Oracle® Fusion Middleware

Release Notes for Oracle Coherence

12*c* (12.1.2)

E28660-08

February 2015

This document contains the release notes for the Oracle Coherence 12*c* (12.1.2) release.

ORACLE®

Oracle Fusion Middleware Release Notes for Oracle Coherence, 12*c* (12.1.2)

E28660-08

Contents

2 Documentation Errata

3 Upgrading to Coherence 12*c* (12.1.2)

Preface

This document describes changes and enhancements that have been made to Oracle Coherence for the 12.1.2 release.

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:

- *Oracle Fusion Middleware Developing Applications with Oracle Coherence*
- *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*
- *Oracle Fusion Middleware Tutorial for Oracle Coherence*
- *Oracle Fusion Middleware Integrating Oracle Coherence*
- *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web*
- *Oracle Fusion Middleware Managing Oracle Coherence*
- *Oracle Fusion Middleware Administering Oracle Coherence*
- *Oracle Fusion Middleware Securing Oracle Coherence*
- *Oracle Fusion Middleware Java API Reference for Oracle Coherence*
- *Oracle Fusion Middleware .NET API Reference for Oracle Coherence*

- *Oracle Fusion Middleware 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

Technical Changes and Enhancements

This chapter describes the changes and enhancements made to the Oracle Coherence product for 12*c* (12.1.2). This document is accurate at the time of publication. Oracle updates the release notes periodically after the software release.

1.1 Download and Install the Latest Software Patch

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

`https://support.oracle.com`

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

1.2 New and Improved Coherence Data Grid Functionality

Oracle Coherence 12*c* (12.1.2) is the latest release of the industry's leading distributed in-memory data grid product. Coherence 12*c* (12.1.2) contains many new features that help reduce complexity, ease integration, and accelerate time to market of scalable solutions.

- Managed Coherence Servers—Managed Coherence Servers leverages the WebLogic Management Framework and provides streamlined application development and management. It introduces the Grid Archive (GAR) format which is a specific directory structure designed to optimize the packaging, deployment, and use of Coherence applications. The archive format also provides application isolation and lifecycle events. Standalone Coherence customers can also use this feature without owning WebLogic Server. For more information, see *Developing Oracle Coherence Applications for Oracle WebLogic Server*.
- Live Events—Coherence provides an event framework that allows your applications to react to operations performed in the data grid. The framework uses an event-based model where events represent observable occurrences of cluster operations. The supported events include partitioned service, cache, and application events. These events can be consumed by registering event interceptors either programmatically or by using the cache configuration. The new API for live events simplifies the creation of event-driven applications in Coherence. Coherence allows you to scale event processing along with data. This is becoming more important in the world of Fast Data, where mobile and social customers expect real-time updates. For more information, see "Using Live Events" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- Integration with Oracle Coherence GoldenGate HotCache (HotCache)—HotCache allows changes to the database to be propagated to objects in the Coherence cache. Third-party updates to the database can cause Coherence applications to work with data which could be stale and out-of-date. HotCache solves this problem by monitoring the database and pushing any changes into the Coherence cache. HotCache employs an efficient push model which processes only stale data. Low latency is assured because the data is pushed when the change occurs in the database. For more information, see "Integrating with Oracle Coherence GoldenGate HotCache" in *Oracle Fusion Middleware Integrating Oracle Coherence*.
- Asynchronous backup for Partitioned Service—Distributed caches support both synchronous and asynchronous backup. With synchronous backup, clients are blocked until a backup operation completes. With asynchronous backup, clients continue to respond to requests during backup operations. Asynchronous backup is typically used to increase client performance. For more information, see "Using Asynchronous Backup" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- Strengthened partition distribution—Partition distribution has been strengthened to ensure that (topology allowing) when using `backup-count > 1`, all backup copies are pushed to different machines (from one another and the primary). This ensures that with `backup-count=N`, customers can tolerate `N` simultaneous machine failures without losing data.
- Name services for Extend clients—A name service is a specialized TCP acceptor that allows extend clients to connect to a proxy by specifying a proxy service name instead of a proxy service address. Clients connect to the name service acceptor, which provides the actual address of the requested proxy. The use of the name service acceptor allows actual proxy addresses to change without having to update a cache configuration file. For more information, see "Using the Name

Service Acceptor to Connect to a Proxy" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.

- Support for proxy address provider—Proxy service, remote cache, and remote invocation definitions can now reference a TCP socket address that is defined in an operational override configuration file instead of explicitly defining an addresses in a cache configuration file. Referencing socket address definitions allows network addresses to change without having to update a cache configuration file. For more information, see "Using Address Provider References for TCP Addresses" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.
- Proxy services improvements—Proxy services use a dynamic thread pool for daemon (worker) threads. The thread pool automatically adds and removes threads based on the number of client requests, total backlog of requests, and the total number of idle threads. The thread pool helps ensure that there are enough threads to meet the demand of extend clients and that resources are not waisted on idle threads. For more information, see "Configure Proxy Service Thread Pooling" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.
- POF configuration file generator—The POF Configuration Generator command line tool automatically creates a POF configuration file that includes user type entries for the classes that contain the `@Portable` annotation. The tool is an alternative to manually creating a POF configuration file and is ideal as part of a build process. For more information, see "Generating a POF Configuration File" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- Security for Coherence applications on WebLogic Server—The security features for coherence applications include authorization between cluster members, authorization to Oracle Coherence caches and services, and authentication for extend clients. For more information, see "Securing Oracle Coherence in Oracle WebLogic Server" in *Oracle Fusion Middleware Securing Oracle Coherence*.
- Support for named queries in the REST API—Named queries are query expression that are configured for a resource in the `coherence-rest-config.xml` file. By default, the query expression must be specified as a CohQL expression (the predicate part of CohQL). For more information, see "Using Named Queries" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.
- Run Multiple REST applications—Your application server can run multiple Coherence REST applications by configuring multiple context paths in the cache configuration file. For more information, see "Deploying with the Embedded HTTP Server" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.
- Security for Coherence REST—Oracle Coherence REST security uses both authentication and authorization to restrict access to cluster resources. Authentication support includes: HTTP basic, client-side SSL certificate, and client-side SSL certificate together with HTTP basic. Authorization is implemented using Oracle Coherence*Extend-style authorization. For more information, see "Securing Oracle Coherence REST" in *Oracle Fusion Middleware Securing Oracle Coherence*.
- Namespace handlers in the cache configuration—Namespace handlers are used to process XML elements and attributes that belong to a specific XML namespace. Typically, namespace handlers extend the base `AbstractNamespaceHandler` implementation class, which provides convenience methods that can simplify the processing of complex namespaces. For more information, see "Extending Cache

Configuration Files" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

- Thread priority for cluster services—Cluster services support thread priority. The priority is used as the basis for determining Java thread execution importance and indicates which threads of a service are considered critical. There are three types of threads that can have a priority: service threads, event dispatcher threads, and worker threads. For more information, see "Specifying Thread Priorities for Services" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- Custom failover access policy—Coherence provides a pre-defined custom action policy that moderates client request load during a failover event. This allows cache servers adequate opportunity to re-establish partition backups. Use this policy in situations where a heavy load of high-latency requests may prevent, or significantly delay, cache servers from successfully acquiring exclusive access to partitions needing to be transferred or backed up. For more information, see "Enabling the Custom Failover Access Policy" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- Oracle Coherence on Oracle Exalogic and the Oracle Exalogic Elastic Cloud (OEEC)—OEEC software provide a foundation for extreme performance, reliability, and scalability. Coherence has been optimized to take advantage of this foundation especially in its use of Oracle Exabus technology. For more information, see "Oracle Exalogic Elastic Cloud Recommendations" in *Oracle Fusion Middleware Administering Oracle Coherence*.
- Oracle Universal Installer (OUI) installation—Added the ability to install Oracle Coherence for Java by using OUI. For more information, see Installing Oracle Coherence for Java in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- Expanded `NamedSerializer` support—Support for the `NamedSerializer` API has been ported to .NET and C++.
- Simple Logging Facade for Java (SLF4J)—Applications that use SLF4J logging can now configure Coherence to use SLF4J logging as well. For more information, see "Using SLF4J for Coherence Logs" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- Integration with Maven—Maven is a build and dependency system that allows the configuration of project dependencies, 3rd party dependencies and definition of a build lifecycle. Software projects often use Maven to simplify and standardize their build process. For more information, see "Integration with Maven" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*. For details about Maven, see `http://maven.apache.org/`.
- Support for ECID—Execution Context ID (ECID) allows correlation of log entries across server instances and Fusion Middleware components. For more information, see “Changing the Log Message Format” in the *Oracle Fusion Middleware Developing Applications with Oracle Coherence* and "Enabling ECID in Coherence Logs" in *Oracle Fusion Middleware Integrating Oracle Coherence*.
- Support for Java SE Version 7—Oracle Coherence 12*c* (12.1.2) is supported and certified on Java SE Development Kit (JDK) 7.
- Support for Berkeley DB Version 5—Oracle Coherence is updated with the embedded version of Berkeley DB 5.0.
- Integration support for Hibernate release 3.3. or later—Oracle Coherence 12*c* (12.1.2) provides support for integrating with Hibernate release 3.3 or later. For

more information, see the Coherence Community pages at the following URL: https://java.net/projects/cohhib

1.3 Oracle Coherence for Java 12*c* (12.1.2)

New features, improvements, and bug fixes have been added to these Oracle Coherence for Java components:

- Management and Monitoring Enhancements and Fixes
- Partitioned Cache Enhancements and Fixes
- TCMP Enhancements and Fixes
- Coherence*Extend Framework Enhancements and Fixes
- Coherence*Web Enhancements and Fixes
- Exabus Framework Enhancements and Fixes
- Serialization Framework Enhancements and Fixes
- REST Framework Enhancements and Fixes
- Configuration Enhancements and Fixes
- Coherence Container Enhancements and Fixes
- Other Enhancements and Fixes

1.3.1 Management and Monitoring Enhancements and Fixes

- The `cache-size-report` has been improved to provide an entry for every unique cache name and service name composite key, whereas formerly only the cache name was used in the key.
- ServiceMBean now contains a property `JoinTime` which shows the time (cluster time) the member has joined a given service.
- Added the `management-config` configuration element `extended-mbean-name` to reflect the member name in global MBean object names.
- Added support for `BigDecimal` aggregators in COHQL `select` statement syntax: `bd_avg`, `bd_max`, `bd_min`, `bd_sum`.
- Coherence console handles `IOException` by graceful shutdown.
- Introduced a new `SimpleStrategyMBean` providing monitoring information related to `SimpleAssignmentStrategy`.
- Added partition-lost notifications to `SimpleStrategyMBean`.
- Execution Context ID (ECID) logging has been added to Coherence. If a client application has an active DMS Execution Context at the time it invokes a Coherence operation, Coherence includes the ECID (Execution Context ID) into any log messages that it creates during its execution on the client side. In addition, any remote exceptions thrown by Coherence while executing the client request on other nodes is propagated back to the client and logged by Coherence on the client side bearing the ECID of the request. The remote exception is then re-thrown to the client. Server side logging does not include ECIDs into its messages.
- Fixed an issue with Near Cache re-registration.
- Fixed an issue that caused `MBeanHelper.getServerFinder()` method to unnecessarily create new `MBeanServerFinder` instances for each call.

- Added `logClusterState` operation to the `cluster` MBean. The operation allows you to log state of all nodes in the cluster at the same time, capturing full thread dump and outstanding polls.
- Added MBean operations to produce simultaneous thread dumps in various cluster member subsets.
- Fixed JMX so that it reports the correct size for the `read-write-backing-map-scheme` cache.
- Added the ability to programmatically invoke a CohQL statement by using the `QueryHelper.executeStatement(String)` method.
- Fixed the issue that caused the domain name of a custom MBean to be ignored and changed to "Coherence". The JMX domain name for custom MBeans registered by applications is now "retained". For more information, see Knowledge Module Doc ID 1392954.1, available from Oracle Support:

  `https://support.oracle.com`
- Added a unique cluster member identifier (UID) to `ClusterNodeMBean`.
- Added support for MXBeans.

1.3.2 Partitioned Cache Enhancements and Fixes

- Fixed a race condition where setting the `ClassLoader` on a service after it has been started would result in the service's serializer being temporarily set to null. Note that it is recommended that the `ClassLoader` only be set prior to starting a service.
- Fixed the `java.lang.ArrayIndexOutOfBoundsException` that occurred when using `LimitFilter`.
- Fixed an issue which could cause missing `MapListener` events for certain kinds of expiry or eviction changes.
- Fixed a problem with Service MBean operation "`start`". It is now possible to start a service using JMX. In addition, when a service is stopped, attributes of the Service MBean reflect the actual state of the service, whereas before they remained uninitialized.
- Fixed an issue that could lead to a `ContinuousQueryCache` over Extend to stop receiving events if the proxy is restarted.
- Fixed an issue where a resource lock timeout during the `getAll`, `putAll`, and `removeAll` operations could result in service failure.
- Fixed the issue that caused a `java.lang.IllegalStateException: invalid minute offset` error to be thrown for non-standard timezone.
- Fixed an issue which prevented `CacheMap.EXPIRY_NEVER` from working properly.
- Fixed a memory leak in transactional caches with read operations and auto commit enabled.
- Fixed an issue that could cause two members to join the cluster from within the same `DefaultCacheServer` process when the cluster is explicitly shutdown.
- Fixed an issue where `AllFIlter` was throwing an `ArrayIndexOutOfBoundsException` when initialized with an empty filter array.
- Reduced additional memory overhead for running `InvokeAll` operation on `PartitionedCache`.

- Fixed an issue which could cause a Coherence server out of memory (OOM) error for off-heap backup.
- Fixed a memory leak caused by Daemon's internal use of `ThreadGroups`.
- `SimpleAssignmentStrategy` now automatically adapts to a large cluster startup.
- Fixed an issue with excessive logging during server startup.
- Introduced an asynchronous backup option for Partitioned Service. It must be explicitly enabled.
- Fixed a deadlock that could occur during query evaluation in the presence of heavy concurrent updates.
- Fixed an issue with `ReplicatedCache` that could cause new service members to hold outdated lease information
- Fixed an issue that could cause a client to be stuck waiting for re-distribution.
- Fixed an issue that caused the aggregation of large data sets to spuriously miss partitions.
- Fixed an issue that could result in an `IllegalStateException` during the `removeIndex` operation.
- Improved resiliency of an index when a cache entry causes an exception during index update. The index is no longer dropped, so that queries relying on the index continue to execute efficiently. Corrupted entries are excluded from query results.
- Fixed a memory leak of pending invocation results, triggered by high `EntryProcessor` load during a failover.
- Fixed an issue that could cause ownership conflict with `backup-count > 1`.
- Fixed a regression in `ReadWriteBackingMap.getAll()` logic that could result in insertion of null values into the cache.
- Fixed an issue that caused an assertion to be thrown for performing single key operations while backups are being transferred.
- Preserved the entry expiry time across partition transfer or failover when the system property `tangosol.coherence.distributed.strictexpiry` is specified as `true`.
- Fixed a regression causing incomplete results for queries that use indexes against collection-based extractors.
- Fixed an issue that could result in data-loss with a multi-node failure.
- Fixed an issue that could prevent the service or cluster from restarting after abnormal failure with a `RequestTimeoutException`.
- Fixed an issue that caused `put` requests to be blocked by eviction when storage is slow.
- Fixed an issue related to using `ConditionalIndex` with `PofExtractor`.
- Fixed an issue where creating an index while performing queries could stall the cache service.
- Fixed an issue that could raise backing map events out of order when `ObservableHashMap` was used as a backing map.
- Fixed an issue with `EntryProcessor` throwing an exception, which in turn caused the backup node to throw a `NullPointerException`.

- Fixed issue that caused an `ConcurrentModficiationException` to be thrown during a partition transfer when Exabus is enabled.
- Fixed an issue that caused nodes to stop due to "`Incompatible BinaryDelta implementation`" when a storage-disabled proxy joined a cluster with a different backup count.
- Fixed issue with `ConcurrentModificationException` or `NegativeArraySizeException` being thrown when running index backed queries.
- Fixed an issue where unindexed filtered invocations employed excessive locking.
- Fixed an issue with Refresh Ahead when the object being updated is locked.
- Fixed an issue with non-monotonic registrations of request SUIDs.
- Introduced `thread-count` < 0, which allows eligible cache requests to be processed outside of the service to improve latency when running on `MessageBus`.
- Triggers and Live Events interceptors can now enlist new entries when running for cache `put` and `remove` operations.
- Fixed an issue where Live Events events for cache entries would indicate the wrong state.
- Changed return type of internal API `load()` and `loadAll()`.
- Fixed an issue that could cause the erroneous error: `RequestPolicyException ("The current quorum policy prevents PartitionSet{...} from being recovered"` to be thrown.
- Fixed an issue when using `backup-count-after-writebehind` and POF.

1.3.3 TCMP Enhancements and Fixes

- Fixed an issue with `PacketPublisher` auto-throttling.
- Coherence now allows the `PacketSpeaker` to be disabled to reduce memory consumption in large WKA clusters.
- Optimized the `IPMonitor` topology so that a single member on each machine performs outgoing pings.
- Reduced the cost of removing and recovering from timed-out members.
- Resolved an issue where default SSL context was loaded in non-SSL configurations.
- Added support for service-level reliable transport selection, for example:

```
<distributed-scheme>
   ...
   <reliable-transport>
      datagram
   </reliable-transport>
   ...
</distributed-scheme>
```

- Added the ability to set the service and worker thread priorities.
- With the introduction of this feature we have removed the ability for the user to specify the settings for the Buffer Pool, therefore `message-pool` and `packet-pool` have been removed from the `XSD`.
- Added Exabus transport attributes to `ServiceMBean`.

- Addressed partial network failures when running over Exabus Account for `MessageBus` failure which occurs independent of network/process failure.
- Resolved an issue preventing SSL handshake from completing on some IBM JDKs.
- Added throttling to avoid `PacketPublisher` thread overwhelming the `PacketSpeaker`.
- With the introduction of this feature we have removed the ability for the user to specify the settings for the Buffer Pool. Therefore, `message-pool` and `packet-pool` have been removed from the XSD. The small number of customer which are using these settings.
- To improve throughput, message deserialization is now off-loaded to the IO thread-pool when running on `MessageBus`.

1.3.4 Coherence*Extend Framework Enhancements and Fixes

- Improved performance by removing unnecessary mutex locking when accessing registered classes and serializers in the `SystemPofContext` class.
- An issue with the number of events received by key and filter listeners has been resolved.
- An issue with an Extend client receiving multiple copies of an event has been fixed.
- Fixed an issue where calling `NamedCache.invoke` using POF/Extend when payload contains circular reference resulted in `java.lang.StackOverflowError`.
- Fixed an issue where `ClassCastException` could be thrown when `LimitFilter` is used with custom `Comparator`.
- When the first property of a user object is not read, later reads of the object can return the type and version, rather than the actual property value.
- Fixed missing cache event call-backs when using a custom `ExtractorEventTransformer` class.
- The log level of `TcpAcceptor.BufferPool` messages has been raised to level 9.
- Upgraded the Coherence REST API to use Jersey 1.17.1 libraries.
- Fixed POF serialization inconsistency between Java and .NET when a string is empty ("").
- Fixed an issue where incorrect results can be returned to an Extend client using a `LimitFilter` when the Coherence cluster has more than one storage node.
- Fixed `ExtractorEventTransformer` serialization issue which caused new value extractor to be ignored on the server when different from the old value extractor.
- Implemented named initiators for placing connect host and port information in operational configuration.
- Implemented named acceptors for placing listening host and port information in operational configuration.
- Added the ability to secure communication between REST clients and proxies using SSL. REST clients can also be authenticated using HTTP basic authentication, client-side certificates, or both. Additionally, the Coherence*Extend Security framework was enhanced to include support for fine-grained authorization of REST requests.

- Removed a potential deadlock which could occur during Coherence*Extend client request timeout.
- Added send-timeout support for Extend clients. The `<request-timeout>` value in `<initiator-config>` now limits the time to send the request, in addition to the prior behavior of limiting the time to receive the response.
- Made the internal REST `EntryProcessors` POF-serializable, to support exposing POF classes by using REST.
- Coherence now ignores a bad host name in authorized hosts, and logs a message.
- Improved performance of `SafeConfigurablePofContext` by caching the "missed" classes.
- Fixed an issue that caused an infinite loop in `TcpAcceptor` class.
- Added `transformer` to `ContinuousQueryCache`.
- The `TCPAcceptor` and `TCPInitiator` elements have been removed from the `cache-config.xml` configuration file.
- Removed the redundant re-resolution of remote addresses in `TcpInitiator.openConnection()` method.
- Fixed an issue with using `LimitFilters` from Coherence*Extend clients.

1.3.5 Coherence*Web Enhancements and Fixes

- Added support for `HttpOnly` cookie attribute in instrumented Coherence*Web web applications.
- Certified Coherence*Web 12*c* (12.1.2) on WebSphere 8.5
- Fixed declaring custom session cache config file using GlassFish SPI.
- Fixed the session affinity token configuration in GlassFish SPI.
- Fixed a potential `NullReferenceException` during concurrent access to a session attribute from a Traditional or Split `ISessionModel`.
- Fixed a regression that caused Coherence*Web extend clients to startup TCMP services.
- Fixed "`attempt to exit session`" `IllegalStateException` during concurrent invalidation by multiple session reapers.
- An exception is no longer thrown when a duplicate `HttpSessionListener` is added in a Web application.
- Fixed Coherence*Web to honor the JVM system property `coherence.cache.configuration.path`.
- The default value for `coherence-session-weblogic-compatibility-mode` system property was incorrect. The default value has been changed to `true`.
- Fixed the issue that caused some Coherence*Web configuration options from being registered when configured by using system properties.
- Added `Mbean` attributes to control storage for Coherence Web.
- Hybrid distribution controller automatically configured when local session attributes is enabled.
- Moved session-cache-config.xml into `coherence-web.jar`. It is no longer inside `coherence-web-spi.war` and `webInstaller.jar`.

- No longer use death certificate cache in Coherence*Web.
- The `coherence-web.jar` is now distributed in the `coherence/lib` directory.
- Added `WebLogicHttpSessionManagerMBean` for WebLogic Coherence*Web SPI.
- Introduced a simpler way of configuring the session locking mode.

1.3.6 Exabus Framework Enhancements and Fixes

- Optimized request latency taking advantage of exabus IO thread pooling
- Resolved an issue in `MessageBus` death-detection which could have lead to the wrong service member being terminated.
- Resolved file descriptor leak in Exabus SocketBus.
- Resolved flow control issue causing stuck nodes for services running on Exabus SocketBus.
- Resolved a service shutdown race condition which could result in an `IllegalStateException`.

1.3.7 Serialization Framework Enhancements and Fixes

- Fixed a potential race condition where a `ConfigurablePofContext` could be used before it is fully initialized.
- Fixed an issue with PofExtractor when targeting extraction at the key and operating against the original binary
- Made `PofExtractor` tolerant to null values.
- Fixed exception handling logic to no longer hide the `ClassNotFoundException` when an `EntryProcessor` class could not be found at deserialization time in a storage member by the default serializer.

1.3.8 REST Framework Enhancements and Fixes

- Fixed registration of custom REST aggregators and processors.
- Upgraded Coherence REST API to work with Grizzly 2.2.1.
- Upgraded the Coherence REST API to use Jackson 1.9.2 libraries.
- Upgraded the Coherence REST API to use Jersey 1.17.1 libraries.
- Added support for the registration of multiple Coherence*Extend REST applications within a single proxy.
- Added support for handling REST requests with the Simple embedded HTTP server.
- Added support for conditional `GET`s to the REST API.
- Added support for pluggable query engines to the REST API.
- Added support for named queries to the REST API.
- Added support for key set retrieval to the REST API.
- Hardened Coherence REST by including a `DefaultServletConextListener` that automatically shuts down a cluster member when the REST application shuts down.

1.3.9 Configuration Enhancements and Fixes

- Fixed an issue which could cause a cluster management node to be stopped by the guardian if the cluster is configured with WKA hostnames and DNS becomes unavailable.
- Coherence parser can now handle UTF-8 with BOM character.
- Fixed an issue in `ClassCastException` with Proxy Server/Extend client when using Optimistic Cache Scheme.
- Fixed a memory leak that occurred when auto commit was enabled for transactional caches with read operations.
- Fixed the Coherence parser so that it can handle UTF-8 with BOM character.
- Fixed an issue that caused `TransactionalCache` to fail with a `DefaultConfigurableCacheFactory`.
- The default unit calculator for distributed cache backing maps is now `BINARY` instead of `FIXED`.
- Fixed an issue which caused the `host-address` element under `authorized-hosts` in operational configuration to be ignored.
- Incremented the schema version for 12*c* (12.1.2).
- Provided a Namespace handler for Spring beans to replace the `DefaultConfigurableCacheFactory` extension.
- Removed the `je.jar` (Berkeley DB) from `coherence.jar` manifest classpath.
- Modified the XSD to allow any XML element to be overridden by using a system-property.
- Fixed an issue which caused the custom service-failure-policy configuration to be ignored.
- The `<connect-timeout>` configuration element has been moved from under `<tcp-initiator>` to under `<initiator-config>`.

1.3.10 Coherence Container Enhancements and Fixes

- Provide a warning if a GAR file is deployed without a Coherence cluster configuration.

1.3.11 Other Enhancements and Fixes

- Additional permissions added to the security policy for Java security manager.
- In the previous beta release, key library files were not installed properly for the TopLink component.
- Coherence now ships a supplemental installer for Oracle install homes created with the Quick Install. This additionally installs the Javadoc and examples in the install home.
- Oracle Coherence 12*c* (12.1.2) RESTful interfaces now depend on the Jackson 1.9.2 and the Jersey 1.17.1 libraries.
- Coherence now includes Jackson and Jersey from the class path `oracle_common/modules`.
- Fixed an issue where invoking the `invokeAll(colKeys, processor)` method resulted in many assertion failures.

- Fixed an issue that caused the `entrySet()` method to return a smaller result if the indexed value was a collection or array.
- Made changes to Flash and RAM Journal to avoid running out of disk space.
- Implemented `hashCode` method on the `PartitionSet` interface.
- Improved the accuracy of machine death detection by IP monitor and reduced the likelihood of false positives.
- Some events could be lost if the primary dies while changes have been updated on the backup but before the events could be sent to the listeners
- Guardian thread logging now includes monitor and synchronization information. In addition to `logStackTraces()`, `GuardSupport` has two new methods: `getThreadDump()` and `getDeadlocks()`.
- Reduced contention in `PortalCacheProvider.get()` method during heavy concurrent access to the same key.
- Fixed a regression causing incomplete results for queries that use indexes against collection-based extractors.
- Fixed Issue with `IllegalMonitorState` being thrown while performing a query during rapid updates.
- Updated embedded version of Berkeley DB included with Coherence to 5.0.
- Improved the process of locating the `MANIFEST.MF` file.
- Added support for the SLF4J logging framework.
- Optimized permission checking when the security framework is used such that running a server in a `Subject` context (`Subject.doAs`) will not require repeated authentications.
- Added the missing security permissions in `security.policy`.
- Fixed an issue to make sure that the transaction timeout value is always reset when a connection is acquired from the pool. This ensures that transactions do not time out unexpectedly.
- Fixed an issue where 0.0.0.0 (`IP_ANY`) could not be used to listen on all IP addresses.
- Resolved a race condition which could result in a `NullPointerException` during service shutdown.
- Fixed the incorrect Jersey library versions that appeared in the `coherence-rest.jar` manifest file.
- Fixed an issue that could cause an assertion error when encountering extremely long communication delays during service startup.
- Fixed an issue where Coherence cannot read its manifest file to obtain the version number.
- Moved the `EventDeathException` class into the `com.tangosol.net.events` package.
- Redirect logging to `System.err` if the configured logger is backlogged to avoid an overflow of the log queue.
- Reduced memory consumption overhead for selective indices.
- Coherence Javadoc now ships as a single JAR file in the `COHERENCE_HOME:/coherence/doc/api` folder. For Java, the `coherence-api.htm` has been

removed.

- When selected, the Coherence examples now install under the `examples` directory of the Coherence home.

1.4 Oracle Coherence for .NET 12*c* (12.1.2)

New features, improvements, and bug fixes have been added to these Oracle Coherence for .NET components:

- Partitioned Cache Enhancements and Fixes
- Coherence*Extend Framework Enhancements and Fixes
- Coherence*Web Enhancements and Fixes
- Other Enhancements and Fixes

1.4.1 Partitioned Cache Enhancements and Fixes

- Fixed an issue where `Request.RequestStatus.WaitForResponse` throws `ArgumentOutOfRangeException` instead of `RequestTimeoutException`.
- Fixed the issue that caused a `java.lang.IllegalStateException: invalid minute offset` error to be thrown for non-standard timezone.
- Fixed an issue where `AllFIlter` was throwing an `ArrayIndexOutOfBoundsException` when initialized with an empty filter array.
- Improved resiliency of an index when a cache entry causes an exception during index update. The index is no longer dropped, so that queries relying on the index continue to execute efficiently. Corrupted entries are excluded from query results.

1.4.2 Coherence*Extend Framework Enhancements and Fixes

- Coherence for .NET Extend connections now have correct timestamps in the proxy server log messages and connection MBeans.
- An issue with the number of events received by key and filter listeners has been resolved.
- Due to a bug in `TimeZone.CurrentTimeZone.GetUtcOffset()` in the .NET Framework 4.5, `PofStreamWriter.WriteLocalDateTime()` may write an incorrect value for certain `DateTime` values. Coherence has implemented a workaround to avoid this problem.
- Removed examples from the .NET distribution.
- An issue with an Extend client receiving multiple copies of an event has been fixed.
- Fixed an issue with the default `<connect-timeout>` value for Coherence for .NET.
- When the first property of a user object is not read, later reads of the object can return the type and version, rather than the actual property value.
- Fixed an issue where `WindowsFormsCacheListener` could hang processing events.
- Fixed an issue where calling `NamedCache.invoke` using POF/Extend when payload contains circular reference resulted in `java.lang.StackOverflowError`.
- Fixed the `NullReferenceException` that was thrown during deserialization of a user-defined object when `<references-enabled>` is set to `true`.

- Fixed an issue where `ClassCastException` could be thrown when `LimitFilter` is used with custom `Comparator`.
- Connection timeouts are now honored by .NET clients.
- Fixed an issue where incorrect results can be returned to an Extend client using a `LimitFilter` when the Coherence cluster has more than one storage node.
- Resolved an issue in enabling POF annotation's auto-indexing feature in .NET.
- Added .NET Extend client support for the `name` service.
- Fixed an issue on .NET and C++ where `ExtractorEventTransformer` methods throw `NullPointerExceptions` when the old or new `ValueExtractor` is null.
- Added missing `<logger-name>` to `<logging-config>` in `coherence.xsd` for .NET.
- Fixed `ExtractorEventTransformer` serialization issue which caused new value extractor to be ignored on the server when different from the old value extractor.
- Fixed `ExtractorEventTransformer.Equals` to throw a `NullReferenceException` when old or new extractor is null.
- Installing Coherence for .NET in quiet mode will no longer throw an error if Microsoft Help 2.x Runtime is not present.
- Fixed .NET `ContinuousQueryCache` missing entries when the cache size is large.
- Implemented named initiators for placing connect host and port information in operational configuration.
- Support for named default serializers has been added to the configuration files.
- Removed a potential deadlock which could occur during Coherence*Extend client request timeout.
- Improved performance of `SafeConfigurablePofContext` by caching the "missed" classes.
- Ported `NamedSerializer` support into .NET and C++.
- Added `transformer` to `ContinuousQueryCache`.
- Moved `ICacheEventTransformer` and `ICacheTrigger` from `Tangosol.Util` to `Tangosol.Net.Cache`.
- The `TCPAcceptor` and `TCPInitiator` elements have been removed from the `cache-config.xml` configuration file.
- Removed the redundant re-resolution of remote addresses in `TcpInitiator.openConnection()` method.
- Fixed an issue with using `LimitFilters` from Coherence*Extend clients.

1.4.3 Coherence*Web Enhancements and Fixes

- Fixed a potential `NullReferenceException` during concurrent access to a session attribute from a Traditional or Split `ISessionModel`.
- Fixed an issue with the Asp.NET `SessionCleanupListener` that caused an `IllegalThreadStateException` to be thrown after a Cluster restart.
- Restored the ability to specify the cache name used by the ASP.NET Session Provider with the "`cacheName`" attribute
- Fixed `LockRecursionException` in `AttributeHolder` when accessing session data in Web Session Model.

- Moved session-cache-config.xml into `coherence-web.jar`. It is no longer inside `coherence-web-spi.war` and `webInstaller.jar`.

1.4.4 Other Enhancements and Fixes

- Coherence now ships a supplemental installer for Oracle install homes created with the Quick Install. This additionally installs the Javadoc and examples in the install home.
- Oracle Coherence 12*c* (12.1.2) RESTful interfaces now depend on the Jackson 1.9.2 and the Jersey 1.17.1 libraries.
- Coherence now includes Jackson and Jersey from the class path `oracle_common/modules`.
- The Coherence .NET version number has been changed to support the .NET version convention that supports only up to 4 digits (`N.N.N.N`). For more details on this enhancement, see "Change to Coherence .NET Version Numbering" on page 2-2.
- Fixed an issue that caused the `entrySet()` method to return a smaller result if the indexed value was a collection or array.
- Coherence for .NET now supports Microsoft Visual Studio 2008 or higher.
- Example applications are no longer included in the Coherence for C++ and .NET distributions.
- Pof Serialization of .NET Decimals now more closely follows Java's format.
- Documented the implementation of `Dispose()` in `INamedCache` calls `Release()` of the cache.
- Coherence Javadoc now ships as a single JAR file in the `COHERENCE_HOME:/coherence/doc/api` folder. For Java, the `coherence-api.htm` has been removed.
- When selected, the Coherence examples now install under the `examples` directory of the Coherence home.
- The `<connect-timeout>` configuration element has been moved from under `<tcp-initiator>` to under `<initiator-config>`.

1.5 Oracle Coherence for C++ 12*c* (12.1.2)

New features, improvements, and bug fixes have been added to these Oracle Coherence for C++ components:

- Partitioned Cache Enhancements and Changes
- Coherence*Extend Framework Enhancements and Fixes
- Other Enhancements and Fixes

1.5.1 Partitioned Cache Enhancements and Changes

- Fixed the issue that caused a `java.lang.IllegalStateException: invalid minute offset` error to be thrown for non-standard timezone.
- Fixed an issue where `AllFIlter` was throwing an `ArrayIndexOutOfBoundsException` when initialized with an empty filter array.

- Improved resiliency of an index when a cache entry causes an exception during index update. The index is no longer dropped, so that queries relying on the index continue to execute efficiently. Corrupted entries are excluded from query results.

1.5.2 Coherence*Extend Framework Enhancements and Fixes

- Fixed an issue where Coherence for C++ may read freed memory during network address lookup.
- Updated the Coherence for C++ header files so that applications can be compiled using GCC 4.7.
- An issue with the number of events received by key and filter listeners has been resolved.
- An issue with an Extend client receiving multiple copies of an event has been fixed.
- Fixed an issue where calling `NamedCache.invoke` using POF/Extend when payload contains circular reference resulted in `java.lang.StackOverflowError`.
- Fixed an issue where `ClassCastException` could be thrown when `LimitFilter` is used with custom `Comparator`.
- When the first property of a user object is not read, later reads of the object can return the type and version, rather than the actual property value.
- Fixed a potential core dump during application shutdown.
- Added a Coherence for C++ distribution for Solaris x86 32-bit for use with the STLport library.
- Added support for adding map listener through configuration.
- Implemented C++ Extend client support for the `name` service.
- Made C++ `LimitFilter::m_vComparator` mutable, to better match how it is used.
- Fixed an issue on .NET and C++ where `ExtractorEventTransformer` methods throw `NullPointerExceptions` when the old or new `ValueExtractor` is null.
- Support for named default serializers has been added to the configuration files.
- Fixed an issue that could cause a crash when registering POF annotated classes.
- Fixed an issue where incorrect results can be returned to an Extend client using a `LimitFilter` when the Coherence cluster has more than one storage node.
- Fixed an issue with C++ Coherence*Extend client ping requests.
- Fixed an issue where Coherence*Extend for C++ clients may not detect a connection being closed by the proxy service.
- Updated the Coherence for C++ distributions to include the header files for the `NearCache` and `LocalCache` classes.
- Fixed an issue which could cause C++ Class and Allocation `HeapAnalyzers` to crash during initialization.
- Fixed a syntax error in the `COH_REGISTER_EXECUTABLE_CLASS` macro.
- Implemented named initiators for placing connect host and port information in operational configuration.
- Changed C++ `COH_STATIC_INIT` methods to be called after library load phase to avoid `DllMain` dependency issues on Windows.

- Ported `NamedSerializer` support into .NET and C++.
- Added `transformer` to `ContinuousQueryCache`.
- The `TCPAcceptor` and `TCPInitiator` elements have been removed from the `cache-config.xml` configuration file.
- Removed the redundant re-resolution of remote addresses in `TcpInitiator.openConnection()` method.
- Fixed an issue with using `LimitFilters` from Coherence*Extend clients.

1.5.3 Other Enhancements and Fixes

- Example applications are no longer included in the Coherence for C++ and .NET distributions.
- Fixed a potential `NullReferenceException` during concurrent access to a session attribute from a Traditional or Split `ISessionModel`.
- Coherence now ships a supplemental installer for Oracle install homes created with the Quick Install. This additionally installs the Javadoc and examples in the install home.
- Coherence now includes Jackson and Jersey from the class path `oracle_common/modules`.
- Fixed an issue that caused the `entrySet()` method to return a smaller result if the indexed value was a collection or array.
- Moved session-cache-config.xml into `coherence-web.jar`. It is no longer inside `coherence-web-spi.war` and `webInstaller.jar`.
- Coherence Javadoc now ships as a single JAR file in the `COHERENCE_HOME:/coherence/doc/api` folder. For Java, the `coherence-api.htm` has been removed.
- When selected, the Coherence examples now install under the `examples` directory of the Coherence home.
- The `<connect-timeout>` configuration element has been moved from under `<tcp-initiator>` to under `<initiator-config>`.

1.6 New Features, Enhancements, and Fixes for Coherence 12*c* (12.1.2.0.1)

The following changes and enhancements have been made for the 12*c* (12.1.2.0.1) release.

- New Features for Coherence 12c (12.1.2.0.1)
- Changes and Enhancements for Oracle Coherence for Java 12c (12.1.2.0.1)
- Changes and Enhancements for Oracle Coherence for .NET 12c (12.1.2.0.1)
- Changes and Enhancements for Oracle Coherence for C++ 12c (12.1.2.0.1)

1.6.1 New Features for Coherence 12*c* (12.1.2.0.1)

The following new features have been added to the 12.1.2.0.1 patch release for Oracle Coherence:

- Support for Apache Tomcat application server version 7.*x* added to Coherence*Web—Support has been added for version 7.*x* of the Apache Tomcat application server. For more information, see "Supported Web Containers" in *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web*.
- POF serialization with HotCache—POF serialization can be used with HotCache but requires an update to the POF configuration file to allow for TopLink Grid framework classes to be registered. For more information, see "Using Portable Object Format with HotCache" in *Oracle Fusion Middleware Integrating Oracle Coherence*.
- Wrapper Classes for TopLink Grid Applications—A new context property can enable the generation of wrapper classes for TopLink Grid applications that make Coherence cache inserts. For more information, see "Enabling Wrapper Classes for TopLink Grid Applications" in *Oracle Fusion Middleware Integrating Oracle Coherence*.
- Expiration and eviction policies for RAM and flash journal—Expiration and eviction policies can restrict the number of entries to store, and automatically evict entries when the journal becomes full. Both the sizing of entries and the eviction policies can be customized. For more information, see "Using Journal Expiry and Eviction" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.
- New attribute for the `ConnectionManagerMBean` MBean—A new attribute, `UnauthorizedConnectionAttempt`, identifies the number of connection attempts that were made from unauthorized hosts. For more information, see "ConnectionManagerMBean" in *Oracle Fusion Middleware Managing Oracle Coherence*.
- Concurrent access to the same session instance in Coherence*Web—A new context parameter allows you to use a cache delegator to ensure that the local cache should be used for storing and retrieving the session instance before attempting to use the distributed cache. This feature must be enabled when working with PeopleSoft applications. For more information, see "Getting Concurrent Access to the Same Session Instance" in *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web*.
- Documentation for Spring integration has been moved from the *Oracle Fusion Middleware Integrating Oracle Coherence* to the Coherence Community pages. For information on Integrating Coherence with Spring, see the following URL: `https://java.net/projects/cohspr/`
- Starting with Coherence 12.1.2.0.1, Extend clients support both forward and backward compatibility with cluster proxies. That is, Extend clients can connect to cluster proxies that have lower or higher version numbers (within a major release). For example, a 12.1.2.0.2 Extend client can connect to a 12.1.2.0.1 proxy. Extend client backward compatibility is not supported on proxy versions prior to 12.1.2.0.1, including 12.1.2.0.0 and proxy versions 3.7.1 or earlier.

1.6.2 Changes and Enhancements for Oracle Coherence for Java 12*c* (12.1.2.0.1)

- Hardened the `PacketReciever` against randomly truncated otherwise valid packets.
- Fixed the issue where custom MBean registration assumes the MBean name to be unique.
- Updated the URL redirect to OTN Coherence documentation in the `documentation.htm` file.

- Fixed an issue with the POF Configuration Generator on Windows, which produces an invalid reference in the application descriptor.
- Fixed an issue where `DefaultClusterDependencies.makeClusterName()` throws NPE due to null group address.
- Fixed an erroneous clean-up of locally stored session attributes.
- Added tracking of unauthorized hosts attempting to connect via Coherence*Extend to include logging of a D6 message as well as recording the number of attempts in a new JMX attribute `UnauthorizedConnectionAttempts` on the `ConnectionManagerMBean`.
- Added the ability to define `MapTriggerListeners` for remote clients by using the `<listener>` element.
- Fixed an issue with Elastic Data in which subsequent to a destroy, data was not removed from the journal.
- Fixed an issue where `pof-config-gen.sh` would throw an error when running on Solaris.
- Fixed an issue in the asynchronous SSL implementation that may cause a request to hang when SSL communication is enabled.
- Fixed the Session Reaper to ensure that local attributes are cleaned up when the session invalidation is initiated from a different node.
- Enhanced the Proxy HostIP MBean attribute to show the subport value.
- Fixed a possible Guardian time out issue which might occur when restoring orphaned partitions.
- Reduced the default `<thread-count-max>` value for the proxy service thread pool to 4 times the number of CPU cores (down from 8 times).
- Fixed an issue in which eviction and expiry could not be configured with Elastic Data, which are now both natively supported.
- Fixed a potential `AssertionException` which could occur in Coherence*Web during an application undeployment.
- Fixed an issue which could cause a `NullPointerException` for cluster with heterogeneous transports.
- Fixed a potential race condition between a guarded thread timing out and the thread no longer being guarded.
- Fixed a `NullPointerException` occurring when an event is generated for orphaned partitions with JMX disabled across the cluster.
- Added a new session configuration in Coherence*Web which keeps an local instance of a session in addition to flushing the session to the distribute cache. For more information, see "Getting Concurrent Access to the Same Session Instance" in *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web*.
- Updated the `ChainedComparator` three parameter constructor so that it no longer ignores the third `Comparator`.
- Fixed a custom `Comparator` issue with proxy service load balancing.
- Fixed an issue with the Coherence console allowing the GAR's resources to be available to the console.

- Fixed an issue causing `NamedCache` iterators to return duplicate results to Extend clients from Standard edition clusters.
- Fixed a `NullPointerException` occurring when an event is generated for orphaned partitions with JMX disabled across the cluster.
- Fixed an issue where log messages that are not in the range of destination logger are enqueued.
- Improved `ConnectionException` to return the cause of the exception from the proxy server.
- Hardened the handling of cache deactivation calls.
- Improved the sending of `DiagnosticPackets` such that they are sent to both advertised and preferred ports.
- Fixed an issue with Elastic Data in which an invalid binary exception is reported during an operation that directly or indirectly calls the `keySet` method.
- Fixed an issue where integer value of -31 is not read properly by the POF serializer.
- Hardened the handling of cache misconfigurations by guarding against NPE and UOE to avoid service termination.
- Hardened the transactional cache handling of `javax.transaction.xa.Xid` to allow non-serializable implementations.
- Removed an inaccurate warning regarding incompatible WKA addresses when starting cluster with WKA where at least one host name is specified.
- Fixed an `InFilter` serialization issue.
- Fixed `CacheFactory.shutdown()` to stop the Logger Thread if there is no running cluster.
- Fixed the `NullPointerException` when `DefaultCacheServer.start()` is followed by call to `DefaultCacheServer.shutdown()`.
- Fixed `ReadWriteBackingMap` to honor the unit calculator configured in the cache configuration.
- Changed `DCCF.applyScopeToServiceName` method name to `getScopedServiceName`.
- Fixed Transactional cache to disallow '$' char as part of its `cache-name`.
- Fixed a race-condition which could cause partition transfers to become stuck when using either `tmb`, `sdmb`, or `imb` transports under heavy load.
- Fixed an issue where no error was thrown when creating a transactional cache for which there is no scheme definition in `cache-config`.
- Fixed an issue with `Filter` execution producing high memory overhead due to unnecessary JMX statistic.
- Enhanced `NameService` so that it runs on all nodes.
- Fixed an issue with the dynamic proxy service thread pool size increasing too aggressively under certain load scenarios.
- Added support to Coherence*Web for Tomcat 7.*x* application servers.
- Fixed an issue with `DefaultCacheServer` validating given arguments.

- Fixed an issue where partitions lost notifications were issued for 0 partitions lost; these notifications will no longer be issued.
- Fixed an issue which caused duplicate delivery of transformed `MapEvents`.
- Fixed an issue with `DefaultCacheServer` to be more tolerant with an absent `MANIFEST` file.
- Fixed an issue with the console to ensure the GAR's cache configuration is used if a server command with a GAR file was previously issued.
- Added support for `HttpOnly` cookie attribute in instrumented Coherence*Web web applications.
- Improved POF annotation log message when a requested property index is already in use.
- Fixed `PofHelper.getPofTypeId()` method to return the correct type ID for user types that derive from `ArrayList`.
- Resolved an issue where custom proxies were not being called when ensuring and/or destroying a cache.

1.6.3 Changes and Enhancements for Oracle Coherence for .NET 12*c* (12.1.2.0.1)

- Fixed a .NET 4.5 framework issue with `SynchronizedDictionary`.
- Improved `ConnectionException` to return the cause of the exception from the proxy server.
- Hardened the handling of cache deactivation calls.
- Fixed an issue where integer value of -31 is not read properly by the POF serializer.
- Fixed an issue where type -29 is not read properly by the POF serializer.
- Improved SSL client certificate selection for SSL-enabled Coherence*Extend connections.
- Removed Coherence for .NET installer prerequisite on .NET Framework 2.0.
- Fixed an `InFilter` serialization issue.
- Added `ReducerAggregator` to Coherence for .NET and C++ so that the `ReducerAggregator` can be invoked from these languages.
- Fixed an issue which caused duplicate delivery of transformed `MapEvents`.
- Fixed the Coherence for .NET library to look for the correct XML configuration elements for using network filters, like the `gzip` compression filter.
- Improved POF annotation log message when a requested property index is already in use.
- Fixed `PofHelper.getPofTypeId()` method to return the correct type ID for user types that derive from `ArrayList`.

1.6.4 Changes and Enhancements for Oracle Coherence for C++ 12*c* (12.1.2.0.1)

- Added support for defining class factories with the `<class-factory-name>` element.
- Fixed `LiteSet` so that it can support more than two elements.

- Added support for defining class initialization parameters with the `<init-params>` element.
- Fixed an issue with `String::substring` prematurely ending a substring search.
- Improved `ConnectionException` to return the cause of the exception from the proxy server.
- Hardened the handling of cache deactivation calls.
- Fixed an issue where integer value of -31 is not read properly by the POF serializer.
- Fixed an `InFilter` serialization issue.
- Fixed `adapter_map/boxing_map` iterator so that it initializes correctly when the map size is 1.
- Added `ReducerAggregator` to Coherence for .NET and C++ so that the `ReducerAggregator` can be invoked from these languages.
- Fixed an issue which caused duplicate delivery of transformed `MapEvents`.
- Improved POF annotation log message when a requested property index is already in use.

1.7 New Features, Enhancements, and Fixes for Coherence 12*c* (12.1.2.0.2)

The following changes and enhancements have been made for the 12*c* (12.1.2.0.2) release.

1.7.1 New Features for Coherence 12*c* (12.1.2.0.2)

The following new features have been added to the 12.1.2.0.2 patch release for Oracle Coherence:

- Extend clients support both forward and backward compatibility with cluster proxies. That is, Extend clients can connect to cluster proxies that have lower or higher version numbers (within a major release). For example, a 12.1.2.0.2 Extend client can connect to a 12.1.2.0.1 proxy. Extend client backward compatibility is not supported on proxy versions prior to 12.1.2.0.1, including 12.1.2.0.0 and proxy versions 3.7.1 or earlier.

1.7.2 Changes and Enhancements for Oracle Coherence for Java 12*c* (12.1.2.0.2)

- Fixed an issue that off-heap (NIO) backing map could lost cache entries during failover.
- Fixed an issue where `MultiExtractor` was not using indexes for its `ValueExtractor`.
- Improved POF date and time deserialization performance.

- Provided limited support for `PofValueParser` and `PofNavigator` when object identity/reference is enabled.
- Fixed an issue with `RefreshableAddressProvider` in which previously unaccepted addresses may not be retried after successful connection.
- Fixed an issue where a `ConcurrentModificationException` may be thrown when invoking `ConditionalRemove` on a `ReplicatedCache` from an Extend client.
- Fixed `NotSerializableException` in Coherence*Web for WebLogic Portal components during session invalidation.
- Fixed an issue which could cause `PacketReceiver` to spin when extracting packets from buffer.
- Fixed an error encountered on Solaris x86-64 while performing OPatch.
- Fixed a regression in the cache listener which can result in increased CPU overhead within cache servers, and performance reduction on clients, primarily impacting NearCaches.
- Changed the proxy service worker threads to close Coherence*Extend connections asynchronously when a connection will be closed due to an error.
- Fixed a serialization buffer allocation performance issue.
- Reduced the contention when `ReadWriteBackingMap` is subject to high load and evictions.
- Fixed an issue with creating custom components specified in the cache configuration when Coherence is on the boot classpath.
- Fixed an issue which could cause services timeout during startup.
- Updated `PofValue.getValue()` API documentation to indicate that the type must be explicitly specified for primitive types.
- Fixed an issue with events not being delivered to pre-3.7.1 Coherence *Extend clients.
- Fixed a bug where `Filter` queries that use indexes could return the wrong results if the query executes at the same time as multiple concurrent updates to the same cache.
- Fixed an issue where WKA address list is unnecessarily refreshed during message serialization.
- During session invalidation, `ConcurrentModificationException` is now logged only when the logging level is set to `FINEST`.
- Fixed a `ConcurrentModificationException` that may occur in `RefreshableAddressProvider` upon retrieving the next address when the last address in the list was rejected.
- Reduced contention on the Proxy service for key-based requests that target the same keys.
- Fixed `ThreadAverageActiveCount` in `Service` MBean to report the correct value.
- Fixed an issue where resources containing '`#`' character in filename/path failed to load.
- Fixed an issue where an Extend client doesn't receive delete events from a replicated cache.

- Improved the performance of `BetweenFilter` when there is a corresponding index available.
- Fixed an issue in `ensureCache` which can result in client threads becoming stuck if the `ensureCache` was unresponsive.
- When using the Coherence Cache Provider with WebLogic Portal, the cache configuration may be set using a system property.
- Fixed an issue where partitioned cache could become uninterruptable due to prolonged orphaned partitions.
- Fixed an issue in which we may duplicate the call to the CacheStore on failover for a successfully stored entry.
- Fixed an issue where Coherence*Web on Tomcat 7.0 reported incorrect Tomcat version information in the log file.

1.7.3 Changes and Enhancements for Oracle Coherence for .NET 12*c* (12.1.2.0.2)

- Corrected an issue when iterating the entries, keys, or values of a `ContinuousQueryCache`.
- Provided limited support for `PofValueParser` and `PofNavigator` when object identity/reference is enabled.
- Fixed an event delivery issue for cache listeners registered with filters that use a `PofExtractor`.
- Fixed a serialization buffer allocation performance issue.
- Changed `Coherence.dll` for Coherence .NET to have no diagnostic instrumentation overhead. `Coherence.pdb` is still provided for debugging purposes.
- Updated `PofValue.getValue()` API documentation to indicate that the type must be explicitly specified for primitive types.
- Fixed an issue with events not being delivered to pre-3.7.1 Coherence *Extend clients.
- Added a `SimplePrincipal` which derives from `GenericPrincipal`. `SimplePrincipal` uses the `Identity.Name` property to determine object equality.

1.7.4 Changes and Enhancements for Oracle Coherence for C++ 12*c* (12.1.2.0.2)

- Fixed a serialization buffer allocation performance issue.
- Updated `TypedMethod.hpp` so that `Xcode` (`clang`) no longer warns about unused functions in the file.
- Changed `DetachFinalizer::set()` to detach from the old object with the old escaped flag.

1.8 New Features, Enhancements, and Fixes for Coherence 12*c* (12.1.2.0.3)

The following changes and enhancements have been made for the 12*c* (12.1.2.0.3) release.

- Changes and Enhancements for Oracle Coherence for .NET 12c (12.1.2.0.3)
- Changes and Enhancements for Oracle Coherence for C++ 12c (12.1.2.0.3)

1.8.1 New Features for Coherence 12*c* (12.1.2.0.3)

A new Weblogic Web Report (`report-web-weblogic.xml`) performance report is available for Coherence*Web on the WebLogic Server platform. It provides all of the information as the Web Report, with the addition of the WebLogic Server name and WebLogic Server domain name. See Section 2.13, "New Performance Report for Coherence*Web in WebLogic Server Environments."

1.8.2 Changes and Enhancements for Oracle Coherence for Java 12*c* (12.1.2.0.3)

- Fixed an issue where an invocation of `QueryPlus.main` preceded by `QueryHelper.createFilter` caused a `NullPointerException`.
- Fixed an issue that resulted in an `AssertionException` or deadlock when under high load using expiry or eviction.
- Fixed an issue where `NearCache` event interceptors defined in the `cache-config` file are not registered when using `ExtensibleConfigurableCacheFactory`.
- Fixed an issue where `<maximum-size>` was not taking effect for `flashjournal-manager`.
- Removed improper `Service.heartbeat()` call from non-service thread that resulted in a concurrent `checkGuardables` call by service thread.
- Fixed a concurrent access issue with UUIDs.
- Fixed an issue where a service failed to restart when configured to use SSL.
- Fixed an issue with `InFilter` that could result in poor performance if used with an initial filter that reduces the data set significantly.
- `BinaryEntry.remove(fSynthetic)` will now generate synthetic delete events when `fSynthetic` is `true`.
- Fixed an issue with `UpdaterProcessor` that could cause missing entry update in `CacheStore`, `BinaryEntryStore`, and `Listeners`.
- Fixed an issue where internal session attribute names are returned without stripping the `InternalWLSAttribute` prefix.
- Fixed a POF serialization issue for negative Dates with nonzero sub-seconds.
- Fixed an issue where session attributes are mandated to implement `java.io.Serializable`.
- Fixed an issue which could cause the server to fail to join the cluster with `tmbs` protocol.
- Enhanced `BetweenFilter` to work like a normal `ExtractorFilter` rather than an `AndFilter`, which improves efficiency of applying indexes and evaluating values.
- Fixed missing Live Events packages from Javadoc.
- Fixed an issue `ConfigurableCacheMap` methods does not return correct values with `ObservableSplittingBackingCache`.
- Changed Extend security to use the `Subject` provided by the client rather than the WebLogic `Subject` when running an Extend Proxy inside WLS Managed Servers. This change removes the need to use WebLogic APIs to obtain the client's `Subject`

and instead standard JAAS APIs can be used and hence removes the need for application code to have any dependency on WLS.

- Fixed an issue which could cause `TcpAcceptor` to hang when decoding non-Coherence messages.
- Fixed an issue where `cmd` scripts shipped by Coherence fail to handle `JAVA_HOME` path containing spaces and parentheses.
- Fixed an issue with `nio-file-manager` using an appropriate directory when not provided.
- For Coherence*Web configurations on WebLogic Server, a new report "`reports/report-web-weblogic.xml`" is available.

1.8.3 Changes and Enhancements for Oracle Coherence for .NET 12*c* (12.1.2.0.3)

- Fixed a concurrent access issue with UUIDs.
- Enhanced `BetweenFilter` to work like a normal `ExtractorFilter` rather than an `AndFilter`, which improves efficiency of applying indexes and evaluating values.

1.8.4 Changes and Enhancements for Oracle Coherence for C++ 12*c* (12.1.2.0.3)

- Fixed a concurrent access issue with UUIDs.
- Fixed an issue where dependent threads of a Coherence service may fail to stop during service shutdown.
- Enhanced `BetweenFilter` to work like a normal `ExtractorFilter` rather than an `AndFilter`, which improves efficiency of applying indexes and evaluating values.
- Fixed a typo in the guard macro for `XorFilter.hpp`.

1.9 New Features, Enhancements, and Fixes for Coherence 12*c* (12.1.2.0.4)

The following changes and enhancements have been made for the 12*c* (12.1.2.0.4) release.

- Changes and Enhancements for Oracle Coherence for Java 12c (12.1.2.0.4)
- Changes and Enhancements for Oracle Coherence for .NET 12c (12.1.2.0.4)
- Changes and Enhancements for Oracle Coherence for C++ 12c (12.1.2.0.4)

1.9.1 Changes and Enhancements for Oracle Coherence for Java 12*c* (12.1.2.0.4)

- Fixed an issue where a `NullException` was seen upon stop of a FlashJournalRM Daemon thread during service restart.
- Fixed an issue which could cause the proxy service thread count to grow unnecessarily large.
- Fixed an issue where daemon threads were not using the specified service classloader.
- Fixed an issue that could cause a `NoSuchElementException` during Ram Journal eviction.
- Fixed an issue resulting in guardian errors on startup when using EventInterceptors and a large number of concurrent requests per member.

- Fixed an issue when using MultiExtractor as an index that may result in inaccurate indexed data.
- Added support for enabling or disabling specified SSL protocols.
- Fixed an issue where the `<cipher-suites>` element in a cluster configuration caused a `SAXParseException`.
- Fixed an issue with invoking `GuardSupport.heartbeat` during execution of post commit LiveEvents.
- Fixed an issue where an `ArrayIndexOutofBoundsException` occurs during a `ClusterModel.getMemberIds()` call due to some of cluster members going down during the operation.

1.9.2 Changes and Enhancements for Oracle Coherence for .NET 12*c* (12.1.2.0.4)

- Removed "DPR" from the Coherence for .NET version string.
- Coherence*Extend message logging can now be enabled by setting the `COHERENCE_MESSAGING_DEBUG` environment variable to `true` or by adding the `<COHERENCE_MESSAGING_DEBUG>true</COHERENCE_MESSAGING_DEBUG>` as a child element of `<coherence>` in the `app.config` file.

1.9.3 Changes and Enhancements for Oracle Coherence for C++ 12*c* (12.1.2.0.4)

There are no changes or enhancements specific to C++ in Oracle Coherence 12*c* (12.1.2.0.4).

1.10 Known Problems and Workarounds

This section describes issues that are know at the time of release.

1.10.1 WebLogic Server Domain Size Limitations

If you are using Managed Coherence Servers with relatively large clusters, you may encounter Administration Server Memory Consumption issues. For more information on this problem and possible workarounds, see "Administration Server Memory Consumption and JMX Notifications" in *Oracle Fusion Middleware Release Notes for Oracle WebLogic Server Release 12c (12.1.2)*.

1.10.2 Manifest File Required for GAR File Deployment

When using the `DefaultCacheServer` class to deploy a Grid ARchive (GAR), the deployment fails if the archive does not contain a manifest file. Use the Java `jar`

command to create the archive and ensure that a `MANIFEST.mf` file is included in the `META-INF` directory.

1.10.3 Deploying a GAR File from the CacheFactory Console

When using the `server` command to deploy a GAR from the `CacheFactory` console, the cache configuration file in the archive fails to override the default cache configuration file that is included in the `coherence.jar` library. To work around this issue, start the console and ensure that all configuration files are located on the classpath before the `coherence.jar` file. If you are connecting to a cache server that has been started using a GAR file, then the cache configuration file on the `CacheFactory` client must define the same scope name being used for the GAR file on the cache server; otherwise, the client is unable to use any defined caches.

1.10.4 JVM Crashes on Linux

On Linux with JDK 1.7, the JVM may crash with a `SIGSEGV at ParMarkBitMap::verify_clear()` error. In JDK 1.7, "large pages" support is enabled by default. To work around this issue, use the `-XX:-UseLargePages` Java command line option to disable large pages support. Bug JDK-8007074 has been filed for this issue.

For more information on configuring large page sizes, see "Tune the Huge (Large) Pages Configuration" in *Oracle Fusion Middleware Administering Oracle Coherence*.

1.10.5 JVM Crashes Due to Offset Error

The JVM crashes and reports a `guarantee(this->is8bit(imm8)) failed: Short forward jump exceeds 8-bit @ offset` error during runtime compilation. To work around this issue disable the compilation of the method indicated in the crash using the Java command line option `-XX:CompileCommand=exclude,`*`fully-qualified-method-name`*. Bug JDK-8010437 has been filed for this issue.

1.10.6 Coherence Cache Override Not Working

This problem can occur if you are using the Coherence cache override feature in WebLogic Server. If the WebLogic Server Configuration Wizard (`config.sh`) is used to create a domain and the WebLogic Coherence Cluster Extension template is specified, then a Coherence cluster will be defined. The Coherence cluster will be associated with any Managed Server or WebLogic Server cluster that is also created by the Configuration Wizard. If no Managed Server or WebLogic Server cluster is created, then the Coherence cluster will be associated with the Administration Server. This association between the Coherence cluster and the servers is not completely defined using the WebLogic Server configuration tool, which results in the Coherence cache configuration override file not being detected by the Coherence cluster.

Use the following workaround:

1. Start the domain created with the Configuration Wizard and connect using the WebLogic Server Administration Console.
2. In the left pane of the Administration Console, expand **Environment** and select **Coherence Clusters**.
3. Select your Coherence cluster. The Coherence cluster settings page is displayed.
4. Select the **Members** tab, which displays all of the members of the Coherence cluster.

5. Deselect the servers and clusters that are members of the Coherence cluster and click **Save**.
6. Reselect the servers and clusters that are the desired members for the Coherence cluster and click **Save**.

This will perform a complete association between the Coherence cluster and the targeted servers, which is required to detect and use the specified Coherence cluster cache configuration override file.

For more information, see "Overriding a Cache Configuration File" in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server*.

1.10.7 Problems with the Cache Size Report

The report for cache size is missing entries when the same cache exists on multiple services.

To workaround this problem, customize the cache size report to have a service column with the `group-by` tag set to `true`. For more information on the cache size report, see "Understanding the Cache Size Report" in *Oracle Fusion Middleware Managing Oracle Coherence*.

1.10.8 Building C++ Applications with OS X Mavericks (for Coherence 12.1.0.2)

When building C++ applications with Apple OS X 10.9 (Mavericks), you must compile with the command "`g++`" (as opposed to "`CC`") and you must use the parameter "`-mmacosx-version-min=10.7`".

1.11 Spring Support Moved to Coherence Community Projects

Spring support has been moved from the Coherence 12*c* (12.1.2) release to Coherence Community projects. Coherence Community projects provide example implementations for commonly-used design patterns based on Oracle Coherence. See the following URL:

`https://java.net/projects/cohspr/`

1.12 Deprecated Features

The following features have been deprecated or removed from the Coherence 12*c* (12.1.2) release.

- ActiveCache (active-cache.jar)
- LH Store Manager
- NamedCache lock APIs
- DefaultConfigurableCacheFactory Class
- XmlConfigurable Interface

1.12.1 ActiveCache (active-cache.jar)

ActiveCache (`active-cache.jar`), the collection of WebLogic Server features which allow deployed applications to easily use Coherence data caches and seamlessly incorporate Coherence*Web for session management, has been deprecated in the current release. ActiveCache can still be used with applications that have been developed to run on older versions of WebLogic Server.

Instead of ActiveCache, users must migrate to Managed Coherence Servers. For more information on Managed Coherence Servers, see *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*.

1.12.2 LH Store Manager

The LH store manager is deprecated as of Coherence 12*c* (12.1.2) release. Use Berkeley DB for similar functionality. For more information, see "Replacement for the Deprecated LH File Manager" on page 3-7

1.12.3 NamedCache lock APIs

The `NamedCache lock` APIs are deprecated. Use the locking support that is provided by the entry processor API instead (`EntryProcessor` for Java and C++, `IEntryProcessor` for .NET). For more information, see "Replacement for the Deprecated NamedCache Lock APIs" on page 3-8.

1.12.4 DefaultConfigurableCacheFactory Class

The `com.tangosol.net.DefaultConfigurableCacheFactory` class has been deprecated. It remains part of the Coherence distribution only to provide backwards compatibility for those applications that have extended it or depend on specific implementation semantics.

Developers are strongly encouraged to refactor implementations that depend on this class to use the extension mechanisms used by the `ExtensibleConfigurableCacheFactory` class. No further development or enhancement of this class will occur in the future.

It is strongly recommended that developers get a `ConfigurableCacheFactory` instance by using the `CacheFactory.getCacheFactoryBuilder().getConfigurableCacheFactory()` method, rather than instantiate a `DefaultConfigurableCacheFactory` instance directly.

1.12.5 XmlConfigurable Interface

The `com.tangosol.run.xml.XmlConfigurable` interface has been deprecated in the Coherence 12*c* (12.1.2) release. This interface is no longer required due to injection support. For a different technique to initialize implementations, see "Replacement for the Deprecated XmlConfigurable Interface" on page 3-8.

The following list describes the interfaces, classes, and methods affected by the deprecation of the `com.tangosol.run.xml.XmlConfigurable` interface.

Interfaces being deprecated:

This interface will no longer be required due to injection support.

- `com.tangosol.run.xml.XmlConfigurable`

Methods on Interfaces being deprecated:

This method is no longer required due to injection support.

- `com.tangosol.util.Controllable.configure(XmlElement xml)`

Classes being deprecated:

These classes are replaced by `ParameterizedBuilder` implementations.

- `com.tangosol.io.ConfigurableSerializerFactory`

- com.tangosol.net.ConfigurableAddressProviderFactory

Interfaces that will be affected:

All implementations of these interfaces will be configured through injection.

- com.tangosol.net.BackingMapManagerContext
- com.tangosol.net.CacheService
- com.tangosol.net.Cluster
- com.tangosol.net.InvocationService
- com.tangosol.net.NameService
- com.tangosol.net.PartitionedService
- com.tangosol.net.ProxyService
- com.tangosol.net.Service
- com.tangosol.util.Service

Classes that will be affected:

These classes will be configured through injection.

- com.tangosol.coherence.reporter.ColumnView
- com.tangosol.coherence.reporter.ReporterColumnView
- com.tangosol.coherence.rest.config.RestConfig
- com.tangosol.coherence.servlet.AbstractHttpSessionCollection
- com.tangosol.coherence.servlet.CoherenceHttpSessionCollection
- com.tangosol.coherence.servlet.SplitHttpSessionCollection
- com.tangosol.coherence.transaction.internal.ClusterWrapper
- com.tangosol.coherence.transaction.internal.ServiceWrapper
- com.tangosol.coherence.weblogic.PortalCacheProvider
- com.tangosol.io.bdb.BerkeleyDBBinaryStoreManager
- com.tangosol.io.pof.ConfigurablePofContext
- com.tangosol.net.ConfigurableAddressProvider
- com.tangosol.net.SocketOptions
- weblogic.servlet.internal.session.WebLogicSPIHttpSessionIdGenerator

Packages that will be affected:

All classes will be configured through injection.

- com.tangosol.coherence.reporter.locator

Classes that will be affected:

No change required, as these classes are already deprecated.

- com.tangosol.net.CompressionFilter (already deprecated)
- com.tangosol.net.DefaultConfigurableCacheFactory (already deprecated)
- com.tangosol.net.security.AbstractEncryptionFilter (already deprecated)
- com.tangosol.net.security.ClusterEncryptionFilter (already deprecated)

Classes that will be affected:

These classes either do not support XML configuration, perform passthrough, or throw `UnsupportedOperationException` errors.

- `com.tangosol.io.journal.AbstractJournalRM`
- `com.tangosol.io.journal.FlashJournalRM`
- `com.tangosol.io.journal.RamJournalRM`
- `com.tangosol.net.WrapperCacheService`
- `com.tangosol.net.WrapperInvocationService`
- `com.tangosol.net.WrapperService`

Project Resources / XSDs that will be affected:

- `coherence-operational-config-base.xsd`

2

Documentation Errata

This chapter describes changes, enhancements, and corrections made to the Oracle Coherence documentation for 12*c* (12.1.2).

The Coherence documentation for 12*c* (12.1.2) can be found at the following URL:

`http://docs.oracle.com/middleware/1212/coherence/index.html`

This chapter contains the following sections:

2.1 Controlling Cipher Suite and Protocol Version Usage

An SSL socket provider can be configured to control the use of potentially weak ciphers or specific protocol versions.

To control cipher suite and protocol version usage, edit the SSL socket provider definition and include the `<cipher-suites>` element and the `<protocol-versions>` elements, respectively, and enter a list of cipher suites and protocol versions using the `name` element. Include the `usage` attribute to specify whether the cipher suites and protocol versions are allowed (value of `white-list`) or disallowed (value of `black-list`). The default value for the `usage` attribute if no value is specified is `white-list`. For example:

```
<socket-provider>
   <ssl>
   ...
```

```
      <cipher-suites usage="black-list">
         <name>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256</name>
      </cipher-suites>
      <protocol-versions usage="black-list">
         <name>SSLv3</name>
      </protocol-versions>
   ...
   </ssl>
</socket-provider>
```

2.2 Coherence*Extend 12*c* (12.1.2) Backward Compatibility Exception

Coherence 12.1.2.*x* cluster proxies are backward compatible with extend clients versions 3.*x*. That is, 3.*x* clients can connect to 12.1.2.*x* cluster proxies. Backward compatibility in 12.1.2.*x* is an exception to the Coherence*Extend version compatibility policy. Cluster proxies beyond version 12.1.2 may not include the current exception and may not maintain backward compatibility.

2.3 Change to Coherence .NET Version Numbering

The Coherence .NET version number has been changed to support the .NET version convention that supports only up to 4 digits (`N.N.N.N`). Because Oracle version numbers use 5 digits (`N.N.N.N.N`), the 4th and 5th Oracle digits are combined for the 4th .NET version digit. For details on the mapping, see "Coherence .NET Version Number Mapping" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.

2.4 Default Near Cache Invalidation Strategy has Changed

The default near cache invalidation strategy `auto` has changed to ensure that reduced network traffic is prioritized over performance. Set the invalidation strategy to `all` for pre-12*c* (12.1.2) default behavior.

2.5 Deploying GAR Files as Shared Libraries

Deploying a GAR file as a shared library requires additional steps over regular shared library deployment. For more information, see "Deploying Coherence Applications as Shared Libraries" in *Developing Oracle Coherence Applications for Oracle WebLogic Server*.

2.6 Using POF Serialization with GoldenGate Hot Cache

You can use Portable Object Format (POF) serialization with Oracle Coherence GoldenGate HotCache (HotCache). POF is a language-agnostic binary format. POF was designed to be very efficient in both space and time and has become a cornerstone element in working with Coherence. For more information, see "Using Portable Object Format with HotCache" in *Oracle Fusion Middleware Integrating Oracle Coherence*.

2.7 Additional Documentation Support for Hibernate

In addition to "Integrating Hibernate and Coherence" in *Oracle Fusion Middleware Integrating Oracle Coherence*, you can find information on integrating Coherence with Hibernate in the Coherence Community projects at the following URL:

`https://java.net/projects/cohhib`

2.8 Documentation Support for Spring has Moved

Documentation for Spring support has been moved from the Coherence 12*c* (12.1.2) product distribution to the Coherence Community projects. It is no longer described in the *Oracle Fusion Middleware Integrating Oracle Coherence* documentation. For information on integrating Oracle Coherence with Spring, see the following URL:

https://java.net/projects/cohspr/

2.9 Incorrect Default Value for coherence-session-thread-locking Parameter

The description of the `coherence-session-thread-locking` parameter in the "Coherence*Web Context Parameters" appendix in *Administering HTTP Session Management with Oracle Coherence*Web* incorrectly lists the default value as `true`. The correct default value should be `false`.

2.10 Incorrect Information Regarding Sticky Sessions and Locking for Coherence*Web

The "Troubleshooting Locking in HTTP Sessions" section in *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web* indicates:

"Enabling Member, Application, or Thread Locking for HTTP session access indicates that Coherence*Web will acquire a clusterwide lock for every HTTP request that requires access to a session. The exception to this is when sticky load balancing is available and the Coherence*Web sticky session optimization is enabled."

The sentence "*The exception to this is when sticky load balancing is available and the Coherence*Web sticky session optimization is enabled.*" has been removed in the current version of the book because it is confusing. Even if sticky sessions are enabled, Coherence*Web will still hold a clusterwide lock for a session—it simply will not release it after the request has finished.

2.11 Text for Coherence*Extend Version Compatibility has been Revised

The text describing compatibility between Coherence*Extend versions has been revised. Here is the new text:

Compatibility for the extend protocol and POF is maintained between point releases (for example, 1.0, 1.1, and so on) but may not be maintained between major releases (for example, 1.0, 2.0, and so on).

> **Note:** Compatibility requires the use of POF, because POF can support backward compatible serialization change.

Prior to version 12.1.2.0.1, extend clients only support forward compatibility with cluster proxies. That is, extend clients can connect to cluster proxies that have either the same or higher version numbers (within a major release).

Starting with version 12.1.2.0.1, extend clients support both forward and backward compatibility with cluster proxies. That is, extend clients can connect to cluster proxies that have lower or higher version numbers (within a major release). For example, a 12.1.3 extend client can connect to a 12.1.2.0.1 proxy. Extend client backward compatibility is not supported on proxy versions 3.7.1 or earlier.

Coherence 12.1.2 Backward Compatibility Exception

Coherence 12.1.2 cluster proxies are backward compatible with version 3 extend clients (3.3, 3.4, 3.5, 3.6, and 3.7). That is, version 3 clients can connect to 12.1.2 cluster proxies. Backward compatibility in 12.1.2 is an exception to the Coherence*Extend major release compatibility policy that is described above. Cluster proxies beyond version 12.1.2 may not include the current exception and may not maintain backward compatibility.

2.12 Incorrect listing for session-cache-config.xml File Without a Near Cache

There was an error in the listing for a sample `session-cache-config.xml` file without a near cache that appeared in "Session Cache Configuration File Without a Near Cache" appendix of *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web*. See the new listing in the "Session Cache Configuration File Without a Near Cache" appendix.

2.13 New Performance Report for Coherence*Web in WebLogic Server Environments

Available since Coherence 12.1.2.0.3. The Weblogic Web Report (`report-web-weblogic.xml`) provides information on Coherence*Web activity when it is being used in WebLogic Server environments. This report provides the same information as provided by the Web Report (see "Web Report" in Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web), with additional columns for the WebLogic Server name and domain name. The report is a tab-delimited file that is prefixed with the date and hour in `YYYYMMDDHH` format and appended with `-web-weblogic.txt`. For example `2009013102-web-weblogic.txt` would be created on January 1, 2009 at 2:00 am.

Table 2–1 Contents of the WebLogic Web Report

Column	Data Type	Description
`Application`	String	The application name.
`Batch Counter`	long	A sequential counter to help integrate information between related files. This value does reset when the Reporter restarts and is not consistent across nodes. However, it is helpful when trying to integrate files.
`Current Overflow Updates`	long	The number of overflow updates since the last time the report was created.
`Current Session Updates`	long	The number of session updates since the last time the report was created.
`DomainName`	String	The name of the WebLogic Server domain in which Coherence*Web is running.
`LocalAttributeCount`	long	The attribute count on the node.
`LocalSessionCount`	long	The session count on the node.
`Node Id`	integer	The node identifier.
`OverflowAvgSize`	float	The average size for attribute overflows.
`OverflowMaxSize`	long	The maximum size for an attribute overflow.

Table 2–1 (Cont.) Contents of the WebLogic Web Report

Column	Data Type	Description
`OverflowUpdates`	long	The total number of attribute overflow updates since the last time statistics were reset.
`Report Time`	Date	The system time when the report was created.
`ServerName`	String	The name of the WebLogic Server on which Coherence*Web is running.
`SessionAverageLifetime`	float	The average number of seconds a session is active.
`SessionAverageSize`	float	The average size for a session.
`SessionMaxSize`	long	The maximum size for a session.
`SessionMinSize`	long	The minimum size for a session.
`SessionStickyCount`	long	The number of sticky sessions on the node.
`SessionUpdateCount`	long	The number of session updates since the last time statistics were reset.

3

Upgrading to Coherence 12*c* (12.1.2)

This chapter provides instructions for upgrading applications that use Coherence, Coherence*Web, ActiveCache, and TopLink Grid to 12*c* (12.1.2).

3.1 Upgrading Applications Using Coherence and Coherence*Web on WebLogic Server

Follow these instructions for upgrading applications running on WebLogic Server that use Coherence and Coherence*Web.

1. In an existing WebLogic Server domain:
   - Stop and undeploy the applications that use Coherence*Web.
   - Undeploy the `coherence.jar` and `coherence-web-spi.war` files if they are deployed.
2. Follow the steps to upgrade WebLogic Server and its domains to WebLogic Server 12*c* (12.1.2). For information on upgrading WebLogic Server, see *Oracle Fusion Middleware Upgrading Oracle WebLogic Server*.
3. Modify your applications to remove all references to the `coherence.jar` file:
   - In the `weblogic.xml` file, remove the `<library-ref>` element that refers to the `coherence-web-spi` file.
   - In the `META-INF/MANIFEST.MF` file, remove the following lines that identify Coherence as an extension:

     ```
     Extension-List: coherence
     coherence-Extension-Name: coherence
     ```

   - Remove any explicit references to the `coherence.jar` file in the classpath.
4. Modify your applications to use the required settings for Coherence 12*c* (12.1.2):

- If you used the default `session-cache-config.xml` file in your Coherence release 3.7.1.*x* application, note that the name has been changed to `default-session-cache-config.xml` in 12*c* (12.1.2).

  For example, if you used this context parameter value in Coherence release 3.7.1.*x* application:

  ```
  tangosol.coherence.cacheconfig=session-cache-config.xml
  ```

  change it to `default-session-cache-config.xml` in 12*c* (12.1.2):

  ```
  tangosol.coherence.cacheconfig=default-session-cache-config.xml
  ```

  You should not have to change the session cache file name. If you created a custom `session-cache-config.xml`, you should be able to leave the file name as it is.

- If your application is in an EAR file, then the packaging for the custom `session-cache-config` file has changed in 12*c* (12.1.2). For 12*c* (12.1.2) packaging instructions, see "Using a Custom Session Cache Configuration File" in *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web*.

5. Redeploy your applications on WebLogic Server.

3.2 Upgrading Applications Using TopLink Grid on WebLogic Server

The following sections describe how to upgrade applications that run in application-scoped, EAR-scoped, and WAR-scoped clusters. Before you upgrade the applications, you must first upgrade the Coherence-side configuration.

- Upgrade the Coherence-side Configuration
- Upgrade Application Server-Scoped Coherence Clusters
- Upgrade EAR-Scoped Coherence Clusters
- Upgrade WAR-Scoped Coherence Clusters

3.2.1 Upgrade the Coherence-side Configuration

Follow these steps to upgrade the Coherence-side configuration. The upgrades must be made to any pre-12*c* (12.1.2) Coherence cache server startup scripts that you are using that reference the pre-12*c* (12.1.2) Coherence JARs. The scripts that are shipped with Coherence do not need to be changed.

1. Shutdown the Coherence servers.
2. Modify the Coherence cache server startup script to change classpath (`${ORACLE_HOME}` in WebLogic Server 12*c* (12.1.2)):

   —Replace `coherence.jar` with `${ORACLE_HOME}/coherence/lib/coherence.jar`.

   —Replace `javax.persistence_1.*.jar` with `${ORACLE_HOME}/oracle_common/modules/javax.persistence_2.0.0.0_2-0.jar`.

   —Replace `com.oracle.toplinkgrid_1.1.0.0_11-1-1-6-0.jar` with `${ORACLE_HOME}/oracle_common/modules/oracle.toplink_12.1.2/toplink-grid.jar`.

   —Replace `org.eclipse.persistence_1.2.0.0_2-3.jar` with `${ORACLE_HOME}/oracle_common/modules/oracle.toplink_12.1.2/eclipselink.jar`.
3. Restart the Coherence servers.

3.2.2 Upgrade Application Server-Scoped Coherence Clusters

If you have application server-scoped applications running in your Coherence clusters, follow these steps to upgrade to Coherence 12*c* (12.1.2).

1. Follow the steps to upgrade WebLogic Server and its domains to WebLogic Server 12*c* (12.1.2). For information on upgrading WebLogic Server and its domains, see *Oracle Fusion Middleware Upgrading Oracle WebLogic Server*.
2. Follow these steps to upgrade the WebLogic Server-side configuration before you start any managed servers:

   Note: In the WebLogic Server 12*c* (12.1.2), the `toplink-grid.jar` and `coherence.jar` files (as well as the `coherence-web.jar` and `eclipselink.jar` files) are automatically included in WebLogic Server system classpath.

   a. Remove any explicit references to the `toplink-grid.jar` file from the WebLogic Server classpath. Examples of old references include the following:

   ```
   com.oracle.toplinkgrid_1.1.0.0_11-1-1-6-0.jar
   javax.persistence_1.*.jar
   org.eclipse.persistence_1.2.0.0_2-3.jar
   ```

   b. Remove any explicit references to the `coherence.jar` file from the WebLogic Server classpath. Examples of old references include the following:

   ```
   coherence.jar
   coherence-web.jar
   coherence-web-spi.war
   ```

3. Restart the WebLogic Servers.
4. Redeploy your applications.

3.2.3 Upgrade EAR-Scoped Coherence Clusters

If you have EAR-scoped TopLink applications running in your Coherence clusters, then you can upgrade your applications to use Coherence 12*c* (12.1.2) in any of the following ways:

- Upgrade your applications to use GAR file format. See "Using GAR File Format".
- Update the shared libraries. See "Updating Shared Libraries".
- Update the `toplink-grid.jar` and `coherence.jar` files packaged in the EAR file. See "Packaging the toplink-grid.jar and coherence.jar Files in the EAR File".

Note: The following procedures assume that you have upgraded WebLogic Server and its domains to WebLogic Server 12*c* (12.1.2). For information on upgrading WebLogic Server, see *Oracle Fusion Middleware Upgrading Oracle WebLogic Server*.

Using GAR File Format

The following steps assume that you are upgrading your application to use GAR file format. This is the approach recommended by Oracle. For more information on packaging applications in GAR file format, see "Creating Coherence Applications for WebLogic Server" in *Developing Oracle Coherence Applications for Oracle WebLogic Server*.

1. Remove any references to Coherence and TopLink Grid shared libraries from the application and WebLogic Server classpaths.

   Note: You can remove these references because in the WebLogic Server 12*c* (12.1.2), the `toplink-grid.jar` and `coherence.jar` files (as well as the `coherence-web.jar` and `eclipselink.jar` files) are automatically included in WebLogic Server system classpath.

   Examples of references to Coherence include:

   ```
   coherence.jar
   coherence-web.jar
   coherence-web-spi.war
   ```

   Examples of references to TopLink Grid include:

   ```
   com.oracle.toplinkgrid_1.1.0.0_11-1-1-6-0.jar
   javax.persistence_1.*.jar
   org.eclipse.persistence_1.2.0.0_2-3.jar
   ```

2. Remove references to the `coherence.jar` and `toplink-grid.jar` from the `weblogic-application.xml` file.
3. Remove the `<coherence-cluster-ref>` element, which configures Coherence cluster properties, from the `weblogic-application.xml` file.
4. If you have configured a filtering classloader, then remove it. This feature is defined in the `<prefer-application-packages>` element in the `weblogic-application.xml` file. Remove any `<prefer-application-packages>` elements that reference the `coherence.jar` and `toplink-grid.jar` files.
5. If you have packaged the `toplink-grid.jar` and `coherence.jar` files within the EAR file, then remove them from the EAR file.
6. Restart the WebLogic Server and the Coherence servers.
7. Redeploy your applications.

Updating Shared Libraries

This approach assumes that you are referencing the `toplink-grid.jar` and `coherence.jar` files as shared libraries and do not want to change the packaging of your applications. In this case, you must redeploy the 12*c* (12.1.2) versions of the `toplink-grid.jar` and `coherence.jar` files as shared libraries.

Packaging the toplink-grid.jar and coherence.jar Files in the EAR File

This approach assumes that you have packaged your `toplink-grid.jar` and `coherence.jar` files in the EAR file. In this case, replace the `toplink-grid.jar` and `coherence.jar` files with the12*c* (12.1.2) version of these files.

3.2.4 Upgrade WAR-Scoped Coherence Clusters

If you have WAR-scoped TopLink applications running in your Coherence clusters, then you can upgrade your applications to use Coherence 12*c* (12.1.2) in any of the following ways:

- Update the shared libraries. See "Updating Shared Libraries".
- Update the `toplink-grid.jar` and `coherence.jar` files packaged in the WAR file.

Note: The following procedures assume that you have upgraded WebLogic Server and its domains to WebLogic Server 12*c* (12.1.2). For information on upgrading WebLogic Server, see *Oracle Fusion Middleware Upgrading Oracle WebLogic Server*.

Updating Shared Libraries

This approach assumes that you are referencing the `toplink-grid.jar` and `coherence.jar` files as shared libraries and do not want to change the packaging of your applications. In this case, all you need to do is to update the shared library references to use the 12*c* (12.1.2) version of the `toplink-grid.jar` and `coherence.jar` files.

Packaging the toplink-grid.jar and coherence.jar Files in the WAR File

This approach assumes that you have packaged your `toplink-grid.jar` and `coherence.jar` files in the EAR file. In this case, replace the `toplink-grid.jar` and `coherence.jar` files with the12*c* (12.1.2) version of these files.

3.3 Upgrading Coherence*Extend

For all Extend client customers (Java, C++, and .NET), you must upgrade the cluster side before upgrading the Coherence*Extend clients. This is in compliance with the Coherence client and proxy upgrade policy. For more information on compatibility between the current release of Coherence*Extend and earlier versions, see "Compatibility Between Coherence*Extend Versions" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.

3.4 Upgrading Coherence*Web

The following sections describe upgrade considerations for Coherence*Web.

- Coherence*Web SPI Reserved for Older Versions of WebLogic
- ActiveCache (active-cache.jar) Replaced with Managed Coherence Servers
- New Session Cache Configuration File

3.4.1 Coherence*Web SPI Reserved for Older Versions of WebLogic

The `coherence-web-spi.war` file, which was included in previous releases of Coherence*Web, is deprecated in 12*c* (12.1.2). The Coherence 12*c* (12.1.2) distribution includes a `coherence-web-spi.war` file in the `coherence\lib` directory, but this file is provided only for backward compatibility with older versions of WebLogic Server. If you are using WebLogic Server 12*c* (12.1.2) or later, you should not have to work with or reference this file. If you attempt to deploy the `coherence-web-spi.war` file to WebLogic Server 12*c* (12.1.2) or later, it will be ignored.

3.4.2 ActiveCache (active-cache.jar) Replaced with Managed Coherence Servers

ActiveCache (`active-cache.jar`), the collection of WebLogic Server features which allow deployed applications to easily use Coherence data caches and seamlessly incorporate Coherence*Web for session management, has been deprecated in the 12*c* (12.1.2) release. ActiveCache is still supported in the 12*c* (12.1.2), primarily to support upgrades of applications developed on older releases. For more information on ActiveCache, see *Oracle Fusion Middleware Using ActiveCache*.

Users must migrate to Managed Coherence Servers when developing new WebLogic Server/Coherence applications for the current release. For more information on Managed Coherence Servers, see *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*.

3.4.3 New Session Cache Configuration File

In previous releases, Coherence cache configurations and services used by Coherence*Web SPI were defined in the `session-cache-config.xml` file, As of the 12*c* (12.1.2), Coherence cache configurations and services used by Coherence*Web are defined in the `default-session-cache-config.xml` file, which can be found in the `coherence-web.jar` file. The default cache and services configuration defined in the `default-session-cache-config.xml` file should satisfy most Web applications.

You can create your own custom session cache configuration by packaging a file named `session-cache-config.xml` in your Web application. For more information see, "Using a Custom Session Cache Configuration File" in *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence*Web.*

3.5 Upgrading ActiveCache Applications on WebLogic Server

The 11g Release 1 (10.3.6) version of ActiveCache is documented in *Oracle Fusion Middleware Using ActiveCache*. This version of ActiveCache will work with WebLogic Server and Coherence 12*c* (12.1.2) but some of the documented steps are no longer required.

> **Note:** ActiveCache is deprecated in the 12*c* (12.1.2) release. Users must migrate to Managed Coherence Servers. For more information on Managed Coherence Servers, see *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*.

- "Choose the ActiveCache Deployment Topology" describes the several different combinations of application and data tiers, or *cluster topologies*, in which ActiveCache can be deployed. In upgrading applications using ActiveCache, you should not use the Out-of-Process topology except for backward compatibility. In the current release, WebLogic Out-of-Process topology is the preferred approach. Using managed Coherence servers makes the WebLogic Out-of-Process topology easier to configure.
- "Locate the Cache Configuration File" describes the location where you place the cache configuration file. The location where you store the cache configuration file determines the cache scope; that is, the visibility of the caches to deployed applications. The approaches described in this section will work, but putting the cache configuration in the system classpath is a bad practice unless there is only one and will only ever be one application using Coherence in the server.

  Oracle recommends that you use a GAR file when you package your application. The cache configuration file is packaged in the GAR file. For more information on the GAR file and its packaging structure, see *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*.
- "Configuring Application-Server Scoped Coherence Clusters" describes a configuration such that all deployed applications on WebLogic Server instances that are directly accessing Coherence caches become part of one Coherence cluster. In the procedure, **do not** perform Step 1: do not put the `coherence.jar` and `active-cache.jar` files in the system classpath. The `active-cache.jar` file uses

the classpath in the MANIFEST file to add the Coherence integration module to the classpath. In release 12*c* (12.1.2), the Coherence integration module will always be in the server classpath, in addition to the coherence.jar file.

- "Configuring EAR-Scoped Coherence Clusters" describes a configuration such that all deployed applications within each EAR become part of one Coherence cluster. Caches will be visible to all modules in the EAR. The procedure described in this section will not work as described. Because coherence.jar is already in the system classpath, you must follow the steps documented in the section "To Define a Filtering Classloader for Application-Scoped Coherence Clusters".

  The only reason to use the EAR-scoped approach is to isolate your application from other Coherence applications. That use case is better handled by the application isolation provided by a GAR file, or by using the scope element in the cache configuration file. Another use case is to use a different version of coherence.jar than is in the system classpath but using a different version should be discouraged.

- "Configuring WAR-Scoped Clusters" describes a configuration such that each deployed Web application becomes its own Coherence cluster. Caches will be visible to the individual modules only. In the procedure, **do not** perform Steps 1 and 2. The coherence.jar and active-cache.jar should not be deployed as shared libraries nor should they appear in the MANIFEST file. You can perform Step 3 to reference the Coherence cluster system resource, but making the managed server a member of the Coherence cluster is the preferred approach.

- "Example 3-10 tangosol-coherence-override.xml" displays a custom cache configuration file that contains a logging configuration. The logging configuration is not needed because of log integration in release 12*c* (12.1.2).

- "Start a Cache Server" section describes several different ways of starting the cache server. The Out-of-Process topology should be replaced with managed Coherence servers. The procedure described in "Starting a Cache Server Using Node Manager" should be performed by using managed Coherence servers, instead of using the external cache server managed by WebLogic Server.

3.6 Replacements for Deprecated Features

The following sections describe replacements for features that have been deprecated in Coherence 12*c* (12.1.2).

3.6.1 Replacement for Deprecated packet-pool and message-pool Elements

The packet-pool and message-pool elements are deprecated. In Coherence 12*c* (12.1.2), the API will now take care of sizing. To upgrade, remove the elements from any configuration files.

3.6.2 Replacement for the Deprecated LH File Manager

The LH store manager is deprecated as of Coherence 12*c* (12.1.2) release. Use Berkeley DB for similar functionality.

3.6.3 Replacement for the Deprecated NamedCache Lock APIs

The `NamedCache lock` APIs are deprecated. Use the locking support that is provided by the entry processor API instead (`EntryProcessor` for Java and C++, `IEntryProcessor` for .NET).

3.6.4 Replacement for the Deprecated XmlConfigurable Interface

The `com.tangosol.run.xml.XmlConfigurable` interface has been deprecated in the Coherence 12*c* (12.1.2) release. Coherence used this interface to inject XML parameters into instances of custom classes.

Note: For information on the interfaces, classes, and methods affected by the deprecation of the `XmlConfigurable` interface, see "XmlConfigurable Interface" on page 1-31.

In the Coherence 12*c* (12.1.2) release, you can initialize parameters by writing XML which nests `<instance>` and `<class-scheme>` (or any other custom namespace) inside of `<param-value>` elements.

For example, given the following Java code:

```
public class MyClass
  {
  public MyClass(String s, OtherClass o, int i) { ... }
  }

public class OtherClass
  {
  public OtherClass(String s) { ... }
  }
```

You can initialize the `MyClass` and `OtherClass` classes by writing the following XML. In the XML, the `MyClass` class is initialized with the string `Hello World` and the integer `42`. The instance of the `OtherClass` class which appears in the `MyClass` class, is initialized with the string `Goodbye World`.

```
<instance>
  <class-name>MyClass</class-name>
    <init-params>
      <init-param>
        <param-value>Hello World</param-value>
      </init-param>
      <init-param>
        <param-value>
          <instance>
            <class-name>OtherClass</class-name>
              <init-params>
                <init-param>
                  <param-value>Goodbye World</param-value>
                </init-param>
              </init-params>
          </instance>
        </param-value>
      </init-param>
      <init-param>
        <param-value>42</param-value>
      </init-param>
    </init-params>
```

```
</instance>
```

3.7 Other Upgrade Issues

The following sections describe issues that you might need to consider when upgrading to Coherence 12*c* (12.1.2).

- New DistributedCache Default for Exalogic Environments
- Connecting from Remote RMI Clients
- Key Associations on the Coherence*Extend Client
- Changes to Invalidation Strategy for Near Caches
- Using BACKUP CACHE Statement to Write a Serialized Representation of a Cache
- New Cache Configuration Element: resource-config
- JVM Upgrades
- Changes to Invocable API Behavior

3.7.1 New DistributedCache Default for Exalogic Environments

`DistributedCache` now defaults to the Exabus optimized transport in Exalogic environments. For more information, see the description of the `<reliable-transport>` element in "DistributedCache Service Parameters" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

3.7.2 Connecting from Remote RMI Clients

When connecting from a remote RMI client (different physical computer), add the `java.rmi.server.hostname` RMI system property to the script with the value set to the cluster member's IP address. The address ensures that the RMI stubs that are sent to the client contain the correct server address. For more information, see "Allowing Remote Access to Oracle Coherence MBeans" in *Oracle Fusion Middleware Managing Oracle Coherence*.

3.7.3 Key Associations on the Coherence*Extend Client

Key association is now processed on the extend client by default. Existing client implementations (including Java clients) that rely on key association on the cluster must set the `defer-key-association-check` parameter in order to force the processing of key classes on the cluster.

To force key association processing to be done on the cluster side instead of by the extend client, set the `<defer-key-association-check>` element, within a `<remote-cache-scheme>` element, in the client-side cache configuration to true. For example:

```
<remote-cache-scheme>
   ...
   <defer-key-association-check>true</defer-key-association-check>
</remote-cache-scheme>
```

For more information, see "Deferring the Key Association Check" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.

3.7.4 Changes to Invalidation Strategy for Near Caches

The default near cache invalidation strategy `auto` has changed to ensure that reduced network traffic is prioritized over performance. Set the invalidation strategy to `all` for pre-12*c* (12.1.2) default behavior. For more information, see "Near Cache Invalidation Strategies" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

3.7.5 Using BACKUP CACHE Statement to Write a Serialized Representation of a Cache

Use the `BACKUP CACHE` statement to write a serialized representation of the given cache to a file represented by the given filename. The backup is not forward or backward compatible between 3.x and 12.1.x. See "Writing a Serialized Representation of a Cache to a File" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

3.7.6 New Cache Configuration Element: resource-config

The `resource-config` element contains the configuration information for a class that extends the `com.sun.jersey.api.core.ResourceConfig` class. The instance is used by the HTTP acceptor to load resource and provider classes for the Coherence REST application that is mapped to the specified context path. Multiple resource configuration classes can be configured and mapped to different context paths. For more information, see "Deploying with the Embedded HTTP Server" in *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*.

3.7.7 JVM Upgrades

Coherence 12*c* (12.1.2) is supported on Oracle HotSpot JVM 1.7 and other 1.7-level JVMs. There is no JRockit 1.7, so JRockit is not longer relevant. Extend clients are supported on JVM 1.6+, including JRockit. For more information, see "JVM Recommendations" in *Oracle Fusion Middleware Administering Oracle Coherence*.

3.7.8 Changes to Invocable API Behavior

Applications that use the Invocable API may receive an error when upgrading from Coherence 3.7.1 to Coherence 12.*x* due to a change in serialization requirements. In Coherence 3.7.1, if an Invocable is sent to a number of nodes including itself, then there is a chance that it will begin local execution before having been serialized for transmission to the remote members. If the Invocable updates non-transient state, this state will be leaked to the other nodes as part of the delayed serialization.

In Coherence 12.*x*, applications that use the Invocable API on local members must make sure that their classes (such as entry processors and aggregators) are serializable.