IBM WebSphere Application Server for z/OS, Version 8.0

Monitoring various types of applications

SA38-0662-01

Note

Before using this information, be sure to read the general information under “Notices” on page 49.

Compilation date: July 15, 2011

Contents

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information.

- To send comments on articles in the WebSphere Application Server Information Center
  1. Display the article in your Web browser and scroll to the end of the article.
  2. Click on the **Feedback** link at the bottom of the article, and a separate window containing an e-mail form appears.
  3. Fill out the e-mail form as instructed, and click on **Submit feedback** .
- To send comments on PDF books, you can e-mail your comments to: **wasdoc@us.ibm.com** or fax them to 919-254-5250.

  Be sure to include the document name and number, the WebSphere Application Server version you are using, and, if applicable, the specific page, table, or figure number on which you are commenting.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Changes to serve you more quickly

Print sections directly from the information center navigation

PDF books are provided as a convenience format for easy printing, reading, and offline use. The information center is the official delivery format for IBM WebSphere Application Server documentation. If you use the PDF books primarily for convenient printing, it is now easier to print various parts of the information center as needed, quickly and directly from the information center navigation tree.

To print a section of the information center navigation:

1. Hover your cursor over an entry in the information center navigation until the **Open Quick Menu** icon is displayed beside the entry.
2. Right-click the icon to display a menu for printing or searching your selected section of the navigation tree.
3. If you select **Print this topic and subtopics** from the menu, the selected section is launched in a separate browser window as one HTML file. The HTML file includes each of the topics in the section, with a table of contents at the top.
4. Print the HTML file.

For performance reasons, the number of topics you can print at one time is limited. You are notified if your selection contains too many topics. If the current limit is too restrictive, use the feedback link to suggest a preferable limit. The feedback link is available at the end of most information center pages.

Under construction!

The Information Development Team for IBM WebSphere Application Server is changing its PDF book delivery strategy to respond better to user needs. The intention is to deliver the content to you in PDF format more frequently. During a temporary transition phase, you might experience broken links. During the transition phase, expect the following link behavior:

- Links to Web addresses beginning with http:// work
- Links that refer to specific page numbers within the same PDF book work
- The remaining links will *not* work. You receive an error message when you click them

Thanks for your patience, in the short term, to facilitate the transition to more frequent PDF book updates.

Chapter 1. Monitoring Dynamic caching

This page provides a starting point for finding information about the dynamic cache service, which improves performance by caching the output of servlets, commands, web services, and JavaServer Pages (JSP) files.

Dynamic caching features include replication of cache entries, cache disk offload, Edge-Side Include caching, web services, and external caching. Use external caching to control caches outside of the application server.

Displaying cache information

Use this task to monitor the activity of the dynamic cache service.

About this task

The dynamic cache monitor is an installable web application that displays simple cache statistics, cache entries, and cache policy information for servlet cache instances.

The cache monitor provides information about the cache in the servant to which your browser connects to interact with the monitor. In an environment with multiple servants, the cache monitor provides a partial view of caching activity.

Procedure

1. Use the administrative console to install the cache monitor application from the *app_server_root*/`installableApps` directory. The name of the application is `CacheMonitor.ear`. For more information about installing applications, refer to the Installing application files with the console topic. Install the cache monitor onto the application server that you want to monitor.

   The cache monitor application must be installed on the *default_host* (908*x*).
2. Configure the web container transport chain and host alias for the server with cache monitor installed.
   a. Add a host alias for the port your server is using. Click **Environment > Virtual hosts >** *host_type* **> Host aliases** and create a new **Host name** and **Port** to add to the list. Creating a host alias enables you to access the cache monitor using `http://`*your_host_name*`:`*your_port_number*`/cachemonitor`

   Tip: You can find the port number in the `SystemOut.log` file. Look for message `TCPC0001I` or `SRVE0171I`.

   Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.
3. Access the cache monitor using a web browser and the URL `http://`*your host_name*`:`*your port_number*`/cachemonitor`, where *your port_number* is the port associated with the host on which you installed the cache monitor application.
4. Verify the list of cache instances that are shown. For each cache instance, you can perform the following actions:

Tip: You must select the servlet cache instance that you want to monitor. If you do not use servlet cache instances by using <cache-instance> tags in your cachespec.xml file, all the content is in the **baseCache** instance.

- View the Statistics page and verify the cache configuration and cache data. Click **Reset Statistics** to reset the counters.
- View the Cache Policies page to see which cache policies are currently loaded in the dynamic cache. Click a template to view the cache ID rules for the template.
- View the Cache Contents page to examine the contents that are currently cached in memory.
- View the Edge Statistics page to view data about the current ESI processors configured for caching. Click **Refresh Statistics** to see the latest statistics or content from the ESI processors. Click **Reset Statistics** to reset the counters.
- View the Disk Offload page to view the disk configuration, statistics, and the content that is currently off-loaded from memory to disk.

When you are viewing contents on memory or disk, click a template to view all entries for that template, click a dependency ID to view all entries for the ID, or click the cache ID to view all the data that is cached for that entry.

5. Use the cache monitor to perform basic operations on data in a cache instance.
   Remove an entry from cache
   Click **Invalidate** when viewing a cache entry.
   Remove all entries for a certain dependency ID
   Click **Invalidate** when viewing entries for a dependency ID.
   Remove all entries for a certain template
   Click **Invalidate** when viewing entries for a template.
   Move an entry to the front of the Least Recently Used queue to avoid eviction
   Click **Refresh** when viewing a cache entry.
   Move an entry from disk to cache
   Click **Send to Memory** when viewing a cache entry on disk.
   Clear the entire contents of the cache
   Click **Clear Cache** while viewing statistics or contents.
   Clear the contents on the ESI processors
   Click **Clear Cache** while viewing ESI statistics or contents.
   Clear the contents of the disk cache
   Click **Clear Disk** while viewing disk contents.

Cache monitor

Cache monitor is an installable web application that provides a real-time view of the current state of dynamic cache. You use it to help verify that dynamic cache is operating as expected. The only way to manipulate the data in the cache is by using the cache monitor. It provides a GUI interface to manually change data.

Cache monitor provides information on the cache in the servant to which your browser connects to interact with the monitor. In an environment that has multiple servants, cache monitor provides a partial view of caching activity.

Cache monitor provides a way to:

- **Verify the configuration of dynamic cache**

  After you create multiple servlet cache instances in the administrative console, you can configure properties, including the maximum size of the cache and disk offload location on each cache instance, as well as advanced features such as controlling external caches. You can verify the configuration of the dynamic cache by viewing of the configured features and properties in the cache monitor.
- **Verify the cache policies**

  To cache an object, unique IDs must be generated for different invocations of that object. To create unique IDs for each object, provide rules for each cacheable object in the `cachespec.xml` file, found

inside the web module WEB-INF or enterprise bean META-INF directory. Refer to the cachespec.xml file topic for more information about cacheable objects. Each cacheable object can have multiple cache ID rules that run in sequence until either a rule returns a cache ID or no more rules remain. If none of the cache ID generation rules produce a valid cache ID, then the object is not cached. There can be multiple cachespec.xml files with multiple cache ID rules. With cache monitor, you can verify the policies of each object. You can also view all of the cache polices for each cache instance that is currently loaded in dynamic cache. This view is also convenient to verify that the cachespec.xml file was read by the dynamic cache without errors.

- **Monitor cache statistics**

  You can view the essential cache data, such as number of cache hits, cache misses, and number of entries in each cache instance. With this data, you can tune the cache configuration to improve the dynamic cache performance. For example, if the number of used entries is often high, and entries are being removed and recreated, consider increasing the maximum size of the cache or enabling disk offload.

- **Monitor the data flowing through the cache**

  Once a cacheable object is invoked, dynamic cache creates a cache entry for it that contains the output of the actions that are performed and metadata, such as time to live, sharing policy, and so on. Entries are distinguished by a unique ID string that is based on the rules specified in the cachespec.xml file for the particular object name. Objects with the same name might generate multiple cache IDs for different invocations, based on request parameters and attributes for each invocation. You can view of all the cache entries that are in the cache instance, based on the unique ID. You can also view the group of cache entries that share a common name (also known as template). Cache entries can also be grouped together by a dependency ID, which is used to invalidate the entire group of entries dependent on a common entity. Therefore, cache monitor also provides a view of the group of cache entries that share a common dependency ID.

  For each entry, cache monitor also displays metadata, such as time to live, priority and sharing-policy, and provides a view of the output that has been cached. This helps the customer to verify which pages have been cached, that the pages have been cached in the correct cache instance with the right attributes such as time to live, priority, and that the pages have the right content.

- **Monitor the data in the edge cache**

  Dynamic cache provides support to recognize the presence of an Edge Side Include (ESI) processor and to generate ESI include tags and appropriate cache policies for edge cacheable fragments. With the ESI processor, you can cache whole pages, as well as fragments, providing a higher cache hit ratio. There can be multiple ESI processors running on multiple hosts configured for caching.

  You can view a list of all ESI processes and their hosts that are enabled for caching. Select a host or a processor, and view the edge cache statistics for it and the current cache entries.

- **View the data offloaded to the disk**

  By default, when the number of cache entries reaches the configured limit for a given server, cache entries are removed, enabling new entries to enter the cache service. With disk offload, the removed cache entries are copied to disk for future access. You can view the content that is copied to disk that corresponds to the view of the contents cached in memory for each cache instance.

- **Manage the data in the cache**

  You can perform the following basic operations on the data in the cache:
  - Remove an entry from a cache instance
  - Remove all entries for a certain dependency ID
  - Remove all entries for a certain name (template)
  - Move an entry to the front of the least recently used queue to avoid removal of the cache entry
  - Move an entry from the disk to the memory within a cache instance
  - Clear the entire contents of the cache instance
  - Clear the contents of the disk for the cache instance

  With these operations, you can manually change the state of the cache without having to restart the server.

Edge cache statistics

Cache monitor provides a view of the edge cache statistics.

The following statistics are available:

- **ESI Processors**. The number of processes that are configured as edge caches.
- **Number of Edge Cached Entries**. The number of entries that are currently cached on all edge servers and processes.
- **Cache Hits**. The number of requests that match entries on edge servers.
- **Cache Misses By URL.** A cache policy does not exist on the edge server for the requested template.

  Note:
  - The initial ESI request for a template that has a cache policy on WebSphere® Application Server results in a miss.
  - Every request for a template that does not have a cache policy on WebSphere Application Server results in a miss by URL on the edge server.
- **Cache Misses By Cache ID**. The policy for the requested template exists on the edge server, and a cache ID is created, based on the ID rules and the request attributes, but the cache entry for this ID does not exist.

  Note: If the policy exists on the edge server for the requested template, but a cache ID match is not found, based on the ID rules and the request attributes, it is not treated as a cache miss.
- **Cache Timeouts**. The number of entries that are removed from the edge cache based on the timeout value.
- **Evictions**. The number of entries that are removed from the edge cache due to invalidations received from WebSphere Application Server.

Dynamic cache MBean statistics

The dynamic cache service provides an MBean interface to access cache statistics.

Access cache statistics with the MBean interface, using JACL

- Obtain the MBean identifier with the **queryNames** command, for example:

  ```
  $AdminControl queryNames type=DynaCache,*  // Returns a list of the available dynamic cache MBeans
  ```

  Select your dynamic cache MBean and run the following command:

  ```
  set mbean <dynamic_cache_mbean>
  ```
- Retrieve the names of the available cache statistics:

  ```
  $AdminControl invoke $mbean getCacheStatisticNames
  ```
- Retrieve the names of the available cache instances:

  ```
  $AdminControl invoke $mbean getCacheInstanceNames
  ```
- Retrieve all of the available cache statistics for the base cache instance:

  ```
  $AdminControl invoke $mbean getAllCacheStatistics
  ```
- Retrieve all of the available cache statistics for the named cache instance:

  ```
  $AdminControl invoke $mbean getAllCacheStatistics "services/cache/servletInstance_4"
  ```
- Retrieve cache statistics that are specified by the names array for the base cache instance:

  ```
  $AdminControl invoke $mbean getCacheStatistics
   {"DiskCacheSizeInMB ObjectsReadFromDisk4000K RemoteObjectMisses"}
  ```

  Note: This command should all be entered on one line. It is broken here for printing purposes.
- Retrieve cache statistics that are specified by the names array for the named cache instance:

  ```
  $AdminControl invoke $mbean getCacheStatistics
   {services/cache/servletInstance_4 "ExplicitInvalidationsLocal CacheHits"}
  ```

  Attention: This command should all be entered on one line. It is broken here for printing purposes.

- Retrieve all the cache IDs in memory for the named cache instance that matches the specified regular expression:

  ```
  $AdminControl invoke $mbean getCacheIDsInMemory {services/cache/servletInstance_4 \S}
  ```

- Retrieve all cache IDs on disk for the named cache instance that matches the specified regular expression:

  ```
  $AdminControl invoke $mbean getCacheIDsOnDisk {services/cache/servletInstance_4 \S}
  ```

- Retrieves the CacheEntry, which holds metadata information for the cache ID:

  ```
  $AdminControl invoke $mbean getCacheEntry {services/cache/servletInstance_4 cache_id_1}
  ```

- Invalidates all cache entries that match the pattern-mapped cache IDs in the named cache instance and all cache entries dependent upon the matched entries in the instance:

  ```
  $AdminControl invoke $mbean invalidateCacheIDs {services/cache/servletInstance_4 cache_id_1 true}
  ```

Dynamic cache PMI counter definitions

The dynamic cache statistics interface is defined as WSDynamicCacheStats under the `com\ibm\websphere\pmi\stat` package.

Dynamic cache statistics are structured as follows in the Performance Monitoring Infrastructure (PMI) tree:

```
__Dynamic Caching+
  |
  |__<Servlet: instance_1>
    |__Templates+
      |__<template_1>
      |__<template_2>
    |__Disk+
      |__<Disk Offload Enabled>
  |
  |__<Object: instance_2>
    |__Object Cache+
      |__<Counters>
+ indicates logical group
```

StatDescriptor locates and accesses particular statistics in the PMI tree. For example:

1. StatDescriptor to represent statistics for cache servlet: `instance_1 templates group template_1: new StatDescriptor (new String[] {WSDynamicCacheStats.NAME, "Servlet: instance1", WSDynamicCacheStats.TEMPLATE_GROUP, "template_1"});`
2. StatDescriptor to represent statistics for cache servlet: `instance_1 disk group Disk Offload Enabled: new StatDescriptor (new String[] {WSDynamicCacheStats.NAME, "Servlet: instance_1", WSDynamicCacheStats.DISK_GROUP, WSDynamicCacheStats.DISK_OFFLOAD_ENABLED});`
3. StatDescriptor to represent statistics for cache object: `instance2 object cache group Counters: new StatDescriptor (new String[] {WSDynamicCacheStats.NAME, "Object: instance_2", WSDynamicCacheStats.OBJECT_GROUP, WSDynamicCacheStats.OBJECT_COUNTERS});`

Important: Cache instance names are prepended with cache type ("Servlet: " or "Object: ").

Counter definitions for Servlet Cache

Name of PMI statistics	Path	Description	Version
WSDynamicCacheStats. ObjectsOnDisk	`WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED`	The current number of cache entries on disk.	6.1

Name of PMI statistics	Path	Description	Version
WSDynamicCacheStats. HitsOnDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of requests for cacheable objects that are served from disk.	6.1
WSDynamicCacheStats. ExplicitInvalidations FromDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of explicit invalidations resulting in the removal of entries from disk.	6.1
WSDynamicCacheStats. TimeoutInvalidations FromDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of disk timeouts.	6.1
WSDynamicCacheStats PendingRemoval FromDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of pending entries that are to be removed from disk.	6.1
WSDynamicCacheStats. DependencyIdsOnDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of dependency ID that are on disk.	6.1
WSDynamicCacheStats. DependencyIdsBuffered ForDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of dependency IDs that are buffered for the disk.	6.1
WSDynamicCacheStats. DependencyIds OffloadedToDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of dependency IDs that are offloaded to disk.	6.1
WSDynamicCacheStats. DependencyIdBased InvalidationsFromDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of dependency ID-based invalidations.	6.1

Name of PMI statistics	Path	Description	Version
WSDynamicCacheStats. TemplatesOnDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of templates that are on disk.	6.1
WSDynamicCacheStats. TemplatesBuffered ForDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of templates that are buffered for the disk.	6.1
WSDynamicCacheStats. TemplatesOffloaded ToDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of templates that are offloaded to disk.	6.1
WSDynamicCacheStats. TemplateBased InvalidationsFromDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of template-based invalidations.	6.1
WSDynamicCacheStats. GarbageCollector InvalidationsFromDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of garbage collector invalidations resulting in the removal of entries from disk cache due to high threshold has been reached.	6.1
WSDynamicCacheStats. OverflowInvalidations FromDisk	WSDynamicCacheStats.NAME - "Servlet: cache_instance_1 " - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of invalidations resulting in the removal of entries from disk due to exceeding the disk cache size or disk cache size in GB limit.	6.1

Counter definitions for Object Cache

Name of PMI statistics	Path	Description	Version
WSDynamicCacheStats. ObjectsOnDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of cache entries on disk.	6.1

Name of PMI statistics	Path	Description	Version
WSDynamicCacheStats. HitsOnDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of requests for cacheable objects that are served from disk.	6.1
WSDynamicCacheStats. ExplicitInvalidations FromDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of explicit invalidations resulting in the removal of entries from disk.	6.1
WSDynamicCacheStats. TimeoutInvalidations FromDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of disk timeouts.	6.1
WSDynamicCacheStats PendingRemoval FromDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of pending entries that are to be removed from disk.	6.1
WSDynamicCacheStats. DependencyIdsOnDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of dependency ID that are on disk.	6.1
WSDynamicCacheStats. DependencyIds BufferedForDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of dependency IDs that are buffered for the disk.	6.1
WSDynamicCacheStats. DependencyIds OffloadedToDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of dependency IDs that are offloaded to disk.	6.1
WSDynamicCacheStats. DependencyIdBased InvalidationsFromDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats.DISK_ OFFLOAD_ENABLED	The number of dependency ID-based invalidations.	6.1

Name of PMI statistics	Path	Description	Version
WSDynamicCacheStats. TemplatesOnDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of templates that are on disk.	6.1
WSDynamicCacheStats. TemplatesBuffered ForDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP / -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The current number of templates that are buffered for the disk.	6.1
WSDynamicCacheStats. TemplatesOffloaded ToDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of templates that are offloaded to disk.	6.1
WSDynamicCacheStats. TemplateBasedInvalidations FromDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of template-based invalidations.	6.1
WSDynamicCacheStats. GarbageCollector InvalidationsFromDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of garbage collector invalidations resulting in the removal of entries from disk cache due to high threshold has been reached.	6.1
WSDynamicCacheStats. OverflowInvalidations FromDisk	WSDynamicCacheStats.NAME - "Object: cache_instance_2" - WSDynamicCacheStats. DISK_GROUP -" WSDynamicCacheStats. DISK_OFFLOAD_ENABLED	The number of invalidations resulting in the removal of entries from disk due to exceeding the disk cache size or disk cache size in GB limit.	6.1

Tuning dynamic cache with the cache monitor

Use this task to interpret cache monitor statistics to improve the performance of the dynamic cache service.

Before you begin

Verify that dynamic cache is enabled and that the cache monitor application is installed on your application server.

About this task

See the Displaying cache information topic in the *Administering applications and their environment* PDF to configure the cache monitor application.

Use the cache monitor to watch cache hits versus misses. By comparing these two values, you can determine how much dynamic cache is helping your application, and if you can take any additional steps to further improve performance and decrease the cost of processing for your application server.

Procedure

1. Start cache monitor and click on **Cache Statistics**. You can view the following cache statistics:

Cache statistic	Description
Cache Size	The maximum number of entries that the cache can hold.
Used Entries	The number of cache entries used.
Cache Hits	The number of request responses that are served from the cache.
Cache Misses	The number of request responses that are cacheable but cannot be served from the cache.
LRU Evictions	The number of cache entries removed to make room for new cache entries.
Explicit Removals	The number of cache entries removed or invalidated from the cache based on cache policies or were deleted from the cache through the cache monitor.

2. You can also view the following cache configuration values:

Cache configuration value	Description
Default priority	Specifies the default priority for all cache entries. Lower priority entries are moved from the cache before higher priority entries when the cache is full. You can specify the priority for individual cache entries in the cache policy.
Servlet Caching Enabled	If servlet caching is enabled, results from servlets and JavaServer Pages (JSP) files are cached. See the *Administering applications and their environment* PDF for more information.
Disk Offload Enabled	Specifies if entries that are being removed from the cache are saved to disk. See the *Administering applications and their environment* PDF for more information.

3. Wait for the application server to add data to the cache. You want the number of used cache entries in the cache monitor to be as high as it can go. When the number of used entries is at its highest, the cache can serve responses to as many requests as possible.
4. When the cache has a high number of used entries, reset the statistics. Watch the number of cache hits versus cache misses. If the number of hits is far greater than the number of misses, your cache configuration is optimal. You do not need to take any further actions. If you find a higher number of misses with a lower number of hits, the application server is working hard to generate responses instead of serving the request using a cached value. The application server might be making database queries, or running logic to respond to the requests.
5. If you have a large number of cache misses, increase the number of cache hits by improving the probability that a request can be served from the cache.

To improve the number of cache hits, you can increase the cache size or configure additional cache policies. See the *Administering applications and their environment* PDF for more information to increase the cache size and to configure cache policies.

Results

By using the cache monitor application, you optimized the performance of the dynamic cache service.

What to do next

See the *Administering applications and their environment* PDF for more information about the dynamic cache.

Chapter 2. OSGi Applications PMI counters

These OSGi Applications statistics are part of the performance monitoring infrastructure (PMI), which provides server-side monitoring and a client-side API to retrieve performance information. For OSGi applications, PMI counters are available for services and for bundle methods.

To see lists of available PMI counters, use the administrative console to navigate to **Monitoring and Tuning** > **Performance Monitoring Infrastructure (PMI)** > ***resource_name*** > **[General Properties] Custom**.

Note: The OSGi Applications PMI counters are only included in the lists if an OSGi application is installed.

Table 1. Service statistics.

This table lists the counters for OSGi Applications services

Name	Key	ID	Description	Granularity	Type	Level	Overhead
Service invocations	osgiAppModule. serviceInvocations	100	The number of invocations of this service	per bundle	CountStatistic	basic	low
Service response time	osgiAppModule. serviceResponseTime	101	The average response time of this service	per bundle	TimeStatistic	basic	medium
Service method invocations	osgiAppModule. serviceMethodInvocations	102	The number of invocations of this service method	per method	CountStatistic	extended	medium
Service method response time	osgiAppModule. serviceMethodResponseTime	103	The average response time of this service method	per method	TimeStatistic	extended	high

Table 2. Bean statistics.

This table lists the counters for OSGi Applications bundle methods

Name	Key	ID	Description	Granularity	Type	Level	Overhead
Bundle method invocations	osgiAppModule. bundleMethodInvocations	200	The number of invocations of this bundle method	per method	CountStatistic	all	high
Bundle method response time	osgiAppModule. bundleMethodResponseTime	201	The average response time of this bundle method	per method	TimeStatistic	all	maximum

Chapter 3. Monitoring Session Initiation Protocol (SIP) applications

This page provides a starting point for finding information about SIP applications, which are Java programs that use at least one Session Initiation Protocol (SIP) servlet written to the JSR 116 specification.

SIP is used to establish, modify, and terminate multimedia IP sessions including IP telephony, presence, and instant messaging.

Monitoring SIP applications

SIP PMI counters

The Session Initiation Protocol (SIP) provides the following counters in the WebSphere Performance Monitoring Infrastructure (PMI) to monitor the performance of SIP.

Counter definitions

Table 3. SIP container module.

This table lists the SIP container module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Incoming traffic	incoming.traffic	Average number of messages handled by the container calculated over configurable period	Per server	CountStatistic	All	Low	3
New SIP App sessions	new.sip.app.session	Average number of new SIP application sessions created in the container and calculated over configurable period	Per server	CountStatistic	All	Low	4
Response Time	response.time	The average amount of time that it takes between when a message gets into the container and when a response is sent from the container.	Per server	CountStatistic	All	Low	5
Queue Size	queue.size	Size of the invoke queue in WebSphere Application Server	Per server	CountStatistic	All	Low	6

Table 4. Session module.

This table lists the session module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of active SIP sessions	active.sip.sessions	The number of SIP sessions belongs to each application	Per application	CountStatistic	All	Low	11
Number of active SIP application sessions	active.sip.app.sessions	The number of SIP application sessions belongs to each application	Per application	CountStatistic	All	Low	12

Table 5. Inbound request module.

This table lists the inbound request module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound NOT SIP STANDARD requests	inbound.request.other	The number of inbound NOT SIP STANDARD requests that belong to each application	Per application	CountStatistic	All	Low	60
Number of Inbound REGISTER requests	inbound.request.register	The number of inbound REGISTER requests that belong to each application	Per application	CountStatistic	All	Low	61
Number of Inbound INVITE requests	inbound.request.invite	The number of inbound INVITE requests that belong to each application	Per application	CountStatistic	All	Low	62
Number of Inbound ACK requests	inbound.request.ack	The number of Inbound ACK requests that belong to each application	Per application	CountStatistic	All	Low	63
Number of Inbound OPTIONS requests	inbound.request.options	The number of Inbound OPTIONS requests that belong to each application	Per application	CountStatistic	All	Low	64
Number of Inbound BYE requests	inbound.request.bye	The number of Inbound BYE requests that belong to each application	Per application	CountStatistic	All	Low	65
Number of Inbound CANCEL requests	inbound.request.cancel	The number of Inbound CANCEL requests that belong to each application	Per application	CountStatistic	All	Low	66
Number of Inbound PRACK requests	inbound.request.prack	The number of Inbound PRACK requests that belong to each application	Per application	CountStatistic	All	Low	67
Number of Inbound INFO requests	inbound.request.info	The number of Inbound INFO requests that belong to each application	Per application	CountStatistic	All	Low	68
Number of Inbound SUBSCRIBE requests	inbound.request.subscribe	The number of Inbound SUBSCRIBE requests that belong to each application	Per application	CountStatistic	All	Low	69
Number of Inbound NOTIFY requests	inbound.request.notify	The number of Inbound NOTIFY requests that belong to each application	Per application	CountStatistic	All	Low	70
Number of Inbound MESSAGE requests	inbound.request.message	The number of Inbound MESSAGE requests that belong to each application	Per application	CountStatistic	All	Low	71
Number of Inbound PUBLISH requests	inbound.request.publish	The number of Inbound PUBLISH requests that belong to each application	Per application	CountStatistic	All	Low	72
Number of Inbound REFER requests	inbound.request.refer	The number of Inbound REFER requests that belong to each application	Per application	CountStatistic	All	Low	73

Table 5. Inbound request module (continued).

This table lists the inbound request module.

Number of Inbound UPDATE requests	inbound.request.update	The number of Inbound UPDATE requests that belong to each application	Per application	CountStatistic	All	Low	74

Table 6. Inbound info response module.

This table lists the inbound info response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 100 responses	inbound.response.info.100	The number of Inbound 100 (Trying) responses that belong to each application	Per application	CountStatistic	All	Low	1100
Number of Inbound 180 responses	inbound.response.info.180	The number of Inbound 180 (Ringing) responses that belong to each application	Per application	CountStatistic	All	Low	1180
Number of Inbound 181 responses	inbound.response.info.181	The number of Inbound 181 (Call being forwarded) responses that belong to each application	Per application	CountStatistic	All	Low	1181
Number of Inbound 182 responses	inbound.response.info.182	The number of Inbound 182 (Call Queued) responses that belong to each application	Per application	CountStatistic	All	Low	1182
Number of Inbound 183 responses	inbound.response.info.183	The number of Inbound 183 (Session Progress) responses that belong to each application	Per application	CountStatistic	All	Low	1183

Table 7. Inbound success response module.

This table lists the inbound success response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of successful Inbound 200 responses	inbound.response.success.200	The number of Inbound 200 (OK) responses that belong to each application	Per application	CountStatistic	All	Low	1200
Number of successful Inbound 202 responses	inbound.response.success.202	The number of Inbound 202 (Accepted) responses that belong to each application	Per application	CountStatistic	All	Low	1202

Table 8. Inbound redirect response module.

This table lists the inbound redirect response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 300 responses	inbound.response.redirect.300	The number of Inbound 300 (Multiple choices) responses that belong to each application	Per application	CountStatistic	All	Low	1300
Number of Inbound 301 responses	inbound.response.redirect.301	The number of Inbound 301 (Moved Permanently) responses that belong to each application	Per application	CountStatistic	All	Low	1301
Number of Inbound 302 responses	inbound.response.redirect.302	The number of Inbound 302 (Moved Temporarily) responses that belong to each application	Per application	CountStatistic	All	Low	1302
Number of Inbound 305 responses	inbound.response.redirect.305	The number of Inbound 305 (Use Proxy) responses that belong to each application	Per application	CountStatistic	All	Low	1305
Number of Inbound 380 responses	inbound.response.redirect.380	The number of Inbound 380 (Alternative Service) responses that belong to each application	Per application	CountStatistic	All	Low	1380

Table 9. Inbound fail response module.

This table lists the inbound fail response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 400 responses	inbound.response.fail.400	The number of Inbound 400 (Bad Request) responses that belong to each application	Per application	CountStatistic	All	Low	1400
Number of Inbound 401 responses	inbound.response.fail.401	The number of Inbound 401 (Unauthorized) responses that belong to each application	Per application	CountStatistic	All	Low	1401
Number of Inbound 402 responses	inbound.response.fail.402	The number of Inbound 402 (Payment Required) responses that belong to each application	Per application	CountStatistic	All	Low	1402
Number of Inbound 403 responses	inbound.response.fail.403	The number of Inbound 403 (Forbidden) responses that belong to each application	Per application	CountStatistic	All	Low	1403
Number of Inbound 404 responses	inbound.response.fail.404	The number of Inbound 404 (Not Found) responses that belong to each application	Per application	CountStatistic	All	Low	1404
Number of Inbound 405 responses	inbound.response.fail.405	The number of Inbound 405 (Method Not Allowed) responses that belong to each application	Per application	CountStatistic	All	Low	1405
Number of Inbound 406 responses	inbound.response.fail.406	The number of Inbound 406 (Not Acceptable) responses that belong to each application	Per application	CountStatistic	All	Low	1406
Number of Inbound 407 responses	inbound.response.fail.407	The number of Inbound 407 (Proxy Authentication Required) responses that belong to each application	Per application	CountStatistic	All	Low	1407
Number of Inbound 408 responses	inbound.response.fail.408	The number of Inbound 408 (Request Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	1408
Number of Inbound 410 responses	inbound.response.fail.410	The number of Inbound 410 (Gone) responses that belong to each application	Per application	CountStatistic	All	Low	1410
Number of Inbound 413 responses	inbound.response.fail.413	The number of Inbound 413 (Request Entity Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	1413

Table 9. Inbound fail response module (continued).

This table lists the inbound fail response module.

Number of Inbound 414 responses	inbound.response.fail.414	The number of Inbound 414 (Request URI Too Long) responses that belong to each application	Per application	CountStatistic	All	Low	1414
Number of Inbound 415 responses	inbound.response.fail.415	The number of Inbound 415 (Unsupported Media Type) responses that belong to each application	Per application	CountStatistic	All	Low	1415
Number of Inbound 416 responses	inbound.response.fail.416	The number of Inbound 416 (Unsupported URI Scheme) responses that belong to each application	Per application	CountStatistic	All	Low	1416
Number of Inbound 420 responses	inbound.response.fail.420	The number of Inbound 420 (Bad Extention) responses that belong to each application	Per application	CountStatistic	All	Low	1420
Number of Inbound 421 responses	inbound.response.fail.421	The number of Inbound 421 (Extention Required) responses that belong to each application	Per application	CountStatistic	All	Low	1421
Number of Inbound 423 responses	inbound.response.fail.423	The number of Inbound 423 (Interval Too Brief) responses that belong to each application	Per application	CountStatistic	All	Low	1423
Number of Inbound 480 responses	inbound.response.fail.480	The number of Inbound 480 (Temporarly Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	1480
Number of Inbound 481 responses	inbound.response.fail.481	The number of Inbound 481 (Call Leg Done) responses that belong to each application	Per application	CountStatistic	All	Low	1481
Number of Inbound 482 responses	inbound.response.fail.482	The number of Inbound 482 (Loop Detected) responses that belong to each application	Per application	CountStatistic	All	Low	1482
Number of Inbound 483 responses	inbound.response.fail.483	The number of Inbound 483 (Too Many Hops) responses that belong to each application	Per application	CountStatistic	All	Low	1483
Number of Inbound 484 responses	inbound.response.fail.484	The number of Inbound 484 (Address Incomplete) responses that belong to each application	Per application	CountStatistic	All	Low	1484

Table 9. Inbound fail response module (continued).

This table lists the inbound fail response module.

Number of Inbound 485 responses	inbound.response.fail.485	The number of Inbound 485 (Ambiguous) responses that belong to each application	Per application	CountStatistic	All	Low	1485
Number of Inbound 486 responses	inbound.response.fail.486	The number of Inbound 486 (Busy Here) responses that belong to each application	Per application	CountStatistic	All	Low	1486
Number of Inbound 487 responses	inbound.response.fail.487	The number of Inbound 487 (Request Terminated) responses that belong to each application	Per application	CountStatistic	All	Low	1487
Number of Inbound 488 responses	inbound.response.fail.488	The number of Inbound 488 (Not Acceptable Here) responses that belong to each application	Per application	CountStatistic	All	Low	1488
Number of Inbound 491 responses	inbound.response.fail.491	The number of Inbound 491 (Request Pending) responses that belong to each application	Per application	CountStatistic	All	Low	1491
Number of Inbound 493 responses	inbound.response.fail.493	The number of Inbound 493 (Undecipherable) responses that belong to each application	Per application	CountStatistic	All	Low	1493

Table 10. Inbound server fail response module.

This table lists the inbound server fail response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 500 responses	inbound.response.serverFail.500	The number of Inbound 500 (Server Internal Error) responses that belong to each application	Per application	CountStatistic	All	Low	1500
Number of Inbound 501 responses	inbound.response.serverFail.501	The number of Inbound 501 (Not Implemented) responses that belong to each application	Per application	CountStatistic	All	Low	1501
Number of Inbound 502 responses	inbound.response.serverFail.502	The number of Inbound 502 (Bad Gateway) responses that belong to each application	Per application	CountStatistic	All	Low	1502
Number of Inbound 503 responses	inbound.response.serverFail.503	The number of Inbound 503 (Service Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	1503
Number of Inbound 504 responses	inbound.response.serverFail.504	The number of Inbound 504 (Server Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	1504
Number of Inbound 505 responses	inbound.response.serverFail.505	The number of Inbound 505 (Version Not Supported) responses that belong to each application	Per application	CountStatistic	All	Low	1505
Number of Inbound 513 responses	inbound.response.serverFail.513	The number of Inbound 513 (Message Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	1513

Table 11. Inbound global fail response module.

This table lists the inbound global fail response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 600 responses	inbound.response.globalFail.600	The number of Inbound 600 (Busy Everywhere) responses that belong to each application	Per application	CountStatistic	All	Low	1600
Number of Inbound 603 responses	inbound.response.globalFail.603	The number of Inbound 603 (Decline) responses that belong to each application	Per application	CountStatistic	All	Low	1603
Number of Inbound 604 responses	inbound.response.globalFail.604	The number of Inbound 604 (Does Not Exit Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	1604
Number of Inbound 606 responses	inbound.response.globalFail.606	The number of Inbound 606 (Not Acceptable Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	1606

Table 12. Outbound request module.

This table lists the outbound request module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound NOT SIP STANDARD requests	outbound.request.other	The number of Outbound NOT SIP STANDARD requests that belong to each application	Per application	CountStatistic	All	Low	80
Number of Outbound REGISTER requests	outbound.request.register	The number of Outbound REGISTER requests that belong to each application	Per application	CountStatistic	All	Low	81
Number of Outbound INVITE requests	outbound.request.invite	The number of Outbound INVITE requests that belong to each application	Per application	CountStatistic	All	Low	82
Number of Outbound ACK requests	outbound.request.ack	The number of Outbound ACK requests that belong to each application	Per application	CountStatistic	All	Low	83
Number of Outbound OPTIONS requests	outbound.request.options	The number of Outbound OPTIONS requests that belong to each application	Per application	CountStatistic	All	Low	84
Number of Outbound BYE requests	outbound.request.bye	The number of Outbound BYE requests that belong to each application	Per application	CountStatistic	All	Low	85
Number of Outbound CANCEL requests	outbound.request.cancel	The number of Outbound CANCEL requests that belong to each application	Per application	CountStatistic	All	Low	86
Number of Outbound PRACK requests	outbound.request.prack	The number of Outbound PRACK requests that belong to each application	Per application	CountStatistic	All	Low	87
Number of Outbound INFO requests	outbound.request.info	The number of Outbound INFO requests that belong to each application	Per application	CountStatistic	All	Low	88
Number of Outbound SUBSCRIBE requests	outbound.request.subscribe	The number of Outbound SUBSCRIBE requests that belong to each application	Per application	CountStatistic	All	Low	89
Number of Outbound NOTIFY requests	outbound.request.notify	The number of Outbound NOTIFY requests that belong to each application	Per application	CountStatistic	All	Low	90
Number of Outbound MESSAGE requests	outbound.request.message	The number of Outbound MESSAGE requests that belong to each application	Per application	CountStatistic	All	Low	91
Number of Outbound PUBLISH requests	outbound.request.publish	The number of Outbound PUBLISH requests that belong to each application	Per application	CountStatistic	All	Low	92
Number of Outbound REFER requests	outbound.request.refer	The number of Outbound REFER requests that belong to each application	Per application	CountStatistic	All	Low	93

Table 12. Outbound request module (continued).

This table lists the outbound request module.

Number of Outbound UPDATE requests	outbound.request.update	The number of Outbound UPDATE requests that belong to each application	Per application	CountStatistic	All	Low	94

Table 13. Outbound info response module.

This table lists the outbound info response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 100 responses	outbound.response.info.100	The number of Outbound 100 (Trying) responses that belong to each application	Per application	CountStatistic	All	Low	2100
Number of Outbound 180 responses	outbound.response.info.180	The number of Outbound 180 (Ringing) responses that belong to each application	Per application	CountStatistic	All	Low	2180
Number of Outbound 181 responses	outbound.response.info.181	The number of Outbound 181 (Call being forwarded) responses that belong to each application	Per application	CountStatistic	All	Low	2181
Number of Outbound 182 responses	outbound.response.info.182	The number of Outbound 182 (Call Queued) responses that belong to each application	Per application	CountStatistic	All	Low	2182
Number of Outbound 183 responses	outbound.response.info.183	The number of Outbound 183 (Session Progress) responses that belong to each application	Per application	CountStatistic	All	Low	2183

Table 14. Outbound success response module.

This table lists the outbound success response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 200 responses	outbound.response.success.200	The number of Outbound 200 (OK) responses that belong to each application	Per application	CountStatistic	All	Low	2200
Number of Outbound 202 responses	outbound.response.success.202	The number of Outbound 202 (Accepted) responses that belong to each application	Per application	CountStatistic	All	Low	2202

Table 15. Outbound redirect response module.

This table lists the outbound redirect response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 300 responses	outbound.response.redirect.300	The number of Outbound 300 (Multiple choices) responses that belong to each application	Per application	CountStatistic	All	Low	2300
Number of Outbound 301 responses	outbound.response.redirect.301	The number of Outbound 301 (Moved Permanently) responses that belong to each application	Per application	CountStatistic	All	Low	2301
Number of Outbound 302 responses	outbound.response.redirect.302	The number of Outbound 302 (Moved Temporarily) responses that belong to each application	Per application	CountStatistic	All	Low	2302
Number of Outbound 305 responses	outbound.response.redirect.305	The number of Outbound 305 (Use Proxy) responses that belong to each application	Per application	CountStatistic	All	Low	2305
Number of Outbound 380 responses	outbound.response.redirect.380	The number of Outbound 380 (Alternative Service) responses that belong to each application	Per application	CountStatistic	All	Low	2380

Table 16. Outbound fail response module.

This table lists the outbound fail response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 400 responses	outbound.response.fail.400	The number of Outbound 400 (Bad Request) responses that belong to each application	Per application	CountStatistic	All	Low	2400
Number of Outbound 401 responses	outbound.response.fail.401	The number of Outbound 401 (Unauthorized) responses that belong to each application	Per application	CountStatistic	All	Low	2401
Number of Outbound 402 responses	outbound.response.fail.402	The number of Outbound 402 (Payment Required) responses that belong to each application	Per application	CountStatistic	All	Low	2402
Number of Outbound 403 responses	outbound.response.fail.403	The number of Outbound 403 (Forbidden) responses that belong to each application	Per application	CountStatistic	All	Low	2403
Number of Outbound 404 responses	outbound.response.fail.404	The number of Outbound 404 (Not Found) responses that belong to each application	Per application	CountStatistic	All	Low	2404
Number of Outbound 405 responses	outbound.response.fail.405	The number of Outbound 405 (Method Not Allowed) responses that belong to each application	Per application	CountStatistic	All	Low	2405
Number of Outbound 406 responses	outbound.response.fail.406	The number of Outbound 406 (Not Acceptable) responses that belong to each application	Per application	CountStatistic	All	Low	2406
Number of Outbound 407 responses	outbound.response.fail.407	The number of Outbound 407 (Proxy Authentication Required) responses that belong to each application	Per application	CountStatistic	All	Low	2407
Number of Outbound 408 responses	outbound.response.fail.408	The number of Outbound 408 (Request Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	2408
Number of Outbound 410 responses	outbound.response.fail.410	The number of Outbound 410 (Gone) responses that belong to each application	Per application	CountStatistic	All	Low	2410
Number of Outbound 413 responses	outbound.response.fail.413	The number of Outbound 413 (Request Entity Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	2413

Table 16. Outbound fail response module (continued).

This table lists the outbound fail response module.

Number of Outbound 414 responses	outbound.response.fail.414	The number of Outbound 414 (Request URI Too Long) responses that belong to each application	Per application	CountStatistic	All	Low	2414
Number of Outbound 415 responses	outbound.response.fail.415	The number of Outbound 415 (Unsupported Media Type) responses that belong to each application	Per application	CountStatistic	All	Low	2415
Number of Outbound 416 responses	outbound.response.fail.416	The number of Outbound 416 (Unsupported URI Scheme) responses that belong to each application	Per application	CountStatistic	All	Low	2416
Number of Outbound 420 responses	outbound.response.fail.420	The number of Outbound 420 (Bad Extention) responses that belong to each application	Per application	CountStatistic	All	Low	2420
Number of Outbound 421 responses	outbound.response.fail.421	The number of Outbound 421 (Extention Required) responses that belong to each application	Per application	CountStatistic	All	Low	2421
Number of Outbound 423 responses	outbound.response.fail.423	The number of Outbound 423 (Interval Too Brief) responses that belong to each application	Per application	CountStatistic	All	Low	2423
Number of Outbound 480 responses	outbound.response.fail.480	The number of Outbound 480 (Temporarly Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	2480
Number of Outbound 481 responses	outbound.response.fail.481	The number of Outbound 481 (Call Leg Done) responses that belong to each application	Per application	CountStatistic	All	Low	2481
Number of Outbound 482 responses	outbound.response.fail.482	The number of Outbound 482 (Loop Detected) responses that belong to each application	Per application	CountStatistic	All	Low	2482
Number of Outbound 483 responses	outbound.response.fail.483	The number of Outbound 483 (Too Many Hops) responses that belong to each application	Per application	CountStatistic	All	Low	2483
Number of Outbound 484 responses	outbound.response.fail.484	The number of Outbound 484 (Address Incomplete) responses that belong to each application	Per application	CountStatistic	All	Low	2484

Table 16. Outbound fail response module (continued).

This table lists the outbound fail response module.

Number of Outbound 485 responses	outbound.response.fail.485	The number of Outbound 485 (Ambiguous) responses that belong to each application	Per application	CountStatistic	All	Low	2485
Number of Outbound 486 responses	outbound.response.fail.486	The number of Outbound 486 (Busy Here) responses that belong to each application	Per application	CountStatistic	All	Low	2486
Number of Outbound 487 responses	outbound.response.fail.487	The number of Outbound 487 (Request Terminated) responses that belong to each application	Per application	CountStatistic	All	Low	2487
Number of Outbound 488 responses	outbound.response.fail.488	The number of Outbound 488 (Not Acceptable Here) responses that belong to each application	Per application	CountStatistic	All	Low	2488
Number of Outbound 491 responses	outbound.response.fail.491	The number of Outbound 491 (Request Pending) responses that belong to each application	Per application	CountStatistic	All	Low	2491
Number of Outbound 493 responses	outbound.response.fail.493	The number of Outbound 493 (Undecipherable) responses that belong to each application	Per application	CountStatistic	All	Low	2493

Table 17. Outbound server fail response module.

This table lists the outbound server fail response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 500 responses	outbound.response.serverFail.500	The number of Outbound 500 (Server Internal Error) responses that belong to each application	Per application	CountStatistic	All	Low	2500
Number of Outbound 501 responses	outbound.response.serverFail.501	The number of Outbound 501 (Not Implemented) responses that belong to each application	Per application	CountStatistic	All	Low	2501
Number of Outbound 502 responses	outbound.response.serverFail.502	The number of Outbound 502 (Bad Gateway) responses that belong to each application	Per application	CountStatistic	All	Low	2502
Number of Outbound 503 responses	outbound.response.serverFail.503	The number of Outbound 503 (Service Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	2503
Number of Outbound 504 responses	outbound.response.serverFail.504	The number of Outbound 504 (Server Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	2504
Number of Outbound 505 responses	outbound.response.serverFail.505	The number of Outbound 505 (Version Not Supported) responses that belong to each application	Per application	CountStatistic	All	Low	2505
Number of Outbound 513 responses	outbound.response.serverFail.513	The number of Outbound 513 (Message Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	2513

Table 18. Outbound global fail response module.

This table lists the outbound global fail response module.

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 600 responses	outbound.response.globalFail.600	The number of Outbound 600 (Busy Everywhere) responses that belong to each application	Per application	CountStatistic	All	Low	2600
Number of Outbound 603 responses	outbound.response.globalFail.603	The number of Outbound 603 (Decline) responses that belong to each application	Per application	CountStatistic	All	Low	2603
Number of Outbound 604 responses	outbound.response.globalFail.604	The number of Outbound 604 (Does Not Exit Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	2604
Number of Outbound 606 responses	outbound.response.globalFail.606	The number of Outbound 606 (Not Acceptable Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	2606

Chapter 4. Monitoring Transactions

This page provides a starting point for finding information about Java Transaction API (JTA) support. Applications running on the server can use transactions to coordinate multiple updates to resources as one unit of work, such that all or none of the updates are made permanent.

The product provides advanced transactional capabilities to help application developers avoid custom coding. It provides support for the many challenges related to integrating existing software assets with a Java EE environment.

Configuring an application server to log heuristic reporting

You can configure the transaction service for an application server to indicate whether to log that the transaction service is about to commit a one-phase commit resource. This configuration is useful when you use last participant support, when there is an increased risk that a transaction has an heuristic outcome.

About this task

To enable an application server to log “about to commit one-phase resource” events from transactions that involve a one-phase commit resource and two-phase commit resources, use the administrative console to complete the following steps:

Procedure

1. From the navigation pane of the administrative console, click **Servers** > **Server types** > **WebSphere application servers** > ***server_name*** > **[Container Settings] Container Services** > **Transaction service**. The transaction service settings for the selected application server are displayed.
2. Select the Configuration tab to display the transaction-related properties that are defined in the configuration file.
3. Click **Enable logging for heuristic reporting**.
4. Click **OK**.
5. Stop, then restart, the application server.

Chapter 5. Monitoring web services

This page provides a starting point for finding information about web services.

Web services are self-contained, modular applications that can be described, published, located, and invoked over a network. They implement a services oriented architecture (SOA), which supports the connecting or sharing of resources and data in a very flexible and standardized manner. Services are described and organized to support their dynamic, automated discovery and reuse.

Monitoring the performance of web services applications

You can monitor the performance of a web service that is implemented in the WebSphere Application Server using Performance Monitoring Infrastructure (PMI) tooling.

About this task

You can use the Performance Monitoring Infrastructure (PMI) to measure the time required to process web services requests. To monitor the performance of a web services application, follow the steps in this task:

Procedure

1. Enable PMI services in an application server through the administrative console. For detailed information, see the topic on enabling PMI using the administrative console. Complete the required steps and the optional step to enable statistics by clicking **Custom** and choosing a component to enable for statistics.
2. Monitor performance with Tivoli® Performance Viewer. For detailed information, see the topic on monitoring performance with Tivoli Performance Viewer. In the left pane of the performance view, expand the host and server. Select **Web Services**. Run the web services client application.

Results

PMI provides detailed statistics that can help you gain clear insight into the runtime behavior and performance of web services. Performance counters enable you to see key performance data for each individual web service including:

- The number of requests dispatched to an implementation bean
- The number of requests dispatched with successful replies
- The average time in milliseconds to process full requests
- The average time in milliseconds between receiving the request and dispatching it to the bean
- The average time in milliseconds between the dispatch and receipt of a reply from the bean. This represents the time spent in business logic.
- The average time in milliseconds between the receipt of a reply from a bean to the return of a result to the client
- The average size of the SOAP request
- The average size of the SOAP reply

To read about other web services PMI counters, see the PMI data organization information.

What to do next

If you are having problems with your web services applications, read about problems and solutions in the troubleshooting web services information.

Web services performance best practices

Learn about best practices for the performance of web services applications.

Web services are developed and deployed based on standards provided by the Web Services for Java Platform, Enterprise Edition (Java EE) specification and the Java API for XML-Based Web Services (JAX-WS) and Java Architecture for XML Binding (JAXB) programming models, and is the mechanism used to access a web service. This article explains performance considerations for web services supported by this specification.

When you develop or deploy a web service, several artifacts are required, including a Web Services Description Language (WSDL) file. The WSDL file describes the format and syntax of the web service input and output SOAP messages. When a web service is implemented in the WebSphere Application Server runtime, the SOAP message is translated based on the Java EE request. The Java EE-based response is then translated back to a SOAP message.

The most critical performance consideration is the translation between the XML-based SOAP message and the Java object. Performance is high for a web service implementation in WebSphere Application Server, however, application design, deployment and tuning can be improved. See the information on monitoring the performance of Web services applications to learn more about analyzing and tuning Web services.

If you are using a web service application that was developed for a WebSphere Application Server version prior to Version 6, you can achieve better performance by running the **wsdeploy** command. The **wsdeploy** command regenerates web services artifact classes to increase the serialization and deserialization performance.

The wsdeploy command is supported by Java API for XML-based RPC (JAX-RPC) applications. The Java API for XML-Based Web Services (JAX-WS) programming model that is implemented by the application server does not support the wsdeploy command. If your web services application contains only JAX-WS endpoints, you do not need to run the wsdeploy command, as this command is used to process only JAX-RPC endpoints.

Basic considerations for a high-performance web services application

The following are basic considerations you should know when designing a web services application:
- Reduce the web services requests by using a few highly functional APIs, rather than several simple APIs.
- Design your WSDL file interface to limit the size and complexity of SOAP messages.
- Use the document/literal style argument when you generate the WSDL file.
- Leverage the caching capabilities offered for WebSphere Application Server.
- Test the performance of your web service.

Additional web services performance features that you can leverage
- In-process optimizations for web services to optimize the communication path between a web services client application and a web container that are located in the same application server process. For details about enabling this feature, see the web services client to web container optimized communication information.
- Access to web services over multiple transport protocols extends existing Java API for XML-based remote procedure call (JAX-RPC) capabilities to support non-SOAP bindings such as RMI/IIOP and JMS. These alternative transports can improve performance and quality of service aspects for web services. For more detailed information see the RMI-IIOP using JAX-RPC information.
- SOAP with Attachments API for Java (SAAJ) Version 1.2 provides a programming model for web services relative to JAX-RPC. The SAAJ API provides features to create and process SOAP requests using an XML API. SAAJ supports just-in-time parsing and other internal algorithms. For information about SAAJ or web services programming, see the SOAP with Attachments API for Java information.

  SAAJ 1.3 provides support for web services that are developed and implemented based on the Java API for XML Web Services (JAX-WS) programming model.

- The web services tooling generates higher performance custom deserializers for all JAX-RPC beans. Redeploying a V5.*x* application into the V6 runtime can decrease the processing time for large messages.
- Serialization and deserialization runtime is enhanced to cache frequently used serializers and deserializers. This can decrease the processing time for large messages.

IBM® provides considerable documentation and best practices for web services application design and development that details these items and more.

Tuning for SOAP

Learn how to tune the soap messages that you use with your web services.

About this task

SOAP is a lightweight protocol which provides a mechanism to use XML for exchanging structured and typed information between peers in a decentralized, distributed environment.

Procedure

- Specify noLocalCopies in servant.jvm.options (-Dcom.ibm.CORBA.iiop.noLocalCopies=1). This will enable passing of parameters by reference instead of by value. This only applies if you are exposing an enterprise bean as a web service. To learn more, see the object request broker service settings information.
- Make certain that all traces are disabled unless you are actively debugging a problem.
- When defining transaction policies for your application, specify TX_NOT_SUPPORTED and select local transactions. Local transactions perform better than global transactions because WebSphere is not required to coordinate commit scope over multiple resource managers.
- Avoid passing empty attributes or empty elements in SOAP messages. Do not include extraneous and unneeded data in SOAP messages. If you can use document/literal style web services invocation to batch requests into a single SOAP message, this is preferable to sending multiple individual SOAP messages. SOAP applications will perform better with smaller SOAP messages containing fewer XML elements and especially fewer XML attributes. The contents of SOAP messages must be serialized and parsed. These are expensive operations and should be minimized. In other words, it is preferable to send 1, 10KB message than 10, 1KB messages. However, very large messages (for example, over 200KB) might impact system resources like memory.
- You might need to increase the default Java heap size. SOAP and XML (DOM) are storage-intensive and small heap sizes can result in excessive Java garbage collection. We found a heapsize of 256M (the default) was optimal for most test cases in the laboratory. You can monitor garbage collection using the verbose:gc Java directive.
- Insure TCP/IP send/receive buffers are large enough to hold the bulk of the xml messages that will be sent.
- Consider using a Document Model rather than the RPC model. It provides complete control over the format of the XML but requires additional programming effort.
- When using RPC-style messages, try to send strings if possible.
- For Java API for XML-based RPC (JAX-RPC) web services, consider writing your own serializers and deserializers, avoiding reflection.

Appendix. Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root
Refers to the top directory for a WebSphere Application Server node.

The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.

The default varies based on node type. Common defaults are *configuration_root*/AppServer and *configuration_root*/DeploymentManager.

configuration_root
Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS®.

The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.

The default is /wasv8config/*cell_name*/*node_name*.

plug-ins_root
Refers to the installation root directory for Web Server Plug-ins.

profile_root
Refers to the home directory for a particular instantiated WebSphere Application Server profile.

Corresponding product variables are *server.root* and *user.install.root.*

In general, this is the same as *app_server_root*/profiles/*profile_name*. On z/OS, this will always be *app_server_root*/profiles/default because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root
Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.

The corresponding product variable is *smpe.install.root*.

The default is /usr/lpp/zWebSphere/V8R0.

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

APACHE INFORMATION. This information may include all or portions of information which IBM obtained under the terms and conditions of the Apache License Version 2.0, January 2004. The information may also consist of voluntary contributions made by many individuals to the Apache Software Foundation. For more information on the Apache Software Foundation, please see http://www.apache.org. You may obtain a copy of the Apache License at http://www.apache.org/licenses/LICENSE-2.0.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Intellectual Property & Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA
Attention: Information Requests

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. For a current list of IBM trademarks, visit the IBM Copyright and trademark information Web site (www.ibm.com/legal/copytrade.shtml).

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

Index