NCache Release Notes

Last Modified:

August 07, 2018

Table of Contents

Release Notes NCache 4.9 SP1

Tuesday, August 07, 2018

Introduction

In NCache 4.9 SP1, community edition has been discontinued and Professional edition has been brought back. There are some important bug fixes and a few new enhancements made in this release.

Enhancements and New Additions

Following are some enhancements made in this release:

1. **.NET Core based NCache server on Linux**
   NCache Server (.NET Core based) can now be hosted on Linux boxes. Separate installers for Linux (.tar.gz) are made available for download.

2. **60-Day Trial is back**
   The 60-day fully working trial is back. There are no performance limitations during evaluation period now.

3. **Professional Edition**
   Professional edition has the same features as they are in Open Source edition plus some extra management PowerShell commands. Both client and server installations have a free 60-day Trial.

4. **Docker Images for Linux and Windows Nano Server**
   Docker images for Linux and Windows Nano server are made available on Docker Hub. Docker files are also available on GitHub.

5. **Bridge queue counter**
   A perfmon counter is introduced to monitor number of items in bridge.

6. **Wildcard search in Tags API**
   Support for wildcard search in GetKeysByTag and GetByTag APIs is provided.

7. **Enable client side logs from API**
   Support is provided to enable client side logging from API through CacheInitParams interface.

8. **Subscription based licensing**
   Subscription based licensing has been introduced to support both cloud and on-premise deployments.

List of Bugs Fixed:

744169 FIX: Timeout on 'IN' predicate queries on large number of parameters.

744170 FIX: Requests timeout to clients continue to run on servers, hence, causing high CPU and memory issues.

744171 FIX: Thread count keeps increasing in inproc client cache.

744172 FIX: In case of client cache, items are fetched from cache without being decrypted.

744173 FIX: Memcache gateway service is missing.

744174 FIX: API logging is missing for a few locking related methods.

744175 FIX: LINQPad integration is compiled with 4.8 assemblies.

Release Notes NCache 4.9

Thursday, February 14, 2018

Introduction

NCache 4.9 contains a few important features related to clustering and caching in ASP.NET Core. This setup also has some performance improvements and is, therefore, a recommended upgrade for all NCache users.

Enhancements and New Additions

Following are some enhancements made in this release:

9. **Recovery from Split-Brain**
   Split-Brain is a situation where due to temporary network failures between cluster nodes result in multiple sub clusters. Each sub cluster, in this case, has its own coordinator node and does not know about the other sub clusters. This can eventually result in inconsistent data.

   With NCache 4.9, users can enable the cache clusters to automatically recover from Split-Brain scenarios.

10. **ASP.NET Core Response Caching**
    NCache's implementation of IDistributedCache utilizes Distributed Cache Tag Helper that provides the ability to dramatically improve the performance of your ASP.NET Core app by caching its responses.

11. **Major Performance Improvements**
    There is 20-25% performance improvement in basic ADD, INSERT and GET cache operations.

12. **More Features in Open Source and Community Editions**
    Open Source and Community editions have now same client API as Enterprise edition has. That means, all Enterprise developer features are now also available in Open Source and Community editions.

List of Bugs Fixed:

744163 FIX: Multi-site sessions with locking enabled resets sessions when a site goes down.

744164 FIX: Eviction index size is not calculated properly.

744165 FIX: IN queries are not thread-safe.

744166 FIX: Management operations are slow.

744167 FIX: Memcache gateway and client side wrappers are missing.

744168 FIX: 32-bit applications can't initialize cache.

744169 FIX: WriteThru Provider is not called when InsertBulk API is used with DSWriteOption.WriteThru in case of inproc client cache.

Release Notes NCache 4.8

Thursday, November 2, 2017

Introduction

NCache 4.8 contains a few important features related to runtime data sharing, messaging and .NET Core. This is a recommended upgrade for all NCache users.

Enhancements and New Additions

Following are some enhancements made in this release:

1. **NCache FREE Community Edition**
   NCache 4.8 has now launched a FREE Community and it provides the powerful GUI based NCache Manager to let you easily configure caches from a central location.

2. **.NET Core Clients**
   NCache now provides a totally native .NET Core Client that can run on both Windows and Linux. On Windows, NCache .NET Core client is installed through a Windows Installer (.msi). However, on Linux a separate installation (.tar.gz) is provided.

3. **Docker Support**
   NCache now fully supports Docker for both cache clients and cache servers. You can configure your .NET applications to be deployed in Docker and include NCache Client with it seamlessly.

4. **ASP.NET Core Session Provider & IDistributedCache**
   NCache now provides full ASP.NET Core support, both on .NET Framework and .NET Core (previously it was only on .NET Framework). This support includes a powerful ASP.NET Core Session Provider that has more features than the regular ASP.NET Core Session Provider. And, it also includes support for IDistributedCache interface in ASP.NET Core.

5. **Publish/Subscribe (Pub/Sub) with Topic**
   Publish/Subscribe (Pub/Sub) messaging paradigm is provided where a publisher sends messages into channels, without knowing who (if any) are the subscribers. And, Subscribers only receive message of their interest without knowing who the publishers are.

6. **Entity Framework Core (EF Core) 2.0 Extension Methods for NCache**
   NCache has implemented very easy to use EF Core 2.0 Extension Methods to allow you to cache application data that you're fetching through EF Core 2.0.

7. **Transport Level Security (TLS) 1.2**
   All communication from NCache clients to NCache servers can now be optionally secured through TLS 1.2 (a newer specification than SSL 3.0). TLS 1.2 ensures that all data traveling between NCache clients and NCache servers is fully encrypted and secured.

8. **Total Cache Management Thru PowerShell**
   NCache traditionally provided powerful GUI based cache management tools and also a rich set of command line tools. Now, NCache has implemented all of its command-line cache management tools in PowerShell. You can now write PowerShell scripts for more sophisticated cache management.

9. **Cache Client Keep Alive**
   Some firewalls break idle network connections which causes problems in cache client to cache server communication in NCache. Cache Client Keep Alive feature, if enabled on client node, automatically sends light a weight packet to cache servers at configurable interval (sort of a heart-beat). These packets are only sent in case of no activity between clients and servers and therefore do not interfere with regular client/server traffic.

10. **Thin NCache Manager Project Files**
    NCache Manager used to keep some cache configuration information inside the project file. However, that used to cause data integrity issues if multiple people tried to modify cache configuration from different machines. To fix this, NCache Manager now does not store any cache configuration information inside its project files. Instead, all configuration information is kept on cache servers that are common from all places and not data integrity issue arises any more.

11. **Cache Server-Only Licensing Option**
    Traditionally, NCache has only provided a client/server licensing option that requires both cache clients and cache servers to be licensed. However, now NCache provides an additional server-only licensing option where the cache clients do not require any licenses. Only the cache server requires licenses. If you wish to use server-only licensing, then please contact your account manager for its details.

List of Bugs Fixed:

744156 FIX: Specific cache id could not be started exception occurs on starting cache. Occurs rarely.

744157 FIX: Enable compression on client cache results in wrong behaviors.

744158 FIX: Memory leak during state transfer.

744159 FIX: Null reference exception occurs to client while multiple add-remove operations on cache having eviction policy LFU.

744160 FIX: An already locked object is being locked when item fetched using GetCacheItem

744161 FIX: Item is locked exception occurs on removing an item with correct lock handle.

744162 FIX: A locked item could not be fetched when correct lockhandle is passed to Cache.Get() with acquireLock= false.

Release Notes NCache 4.6(Service Pack 3)

Wednesday, May 10, 2017

Introduction

NCache 4.6 SP3 contains a few minor features and client cache optimizations. There are also various bug fixes reported by customers. This is a recommended upgrade for all NCache users.

Enhancements and New Additions

Following are some enhancements made in this release:

1. **SignalR Backplane with NCache**
   With NCache 4.6 SP3, cache can be used to distribute messages across a SignalR application that is deployed on multiple web servers.

2. **Client Side Data Reader**
   Client side data reader is introduced which is more stable and fault tolerant during the state transfer in cache.

3. **More Samples**
   More samples have been shipped with NCache installation. Existing samples are also refactored.

List of Bugs Fixed:

744149 FIX: Cache operations timeout if application is using threadpool extensively.

744150 FIX: State transfer takes too much time.

744150 FIX: ExecuteReader call throws statetransfer lost exception on node up or down.

744151 FIX: Value cannot be null exception when client cache is being used.

744152 FIX: "Input is not a complete block" exception is thrown on a few requests if client cache is being used with encryption enabled.

744153 FIX: Object reference exception on in-proc client cache initialization from ASP.NET application.

744154 FIX: Value cannot be null exception when client cache is being used.

744155 FIX: Caches configured for auto-start do not start automatically if security in configured on cache.

Release Notes NCache 4.6(Service Pack 2)

Tuesday, January 3, 2017

Introduction

NCache 4.6 SP2 contains a few client cache optimizations, enhancements in Bridge topology and feature related to monitoring of cache. There are also various bug fixes reported by customers. This is a recommended upgrade for all NCache users.

Enhancements and New Additions

Following are some enhancements made in this release:

12. **Monitoring Cache Clients from API**
    A new API is introduced to monitor cache client connected with a cluster. Using this API, a list of connected clients can be fetch as well as notifications can be registered in case of any new client connects or an existing one disconnects.

13. **Optimized Client Cache**
    Client Cache has now optimized way of synchronizing itself with the clustered cache. Instead of receiving events on each update made to clustered cache, client cache can now receive events in bulk at a configurable interval. This reduces the traffic as well as memory usage.

14. **Same Name Caches across Bridge**
    NCache Bridge can now have caches with identical names.

15. **Distributed Cache Loader**
    Cache Loader in NCache can now run on multiple nodes which can help in faster cache loading.

16. **FIPS-Compliant AES Encryption**
    FIPS-Compliant AES encryption is introduced in SP2.

17. **ASP.NET Core Session Provider**
    NCache ASP.NET session storage provider for ASP.NET Core applications is now supported.

18. **Changes in Licensing**
    NCache is now licensed based on the number of cores a machine has. For every 4 cores, 1 license is used. Minimum of 2 licenses are used to activate a machine even if the number of cores are less than 8. For 16 cores, 4 licenses are used, for 20 cores 5 are used and for 32 cores, 8 licenses are used.

    Developer licensed machines can now connect to a remote cache but with limitations on number of requests per second and total number of requests a client can make to a remote cache.

List of Bugs Fixed:

744134 FIX: Client Cache with Security throws error when credentials are passed through Init Params.

744136 FIX: OperationFailedException while creating CacheSyncDependency when Clientcache and security is enabled on cache.

744137 FIX: Clientcache throws no permission exception if security is enabled on cache.

744139 FIX: A few commands are not automatically retried on failure.

744140 FIX: Bridge Exception "object is not HP time".

744141 FIX: Object reference exception on Bridge's IConflict resolver.

744142 FIX: "Key already exists" exception is thrown when caching a Parsed Query.

744144 FIX: Client side counters for Client Cache are not working.

744145 FIX: Exception "Collection was modified" occurs on GetByTag calls during state transfer.

744146 FIX: Bulk Calls return data from just one node in case of node down.

744148 FIX: Failed to replicate queue if a new node (passive) added in bridge at run time.

Release Notes NCache 4.6 (Service Pack 1)

Tuesday, May 03, 2016

Introduction

NCache 4.6 SP1 contains a few optimizations related to memory and huge improvements in performance of NCache. This is a recommended upgrade for all NCache users.

List of Bugs Fixed:

744118 FIX: Without primary or secondary credentials, NullReferenceException is thrown on management operations with security.

744119 FIX: Cache host process crashes on SecurityException while starting cache.

744120 FIX: Client.ncconf client doesn't reflect security configuration changes.

744121 FIX: On adding security credentials through manager on client node, password is visible.

744122 FIX: Clientcache throws db sync dependency exception if security is enabled on cache.

744123 FIX: Clientcache throws no permission exception if security is enabled on cache.

744124 FIX: Callbacks are not thrown on performing bulk operations with Write-Behind (i.e insert bulk and add bulk).

744125 FIX: CacheItem is always added with version 1 after it is removed from cache resulting into data integrity issues.

744126 FIX: "Input string was not in a correct format" exception is thrown on service start for various locales.

744127 FIX: No exception is thrown if null dependency key is given.

744128 FIX: Exception is thrown on adding null subgroup.

744129 FIX: Cache host process is left unkilled between service restarts, hence resulting in multiple orphaned processes.

744130 FIX: Cache-all function for caching all queries is missing.

Release Notes NCache 4.6

Monday, December 7, 2015

Introduction

NCache 4.6 contains a few important features related to runtime data analytics, performance and memory optimization of NCache. This is a recommended upgrade for all NCache users.

Enhancements and New Additions

Following are some enhancements made in this release:

19. **Object Data Format**
    NCache cache can now be configured to store data in objects form. By default, all data is stored in binary form.

20. **Each Cache in Separate Process**
    Each cache in NCache is now hosted in its own process which runs independent of all other caches. Management of this process is done via NCache service which has information about all the cache processes running on the machine.

21. **Map Reduce**
    MapReduce in NCache will allow developers to write programs that process massive amounts of unstructured data in parallel across a NCache cluster. To distribute input data and analyze it in parallel, MapReduce operates in parallel on all nodes in a cluster of any size.

22. **Aggregator**
    Aggregator processes data records and returns compiled results. It groups values from multiple sources and can perform variety of operations like sum up values, calculating averages, finding minimum/maximum values etc. and returns single result.

23. **Entry Processor**
    NCache provides the ability to execute users' code on server side against a set of cache entries. Entry processors can modify cache entries on the server side without involving these entries to travel on the network for fetch and update operations.

24. **Data Reader**
    Queries can now be executed on cache using data reader just like the databases do. Using data reader, result set can be retrieved from servers in multiple chunks of configurable size. This approach gives a better performance and uses less memory on client end.

25. **Default Expiration**
    User can now configure NCache with default named expirations. NCache server will use default expirations when items are either inserted into cache without any expirations or inserted explicitly with named default.

26. **Order-by Clause**
    Select statements can now have orderby clause which will return the result set sorted on one or more specified attributes.

27. **Log Viewer**
    A GUI tool to view and monitor logs generated by NCache server and clients in one place. Log files of all servers can be viewed in same tool. The tool can filter log entries and important log entries can also be bookmarked.

28. **LIVE Upgrade**
    NCache 4.4 SP2 can be upgraded to NCache 4.6 without losing any data using its Bridge topology.

29. **Core Based Licensing**
    NCache 4.6 onwards, cache servers and clients will be licensed based on the number of cores instead of numbers of CPUs that box has.

Release Notes NCache 4.4 (Service Pack 2)

Tuesday, July 28, 2015

Introduction

NCache 4.4 SP2 contains a few important enhancements related to memory and performance of NCache. This is a recommended upgrade for all NCache users.

Enhancements and New Additions

Following are some enhancements made in this release:

1. **DumpCacheData Tool**
   Dumpcachedata tool takes a dump of a running cache by enumerating cache data and stores this data on a location specified by user. This data can later be reloaded to a new or same cache. This tool is helpful when customers need to restart their cache servers while they don't want to lose their business critical data in cache.

2. **DumpCacheKeys Tool**
   DumpCache tool is renamed to DumpCacheKeys.

3. **LinqPad Integration**
   LinqPad is a third party tool that can be used with NCache to query cache data. Data is shown on LinqPad console in a tabular form.

4. **HotApplicable Settings in Service Configuration**
   There are a few settings at service config level that now can be hot applied to caches running on a server. Previously, service restart was needed to apply settings defined in service configuration file. The list of HotApplicable settings is as follows:

#	Property Name	Hot Applicable
1.	NCacheServer.LicenseLogging	Yes
2.	NCacheServer.EnableNagling	Yes
3.	NCacheServer.NaglingSize	Yes
4.	NCacheServer.EventBulkCount	Yes
5.	NCacheServer.ExpirationBulkRemoveSize	Yes
6.	NCacheServer.ExpirationBulkRemoveDelay	Yes
7.	NCacheServer.EvictionBulkRemoveSize	Yes
8.	NCacheServer.EvictionBulkRemoveDelay	Yes
9.	NCacheServer.BulkItemsToReplicate	Yes
10.	NCacheServer.EnableCacheLastAccessCount	Yes
11.	NCacheServer.EnableCacheLastAccessCountLogging	Yes
12.	NCacheServer.CacheLastAccessCountInterval	Yes
13.	NCacheServer.CacheLastAccessLogInterval	Yes
14.	NCacheServer.LOHPoolSize	Yes
15.	NCacheServer.LOHPoolBufferSize	Yes
16.	NCacheServer.CacheSizeThreshold	Yes
17.	NCacheServer.CacheSizeReportInterval	Yes
18.	NCacheServer.LogClientEvents	Yes
19.	NCacheServer.EventLogLevel	Yes
20.	NCacheServer.AllowRequestEnquiry	Yes
21.	NCacheServer.RequestEnquiryInterval	Yes

22.	NCacheServer.ResponseDataSize	Yes
23.	NCacheServer.EnableSnapshotPoolingCacheSize	Yes
24.	NCacheServer.SnapshotPoolSize	Yes
25.	NCacheServer.SnapshotCreationThreshold	Yes
26.	NCacheServer.RequestInquiryCleanInterval	Yes

5. **Optimizations in Client Cache**
   A few architectural changes are made in client cache to have a better performance in case of bulk operations.

List of Bugs Fixed:

744113 FIX: Operations are not retried in case of connection failures or other internal exceptions. Instead these exceptions are thrown to client applications.

744114 FIX: Eviction is turned off by default.

744115 FIX: Session locking is turned off by default.

744116 FIX: NCache Manager allows non-admins to perform administrative operations.

744117 FIX: Activation tool crashes while activating machines with more than or equal to 32 cores.

Release Notes NCache 4.4 (Service Pack 1)

Monday, April 04, 2015

Introduction

NCache 4.4 SP1 contains a few new features as well as important enhancements related to memory and performance of NCache. This is a recommended upgrade for all NCache users.

Enhancements and New Additions

Following are some enhancements made in this release:

6. **Visual Studio Integration**
   Basic management and configuration operations can now be performed within the Visual Studio. With NCache 4.4 SP1, the Developer installation comes with an 'NCache Manager' extension which helps developers manage NCache from Visual Studio. Visual Studio 2010/2012/2013 are supported by NCache.

7. **NuGet Package for NCache Enterprise 4.4 SDK**
   A NuGet package is provided for developers to build applications using NCache without installing NCache on their machines. With this package, developers can write their applications using NCache API and test them with InProc cache.

8. **Entity Framework 6.0 Support**
   Entity Framework 6.0 and 6.1 integration is provided in NCache 4.4 SP1. Previously, NCache supported Entity Framework 5.0 or earlier. In this service pack, NCache also provides caching extensions for Entity Framework 6.0 which allows developers more control over which entities to cache. This is an alternate to no-code-change configuration option for developers who want to have a greater control over the entities being cached.

9. **Memory and Performance Optimizations**
   NCache 4.4 SP1 uses customized data structures which are enhanced to take up less memory and perform better that .NET's native data structures. These data structures avoid allocations on Large Object Heap (LOH) as much as possible to prevent NCache processes from entering into a state of severe memory fragmentation. Hence, NCache 4.4 SP1 has a major boost for both memory and performance.

10. **Split-brain Monitoring**
    Network partitioning or split-brain occurs when the cluster gets divided in such a way that some of the servers are unable to connect with the rest of servers. These instances are now logged into event viewer and email alerts for such scenarios can also be configured from NCache Manager.

11. **Windows Server 2012 R2 Certification**
    NCache 4.4 SP1 is certified for Windows Server 2012 R2. It has passed all Microsoft's compatibility tests. Microsoft validated that NCache works in accordance with Microsoft's standards.

List of Bugs Fixed:

744105 FIX: A few user specific NCache settings are saved to the HKLM section of the registry which throws errors in secured environments.

744106 FIX: Java client in Developer edition cannot connect to cache.

744107 FIX: NActivate shows NCache is successfully activated on the machine although it failed to save its activation information into the registry due to restricted rights.

744108 FIX: Aggregate dependency does not work when an item is added with both Aggregate dependency and Sliding dependency.

744109 FIX: NCache crashes during state transfer in some scenarios.

744110 FIX: Data sharing between Java and .NET clients does not work because of case-sensitivity of cache name specified in configuration.

744111 FIX: `GetIfNewer` does not reset item version when client cache is configured.

744112 FIX: NCache installation rolls back on its failure to start NCache service.

Release Notes NCache 4.4

Wednesday, January 08, 2015

Introduction

NCache 4.4 contains some important enhancements and few major bug fixes related to the stability of the cache. It is a recommended upgrade for all users.

Enhancements and New AdditionsAdditions

Following are some enhancements made in this release:

12. **Annual Licensing**
    NCache licensing has been moved to Annual licensing model from previously supported perpetual model.

13. **Memory optimization of internal data structures**
    Internal data structures are optimized to reduce the memory overhead of stored items.

14. **Cluster startup time improvements**
    Huge improvements in cluster startup time for cluster sizes over 20 nodes.

15. **New perfmon counter for memory usage**
    New perfmon counters are introduced to measure the memory acquired by various indexes. Following are the brief descriptions of each of them.

    Eviction Index Size: Size of eviction indices define on the cache.
    Expiration Index Size: Size of expiration indices defined on cache. (All dependencies meta info is also covered under this counter like meta info of 'Key Dependency', 'SQL Dependency' etc…)
    Group Index Size: Size of group and sub group indices
    Query Index Size: Size of query indices defined on cache.

    NOTE: "Cache Size" counter now includes the size of actual key, value pair and the memory utilized by above mentioned indexes.

16. **OQL related new permfon counters**
    OQL related new perfmon counters are also introduced to measure the performance of the query related cache operations like:

    Average µs/Query Execution: Average time query take while executing.
    Average Query Size: Average Number of items returned by queries.
    Queries/sec: Number of queries per sec on cache.

17. **Support for Windows Server 2012 R2**
    NCache is now fully compatible with Windows Server 2012 R2.

List of Bugs Fixed:

744101 FIX: Timeouts to some of the connected clients on server node join.

744102 FIX: Item loss in partitioned-replica of two nodes, when one of them goes down while each node has more than 50% of cache filled.

744103 FIX: NCache Manager lists down IPv6 addresses on multiple NIC configuration dialog.

744104 FIX: Few other minor bug fixes in command line tools.

Release Notes NCache 4.3 (Service Pack 1)

Wednesday, June 30, 2014

Introduction

NCache 4.3 Service Pack1 (SPs) contains some important enhancements and major bug fixes related to the stability of the cache. It is a recommended upgrade for all NCache 4.3 users.

Enhancements and New Additions

Following are some enhancements made in this release:

18. **Client side logging is configurable through client.ncconf file**
    You can configure the client side logging by modifying the client.ncconf file (exists at %NCHOME%/config folder. By default client-side logging is disabled for all caches, but you can enable client side logging for a cache at any time by specifying the 'true' value for '`enable-client-logs`' e.g.

    ```
    <cache id="mycache" ... enable-client-logs="false|true" log-level="error|info"/>
    ```

    Possible values for '`enable-client-logs`' attribute are 'false' and 'true'. Default value is 'false'.
    Possible values for '`log-level`' attribute are 'error' and 'info'. Default value is 'error'.

19. **Support for remote clients on a different network**
    Now NCache clients from a different or remote network can also access the cache which exists on a different network. To achieve this just add the following two tags in '*Alachisoft.NCache.Service.exe.config'* files at the cache server nodes.

    ```
    <add key="NCacheServer.MgmtEndPoint" value="public-ip:public-port" />
    <add key="NCacheServer.ServerEndPoint" value="public-ip:public-port" />
    ```

    Where the public-ip is ip-address which is exposed outside the network and public-port is the port which is accessible outside of the network.

20. **Client caches are browsable in NCache Manager**
    Client caches were previously only visible in NCache Manager under their respective second level (2nd level) caches, but now you can view the existing client cache in NCache Manager under the 'Local Caches' tree node of the 'Cache Explorer'.

21. **Email Alerts for out-proc local and client caches**
    Email alerts are supported for out-proc local caches and out-proc client caches. You can select the events on which NCache would send you email notification e.g. on Cache Start, on Cache Stop etc.

22. **Client caches can be configured from outside of cache cluster network**
    Client caches can be configured from outside the cache cluster network using command line tools e.g. CreateClientCache, AddClientNode etc.

23. **Support for Windows Server 2012**
    NCache is now fully compatible with Windows Server 2012.

24. **Java client for NCache**
    NCache provides Java client API to use the NCache as cache store from within the Java based applications. NCache Java client is fully compatible with .NET client API.

List of Bugs Fixed:

743101 FIX: NCache service crashed while using DELETE query with client cache.

743102 FIX: GetCacheItem() API cause the problem when used with ItemVersion.

743103 FIX: NCache Manager is not able to apply configurations when ‘CacheLaoder’ is enabled but not configured.

743104 FIX: Cache returns the item for invalid sub-group.

743105 FIX: Null reference exception on calling Remove/Delete (key, version) for non-existing key.

743106 FIX: State transfer is not working if read-through is configured and write-through provider is not configured or vice versa.

743107 FIX: ‘Request Queue Size’ client counter never return to zero.

Release Notes NCache 4.3

Wednesday, March 05, 2014

Introduction

NCache 4.3 has introduced new features and important enhancements that are critical for enterprise level applications. This new release allows NCache to be installed in cloud; on Windows Azure and Amazon. Also, a wrapper for Memcached is now available for existing users wanting to replace Memcached with NCache. This new release provides a package for run time data sharing between multiple platforms (Java & .Net). Java has been made 100% compatible with .NET and now you can manage your Java clients with NCache Manager.

Enhancements and New Additions

For a comprehensive list of all features in 4.3, please read NCache Features:

1. **Events with Data**
   While registering events with cache, cache clients can tell the cache whether they are interested in data or metadata when the events occur. By default, data or metadata are not sent with the events to clients.
   New API to register events has been introduced while to old API has been marked obsolete. Old API can't be used to receive data with events.

2. **Write-through / Write-behind Enhancements**
   Write-through and write-behind can now be configured for following behaviors:
   a. Batching – Where multiple write-behind operations can be batch written to the database after a configurable interval. The maximum number of items in a single batch is also configurable.
   b. Keep failed operations – Providers can now dictate whether to keep an item in the cache or not even if it is failed on database.

3. **Group-by for Queries**
   Queries can now be registered with 'Group by' clause as in database to group the results as needed.
   A new API has been introduced for this method. Currently, this new method '`ExecuteReader`' in the API can only be used if 'group by' is used. For all other select statements, old method should be used.

4. **Delete Statement in Queries**
   As in database, now items can be removed from cache by writing delete statements. Previously, only select and update statements were supported.
   A new API has been introduced to support delete statement. `ExecuteNonQuery` will be used for delete statements.

5. **Graceful Node Stop**
   A node can now be gracefully stopped in a cluster. This action will make sure that all client requests that have reached the node are executed on cache before it comes to complete stop. Similarly, all write behind operations pending in the queue at that time are also executed on the data source. However, no more client requests are accepted by this node.

6. **Encryption Enhancements**
   Following enhancements are made to the encryption feature:
   a. AES-128, AES-192, AES-256 encryption is supported.
   b. When encryption is enabled, indexed data is also encrypted.

7. **Compact Serialization Enhancements**
   Following enhancements have been made to compact-serialization:
   a. Users can select and de-select the data members to be compact serialized.
   b. Byte arrays are no more serialized.
   c. Compact types are hot-appliable.

8. **Enhanced CacheInitParams**
   `CacheInitParams` while initializing cache can now cover everything that can be configured in client.ncconf. Previously, client.ncconf was always required to initialize a cache. Configurations passed through `CacheInitParams` have an overriding effect on the settings configured in client.ncconf.

9. **De-serialized Data in InProc**
   InProc cache now keeps objects in de-serialized form. This removes the cost of serialization and de-serialization and hence, improving the performance. InProc client caches also keep objects in de-serialized form.

10. **API Calls Logging**
    API calls can now be logged by just configuring few options in client configuration. These logs are generated on the client boxes and are very helpful to determine which cache methods are being called and in what sequence.

11. **Configurable Log Locations**
    Users can configure to generate the log files at the location of their own choice. Each cache can have its own log location. By default, all log files will be generated in the log-files folder of install directory.

12. **Switch between Active/Active and Active/Passive Bridge Topologies**
    While adding caches to bridge, users can configure a cache to participate as an active or a passive member of bridge. Even when bridge is up and running, users can turn a passive into active and an active into passive without losing any data. User experience to configure a bridge is also changed as the topologies in bridge can be switched between Active-Active to Active-Passive at any time. Other topologies 'Star' and 'Hub-spoke' are currently not available in bridge.

13. **Master Cache in Bridge Topology**
    User can pick one of the two caches in bridge as a 'Master cache'. Whenever, there is a need for a state transfer between caches in bridge, data is transferred from a master cache to the non-master. When the master cache goes down, the only remaining cache becomes master automatically.

14. **Connect/Disconnect Caches in Bridge Topology**
    Cache administrators can temporarily connect and disconnect caches from the bridge while bridge is running. When a cache is disconnected, no data is transferred between bridge and the disconnected cache. Similarly, the cache on the other side of the bridge stops queuing data to the bridge as the disconnected cache is no more receiving any data.
    Cache can be reconnected at any time.

15. **Socket Protocol Management**
    Communication protocol for management and monitoring operations are changed to 'Socket' from .Net Remoting'. This makes NCache and JvCache's management and monitoring tools inter-compatible.

16. **NCache Manager Compatibility**

NCache Manager can now be used manage JvCache clients as well. NCache Manager can also fetch SNMP counters for JvCache clusters.

17. **Connect to Remote Perfmon via machine name or IP address**
    We have observed that in some environments, remote perf counters are accessible via their machine names only and on a few via their IP addresses. So in this version of NCache, NCache Manager has an option where user can pick to collect remote perf counters via their IP or machine name.

18. **DLLs Locking Issue Resolved**
    NCache Manager used to lock the dlls when query indexes were configured by the users. In this version, NCache Manager opens the given dlls in a separate app domain and therefore, never locking the dlls.

19. **ReportView Control for NCache Monitor**
    There is another type of dashboard available in NCache Monitor that allows users to create a report view style dashboard. In this dashboard, users have two report controls. One is for cache server nodes, while other one for client nodes.
    Users can drop the counters in this control and their values are shown in a report view style as shown in perfmon.

20. **Logging of Counters in NCache Monitor**
    Counters added in report view can also be configured to be logged. Users can start and stop logging at any time. They can also schedule the logging to start automatically by specifying the start and stop time. These log files are generated in .csv format.

21. **NCache Monitor Compatibility**
    NCache Monitor can now also be used to monitor JvCache. Depending upon whether the selected cluster is of NCache or JvCache, it fetches counters from perfmon or SNMP respectively.

22. **New Command Line Tools Added**
    Following new command line tools are added to NCache:
    1. Create Cache: Configure Cache Tools is now changed to Create Cache Tool. It will handle 2 cases.
        a. Simple case: In this case it will take minimum required parameters (i.e. cache name, server, port, topology, size, eviction policy etc.) and create a cache with default values (like NCache Manager).
        b. Advance case: In this case a configuration file containing cache settings will be taken as parameter with server list on which cache will be created.
    2. Add Backing Source: It will take cache id, assembly path, class name, provider name, read thru/write thru option and also dependent assembly folder as input parameter.
    3. Add Startup Loader: It takes cache id, assembly path, class name and also dependent assembly folder as input parameters.
    4. Add Compact Types: It takes cache id, assembly path, class name and also dependent assembly folder as input. If Specified class is the implementation of `IGenericTypeProvider`, this will register generic type through implemented provider.
    5. Add Query Index: It takes cache id, assembly path, class name, attribute/attribute list and also dependent assembly folder as input.
    6. Remove Backing Source: It takes cache id, server, provider name and read-through/write-through option as input parameters. It will remove backing source from specified cache on all registered nodes.

7. Remove Startup Loader: It takes cache id and server as input parameters and remove startup loader from specified cache on all registered nodes.
8. Remove Compact Types: It takes cache id, server and class as input parameters and remove specified compact type from cache on all registered nodes.
9. Remove Query Index: It takes cache id, server and class name/attribute(s) as input parameters and remove specified query index from cache on all registered nodes.
10. Add Data Share: It takes a configuration file containing data sharing configuration (mapping between 2 assemblies/jar files), cache id, server and dependent assembly folder as input parameters.
11. Remove Data Share: It takes cache id and server as input parameters and remove data sharing type from cache on all registered nodes.
12. Get Cache Configuration: It takes cache id, server and path (where file containing cache setting should be created) as input parameters. This tool will generate only cache settings (environment independent settings) in a file named as cache id on specified path.
13. Deploy Assembly: It takes cache id, server and assembly/folder path as input parameters. Specified assembly will be deployed on all registered servers of cache. If folder path is specified, then all assemblies in that folder will be deployed.

23. **Memcached Wrapper for NCache**

Existing Memcached users can now switch to NCache without code change. There are two ways to replace memcached with NCache:

1. Memcached Gateway

   This gateway is installed on client and server boxes as a windows service. It is an implementation of memcached protocol which wraps the NCache calls inside it. This way, all API calls from memcached clients are routed to NCache servers via this gateway.

   Only configuration changes are required in this approach.

2. Memcached Plug-In

   Some of the open source client implementations for memcached are modified to work with NCache servers while keeping their API as is. The source and binaries of these implementations are shipped with NCache installation.

Following memcached client implementations are supported in this approach:

a. Enyim
b. BeIT
c. .NET Memcached Client Library

24. **NHibernate Integration**

NHibernate integration is written from scratch to remove the limitations of previous implementation. Following are the few enhancements made in new implementation:

a. There is only one configuration file for all NHibernate applications.
b. Settings can now be configured at region level. Each class in that region will use these settings.
c. Clearing on region now clears only the entries residing in that particular region.

25. **ASP.NET OutputCacheProvider Hooks**

Users can now write their own code to modify the cache items before they are inserted in NCache. Users can change the expiration, dependencies etc. of output cache items by writing these hooks.

For this, users have to implement an interface provided with `OutputCacheProvider` and then register this assembly and class in web.config.

26. **Tagging of Cached Items**
    All items cached from various NCache integrations are tagged with special tags that determine the type of cache items. For example all sessions created in cache have a special tag that tells it's a session cache item. This way users can identify any item in cache whether it's a session or not.
    Similarly, `OutputCache` and `ViewState` items are also tagged with their own tags.

Release Notes NCache 4.1 (Service Pack 3)

Tuesday, October 24, 2013

Introduction

NCache 4.1 Service Pack3 (SP3) contains some important enhancements and major bug fixes related to the stability of the cache. It is a recommended upgrade for all NCache 4.1 users.

Enhancements and New Additions

Following are some enhancements made in this release:

1. **Operation retries in session store provider**
   NCache session store provider for ASP.NET session caching now has the ability to retry any cache operation if operation fails.

   Add the following attributes in the NCache session store provider settings:
   `operationRetry="3"`
   `operationRetryInterval="3000"`
   'OperationRetryIterval' attributes takes the value in w seconds.

2. **Operation retries in object cache provider**
   NCache Object Cache provider for .NET4x now has the ability to retry any cache operation if operation fails.

   Add the following entry under the `<appSettings>` in application config file (app.config or web.config) like:
   ```
   <add key="operationRetries" value="3"/>
   <add key="operationRetryInterval" value="2000"/>
   ```
   'operationRetryInterval' is the value in milli seconds

3. **Viewstate grouping page-wise**
   A new feature is implemented in ViewState caching, which allows grouping of viewstate of related pages. This will provide more control for viewstate caching. Using this feature it is possible to cache different groups of veiwstate in different caches, or cache veiwstate using different expirations for each group of viewstate.

4. **Maximum VeiwState to cache, per session**
   This enhancement in NCache ViewState caching module provides the option to restrict the number of viewstate per page in cache.
   Add the 'maxViewStatesPerSession' attribute in the '<settings …> tag under the 'ncContentOptimization' tag. This attribute take value in integers.

5. **Delay in cache startup loader**
   At Cache start, Cache startup loader is not started instantly. Its start is delayed for 20 seconds after cache start. This delay is configurable by using the following attribute in 'Alachisoft.NCache.Service.exe.config' file:

   Add the following entry under the `<appSettings>` in NCache service config file like:
   ```
   <add key="NCacheServer.CacheLoaderStartupDelay" value="20"/>
   ```

   Default value for this attribute is 20 seconds.

6. **Delay between auto-start caches**

To avoid partial cluster connectivity problems when caches are started simultaneously using auto-start feature, a delay can be introduced.

Add the following entry under the <appSettings> in NCache service config file like:
```
<add key="NCacheServer.CacheStartDelay" value="3"/>
```

Default value for this attribute is '3 seconds'.

7. **Suppress IndexNotDefinedException**
   'IndexNotDefined' exception is thrown to the client, when it queries an attribute that is not indexed. This problem can occurred easily when NamedTags are used as indexes are created for NamedTags at runtime. Now it is configurable to suppress this exception.

   Add the following entry under the <appSettings> in NCache service config file like:
   ```
   <add key="NCacheServer.DisableIndexNotDefinedException" value="true|false"/>
   ```

8. **NCache client apps can consume cache generated events 'Synchronously' or 'Asynchronously'**
   Client app can be configured to consume all of the received events 'Synchronously' or 'Asynchronously'. Default event consumption mode for NCache Client is 'Asynchronous' but events can be consumed on client side 'Synchronously'.

   Add the following entry under the <appSettings> in client application 'App.config' file like:
   ```
   <add key="NCacheClient.AsynchronousEventNotification" value="true|false"/>
   <add key="NCacheClient.NumberofEventProccesingThreads" value="2"/>
   ```

   For synchronous event processing mode, in this example 2 threads are used but it is configurable to adjust the event processing needs according to the client application requirements. Default minimum value is '1' and default maximum value is '5'.

9. **More perfmon based counters for cache and cache client**

   New Server Side Counter
   - **Response Queue Count**: Number of items in response queue, all responses are queued in this queue before actual send.
   - **Response Queue Size**: Size of response queue specified in bytes. Data size of all the responses in the queue.
   - **Event Queue Count**: Number of events in event queue.

   New client side counters
   - **Events Processed/sec**: Number of events processed per sec on client.
   - **Events Triggered/sec**: Number of events triggered and Received by client per second
   - **Average ms/event**: Average time taken in single event processing on the client.

10. **Cache clear will fire ItemRemoved events for all registered ContinuousQuery if some data exist in a ContionousQUery resultset.**
    Now cache clear call will fire the ItemRemoved events for all those keys which exist in a ContinuousQuery result set.

11. **Client cache now manage its own expiry**
    If there is a client cache enabled and a lot of items are being expired from the L2 cache then L2 sends item expiry events to client cache (L1 cache) to remove these items from its local store. This hurts the performance of active clients and utilizes

high CPU. Now client cache (L1 Cache) will also maintains the expiry of items, and L2 cache will not send the expiry events to L1 cache.

12. **EntitiyFramework caching enhancements**
    NCache provided EFCaching module is enhanced for EF query analysis and caching.

13. **Auto Start Bridge on bridge service start**
    Now bridge service can be configured to start the specified bridges on service start. To configure a bridge to start automatically when a bridge service (Alachisoft.NCache.Bridge.exe) gets started / restarted. Uncomment the following tag in '*Alachisoft.NCache.BridgeService.exe.*config' file and provide the bridge name(s) which are configured on current machines, like:
    `<add key=" NBridgeServer.AutoStartBridges" value="Bridge-1,Bridge-2"/>`

List of Bugs Fixed:

741301 FIX: Connection failover problem in replicated topology, if other server nodes info is not available in 'client.ncconfig' file at client machine then client was unable to connect to other server.

741302 FIX: Not all ContinuousQuery notifications are delivered to the client application in case there is a change in an item that falls under a query.

741303 FIX: ContinuousQuery notifications (ItemAdded, ItemUpdated, ItemRemoved) are not fired after clearing the cache.

741304 FIX: Performance degradation in ContinuousQuery with Add, Update & Remove operations. Now Continuous query will be evaluated for each operation 'Asynchronously'.

741305 FIX: NCache bulk operations that were consuming a lot of NCache process memory due to LOH occupancy and results in timeouts and memory issues. Now NCache divide the large responses into multiple chunks of 512kb and this will help to reduce the memory issues.

741306 FIX: All server information gets removed from 'client.ncconf' file on cache client nodes on updating the cache settings if cache is configured to use multiple NIC's.

741307 FIX: NCache manager was not showing the active directory user after 1000 user count.

741308 FIX: NCache.caches[] API returns only the clustered cache (L2 cache) handle even though it has a client cache (L1 cache).

741309 FIX: ConfigureCacheSecurity command line tool does not work properly, when node level security is enabled.

741310 FIX: EF Caching analysis report is not generated if application exist before analysis time.

741311 FIX: NActivate tool does not work on boxes with 32 cores.

741312 FIX: Bridge: State-transfer started when a server node left the source cache [Active-Passive].

741313 FIX: Bridge: Item version mismatch problem when active-node goes down.

741314 FIX: Bridge: Data replication occurred twice if coordinator-node of the target cache goes down.

741315 FIX: Bridge: Replication stops if the source cluster faces some connection issue due to network.

741316 FIX: Bridge: No replication across the bridge, if cluster caches started before the bridge.

741317 FIX: Bridge: State-transfer initiated from target to source, if coordinator node of the target cache left the cluster.

741318 FIX: Bridge: Target cache is cleared when bridge reconnected or target cache reconnected with the bridge due to connection lost on WAN.

741319 FIX: Bridge: Bridge Queue count is not displayed for bridge active node.

741320 FIX: Bridge: Some other problems related to state-transfer and data mismatch are also fixed for ‘Active-Passive’ and ‘Active-Active’ bridge topologies.

Release Notes NCache 4.1 (Service Pack 2)

Tuesday, October 10, 2012

Introduction

NCache 4.1 Service Pack2 (SP2) contains some important enhancements and major bug fixes related to the stability of the cache. It is recommended upgrade for all NCache 4.1 users.

Enhancements and New Additions

Following are some enhancements made in this release.

1. **Data Encryption**
   Encryption feature is provided to make sure that data traveling between NCache client and NCache server or between cluster nodes is encrypted. This will prevent the user data leakage even if data packets are sniffed from the network.

2. **Multiple database dependencies are allowed on single item**
   Now multiple database dependencies can be added for single cached item. This feature will compensate the 'SQL Notification' dependency limitation of a single database dependency.

3. **Auto start delay for caches**
   Caches which are configured to 'auto start' now can be delayed according to user specified time. This will improve the NCache response time in case there are too many caches configured for 'auto start'.

   Add the following entry under the `<appSettings>` in NCache service config file like:
   `<add key="NCacheServer.AutoStartDelay" value="5"/>`

4. **Auto start of client caches**
   Now you can configure the 'client caches' to start automatically on service restart or when client machine is rebooted. This will make the client cache available without user intervention.

5. **Viewstate can be associated with session**
   Now veiwstate is cached along with 'session id' as 'group' info. This wills facilitate the user to relate the viewstate and session of the same application and user. Using this feature user can remove all the viewstate related to a specific session when a session is closed or expires.

   Add the following attribute inside `groupedViewStateWithSessions="true|false"`

List of Bugs Fixed:

741201 FIX: Extra GC collection calls in SSP module which caused a performance overhead.

741202 FIX: Session size increase 3 times after serialization.

741203 FIX: Viewstate crashed the application if viewstate logging is turned off.

741204 FIX: View State Caching with Ajax causes problem in page loading.

741205 FIX: Cluster get partially connected if cache name of cluster nodes is not in same case.

741206 FIX: Newly added server info is not updated in ‘client.ncconf’ file at client nodes.

741207 FIX: Cache initialization problem from inside of SQL CLR based stored procedure.

741208 FIX: Local cache cannot be run on a loop back IP (127.0.0.1) in developer edition.

741209 FIX: Dead lock occurred if clients are frequently connected/disconnected.

741210 FIX: General Notification stop working if cache is not disposed.

Release Notes NCache 4.1 (Service Pack 1)

Tuesday, May 10, 2012

Introduction

NCache 4.1 Service Pack1 (SP1) contains some important enhancements and major bug fixes relating to the stability of the cache. It is commended upgrade for all NCache 4.1 users.

Enhancements and New Additions

Following are some enhancements made in this release.

1. **Generics types support for Compact Serialization**
   You can add custom generic types for compact serialization. All generic types with any number of arguments can be serialized through compact serialization. You can register generic types through NCache Manager or through a custom handler by implementing the interface IGenericTypes. Currently, this feature is only available in for .NET clients.

2. **Update CacheItem hint at runtime**
   You can now modify cache item attributes at runtime without modifying the data. Currently, the API (*SetAttribute*) allows you to update dependency and expiration hints.

3. **UTC support for Different Time zones**
   You can have cache servers as well as clients running under different time zones; NCache will maintain a standard time to expire an item on the basis of local time zone. Whenever an item replicates or moves from one cache server to the other the expiration is reset according to the local time zone hence makes the item expiration possible according to the configured timeslot.

List of Bugs Fixed:

741101 FIX: Client was unable initialize NCache through a SQL CLR based procedure or trigger.

741102 FIX: Object ref exception if only WritBehind is configured for a cache and WithThru operation is performed in API.

741103 FIX: Output caching in 2-node web-farm, item is deleted from cache if 2nd request goes to 2nd node of the farm.

741104 FIX: Query parser throws buffer overflow if the query length exceeds 256 characters.

741105 FIX: GetGroupsKeys, GetBulk, Tags, Query does not return correct results during state transfer.

741106 FIX: In case of partition topology only one server was sending general notifications to remote clients.

741107 FIX: NamedTags query throws exception "Index not defined” if only one item is added in partitioned cache.

741108 FIX: Java client throws exception that it is unable to read client.ncconf file even though you have provided the cache-id, server-ip and port through the API.

741109 FIX: Slow response on right clicking a cache server.

741110 FIX: Content optimization log file was not generated.

741111 FIX: Licensing does not work on machines with 32 or more cores.

741112 FIX: AddDependency() resets the existing dependency instead of appending a new dependency.

741113 FIX: Client Application does not receive updated hash-map when a network is disabled on the cache server.

741114 FIX: Security was using case sensitive username.

741115 FIX: Cluster becomes partial and sometimes also unresponsive. This happens when machine is rebooted and cache from that server tires to join the cluster again, before cluster has removed this server from its cluster membership.

Release Notes NCache 4.1

Tuesday, August 23, 2011

Introduction

NCache 4.1 has introduced very important new features and enhancements that are critical for enterprise level applications. This new release gives a whole new package for run time data sharing between multiple platforms (java & .Net). Java has been made 100% compatible with .NET and now you can even plug-in your Java code with NCache process. NCache request-response model has also been enhanced to handle large responses where GBs of data can be retrieved from the clustered cache in a single request. All the bugs that were reported in earlier release have been fixed in this release.

Enhancements and New Additions

For a comprehensive list of ALL FEATURES in 4.1, please read NCache Features:

1. **Runtime Data Sharing between .NET & Java**
   NCache now allows you to store either .NET objects in the cache and read them as Java objects from your Java applications, or vice versa. And, instead of doing .NET to Java transformation through XML, NCache uses binary-level transformation. As a result, the performance is super fast. NCache automatically resolves type conflicts between Java and .NET.

   You can also utilize multiple versions in Runtime Data Sharing between .NET and Java. See details below.

2. **Multiple Objects Version Support (.NET & Java)**
   You can now share multiple versions of the same .NET or Java classes across multiple applications. One application may be using version 1.0 of a class while another application may have a later version 2.0 of this same class. When version 2.0 of this class is stored in the cache, the earlier application can still fetch this class as version 1.0, and vice versa. NCache lets you configure version mappings through XML configuration files.

   You can utilize version also in Runtime Data Sharing between .NET and Java.

3. **Continuous Query (.NET & Java)**
   NCache lets you specify a data-set based on an SQL-like query. It then maintains this data-set in the cache cluster for you and monitors any runtime changes in it, including Additions, updates, or deletes. And, NCache notifies your application whenever any of these changes occur in the dataset. This allows your applications to more intelligently monitor for either data changes or addition of data matching a certain criteria and be notified by NCache.

   This is a powerful feature if you want to share data at runtime between multiple applications.

4. **Much Enhanced Bridge Topology**
   Bridge Topology allows you to intelligently and asynchronously replicate the entire cache across the WAN. NCache 4.1 now offer four different configurations in Bridge Topology. They are:

   1. **Active/Passive:** In this configuration, one cache is active and the other passive. Whatever updates you make to the active cache are asynchronously

applied to the passive cache by the Bridge. Since the connection between active and passive is across the WAN, it is likely to be unstable. But, the Bridge is aware of it and if the connection breaks, it automatically reestablishes it.

2. **Active/Active:** In this configuration, both caches are active and the Bridge receives update requests from both sides. Both caches are also maintaining an identical "clock" for time-stamping. This clock is synchronized through the Bridge. Whenever there is a conflict, meaning the same cache item is being updated in both caches, it is resolved by default on a last-update-wins strategy. But, if you want, you can provide a custom resolution handler which is invoked in case of a conflict. You can then determine which update should stay and which one should be discarded.
3. **Hub-Spoke:** In this configuration, there is one active Hub and multiple passive Spokes (or satellite caches). All updates originate from the Hub and are asynchronously applied to all the Spokes. The Spokes are always read-only.
4. **Star:** This configuration looks identical to a Hub-Spoke except that all the Spokes are also active. In this configuration, there is one centralized cache and multiple satellite active caches. All updates are synchronized through the centralized cache to ensure consistency.

5. **Named Tags (.NET & Java)**
   Previously, you could only assign tags as values. Now, you can assign tags with names. This allows you to index data based on attribute name and attribute value concept. Previously, you could index objects but all string data could not be indexed. Now, even string data (e.g. XML) could be indexed with named tags. Then, you could either use NCache API to fetch data belonging to one or more named tags or you could issue SQL-like query (through LINQ or OQL) for it.

6. **Java Features Now 100% Equivalent to .NET**
   NCache 4.1 brings support for Java at the same level as .NET. Here are the new Java based features intended to catch up to .NET feature-set:

   - **Java Client API 100% equivalent to .NET**
   - **Java Client Cache (InProc):** You can now use Client Cache feature in Java applications on Windows or Unix. There is no code change required in your applications to enable Client Cache.
   - **Read-Thru, Write-Thru, Write-Behind in Java:** You can now write your Read-Thru and Write-Thru handlers in Java and register them with NCache. NCache runs your native Java code on the cache servers just like it does with .NET code.
   - **Cache Loader in Java:** Write your cache loading code in Java and register it with NCache. Your native Java code will run on cache servers just like the .NET code.
   - **Dynamic Compact Serialization for Java:** Now, you can register your Java classes with NCache and NCache generates their serialization code at initialization time, compiles it in-memory, and sends it to NCache clients. This compiled Java code is then run to serialize/de-serialize your Java classes. This obviously speeds up your performance because compiled serialization code runs much faster than Reflection based serialization that is otherwise done in Java.
   - **Spring/Hibernate Support:** NCache now provides an L2 Cache for Hibernate. This allows you to start using NCache without making any code changes in your Java application.

7. **Multi-Response Model**
   NCache now allows the cache servers to return larger response in smaller chunks for a given request. This improves the overall cache performance because the serialization cost reduces with the size and also helps applications to fetch larger data sets in a single call. It also eliminates the .NET serialization limitation where it fails to serialize data size larger than 1.9 GB. Response threshold and chunk size both are configurable and can be modified from NCache service config file "NCache\bin\service\ Alachisoft.NCache.Service.exe.config"

```
<!-- Response size in MB -->
<add key="NCacheServer.ResponseDataSize" value="1024"/>
```

List of Bugs Fixed:

741001 FIX: When using Client-Cache GetCacheItem returns incorrect Group name.

741002 FIX: Object Query does not work if the IN operator has just one parameter e.g `Select Northwind.Customers where this.City In(?).`

741003 FIX: Alachisoft.NCache.Web.dll is missing in "4.0 GAC" folder in 64 bit remote client installation of NCache

741004 FIX: Alachisoft.NCache.Security.dll (32 bit) missing in Developer installation.

741005 FIX: When a node joins the cluster the State transfer throws an object reference not set exception in cache server log file. This happens if one write through provider is configured with write behind enabled.

741006 FIX: In a two or three node partitioned replica cluster, if a node goes down and immediately comes back before the state transfer is completed, items are lost.

741007 FIX: If the indexed items are also associated with tags, the LINQ query without any where clause may return an exception that the key is already added in the dictionary. This was happening because of duplicate keys were added in the result.

741008 FIX: Cannot increase the cache size at runtime from NCache Manager.

741009 FIX: Cache server sends multiple status inquiry requests to the other cache server if it does not receive a response in a certain time for a given request. In case of network failures or cache node unresponsiveness the request queue becomes full and timeout errors are thrown to the user. This status inquiry mechanism is now disabled by default and can be turned on through NCache service config file.

741010 FIX: If there are two or more caches running on the same machines and both are using TAG feature then you may get object reference not set errors in the cache server log files. This may also slow down the cache response in some situations. This has been fixed.

741011 FIX: If the response size is larger than 1.9 GB the cache will hang and clients will timeout. This happens because of the .NET serialization limitation. This happens usually with bulk operations like Queries, Tags, Groups, and bulk operations.

Release Notes NCache 3.8 (Service Pack 4)

Thursday, Mar 17, 2011

Introduction

NCache 3.8 Service Pack 4 (SP4) contains important fixes which are important for production environments. It is a recommended upgrade for all NCache 3.8 users.

Enhancements and New Additions

Following are some enhancements made in this release.

4. **String sharing between Java and .NET (vice versa)**
   You can share string values between .NET and Java applications.
5. **SQL Cache Dependency Custom Queue / Notification Service**
   SQL Cache dependency architecture has been enhanced to support custom queues and notification services that does not require extra user permissions like “create queue” and “create service”. NCache now allows you to choose either default mode (with default SQL notification and queue service) or custom mode where you can specify your own queue and notification service.

   The service name format should be "NCacheSQLService-[ip-address]" and "NCacheSQLQueue-[ip-address]" where the IP-Address will be of machine on which NCache service process will be running. You can specify this setting in service configuration file “NCache/bin/service/Alachisoft.NCache.Service.exe.config”.

   ```
   <add key="NCacheServer.NCacheSQLNotificationService"
   value="NCacheSQLService"/>
   ```

List of Bugs Fixed:

738401 FIX: Enumeration returns byte array if the item was added through cache loader.

738402 FIX: ViewState module throws Null Reference Exception if the traces are disabled and module tries to write something in log.

738403 FIX: Command line activation throws invalid license key.

738404 FIX: Cache becomes unresponsive or timeouts when the number of items exceed 7 million.

738405 FIX: Query Sample in x64 has some compilation errors.

738406 FIX: NCache Session State Management assembly was missing in 64-bit developer installation.

738407 FIX: Connection Balancing and failover in Partitioned and Partitioned Replica topologies does not work if dual NICs are used where one NIC is bound to client-server communication and the second NIC is bound to cache-server (cluster) communication.

738408 FIX: A 32 bit security assembly “Alachisoft.NCache.security.dll” is missing in windows GAC in NCache 64 bit setup. This does not allow 32-bit applications to run on 64 bit caching server.

738409 FIX: CacheProvider 4.0 - GetValues() method returns all key-values including those which are also not available in cache. As per Microsoft documentation it should return only the key/values that exist in cache.

738410 FIX: CacheProvider 4.0 - NSQLChangeMonitor and NOraChangeMonitor are Enterprise Edition features and are unintentionally exposed in NCache Professional Edition.

Release Notes NCache 3.8 (Service Pack 3)

Tuesday, Nov 02, 2010

Introduction

In this release of NCache 3.8 Service Pack 3 (SP3), NCache Enterprise Edition is separated into two products named as 'NCache for .NET' and 'NCache for Java'. Previously 'NCache' had both components combined into a single package.

Two new features are also added in 'NCache Java Client'.

Enhancements and New Additions

Following features are added in 'NCache Java Client' in this release.

1. **Database Dependency**
   Now you can add items in cache with 'database dependency' from your java application. Any change in related database invalidates the cache entry, and your application will receive the notifications about this change.

2. **Streaming API**
   NCache Java Client has added streaming support in the API where you can read and write binary data stream in the cache.

List of Bugs Fixed:

738301 FIX: On adding new node in cluster, new node info was not added in "client.ncconf" file on all cluster nodes.

APPLIES TO:

- NCache Enterprise Edition 3.8.x

Release Notes NCache 3.8 (Service Pack 2)

Monday, Oct 10, 2010

Introduction

NCache 3.8 Service Pack 2 (SP2) contains important fixes and enhancements that have been reported by customers. It is a recommended upgrade for all users of NCache 3.8.

Enhancements and New Additions

Following features are added in this release.

3. **License Support for Xen VM**
   NCache now support licensing on Xen VM environments.

4. **Visual Studio 2010 Assembly Reference solution**
   Visual Studio 2010 shows a warning message when you add reference to NCache assemblies and then the build also fails. This only happens if the application target frame work is .NET 3.5. This is a known issue of Visual Studio that it gets confused when you have the same assemblies available in the GAC 2.0 and GAC 4.0 and for dependent assemblies it tries to load them from GAC 4.0 which results in the warning message.

   We have resolved this issue by copying all the assemblies in the NCache bin/assembly folder and now Visual Studio does not have to locate the dependent assemblies in GAC. This has resolved the issue.

List of Bugs Fixed:

738211 FIX: ViewState Caching does not work with AJAX controls especially GridViewPager.

738210 FIX: Session Store Provider throws an exception “The SessionStateStoreData returned by ISessionStateStore has a null value for items” if the session is empty.

738209 FIX: ViewState assemblies were build with wrong version in NCache Enterprise Developers installation.

738208 FIX: There was a performance issue in the feature object query. The search result was slower when you have a large set of items in the cache.

738207 FIX: NHibernate sample has a build issue in NCache Professional Edition.

738206 FIX: NCache Manager throws an object reference not set error if you enable the security without specifying users.

738205 FIX: NCache Manager throws an object reference not set error if you change the bind to IP (using the option “Select Network Interface Card”) for local cache server. This option is now removed from local cache server.

738204 FIX: NCache Manager fails to load ReadThru/WriteThru Provider if the provider is an exe file. When you deploy the ReadThru/WriteThru provider using the “Deploy Provider” button, NCache Manager rename the file extension to .dll

738203 FIX: NCache Professional does not allow remote client connections. This happens if the remote client is installed with NCache Professional Remote Client installation.

738202 FIX: NCache integration with Microsoft Enterprise Library v4.1 returns Boolean value for indexer method `Cache["Key"]` instead of the actual object.

738201 FIX: Multiple bugs are fixed in JavaScript and CSS Minification. The JavaScript Minifiction fails if the rendered output contains .axd files in the JavaScript tag, and if the <script> tag ends with </Script>. Similarly, the CSS minification fails if relative paths are used for loading images. Now the relative paths in CSS are converted to absolute path.

APPLIES TO:

- NCache Enterprise Edition 3.8.x
- NCache Professional Edition 3.8.x

Release Notes NCache 3.8 (Service Pack 1)

Tuesday, August 24, 2010

Introduction

NCache 3.8 Service Pack 1 (SP1) contains important fixes and enhancements. The most important and demanded feature added in this release is the support of .Net frame work 4.0

The API is completely compatible with the 3.8 release version and applications can upgrade without re-building/re-compiling the application.

Enhancements and New Additions

Following features are added to this service pack 1.

1. **.Net 4.0 support available.**
   The code base of NCache cache server has been converted to .NET 4.0 and the NCache client is available in both .NET 2.0 and 4.0 versions.

2. **Improvement in client cache management**
   There is an improvement in client cache management through NCache Manager where project files will contact to client nodes on refresh option and this has improved fast loading of Ncache Manger project file.

3. **Samples are now builded with Visual studio 2008.**
   Ncache samples are now builded with visual studio 2008.

4. **VeriSign issue.**
   VeriSign issue, default in service configuration file should be generatePublisherEvidence enabled ="false"/>. The enhancement is made.

5. **ReadThru Interface improved**
   ReadThru provider interface signature has been modified to support maximum features of NCache. There is new structure introduced under the namespace "Alachisoft.NCache.Runtime.Caching" called ProviderCacheItem which is similarly to the CacheItem. You can now easily specify expirations, tags, eviction hints, dependencies etc.

   New interfaces
   ```
   public void LoadFromSource(string key, out ProviderCacheItem cacheItem)
   public Dictionary<string, ProviderCacheItem> LoadFromSource(string[] keys)
   ```

6. **CacheLoader Supports IsResyncExpiredItem**
   Now, you can specify IsResyncExpiredItem property in Cache Loader so that the expired items can be reloaded automatically.

7. **NHibernate Integration**
   NCache is not supporting the latest version of NHibernate 2.1.2. We have also added region support in this release. NHibernate sample application is also modified with NHibernate regions support.

List of Bugs Fixed:

738108 FIX: AutoStart fails to start the cache if the cache is under heavy load. Now the AutoStart starts the cache in Asynchronous mode.

738107 FIX ViewState assemblies were built with wrong version in NCache Enterprise Developers installation.

738106 FIX: CAB Integration does not work in NCache professional Developers installation because of aggregate dependency.

738105 FIX: Create Cache tool bug fix.

738104 FIX: NHibernate integration requires complete DateTime format for absolute expiration which is wrong in. Now, you can specify absolute expiration in terms of seconds.

738105 FIX: There was a serialization bug in Session Store Provider which occurs in rare scenarios only.

738103 FIX: SQL Dependency bug fix in NHibernate where client was unable to use queries with composite key.

738102 FIX: Client cache does not work with NHibernate Integration.

738101 FIX: GetByTag performance is slow for large number of items.

736230 FIX: Object query returns empty result if the item was automatically reloaded (`IsResyncExpiredItem` is true) through the ReadThru provider.

APPLIES TO:

- NCache Enterprise Edition 3.8
- NCache Professional Edition 3.8

Release Notes NCache 3.8

Monday, June 7, 2010

Introduction

NCache 3.8 contains new important Additions and enhancements based on the customer's feedback. The API backward compatibility is the most important concern for most of the customers and keeping this in mind we have added a new protocol for API compatibility. From now (3.8) onwards all the new releases will be automatically compatible with older versions of NCache (starting from 3.8).

Enhancements and New Additions

Following features are added in this release.

1. **LINQ Support in NCache**
   NCache provides LINQ integration with the help of **IQueryable** interface which allows the cached items to be searchable. NCache support both Lambda Expressions and LINQ operators for querying cached items.

2. **Entity Framework (EF) Caching**
   NCache provides seamless integration with EF caching where it gets plugged-in at the ADO level and lets your application use distributed caching without any code change. You only need make changes in the application configuration file.

3. **.NET Cache Provider 4.0**
   NCache provides integration with .NET cache provider 4.0. NCache also provides different **Change Monitors (file based, key based, database dependency)** for managing cache dependencies.

4. **Backward Compatibility Client API / Support**
   NCache now follows backward compatibility protocol and in future version application will be able to connect to newer versions without upgrading the clients.

5. **Streaming API**
   NCache has added streaming support in the API where you can read and write binary data stream in the cache.

6. **Java / CSS minification**
   NCache combines multiple JavaScript files and CSS files into a single resource file and store it in the cache. It also replaces the rendered output with single HTTP reference for all CSS and JS files so that browser can make a single call for loading all the resources. This helps improve your application response time.

7. **ViewState Caching**
   NCache replaces the long ViewState string into a smaller one and sends it to the client. This helps improve the application performance and save bandwidth.

8. **New NCache Monitoring Tool**

9. **NCache Email Alert System**
   You can now receive alert through emails on certain cache events like "State Transfer", Cache Stopped, Member Left, Member Joined etc.,

10. **Cache Meta information API**

This API allows Meta information about cache items like LastAccessed Time and Creation DateTime.

11. Multi-ReadThru / Write Thru support

NCache allows multiple readthru and writethru providers. NCache Manager automatically deploys the data source assemblies into the deployed folder so you don't have to manually copy the provider assemblies in NCache service folder.

12. Partitioned Replica Synchronous replication

Partitioned Replica Topology now supports Synchronous replication

13. Security Configuration command line tool

You can now configure security through command line tool

14. Cache Config Upgrade tool

If you have an old cache config.ncconf file then you can upgrade it by using this tool.

15. Locking support with GetCacheItem

16. Remote Client Management from NCache Manager

Now you can configure Remote Clients from NCache Manager and can easily change individual client settings all from a single point.

17. Client-Cache Management from NCache Manager

Client Cache can also be managed from NCache Manager.

18. Client / Client Cache management command line tools

You can also add remote clients and client-cache through command line tools.

19. New interface for ReadThru / WriteThru providers.

Now you can specify your cache provider assemblies through a new wizard where you can pick your assembly. Now, you don't have to type assembly information manually.

20. Automatic deployment of ReadThru / WriteThru providers

NCache Manager provides the automatic deployment of data source providers.

21. Dynamic Compact Serialization Support (no code change required)

Now you don't need to implement any ISerializable or IDeserializable Interfaces for compact serialization. You only need to register the compact types in NCache Manager and NCache automatically serialize the types at runtime.

22. Security Enhancement (Security Management from NCache Manager)

Security can be configured from NCache Manager now.

Release Notes NCache 3.6.2 (Service Pack 3)

Thursday, April 21, 2010

Introduction

NCache 3.6.2 Service Pack 3 (SP3) contains important fixes and enhancements that have been reported by customers. The most important and demanded feature added in this release is the support of Java session caching for J2EE platforms. Now, you can easily use NCache as a session store for your Java applications without making any code change. You only need to add a Java session filter in your web.xml file.

Those using NCache 3.6.x version can upgrade without re-building/re-compiling the application.

Enhancements and New Additions

Following features are added to this service pack 3.

1. **Java Session Support for WebLogic (no-code change required)**
   NCache Java Session Provider is supported for WebLogic and can be used on any platform of J2EE (web servers), if underlying platform (web server) follows the Java Servlet 2.3 (or later) specification. Java Sessions are supported on both Linux and Windows platforms.

2. **Java Session Support for JBoss (no-code change required)**
   NCache Java Session Provider is supported for JBoss on both Linux and Windows platforms.

3. **Java Session Support for WebSphere (no-code change required)**
   NCache Java Session Provider is supported for WebSphere on both Linux and Windows platforms.

4. **Java Session Support for ApacheTomcat (no-code change required)**
   NCache Java Session Provider is supported for Apache-Tomcat on both Linux and Windows platforms.

5. **NCache support for Server GC**
   There are two flavors of Garbage Collector based on server operating system and workstation called “Server GC” and “Workstation” respectively. Unless it is specified .NET framework uses workstation GC by default even it is running under server operating system. Now, NCache allows you to specify GC mode depending on your operating system.

   Server GC is designed for maximum throughput, and scales with very high performance. NCache is now by default configured to use Server GC instead of Workstation GC. You can change this mode from Alachisoft.NCache.Service.exe.config

   ```
   <add key="NCacheServer.EnableForcedGC" value="true"/>
   <add key="NCacheServer.ForcedGCThreshold" value="80"/>
   ```

List of Bugs Fixed:

736228 FIX: NCache installation fails on Windows 7 because of new security model in Windows 7. This has been fixed now.

736227 FIX: There was a bug in Client Queue Counter which becomes negative at some particular stage. This bug has been fixed.

736226 FIX: There was a bug in item expiration in Java API which has been fixed now.

APPLIES TO:

- NCache Enterprise Edition 3.6.x

Release Notes NCache 3.6.2 (Service Pack 2)

Monday, Jan 11, 2010

Introduction

NCache 3.6.2 Service Pack 2 (SP2) contains important fixes and enhancements that have been reported by customers. It is a recommended upgrade for all users of NCache. Those using NCache 3.6.x version can upgrade without re-building/re-compiling the application.

Enhancements and New Additions

Following features are added to this service pack.

1. **Logging of Important events into event viewer**

   An event will be logged in windows event viewer and cache-error log when

   - you stop or start a cache
   - cache fails to start
   - cache node joins/leaves a cluster
   - state transfer start or stop.
   - client connect or disconnect a cache
   - cache memory crosses a certain threshold specified in service config file.
   - you are on last 10 days of NCache evaluation and a log warning event will be logged per day until you extend the evaluation or activate NCache with a license key.

   These events are categorized as information, warning and information.

   ```
   <add key="NCacheServer.EventLogLevel" value="error | warning| all"/>
   ```

   User can specify only one of the following levels:

   error: Only the error events will be logged
   warning: Both error and warning events will be logged.
   all: This level allows events of all categories to be logged. This is the default level.

   Client Connection Log entry for event viewer/log-file can be enabled from NCache Service configuration file. By default this option is disabled but you can enable it by modifying the following attribute in NCache service config file "Alachisoft.NCache.Service.exe.config".

   ```
   <add key="NCacheServer.LogClientEvents" value="false" />
   ```

   NOTE: The above change requires NCache service to restart.

2. **Oracle 11g is supported**

   NCache now supports Oracle 11g for database dependency and synchronization feature.

List of Bugs Fixed:

736225 FIX: Some of NCache clients stop responding when the cache cluster is under stress and you add a new caching server or the cache servers are using shared network card on VM.

736224 FIX: NCache service (Alachisoft.NCache.Service.exe) and NCache Manager stop responding when the cache is under stress and you add a new caching server or the cache servers are using shared network card on VM.

736223 FIX: NCache service stops responding due to Hashmap corruption. This is similar to the above mentioned service hanging scenario, however in this scenario, service hangs for indefinite time.

736222 FIX: Cache stops expiring items from the cache if a WMI call does not respond for a long period of time because of WMI unresponsive state.

736221 FIX: Client unbalancing in Partitioned and Partitioned Replica on node joining is fixed now. This only happens in very rare situations.

736220 FIX: Clients were not able to get real performance benefits of client-optimization feature of Partitioned Replica if the clients are 32-bit and cache is 64bit or vice versa.

736219 FIX: Cache state transfer (starts when a cache node joins the cluster) does not transfer those items which have multi-level key dependent items.

736218 FIX: Java client environmental issues have been fixed. Now you can pass client.ncconf file path from API. Java client now supports both Windows and Linux environment.

736217 FIX: Java sample application bugs are fixed now

736216 FIX: NCache service does not start if the socket call fails to bind itself network card. NCache now tries to establish the connection without binding.

736215 FIX: When you unplug the network cable Client hanging issue when cable is unplugged during stress is fixed now. This normally happened in replicated topology and the client operations were hanged for sometime.

736214 FIX: NCache tag query throws “Null Reference” in partitioned and partitioned replica topologies when used with LIKE operator and the tag does not exist.

736213 FIX: NCache throws “Data group mismatch” exception when you update and item by calling INSERT method with locking and group parameters.

APPLIES TO:

- NCache Enterprise Edition 3.6.x
- NCache Professional Edition 3.6.x

Release Notes NCache 3.6.2 (Service Pack 1)

Tuesday, Oct 06, 2009

Introduction

NCache 3.6.2 Service Pack 1 (SP1) contains important fixes and enhancements that have been reported by customers. It is a recommended upgrade for all users of NCache. Those using NCache 3.6.x version can upgrade without re-building/re-compiling the application.

Enhancements and New Additions

Following features are added to this service pack.

1. **Cluster Rejoining feature**

   To overcome socket breaking issues within cluster nodes, NCache has a new connection retry logic which is configurable in config.ncconf in cluster configuration.

   ```
   connection-retries="10" connection-retry-interval="1secs"
   ```

   In case a socket breaks due to Network problems, NCache will try to re-establish the connection after every connection-retry-interval as many times as connection-retries. This requires a restart of NCache service after update.

2. **Addition of NCache Client side counters**

   NCache client side counters are introduced to detect and debug client side issues. These counters appear in Windows perfmon counters in category 'NCache Client'.

3. **Asynchronous startup of caches in Autostart**

   NCache autostart feature is enhanced and starting up of various user specified caches with NCache service start up, is made asynchronous. If some caches are corrupt and are not started, a warning is logged but NCache service starts up normally.

4. **Client-cache-sync mode description**

   Client cache sync mode description is added in NCache Help.

5. **Memory estimation and warning mechanism in cluster configuration**

   Description about setting up cluster memory, its usage and estimation is added in cluster configurations.

6. **NCache Installation in Admin mode**

   NCache installation wizard warns the user who is not an admin. Administrative privileges and permissions are required for the user to install NCache on a system.

List of Bugs Fixed:

736212 FIX: Items with same Tags if added and removed frequently resulted in corrupting the tag. Items exist in the cache but not gettable with their tag.

736210 FIX: The memory leakage related with LOH results in high memory consumption in case of large objects (more than 80KB) which are not garbage collected.

736209 FIX: NCache generates extra log entries in NCache log files which result in huge log files. These extra logs are removed.

APPLIES TO:

- NCache Enterprise Edition 3.6.x
- NCache Professional Edition 3.6.x

Release Notes NCache 3.6.2

Thursday, July 09, 2009

Introduction

NCache 3.6.2 contains a number of useful enhancements and bug fixes that have been reported by customers. It is a recommended upgrade for all users of NCache. Those using NCache 3.6.x version can upgrade without re-building/re-compiling the application.

Enhancements and New Additions

Following features are added to this version.

1. **Log Traces**
   Log traces are added for the following events

   1. When a node successfully joins a node in the cluster.
   2. When a node can not join a node in the cluster.
   3. When a node leaves the cluster.

2. **Polling based database dependency improvements**
   Polling based database dependency has the following improvements.

   4. User can specify the 'db-cache-key' other than the 'cache-key' to add to the ncache_db_sync table.
   5. Resync-expired-items option is now available for this dependency.

3. **Client.ncconf can be opened in notepad**
4. **Improved Error messages**
   Error messages are improved when service fails to start because of unavailable bind-to-ip addresses.

List of Bugs Fixed:

736208 FIX: Expiration problem that arises in around 20-30 days of using NCache results in items occupying memory leading to high memory usage.

736207 FIX: Client.ncconf is rewritten by NCache service restart and old changes are lost.

736206 FIX: Groups and tags information is not preserved if the items are read-thru from a datasource. Items are loaded in the cache but do not have the tag or group already specified.

736205 FIX: IIS worker process high CPU consumption bug is fixed

---

APPLIES TO:

- NCache Enterprise Edition 3.6.x
- NCache Professional Edition 3.6.x

Release Notes NCache 3.6.1

Thursday, March 26, 2009

Introduction

NCache 3.6.1 contains a number of useful enhancements and bug fixes that have been reported by customers. It is a recommended upgrade for all users of NCache. Those using NCache 3.6.x version can upgrade without re-building/re-compiling the application.

Enhancements and New Additions

Following features are added to this version.

1. **NCache client rebalancing among cache server nodes**

   Now the NCache client connections are automatically rebalanced among the cluster nodes when a new node joins the cache. So now you do not have to worry any more about client load balancing on cache servers.

2. **New Counters for performance monitoring**

   New counters for performance monitoring are added in the 'NCache' category in Windows Perfmon counters.

3. **Improvements in DB dependency**

   NCache now supports the use of Stored-Procedures and Command Object in DB Dependency.

List of Bugs Fixed:

736204 FIX: NCache service constantly consumes CPU if there is one item left in the cache with the SQL Yukon dependency and the dependency is triggered.

736203 FIX: AVG counters show incorrect values for Average Add/sec, Update/sec and Remove/sec.

736202 FIX: Memory leakage and performance issue related with Least Recently Used and Least Frequently Used eviction policies (LRU/LFU).

736201 FIX: Bind to IP Dialog corrupts the display if there are more than 2 network cards on a system.

736200 FIX: NCache Service consumes CPU at some fixed interval even when there is no activity on the cache after a load test is stopped after running for hours.

APPLIES TO:

- NCache Enterprise Edition 3.6.x
- NCache Professional Edition 3.6.x