of search results, select the IPEADMIN user; then click **OK**.
   g. From the Select Users or Groups dialog, click **OK**. The IPEADMIN user is now displayed in the list of users with launch permissions.
   h. Click the IPEADMIN user and ensure the **Allow** check box is selected. If it is not select, select the **Allow** check box.
   i. Click **OK** to return to the My Computer Properties dialog.
   j. Click **OK** to return to the Component Services dialog.
   k. Exit Component Services.

Chapter 3 TIBCO iProcess Objects Server Log

This chapter describes the use of the TIBCO iProcess Objects Server log.

Topics

Introduction

The TIBCO iProcess Objects Server log records messages generated by TIBCO iProcess Objects Server. These messages can be used to determine where problems are occurring in the server.

The server log contains one line for each message. They are in the format:

```
ppppp|ttttt|dd/dd/dddd dd:dd:dd.ddd|cccccccc|llll|Message
```

where:

ppppp	Process ID
ttttt	Thread ID
dd/dd/dddd dd:dd:dd.ddd	Date and time
cccccccc	Log category (hex format)
llll	Log level (ERROR, WARN, INFO, DEBUG) or TRACE indicating that this message is logging a request/response message
Message	Server log message

An example server log entry is shown below:

```
00181|0011C|07/02/2004 14:54:44.878|00100000|ERROR|manadatory marking (PURPOSE) not
specified
```

The TIBCO iProcess Objects Server log file contains a header that provides information about TIBCO iProcess Objects Server, including its version number. Note that the version number provided in the header is in the "older" format, i.e., it contains parentheses and an "i" if it is TIBCO iProcess Objects Server that is compatible with a TIBCO iProcess Engine. The "newer" version number format does not contain parentheses and it does not contain an "i" if it is version 10.2.0 or newer. This newer format is shown in most other places in which you will see a version number. For more information, see TIBCO iProcess Objects Server Version on page 3.

Types of TIBCO iProcess Objects Server Logs

The TIBCO iProcess Objects Server log is available in two different formats: on-disk and in-memory. These are described below.

On-Disk Log File

This is the "standard" way of recording TIBCO iProcess Objects Server messages. The messages are written to a file on the hard disk and are available via a text editor.

The on-disk TIBCO iProcess Objects Server log is always "turned on" (as opposed to the in-memory log, which can be turned on or off — see below). The primary controller of the amount of information that is written to the on-disk TIBCO iProcess Objects Server log is the setting of the *log level*. The log level can be set so that only error-level messages (the fewest) are written to the log, or it can be set so that debug-level messages (the greatest amount) are written to the log. You can also control the size of the log file, the types of categories that are written to the log, etc. These are described in the following subsections.

In-Memory Log File

This method of storing log messages allows you to specify that they be stored in memory during server operations, rather than on disk; the log is written to disk if the server terminates abnormally (crashes). This method of TIBCO iProcess Objects Server logging can be used if the performance impact of using the "standard" on-disk logging is too great (especially when logging at the debug level).

In-memory logging is available only if your TIBCO iProcess Objects Server has CR 14205 implemented.

This method of logging is turned off by default. To turn it on, you must do the following:

- Windows - Check the **Enable Memory Logging** check box on the Memory Log tab using the TIBCO iProcess Objects Server Configuration Utility. This then enables the rest of the selections on the tab so you can set the in-memory log level and categories.
- UNIX - Set the MemLogLevel TIBCO iProcess Objects Server configuration parameter to a value from 1 (error level) to 4 (debug level). A setting of zero disables in-memory logging.

For more information, see the `MemlogLevel` parameter on MemLogLevel on page 51

Writing the In-Memory Log to Disk

The following describes the ways in which the in-memory log is written to disk (if in-memory logging is enabled):

- If TIBCO iProcess Objects Server crashes, an attempt is made to automatically write the in-memory log to a file.

This only occurs if the "auto dump" feature has been turned on in the engine (using the `PROCESS_AUTO_DUMPLOG` process attribute). If auto dump is enabled (the default), the contents of the processes' debug shared memory segment is written to disk when that process fails.

For more information, see *TIBCO iProcess Engine Administrator's Guide*.

- You can manually write the in-memory log to a file using the `swsvrmgr DUMPLOG` command (even when the server is still running). The syntax is:

```
swsvrmgr DUMPLOG [<MachineName>|<MachineID>[<ProcessName>[<ProcessInstance>]]]
```

Note that if the server crashes, and the log cannot be written to a file for some reason, the shared memory containing the log may still exist. If the shared memory still exists, you can manually write the in-memory log to a file using the `swsvrmgr DUMPLOG` command shown above.

Name and Location of the TIBCO iProcess Objects Server Log

On-disk Log File

The on-disk TIBCO iProcess Objects Server log file can be found in the following locations:

- Windows - *SWDIR*\logs\SWEntObjSv*XX*.log
- UNIX - $*SWDIR*/logs/swentobjsv*XX*.log

  where *XX* is the instance number of TIBCO iProcess Objects Server. See Log File Names on page 10 for more information.

In-memory Log File

If the in-memory log file is written to the disk, the log file can be found in the following locations:

- $*SWDIR*/logs/SPO*xx*_*YYYYMMDD_HHMMSS*_<*PID*>.dmp (primary dump file)
- $*SWDIR*/logs/SPO*xx*_*YYYYMMDD_HHMMSS*_<*PID*>b.dmp (secondary dump file)

  where:

  — *xx* is the instance number of TIBCO iProcess Objects Server.

  — *YYYYMMDD_HHMMSS* is the date and time the log was written to disk.

  — <*PID*> is the process ID of TIBCO iProcess Objects Server that is or is not running.

Only the primary dump file is created (and not a secondary dump file) if the SAL SDK and TIBCO iProcess Objects Server logs are merged. If the logs are not merged, the primary dump file will contain just the SAL SDK log messages and there will be a secondary dump file which will contain the TIBCO iProcess Objects Server log messages. (For information about merging the logs, see the DEBUG Process Attribute in *TIBCO iProcess Engine Administrator's Guide*.)

Note that each server's log information is stored in a shared memory segment. These segments will persist until the server is restarted (every time a server is started, it creates and uses a new memory segment).

To change the log files directory, specify the directory in the staffpms file located in the *SWDIR*\etc directory. For more information, see "Configuring Log Files Directory" in *TIBCO iProcess Engine Administrator's Guide*.

Archiving TIBCO iProcess Objects Server Log Files

When the size of the on-disk TIBCO iProcess Objects Server log file reaches the value specified in the `LogFileMaxSize` configuration parameter (see LogFileMaxSize on page 48), the log rolls over. When it rolls over, the log is cleared and a new log is created.

You can specify that the log file be written to an archive log before it is cleared by specifying a value in the `LogFileMaxArchives` configuration parameter. The system will save the number of archived log files as specified in LogFileMaxArchives, allowing you to save many megabytes of debug log without setting the LogFileMaxSize to a very large size. The default is to not write the log to an archive file. See LogFileMaxArchives on page 48 for information about the naming conventions used for archived log files.

The in-memory log file cannot be archived.

Controlling the Server Log

You can control certain aspects of the server log, such as the level of information that is written to the log, the maximum size of the log, etc.

You can control the server log:

- using the object model
- using the TIBCO iProcess Objects Server Configuration Utility (Windows)
- using the TIBCO iProcess Objects Server Configuration File (UNIX)

Using the Object Model

TIBCO iProcess Objects

If you are using TIBCO iProcess Objects (COM, Java, or C++), the SWNode object is used to control the TIBCO iProcess Objects Server log. The SWNode object contains a number of methods that are used to control the on-disk TIBCO iProcess Objects Server log. These methods allow you to specify the amount of information to write to the log, reset the log, etc. The functionality provided through these methods is described in the subsections that follow.

TIBCO iProcess Server Objects

If you are using TIBCO iProcess Server Objects (Java or .NET), the sNodeManager object is used to control the TIBCO iProcess Objects Server log. The sNodeManager object contains two methods that are used to control the on-disk TIBCO iProcess Objects Server log:

- `SetSrvLogOptions` - This method allows you to specify the amount and type of information to write to the log.
- `ResetSrvLog` - This method allows you to reset the log.

The functionality provided through these methods is described in the subsections that follow.

Throughout this chapter, the TIBCO iProcess Objects method names provided are for COM — the Java and C++ method names are the same, except the first character is always lowercase (e.g., SetLogLevel vs. setLogLevel).

Also, the TIBCO iProcess Server Objects method names provided are for .NET — the Java method names are the same except the first character is always lowercase (e.g., SetSrvLogOptions vs. setSrvLogOptions). This is done for brevity.

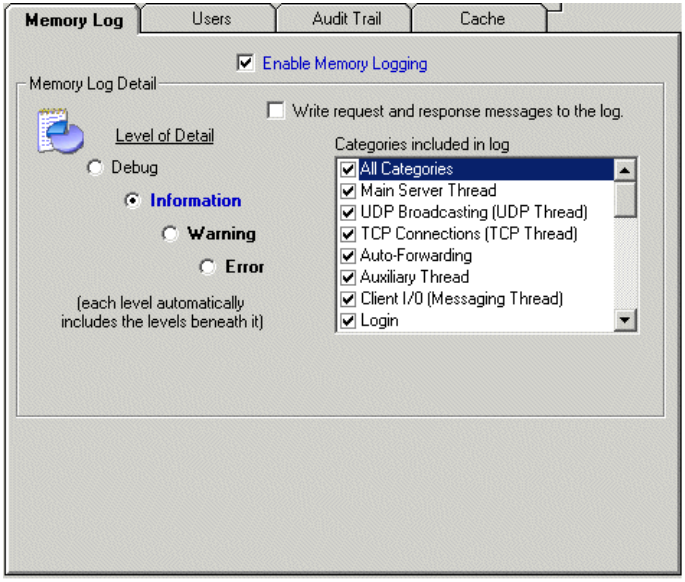
The in-memory log cannot be controlled through the object model; it can be controlled only through the Configuration Utility (Windows) or configuration parameters (UNIX).

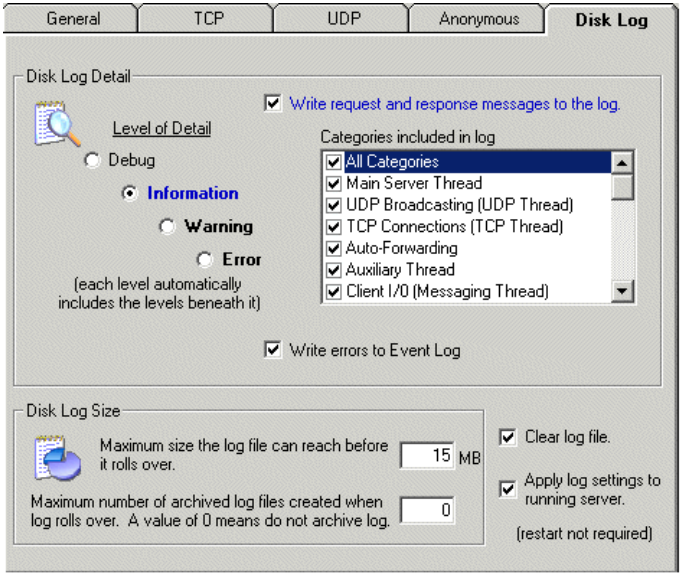
Using the TIBCO iProcess Objects Server Configuration Utility (Windows Only)

In Windows-based systems, the TIBCO iProcess Objects Server log can be controlled using the TIBCO iProcess Objects Server Configuration Utility. This utility can be accessed in one of two ways:

- Through TIBCO iProcess Objects Server control panel applet.
- Executing the *SWDIR*`\bin\SWEntObjSvCfg.exe` file.

Both of these methods will cause the TIBCO iProcess Objects Server Configuration Utility dialog to display. Select the appropriate server from the drop-down list, then click on the Memory Log tab (for in-memory log settings) or Disk Log tab (for on-disk log settings) . One of the following dialogs is displayed:





The functions available through these dialog are described in the subsections that follow.

Notice the Apply Log Settings... check box in the lower right corner on the Disk Log tab. If you make a change to one or more of the disk log settings, then check this box, the change takes effect immediately after you click on **OK**, without requiring you to stop and restart TIBCO iProcess Objects Server. If you do not check this box, any changes made are saved, but they will not take effect until you stop, then restart TIBCO iProcess Objects Server.

Using the TIBCO iProcess Objects Server Configuration File (UNIX Only)

In UNIX-based systems, the TIBCO iProcess Objects Server memory log and disk log can be controlled by making changes to the TIBCO iProcess Objects Server configuration file, `swentobjsv.cfg`. This file can be modified with your choice of text editors.

The TIBCO iProcess Objects Server configuration file is located in the `$`*SWDIR*`/seo/data` directory.

Before making changes to the TIBCO iProcess Objects Server configuration file in UNIX, you must shut down TIBCO iProcess Objects Server. After making the desired changes, restart TIBCO iProcess Objects Server. See Starting/Stopping the TIBCO iProcess Objects Server on page 5 for information.

Setting the Level of Detail to Log

You can log up to four levels of increasing detail in the server log. They are:

Log Level	Value	Amount of Detail
Error	1	Least Amount
Warning	2	(Default)
Informational	3	
Debug	4	Most Amount

The log levels are hierarchical, from the least amount of information to the most, with each higher level including the information from the levels below it. The default log level is `Warning`.

Unless directed by TIBCO Support, the log level should be specified as either "Error" or "Warning." Setting the level to "Debug" causes an extremely large number of messages to be written to the log file. This will cause the performance and response time of TIBCO iProcess Objects Server to be seriously degraded. It also causes the possibility of critical error messages being lost if the log file fills up and rolls over.

The log level can be changed in the following ways:

- Object Model - Only the on-disk log level can be set via the object model. The following methods can be used to specify the on-disk log level:
  - TIBCO iProcess Objects - `SetLogLevel` method
  - TIBCO iProcess Server Objects - `SetSrvLogOptions` method (using the *aLevel* parameter)

  The available log levels are defined in the enumeration type `SWLogLevelType`. They are:

SWLogLevelType	Value
swLogError	1
swLogWarning	2
swLogInformational	3

SWLogLevelType	Value
swLogDebug	4

You must have system administrator authority (MENUNAME = ADMIN) to call these methods.

- TIBCO iProcess Objects Server Configuration Utility:
  - For on-disk logging, click on the appropriate **Level of Detail** radio button on the Disk Log tab.

    For information about the available log levels, see LogLevel on page 47.
  - For in-memory logging, click on the appropriate **Level of Detail** radio button on the Memory Log tab.

    For information about the available log levels, see MemLogLevel on page 51.
- TIBCO iProcess Objects Server Configuration File:
  - For on-disk logging, locate the LogLevel entry in the `swentobjsv.cfg` file and set it to the appropriate numeric value for the desired level of detail. For information about the LogLevel parameter, see LogLevel on page 47.
  - For in-memory logging, locate the MemLogLevel entry in the `swentobjsv.cfg` file and set it to the appropriate numeric value for the desired level of detail. This parameter can also be set to 0 (zero) to disable in-memory logging (the default). For information about the MemLogLevel parameter, see MemLogLevel on page 51.

Logging Request/Response Messages in the Server Log

Logging request/response (send/receive) messages can be very useful for TIBCO Support when trying to debug a problem. This ability adds detailed information to the log concerning messages the client sends to the server, and the responses received back from the server. This is also known as "trace" message logging.

Example request/response messages in the server log are shown below:

```
00181|0011C|07/03/2001 08:11:25.988|00010000|TRACE|SEND|10.20.30.43:44812|Len(4092)
|MsgCode(QQ)
00000: 00 00 0f f8 00 00 02 5c  46 00 00 00 00 00 00 0f  |.......\F.......|
00016: fc 51 51 00 73 77 5f 64  61 6e 61 00 32 00 30 00  |.QQ.sw_quest.2.0.|

00181|0011C|07/03/2001 08:11:26.229|00000020|TRACE|RECV|10.20.30.43:44812|Len(134)|
MsgCode(QQ)
00000: 00 00 02 30 73 77 61 64  6d 69 6e 0a 73 77 61 64  |...0swadmin.swad|
00016: 6d 69 6e 0a 51 51 0a 73  77 61 64 6d 69 6e 40 6d  |min.QQ.swadmin@m|
```

Notice TRACE is logged in place of one of the log levels described in the previous section. This is followed by either SEND or RECV to indicate if it's a request or response message, respectively. This is then followed by the data in the message to/from the server.

Unless directed by TIBCO Support, it is highly recommended that the request/response logging functionality be turned off. Turning this functionality on could cause an extremely large amount of information to be written to the server log. This will cause the performance and response time of TIBCO iProcess Objects Server to be seriously degraded. It also causes the possibility of critical error messages being lost if the log file fills up and rolls over

The default is for request/response messages to not be logged.

Request/response message logging can be turned on/off using the following:

- Object Model - Only the on-disk log tracing be set via the object model. The following methods can be used to set log trace for the on-disk log:
  - TIBCO iProcess Objects - `SetLogTrace` method
  - TIBCO iProcess Server Objects - `SetSrvLogOptions` method (using the *aTrace* parameter)

- TIBCO iProcess Objects Server Configuration Utility:
  - For on-disk logging, click in the Write request and response messages to the log check box on the Disk Log tab to enable tracing.
  - For in-memory logging, click in the Write request and response messages to the log check box on the Memory Log tab to enable tracing,
- TIBCO iProcess Objects Server Configuration File:
  - For on-disk logging, locate the TraceMsg entry in the `swentobjsv.cfg` file and set it to “0” to turn off trace message logging or “1” to turn on trace message logging. See TraceMsg on page 44 for more information about the TraceMsg parameter.
  - For in-memory logging, locate the MemTraceMsg entry in the `swentobjsv.cfg` file and set it to “0” to turn off trace message logging or “1” to turn on trace message logging. See MemTraceMsg on page 53 for more information about the MemTraceMsg parameter.

Filtering the Server Log by Category

When logging information to the server log, you can filter the information according to categories, causing only some categories of information to be logged and others to be disregarded. The available categories are:

Category	Value	SWSrvLogCategoryType
All Categories	0x7FFFFFFF	swAllSrvLogCategories
Main Server Thread	0x00000001	swCatMainThd
UDP Broadcast (UDP Thread)	0x00000002	swCatUDPThd
TCP Connection (TCP Thread)	0x00000004	swCatTCPThd
Auto Forward Thread	0x00000008	swCatAutoFwdThd
Auxiliary Thread	0x00000010	swCatAuxThd
Client I/O (Message Processing Threads)	0x00000020	swCatMsgThd
Logins/Logouts	0x00000040	swCatLogin
Password Changes	0x00000080	swCatPassword
User Operations (add, remove, change, list users)	0x00000100	swCatUser
Attribute Operations (add, remove, change, list attributes)	0x00000200	swCatAttribute
Role Operations (add, remove, list roles)	0x00000400	swCatRole
Group Operations (add/remove group, add/remove user from/to group)	0x00000800	swCatGroup
Procedure Operations (list/query)	0x00001000	swCatProc
Procedure Queries	0x00002000	swCatProcQuery
Procedure Definitions	0x00004000	swCatProcDef
Work Queue Operations (add, remove, list)	0x00008000	swCatQueueAccess

Category	Value	SWSrvLogCategoryType
Work Item Lists (queue query, get queue item, destroy view)	0x00010000	swCatQueueQuery
Case Operations (close, purge)	0x00020000	swCatCase
Node Operations (list)	0x00040000	swCatNode
Event Operations (trigger)	0x00080000	swCatEvent
Work Item Operations (get, keep, release, forward)	0x00100000	swCatWorkItem
Work Item Forwarding (forward item, add/remove/list auto-forward list)	0x00200000	swCatForwarding
Server Instrumentation	0x00400000	swCatInstrumentation
Lists and Tables	0x01000000	swCatSWListTable
Log Operations (set level/categories)	0x02000000	swCatLog
SAL SDK Timing	0x04000000	swCatSALTiming
TIBCO iProcess Objects Director Operations	0x08000000	swCatDirector

The default is to log all categories.

You can specify which categories of information to write to the server log using the following:

- Object Model - Only the on-disk log categories can be set via the object model. The following methods can be used to set log categories:
  - TIBCO iProcess Objects - `SetLogCategories` method
  - TIBCO iProcess Server Objects - `SetSrvLogOptions` method (using the *aCategories* parameter)

  These methods use the SWSrvLogCategoryType enumeration type to specify the categories to log. These enumerations are shown in the table above.

  Besides using the enumerations, you can specify the categories using the hex values. This allows you to combine the values to specify any combination of categories you wish.

- TIBCO iProcess Objects Server Configuration Utility:
  - For on-disk logging, check the appropriate category check boxes in the Categories included in log section on the Disk Log tab.
  - For in-memory logging, check the appropriate category check boxes in the Categories included in log section on the Memory Log tab.
- TIBCO iProcess Objects Server Configuration File:
  - For on-disk logging, locate the LogCategories entry in the `swentobjsv.cfg` file and set it to the appropriate hex value for the categories you want to log (see the table above for the hex values). You can combine the hex values to specify any combination of categories to log. See LogCategories on page 45 for more information about the LogCategories parameter.
  - For in-memory logging, locate the MemLogCategories entry in the `swentobjsv.cfg` file and set it to the appropriate hex value for the categories you want to log (see the table above for the hex values). You can combine the hex values to specify any combination of categories to log. See MemLogCategories on page 52 for more information about the MemLogCategories parameter.

Setting the Size of the Server Log File

For on-disk logging, when the log file exceeds its specified maximum size, it is cleared and restarted (rolled over). A message is written to the log indicating it has been cleared and restarted. The default size of the on-disk log is 15MB.

The size of the in-memory log is determined by the DBGMEMSIZE_KB Process Attribute in the TIBCO iProcess Engine — it defaults to 256K. For more information, see *TIBCO iProcess Engine Administrator's Guid*e.

Only the on-disk server log file (`swentobjsv`*XX*`.log` on UNIX; `SWEntObjSv`*XX*`.log` in Windows) is rolled over when it reaches the maximum size. (The audit log (`swentobjua`*XX_timestamp*`.log` on UNIX; `SWEntObjUa`*XX_timestamp*`.log` in Windows) is never rolled over by TIBCO iProcess Objects Server. It is the customer's responsibility to back up and remove the audit log. See Log File Names on page 10 for more information.

You can specify the size of the server log using the following:

- Object Model - The following methods can be used to set the log size:
  - TIBCO iProcess Objects - `SetMaxLogSize` method
  - TIBCO iProcess Server Objects - `SetSrvLogOptions` method (using the *aMaxSize* parameter)
- TIBCO iProcess Objects Server Configuration Utility - The Log Size section on the Log tab contains a field in which you can specify the number of megabytes to set the maximum size of the log.
- TIBCO iProcess Objects Server Configuration File - Locate the LogFileMaxSize entry in the `swentobjsv.cfg` file and set it to the desired maximum size, in megabytes. See LogFileMaxSize on page 48 for more information about the LogFileMaxSize parameter.

Resetting the Server Log

This only applies to the on-disk log file. The following methods are available to reset the server log.

- Object Model:
  - TIBCO iProcess Objects - `ResetLog` method
  - TIBCO iProcess Server Objects - `ResetSrvLog` method

  These methods clear the existing log and write an initial header message to the log.
- TIBCO iProcess Objects Server Configuration Utility (Windows) - Click in the **Clear log file** check box. The log will be cleared when you click either the **Apply** or **OK** button. (Note - The only other way to clear the server log on TIBCO iProcess Objects Server on UNIX, besides using the `ResetLog` method, is to manually delete it.)

Chapter 4 Audit Log

This chapter describes how to use the audit log generated by TIBCO iProcess Objects Server.

Topics

Introduction

An entry is written to the audit log whenever a TIBCO user performs any of the following administrative-type functions:

- Creates or removes a user
- Creates, removes, or sets an attribute
- Creates or removes a group
- Adds or removes a member from a group
- Adds or removes a role
- Changes a user's password

To log the changes of the case data made by iProcess Insight, iProcess Workspace (Browser), or the `setCaseData` TIBCO iProcess Server Objects interface, rather than by using normal step processing, you need to configure the `audit_casedata_changed` attribute. For more information about this attribute, see *TIBCO iProcess Engine Administrator's Guide*.

This log documents who performed an administrative function, and when they performed it, which can be very helpful when trying to track down problems. An example entry in the audit log is:

```
07/05/2001 14:31:25.076|swadmin set attribute for emartinez, MENUNAME = ADMIN
```

New audit log messages that are generated by TIBCO iProcess Objects Server are appended to the existing audit log. Note that the audit log is never rolled over, and there are no functions for clearing or deleting it. It is the customer's responsibility to back up and remove this log.

The audit log is located in the following locations:

- Windows - *SWDIR*`\logs\SWEntObjUa`*XX*_*timestamp*`.log`
- UNIX - `$`*SWDIR*`/logs/swentobjua`*XX*_*timestamp*`.log`

where *XX* is the instance number of TIBCO iProcess Objects Server, and the *timestamp* variable is the date when the log was generated. If you are only running a single instance of TIBCO iProcess Objects Server on UNIX, this number will be `01`.

Activating/Deactivating the Audit Log

By default, the audit log is active, and will record all of the activities listed above. It can be turned on and off in the following ways:

- Windows - Execute TIBCO iProcess Objects Server Configuration Utility control panel applet. The Audit check box on the Users tab can be used to turn on (checked) or turn off (unchecked) the audit log.
- UNIX - Locate the AuditUserAdmin entry in the TIBCO iProcess Objects Server Configuration File (`$`*SWDIR*`/seo/data/swentobjsv.cfg`) and set it to "0" (off) or "1" (on).

Chapter 5 UNIX System Log

This chapter describes using the UNIX System Log.

Topics

Using the UNIX System Log

The UNIX system log (also known as `syslog`) is a general-purpose logging facility available when you are running TIBCO iProcess Objects Server on a UNIX system.

Messages that are written to the audit log (see Audit Log on page 91) are always written to the UNIX system log. You can also optionally specify that messages that are written to the TIBCO iProcess Objects Server log (other than Debug messages) be written to the UNIX system log.

To specify that you want TIBCO iProcess Objects Server log messages written to the UNIX system log, locate the `UseSysLog` parameter (see UseSysLog (UNIX Only) on page 49) in the TIBCO iProcess Objects Server configuration file (`$`*SWDIR*`/seo/data/swentobjsv.cfg`) and set it to "1" (the default is "0" — information is not written to the UNIX system log).

The location of the UNIX system log can be configured on each UNIX system, but the usual locations are:

- Solaris - `/var/adm/messages`
- HP-UX - `/var/adm/syslog/syslog.log`
- Linux - `/var/log/messages`
- AIX - various log files in `/var/adm`

All syslog messages are categorized by the type of "subsystem" or "facility" that originated the message, and by the "priority" given the message. The "subsystems" are areas such as "kernel" (message generated by the kernel, i.e., UNIX itself), "user" (messages from various user programs), "mail," "daemon," "auth," and "lpr." There are also "local" subsystems (local0 through local7) that are reserved for local program use. The TIBCO iProcess Objects Server uses one of these — local0.

Within each subsystem, there are various priority levels. In the TIBCO iProcess Objects Server, the priorities that are used correspond to the TIBCO iProcess Objects Server log levels/types. They are:

- local0.info - includes "info," "notice," "warn," and "err"
- local0.notice - includes "notice," "warn," and "err"
- local0.warn - includes "warn," and "err"
- local0.err - includes "err" only

Notice that each priority also includes the levels below it.

The UNIX system log file is controlled by the configuration file `/etc/syslog.conf`.

You could optionally choose to send all TIBCO iProcess Objects Server messages to a different file by adding a line similar to the following to the `syslog.conf` file:

```
local0.info /var/adm/spo_messages_only
```

Note that whenever the `syslog.conf` file is changed, the syslogd daemon must be sent a SIGHUP signal. For example:

```
kill -HUP ‘cat /etc/syslog.pid’
```

Index

A

B

C

D

E

F

G

I

J

K

L

M

N

O

P

R

S

T

U

V

W