Peoplesoft messaging server

Science, Computer Science

Contents PeopleSoft Messaging Server Settings Guide1 Contents2 Introduction Introducing the PeopleSoft Messaging Server3 Messaging Server Processes4 Configuring Messaging Servers in PSADMIN4 Understanding Dispatcher Parameters5 Understanding Handler Parameters7 Understanding Integration Broker Parameters8 Minimum and Recommended Values. 9 Edit History10 Introduction Introducing the PeopleSoft Messaging Server PeopleSoft Messaging Services exist on the application server and are the heart of the Integration Broker.

Before using Integration Broker, you must configure and start the Messaging Server, aka PUB/SUB. Although the server processes devoted to your messaging system are all part of the larger application server domain, they comprise a distinct set of processes that aren’t involved with the ordinary transactions associated with PIA connections. Six processes of two different types, dispatchers and handlers, are combined in pairs to produce the messaging servers needed for transmitting messages throughout your messaging system. Each messaging server is a different type.

A set of three — a publication broker, a publication contractor, and a subscription contractor — constitute the messaging server set required by Integration Broker. Following is a listing of the generic names for the processes: Messaging ServerDispatcher NameHandler Name Publication Broker (BRK)PSBRKDSPPSBRKHND Publication Contractor (PUB)PSPUBDSPPSPUBHND Subscription Contractor (SUB)PSSUBDSPPSSUBHND PeopleSoft delivers default PUB/SUB services with _dflt added to the above naming convention. For example

PSBRKDSP_dflt. It is recommended that you use these services unless you have a specific need for dedicated handlers.

To boot PUB/SUB use PSADMIN to configure your domain and simply answer Y to the following question at the end of the configuration process: Command to execute (1-7, q) : 4 Do you want the Publish/Subscribe servers configured (y/n)? [y]: y For typical implementations, there is no need to configure custom or additional dedicated messaging servers as the default messaging services will handle all basic messages. Please see the last section of this guide for recommended values More information about managing the application server can be found in the PeopleSoft Server Tools Administration Peoplebook.

Additional Information available in Peoplebooks under: Home > PeopleBooks Library > PeopleSoft Integration Broker > Configuring the Messaging Messaging Server Processes There are a variety of server processes devoted to application messaging. If you are not implementing the application messagingtechnologythen you may skip through the delivered, default server processes. The delivered server processes are: •PSBRKDSP •PSBRKHND •PSPUBDSP •PSPUBHND •PSSUBDSP •PSSUBHND These server processes act as brokers, dispatchers, and handlers of the messages in your messaging system.

For the purposes of this paper we will divide these into two categories: Dipatchers and Handlers. Configuring Messaging Servers in PSADMIN This section provides overviews of messaging server configuration, dispatcher parameters, and handler parameters. Understanding Messaging Server Configuration Once you create dedicated messaging servers, you must

configure their dispatcher and handler processes so they boot when you start the application server. You configure these processes using PSADMIN just as you do any other server process that runs on the application server.

Before you attempt to configure additional messaging server processes, you should be familiar with the other server processes that run on the application server. For more information, please see Peoplebooks Working With PSADMIN Menus. As stated earlier, two types of server processes comprise each messaging server: a dispatcher and a handler. Each process type requires you to set a different set of parameters. Most of the parameters are similar to other server processes, such as PSSAPPSRV, but some parameters are specific to messaging servers. Note.

The following sections also apply to the _dflt messaging server processes. Only one parameter is different between a dedicated messaging server process and its _dflt counterpart: the Channels parameter, which enables you to add message channels to the channel list. The _dflt server processes can’t be associated with any specific message channel. Understanding Dispatcher Parameters There are three generic process types that are the basis for all dispatcher processes: •PSBRKDSP — the publication broker dispatcher. •PSPUBDSP — the publication contractor dispatcher. PSSUBDSP — the subscription contractor dispatcher. The following parameters apply to all three process types. Recycle Count Specifies the number of times each dispatcher process will be executed before being terminated (intentionally) by the system and then immediately restarted. Servers must be intermittently recycled to clear buffer areas. The time required to recycle a server is negligible—occurring in milliseconds. Recycle Count does not

translate into a native Tuxedo parameter in the PSAPPSRV. UBB file. Instead the value is stored in memory and is managed by the system.

Allowed Consec Service Failures This option allows for dynamic server process restarts in the event of service failures. To enable this option, enter a number greater than zero, and to disable this option enter 0. The default value for this parameter is 2. The value you enter is the number of consecutive service failures that will cause a recycle of the server process. This is a catchall error handling routine that allows a dispatcher to terminate itself if it receives multiple, consecutive, fatal error messages from service routines.

Such errors should not occur consecutively, but if they do it indicates that the server process needs to be recycled or cleansed. A “ Retry” message appears when the number of service failures you specified occurs. Handler Status CheckcountHandler check count is used to determine how often the dispatcher should look to get the number of associated handlers. The value of Handler Status Checkcount is the number of cycles that the dispatcher will perform before reading the MIB and getting the number of associated handlers. This comes into play when the number of handlers change (add more, some crash etc. by having the proper count , the dispatcher can queue up messages to the handler more efficiently. Also if there are no handlers, then the dispatcher will not queue up any publications causing the application server log to fill up. For 8. 4 it is simply used to determine if there are any handlers, and if not don't send the message to the handler. This is to eliminate any the informational messages in the appserv. log if the handlers are down. For 8. 42 it is used to merely look at see if any associated handler

is booted. Going forward 8. 3 it will be used as one of the determinate of how much work should the dispatcher send out at one time. Scan IntervalSpecifies the number of seconds between scans of the work queue when idle. The scan interval is necessary to detect messages published from two-tier connections, because when a message is in the queue the broker server doesn’t receive a notice of the publication. A scan interval is required to make sure that two-tier messages get processed in a timely manner. The scan interval is analogous to the Process Scheduler polling the Process Request table.

In addition, the scan interval detects messages that have been resubmitted after an error, for example. Decreasing the scan interval will decrease latency for two-tier publishes and error recovery Ping RateUsed for PSPUBDSP only. After this many seconds of inactivity, the server will scan the database queues and restart any stalled/crashed items. The scan rate and Ping rate (as percentage) will determine the actual interval for pinging any unavailable remote nodes (algorithm used: Attempts * Ping Rate * Scan Inteval).

Maximum Ping IntervalThe maximum Ping Interval (in Hours) is the maximum interval between subsequent attempted pings of any unavailable remote nodes. Memory Queue Refresh Rate PeopleSoft Integration Broker maintains current asynchronous messaging queues in system memory for quick access. On rare occasions these cached queues can become corrupted, at which point they must be refreshed from the Integration Broker data tables. The likelihood and frequency of cache corruption depends on a combination of factors specific to your messaging system.

If you need to periodically refresh the in-memory queues, you can use this parameter to tailor the frequency of the refresh to fit your situation. Each dispatcher on your system has its own queue. For each queue you set the rate equal to the number of dispatch attempts that must occur before the queue is refreshed. The refresh occurs only when the specified number of dispatch attempts is reached for a given message channel. For example, with a memory queue refresh rate of 8, multiple channels could have up to seven dispatch attempts each without triggering any refresh.

The following settings are also significant: •A setting of 0 disables the refresh altogether. This is the default value. A setting of 1 triggers a refresh immediately after every dispatch attempt, effectively disabling memory caching. Restart Period Specifies the number of seconds between restart attempts on Started items in the work queue. An item which stays in Started state for more than a few seconds might be stalled — for example, the service request might have been lost, or the handler might have crashed. Decreasing the restart period will reduce the latency for recovering stalled items with a status of Started.

However, under high load, items might stay in the Started state longer than normal for valid reasons — all the handlers might be busy, and the handler service request for the item might be queued at the Tuxedo level. Setting the restart period too low will result in redundant restarts — the dispatcher will dispatch the item again, even though the original request is still in the Tuxedo queue. A small number of extra restarts is benign, but at higher volumes, the unnecessary restarts can fill up the queue and block real requests.

The formula for a reasonable value for the Restart Period is: ((incoming requests per second) / (# of handlers)) * (average processing time per request) For example, if you have an incoming rate of twenty per second, and you have four handlers, each handler will be busy processing one item and will have four others waiting in the queue. A new item will have to wait for the currently processing item, plus the four enqueued items, before it will be processed. If each item takes 10 seconds to process, the new item will stay in " started" status for approximately 50 seconds before the handler works on it.

If it stays in " started" status longer, it's likely that the request to the handler has been lost, and the item should be restarted. Understanding Handler Parameters There are three generic process types that are the basis for all handler processes: •PSBRKHND — the publication broker handler. •PSPUBHND — the publication contractor handler. •PSSUBHND — the subscription contractor handler. The following parameters apply to all three process types. Min Instances Specifies the number of handler server processes started at boot time. Max Instances Specifies the maximum number of handler server processes that can be started or spawned.

Service Timeout Specifies the number of seconds a handlers waits for a service request before timing out. Service Timeouts are recorded in the TUXLOG and APPSRV. LOG. In the event of a timeout, the handler terminate itself and Tuxedo automatically restarts the process. Recycle Count Specifies the number of times the system executes each server before PeopleSoft intentionally terminates the process. Server processes must be intermittently recycled to clear buffer areas. The time required to recycle a server is

negligible—occurring in milliseconds. Recycle Count does not translate into a native Tuxedo parameter in the PSAPPSRV.

UBB file. Instead the value is stored in memory and is managed by PeopleSoft. Allowed Consec Service Failures This option allows for dynamic server process restarts in the event of service failures. To enable this option, enter a number greater than zero, and to disable this option enter 0. The default for this parameter is 2. The numerical value you enter is the number of consecutive service failures that will cause a recycle of the server process. This is a catchall error handling routine that allows a handler to terminate itself if it receives multiple, consecutive, fatal error messages from service routines.

Such errors should not occur consecutively, but if they do it indicates that the server process needs to be recycled or cleansed. A “ Retry” message appears when the number of service failures you specified occurs. Max Retries Specifies the maximum number of times the server should attempt to restart a failed action. This parameter prevents a bad item from continuously crashing a handler process — its counter is incremented when the handler sets the status to " working," but before it actually starts processing the item. Understanding Integration Broker Parameters The following parameters applies to the Integration Broker technology.

Min Message Size for Compression The Min Message Size for Compression parameter enables you to configure the threshold of message before the system compresses the message. Local Compression The integration engine compresses and base64 encodes messages destined for the PeopleSoft listening connector on its local integration gateway, based on a setting for

the application server domain in the PSAPPSRV. CFG file, which you can configure using the PSADMIN utility. The setting is a threshold message size, above which messages will be compressed. PSADMIN presents the setting as follows: Values for config section - Integration Broker

Min Message Size For Compression= 10000 Do you want to change any values (y/n)? [n]: The value is the message size in bytes; the default value is 10000 (10 KB). You can specify a setting of 0 to compress all messages. See Understanding Application Server Domain Parameters. Note. This setting has no effect on the compression of messages that the integration gateway sends using its target connectors. Information Set Profiling information for both Sync and Async processing External Configuration Set External Configuration = Y if you run the Pub/Sub processes on a different domain then where the appserver processes are run for PIA/ PORTAL.

This will enhance the Integration Broker performance for Asynchronous processing Minimum and Recommended Values. Specific application server tuning needs vary by customer site based on volume and server capacity. Requests for tuning issues and assistance should be addressed to Peoplesoft Consulting. However, some specific information is available below: PSAPPSRV should have a minimum of 3 instances booted when starting Pub/Sub. PSBRKDSP/HND settings should be sized up. A minumum of 3 instances should be used for all application messaging scenarios. For one particular customer I recommend increasing the PSBRKHND settings to 10/10.

Same with the PUB and SUB handler settings: set min/max of 10/10. Other customers have used as many as 20 instances for PSSUBHND. This is generally a tuning issue, and settings vary greatly from site to site. Recycle

count: For dispatchers this should always be 0. For Handlers this can be 0, or reduced based on need. A single handler is restarting itself after this number services (this is not the number messages, but the number of calls from the tuxedo service). Setting this too low can create performance problems. When a service recycles itelf, all requests must wait for the handler to come back up and re-submit.

It is generally recommended using 0 for this value. Otherwise a high number like 100, 000 is recommended unless memory problems are encountered in which case this value can be lowered. Restart Period. Since restart period controls how long before a started item will be resubmitted, dispatcher requests may be resubmitting themselves over and over again resulting in a higher queue number. This can be adjusted by changing Restart Period= 5 to a higher number. Customers will need to play with this and monitor results, but setting this to 120 would be better than the delivered 5 second interval, especially when using a lower value recycle count.