Oracle® Fusion Middleware

Release Notes for Oracle Coherence

12*c* (12.2.1.3.0)

E80352-04

April 2018

Oracle Fusion Middleware Release Notes for Oracle Coherence, 12*c* (12.2.1.3.0)

E80352-04

Primary Author: Joseph Ruzzi

Contents

Preface

Release Notes for Oracle Coherence summarizes the release information related to new and updated features, fixed issues and their workaround, deprecated and removed functionality, and more.

This preface includes the following topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

This document is intended for users of Oracle Coherence.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Coherence documentation set:

- *Administering Oracle Coherence*
- *Administering HTTP Session Management with Oracle Coherence*Web*
- *Developing Applications with Oracle Coherence*
- *Developing Remote Clients for Oracle Coherence*
- *Installing Oracle Coherence*
- *Integrating Oracle Coherence*

- *Managing Oracle Coherence*
- *Securing Oracle Coherence*
- *Java API Reference for Oracle Coherence*
- *.NET API Reference for Oracle Coherence*
- *C++ API Reference for Oracle Coherence*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

You can use the Oracle Coherence Release Notes to learn about important production information such as Coherence certifications, support, and licensing.
This chapter contains the following sections:

- Latest Release Information
- Purpose of this Document
- System Requirements and Specifications
- Certification Information
- Product Documentation
- Oracle Support
- Licensing Information
- Downloading and Applying Required Patches

1.1 Latest Release Information

This document is accurate at the time of publication. Oracle will update the release notes periodically after the software release. You can access the latest information and additions to these release notes at Oracle Help Center.

1.2 Purpose of this Document

This document contains the release information for Oracle Fusion Middleware Release for Oracle Coherence.

Oracle recommends you review its contents before installing, or working with the product.

1.3 System Requirements and Specifications

Oracle Coherence follows the Fusion Middleware system requirements and certifications for production environments. For more information, see http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html.

For system requirements for installations of development environments, visit:

- Coherence for Java — *Installing Oracle Coherence for Java*.
- C++ Client — *Installing the C++ Client Distribution*.
- .Net Client — *Installing the .NET Client Distribution*.

1.4 Certification Information

To see versions of platforms and related software for which Oracle Coherence is certified and supported, go to http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html.

1.5 Product Documentation

For complete documentation on Oracle Coherence, go to Oracle Help Center.

1.6 Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support .

1.7 Licensing Information

Detailed information regarding license compliance for Oracle Fusion Middleware is available at Licensing Information.

1.8 Downloading and Applying Required Patches

Go to `My Oracle Support` to download the latest software patches.

See the `README` file in the patch distribution for up-to-date information on the software fixes provided by the patch.

To download and install the latest software patch:

1. Login to My Oracle Support.
2. Click the **Patches & Updates** tab.
3. Under the Patch Search tab, select **Product or Family (Advanced Search)**, and select the **Include all patches in a product family** check box.
4. Enter **Oracle Coherence** as the product, select the platform and release, and click **Search**.

The list of currently available patches for Oracle Coherence is returned.

2

What's New in this Release

Learn the features, enhancements, and changes made to Oracle Coherence.Oracle updates the release notes periodically after the software release. This document is accurate at the time of publication.
This chapter includes the following sections:

- New Features
- Deprecated Features

2.1 New Features

This section contains new features for Oracle Coherence that are organized by release. The features in this release help reduce complexity, ease configuration, and accelerate time to market of scalable solutions.

New and Improved for 12.2.1.3.0

- **Revised Active-Active Federation Configuration** – The new `<active-active>` configuration element is used to define active-active federation topologies. See Defining Active-Active Topologies in *Administering Oracle Coherence*.
- **Global Consistent Snapshots** – A snapshot can be created either on a running service (a service that is accepting and processing requests) or on a suspended service. The former provides consistency at a partition level while the latter provides global consistency. See Create a Snapshot in *Administering Oracle Coherence*.
- **HotCache Enhancements** – Coherence includes new HotCache functionality. The functionality includes:
  - **HotCache multi-threading** – HotCache can use multiple threads to apply trail file operations to Coherence caches. Multiple threads can increase the throughput of a HotCache process as compared to using a single thread to apply trail file operations. See Configuring Multi-Threading in HotCache in *Integrating Oracle Coherence*.
  - **HotCache Multitenant mode** – HotCache can refresh caches for specific tenants. In Multitenant mode, a tenant identifier is used in conjunction with a Coherence scope name to ensure that HotCache only refreshes caches for a specified tenant. See Using HotCache Multitenant Mode in *Integrating Oracle Coherence*.
  - **HotCache JMX management** – HotCache includes management data for monitoring the performance of cache update operations. See Managing HotCache in *Integrating Oracle Coherence*.

New and Improved for 12.2.1.2.0

- **Custom Federation Participants** – Custom participants allow applications to federate data to multiple non-Coherence end-points or resources. For example, cache data can be federated to a database, message bus, log file, and so on.

Custom participants are implemented as event interceptors and are configured as part of a federation topology. See Federating Events to Custom Participants in *Administering Oracle Coherence*.

- **HotCache Enhancements** – Coherence includes new HotCache functionality. The functionality includes:
  - HotCache JPA properties – JPA properties can be used to configure HotCache behavior. See Configuring HotCache JPA Properties in *Integrating Oracle Coherence*.
  - Warming Caches – HotCache can be used to warm caches by loading an initial dataset. See Warming Caches with HotCache in *Integrating Oracle Coherence*.
  - Support for Oracle `SDO_GEOMETRY` and `XMLType` types – Oracle-specific data types that are supported by EclipseLink can be used by HotCache. See Support for Oracle Data Types in *Integrating Oracle Coherence*.

New and Improved for 12.2.1.1.0

- **Dynamic Active Persistence Quorum Policy** – The dynamic recovery quorum policy is used with active persistence and automatically configures the persistence recovery quorum based on a predefined algorithm. See Using the Dynamic Recovery Quorum Policy in *Administering Oracle Coherence*.
- **Simplified Federation Participant Configuration** – An address port is no longer required when configuring Federation participants. See Defining Federation Participants in *Administering Oracle Coherence*.
- **Federation Support for Read-Through Caching** – Cache entries that are loaded into a cache using a `CacheStore` implementation can now be federated across clusters. See Plugging in a Cache Store Implementation in *Developing Applications with Oracle Coherence*.
- **Updated Federation Management** – The `DestinationMBean` MBean now includes management information for `replicateAll` operations. See DestinationMBean in *Managing Oracle Coherence*.
- **Federating HTTP Sessions** – HTTP Session caches can now be federated. See Federated Session Caches in *Administering HTTP Session Management with Oracle Coherence*Web*.
- **Disabling Federation** – Federated caching can be disabled for specific caches. See Excluding Caches from Being Federated in *Administering Oracle Coherence*.
- **Coherence Session API** – The Coherence `Session` API provides applications with a new way to get a reference to a `NamedCache` instance. See Getting a Cache Instance in *Developing Applications with Oracle Coherence*.
- **HTTP Acceptor Management** – Management information is now provided for proxy servers and includes the number of connections across each proxy server and the total messages that were sent and received. Management attributes are included on the `ConnectionManagerMBean` MBean and are viewable on the Proxy HTTP Report and the HTTP Servers tab in the Coherence-JVisualVM Plug-in. See Understanding the Proxy HTTP Report in *Managing Oracle Coherence*.
- **Two Member Partition Assignment** – The default partition assignment strategy now uses an active-passive distribution algorithm for clusters with only two storage-enabled members. The algorithm provides optimal distribution for SEOne

use cases. See Changing the Partition Distribution Strategy in *Developing Applications with Oracle Coherence*.

- **Simplified Persistence and Federation in WebLogic Server** – Persistence and federation can now be configured in the WebLogic Server Administration Console or using WebLogic Server MBeans. See Configuring Cache Persistence and Configuring Cache Federation, respectively in *Administering Clusters for Oracle WebLogic Server*.
- **Zero Down-Time Support** – Coherence Applications that are deployed in WebLogic Server can now participate in zero down-time patching. See the `options` argument in Using WLST to Initiate and Monitor Workflows in *Administering Zero Downtime Patching Workflows*.
- **IBM WebSphere Liberty Support** – Coherence*Web supports IBM WebSphere Liberty 8.5 and higher. Support for all other IBM WebSphere versions has been removed. See Supported Web Containers in *Administering HTTP Session Management with Oracle Coherence*Web*.

New and Improved for 12.2.1.0.0

- **Persistence** – Coherence persistence is a set of tools and technologies that manage the persistence and recovery of Coherence distributed caches. Cached data is persisted so that it can be quickly recovered after a catastrophic failure or after a cluster restart due to planned maintenance. See Persisting Caches in *Administering Oracle Coherence*.
- **Federated Caching** – Federated caching replicates and synchronizes cache data across multiple geographically dispersed clusters. Cached data is replicated across clusters to provide redundancy, off-site backup, and multiple points of access for application users in different geographical locations. See Replicating Caches Across Clusters in *Administering Oracle Coherence*.
- **Security Enhancements** – Coherence includes new security functionality. The functionality includes:
  - Audit logs – Audit logs are used to record user access to cluster operations. See Enable Security Audit Logs in *Securing Oracle Coherence*.
  - Access control authorization – Access control authorization allows applications to define their own authorization logic to limit access to cluster operations. See Authorizing Access to Server-Side Operations in *Securing Oracle Coherence*.
  - Additional JAAS permissions – JAAS permissions protect various parts of the Coherence API using the Java Security Manager. See Programmatically Specifying Local Permissions in *Securing Oracle Coherence*.
  - SSL Protocols and Cipher Suites – An SSL socket provider can be configured to control the use of potentially weak ciphers or specific protocol versions. See Controlling Cipher Suites and Protocol Version Usage in *Securing Oracle Coherence*.
- **Support for Java 8 Features** – The Coherence API has been updated so that you can use programming features that were introduced in the Java 8 release. The features include lambda, streams, and default methods. These features provide ease of use and flexibility when performing data grid operations. See Support for Java 8 Features in *Developing Applications with Oracle Coherence*.
- **Support for Generics** – Java Generics provide compile and runtime type checking together with compile type-inference. The Coherence API has been refactored to support Java Generics. In addition, types can be explicitly configured

as part of the `NamedCache` API. See Support for Generics in *Developing Applications with Oracle Coherence*.

- **Cluster Port and Address Management** – Port and address selection has been changed to simplify cluster management and to allow the sharing of cluster ports and addresses among multiple clusters. For most use cases, ports and addresses do not need to be explicitly set.
  - Coherence port selection has been simplified to facilitate port management. Coherence now includes a common cluster port that is used for multicast communication, well known addresses, and extend proxies that are using the name service. In addition, unicast ports are automatically assigned. See Setting Up a Cluster in *Developing Applications with Oracle Coherence* and Configuring Extend Proxies in *Developing Remote Clients for Oracle Coherence*.
  - Coherence address selection has been simplified to facilitate address management. For unicast communication, Coherence automatically selects a routable IP with the highest MTU for computers that have multiple IPs or NICs. For well known addresses, Coherence selects the IP which is routable to the IPs on the WKA list. In addition, Coherence can now automatically resolve IP addresses (at runtime) that are associated with a DNS name. Well known addresses and proxy addresses can be stored in a DNS server and centrally managed and updated in real time. See Setting Up a Cluster in *Developing Applications with Oracle Coherence*.
- **Multitenancy** – Coherence applications that are deployed using managed Coherence servers can take full advantage of the density and operational efficiencies that are provided by Weblogic Server Multitenant. Coherence features include: isolating caches across domain partitions, sharing caches across domain partitions, and tooling support exposed through WLST and Fusion Middleware Control. See Configuring Coherence in *Using Oracle WebLogic Server Multitenant*.
- **Asynchronous NamedCache API** – The `AsyncNameCache` interface allows cache operations to be completed in parallel and can improve throughput and result in more responsive user interfaces. See Performing NameCache Operations Asynchronously in *Developing Applications with Oracle Coherence*.
- **Truncate Operation** – The `truncate` method on the `NamedCache` interface clears a cache but does not raise any entry-level cache events. This new API significantly reduces the memory pressure on the server side and dramatically reduces the network consumption for caches with listeners and is especially beneficial for near caching. See Clearing Caches in *Developing Applications with Oracle Coherence*.
- **Sliding Expiry** – Sliding expiry extends the expiry of cache entries that are being accessed. Sliding expiry is enabled by default for cache entries that are updated, but can also be enabled based on read operations and operations that are non-mutative. See Capacity Planning in *Developing Applications with Oracle Coherence*.
- **Dynamic Management Mode** – Dynamic management elects one of the nodes in the cluster to be the management node. This is the default management mode for managed Coherence servers. Cluster members no longer need to be explicitly configured for JMX management. See Using Dynamic Management Mode in *Managing Oracle Coherence*.
- **Dynamic Thread Pools** – All Coherence services use dynamically sized daemon thread pools. Dynamic thread pools are enabled by default and are configured for

a service using the `<thread-count-max>` and `<thread-count-min>` elements when defining a cache scheme.

- **Coherence-JVisualVM Plug-In** – A new version of the Coherence-JVisualVM plug-in is available and includes new functionality. The functionality includes: reporting node state, displaying near cache statistics, displaying partition statistics, managing cache persistence, and managing federated caching. See Using the Coherence-JVisualVM Plug-In in *Managing Oracle Coherence*.
- **Coherence CohQL** – New functionality is included in Coherence CohQL. The functionality includes: support for loading GAR modules, support for WLS Multitenant, and support for cache persistence. See Using Coherence Query Language in *Developing Applications with Oracle Coherence*.
- **Server-Sent Events** – Server-sent events allow Coherence REST applications to automatically receive cache events from the Coherence cluster. See Using Server-Sent Events in *Developing Remote Clients for Oracle Coherence*.
- **Log4J 2 support** – Coherence logging can be configured to use Log4J 2 logging. See Using Log4J 2 Logging for Coherence Logs in *Developing Applications with Oracle Coherence*.

2.2 Deprecated Features

This section describes the deprecated and desupported features of Oracle Coherence.

This section includes the following topics:

2.2.1 BACKUP CACHE and RESTORE CACHE

The `BACKUP CACHE` and `RESTORE CACHE` commands in CohQL are deprecated. A new set of snapshot commands has been provided for use with the new the persistence feature. See Persisting Cache Data to Disk in *Developing Applications with Oracle Coherence*.

2.2.2 Managed Coherence Server MBean Attributes

The following MBean attributes are deprecated in Managed Coherence Server:

- `CoherenceClusterParamsBean.UnicastListenPort`
- `CoherenceClusterParamsBean.UnicastPortAutoAdjust`
- `CoherenceClusterParamsBean.MulticastListenPort`
- `CoherenceClusterWellKnownAddressBean.ListenPort`

The following deprecated MBean attribute is removed:

- `CoherenceClusterParamsBean.UnicastListenAddress`

2.2.3 Coherence*Web Container Support

Coherence*Web no longer supports the following web containers: Apache Tomcat 5.5.*n*, Apache Tomcat 6.0.*n*, Caucho Resin 3.1.*n*, IBM WebSphere 5.*n*, IBM WebSphere 6.*n*, IBM WebSphere 7.*n*, Sun GlassFish 2.*n*, Sun Application Server 8.*n*, Oracle OC4J 10.1.3.*n*, Oracle OC4J 10.1.2.*n*, Oracle GlassFish 3.*n*, Oracle GlassFish 4.*n*, Jetty 6.1.*n*, Jetty 5.1.*n*, JBoss Application Server. Applications that require Coherence HTTP session management must be migrated to use a supported web container version. See *Administering HTTP Session Management with Oracle Coherence*Web*.

2.2.4 ActiveCache (active-cache.jar)

ActiveCache (`active-cache.jar`) has been deprecated. ActiveCache can still be used with applications that have been developed to run on older versions of WebLogic Server.

ActiveCache functionality has been replaced by Managed Coherence Servers. For more information on Managed Coherence Servers, see *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*.

2.2.5 Thread Count

The `<thread-count>` element is deprecated. Use the `<thread-count-min>` and `<thread-count-max>` elements instead.

2.2.6 Specifying Ports in the WKA List

The functionality to specify ports in the WKA list is deprecated as of the of the 12.2.1 release. Support for this feature will be removed in a future release.

2.2.7 Specifying `tangosol` in System Properties

Coherence system property names no longer require the `tangosol` prefix. For example, the system property `tangosol.coherence.distributed.localstorage` can now be written as `coherence.distributed.localstorage`. System properties that only contained the `tangosol` prefix now use the `coherence` prefix. For example, the system property `tangosol.pof.enabled` can now be written as `coherence.pof.enabled`. The changes are also applicable to Unix-based environments. For example `TangosolCoherenceCacheConfig` can be written as `CoherenceCacheConfig`.

System properties that include the `tangosol` prefix are still supported; however, support may be removed in a future release.

2.2.8 TopLink Implementations

The `TopLinkGrid` as well as the `TopLinkCacheLoader` and `TopLinkCacheStore` implementations are deprecated as of the of the 12.2.1 release.

2.2.9 Object::toStream Deprecated

The `Object::toStream` method has been deprecated. Applications should use the `Object::toString` method instead.

2.2.10 ParallelAwareAggregator

The `ParallelAwareAggregator` interface has been deprecated and should no longer be used. Applications should use the `StreamingAggregator` interface to implement custom aggregators. See Performing Data Grid Aggregation Using Streams in *Developing Applications with Oracle Coherence*.

2.2.11 Remote Addresses

The `<remote-addresses>` element, within the `<participant>` element, that is used to configure federation cluster participants in an operational configuration file is deprecated. Use the `<name-service-addresses>` element instead.

2.2.12 Encryption Network Filters

Encryption network filters, which were deprecated in Coherence 3.7, have been removed and are no longer supported as of the 12*c* (12.2.1) release. TCMP inter-cluster and *Extend TCP connections should be secured using SSL instead. The following APIs have been removed:

- `com.tangosol.net.security.AbstractEncryptionFilter.java`
- `com.tangosol.net.security.AsymmetricEncryptionFilter.java`
- `com.tangosol.net.security.BlockCipherInputStream.java`
- `com.tangosol.net.security.BlockCipherOutputStream.java`
- `com.tangosol.net.security.ClusterEncryptionFilter.java`
- `com.tangosol.net.security.PasswordBasedEncryptionFilter.java`
- `com.tangosol.net.security.SymmetricEncryptionFilter.java`

2.2.13 C++ libraries for Solaris

Starting 12.2.1.3, the C++ libraries for Solaris SPARC32 and Solaris Intel x86 are no longer supported with Coherence. In 12.2.1.3, only the libraries for SPARC64 and Solaris x64 are supported.

If you need Solaris SPARC32 or Solaris Intel x86 (32 bit) libraries, then use the fully compatible Coherence for C++ 12.2.1.2 libraries.

3

Known Issues and Workarounds

Learn about the known issues at the time of release.
This chapter includes the following sections:

- SSL Restrictions Cause Cluster Failure
- Changing the Partition Count When Using Active Persistence
- Impact of Generics
- Support for JVisualVM Plugin

3.1 SSL Restrictions Cause Cluster Failure

Issue

When upgrading to Java 8 Update 151 (8u151) or higher, a cluster may fail due to tighter security checks on algorithm constraints. The following error is emitted:

```
Error generating DH server key exchange; The security strength of SHA-1 digest
algorithm is not sufficient for this key size
```

Workaround

Regenerate SSL certificates and explicitly set the key algorithm and key size. For example:

```
keytool -genkeypair -keyalg RSA -keysize 2048 -dname "cn=administrator,
ou=Coherence, o=Oracle, c=US" -alias admin -keypass password -keystore /test/
server.jks -storepass password -validity 180
```

3.2 Changing the Partition Count When Using Active Persistence

Issue

The partition count cannot be changed when using active persistence. If you change a services partition count, then on restart of the services all active data is moved to the persistence trash and must be recovered after the original partition count is restored. Data that is persisted can only be recovered to services running with the same partition count.

Ensure that the partition count is not modified if active persistence is being used. If the partition count is changed, then a message similar to the following is displayed when the services are started:

```
<Warning> (thread=DistributedCache:DistributedCachePersistence, member=1):
Failed to recover partition 0 from SafeBerkeleyDBStore(...); partition-count
mismatch 501(persisted) != 277(service); reinstate persistent store from
trash once validation errors have been resolved
```

The message indicates that the change in the partition-count is not supported and the current active data has been copied to the trash directory.

Workaround

To recover the data:

1. Shutdown the entire cluster.
2. Remove the current active directory contents for the cluster and service affected on each cluster member.
3. Copy (recursively) the contents of the trash directory for each service to the active directory.
4. Restore the partition count to the original value.
5. Restart the cluster.

3.3 Impact of Generics

When using generics with Federated Caching and Persistence, you must:

- Ensure that federated caches across a federation are configured to use the same types as no runtime-type-checking is performed between clusters in a federation.
- Ensure that recoverable caches are consistently configured to use the same types during restarts as no runtime-type-checking is performed.

3.4 Support for JVisualVM Plugin

The Coherence JvisualVM Plugin is supported for Coherence 3.7.1.X and above. Using the plugin to connect to older clusters is not supported.

4 Bugs Fixed and Enhancements in this Release

Learn about the bugs fixed and enhancements in this release.
This chapter includes the following sections:

- Oracle Coherence for Java
- Oracle Coherence for .NET
- Oracle Coherence for C++

4.1 Oracle Coherence for Java

New features, improvements, and bug fixes added to Oracle Coherence for Java components.

Enhancements and Fixes for 12.2.1.3.0

- Added MBean support to collect heap dumps from `ClusterMBean` and `ClusterNodeMBean`.
- Added an `active-active` topology element to the topology-definitions for defining federation `active-active` topologies.
- Added the `ContainerPassThroughResourceConfig` class to allow JSON passthrough (using REST) to work correctly when in a container environment such as WebLogic Server.
- Added support for listen addresses to be configured with non-local NAT addresses.
- Added automatic detection of persistence targeting remote file-systems and auto-enabled `je.log.useODSYNC` to ensure database integrity in such environments.
- Coherence command line utilities with `JLine` support now use `JLine2`.
- Fixed an issue where having multiple local IPs on the same subnet can prevent nodes from joining the cluster.
- Fixed an issue where accessing a cache after a service restart fails with an `IllegalStateException` in managed Coherence servers.
- Fixed a memory leak issue in the bulk key listener (de)registration which impacts `NearCaches`.
- Fixed an issue where setting an expiry against an entry in `SerializationCache` may be ignored, thus potentially affecting the uses of an external-scheme.
- Fixed an issue where extra persistence configuration parameters are not propagated to the persistence implementation.
- Fixed an issue where using a class-scheme in the cache configuration containing both scheme-name element and custom namespace element causes the custom namespace element to be ignored.

- Fixed an issue when a service is abruptly restarted and attempted to be accessed via a `MapListener`.
- Fixed an issue where `ConcurrentModificationException` during the `addIndex` operation causes client requests and server failover to hang.
- Fixed an issue where two node clusters use the standard partition distribution strategy instead of the `SE One` active/standby mode, by default.
- Fixed an issue where `NullPointerException` can be thrown during an `addIndex` operation.
- Fixed an issue where the partition distribution can be stuck due to an unreconcilable ownership conflict
- Fixed an issue where `IllegalArgumentException` can be thrown when destroying a cache concurrently with partition transfer.
- Fixed an issue where the `IllegalArgumentException` with `unknown extent identifier` can be thrown when using federation with persistence.
- Fixed an issue where the default operational configuration was being ignored instead of being handled hierarchically when a custom operational configuration was specified.
- Fixed the performance of cache inserts and updates when there are multiple filter based event listeners.
- Fixed an issue where the cache prune during redistribution can result in termination of the service.
- Fixed an issue with `DaemonPool` where a failure is triggered when the number of executed tasks exceeded 2,147,483,647.
- Fixed an issue where the expiry was inadvertently encoded causing an unexpected eviction to the backup and persistent copies.
- Fixed an issue where a Coherence Java streams implementation can cause null results.
- Fixed an issue where `ensureCache` called during recovery from persistence was resulting in `NullPointerExceptions`.
- Fixed a regression where an `EvictionPolicy` calling a read method on the backing map can cause a stack overflow.
- Fixed a performance issue with proxy nodes processing `MemberLeft` service notifications.
- Fixed an issue where the federation `replicateAllTotalTime` and `replicateAllPercentComplete` values can be calculated correctly.
- Fixed an issue where the initial state setting for a federation destination may not be used if storage disabled Coherence nodes are running prior to starting any storage enabled nodes
- Fixed the persistence mode system property to be `coherence.distributed.persistence-mode` instead of `coherence.distributed.persistence.mode`. However, `coherence.distributed.persistence-mode` is still used for backward compatibility.

- Fixed an issue where Coherence fails to start if a federation custom participant is defined without a corresponding interceptor. Coherence now starts and logs a warning message about the missing interceptor.
- Fixed an issue where the `ReplciateAllPercentComplete` attribute in the `DestinationMBean` of a federated cache in some nodes may stay at zero.
- Fixed an issue where the ephemeral port bindings to 65535 resulted in `Address already in use` `BindExceptions`.
- Fixed an issue where running a cluster on machines with differing MTUs can result in a cluster failure.
- Fixed an issue in which a WKA split brain results in high rate of heartbeat packets.
- Fixed an issue in which mutating operations can cause the partitioned cache to deadlock.
- Fixed an issue where a misconfigured *Extend client can throw a cryptic exception on proxy nodes.
- Fixed an issue where the `CacheService.isRunning` method returns incorrect results for *Extend clients.
- Fixed the federation thread names to follow the naming convention of the Coherence threads.
- Fixed an issue where the federation local participant name cannot be changed on a `ClusterService` restart.
- Fixed an issue in the federated scheme where the erase method of `CacheStore` is called incorrectly on entry expiration.
- Fixed an issue where partial object queries were not working in Coherence REST when used in multiple applications.
- Fixed an issue where the `CacheMBean.Unit` attribute returns an incorrect value for caches using Elastic Data.
- Fixed an issue where creating persistence snapshots through Coherence command line tools is not possible.
- Fixed an issue where the `list services` command fails to list the services in a GAR deployment.
- Fixed an issue where the `SerializationMap.setBinaryStore` method unnecessarily loads the values from the `BinaryStore` instance.
- Fixed an issue where the pre-12.2.1.x version `*Extend` client fails to receive notifications of updates to `NearCache` entries.
- Fixed an issue where the `IllegalStateException` is incorrectly thrown during the execution of `InvokeAll` method.
- Fixed an issue where an archived persistence snapshot for a two node cluster fails to load.
- Fixed an issue with the JVisualVM Plug-in where the right-click options on the Member and Services tab displays an error when connected to a managed Coherence server environment.
- Fixed an issue in the JVisualVM Plug-in where a cache byte is negative when the `unit-factor` property is used within the cache configuration file.
- Fixed the description of `connect-timeout` default value.

- Fixed an issue where a Coherence services startup race condition could result in services deadlock.
- Fixed an issue where the log message description of the `PartitionedService` was missing some fields which are present in other service types.
- Fixed an issue where the `BinaryStore` of the `SimpleSerializationMap` is not disposed when releasing the cache.
- Fixed an issue where a federation connection may not failover properly to another Coherence server.
- Fixed an issue where a `NullPointerException` can be thrown under certain circumstances when processing JMX requests.
- Fixed an issue where a partition transferred to another member results in service restart due to delayed processing of a mutation.
- Fixed an issue where client requests can hang with a service worker thread stuck on `ensureIndexReady` operation.
- Fixed an issue where a `PartitionedCache` service can be incorrectly terminated with an AssertionError in a prepareBackupAllRequest operation.
- Fixed an issue where a partition distribution can hang indefinitely.
- Fixed an issue where a `NullPointerException` can be thrown during partition redistribution.
- Fixed an issue with the MBean unregistration time and improved it to no longer be proportional to the number of registered MBeans.
- Fixed the description of the `connect-timeout` default value
- Fixed an issue where a Coherence services startup race condition can result in services deadlock.
- Removed access to disposed socket when an `EventDispatcher` logs an unhandled exception in `OnMemberLeft`.
- Fixed the `TopNAggregator` results to appear in descending order. If an `inverseComparator` is used for `LowestN` then the `TopNAggregator` results appear in ascending order.
- Updated the Coherence REST dependency versions. See the Coherence REST pom for details.
- Fixed an issue where reestablishment of failed TMB connections can lead to connection stalls.
- Fixed an issue where TMB based connections between members can stall.
- Fixed an issue during unexpected TCP disconnects where the TMB is improved to allow automatic reconnects of the underlying TCP socket.
- Fixed an issue where a Federation may throw an `OutOfMemoryException` in `JournalRecordGCDaemon` after recovering through active Persistence.
- Fixed an issue where Cache MBeans were not un-registered from the MBean server when the Extended MBeans feature was enabled.

Enhancements and Fixes for 12.2.1.2.0

- Added the `setLocalOnly` and `isLocalOnly` methods to the `ChangeRecord` interface to specify that a `ChangeRecord` object must not be federated.

- Added a `ParticipantType` attribute to the `DestinationMBean` and `TopologyMBean` MBeans.
- Added a `listNonFederatedCaches` operation to the `FederationManagerMBean` MBean that gets a list of the caches that belong to a Federated Cache service but are not being federated.
- Added a `Member` attribute to the `DestinationMBean` and `OriginMBean` MBeans.
- Added two new start operations to the `FederationManagerMBean` MBean: `startWithNoBacklog`, which clears the backlog before the start operation; and `startWithSync`, which performs a `ReplicateAll` operation to the destination after the start operation.
- Fixed a HotCache limitation where cache writes were always synthetic and prevented the federation of HotCache-refreshed caches. The use of synthetic cache writes can now be configured on a JPA entity class basis.
- Fixed an issue with HotCache where `UPDATE` operations performed on a primary key in the database were not reflected in a Coherence cache. When a database `UPDATE` operation involved other updates in addition to the primary key, then HotCache would only merge the non-primary key updates to the existing cache-entry and skip the updates to primary keys. This caused the cache to have incorrect data as compared to the database. The keys in the cache were old and obsolete as compared to the database primary keys.
- Fixed an issue with HotCache where a change to the primary key in a database row did not cause a cache refresh.
- Fixed an issue where a `LicenseException` exception was being thrown incorrectly when registering an *Extend `MapListener` implementation.
- Fixed an issue where a REST XML query for cache keys resulted in a collection of keys without a delimeter.
- Fixed an issue where a Coherence Partitioned Cache service process is terminated due to an unhandled exception when using more than one `ContinuousQueryCache` instance on the same cache.
- Fixed an issue where a license exception was being thrown in thread `main` due to licensing checks.
- Fixed an issue where releasing or destroying a cache with listeners that use an `InKeySetFilter` filter can result in an error.
- Fixed an issue where query related requests can yield incorrect result during a rolling restart.
- Fixed an issue with transactional concurrency.
- Fixed an issue resulting in the failure to remove a persistent snapshot.
- Fixed the `isDestroyed` and `isReleased` methods for a remote near cache and local cache.
- Fixed an issue where HTTP sessions are not saved when Coherence*Web is being used in PeopleSoft applications.
- Fixed an issue where the LRU eviction policy does not work in a standalone local cache.

- Fixed an issue with the `BinaryEntry` API where the `synthetic` flag is not always honored.
- Fixed an issue when using a large number of lambda invocations.
- Fixed an issue in the Coherence-JVisualVM Plug-in Server tab where the average request time field was showing the incorrect number of decimals.
- Fixed a HotCache issue where insert operations that represented updates to rows in materialized views were being ignored. HotCache can now be configured to honor redundant inserts on a JPA entity type basis.
- Fixed an issue where a deadlock may occur in an *Extend client when the connection to the proxy server is closed.
- Fixed an issue where entries that are enlisted by a `MapTrigger` implementation are not federated.
- Fixed the `BufferManager` API to avoid excessive native memory allocation.
- Fixed an issue that resulted in the false death detection of Coherence cluster members due to short network outages when using Windows TCP.
- Fixed an issue in CohQL where detailed error messages were being suppressed when persistence commands failed.
- Fixed a failure in the `DaemonPool` implementation that was triggered when the number of executed tasks exceeded 2,147,483,647.
- Fixed an issue where a partition distribution can get stuck due to an ownership conflict error.
- Fixed an issue in the Coherence-JVisualVM Plug-in where cache bytes can become negative when a unit-factor is configured within a cache configuration file.
- Fixed an issue where a Partitioned Cache service `InvocationContext` instance is set to a committed state prematurely when a deadlock occurs during the execution of the `invokeAll` method.
- Fixed an issue that can result in duplicate keys in an index map.

Enhancements and Fixes for 12.2.1.1.0

- Added `size`, `isEmpty`, `clear`, and `containsKey` methods to `AsyncNamedCache`.
- Added the ability for interceptors to see events from `PreLoadRequests` operations.
- Added the `ReplicateAllPercentComplete`, `EstimatedReplicateAllRemainingTime`, and `ReplicateAllTotalTime` attributes to `DestinationMBean` to monitor the `replicateAll` operation during federated caching.
- Added `ChangeRecord.getEventType` which can be used to obtain the `FederatedChangeEvent` type.
- Added the ability to specify a connection retry timeout for federation participants using the `<connect-retry-timeout>` element within the `<participant>` element.
- Added the ability to configure federation participant addresses without specifying a port.
- Added the ability to perform a `FORCE RECOVERY` operation using CohQL and JVisualVM when using the Dynamic Active Persistence Quorum Policy.
- Added federated cache replication support for entries loaded by read-through caching when the backing map is backed by a `CacheStore` implementation.

- Added `FederatedPartitionEvent` `SYNCING` and `SYNCED` events which can be used to track federated caching `ReplicateAll` operations.
- Added the ability to allow interceptors to receive events from entries that are loaded by read-through from `get` and `getAll` operations.
- Updated the minimum required JDK version for JVisualVM Plugin to JDK 7 Update 79 or JDK 8 Update 40.
- Updated the Coherence JVisualVM Plugin to use `coherence.` prefix instead of `com.oracle.coherence.` prefix for system properties.
- Updated the client-side `NamedCache` to throw an `IllegalStateException` when referencing a cache destroyed by another client.
- Increased the machine-name string length limit from 32 to 66 bytes.
- Fixed an issue where releasing or destroying a cache with listeners that use `InKeySetFilter` can yield an `AssertionError` error.
- Fixed a null pointer exception issue when calling `BackingMapContext.getReadOnlyEntry` for a newly enlisted entry.
- Fixed a near cache issue on a storage disabled node where the front map is not cleared upon backing map truncation.
- Fixed an issue with `NamedCache.truncate` to ensure it is executed as a silent operation on all ownership enabled nodes.
- Fixed a null pointer exception when an SSL definition is added inline in an operational configuration file.
- Fixed an issue where `AsyncNamedCache.invoke` and `AsyncNamedCache.invokeAll` do not propagate exceptions correctly.
- Fixed the `toString` method in `AbstractHttpSessionModel` to have proper checks so that exceptions are not thrown on non-standard use.
- Fixed a memory leak of `SafeNamedCache` for a client or proxy when repeatedly calling create and destroy on unique named caches.
- Fixed an issue where `UnsupportedOperationException` was being thrown while calling `entry.setValue` from an entry processor.
- Fixed a memory leak in `ContinuousQueryCache`.
- Fixed an issue with `ContinuousQueryCache.release` which may not unregister from the service and thus can not be reclaimed by the garbage collector.
- Fixed an issue with binary keys being decorated twice in rare scenarios.
- Fixed an issue where the Coherence JVisualVM Plugin is not using the Java logger correctly.
- Fixed an issue where `ensureCache` unnecessarily performs access.
- Fixed an issue where a connection hangs during SSL handshake by including the handshake within any configured connect timeout.
- Fixed an issue where an `IndexOutOfBoundsException` exception was being thrown on `ChainedRequest` modification.
- Fixed an issue where configured interceptors are not registered if the associated service scheme is referenced by a near cache.

- Fixed an issue where the `page-duration` element was not being recognized in a `paged-external-scheme` element definition.
- Fixed an issue that caused a hang during configuration processing due to a recursive macro value in a system property (that is, `-Dtangosol.pof.enabled=$ Unknown macro: {tangosol.pof.enabled}`) where OS system property was not set.
- Fixed an issue with resolving Coherence configuration element names against system properties (that is, system property `autostart` will not override Coherence configuration element `autostart`).
- Fixed an issue where cache operations which do not change an Entry's value, such as cache store write-behind, can result in an `UnsolicitedCommitEvent` event.
- Fixed `AsyncNamedCache` operations to return empty collection instead of null.
- Fixed an issue where asynchronous ownership is requested by WLS SPI code for coherence web.
- Fixed an issue where the partitioned cache configuration and statistics MBeans for a cache are unregistered when the cache is closed. These are now unregistered when the cache is destroyed.
- Fixed an issue with the Coherence JVisualVM Plugin where the Machine tab and Load Average graph were blank when connected to a cluster using IBM JDK on AIX.
- Fixed an issue where the `ReportTime` column is missing in `report-cache-size.xml` report.
- Fixed an issue that caused unnecessary unregister MBean messages sent to a managed node after a cache is destroyed.
- Fixed an issue where using CohQL to query JSON data stored in the cache may return a local class incompatible error.
- Fixed an issue with transactional concurrency.
- Fixed an issue when submitting an `AsynchronousEntryProcessor` on a storage-enabled node while primary ownership is being changed.

Enhancements and Fixes for 12.2.1.0.0

- The read-write backing map now accelerates store and write operations for entries that are being evicted but that have not yet been written to the cache store.
- Added `SamplingEventTransformer` class that implements server-side event throttling.
- Added asynchronous `NamedCache` API support (`AsyncNameCache` interface).
- Added support for non-blocking invocable. The `NonBlockingInvocable` interface is designed to allow the invocation service thread to execute the corresponding task and get an invocation result without blocking.
- Added support for callbacks to asynchronous entry processors.
- Added support for streaming aggregation to reduce memory overhead and improve performance.
- Added support for the use of Java Generics.
- Added the `<key-type>` and `<value-type>` elements within the `<cache-mapping>` element to declare the types that are supported by a `NamedCache` cache.

- Added support for lambdas to the `NamedCache` API
- Added support for dynamic lambda execution.
- Added support for default `Map` methods introduced in Java 8 to the `InvocableMap` interface.
- Added the ability to rebuild index data structures asynchronously after a partition transfer or failover to significantly reduce the time to recovery.
- Added support for executing asynchronous aggregators.
- Added support in Elastic Data to use pure RAM Journal without overflowing to Flash.
- Improved near cache performance when using the `present` invalidation strategy.
- Added the ability to enforce filter evaluation order for array based filters (for example, `AndFilter`, `AnyFilter`).
- Enhanced CohQL so that it is possible to provide a custom value extractor (for example `POFExtractors`) to be used in CohQL queries.
- Added the `truncate` method to the `NamedCache` interface that is functionally equivalent to `clear` but doesn't raise any entry-level cache events. This new API significantly reduces the memory pressure on the server side and dramatically reduces the network consumption for caches with listeners and is especially beneficial for near caching.
- Added support for sliding expiry. The expiry of cache entries are extended upon read access.
- Cache entries are proactively evicted from a backing map after the cache expiry time is reached and no longer require a cache operation to initiate the eviction.
- Fixed inconsistent duplicate event interceptor registration behavior when using name and identifier.
- Fixed a problem causing a slow memory leak for map listeners that receive local-only (in-process) events.
- Fixed an issue where repeated calls to `NamedCache` operations (`create`, `write` and `destroy`) cause `Storage` instance leak.
- Fixed the inability to determine the number of tasks waiting to be executed (`ServiceMBean.EventBacklog`) on the `EventDispatcher` thread, or the total number of tasks executed (`ServiceMBean.EventCount`).
- Fixed an issue where the `NamedCache.lock(key, -1)` call does not wait until the lock is available.
- Fixed an issue where the `addIndex` method that is run in a non-synchronized thread gave the wrong results for a replicated cache.
- Fixed an issue when using the `MultiExtractor` class as an index that may result in inaccurate indexed data.
- Fixed an issue with invoking the `GuardSupport.heartbeat` method during execution of post commit live events.
- Fixed an issue with the use of expiry or eviction together with high load causes an `AssertionException` exception or deadlock.

- Fixed an issue where near cache event interceptors that are defined in a cache configuration file are not registered when using the `ExtensibleConfigurableCacheFactory` class.
- Fixed an issue that caused asynchronous entry processors to be deferred after a service is suspended and resumed.
- Fixed an issue with `ConfigurableCacheMap` methods returning incorrect values when using an `ObservableSplittingBackingCache` instance.
- Fixed an issue with the `UpdaterProcessor` class that can cause `CacheStore`, `BinaryEntryStore`, and `Listener` instances to miss entry updates.
- Fixed an issue where a backing map listener does not process events immediately and can result in a stale backup value to be returned.
- Fixed an issue where precommit transaction events (`COMMITTING`) show entries that have not changed.
- Fixed an issue where the front map of a near cache does not get cleared when the back cache is destroyed.
- Fixed an issue where the front cache of a near cache is not updated after all storage nodes are restarted.
- Added an ability to call `MBeanServer` remotely.
- Added a new dynamic management mode that elects one of the nodes in the cluster to be the management node.
- Added an attribute called `IndexTotalUnits` in the `StorageManagerMBean`.
- Improved index size accuracy.
- Add the ability to pass report file and report group XML as a string when using `ReporterMBean` operations.
- Fixed an issue where the `ClusterMBean.logClusterState` method call is not generating thread dumps if the `extendedmbeanname` system property is set to `true`.
- Fixed a bug where a value set for the `<default-domain-name>` element in an operational override configuration file was being ignored.
- The default unicast listener address is derived from the Well Known Address (WKA) list when available, selecting a local IP on the same subnet as the WKA addresses.
- The WKA lists can be represented using a single hostname where DNS has been configured to return a list of IPs.
- Coherence uses port 7574 as the default cluster port for multicast communication as well as for the Name Service. Unicast ports are automatically selected.
- Multiple clusters can now share a cluster port and Multicast or WKA address. For most use cases, there is no reason to change the cluster port, or multicast address.
- Added an ability to authorize access to cache data
- Added the ability to capture audit logs to record user access to clustered data.
- Added additional JAAS permissions to protect various parts of the Coherence API using the Java Security Manager.
- Added the ability to configure an SSL socket provider to control the use of potentially weak ciphers or specific protocol versions.

- The Proxy service now binds to the same address as the Name service.
- Coherence*Extend clients can be redirected by a proxy server that is at its connection limit.
- Added Guardian support to the `NameService` interface to auto-restart the acceptor.
- Added support for server-sent events to Coherence REST.
- The `<key-class>` and `<value-class>` elements can either be defined within the `<resource>` element in the REST configuration file or within the `<cache-mapping>` element in the cache configuration file.
- Added support for Coherence*Extend clients to use JCache.
- Added support for Jetty HTTP server to Coherence REST.
- Oracle Traffic Director (OTD) can be used to load balance Coherence*Extend connections.
- Added `SimplePrincipal` which derives from `GenericPrincipal`. `SimplePrincipal` uses the `Identity.Name` property to determine object equality.
- Fixed an issue which causes the proxy server `TcpProcessor` thread to get stuck in a CPU busy loop.
- Removed the `ConnectionManager` MBean from the name service.
- Fixed a concurrent access issue with UUIDs.
- The `<connect-timeout>` element has been removed from the `<tcp-initiator>` element. It is defined in the `<initiator-config>` element instead.
- Fixed an issue where an Extend client doesn't receive delete events from a replicated cache.
- Reduced contention on the proxy service for key-based requests that target the same keys.
- Added a new HTTP session reaping mechanism that uses entry processors to increase performance when deleting sessions.
- Added the ability to have HTTP session caches replicated across federated clusters.
- Fixed an issue where HTTP session attributes are mandated to implement `java.io.Serializable`.
- Added support for Java 8 `Date` and `Time` API to POF.
- Added support for optional types to the `PofReader` and `PofWriter` interfaces.
- Fixed an issue with the POF configuration generator not allowing directories with spaces as a root directory of class files.
- Fixed a POF serialization issue for negative dates with nonzero subseconds.
- Coherence JCache POF ids were moved from 700-799 in previous releases to 610-699 in order to avoid collisions with transaction POF ids.
- Coherence system property names no longer require the `tangosol` prefix.
- The `<port-auto-adjust>` element has been changed to support either a boolean value or an upper bound on the port range.
- Added support for enabling or disabling specified SSL protocols.

- Added the ability to use system properties for element values in configuration files. The syntax is:

  `${system.property default_value}`

- Fixed an issue with the `partitioned` setting of the backing map. When explicitly set to `false`, it was also setting the in-memory backup partitioned setting to `false`.
- Fixed an issue where resources containing `#` character in filename or path failed to load.
- Added a new dynamic management mode that elects one of the nodes in the cluster to be the management node. This is the default management mode for managed Coherence servers.
- A GAR file can reference shared libraries.
- A GAR file can be deployed across multiple domain partitions and used by all tenants. Isolation is provided at the domain partition level and is transparent to the application.
- Added the ability override cache properties for each domain partition.
- Added the ability to allow all domain partitions in a cluster to share caches.
- Added the ability to use JCache with managed Coherence servers.
- The coherence REST library (`coherence-rest.jar`) is located on the WebLogic Server classpath by default and does not need to be packaged and deployed as part of an application.
- Fixed an issue where internal session attribute names are returned without stipping the `InternalWLSAttribute` prefix.
- The `MBeanServerFinder` interface has a new method to return the `JMXServiceURL` value for the MBean server.
- Fixed an issue where daemon threads were not using the specified service classloader.
- Added support for log4j 2.
- Fixed an issue with Windows command scripts that fail to handle `JAVA_HOME` path containing spaces and parenthesis.

4.2 Oracle Coherence for .NET

New features, improvements, and bug fixes added to Oracle Coherence for .NET components.

Enhancements and Fixes for 12.2.1.3.0

- Removed access to the disposed socket where the `EventDispatcher` logs an unhandled exception in the `OnMemberLeft` handler.
- Fixed an issue where Channel and Codecs had incomplete descriptions in log messages.
- Fixed the description of the `connect-timeout` default value
- Fixed an issue where using `TopNAggregator` on .NET local caches does not work.

- Fixed the `TopNAggregator` results to appear in descending order. If an `inverseComparator` is used for `LowestN` then the `TopNAggregator` results appear in ascending order.
- Fixed an issue with Member location information display and improved the Member `ToString()` output to closely match the display of Java.
- Fixed an issue where implementations of various `AbstractProcessor`, `AbstractInvocable`, and `AbstractAggregator` methods to facilitate the .NET development of these type of classes.

Enhancements and Fixes for 12.2.1.2.0

- Fixed an issue with the .NET serialization of surrogate pairs where `java.io.UTFDataFormatException` was being thrown.
- Fixed an issue where differing locales between a POF client and the cluster caused problems with decimals in .NET.
- Fixed an issue where a deadlock may occur in an *Extend client when the connection to the proxy server is closed.
- Fixed an issue that resulted in the false death detection of Coherence cluster members due to short network outages when using Windows TCP.

Enhancements and Fixes for 12.2.1.1.0

- Fixed a memory leak in `ContinuousQueryCache`.
- Fixed an issue with `ContinuousQueryCache.release` which may not unregister from the service and thus can not be reclaimed by the garbage collector.
- Fixed a concurrent access issue on `CacheFactory` methods.
- Fixed an issue where attributes that exceed the `SplitSessionModel` minimum size may require a retry or delay before access by the client.
- Fixed `.csproj` files for the .NET examples bundled in the Java install to target .NET framework 4.0.

Enhancements and Fixes for 12.2.1.0.0

- .NET clients can now configure POF configuration files in the cache configuration file rather than the application configuration file. Note that a custom serializer must now implement a constructor which initializes the `<init-params>` of the `<serializer>` element in the cache configuration file.
- The default connect time out value is consistent across platforms.
- Added `SimplePrincipal` which derives from `GenericPrincipal`. `SimplePrincipal` uses the `Identity.Name` property to determine object equality.
- The `<connect-timeout>` element has been removed from the `<tcp-initiator>` element. It is defined in the `<initiator-config>` element instead.
- Fixed an issue with client read and write lock acquisition.
- Fixed an issue where the `SynchronizedDictionary.AcquireWriteLock` method may hide thread interrupts.
- Fixed a concurrent access issue with UUIDs.
- Added support for optional types to the `PofReader` and `PofWriter` interfaces.

- Fixed an issue in the `ValueChangeEventFilter` class to correctly extract the value used by the filter.
- Fixed an issue where `SimpleMapIndex` instances that use key-based extractors are very slow.
- Fixed an issue where the front cache of a near cache is not updated after all storage nodes are restarted.
- Cache entries are proactively evicted from a backing map after the cache expiry time is reached and no longer require a cache operation to initiate the eviction.
- Coherence system property names no longer require the `tangosol` prefix.
- Added the ability to use system properties for element values in configuration files. The syntax is:

  ```
  ${system.property default_value}
  ```

4.3 Oracle Coherence for C++

New features, improvements, and bug fixes added to Oracle Coherence for C++ components.

Enhancements and Fixes for 12.2.1.3.0

- Added `COH_OPEN_NAMESPACE_ANON` and `COH_CLOSE_NAMESPACE_ANON` macros to facilitate declaring Coherence managed classes in pseudo anonymous namespaces when using Visual Studio 2017.
- Added C++ classes which were missing from the API documentation.
- Added reverse and swap methods to the Collections and Arrays utility classes.
- Fixed an issue where implementations of various `AbstractProcessor`, `AbstractInvocable`, and `AbstractAggregator` methods to facilitate the C++ development of these type of classes.
- Fixed the description of the `connect-timeout` default value.
- Fixed an issue where an `*Extend` connection can be closed with a WSA_IO_PENDING error.
- Fixed the formatting of log messages to be similar to that of the Coherence Java log messages.
- Fixed the `TopNAggregator` results to appear in descending order. If an `inverseComparator` is used for `LowestN` then the `TopNAggregator` results appear in ascending order.
- Fixed an issue with Member location information display and improved the Member `toString()` output to closely match the display of Java.
-

Enhancements and Fixes for 12.2.1.2.0

- Added the `ConnectionException.hpp` header file to make connection exceptions public.
- Fixed an issue with client-side `NamedCache` reference handling where intermittent network failures caused a Cache service to restart and inactive `NamedCache` references to be released.

- Fixed an issue in the SafeNamedCache class that caused a memory leak when repeatedly creating and destroying a unique NamedCache instance.
- Fixed an issue where a deadlock may occur in an *Extend client when the connection to the proxy server is closed.
- Fixed an issue where a memory leak occurs if an attempt to connect to a proxy server fails or a proxy server connection is lost.
- Fixed an issue that resulted in the false death detection of Coherence cluster members due to short network outages when using Windows TCP.
- Fixed an issue where the CacheFactory::shutdown member function may hang if Coherence services are subject-scoped.
- Fixed an issue where constructors and deconstructors of statics may be called more than once on Solaris.
- Fixed an issue where the ClassBasedHeapAnalyzer class may deadlock on initialization.

Enhancements and Fixes for 12.2.1.1.0

- Fixed a memory leak in ContinuousQueryCache.
- Fixed an issue with ContinuousQueryCache.release which may not unregister from the service thus can not be reclaimed by the garbage collector.
- Fixed an issue whereby LimitFilter with PofExtractor was throwing an UnsupportedOperationException exception.
- Fixed an issue where C++ Extend clients incorrectly validated UTF8 strings.

Enhancements and Fixes for 12.2.1.0.0

- Added support for STLport on all Solaris distributions.
- Added support for C++11.
- Added support for Windows 64 bit BackTrace.
- Added a Thread::isAlive method.
- Added new methods to the coherence::util::Map interface, as per the Java 1.8 JDK additions. These methods implement the defaults but can be overridden. See the API documentation for coherence::util::Map for more information.
- Added property tangosol.coherence.heap.logging to enable logging details about the thread local memory allocator hit and miss rate.
- Changed FinalView InKeySetFilter.f_vSetKeys to MemberView m_vSetKeys as InKeySetFilter::ensureConverted requires that m_vSetKeys be modifiable.
- The <connect-timeout> element has been removed from the <tcp-initiator> element. It is defined in the <initiator-config> element instead.
- The Object::toStream method has been deprecated in favor of Object::toString.
- The default value for connect timeout is now consistent across platforms.
- Fixed an issue with getting debug stack traces.
- Fixed an issue where TreeMap::put returns the new value instead of the old value.
- Fixed an issue with strict-aliasing compilation warnings that occurred from the use of the -O3 optimization option.

- Fixed an issue with the memory allocator always running with heap padding diagnostics enabled.
- Fixed an issue where dependent threads of a Coherence service may fail to stop during service shutdown.
- Fixed a serialization buffer allocation performance issue.
- Fixed a concurrent access issue with UUIDs.
- Fixed an issue with `String::substring` prematurely ending a substring search.
- Fixed an issue in the `ValueChangeEventFilter` class to correctly extract the value used by the filter.
- Added support for outputting managed objects to `std::wostream`.
- Fixed an issue where `SimpleMapIndex` instances that use key-based extractors are very slow.
- Fixed an issue where the front cache of a near cache is not updated after all storage nodes are restarted.
- Cache entries are proactively evicted from a backing map after the cache expiry time is reached and no longer require a cache operation to initiate the eviction.
- Coherence system property names no longer require the `tangosol` prefix.
- Added the ability to use system properties for element values in configuration files. The syntax is:

  ```
  ${system.property default_value}
  ```

5 Documentation Changes

The documentation errata lists any corrections to the Oracle Coherence documentation.
The Coherence documentation can be found at the following URL:

`http://docs.oracle.com/en/middleware/`

There are currently no corrections to the Oracle Coherence documentation.

Index