Hazelcast Documentation

version Not-Needed

Jun 14, 2016

In-Memory Data Grid - Hazelcast | Documentation: version Not-Needed

Publication date Jun 14, 2016

Contents

Chapter 1

3.7-EA

This section lists the new features, enhancements and fixed issues for 3.7 release.

1.1 New Features

The following are the new features introduced with 3.7 release.

- **Custom Eviction Policies**: You can implement and use your own eviction policy. Please refer to the Custom Eviction Policy section.
- **Discovery SPI Implementation for Microsoft Azure Services**: Hazelcast members can be discovered within your Azure resource group. You can add this implementation as a plugin to your projects. Please refer to Hazelcast-Azure plugin.
- **Hazelcast CLI with Scripting**: A command line interface that supports scripting. You can automate cluster operations such as start, stop, and force start using shell scripting. Please refer to Hazelcast CLI plugin page.
- **Hazelcast for OpenShift**: Hazelcast members on OpenShift can discover each other.
- **Alignment of WAN Replication Clusters:** This feature provides a mechanism to align or realign distributed objects in the clusters connected through WAN. WAN replication mirrors changes from a map or cache in one cluster to another. It was maintaining the alignment but was not attaining. Now when the receiving cluster is empty and the sending cluster is not, objects are aligned. And if changes have been dropped for any reason, realignment is attained.

1.2 Enhancements

NOTE: *You will find that some of the previously existing libraries do not exist in your Hazelcast 3.7-EA download package. Starting with this release, these libraries will be offered as separate plugins. They will have their own lifecycles and they will not cause any interdependency issues such as delays in releases. Please refer to the Plugins page for more information on each plugin. The libraries which are not included the Hazelcast package are as follows:*

- *hazelcast-cloud.jar*
- *hazelcast-hibernate3.jar*
- *hazelcast-hibernate4.jar*
- *hazelcast-jca.jar*
- *hazelcast-jca-rar.jar*
- *hazelcast-jclouds.jar*
- *hazelcast-spring.jar*

- *hazelcast-wm.jar*

The following are the the enhancements introduced with 3.7 release.

- **Near Cache (JCache) Notification Enhancements:** You can disable the near cache notifications for each entry and enable/disable notifications of full-flush events. Please refer to the ICache Configuration section and see definition of the new configuration element `disable-per-entry-invalidation-events`.
- **Migration Algorithm Enhancements:** With these improvements the possibility of a data loss due to a member crash while the partitions are being migrated is eliminated.
- **WAN Replication Integrated with Solace**: This integration is achieved through a new JMS endpoint.
- **Cloud Discovery SPI Plugin for Multicast:** You can use multicast discovery for your Hazelcast platform with client/server topology. Only Hazelcast Java client is supported for this release.
- **IMap Eviction Sync with JCache:** Hazelcast Map uses now the Hazelcast JCache's eviction algorithm when evicting map entries.
- **Docker Image Enhancements:** Hazelcast Docker image is able to integrate with the Service Discovery products such as Zookeeper, Consul, Etcd, and Eureka.
- **Phone Home Enhancements:** Performed to collect new phone home data to learn more about the environment on which Hazelcast runs.
- **`IMap.putAll()` Performance Enhancements:** The performance of `putAll` operation is improved.
- **Hazelcast Instance and JCache Integration Enhancements:** A direct relation between a Hazelcast instance and JCache is established with this enhancement. You can retrieve and access caches via the method `getCache(String name)` over `HazelcastInstance` API. Please refer to the JCache - Hazelcast Instance Integration section.
- **Indexing with Predicates for Entry Processors:** Entry Processor can use indexing when a predicate is used to filter entries that need to be processed.
- **Partition Grouping Enhancements:** You can define partition groups, in the same way as the IP address network configuration with wildcard support. You can also configure zone-aware partition groups for your Hazelcast clusters on AWS.
- **Prevention of Blocking Reads in Transactions:** Now the read operations are blocked only during committing the transactions.
- **Jetty and Tomcat Based Web Session Replications:** These features have been made open source. You can reach them at Tomcat Session Manager and Jetty Session Manager GitHub repos.

The following are the other improvements performed to solve the enhancement issues opened by the Hazelcast customers/team.

- Collection querying in Portables. This allows querying a collection like `car.wheels[0].pressure` or `car.wheels[any].pressure` in the Portable data format. It also allows using a `ValueExtractor` in the Portable data format. [8132]
- Performance improvements for the invocation system. [8009]
- The performance log should have an option to log to the 'shared' logger instead of its own file. It would be useful for environments where Hazelcast cannot create or get a performance log file. [7973]
- The path for performance logs, which is currently fixed to the user's working directory should be configurable. [7968]
- Hazelcast `IAtomicLong` data structure provides synchronous methods only; async versions already exist and are available to Hazelcast members via `AsyncAtomicLong` interface. Lack of public a async API for IAtomicLong is impeding Hazelcast integrations. [7957]
- It would be better to have a way where near cache notifications for each entry are disabled, but an ability to enable/disable notifications of full-flush events (clear, etc.). [7580]
- Hazelcast should support Transaction API of Spring. [7469], [611]

- For Hazelcast Topic, even the event service's pool size is set to a number larger than one, all of the messages are consumed by only one Hazelcast event thread. The use case includes a single Hazelcast member, both producer and consumer being singletons within the member, and message rate of more than 1000 per second. [7443]
- Partition strategy should be able to be specified not only in the Hazelcast configuration, but also within the Spring context. In addition, an implementing instance should be specified besides the class which implements the partition strategy. [7363]
- Async put operations should be reflected at near cache as soon as the method `future.get()` returns. In the case of async put operations and `LocalUpdatePolicy` being `CACHE` at the client side, entries are put to the near cache asynchronously from another task when the response is received. But with this way, when `future.get()` returns, entry might not be inside the near cache (but it will be there eventually). [7155]
- For `ICache.iterator()`, `fetchSize` is not configurable. [7041]
- Unit tests should have a default timeout. [6978]
- Outgoing ports on Hazelcast clients should be configurable. [6845]
- The method `IMap.set` does not have a corresponding async version, unlike `put` and `putAsync`. The method `putAsync` is not entirely suitable as an async set, since put returns the previous value mapped to the key, and triggers EntryListeners which may not be desirable. IMap should expose a dedicated `setAsync` to fulfill the contract for set and have the means to do so asynchronously. [6726]
- Javadoc for `EntryProcessor.java` should be enhanced by adding notes related to its thread safety. [6593]
- Custom SPI services should be more Spring-friendly. [6567]
- The "spring-aware" should be enabled programmatically too. [6514]
- It would be nice if the type parameters of `Predicate` were inherited by the `IndexAwarePredicate`. [1686]
- The class `MigrationEndpoint` should be a part of Hazelcast SPI package. [1427]
- When a task is submitted to all members, and an `executeOnEntries` is invoked in the call with a predicate that is based on an index, then the index is ignored and a "full scan" of the "local" members is performed. [1156]
- Inconsistency between the declarative and programmatic configuration of network elements should be solved. [945]

1.3 Fixes

The following are the issues solved for Hazelcast 3.7 and 3.7.x releases.

3.7-EA Fixes

This section lists the fixed issues for 3.7-EA release.

- The method `ICache::destroy` should remove the cache itself from the owner `CacheManager` because, otherwise, it causes memory leaks due to the cache proxies which are dead but deemed as working, in `AbstractHazelcastCacheManager::caches`. [8186]
- Partition promotion is skipped when a node is terminated during the commit. [8174]
- The method `IAtomicReference:alter` does not persist the changes. When a reference is tried to be altered, no alteration happens. [8149]
- Cache should not expire entities when `Duration` value is 0. [8148]
- Deserialization of dynamic proxy instances ignores the configured class loader. [8033]

- The attribute "binary" is missing in the MultiMap configuration within Spring context. It does not exist in Hazelcast configuration schema either. [8000]
- If you setup an interceptor to change the data being inserted, the entry listeners still fire with the old value. [7991]
- Unlike the `InvocationFuture` at the server side, `ClientInvocationFuture` immediately propagates `InterruptedException` if the calling thread gets interrupted. This can be a problem when both caller and callee need to agree on whether the operation has executed or not. [7963]
- Hazelcast 3.2.6 uses too much CPU when it is idle. [7943]
- Old version of Portable object from a map cannot be read if new `UTF_ARRAY` type field is added. [7926]
- Isolated thread pool for priority generic operations. [7857]
- There is an issue when detecting JCache in the classpath. The exception `NoClassDefFound` is thrown when upgrading to a newer Hazelcast version. [7810]
- Better separators should be used in the exceptions for a clearer read between local and remote stacktraces. [7744]
- Under the section "Operation Threading" of Hazelcast Reference Manual, it states that the default number of partition-aware operation threads is (2 x number of cores). However, when looking at the code and observing the actual number of threads created runtime, it seems like the default value is instead 1 x number of cores instead. [7741]
- The method `IMap.executeOnKeys()` does not support the empty set (it throws a misleading `NullPointerException`), and is inconsistent with the method `getAll()`. [7631]
- Replicated map updates take a very long time. The problematic method is `putAll()`. The replication logic in this method checks whether the data owners are in sync with the replicas. If they are not, this logic syncs them every 30 seconds. This means, when the updates are not replicated to callers, it takes up to 30 seconds to make all the members synchronized. This period should be configurable. [7617]
- `ScheduledExecutorServiceDelegate` violates contract of `ScheduledExecutorService`. It wraps tasks in `ScheduledTaskRunner` which delegates to a different executor. As a consequence, a task can be executed concurrently and this is a violation of a contract of `ScheduledExecutorService`. [7611]
- If `javax.cache.CacheManager` is created with the default settings, the underlying `HazelcastInstance` is not shutdown when the method `close` is called on the `CacheManager`. [7606]
- The method `containsKey()` of `TransactionalMap` is blocked when the key was previously locked by the method `getForUpdate()`. [7588]
- There is an inconsistent behavior when removing from `TransactionalMap` while the key is locked on IMap. In order to avoid trying to remove an entry that may have already been removed in another uncommitted transaction, `IMap.tryLock` is used before performing `TransactionalMap.remove`. This works as expected if the operations occur on a member. But a `TransactionException` is thrown when it occurs on a client when using XA Transaction. [7587]
- Hazelcast instance should be exposed through `com.hazelcast.spring.cache.HazelcastCacheManager`. [7571]
- Instance name should not be overridden while creating cache manager from the specified configuration file. Also, it would be better to specify instance name via (]`HazelcastCachingProvider.HAZELCAST_INSTANCE_NAME` property when instance configuration is taken from the specified configuration file via `HazelcastCachingProvider.HAZELC`
- The `addInterceptor()` method in `com.hazelcast.map.impl.MapContainer()` is not thread safe. For example, if two concurrent attempts are made to inject the same interceptor, these will be different interceptor objects with the same ID. In this case, the call to `interceptorMap.put(id, interceptor)` will increase the map size by one, but the call to `interceptors.add(interceptor)` will increase the list size by two. [7520]
- There are unused elements for Management Center configuration: `cluster-id` and `security-token`. [7446]

- For clients, `InitialMembershipListener.init` is called after `MembershipListener.memberAdded`. This contradicts the content in the Reference Manual. [7430]
- DiscoveryService's `start` and `destroy` methods should be called during the start and shutdown of client when Discovery SPI is enabled. [7347]
- Return cache config as response even though found and created cache config could not put into cache configs inside cache service. [7208]
- In Hazelcast Management Center shutting down a node seems to prevent a node from restarting. [7101]
- `MapStoreConfig` does not override `hashCode` and `equals` methods. Implementation for these two methods should be added. [7035]
- Data is lost when the member dies during repartitioning. [6628]
- Some of the map statistics, such as cost and last access time, are calculated by the traversing map entries. Therefore the calculation time exceeds the time interval reserved for management center state sending thread when entry count is too high. [6442], [5905]
- `InvocationFuture`'s asynchronous calls do not detect the lost operations. [6250]
- The invocation mechanism for blocking operations relies on a periodic timeout so that the operation gets retried. To prevent the calling thread (the thread for `future.get`) from waiting indefinitely, it will periodically ask the `isstillrunning` service if the operation is lost. [6248]
- Under some circumstances Hazelcast is getting a corrupt value for `IAtomicLong`s when a member leaves the cluster. [6074]
- When the client disconnects normally, the server logs an info and a warning message containing the text `java.io.EOFException`. [6035]
- Some operating systems (such as HPUX or Solaris) and hardware platforms have constraints about the aligned memory operations. In these architectures memory operations must be byte-by-byte as implemented in `DirectByteBuffer`. [5532]
- Data is lost when a member crashes or is killed during the repartitioning. [5444]
- Data is lost when a member is terminated. Related scenario is a follows [5388]:

  1. Start the first member, and let it populate a map with 100k entries.
  2. Start the second member, and let it start joining the cluster.
  3. Terminate the second member during the join operation.
  4. Observe that data is lost from the first member.

- As for now it is very complicated to listen a `getAsync` or `putAsync` result and to integrate it with completable futures or listenable futures. An `ICompletableFuture` should be returned since it is an interface which seems to extend JDK `future`s and is returned by an IMap. [5315]
- If multiple Hazelcast members attempt to remove values from a key of a multimap concurrently, and then the members are shut down, the multimap can remain in an inconsistent state with entries remaining after all have been removed. [5220]
- `ClassNotFoundException` is thrown when trying to get an entry from a `TransactionalMap`. [4969]
- Profiling a Hazelcast application reveals a thread contention in `SpringManagedContext` on `java.lang.Class.getAnnotati`
  And this calls a synchronized method called `initAnnotationsIfNecessary()`. [4506]
- Hazelcast IMap statistics show negative values. After heavy usage of the cache, the number of misses starts showing up negative. [4022]
- When there is a map with write-behind mode and a map store is configured (eviction is not needed); when the method `flush` is called in the IMap, the map store's `store` method can be called concurrently for the same key, namely for those keys which are in the write-behind queue and then forcibly stored by the flush. This is because the flush operation storing all entries in the write-behind queue seems to be executed in the operation thread, while the periodic processing of the write-behind queue is done by an executor service defined in the `WriteBehindQueueManager`. [3338]

Chapter 2

3.6.3

The following are the fixed issues for 3.6.3 release.

- `MapStatisticsAwareService` should obtain the map names from proxy registry. [8209]
- Cache should not expire the entities when `Duration` value is set to `0`. [8148], [8206]
- Collection querying in Portables. [8172]
- Type extraction by the method `extractedMultiResult` should be fixed when the extraction result is null. [8134]
- Expiration time for the map entries should be calculated based on their latest update times. [8111], [8113]
- Support for the usage of IAM role's defaults should be added. [8100]
- Binary property for MultiMap should be added to XmlConfigBuilder. [8094]
- EC2 auto-discovery in China throws an exception due to a bug in host header handling mechanism. [8073]
- The method `Cluster.shutdown()` ignores lifecycle listeners. [8070]
- Resource adapters should not cast to implementation when using a container to get a connection. [8019]
- Unnecessary deserialization step during the execution of put operations should be removed. [8018]
- A URL text with spaces should not be transformed into a URI. [8016]
- Unnecessary deserialization step in the replicated map data structure should be removed. [8014]
- When using `InstanceOfPredicate`, the method `toObject` at the server side causes `NullPointerException`, since the class loader is not set and it is null. Default class loader should be used if it is not set by the configuration to avoid the exception. [7977]
- There may be cases when the server may return a list of entries larger than the requested page size. In this case the client should not put any anchor into the list that is on a page greater than the requested page. [7976]
- When a remote invocation waits indefinitely on the internal executor, a deadlock may occur since its response would use the same executor and since the invocation monitor runs on the same executor. [7944]
- Already published messages should be retrieved after `StaleSequenceException` occurs when `ReliableMessageListener` is registered at the client side. [7928]
- The case of response being equal to null in the class `ClientDelegatingFuture` should be handled. [7901]
- Detection of JCache should be improved. Currently Hazelcast checks if the class `javax.cache.Caching` is available and, if so, Hazelcast creates the JCache service and configures the client protocol messages. However, some older snapshots of JCache JARs have the Caching class available, but other classes are missing. This causes `Class not found` exceptions when members are started. [7899]
- When connection gets an exception from the socket and the method `destroyConnection` is called before connection is authenticated, the client does not have an endpoint and it is not in the map of connections. Hence, the connection cannot be closed. [7866]
- Upon subsequent destroy and create operations of IMap, there can be more than one map container referenced by different record stores at the same time. Therefore, indexes can be created in an unexpected map container and this can lead to return less than expected number of results when IMap is queried. [7838]
- After the client is shut down, there should be no invocations left. The method `assert()` of `ClientInvocationServiceSupport` fails. [7836]

The following are the enhancements performed for 3.6.3 release.

- Phone home should include environment information, such as the operating system name, version and kernel architecture. [7970]
- Exceptions, which are thrown when the method `isMemberSafe()` is called, should go into finest level. When there are topology changes some exceptions are expected. The method `getFutureResult()` should log exceptions as `finest()`, not `warn()`. [7904]
- Protocol version should be 1 instead of 0. It was set as 0 for the response messages. [7900]

Chapter 3

3.6.2

The following are the fixed issues for 3.6.2 release.

- Upon subsequent destroy and creation of IMap, there is a possibility that there can be more than one map-containers referenced by different record-stores at the same time. Hence, indexes can be created in an unexpected map-container and this can lead to return less than expected number of results when IMap is queried. [7874]
- Couple of issues related to client should be fixed: (a) There should be no invocations left after client is shut down. (b) When the client-member connection gets an exception from the socket and the method `destroyConnection` is called before the connection is authenticated, the client does not have an endpoint and it is not in the connections map anymore. Connection cannot be closed because of this. (c) Authentication `future` waits infinitely. Because of (b), heartbeat does not work on that connection yet. [7867]
- In Hazelcast 3.6.1, the OSGi bundle install fails. [7772]
- Timeout happens at the `ClientMapBasicTest`. [7718]
- IMap does not send invalidations to the client's near caches after `putAll/loadAll` operations at member side. [7715]
- The test `ClientXAStressTest.testCommitConcurrently` fails due to an assertion error that reads as "the size of the map is not correct". [7709]
- The test `ClientTransactionalMapQuorumTest` fails since it timed out. [7693]
- Wildcards do not work with permissions. For example, when a queue permission is defined using wildcard, i.e. `<queue-permission name="secure.*" principal="admin">` and the queue `secure.orders` is created at the client, Hazelcast throws `AccessControlException`. [7616]
- The FREE_HEAP_PERCENTAGE eviction policy does not work as documented because the `EvictionChecker` is incorrectly calculating available memory percentage in `checkFreeHeapPercentageEviction`. [7599]
- `DiscoveryStrategy`'s `destroy()` method is not called on shutdown. [7545]
- The method `MapStore.storeAll()` is never called when the objects are updated with a delay. [7464]
- WebFilter(3.6-RC1): Session attributes are lost or overwritten with old values when using `deferred-write` by reading them. [7291]
- The method `HazelcastHttpSession.getAttributeNames()` returns entries that were only read but not written. [7033]
- Hazelcast 3.5.3 conflicts with FUSE 6.1. When Hazelcast is integrated into a system with JBoss Fuse 6.1, there appeared some warnings in the logs. [6821]
- An exception is thrown on the server when attempting to deserialize `HIBERNATE4_TYPE_HIBERNATE_CACHE_ENTRY` value. [6683]
- The test `IOBalancerMemoryLeakTest.testMemoryLeak` fails due to a socket exception that reads as "unexpected end of file from server". [6496]

The following are the enhancements performed for 3.6.2 release.

- Hazelcast Docker images are big in size; it would be better to clean them up and reduce their sizes. [7553]

- It would be better to provide an `enterprise-javadoc.jar` in the Hazelcast Enterprise bundle. This way IDEs can auto-import the Javadocs for the Enterprise APIs and provide context sensitive completion/help within the IDE. [7245]
- Management Center WAN Replication metrics take a long time to stabilize. The responsiveness of the Outbound records/sec. metric should be improved. It currently seems to under-report WAN replication performance for 10s of seconds before reaching the same figure as the client side reported metrics.[7244]

Chapter 4

3.6.1

The following are the fixed issues for 3.6.1 release.

- Hazelcast 3.6 fails to create `ObjectName` for an instance created with JCache API. [7548]
- The test `com.hazelcast.client.spi.impl.ClientInvocationTest.executionCallback_FailOnShutdown` fails. [7462]
- There are failures in `SystemClockChangeTest`. [7444]
- When you use EntryProcessor with a map interceptor, it does not get the correct value. [7414]
- Backup records also increase the owned cache entry count but they should be ignored. Cache statistics show incorrect entry count via the method `getOwnedEntryCount`. [7403]
- When getting values from `PredicateSupplier`, the method `DistinctValueAggregation.DistinctValueMapper.map()` throws a `ClassCastException`. [7398]
- Timeout is not enough for the test `TransferStressTest.testLargePackets`. [7393]
- There are failures in the test `ClientExceptionFactoryTest.testException`. [7360]
- The method `getReplicationImplObject` in `WanTargetClusterConfig` is not used. [7353]
- Entry processor and map put/remove tests in WAN replication module fail. [7352]
- Hazelcast namespace for Spring configuration does not support Discovery SPI. [6913]
- When Hazelcast Spring configuration is used for the client and if a serialization configuration is present in the Spring client, then the client hangs forever during the authentication method. [5815]

Chapter 5

3.6

This section lists the new features, enhancements and fixed issues for 3.6 release.

5.1 New Features

The following are the new features introduced with 3.6 release.

- **High-Density Memory Store for Hazelcast Map**: With this release, Hazelcast Map data structure is now equipped with the High-Density Memory Store, previously implemented for Hazelcast JCache. Please refer to the Setting In Memory Format section.
- **Discovery Service Provider Interface (Discovery SPI)**: You can use this SPI to discover Hazelcast instances on cloud environments provided by jclouds®, Kubernetes and many more. The existing discovery mechanisms that Hazelcast provides (Multicast, TCP/IP and Amazon EC2) have been re-implemented on top of this new Discovery SPI. Please refer to the Discovery SPI section.
- **Client Protocol**: This feature presents the Hazelcast's new open binary client protocol. Please refer to Open Binary Client Protocol Documentation.
- **Client Cross Version Compatibility**: Now you can upgrade your Hazelcast clients independently from servers and other clients. Please refer to Open Binary Client Protocol Documentation.
- **Support for cloud providers through jclouds®**: Hazelcast now supports deployments on all the well-known cloud providers through the jclouds® open source library. Please refer to the Discovering Members with jclouds section.
- **Hot Restart Persistence**: This new feature provides fast restarting of the Hazelcast clusters. This is achieved by storing the state of the cluster members to the disk. Please refer to the Hot Restart Persistence section for more details.
- **Ringbuffer and Reliable Topic in Hazelcast Clients**: The data structures Ringbuffer and Reliable Topic recently introduced by Hazelcast (with the release 3.5) are now implemented for Hazelcast Java Client. Ringbuffer has also been implemented for .NET Client.
- **Cluster Quorum for Hazelcast JCache**: Cluster Quorum checks are now provided for Hazelcast JCache implementations, too. Please refer to the Defining a Cluster Quorum section to refresh and to the ICache Configuration section to learn configuring it for JCache.
- **Split Brain Syndrome handler for Hazelcast JCache**: Now Split Brain Syndrome is handled in JCache as it is taken care in Hazelcast Map. Please refer to the JCache Split-Brain section.

- **Partition Lost Listener for Hazelcast JCache**: You can listen to partition lost events fired in your Hazelcast JCache implementation. Please refer to the ICache Configuration section.
- **Hazelcast Docker image**: Now you can run Hazelcast using our image in the Docker platform. Please refer to Deploying using Docker.
- **Lite Members**: With the re-introduction of Hazelcast Lite Members (it was removed starting with Hazelcast 3.0 release), you are able to specify certain members in your cluster so that they do not store data. You can use these lite members mostly for your task executions and listener registrations. Please refer to Enabling Lite Members.

- **Querying in collections and arrays**: Hazelcast is now able to query and index attributes of objects stored in a collection or array. Please refer to the Querying in collections section.
- **Custom attributes extraction**: It is now possible to extract a value of an object's attribute using a custom extractor class. Please refer to the Custom attributes.
- **Acquiring locks with a lease time**: Now, you can try to acquire locks with a lease time. Please refer to the the comment for the method `tryLock()` in ILock code.
- **Monitoring the WAN replication**: You can now monitor the state of your WAN replications using the Hazelcast Management Center. Please refer to the Monitoring WAN Replication section.

5.2 Enhancements

The following are the the enhancements introduced with 3.6 release.

- **Replicated Map improvements**: The implementation of Hazelcast replicated maps has been revisited. Please especially refer to the Considerations for Replicated Map section.
- **Management Center improvements**: Alerting mechanism added. Please refer to the Management Center section.
- **Paging Predicate improvements**: With the performed improvements, now random page accessing is supported. Please refer to the Filtering with Paging Predicates section.
- **Rule based query optimizations**: This improvement introduces a query optimizer based on static rewriting rules. The optimizer treats predicates as immutable and returns a modified copy when the optimized one is found. Please refer to the `hazelcast.query.optimizer.type` property definition in the System Properties section.
- **WAN replication improvements**: With the improvements performed on Hazelcast's WAN replication feature, you can now monitor WAN replication events for each data structure and WAN replication now supports different acknowledge types for each target cluster group. Please refer to the WAN Replication Event Filtering API section and WAN Replication Acknowledge Types section for more information.
- **Improvements on Hazelcast's OSGI support**: With this improvement, Hazelcast bundles provide OSGI services so that the users can manage (create, access, shutdown) the Hazelcast instances through this service on OSGI environments. Having the `hazelcast.osgi.start` property enabled, when an Hazelcast OSGI service is activated, a default Hazelcast instance is created automatically. These instances can be served as an OSGI service to be accessed by other bundles. Registering the created Hazelcast instances behavior is enabled by default and can be disabled using the `hazelcast.osgi.register.disabled` property. Each Hazelcast bundle provides a different OSGI service and their instances can be grouped (clustered) together to prevent possible compatibility issues between different Hazelcast versions/bundles. This grouping behavior is enabled by default and can be disabled using the `hazelcast.osgi.grouping.disabled` property. Hazelcast OSGI service's lifecycle (and also the owned/created instances' lifecycles) are the same as the owner Hazelcast bundles. When the bundle is stopped (deactivated), owned service and Hazelcast instances are also deactivated/shutdown and deregistered automatically. Then, when the bundle is re-activated, its service is registered again. In addition, the Hazelcast Enterprise JAR file is also an OSGI bundle like the Hazelcast OSS JAR file.

The following are the other improvements performed to solve the enhancement issues opened by the Hazelcast customers/team.

- Approximate `max-size` calculation should be removed for IMap eviction. [6463]
- `SpringAwareWebFilter` should have a constructor which takes properties as arguments. [6438]
- Client side and server side cache proxies handle `putAll` operation one by one. This is not efficient. Records for this operation should be grouped as per their partitions and should be sent and processed in batches. [6367]
- Not requested events should not be sent to `MapListener` [6349]
- Inconsistent and potentially buggy design in `BasicCompletableFuture`. [6080]
- Starting with "hazelcast-wm 3.3", OSGI Manifest Spring package imports should be optional. [6072]
- The new client determines the partition ID for every invocation for data structures like queue and list where the partition ID is static. There is no need for this behavior. It should calculate the partition ID for once when the proxy is created and continue to re-use it. [5848]

- `Map.Entry` supplied to Entry Processor is not Serializable any more. [5611]
- The configuration file `minimal-json` with the provided scope is not picked up by the *shade* plugin. [5543]
- In Spring configuration, when a boolean property is injected for *hazelcast* bean (`<hz:hazelcast:....</hz:hazelcast`) a `SAXParse` exception is thrown. [5528]
- Currently, key/value pairs are deserialized prior to the execution of entry processor by default. This leads to the need of domain object at the server side, even if entry processor never uses it. [5301]
- In Spring XML configuration, the attributes of `socket-options` should be of type `xs:string`. [4700]
- `ClientMembershipEvent` does not need to have the `member` field. [4282]
- Hazelcast has `lock` with lease time feature but does not support `tryLock` with lease time. [1564]

5.3 Fixes

The following are the fixed issues solved for 3.6 release.

3.6 Fixes

- In the manifest file, `org.jclouds.*` should be marked as optional dependencies. [7318]
- Tests are needed for `WanReplicationPublisherDelegate`, `WanReplicationEvent`, `MapReplicationUpdate` and `AbstractMultipleEntryBackupOperation` in the Open Source WAN API. [7315]
- Invocation of quorum listener requires at least an attempt to perform a map operation. But it should not require this; just the crash of nodes should be enough to use the quorum mechanism. [7300]
- Owned entry count to be used as the expected near cache hit count should be calculated by checking the partition ownership in the `NearCacheTest::testGetAll`. [7285]
- The parameter `minEvictionCheckMillis` controls the maximum frequency of evictions. It is 100ms by default. It means at most 1 eviction is executed in a 100ms interval. No other `put()` operation within this interval triggers an eviction. So, if the put rate is greater than 1 per 100ms, then the number of entries is growing regardless of the `max-size-policy`. This eventually triggers a forced eviction which will prevent `OutOfMemoryException`. Forced evictions are only hiding this issue. Another possible solution is to keep the default interval as it is and apply batching: When X eviction cycles are skipped due the `minEvictionCheckMillis` parameter, then during the next cycle X + 1 entries should be evicted instead of just 1. [7268]
- Descriptions of some maximum size policies defined in the `com.hazelcast.config.EvictionConfig.MaxSizePolicy` and `com.hazelcast.config.MaxSizeConfig.MaxSizePolicy` are not clear and confusing. They should be clarified. [7267]
- Tests under `TopicOverloadDistributedTest` are spuriously failing even on the local machine. They need to be reviewed. [7266]

3.6-RC1 Fixes

This section lists the enhancements and fixed issues for 3.6-RC1 (Release Candidate 1) release.

- Javadoc for `IMap.putAll()` does not mention the lack of atomicity in the invocation. [7256]
- When a WAN Queue overrun occurs (with exception enabled), the source cluster logs an excessive amount of noise. This should to be logged. [7242]
- On WAN Replication Queue overrun, a `WANReplicationQueueFullException` is expected, but instead, the client receives an `UndefinedErrorCodeException`. [7241]
- When using Hazelcast as a JCache provider: As JSR-107 Javadoc states, an update should not reset expiry time for `CreatedExpiryPolicy`. However, when a cache entry is updated, it does not expire. [7236]
- Default WAN acknowledge type should be `ACK_ON_RECEIPT`. [7160]
- `NullPointerException` is thrown in `ClientRegressionWithMockNetworkTest`. [7148]
- Changing clusters in the Management Center does not update/refresh the cluster members in the Scripting tab. [7119]
- A fix is needed for operation retries in `PartitionCheckIfLoadedOperation`. [7114]
- WAN Queue counts in the Management Center for Hazelcast 3.6-EA3 are not correct. [7100]
- Hazelcast 3.6 Reference Manual is not correct for its Enterprise WAN Replication content. [7099]

3.6-EA3 Fixes

This section lists the enhancements and fixed issues for 3.6-EA3 (Early Access 3) release.

- `NullPointerException` is thrown for the thread `cached4` in a test which uses `MapLoader`. [7098]
- The method `loadInternal` of `MapProxySupport` requires `dataKeys`. Hence, a serialization step should be added to `MapProxy.loadAll()`. [7090]
- Near cache heap cost calculation is not proper when the cache gets concurrent misses. [7057]
- `IQueue` accepts null values from the Hazelcast Java client. [7048]
- `WriteBehindMapStore` for a map that has `OBJECT` as the in-memory format causes the entry processors to serialize the objects. [7040]
- Latest code does not include the file `com.hazelcast.client.impl.protocol.codec.CacheContainsKeyCodec` and build fails. [7019]
- Two members of a cluster become masters and ignore each other. [7016]
- `AbstractCacheRecordStore` should update the field `isOwner` while it is being cleared after migration. [6983]
- There are memory leaks in the local map statistics and near cache invalidation queues. The map containers also leak memory caused either by the near cache invalidation mechanism (when re-creating objects to check whether the near cache is enabled) or `MapPartitionDestroyOperation` (when re-creating objects and trying to the backup count). [6972]
- When the `lite-member` flag is used within the Spring context, its `enabled` attribute does not work properly. [6945]
- `LoadAllTask` for the client and server side cache proxies should also handle the `Throwable`, not just the `Exception`. [6944]
- The `enable` attribute of the `partition-group` element in the `Hazelcast-Spring.xsd` scheme should have the type `string`, not `boolean`. [6927]
- There is a left-over method in the Discovery SPI configuration, namely `addDiscoveryProviderConfig`. [6911]
- `InMemoryFormat.OBJECT` does not work with the `max-size` policies `USED_HEAP_SIZE` and `USED_HEAP_PERCENTAGE`. [6875]
- `PublicAddressTest` has been ignored due to the running time. [6858]
- `NullPointerException` is thrown in `ClientExecutionPoolSizeLowTest`. [6853]

3.6-EA2 Fixes

This section lists the enhancements and fixed issues for 3.6-EA2 (Early Access 2) release.

- MapLoader may insert null values into IMap causing memory leak. [6830]
- When replicated map entries are migrated to a new destination; TTL eviction should be scheduled, eviction should be retried when a failure caused by the migration happens and the sync interval should be increased. [6799]
- There is a logical error in the method `Ringbuffer.readManyAsync()` when `minSize = 0`. In this case, the Ringbuffer is not read and nothing is returned. [6787]
- When a listener's registration is made from the listener configuration, an error occurs during the listener initialization. [6784]
- Remaining cache invalidation messages should be flushed on the `ICacheService` while the member is in the `SHUTTING_DOWN` state. [6778]
- When a client cannot send a request to one of the connections, `TargetNotMemberException` is thrown. This name is confusing the Hazelcast users. [6766]
- `ClassCastException` is thrown when using `Timestamp` within `DataSerializable`. [6759]
- The method `destroyDistributedObject()` of `ReplicatedMapService` iterates over partition containers and record stores and destroys them. While destroying, record store calls `destroyDistributedObject()` which leads to an infinite loop. [6754]
- Hazelcast does not inject its instance into `HazelcastInstanceAware` registered via classname. [6697]
- There is a sporadic startup failure in 3.6-EA. [6684]
- There is no need to use `CacheLoader` inside the client/server side cache proxies. [6676]
- Fixed wrong calculation of eviction removal size when `PER_NODE` `max-size` policy is used. [6675]
- If the cluster state is not active `RepartitioningTask` should not be triggered. Otherwise, it causes infinite retries and prevents the member from shutdown. [6663]

- There are broken XML configuration tests in the Hazelcast client package. [6633]
- There is a memory leak since the method `publishBathcedEvents` does not remove the events from `batchEvent`. [6618]
- Custom credentials class is not de-serialized on the server side. [6615]
- Lite member element should be added to the Hazelcast Spring configuration. [6605]
- `EntryListener` shows the unprocessed value in combination with `PostProcessingMapStore`. [6588]
- Clients cannot submit `HazelcastInstanceAware` callables. [6570]

3.6-EA Fixes

The following are the issues solved for Hazelcast 3.6-EA (Early Access) release.

- The method `map.size()` waits indefinitely after the shutdown of a node. [6538]
- `HazelcastCachingProvider` does not use the specified instance (by the object) when `instance-name` is not specified. [6454]
- `onExecutionFailure` should be called before returning from `run`, if backup is not valid. [6420]
- `OperationThread.priorityPendingCount()` should return `scheduleQueue.prioritySize()` instead of `scheduleQueue.normalSize()`. [6318]
- There is a growth in heap usage caused by a memory leak in the following scenario: A node in the cluster regularly creates maps and puts entries into it, again in regular intervals. Another node removes the entries minutes after they were put, and if the map is empty, it destroys the map. [6317]
- Currently, there is an `EntryEvictedListener` that is notified both for expiration and eviction events. There should be a separate listener for expired entries: eviction happens due to size constraints, and expiry is once the entry has expired. [6311]
- `InvocationFuture`s async calls do not detect the lost operations. [6250]
- When the method `setBooleanAttribute` of the class `Member` is run, Null Pointer Exception is occurred on `STDOUT`. The problem is in the method `sendMemberAttributeEvent` of the class `ClusterServiceImpl`. [6223]
- `IOBalancer` keeps references of all the socket reader/writers but when destroying the connection, they release the references for only the ones which has endpoints. This causes a memory leak. [6199]
- `ILIKE` and `Regex` examples should be added to the Reference Manual under the "Supported SQL Syntax" section. [6190]
- `GroupProperty` defaulting does not work properly when programmatic configuration is used. [6174]
- When integrating Hazelcast in Spring Boot: if `HazelcastInstance` is created using the default `newHazelcastInstance` static method, then an `HazelcastInstance` whose `Config` has a valid `configurationUrl` property is created. However, `XmlBuilder` does not set this URL in the configuration it parses. [6061]
- Hazelcast's latest snapshot run fails due to the introduction of `ClientExceptionFactory` which has been developed for exception processing and working well in that sense. [6010]
- The class `HazelcastXATest` has only fast and slow modes (nothing in between) and possibly due to this, sometimes a transaction is waiting for a timeout. Either the transaction recovery or the test class itself is racy. [5923]
- A memory leak occurs when a listener is added and removed from client. A "remove" runnable in the collection that is stored in `ClientEndpointImpl` is the leftover. This runnable collection is used to cleanup the listeners when client is disconnected, it should be removed too after the listener is removed. [5893]
- The class `CacheRemoveAllOperation` does not send the "completed" event in some cases, e.g. if `CacheRecordStore` for that partition is not created yet or if the filtered keys are empty. [5865]
- In the class `MapProxyImpl`, the methods `executeOnKey` and `submitToKey` create an `EntryOperation` with the thread ID set. This does not happen with the class `ClientMapProxy`. Therefore, the class `MapExecuteOnKeyRequest` should take a thread ID and set this on the generated `EntryOperation`. [5857]
- The method `IndexImpl.getRecords()` fails with Null Pointer Exception due to the inconsistency between the `not(...equals())` and `notEquals()`. [5807]
- The method `HazelcastHttpSession.getAttribute()` for WebFilter does not work when `deferredWrite` is set to `true`. [5798]
- When `hazelcast.nio.faststring` is enabled, `UTFEncoderDecoder` tries to create a `FastStringCreator`. However, if the reflection is not available due to the security manager, `buildFastStringCreator` returns null and consequently `StringCreator` becomes null. [5777]

- `hazelcast-jca-rar/pom.xml` references to `src/main/rar/ra.xml` which does not exist. [5760]
- The Maven profile `mvn clean compile -Pqa` does not exist but it is documented in the README of Hazelcast. [5746]
- `PerformanceLogFile` only compiles if JDK 1.7 or above is used. [5729]
- Currently, for every deserialization a `BufferObjectDataInput` is created. This generates waste since it is created with an array of data for every deserialization. The `BufferObjectDataOutput` is already cached; the input should use a similar approach. [5562]
- When any entities are defined as read only in the Hibernate L2 cache, an invalidation of the cache (such as caused by executing a native SQLQuery) leads to the error `UnsupportedOperationException`. [5562]
- The performance impacts of TWO_PHASE and LOCAL transaction types should be documented. [5075]
- Client requests are very inefficient when determining the partition ID. [4940]
- The method `keySet()` relies on `QueryOperation`. The `QueryOperation` does not accept `IterationType` - it always returns both keys and values. This can lead to unnecessary load and potentially even an OOM exception. [4642]
- Hazelcast is stuck in TIMED_WAITING when used as 2nd level cache for Hibernate. [4406]
- Management Center license loading problem when REST API is used. [189]
- Executor monitoring in Management Center does not show the “cancelled” operations" [177]
- When an alert for a data structure (map, queue, etc.) with its specific name is created, a `NullPointerException` is thrown after the cluster is reset. [175]
- Default directory name is hardcoded as “mancenter3.5” and it needs to be maintained for every major release. This process should be dynamic. [174]
- Throughput statistics for Map shows nothing when the `putAll()` method is used. [159]

Chapter 6

3.5.5

The following are the issues solved for Hazelcast 3.5.5 release.

- When `hazelcast.jmx` option is enabled, MBeans are created for every Hazelcast object but they are never removed. When destroying Hazelcast object with the method `destroy()`, `DistributedObjectEvent::getDistributedObje` throws `DistributedObjectDestroyedException` and stops the process of unregistering MBeans. MBeans are left forever causing memory leaks and they can be seen in VisualVM. [#7329]
- `IdGenerator` sometimes generates duplicate IDs if it is put under stress. [#7299]
- The method `IAtomicLong.compareAndSet()` does not properly backup its updated state. In a two node cluster, if you use this method and then shutdown the owner of the `IAtomicLong`, the remaining node no longer sees the updated value. [#7290]
- `InvocationMonitor` checks for the same invocations continuously. [#7170]
- The methods `IMap.getAsync` and `IMap.putAsync` should update the statistics `getCount` and `putCount`, etc. [#7109]
- `NullPointerException` is thrown for `CoalescedWriteBehindQueue.removeFirstOccurrence()`. [#7082]
- The quorum definition in the Spring context is not correct. [#6946]
- While publishing events, IMap operations convert the value to data even if the registered listener does not request the value. [#6866]
- Map entry event listeners are not invoked on the clients of WAN replication target cluster. [#6802]
- The method `putAsync` does not affect `LocalMapStats.getPutOperationCount()`. [#6731]
- Possible memory leak when using `IMap.containsKey`. Problem happens when `containsKey` is used for a value that exists in the `MapStore`. Entry is loaded from the store but it does not get added to the map (no event is triggered in this case), eventually the system crashes with no memory. [#6517]
- Performance test with Spring Batch throws `TargetDisconnectedException`. [#4230]
- Web session replication does not work as expected during a shutdown. [#3362]
- The parameter `session-ttl-seconds` is set after the instance creation. It should be set before the instance is created. [#2377]

Chapter 7

3.5.4

The following are the issues solved for Hazelcast 3.5.4 release.

- Fixed wrong calculation of eviction removal size when `PER_NODE` `max-size` policy is used. [#6674]
- Lazy deserialization is required while events are being processed. [#6582]
- Thread Dumps freeze in the Management Center. It shows the same thread dumps for all connected members, same traces and same thread IDs. And they are shown on the same line numbers. [#6536]
- In Hazelcast 3.4.*, the methods `migrationStarted`/`migrationCompleted` were only called once when a partition migration was done. But in 3.5, these methods are called twice for each partition. [#6396]

Chapter 8

3.5.3

The following are the fixed issues for Hazelcast 3.5.3 release.

- `ClientInvocationFuture` may hang when the deserialized response is null. [#6363].
- The method `CacheStatisticsImpl::getAverageRemoveTime()` uses the "get" count on the cache but it must use the "remove" count. [#6314].
- Hazelcast `console.sh` should support changing namespaces that contain space characters. [#6307].
- The client fails to properly reconnect to a single node cluster after the Hazelcast server is restarted. [#6168].
- Transactional Queue ordering on rollback can be violated. The reason is that the `QueueContainer` does not rollback the changes in the order that is opposite to the order of the items when they were added. It is a random order due to the fact that the changes are stored in a hashmap. [#6156].
- When a field, that is only available in a subclass of an interface, is indexed, Null Pointer Exception is thrown. [#6151].
- Extra `\r\n` in the body of the REST API responses causing warnings. [#6144].
- Near cache on the client size for replicated map does not get invalidated after replicated map changes. [#5694].
- `IList.iterator()` and `listIterator()` do not support the method `iterator.remove()`. [#5508].

The following are the enhancements performed for Hazelcast 3.5.3 release.

- Cache statistics are only supported at the server side and exposed by `CacheProxy`. At the client side, cache statistics are not calculated and supported so `UnsupportedOperationException` is thrown at `ClientCacheProxy` [#6262].
- Added `iam-role` support to Hazelcast Cloud module. So users can use Hazelcast Cloud module without configuring access keys. [#6262].

Chapter 9

3.5.2

The following are the fixed issues for Hazelcast 3.5.2 release.

- There is a performance issue: Even when the Spring boot application is doing nothing, CPU consumption is very high. A thread named ".hazelcast-wm.ensureInstance" consumes CPU around 70% because of the method `ClusteredSessionService.run()` [#6052].
- MapLoader blocks the entire partition when loading a single entry [#5818].
- The method `IMap.getAll` by-passes interceptors in the Hazelcast 3.3 and higher versions [#5775].
- `AWSJoiner` fails for the regions except us-east-1 [#5653].
- Getting an instance of `sun.misc.Unsafe` class does not work on HP-UX operating system [#5518].
- `AWSAddressTranslator` always uses the default region and this causes the HazelcastClient to be unable to join a Hazelcast AWS cluster in a non-default region [#5446].
- The test code `JettyWebFilterTest.java` does not fail properly [#5188].
- Management Center behaves unfriendly when map entries increase [#4895].
- In `hazelcast-client.xml`, if the region is configured but `host-header` is not provided, the configuration gives a default endpoint value of `ec2.amazonaws.com`. It should give, for example, `ec2.eu-west-1.amazonaws.com` when the region is eu-west-1 and `host-header` is not provided [#4731].
- Too much CPU is used when Hazelcast is idle [#81].

Chapter 10

3.5.1

The following are the fixed issues for Hazelcast 3.5.1 release.

- Hazelcast Management Center uses `UpdateMapConfigOperation` to update map configurations. This operation simply replaces the map configuration of the related map container. However, this replacement has no effect for `maxIdleSeconds` and `timeToLiveSeconds` properties of the map configuration since they are not used in the map container directly. They are assigned to the final variables during map container creation and never touched again [#5593].
- Destroying a map just after creating it produces double create/destroy events for `DistributedObjectListener` [#5592].
- Map does not allow changing its maximum size, TTL and maximum idle properties. However, these fields are editable in the "Map Config" popup of Management Center. These fields should be disabled to prevent misguiding [#5591].
- Map is destroyed using `IMap.destroy()` but then it is immediately recreated [#5554].
- There should be a better calculation when calling the method `getApproximateMaxSize()` related to casting. Its return type is `int` and this causes the map entries to be evicted all the time when, for example, the eviction policy for an IMap is set to heap percentage with the value 1% [#5516].
- All `onResponse()` calls on a `MultiExecutionCallback` should be made before the method `onComplete()` is called. There exists a race condition in `ExecutionCallbackAdapterFactory` which permits the method `onComplete()` to be called before all `onResponse()` calls are made [#5490].
- Hazelcast Management Center "Scripting" tab is not refreshed when a new node joins to the cluster [#4738].
- When updating a map entry which is replicated over WAN, the TTL (time to live) is not honored in the remote cluster map. When the timeout expires, the entry disappears from the cluster in which the key is owned, however it remains in the remote cluster [#254].

The following are the enhancements performed for Hazelcast 3.5.1 release.

- Client instances should spawn threads with their instance names added as prefix [#5671].
- The method `com.hazelcast.spi.impl.classicscheduler.ResponseThread::process` may catch throwables. When this occurs, it logs an unhelpful message, and ignores the actual exception. This method should be improved to additionally log the cause, or at least the exception class and message [#5619].
- The element `min-eviction-check-millis` in the map configuration does not exist in documentation [#5614].

Chapter 11

3.5

This section lists the new features, enhancements and fixed issues for 3.5 release.

11.1 New Features

The following are the new features introduced with 3.5 release.

- **Async Back Pressure**: The Back Pressure introduced with Hazelcast 3.4 now supports async operations. For more information, please see the Back Pressure section.
- **Client Configuration Import**: Hazelcast now supports replacing variables with system properties in the declarative configuration of Hazelcast client. Moreover, now you can compose the Hazelcast client declarative configuration out of smaller configuration snippets. For more information, please see the Composing Declarative Configuration section.
- **Cluster Quorum**: This feature enables you to define the minimum number of machines required in a cluster for the cluster to remain in an operational state. For more information, please see the Cluster Quorum section.
- **Hazelcast Client Protocol**: Starting with 3.5, Hazelcast introduces the support for different versions of clients in a cluster. Please keep in mind that this support is not valid for the releases before 3.5. Please see the important note at the last paragraph of the Hazelcast Java Client chapter's introduction.
- **Listener for Lost Partitions**: This feature notifies you for possible data loss occurrences. Please see the Partition Lost Listener section and MapPartitionLostListener section.
- **Increased Visibility of Slow Operations**: With the introduction of the `SlowOperationDetector` feature, slow operations are logged and can be seen on the Hazelcast Management Center. Please see the SlowOperationDetector section and Management Center:Members section.
- **Enterprise WAN Replication**: Hazelcast Enterprise implementation of the WAN Replication. Please see the Enterprise WAN Replication section.
- **Sub-Listener Interfaces for Map Listener**: This feature enables you to listen to map-wide or entry-based events. With this new feature, the listener formerly known as `EntryListener` has been changed to `MapListener` and `MapListener` has sub-interfaces to catch map/entry related events. Please see the Map Listener section for more information.
- **Scalable Map Loader**: With this feature, you can load your keys incrementally if the number of your keys is large. Please see the Incremental Key Loading section.
- **Near Cache for JCache**: Now you can use a near cache with Hazelcast's JCache implementation. Please see JCache Near Cache for details.
- **Fail Fast on Invalid Configuration**: With this feature, Hazelcast throws a meaningful exception if there is an error in the declarative or programmatic configuration. Please see the note at the end of the Configuration Overview section.
- **Continuous Query Caching**: (Enterprise only, since 3.5) Provides an always up to date view of an IMap according to the given predicate. Please see the Continuous Query Cache section
- Dynamic Selector Rebalancing
- Management of Unbounded Return Values

11.2 Enhancements

The following are the the enhancements introduced with 3.5 release.

- **Eventing System Improvements**: RingBuffer and Reliable Topic structures are introduced.
- **XA Transactions Improvements**: With this improvement, you can now obtain a Hazelcast XA Resource instance through `HazelcastInstance`. For more information, please see XA Transactions.
- **Query and Indexing Improvements**

The following are the other improvements performed to solve the enhancement issues opened by the Hazelcast customers/team.

- While configuring JCache, duration of the `ExpiryPolicy` can be set programmatically but not declaratively [#5347].
- Since near cache is not supported as embedded but only at client, at the moment, there is no need for `NearCacheConfig` in `CacheConfig` [#5215].
- Support for parametrized test is needed [#5182].
- `SlowOperationDetector` should have an option to not to log the stacktraces to the log file. There is no need to have the stacktraces written to the normal log file if the Hazelcast Management Center or the performance monitor is being used [#5043].
- The batch launcher should include the JCache API [#4902].
- There are no Spring tags available for Native Memory configuration [#4772].
- In the class `BasicInvocationFuture`, there is no need to create an additional `AtomicInteger` object. It should be replaced with `AtomicIntegerFieldUpdater` [#4408].
- There is no need to use the class `IsStillExecutingOperation` to check if an operation is running locally. One can directly access to the scheduler [#4407].
- Configuring NearCache in a Client/Server system only talks about the programmatic configuration of NearCache on the clients. The declarative configuration (XML) of the same is not mentioned [#4376].
- XML schema and XML configuration validation is not compliant for AWS configuration [#4310].
- The JavaDoc for the methods `KeyValueSource.hasNext/element/key` and `Iterator.hasNext/next` should emphasize the differences between each other, i.e. the state changing behavior should be clarified [#4218].
- While migration is in progress, the nodes will have different partition state versions. If the query is running at that time, it can get results from the nodes at different stages of the migration. By adding partition state version to the query results, it can be checked whether the migration was happening and the query can be re-run [#4206].
- XML Config Schema does not allow to set a `SecurityInterceptor` Implementation [#4118].
- Currently, certain types of remote executed calls are stored into the `executingCalls` map. The key (and value) is a `RemoteCallKey` object. The functionality provided is the ability to ask on the remote side if an operation is still executing. For a partition-aware operation, this is not needed. When an operation is scheduled by a partition specific operation thread, the operation can be stored in a volatile field in that thread [#4079].
- The class `TcpIpJoinerOverAWS` fails at AWS' recently launched eu-central-1 region. The reason for the fail is that the region requires v4 signatures [#3963].
- API change in `EntryListener` breaks the compatibility with the Camel Hazelcast component [#3859].
- The `hazelcast-spring-<`*version*`>.xsd` should include the User Defined Services (SPI) elements and attributes [#3565].
- XA Transactions run on multiple threads [#3385].
- Hazelcast client fails to connect when you provide variables from the system properties [#3270].
- Entry listeners are not called when the entries are modified by WAN replication [#2981].
- Map wildcard matching is confusing. There should be a pluggable wildcard configuration resolver [#2431].
- The method `loadAllKeys()` in map is not scalable [#2266].
- Back pressure feature should be added [#1781].

11.3 Fixes

The following are the issues solved for Hazelcast 3.5 release.

- Operation timeout mechanism is not working [#5468].
- `MapLoader` exception is not logged: Exception should be logged and propagated back to the client that triggered the loading of the map [#5430].
- Replicated Map documentation page does not mention that it is in the beta stage [#5424].
- The method `XAResource.rollback()` should not need the transaction to be in the prepared state when called from another member/client [#5401].
- The method `XAResource.end()` should not need to check `threadId` [#5400].
- The method `IList::remove()` should publish the event `REMOVED` [#5386].
- `IllegalStateException` with wrong partition is thrown when the method `IMap::getOperation()` is invoked [#5341].
- `WrongTarget` warnings appear in the log since the operations are not sent to the replicas when a map has no backups [#5324].
- When the method `finalizeCombine()` is used, Hazelcast throws `NullPointerException` [#5283].
- `WanBatchReplication` causes `OutOfMemoryException` when the default value for WAN Replication Batch Size (50) is used [#5280].
- When testing Hazelcast, it does not start as an OSGI bundle. After the OSGI package was refactored, the dynamic class loading of the Script engine was missed [#5274].
- XA Example from Section 11.3.5 in the Reference Manual broken after the latest XA Improvements are committed [#5273].
- XA Transaction throws `TransactionException` instead of an `XAException` on timeout [#5260].
- The test for unbounded return values runs forever with the new client implementation [#5230].
- The new client method `getAsync()` fails with a `NegativeArraySizeException` [#5229].
- The method `putTransient` actuated the MapStore unexpectedly in an environment with multiple instances [#5225].
- Changes made by the interceptor do not appear in the backup [#5211].
- The method `removeAttribute` will prevent any updates by the method `setAttribute` in the deferred write mode [#5186].
- Backward compatibility of eviction configuration for cache is broken since `CacheEvictionConfig` class was renamed to `EvictionConfig` for general usage [#5180].
- Value passed into `ICompletableFuture.onResponse()` is not deserialized [#5158].
- Map Eviction section in the Reference Manual needs more clarification [#5120].
- When host names are not registered in DNS or in `/etc/hosts` and the members are configured manually with IP addresses and while one node is running, a second node joins to the cluster 5 minutes after it started [#5072].
- The method `OperationService.asyncInvokeOnPartition()` sometimes fails [#5069].
- The `SlowOperationDTO.operation` shows only the class name, not the package. This can lead to ambiguity and the actual class cannot be tracked [#5041].
- There is no documentation comment for the `MessageListener` interface of ITopic [#5019].
- The method `InvocationFuture.isDone` returns `true` as soon as there is a response including `WAIT_RESPONSE`. However, `WAIT_RESPONSE` is an intermediate response, not a final one [#5002].
- The method `InvocationFuture.andThen` does not deal with the null response correctly [#5001].
- `CacheCreationTest` fails due to the multiple `TestHazelcastInstanceFactory` creations in the same test [#4987].
- When Spring dependency is upgraded to 4.1.x, an exception related to the `putIfAbsent` method is thrown [#4981].
- HazelcastCacheManager should offer a way to access the underlying cache manager [#4978].
- Hazelcast Client code allows to use the value *0* for the `connectionAttemptLimit` property which internally results in `int.maxValue`. However, the XSD of the Hazelcast Spring configuration requires it to be at least 1 [#4967].
- Updates from Entry Processor does not take `write-coalescing` into account [#4967].
- CachingProvider does not honor custom URI [#4943].

- Test for the method `getLocalExecutorStats()` fails spuriously [#4911].
- Missing documentation of network configuration for JCache [#4905].
- Slow operation detector throws a `NullPointerException` [#4855].
- Consider use of `System.nanoTime` in `sleepAtLeast` test code [#4835].
- When upgraded to 3.5-SNAPSHOT for testing, Hazelcast project gives a warning that mentions a missing configuration for `hazelcastmq.txn-topic` [#4790].
- `ClassNotFoundException` when using WAR classes with JCache API [#4775].
- When Hazelcast is installed using Maven in Windows environment, the test `XmlConfigImportVariableReplacementTest` fails [#4758].
- When a request cannot be executed due to a problem (connection error, etc.), if the operation redo is enabled, request is retried. Retried operations are offloaded to an executor, but after offloading, the user thread still tries to retry the request. This causes anomalies like operations being executed twice or operation responses being handled incorrectly [#4693].
- Client destroys all connections when a reconnection happens [#4692].
- The `size()` method for a replicated map should return `0` when the entry is removed [#4666].
- `NullPointerException` on the `CachePutBackupOperation` class [#4660].
- When removing keys from a MultiMap with a listener, the method `entryRemoved()` is called. In order to get the removed value, one must call the `event.getValue()` instead of `event.getOldValue()` [#4644].
- Unnecessary deserialization at the server side when using `Cache.get()` [#4632].
- Operation timeout exception during `IMap.loadAllKeys()` [#4618].
- There have been Hazelcast AWS exceptions after the version of AWS signer had changed (from v2 to v4) [#4571].
- In the declarative configuration; when a variable is used to specify the value of an element or attribute, Hazelcast ignores the strings that come before the variable [#4533].
- `LocalRegionCache` cleanup is working wrongly [#4445].
- Repeatable-read does not work in a transaction [#4414].
- Hazelcast instance name with `Hibernate` still creates multiple instances [#4374].
- In Hazelcast 3.3.4, `FinalizeJoinOperation` times out if the method `MapStore.loadAllKeys()` takes more than 5 seconds [#4348].
- JCache sync listener completion latch problems: Status of `ICompletableFuture` while waiting for completion latch in the cache must be checked [#4335].
- Classloader issue with `javax.cache.api` and Hazelcast 3.3.1 [#3792].
- Failed backup operation on transaction commit causes “”Nested transactions are not allowed!" warning [#3577].
- Hazelcast Client should not ignore the fact that the XML is for server and should not use default XML feature to connect to `localhost` [#3256].
- Owner connection `read()` forever [#3401].

Chapter 12

3.4.6

No changes for this release. There are some minor internal improvements.

Chapter 13

3.4.5

No changes for this release. There are some minor internal improvements.

Chapter 14

3.4.4

The following is the fixed issue for 3.4.4 release.

- MultiMap entry listener provides incorrect null values [#5538].

Chapter 15

3.4.3

The following is the the enhancement performed for 3.4.3 release.

- Expose `TcpIpJoiner.MAX_PORT_TRIES` as a configurable property [#5062].

The following are the fixed issues for 3.4.3 release.

- Subsequent remove operations may cause reading of stale value from the map store [#5368].
- Write-behind may cause reading of stale value upon migration [#5339].
- Hazelcast client is unresponsive. `OperationTimeoutException` is seen in the logs [#5338].
- Last update time of an entry should not be changed after `getAll()` is invoked [#5333].
- `AtomicReference.alterAndGet()` throws `HazelcastSerializationException` [#5265].
- `ICompletableFuture` callback from the method `getAsync` is not always invoked [#5133].
- Warnings and exceptions are logged when closing the client connection [#4966].
- `CacheConfig` is not created on the cluster if the executer of `CacheCreateConfigOperation` has already a `CacheConfig` [#4960].
- The schema does not allow for an explicit `hz:replicatedMap` element to be created. One can be created inside `hz:config` but not as a definition for a concrete Replicated Map. Therefore, at present it is impossible to define a Replicated Map using Spring. [#4958].
- `ResponseThread` and `InvocationRegistry.InspectionThread` reset and retry operations. Since these threads did not implement `NIOThread`, the `OperationExecutor` is free to execute tasks on these threads and that is not desirable [#4929].
- The method `CacheManager.getCache()` does not re-open the closed cache. It should let access to the closed cache and re-open it. Cache can be accessed by `getCache` but it is still closed [#4631].
- The method `close()` of a Closeable `CacheLoader` is called without explicitly calling the method `Cache.close()` [#4617].
- The method `Cache.close()` does not call the method `close()` of registered Closeable `CacheEntryListener` [#4616].
- The method `awaitNanos()` returns the wrong value for both the `ClientConditionProxy` and `ConditionImpl` classes [#4603].
- The method `NotEqualPredicate` should return false if entry is null (without index) and also if index is present, it should not throw an exception with null values [#4525].
- When running Hazelcast with Spring and Hibernate 4 and when an application is started, the error related to `org/hibernate/cache/QueryResultsRegion` is produced [#4519].
- `OperationTimeoutException` when calling `get` on task future after `hazelcast.operation.call.timeout.millis` [#4398].
- Predicates with null values throws exception for unordered indexes [#4373].
- The method `queue.take()` does not get interrupted on shutdown [#4143].

Chapter 16

3.4.2

The following is the enhancement performed for 3.4.2 release.

- The method `contains` for ISet scans all the items [#4620].

The following are the fixed issues for 3.4.2 release.

- While executing unit tests, `SlowOperationDetectorThread` and `CleanupThread` may not be terminated before the next test is started [#4757].
- When multiple nodes join sequentially after partitions are assigned/distributed, old nodes fail to clean backup replicas larger than the configured backup count. This causes a memory leak. Also, when multiple nodes leave the cluster at the same time (or in a short period), the new partition owner looses some partition replica versions and this causes backup nodes for those specific replica indexes to fail synchronizing data from the owner node, although the owner node holds the whole partition data [#4687].
- After cluster merges due to a network-split, Hazelcast infinitely logs `WaitNotifyServiceImpl$WaitingOp::WrongTargetE` warnings [#4676].
- A strange `mapName` parameter occurred when using wildcard configuration for a custom `MapStoreFactory` [#4667].
- The method `IExecutorService.submitToKeyOwner` encountered two errors: the `onResponse` method is invoked with null and a cast exception is thrown in a Hazelcast thread [#4627].
- The method `init` in an implementation of the `MapLoaderLifecyleSupport` interface is not invoked [#4623].
- The method `readData` in `NearCacheConfig` reads the `maxSize` twice [#4609].
- The system property `hazelcast.client.request.retry.count` is not handled properly [#4592].

Chapter 17

3.4.1

The following are the enhancements performed for 3.4.1 release.

- When the near cache is used, cached entries from the remote node are evicted by idleness, despite being read [#4358].
- HazelcastQueryResultsRegion is never expired/evicted. The cleanup() method in LocalRegionCache for the query cache instances is never called, thus the query cache lives forever [#3882].

The following are the fixed issues for 3.4.1 release.

- IMap.getAll does not put data to RecordStore upon loading from map store [#4458].
- In the ClientNearCache class, there is a comparator which is used in a TreeSet to find the entries that should be evicted. If there are CacheRecords with the same hit count or lastAccessTime (depending on the policy, i.e. LFU or LRU), all of them should be evicted [#4451].
- When using write-behind and the entries, which have not been stored yet, are evicted, duplicate calls to the map store is made [#4448].
- There is a memory leak caused by the empty await queues in WaitNotifyService. When more then one thread try to lock on an IMap key at the same time, a memory leak occurs [#4432].
- ClientListener is not configurable via ListenerConfig. HazelcastInstanceImpl.initializeListeners(Config config) does not honor ClientListener instances [#4429].
- The CacheConfig(CacheSimpleConfig simpleConfig) constructor is broken. Variable assignments should be fixed [#4423].
- In ReplicatedMap, the containsKey method should return false on the removed keys [#4420].
- During the Hazelcast.shutdownAll() process, LockService is shut down before the MapService and this may cause null pointer exception if there is something like isLocked check in some internal IMap operations [#4382].
- Hazelcast clients shut down in the case of an IP change of one or more of the configured node (DNS) addresses [#4349].
- Write-behind system coalesces all operations on a specific key in a configured write-delay-seconds window and it should also store only the latest change on that key in that window. Problem with the current behavior is; a continuously updated key may not be persisted ever due to the shifted store time during the updates [#4341].
- Issue with contains pattern in Config.getXXXConfig(). Since the actual wildcard search always does a contains matching, you cannot set a configuration for startsWith, for instance [#4315].
- ReplicatedMapMBean is not present in JMX [#4173].

Chapter 18

3.4

This section lists the new features, enhancements and fixed issues for 3.4 release.

18.1 New Features

The following are the new features introduced with 3.4 release.

- **High-Density Memory Store**: Used with the Hazelcast JCache implementation, High-Density Memory Store is introduced with this release. High-Density Memory Store is the enterprise grade backend storage solution. This solution minimizes the garbage collection pressure and thus enables predictable application scaling and boosts performance. For more information, please see High-Density Memory Store section.
- **Jetty Based Session Replication**: We have introduced Jetty-based web session replication with this release. This is a feature of Hazelcast Enterprise. It enables session replication for Java EE web applications that are deployed into Jetty servlet containers, without having to perform any changes in those applications. For more information, please see Jetty Based Web Session Replication section.
- **Hazelcast Configuration Import**: This feature, which is an element named `<import>`, enables you to compose the Hazelcast declarative (XML) configuration file out of smaller configuration snippets. For more information, please see Composing XML Configuration section.
- **Back Pressure**: Starting with this release, Hazelcast provides the back pressure feature which prevents the overload caused by pending asynchronous backups. For more information, please see Back Pressure section.

18.2 Enhancements

The following are the the enhancements performed for 3.4 release.

- Event packets sent to the client do not have "partitionId" [#4071].
- Spring Configuration for ReplicatedMap is Missing [#3966].
- `NodeMulticastListener` floods log file with INFO-level messages when debug is enabled [#3787].
- A Hazelcast client should not be a `HazelcastInstance`. It should be a "factory" and this factory should be able to shut down Hazelcast clients. [#3781].
- `InvalidateSessionAttributesEntryProcessor` could avoid creating strings at every call to process [#3767].
- The timeout for `SocketConnector` cannot be configured [#3613].
- The method `MultiMap.get()` returns `collection`, but this method should return the correct collection type (`Set` or `List`) [#3214].
- `HazelcastConnection` is not aligned with `HazelcastInstance` [#2997].
- Support for Log4j 2.x has been implemented [#2345].
- Management Center console behavior on node shutdown [#2215].
- When `queue-store` is not enabled, `QueueStoreFactory` should not be instantiated [#1906].
- Management Center should be able to say when cluster is safe and all backups are up to date [#963].

18.3 Fixes

The following are the fixed issues for 3.4 release.

- Deadlock happens in MapReduce implementation when there is a high load on the system. The issue has been solved by offloading Distributed MapReduce result collection to the async executor [#4238].
- When the class `ClientExecutorServiceSubmitTest.java` is compiled using the Eclipse compiler, it gives a compile error: "*The method submit(Runnable, ExecutionCallback) is ambiguous for the type IExecutorService*". The reason is that the `IExecutorService.java` class does not have some generics. The issue has been solved by adding these missing generics to the `IExecutorService.java` class [#4234].
- JCache declarative listener registration does not work [#4215].
- JCache evicts the records which are not expired yet. To solve this issue, the `clear` method should be removed that runs when the size is smaller than the minimum eviction element count (`MIN_EVICTION_ELEMENT_COUNT`) [#4124].
- Hazelcast Enterprise Native Memory operations should be updated in relation with the Hazelcast sync listener changes [#4089].
- The completion listener (JCache) relies on event ordering but if the completion listener is registered in another node then event ordering is not guaranteed [#4073].
- AWS joiner classname should be fixed since EC2 discovery is not working after the restructure [#4025].
- If an IMap has a near cache configured, accessing the near cache via the method `get(key)` does not count as an access to the underlying IMap. The near cache has its own `max-idle-seconds` element. However, if an entry is expired/evicted in the IMap, it also causes a near cache removal operation for the entry regardless of the `max-idle-seconds` of that entry in the near cache. The entry expires and is evicted even if the near cache is being hit constantly. When a near cache is hit, the underlying map should reset the idle time for that key [#4016].
- Getting a pre-configured Cache instance is not working as expected [#4009].
- Bounded Queue section in the Reference Manual is unclear and wrong [#3995].
- The method `checkFullyProcessed` of MapReduce throws null pointer exception. The reason may be that multiple threads attempt to start the final processing state in the JobSupervisor [#3952].
- Merge operation after a split brain syndrome does not guarantee that the merging is over [#3863].
- When a client with near cache configuration enabled is shut down, `RejectedExecutionException` is thrown [#3669].
- In Hazelcast `IMap` and `TransactionalMap`, read-only operations such as `get()`, `containsKey()`, `keySet()`, and `containsValue()` break the transaction atomicity [#3191].
- Documentation should clearly list features of and differences between native clients [#2385].
- Sections of Hazelcast configuration should be able to be imported so that these sections can be shared between other Hazelcast configurations [#406].

Chapter 19

3.3.5

The following is the fixed issue for 3.3.5 release.

- Make write-coalescing configurable for write-behind map-stores [#4438].

Chapter 20

3.3.4

The following are the fixed issues for 3.3.4 release.

- Predicate with short values is not working [#4293].
- Hits statistics copy and paste error in the method `ReplicatedRecord` [#4254].
- Serialization error on the ReplicatedMap when in-memory format is set to `BINARY` [#4205].
- Too long exception stacktraces if the Hazelcast client fails to receive data, and this leads to a failure on the client [#4192].
- Hazelcast client registers the translated public address instead of its own private address to the list of connections. This causes the client not to be able to remove the connection correctly [#4190].
- `TransactionType:Local` emits exceptions while committing. The normal behavior should be throwing the exceptions to the user [#4160].
- Map replication should mark expirable recordstore. Otherwise, in some situations, if one does not set the map wide expiration or map wide TTL, the key based TTL expiration may not work [#4144].
- The method `BasicInvocationFuture.response` should be cleared when `BasicInvocation.WAIT_RESPONSE` is read by the waiter thread. Otherwise, when the retry operation takes too much time, the waiting thread sees the same wait response multiple times and the operation may not timeout forever [#4123].
- Topic listeners should be unregistered when topic is destroyed [#4117].
- Invocations (and their operations) remain in the invocations map forever if the operation timeouts without a response [#4113].
- Timeout is needed for parallel query operations [#4074].
- Initial map load and `max-size-policy` conflict [#4066].
- MapStore operations should be retried and performed eventually after a temporary failure [#4061].
- The class `SynchronizedWriteBehindQueue` (from `com.hazelcast.map.mapstore.writebehind` package) is declared threadsafe in JavaDocs, but it is not [#4039].
- The method `RemoveIfSameOperation` does not set `dataOldValue` for the `entryRemoved` event [#4037].
- When a new node with a new field is added to a cluster and when a query over this node is attempted, the old nodes throw an exception (`com.hazelcast.query.QueryException: Unknown Portable field: newFieldName`) and the query fails by throwing the same exception. [#3927].
- At the moment, the internal state fields of a `Reducer` are required to be volatile to ensure the memory visibility effects after the suspension and continuation of a reducer. This requirement should be moved to be handled by the framework itself since it is tend to be forgotten [#3866].
- The method `executeOnKey` hangs when the server fails to handle a query [#3842].
- The `GlobalSerializerConfig#setImplementation()` parameter should be compatible with the implementation field [#3569].
- `ClientConsoleApp` should not define the file `hazelcast-client.xml` [#3554].
- When using a custom partitioning strategy and the configured backup count of a map cannot be fulfilled since a node defined in the custom partition group is down, a JMX service call is blocked in the while-loop at `com.hazelcast.map.MapService.createLocalMapStats` [#3526].

Chapter 21

3.3.3

The following are the fixed issues for 3.3.3 release.

- JCache average put time statistic is not calculated correctly [#4029].
- When sending backup, the replica address can be seen as null [#4001].
- Evicted events are sent before the added events to EntryListeners [#3992].
- In Management Center, the default login credentials cannot be deleted [#3990].
- Logger for `NodeMulticastListener` does not belong to `com.hazelcast` hierarchy [#3941].
- MapInterceptors are not removed when a node leaves the cluster [#3932].
- MapInterceptors of same type (`Class`) are chained [#3931].
- Expiration Time should not be updated. Its value is updated on every set operation on a map, but Hazelcast uses only the first value set for ExpirationTime. So a `getExpirationTime()` operation returns a wrong and misleading value. [#3923].
- When using the XML file to configure a Queue to use a `QueueStoreFactory`, a null pointer exception is thrown at `QueueStoreWrapper` [#3907].
- Excess logging on startup [#3869].
- `LifecycleService` should be terminated after the node cannot join to the cluster [#3843].
- The method `MapProxyImpl.aggregate` hangs sporadically [#3824].
- Currently, there is no class named `com.hazelcast.nio.utf8.EnterpriseStringCreator` in Hazelcast. So the class and its log messages should be removed from the code [#3819].
- Bad user interface experience in the management center. Maps menu item that contains maps with longer names cannot be expanded [#3815].
- When the shutdown button in the management center is hit multiple times, the nodes are shutdown again, after they are shutdown at the first place and restarted [#3718].
- Alert e-mails from the management center are not sent to the e-mail address [#3693].
- Instances with private IPs cannot be discovered on Amazon EC2 [#3666].
- Null pointer exception in the method `Records.buildRecordInfo` from the stabilizer `MapStoreTest` [#2956].

Chapter 22

3.3.2

The following are the fixed issues for 3.3.2 release.

- Reject multicast messages if the group configuration is not matching [#3806].
- `Map#getEntryView` should check expiration of a key [#3801].
- Hazelcast gets stuck in `HazelcastInstanceNotActiveException` loop during multicast join [#3732].
- Hazelcast fails to comply with `maxIdleTime` expiration when running EntryProcessors. A delay should be added to expiration times on backups [#3710].
- `containsKey()` in transactional context returns wrong value for keys deleted within transaction [#3682].
- `TransactionalMap.values()` returns stale values that was updated within the transaction boundary [#3668].
- Number of loaded keys should not exceed map's maximum size [#3608].
- During client node shutdown, if the cluster happens to be down, Hazelcast logs some extra messages at SEVERE level [#3493].

Chapter 23

3.3.1

The following are the fixed issues for 3.3.1 release.

- MapReduce Combiner creation is not threadsafe, but certain operations on mapping phase might need a concurrent creation of the combiners [#3625].
- When `connectionTimeout` property in ClientNetworkConfig is set to `Integer.MAX_VALUE`, the client could not connect to cluster since a default 2000 ms. extra value is added to `connectionTimeout` while connecting [#3615].
- User provided list results from combiner is colliding with the internally used multi-result list [#3614].
- While committing collection transactions, the collection item is being added to the collection container. However, this gives the warning "There is no suitable de-serializer for type" warning. Instead of collection item, transactional item should be added to the container [#3603].
- `MaxSizeConfig` constructor should convert zero size to `Integer.MAX_VALUE` [#3579].
- If deserialization of the client request fails, the exception is not propagated back to the client [#3557].
- "Lock is not owned by by the transaction" exception. This exception was received while testing how transactions are working with Map and MultiMap for some last Hazelcast releases [#3545].
- Main classes in `manifest.mf` files are not correctly set #3537.
- Count of evicted events may exceed the map size when "read backup data" feature is enabled #3515.
- `mancenter.war` from Hazelcast release 3.2.5 cannot be deployed to Glassfish 3.1.2.2 and it fails to deploy #3501.
- While evicting entries from a map with the method `evictAll`, locked keys should stay in the map #3473.
- In `hazelcast-vm` module, before every test, new server container is started. And after every test, running server is terminated. This behavior causes a long test execution time. Server start-up and termination should be done before and after test class initialization and finalization #3473.
- The method `IQueue.take()` method should throw InterruptedException, but throws HazelcastException instead #3133.
- Multicast discovery doesn't work without network #2594.

Chapter 24

3.3

This section lists the new features, enhancements and fixed issues for 3.3 release.

24.1 New Features

The following are the new features introduced with 3.3 release.

- Heartbeat for Java client: Before this release, a Java client could not detect a node as dead, if the client is not trying to connect to it. With this heartbeat feature, each node will be pinged periodically. If no response is returned from a node, it will be deemed as dead. Main goal of this feature is to decrease the time for detection of dead (disconnected) nodes by Java clients, so that the user operations will be sent directly to a responsive one. For more information, please see Client Properties.
- Tomcat 6 and 7 Web Sessions Clustering: Please see Web Session Replication.
- Replicated Map implemented: Please see Replicated Map
- WAN Replication improved: Added configurable replication queue size WAN Replication Queue Size.
- Data Aggregation implemented: Added common data aggregations, please find Aggregators documentation.
- EvictAll and LoadAll features for IMap: `evictAll` and `loadAll` methods have been introduced to be able to evict all entries except the locked ones and that loads all or a set of keys from a configured map store, respectively. Please see Evicting All Entries and Forcing All Keys to be Loaded sections for more information.
- Hazelcast JCache implementation introduced: Starting with release 3.3.1, Hazelcast offers its JCache implementation. Please see Hazelcast JCache Implementation for details.

24.2 Fixes

The following are the fixed issues 3.3 release.

- TxQueue cannot find reserved items upon ownership changes [#3432].
- Documentation update is needed to tell that PagingPredicate is only supported for Comparable objects if there is no comparator [#3428].
- `java.lang.NullPointerException` is thrown when publishing an event in ClientEndPointImpl [#3407].
- The `entryUpdated()` callback of a listener during a transaction always has a null `oldValue` in the EntryEvent [#3406].
- Documentation update with the links to code samples for integration modules [#3389].
- Hazelcast write-behind with `map.replace()` stores replaced items [#3386].
- XAResource's `setTransactionTimeout()` method is not correctly implemented [#3384].
- Hazelcast web session replication filter may die if response committed [#3360].
- Resource adapter state is never reset to `isStarted == false`, resulting in errors down the line [#3350].
- `PagingPredicate.getAnchor` does not return the correct value [#3241].

- If deserialization fails, calling node is not informed [#2509].
- CallerNotMemberException and WrongTargetException exceptions are thrown at random intervals [#2253].

RC3 Fixes

This section lists issues solved for 3.3-RC3 (Release Candidate 3) release.

- Parallel execution of `MapStore#store` method for the same key triggered by `IMap#flush` [#3338].
- When offering null argument in queue throws an exception but it adds null argument to collection, then `addAll()` performed on this list does not throw an exception [#3330].
- `java.io.FileNotFoundException` thrown by MapLoaderTest [#3324].
- MapMaxSizeTest Stabilizer test with SoftKill [#3291].
- Incompatible Spring and Hazelcast configuration XSDs [#3275].
- `ExpirationManager` partition sorting can fail [#3271].
- Configuration validation is broken [#3257].
- Code Samples for Spring Security and WebFilter Integration [#3252].
- WebFilter Test Cases are slow [#3250].
- Management Center and Weblogic Deployment Problem [#3247].
- Enabling Multicast and TCP/IP node discovery methods freeze the instances [#3246].
- `getOldValue` and `getValue` returns the same value when removing item from IMap [#3198].
- MapTransactionContextTest: member SoftKill and then HazelcastSerializationException and IegalStateException: Nested are thrown [#3196].
- `IMap.delete()` should not call `MapLoader.load()`[#3178].
- 3.3-RC3+: NPE in the method `connectionMarkedAsNotResponsive` [#3169].
- `WebFilter.HazelcastHttpSession.isNew()` does not check the Hazelcast Session Cache [#3132].
- Hazelcast Spring XSD files are not version agnostic [#3131].
- `ClassCastException: java.lang.Integer` cannot be cast to `java.lang.String` Query [#3091].
- Predicate returns a value not matching the predicate [#3090].
- Modifications made by Entry Processor are lost in 3.3-RC-2 [#3062].
- Hazelcast Session Clustering with Spring Security Problem [#3049].
- PagingPredicate returning duplicated elements results in an infinite loop [#3047].
- `expirationTime` on EntryView is not set [#3038].
- `BasicRecordStoreLoader` cannot handle retry responses [#3033].
- Short `await()` on condition of contended lock causes IllegalStateException [#3025].
- Indices and Comparable: not documented [#3024].
- Marking Heartbeat as healthy is too late [#3014].
- 3.3-RC2: `IMap#keySet` triggers value deserialization [#3008].
- `map.destroy()` throws DistributedObjectDestroyedException [#3001].
- Stabilizer tests Final profile, Xlarge cluster OperationTimeoutException [#2999].
- `com.hazelcast.jca.HazelcastConnection::getExecutorService` returns plain ExecutorService [#2986].
- Serialization NPE in MapStoreTest stabilizer, 3.3-RC3-SNAPSHOT [#2985].
- Bug with `IMap.getAll()` [#2982].
- Client deadlock on single core machines [#2971].
- Retrieve number of futures in loop in calling thread [#2964].

RC2 Fixes

This section lists issues solved for 3.3-RC2 (Release Candidate 2) release.

- `evictAll` should flush to staging area #2969.
- NPE exception in MapStoreTest [#2956].
- Fixed `AddSessionEntryProcessor` [#2955].
- `Added `StripedExecutor` to WanReplicationService [[#2947]](https://github.com/hazelcast/hazelcast/is`

- All read operations of map should respect expired keys [#2946].
- Fix test EvictionTest#testMapWideEviction [#2944].
- `Heartbeat check of clients from nodes [[#2936]](https://github.com/hazelcast/hazelcast/issues/2936)`
- WebFilter does not clean up timed- out sessions [#2930].
- Fix leaking empty concurrent hashmaps [#2929].
- Data loss fix in *hazelcast-wm* module [#2927].
- Configured event queue capacity [#2924].
- Client closes owner connection when a connection to the same address is closed [#2921].
- Close the owner connection if heartbeat timeout when client is smart [#2916].
- Set application buffer size to not exceed `tls` record size [#2914].
- EntryProcessor makes unnecessary serialization [#2913].
- Make evictable time window configurable [#2910].
- Fixes data loss issue when partition table is being synced and a node is gracefully shutdown [#2908].
- MapStoreConfig; implementation instance is not set, when configured via XML [#2898].
- LocalMapStats does not record stats about locked entries in 3.x [#2876].
- Concurrency security interceptor [#2874].
- Client hangs during split, if split occurs due to network error [#2850].
- Network connection loss does not release lock [#2818].

RC1 Fixes

This section lists issues solved for 3.3-RC1 (Release Candidate 1) release.

- It is not possible to copy the link from *http://hazelcast.org/download/* and run `wget` on it [#2814].
- `mapCleared` method for EntryListener is needed [#2789].
- The method `keySet` with predicate should trigger loading of MapStore [#2692].
- MapStore with write-behind: The method `IMap.remove()` followed by `IMap.putIfAbsent(key,value)` still returns the old value [#2685].
- Hazelcast cannot read UTF-8 String if "multiple-byte" characters end up at position that is an even multiple of buffer size [#2674].
- Current implementation of record expiration relies on undefined behavior of `System.nanoTime()` [#2666].
- Inconsistency at Hazelcast Bootup "Editions" message [#2641].
- `AbstractReachabilityHandler` writes to standard output [#2591].
- `IMap.set()` does not not remove a key from write behind deletions queue [#2588].
- `com.hazelcast.core.EntryView#getLastAccessTime` is invalid[#2581].

Chapter 25

3.2.7

No changes for this release. There are some minor internal improvements.

Chapter 26

3.2.6

The following are the fixed issues for 3.2.6 release.

- MapStore in write-behind mode throws Exception (Spring configured) [#3397].
- Wildcard pattern of the map (map is configured with a wildcard in its name) should be respected during evictions [#3345].
- Map eviction does not work when the policy is "USED_HEAP_PERCENTAGE" or "USED_HESAP SIZE" [#3321].
- Exceptions when using Portable serialization [#3313].
- When Hazelcast is used as drop-in replacement for Memcached, it causes errors [#3182].
- Null Pointer Exception is thrown by `MapService.dispatchEvent` [#3101].
- PagingPredicate returns duplicated elements which result in an infinite loop [#3047].
- `ContextClassLoader` is by default only set on some cached operation threads, not on most others [#2721].

Chapter 27

3.2.5

The following are the fixed issues for 3.2.5 release.

- Txn map keyset and values with portable entries is not working correctly. [#3152]
- `TransactionalMap.{putIfAbsent(k, v), replace(k, v), replace(k, v1, v2), remove(k, v)}` methods never release lock after transaction ends. [#3149]
- Test failure at `ClientMapTest.testMapStatistics`. [#3138]
- `NetworkConfig.setReuseAddress` is not available in the XML. [#3122]
- When a selector fails to open, the `AbstractSelector` does not throw an exception, but logs it and then continues. Also, when `select` throws an IOException, this exception is not dealt correctly. [#3105]
- Test failure at `QueryBasicTest.testInPredicateWithEmptyArray`. [#3060]
- Hibernate cache flush leaves ClientMapProxy in an inconsistent state. This cache flush triggers `IMapRegionCache.clear()` and the implementation here does not look correct since it leaves the "map" field in the inconsistent state (*context* = *null*) and prevents any further use of it. [#3004]
- Fixes operation execution/invocation on IO threads issue. [#2994]
- Node cannot recover from `MergeOperation` if target node exits the cluster. [#2937]
- Client fails to run due to the lack of `ClientTestApp` class. [#2817]
- Using Hazelcast Queue, assume that there is a system in which messages are actively being consumed by one consumer. When a second Hazelcast instance is started (i.e. second consumer for the same queue), Hazelcast throws an exception, then continues normally and there are two competing consumers on the same queue. [#2805]
- `IMap.submitToKey` and `IMap.executeOnKey` in combination with nodes joining/leaving the cluster result in data loss. [#2785]
- Too much delay for deciding heartbeat timeout. [#2766]
- When multiple predicates are combined by an `AndPredicate`, the first `IndexAwarePredicate` that is not indexed will be added to the "no index" list twice. [#2531]
- There appears to be a leak in the memory in `SecondsBasedEntryTaskScheduler` when idle timeout is enabled on a map. [#2343]

Chapter 28

3.2.4

The following are the fixed issues for 3.2.4 release.

- Assigning wrong item ID on backup when instance is shutdown in QueueStore. [#2842]
- `IQueue.take` throws HazelcastException instead of InterruptedException. [#2833]
- Hazelcast distribution has some left-over files. [#2821]
- Management Center cannot update map configuration on already created map proxies. [#2810]
- Transient test failure: `IMap.get(k)` returns null. [#2804]
- `IllegalArgumentException: Target cannot be null!` shows up when `MultipleEntryBackupOperationForNullValu` test is run by `executeOnKeys()` firstly. [#2754]
- When creating an instance with EntryProcessor, the backup process does not work. [#2752]
- Data loss happens in the web filter. [#2746]
- `BackupEntryProcessor` stores the value even though it is not set explicitly. [#2613]
- The test `listenerAddStress` fails often. [#2611]
- Predicate should fail when null argument is passed. [#2577]
- XAResourceWrapper does not honor contract of `XAResource#setTransactionTimeout`. [#2569]
- Allow `Predicates.and` and `Predicates.or` for more than 2 arguments. [#2563]
- Semaphore is given to the thread that is coming late. [#2472]
- `UnknownHostException` is logged when using hostnames for seed addresses. [#2125]
- The Java client seems to hang if there is blocking of a map call in map listener. [#2106]
- The cluster is not responsive when 2nd node joins. [#2105]
- Hibernate query caches are not configurable. [#2064]
- XA Transactions should be explained in the Reference Manual. [#2020]
- Encryption enabled results in cluster to hang under load. [#1897]
- SSL connections are unreliable. [#1896]

Chapter 29

3.2.3

The following are the fixed issues for 3.2.3 release.

- The method `TransactionalQueue.poll` fails to find an item that is put into the queue in the same transaction with a non-zero timeout. [#2669]
- IExecutorService fails when it is called with MemberSelector instance that returns 0 Members. [#2650]
- If statistics are enabled, map's `InMemoryFormat.Object` does not work. [#2622]
- There is a memory leak in long running Hazelcast instances because of a bug in `MapEvictionManager.MapEvictTask`. [#2596]
- Hazelcast client is missing an extensive XML Config Test and XSD Test. [#2587]
- The client may consider the original address list if no cluster member can be reached. [#2585]
- Locks are not cleaned upon the operation `map.destroy`. [#2582]
- Classpath: Configuration file discovery is not working. [#2565]
- The method `ClientService.getConnectedClients()` does not always return the correct number of clients. [#2541]
- Nodes leaving and joining from/to the cluster can cause multiple subscription callbacks to be sent. [#2539]
- Predicate fails with `NullPointerException` if the value is null. [#2522]
- Messages' order breaks with versions 3.2+. [#2513]
- The method `ClientProxy.destroy` should always clean the resources. [#2475]
- The method `HazelcastHttpSession.getAttributeNames` returns entries that were only read but not written. [#2434]

Chapter 30

3.2.2

The following are the fixed issues for 3.2.2 release.

- Client security callable fix. [#2561]
- Updating a key in a transaction gives listeners an entryAdded() callback instead of entryUpdated(). [#2542]
- Client ssl engine doesn't need keyStore and keyStorePassword. [#2525]
- Added support for Mapper, Combiner, Reducer, KeyValueSource to implement HazelcastInstanceAware. [#2502]
- Fixed alter function. [#2496]
- Return cached value upon IMap.get() if near cache is enabled. [#2482]
- Exception initializing hz:client. [#2480]
- Fixed portable serialization between different services versions. [#2478]
- Resolves a data race in the client proxy that can lead to an NPE. [#2474]
- Fixed partition group hostname matching. [#2470]
- Client shutdown issue: Improve logging. [#2442]
- Unnecessary synchronized lock when invoking com.hazelcast.instance.LifecycleServiceImpl.isRunning(). [#2454]
- If MapStoreFactory throws exception, instance hangs. [#2445]
- Semaphore is given to the thread that is coming late. [#2443]
- Lots of exceptions when shutting down connection. [#2441]
- Migration fails when statistics are disabled. [#2436]
- 3.2.1 regression: nested transactions are not caught and prevented. [#2404]
- Client proxy init synced. [#2376]
- Fixes hostname matching problem when interface has wildcards. [#2398]
- Fix weblogic shutdown backport. [#2391]
- NotWritablePropertyException connectionAttemptLimit with ssl client config. [#2335]
- Map-Reduce Operation fails, when another instance tries to form a cluster with an instance running a map reduce task. [#2354]
- EntryEvent getMember returning null when a node leaves the cluster. [#2358]
- NullPointerException in Bundle Activator. [#2489]

Chapter 31

3.2.1

The following are the fixed issues for 3.2.1 release.

- JCA problems have been fixed. #2025.
- C++ client compilation problems are fixed.
- Redo problem about Java dummy client is fixed.
- Round robin load balancer of Java client is improved.
- Initial timeout is for the initial connections in Java clients.
- Wildcard configuration improvement in near cache configuration.
- Unneeded serializations in EntryProcessor should be removed when the object format is *In-Memory* #2139.
- Race condition in near cache has been solved, immediate invalidation of local near cache was needed #2163.
- Predicate issue seen in transactions is solved.
- Comparator issue in map eviction is solved.
- Map eviction part has been refactored due to a race condition on map listener #2324.
- Stale data problem in client near cache has been solved #2065.
- Many checkstyle and findbugs issues are solved.

Chapter 32

3.2

This section lists the new features, enhancements and fixed issues for 3.2 release.

32.1 New Features

- **NIO Client**: New architecture based on NIO introduced to support more scalable and concurrent client usage.
- **MapReduce Framework**: MapReduce implemented for your key-value collections that need to be reduced by grouping the keys. Please see the interview and MapReduce section.
- **Order/Limit Support**: Now you can order and limit results returned by queries performed on Hazelcast Distributed Map.
- **C++ Client**: Native C++ client developed for C++ users which can connect to a Hazelcast cluster and realize almost all operations that a node can perform. Please see Native Clients.
- **C# Client**: Also, Native C# client that has a very similar API with Native Java client developed. Please see Native Clients.

32.2 Enhancements

- Size of a distributed queue via REST API can be returned. [#1809]
- InitialLoadMode configuration parameter (having Lazy and Eager as values) added to MapStoreConfig. [#1751]
- Tagging support for Executor Service introduced such that nodes can be tagged for IExecutorService. [1457]
- `getForUpdate()` operation for transactional map introduced. [#1033]
- Entry processor can run on a set of keys with the introduction of `executeOnKeys(keys,entryprocessor)` method for IMap. [1423]
- `getNearCacheStats()` introduced. Statistics for near cache can be retrieved. [#30]

32.3 Fixes

- `LocalMapStats.getNearCacheStats()` can return null when it is called before a map get that calls `initNearCache()`. [#2009]
- `testMapWithIndexAfterShutDown` fails in OpenJDK. [#2001]
- Portable Serialization needs objects to be shared between client and server. [#1957]
- Near cache entries should be locally invalidate on `IMap.executeOnKey()`. [#1951]
- OperationTimeoutException is thrown when executing task that runs longer than `hazelcast.operation.call.timeout.m` [#1949]
- `MapStore#store` was called when executing AbstractEntryProcessor on backup. [#1940]

- After an OperationTimeoutException is thrown from 'ILock.tryLock() (and after the system is back in a normal state), the named lock remains locked. [#1937]
- Hazelcast client needs OutOfMemoryErrorDispatcher. [#1933]
- Near Cache: Caching of local entries may lead to race condition. [#1905]
- After key owner node dies, it takes too much time for threads to wakeup from `condition.await()`. [#1879]
- Possible improvements/fixes for NearCache. [#1863]
- `MultipleEntryBackupOperation` does not handle deletion of entries. [#1854]
- If topics are created/destroyed, then the statistics for that topic are not destroyed and this can cause a memory leak. [#1847]
- PartitionService backup/replication fixes. [#1840]
- Cached null values remain in near cache after `evict` is called. [#1829]
- NullPointerException in MultiMap when the service is shutdown before the migration is processed. [#1823]
- Network interruption causes node to continually warn with WrongTargetException. [#1815]
- `DefaultRecordStore#removeAll` should be modified so that it keeps "key objects to delete" as a list, not a set. [#1795]
- Very long `operation.run()` call stack especially when high partition count is used. [#1745]
- When executing an entry processor with an index aware predicate, the index is not used, instead the predicate is applied to the entire entry set. [#1719]
- When one node goes down in a cluster with 2 nodes (where near cache is enabled), `containsKey` call hangs in the second node. [#1688]
- When deleting an entry from an entry processor by setting the value to null, it is not removed from the backup store. [#1687]
- Client calls executed at server side cause unwanted (de)serialization. [#1669]
- In `TrackableJobFuture.get(long, TimeUnit)`, there is a 100 ms of sleep-spin while waiting for the result of a MapReduce task to be set. [#1648]
- If `storeAll` takes much time and if instance terminates while map store is running, data can be lost. [#1644]
- A missing Spring 4 Cache method added to hazelcast-spring package (namely `public T get(Object key, Class type)`). [#1627]
- When eviction tasks are canceled, `scheduledExecutorService` is not cleaned. [#1595]
- `storeAll()` with new value for the same key should not be executed until any previous `storeAll()` operations with the same key are not completed. [#1592]
- When using native client to interact with Hazelcast cluster, some JMX MBean attribute values on cluster nodes are not set/updated. [#1576]
- `IMap.getAll(keys)` method does not read from near cache. [#1532]
- Near Cache *cache-local-entries* attribute is missing in `hazelcast-spring-3.2` XSD. [#1524]
- Exception while executing script in OpenJDK 8. [#1518]
- Infinite waiting on merge operations when cluster shuts down. [#1504]
- Client side socket interceptor is not needed to be MemberSocketInterceptor. [#1444]
- Near cache on the local node should be enabled if its InMemoryFormat is different from that of the map. [#1438]
- Async EntryProcessor does not deserialize the value before it is called back. [#1433]
- A submitted task cannot be canceled via the native client. [#1394]
- `executeOnKeys(keys,entryprocessor)` introduced on IMap. With this feature entry processor can be run on a set of keys. [#1339]
- FINEST logging should be guarded where appropriate. [#1332]
- False errors reported in Eclipse due to schema definition. [#1330]
- Index based operations are not synchronized with partition changes. [#1297]
- Management Center: InvocationTargetException in Tomcat console when a node is started and then stopped. [#1267]
- The system property `hazelcast.map.load.chunk.size` is being ignored in Hazelcast 3.1. [#1110]
- Master should fire repartitioning after getting confirmation from nodes. [#1058]
- SqlPredicate does not Implement equals/hashCode. [#960]
- `DelegatingFuture.isDone` seems to always return false until the method `DelegatingFuture.get` is called. [#850]
- Predicate support for entry processor. [#826]

RC2 Fixes

The following are the fixed issues for 3.2-RC2 (Release Candidate 2) release.

- `ClientService.getConnectedClients` returns all end points [#1883].
- MultiMap is throwing `ConcurrentModificationExceptions` [#1882].
- `executorPoolSize` field of ClientConfig cannot be configured using XML [#1867].
- Partition processing cannot be postponed [#1856].
- Memory leak at client endpoints [#1842].
- Errors related to management center configuration on startup [#1821].
- XML parsing error by client [#1818].
- `ClientReAuthOperation` cannot return response without call ID [#1816].
- `MemberAttributeOperationType` should be introduced to remove the dependency to `MapOperationType` [#1811].
- Entry listener removal from MultiMap [#1810].

RC1 Fixes

The following are the fixed issues for 3.2-RC1 (Release Candidate 1) release.

- *TransactionalMap* does not support `put(K,V,long,TimeUnit)` [#1718].
- Entry is not removed from backup store when it is deleted using entry processor [#1687].
- Possibility of losing data when MapStore takes a long time [#1644].
- When eviction tasks are cancelled, `scheduledExecutorService` should be cleaned [#1595].
- A fix related to *StoreAll* is needed in a write-behind scenario [#1592].
- Update problem at map statistics [#1576].
- Exception while executing script in OpenJDK 8 [#1518].
- StackOverflowError at `AndResultSet` [#1501].
- Near Cache using `InMemoryFormat.OBJECT` also for local node [#1438].
- Async entry processor is not deserializing the value before returning [#1433].
- Distributed Executor; *Future Cancel* is not working [#1394].
- `HazelcastInstanceFactory$InstanceFuture.get()` never returns when `newHazelcastInstance()` method fails/throws exception [#1253].
- Changes for *Vertx* on Openshift [#1176].
- Serialization should be performed after database interaction for MapStore [#1115].
- System property related to chunk size is passed over in Hazelcast 3.1 [#1110].
- Map backups lack eviction of some specific data [#1085].
- `DelegatingFuture.isDone` always returns false until get is called [#850].
- Predicate support for entry processor [#826].
- Full replication of Maps should be performed [#360].

Chapter 33

3.1.8

This section lists the new features, enhancements and fixed issues for 3.1 and 3.1.x releases.

33.1 New Features

This section provides the new features introduced with Hazelcast 3.1 release.

- Elastic Memory (Enterprise Extensions Only) is now available.
- Hazelcast Security is now available.
- Hazelcast JCA integration is back.
- Controlled Partitioning: Controlled Partitioning is the ability to control the partition of certain DistributedObjects like the IQueue, IAtomicLong or ILock. This will make collocating related data easier. Hazelcast map also supports custom partitioning strategies. A PartitioningStrategy can be defined in map configuration.
- TransactionalMap now supports `keySet()`, `keySet(predicate)`, `values()` and `values(predicate)` methods.
- Eviction based on USED_HEAP_PERCENTAGE or USED_HEAP_SIZE now takes account real heap memory size consumed by map.
- SqlPredicate now supports ” as escape character.
- SqlPredicate now supports regular expressions using REGEX keyword. For example, `map.values(new SqlPredicate("name REGEX .*earl$"))`.
- Hazelcast queue now supports QueueStoreFactory that will be used to create custom QueueStores for persistent queues. QueueStoreFactory is similar to map's MapStoreFactory.
- TransactionalQueue now supports `peek()` and `peek(timeout, timeunit)` methods.
- Client now has SSL support.
- Client also supports custom socket implementations using SocketFactory API. A custom socket factory can be defined in ClientConfig.
- Hazelcast IList and ISet now have their own configurations. They can be configured using config API, XML and Spring.
- `HazelcastInstance.shutdown()` method added back.
- OSGI compatibility is improved significantly.

33.2 Fixes

This section lists issues solved for 3.1 and 3.1.x releases.

- `ClassCastException` when using near cache. [#1941]
- Hazelcast management console does not allow update of map configuration in runtime. [#1898]
- MultiMap ConcurrentModificationExceptions. [#1882]

- Near Cache: Race Condition. [#1861]
- ClientReAuthOperation cannot return response without call ID. [#1816]
- REST put overrides TTL from the configuration. [#1783]
- Management Center usage should not trigger loading of map. [#1770]
- ResourceAdaptor fix to handle external HazelcastInstance breaking scheme. [#1727]
- There is no `executeOnAllKey` for IMap, the documentation should be changed. [#1711]
- Memory leak; `backupCalls` record is not removed if `Future.get()` is not called during `IMap.putAsync()` or `IMap.removeAsync()`. [#1693]
- `ClassCastException` when removing from Multimap. [#1667]
- com.hazelcast.spi.exception.RetryableHazelcastException: Map is not ready. [#1593]
- Member join while loading persistent maps throws NPEs. [#1577]
- Near cache does not work properly for null values. [#1570]
- Memory Leak on operation timeout. [#1559]
- EOFException: Remote socket is closed. [#1551]
- Severe NullPointerException but then everything works fine. [#1548]
- `IMap.putTransient(K,V)` method (without TTL) is missing. [#1526]
- Object is not an instance of declaring the class com.hazelcast.query.impl.QueryException. [#1503]
- Setting socket timeout, Hazelcast client will throw SocketTimeoutException. [#1435]
- `QueueStore.store(Long, Object)` is called multiple times from `TransactionalQueue.offer`. [#1412]
- Warning: No transaction item for itemId: . [#1393]
- StackOverflow in `AndResultSet`. [#1349]
- `maxIdleSeconds` is set to DEFAULT_TTL_SECONDS in `com.hazelcast.config.MapConfig`. [#1308]
- Serializer implementation registration with Spring config fails. [#1294]
- `ReadBackupData` on Map exposes internally stored value. [#1292]
- Multimap: Lock is not owned by the transaction. [#1276]
- Java deserialization of non-static inner classes fails. [#1265]
- HazelcastInstance is not injected into MapInterceptor when HazelcastInstanceAware is implemented. [#1233]
- Remove `compare` optimization from map put. [#1231]
- Unneeded deserialization in EntryOperation. [#1182]
- Adding an InitialMembershipListener to ClientConfig using `ClientConfig.addListenerConfig` throws NullPointerException. [#1181]
- Insufficient calculation of next key/value when doing a restore of message queue from database. [#1165]
- Case insensitivity of configurations in XML. [#1152]
- The scheme `hazelcast-spring3.1.xsd` throws SAXParseException when `hz:interfaces` have multiple `hz:interface` elements. [#1145]
- Bundle start fails due to `OSGiScriptEngineManager`. [#1136]
- Apparent memory leak when using near cache. [#1087]
- Semaphore can be initialized multiple times if permit-count = 0. [#1080]
- Exception when running continuous query. And non-existent key is deleted under transaction. [#1076]
- Race condition between first map operation and re-populating a map after a cluster restart. [#1070]
- Eviction tasks should be cancelled on `remove()`, `evict()`. [#1068]
- Stale cache issues when using `HazelcastLocalCacheRegionFactory`. [#1039]
- When multiple nodes start simultaneously, post join process duplicates some event registrations and causes warning logs on each publish. [#1024]
- Multimap entry listener is called twice. [#993]
- Clear logic on `ByteArrayObjectDataOutput` can create memory leak with large objects. [#989]
- TransactionException: Lock is not owned by the transaction. [#988]
- Spring schema declaration is missing. [#982]
- Null object indexing exception. [#978]
- Entries not changed by an EntryProcessor should not render an EntryListener event. [#969]
- Unlock parameter in TxnSetOperation backup operation is not serialized. [#956]
- PartitioningStrategy is not set in HazelcastClient. [#923]
- Enhancement request: Consider extending SqlPredicate to allow using regexes. [#914]
- Map `getAll()` is blocked with “Map is not ready exception”. [#887]

- QueueStoreConfig needs a factory to support wildcard creation. [#884]
- Data is lost when transaction manager node terminates/crashes without preparing transaction. [#863]
- Make `cache-null-value-seconds` customizable. [#806]
- Reintroduce the `Instanceof` predicate. [#790]
- Add `IMap.addEntryListener()` without key (continuous query). [#710]
- The calculation `used_heap_percentage` should take backups into account. [#403]

Chapter 34

3.0.3

This section lists the new features, enhancements and fixed issues for 3.0 and 3.0.x releases.

34.1 New Features

This section provides the new features introduced with Hazelcast 3.0 release.

- Multi-thread execution: Operations are now executed by multiple threads (by factor of processor cores). With Hazelcast 2, there was a only single thread.
- SPI: Service Programming Interface for developing new partitioned services, data structures. All Hazelcast data structures like Map, Queue are reimplemented with SPI.
- IdentifiedDataSerializable: A slightly optimized version of DataSerializable that does not use class name and reflection for de-serialization.
- Portable Serialization: Another Serialization interface that does not use reflection and can navigate through binary data and fetch/query/index individual field without having any reflection or whole object de-serialization.
- Custom Serialization: Support for custom serialization that can be plugged into Hazelcast.
- Entry Processor: Executing an EntryProcessor on the key or on all entries. Hazelcast implicitly locks the entree and guarantees no migration while the execution of the Processor.
- In-Memory Format: Support for storing entries in Binary, Object and Cached format.
- Continuous Query: Support for listeners that register with a query and are notified when there is a change on the Map that matches the Query.
- Interceptors: Ability to intercept the Map operation before/after it is actually executed.
- Lazy Indexing:Ability to index existing items in the map. No need to add indexes at the very beginning.
- Queue: No more dependency on the distributed map
- Queue: Scales really well as you have thousands of separate queues.
- Queue: Persistence Support for persistence with QueueStore.
- Multimap: Values can be Set/List/Queue.
- Topic: Support for global ordering where all Nodes receive all messages in the same order.
- Distributed Transaction: Support for both 1-phase (local) and 2 phase transactions with a totally new API.
- New Binary Protocol: A new binary protocol based on portable serialization. The same protocol is used for Java/C/C# and other clients.
- Smart client: Support for dummy and smart client. A dummy client will maintain a connection to only one member, whereas the smart client can route the operations to the node that owns the data.

34.2 Fixes

This section lists issues solved for 3.0 and 3.0.x releases.

- A class cannot be LifecycleListener and MigrationListener at the same time. [#970]

- EntryProcessor does not fire entry events. [#950]
- `IMap.localKeySet()` is very slow after upgrade to 3.0. [#946]
- Eviction logic error for `USED_HEAP_PERCENTAGE` and `USED_HEAP_SIZE`. [#891]
- Failed to execute SqlPredicate with LIKE and OR keywords on the same field of the class. [#885]
- Infinite loop in native client when cluster is gone. [#821]
- TransactionalMap: DistributedObjectListener is not working. [#812]
- Could not find PortableFactory for factoryId: -3. [#800]
- Eviction occurs when it is not expected to occur. [#776]
- Possible dead-lock while updating an entry which loaded by map-loader. [#768]
- `IMap.destroy()` clears map instead of destroy. [#764]
- `IMap.destroy()` throws an exception on client. [#726]
- Client replace method issue. [#724]
- Classloader issue in deserialization. [#723]
- JMX preDeregister of some beans throws HazelcastIsNotActiveException during `shutdownAll()`. [#713]
- Optional sequencing support for Hazelcast write-behind implementation is needed. [#712]
- ManagementService.destroy fails to find HazelcastMBean on shutdown. [#683]
- Client `getDistributedObject()` does not work. [#678]
- Spring ManagedContext does not run on the local node when a task is executed. [#659]
- MemberLeftException during SqlPredicate search. [#654]
- NearCacheConfig: maxSize = 0 is no longer interpreted as `Integer.MAX_VALUE`. [#650]
- `statistics-enabled` is missing for most items in the XSD scheme. [#642]
- Could not find PortableFactory for factoryId: 1095521605 on SqlPredicate query. [#617]
- `Map.containsKey()` should delay eviction. [#614]
- Error when serializing portable nested inside other types: Could not find PortableFactory for factoryId: 0. [#600]
- Client related configuration improvements. [#599]
- Index on `enum` properties does not work. [#597]
- Executing Query operation with a heavy CRUD load throws an IllegalArgumentException. [#596]
- `Map.set()` always calls the `onAdded` entry event. [#589]
- Client throws 'Could not register listener' often - Near Cache entries get never invalidated after. [#584]
- Unexpected authentication exception. [#576]
- Map `clear` does not call MapStore. [#572]
- Transactional collections should check the transaction state to validate proper access. [#568]
- Client threads are not destroyed on authentication failures. [#555]
- Map index from configuration is not working. [#553]
- Client disconnects and never tries to reconnect. [#538]
- Map local entry listener gives no Value on `entryEvicted`. [#537]
- The method `map.getAll()` is not working if entry does not exist via Client. [#535]
- EC2 Auto discovery bug: Instances do not see each other. [#516]
- Custom types (other than primitives, string, date, enum, etc) can not be queried. [#503]
- OOME is thrown when queues are not explicitly destroyed. [#417]
- The method `loadAll` is called redundantly on a new node joining the cluster. [#341]
- Support for Hibernate 4.0 cache is needed. [#72]
- Add a functionality that performs entry processings. [#71]
- Support for JVM system property reference in Hazelcast XML is needed. [#59]
- Support `invalidation-only` 2nd level cache for Hibernate. [#57]
- Hazelcast resource adapter does not work on WebSphere 6.1/7. [#37]

Chapter 35

2.6.9

The following are the fixed issues for 2.6.9 release.

- Hazelcast cannot match hostnames when wildcards are used in `interfaces` configuration element. [2396]

Chapter 36

2.6.8

The following are the fixed issues for 2.6.8 release.

- `OperationTimeoutException`: [CONCURRENT_MAP_SET] Redo threshold[90] exceeded! Last redo cause: REDO_MAP_OVER_CAPACITY. [1685]

Chapter 37

2.6.6

The following are the fixed issues for 2.6.6 release.

- Hazelcast 2.6.5 throws `NullPointerException` if the method `MapStore.load()` returns null during `IMap.getAll()` execution. [1605]
- In Hazelcast 2.x branch, Hazelcast instance is shutdown but exception is not logged. [1488]

Chapter 38

2.6.3

The following are the fixed issues for 2.6.3 release.

- Memcache client content in the Reference Manual needs to be improved . [1047]
- Hazelcast Hibernate `pom` is using a retired maven repo. [831]
- “Marked for removal” mechanism should be more clarified. [792]

Chapter 39

2.6.2

The following are the fixed issues for 2.6.2 release.

- Client Listener: the `Client` argument has null `Socketaddress` for `ClientDisconnected` listener. [756]
- Entries put by the method `IMap.putAll` were never evicted from the map with the TTL configured. [670]
- The Java client is initialized with two cluster members. After killing one member in the cluster where it is connected, the client is unable to reconnect. [653]
- Put operation after a missed get operation in a transaction does not have any effect. [644]
- Near cache functionality does not work for Java client. `MapConfig` always returns the generic map name since in the client, the client prefix is added to map's name. [620]

Chapter 40

2.6.1

The following are the fixed issues and enhancements for 2.6.1 release.

- The method `set(Key, Value, 0, Timeunit)` applies the configured TTL values where it should cancel eviction. Also, the method `set(Key, Value)` should apply the inherited TTL if exists. [585]
- Operations like `contains()` should use a specific executor service. Calling the default executor service may cause a deadlock. [579]
- Empty array for the predicate `in` throws exception. [573]
- The methods `ILock.destroy()` and `ISemaphore.destroy()` throw exceptions on Hazelcast 2.5. [566]
- `NullPointerException` in `com.hazelcast.nio.ConnectionManager` in Hazelcast 2.5. [530]
- Hazelcast instance does not start with TCP configuration in Hazelcast 2.5 when the hosts are not reachable. [523]
- Hazelcast source archives in Maven central repository does not contain the Java source files but the class files. [514]
- There is an unexpected store call behavior in Hazelcast 2.6. Every time when the method `get()` is used in a transaction and when the method `commit()` is called, it throws "Duplicated key order_id" exception. [506]
- When the method `getInstances()` on HazelcastInstance is called (acquired via `HazelcastClient.newHazelcastClient(`
the collection is retrieved. After some time, `NullPointerException` is thrown when the same method is called. [478]
- There are some formatting problems when using SQL predicates with date. [473]
- There is a memory leak in the member when clients are connected. An `OutofMemoryException` is thrown related to a client authentication failure. [450]
- Hazelcast client's `putAll()` method throws `NullPointerException` when called with an empty map. [397]
- The `InputStream` in the `XmlConfigBuilder` can never be closed. If the constructor which provides an XML file is used, it creates a new `FileInputStream`. There is no ability to close this input stream after it is created. [390]
- The heartbeat timer for Hazelcast client does not run again after a `NullPointerException`. [382]
- The queue `SysmteLogService.joinLogs` holds some data and causes memory consumption. [325]
- Doing a read of a value not modified in a transaction causes the listener to fire an update for that entry. [311]

Chapter 41

2.6

The following are the fixed issues and enhancements for 2.6 release.

- Issues with Spring configuration and merge policies. [488]
- Trying to create a configuration object by API, serializing and deserializing the objects (by Hazelcast builder), due to an error of PartitionGroup, `RuntimeException` is thrown. If set by API, it works. [487]
- OSGi: The manifest data in `hazelcast-all.jar` for 2.5.1 is not correct. [484]
- Consistency of the set of members in the cluster should be guaranteed. [477]
- Hazelcast IMap's `containsKey()`method does not reset the idle timeout. [472]
- `@SpringAware` annotation leads to memory leak when using Hazelcast distributed executor service. [470]

Chapter 42

2.5.1

The following are the fixed issues and enhancements for 2.5.1 release.

- Deadlock happens when Hazelcast client is shut down. [466]
- There is a data inconsistency and loss following the `ClassCastException: CONCURRENT_MAP_REPLACE_IF_SAME`. [462]
- Put operations with zero TTL does not prevent the eviction. [455]
- `NullPointerException` is thrown at `com.hazelcast.query.MapIndexService.remove` during CONCURRENT_MAP_REMOVE operation. [454]
- The exception `Failed migrating from Member` is thrown when two clients are started in Eclipse and one is stopped before it is completely initialized. [452]
- The method `IList.destroy()` does not empty the list. [449]
- The package `hazelcast-all.jar` should build its manifest dynamically. [448]
- The package `hazelcast-client.jar` does not deploy as OSGi bundle in JBoss7. [447]
- The package `hazelcast-wm.jar` should not require `servlet-api` v3.0. [432]
- `NullPointerException` is thrown when trying to run the method `clear()` on an empty distributed set when migrating from Hazelcast 2.1.2 to 2.4 and 2.5 [430]
- The method `getOldValue()` consistently returns wrong results for `entryUpdated` listener notifications after the `replace()` operation. [418]
- In the Hazelcast configuration schema XSD, the `tcp-ip` and `aws` elements do not have `conn-timeout-seconds` sub-element. [410]
- It would be nice if the `hazelcast.config` option can be used in combination with the classpath. [408]
- Deadlock happens when adding an index to a map that is not empty. [310]
- When custom loader is used with write-behind persistence mode, every time `store()` is called in loader and then `remove()` is called on locked map, value for the key is read from the store and not from map. Consequently, put/set/update is overridden and data is lost. [187]

Chapter 43

2.5

This section lists the new features, enhancements and fixed issues for 2.5 release.

43.0.1 New Features

The following are the new features introduced with 2.5 release.

- Added near cache support for Java Client.
- Management Center alert system can now receive alerts based on custom filters.
- Management Center has now better support for Hazelcast cluster running in OSGI environment.
- Nodes can be easily shutdown or restarted using Management Center interface.

43.0.2 Fixes

The following are the fixed issues and enhancements for 2.5 release.

- Management Center does not respond for the maps with names having underscore characters. [394]
- MultiTask operation returns `Constants$Object$6` instead of a map. [392]
- In the method `forceUnlock()`, local lock is not removed. [386]
- Deadlock happens when the cluster is being initialized. [386]

Chapter 44

2.4.1

This section lists the new features, enhancements and fixed issues for 2.4.1 release.

44.0.3 New Features

The following are the new features introduced with 2.4.1 release.

- Added Hibernate 2nd level cache local/invalidation mode.
- Added quick clear feature for maps.

44.0.4 Fixes

The following are the fixed issues and enhancements for 2.4.1 release.

- After a split-brain syndrome, states sent by a non-member should be ignored. [374]
- Indexing: Negative double values comparison causes problems in SQL predicate. [368]
- Query on Hazelcast IMap with indexes returns stale data. [359]
- By default, the method `map.clear()` clears the map entry by entry since the entries should be checked if they have a related lock, should be persisted, etc. If you do not use map store and can assume that there is no locked entries, then there should be an option to clear the map directly. [356]
- System logs should be optional. [351]
- Management center should show the Hazelcast version. [349]
- `UnsupportedOperationException` is thrown when the method `getConfig()` is called during the startup of the web application. Used versions are Hazelcast 2.4 and Hibernate 3.6.6. [339]
- Documentation is wrong. Hazelcast supports `REPEATABLE_READ` instead of `READ_COMMITTED`. [337]
- When you need a lite member, since it cannot be specified in the configuration file, you need to do the whole loading mechanism yourself. [333]
- In the local map statistics, currently there is only `getHits()`. There should be statistics for misses (get requests which return null) to see hit/miss ratio. [328]
- When Hazelcast OSGi bundle is used in Glassfish 3.1.2, `ClassNotFoundException` is thrown. [322]
- The method `MapConfig.toString()` should report `mapIndexConfig` too. [321]
- When the connection between the client and member is lost, the client does not notice it and tries to send the data for a very long time and it does not shut down.[315]
- Hazelcast tends to consume large amounts of memory in `com.hazelcast.nio.Packet` instances. [312]
- Map index entries are not deleted upon migration. When using SQL predicate in the method `localKeySet()`, the same key can be retrieved on different nodes. [308]
- Hazelcast IMap entries are evicted even they are touched by the method `put()`. [466]

Chapter 45

2.4

This section lists the new features, enhancements and fixed issues for 2.4 release.

45.0.5 New Features

The following are the new features introduced with 2.4 release.

- Client threads does not have fixed size anymore, now it uses internal Hazelcast cached thread pool.
- Added ability to restrict outbound ports that Hazelcast uses to connect to other nodes.

45.0.6 Fixes

The following are the fixed issues and enhancements for 2.4 release.

- Hazelcast Management Center's "Configuration has been successfully updated." notification is never removed. [301]
- Map maximum size does not take into account that some cluster members are lite members and cannot hold any data. Thus, the map becomes smaller than the size it was configured. [292]
- The method `map.containsKey()` does not reset the idle time counter. [288]
- In Hazelcast 2.3.1, the property `hazelcast.local.localAddress` is used only when provided directly from `System.properties`. [282]
- Data removed under transaction sometimes could not be removed from the backup. [277]
- Map initialization should not use the default executor service. [276]
- In Hazelcast 2.3.1, Hazelcast got into a state where the list of members on some of the nodes do not match up. [274]
- Map `EntryListener` is not working correctly if one master is restarted. [269]
- `IMap.tryLockAndGet`: There is a concurrency error when the map has map store. [268]
- Error in `com.hazelcast.query.Predicates.NotEqualPredicate`. [262]
- Partition Group with group type CUSTOM should honor configured backups. [260]
- Messages are lost while moving them inside a transaction. [259]
- Enable Hazelcast to use specified port ranges to accommodate firewall policies. [251]
- Seeing multiple concurrent locks on a distributed lock. [168]

Chapter 46

2.3.1

The following are the fixed issues and enhancements for 2.3.1 release.

- Changed `hazelcast.partition.migration.interval` property's value from 1 to 0.
- The method `ILock.newCondition()` now throws `UnsupportedOperationException`.
- After upgrading Hazelcast to 2.3, IPv6 is no longer supported in TCP join mechanism. [258]
- `ClassCastException` is thrown when the method `MultiMap.get()` is run. [256]
- Currently, the method `MultiMap.get(K key)` returns null if there are no values associated with the key. However, it should return an empty collection instead. [167]

Chapter 47

2.3

This section lists the new features, enhancements and fixed issues for 2.3 release.

47.0.7 New Features and Changes

The following are the new features and changes introduced with 2.3 release.

- Changed `hazelcast.max.operation.timeout` unit from seconds to milliseconds.
- Added `hazelcast.max.concurrent.operation.limit` property to be able to limit the number of concurrent operations that can be submitted to Hazelcast.
- Added `hazelcast.backup.redo.enabled` property to enable/disable redo for backup operations.
- Added MultiMap and Distributed ExecutorService statistics to Management Center application.
- `MigrationListener` has now an additional method to receive failed migration events; `void migrationFailed(MigrationE migrationEvent)`.
- `ItemEvent` has now an additional method returning Member firing that event; `public Member getMember()`.
- Improved out of memory (OOM) error detection and handling. Now it is possible to register a custom hook when OutOfMemoryError is thrown.
- Fixed some issues related to domain name handling and networking/join.
- During cluster merge after a network split-brain, merging side will now fire MERGING and MERGED before and after RESTARTING and RESTARTED LifecycleEvents.

47.0.8 Fixes

The following are the fixed issues and enhancements for 2.3 release.

- Operation exceptions occur randomly while the SQL predicates are stress tested. [263]
- `EOF` exception is thrown in Management Center. [252]
- The method `IMap.flush()` delegates map store exceptions to the caller instead of suppressing them. [250]
- Node appears to be in multiple clusters. [247]
- It would be nice to have Spring support in Runnable tasks. [244]
- MERGING and MERGED events should be fired during a split-brain merging process. [241]
- A problem occurs during a quick restart of master node. [235]
- Distributed `tryLock()` throws `NullPointerException` from lite member. [233]
- Using domain instead of IP address is problematic. [230]
- The method `MultiMap.lock()` blocks forever. [228]
- `SimpleMapTest` uses deprecated method `Hazelcast.getLoggingService()` and annoys WebLogic 12. [227]
- `ItemEvent` should have a `getMember` method to return the member that offered/polled an item. [226]
- The consumers do `queue.take()` and block until something is put on the queue. In Hazelcast 2.1.2, this was fine, but in 2.2, it causes a massive amount of WARN-level log spam. [225]

- The method `MultiMap.put()` is inconsistent when the key is locked by another node/thread. [223]
- FAILED event should be fired when migration fails for some reason. [222]
- `Runtime` exception is thrown when accessing to queue after client restarted. [220]
- Using the client, once it tries to access the cache while the cache is unavailable, the client cannot be used anymore after the server cache becomes available. [218]
- The methods `DistributedTask.get()` and `MultiTask.get()` throw `OperationTimeoutException`. [217]
- There are a couple of classes that do not implement `toString`, that makes the `Config.toString` message incomplete. [209]
- Hazelcast IMap ignores the size capacity. [188]
- Hazelcast resource adapter package (version 2.1.2) does not deploy on JBoss 7.1.1. [182]

Chapter 48

2.2

This section lists the new features, enhancements and fixed issues for 2.2 release.

48.0.9 New Features and Changes

The following are the new features and changes introduced with 2.2 release.

- Improved redo logs and added maximum call/operation timeout.
- Improved domain name handling; Hazelcast will use defined addresses/domain-names in TCP-IP Config as they are, without resolving an IP address.
- Added Cluster Health Check to Management Center application.

48.0.10 Fixes

The following are the fixed issues and enhancements for 2.2 release.

- The Hazelcast manual show a default config for WAN replication that includes the config item `<replication-impl>com.hazelcast.impl.wan.WanNoDelayReplication</replication-impl>`. This item appears to be used for configuring WAN replication implementation to be used. This item is not being used at all. Instead the replication implementation is hard coded in `WanReplicationService.java`. [152]
- It would be nice to have possibilities to instantiate native client from `java.util.Properties` and/or from classpath-located configuration file. [93]
- The method `ILock.isLocked()` should have the ability to implement a spin lock / wait for a lock release to perform non-blocking code. [39]

Chapter 49

2.1.3

The following are the fixed issues and enhancements for 2.1.3 release.

- Hazelcast IList evicts the items using the default map TTL. [196]
- Nodes with EC2 auto-discovery do not rejoin the cluster after a network outage. [195]
- The method `remove()` cannot be called on a Hazelcast Set iterator. [189]
- Topic does not receive messages when using Spring injection. [186]
- The method `IMap.set()` should not load data from map store. [185]
- `EntryListener` for native clients receives duplicate events. [183]
- Listeners do not work when used with locks and `remove`. [196]
- The package `hazelcast-spring` dependencies on Hibernate and MongoDB should be optional. [179]
- Backup entry count is smaller than the owned entries. [177]
- Queries on enum indexed types return an empty result set. [176]

Chapter 50

2.1.2

The following are the fixed issues and enhancements for 2.1.2 release.

- The method `containsKey()` does not work on a single node with the default near cache configuration. [174]
- The method Transactional `IMap.get(key)` causes the key to be added to map. [161]

Chapter 51

2.1.1

The following are the fixed issues and enhancements for 2.1.1 release.

- Item listener does not work when using client. [158]
- The method `IMap.remove(key)` returns incorrect object. [149]
- Map eviction policies are not documented. [148]
- When calling the method `IQueue.remove(object)`, the item listener added for that queue is not invoked. [146]
- List is not keeping the order in transactional context. [73]

Chapter 52

2.1

This section lists the new features, enhancements and fixed issues for 2.1 release.

52.0.11 New Features and Changes

The following are the new features and changes introduced with 2.1 release.

- Hazelcast now supports IPv6 addresses seamlessly.
- Added async backup support.
- Hazelcast now can be used as Spring Cache provider.
- Spring Dependency Injection support: Hazelcast can apply bean properties or to apply factory callbacks such as ApplicationContextAware, BeanNameAware or to apply bean post-processing such as InitializingBean, @PostConstruct like annotations while using Hazelcast distributed ExecutorService or DistributedTasks or more generally any Hazelcast managed object.
- Added persistence support with Spring-Data MongoDB and JPA integration.
- `Member.getUuid()` now will return UUID for node.
- Improved session clustering configuration.

52.0.12 Fixes

The following are the fixed issues and enhancements for 2.1 release.

- Hazelcast client breaks the lifecycle listeners. [130]
- The Spring bean attributes `lazy-init` and `scope` should be added. [118]
- Spring configuration namespace is missing `lock`. [116]
- `MaxSizeHeapPolicy` issue causing improper evictions and freezes. [110]
- `NullPointerException` is thrown by the method `tx.commit()`. [108]
- Hazelcast client does not shutdown properly if no connection can be established. [101]
- It would be nice to have “semi-asynchronous” replication for Hazelcast. [92]

Chapter 53

2.0.4

The following are the fixed issues and enhancements for 2.0.4 release.

- `NullPointerException` is thrown by the method `AbstractRecord.getLockCount()`. [166]
- There are "Connection refused" in Hazelcast 2.0.3. [140]
- Predicate value on the date field should not be converted from a String by the method `getRealObject()`.[135]
- Some IMap removed/evicted entries will go back when one cluster node crashes. [132]
- `Predicates.GreaterLesserPredicate`'s `doApply` operation is broken. [131]
- Incorrect selection by predicate with comparison of non-numerical fields and indexing problems. [98]
- Evicted/deleted entries should not be stored to the map store. [96]

Chapter 54

2.0.3

The following are the fixed issues and enhancements for 2.0.3 release.

- Increasing amount of stuck Hazelcast threads in weblogic; in a clustered environment and also when running a single instance. [160]
- Hyphens in the group names confuse the Management Center. It treats the group name as two separate tokens. [128]
- Split-brain merge handling causes the locks never to be acquired again. [127]
- `ConfigXmlGenerator` does not handle the TTL correctly when multicast joining mechanism is used. [127]
- Data is lost in a map when adding a key the second time after a new member joins the cluster. [117]
- Listeners are not called when a queue is changed within a transaction. [114]
- Programmatic session destruction does not lead to the destruction of the clustered session. [104]
- The name for `hazelcast.sessionId` should be configurable. [103]
- There is an issue in concurrent `queue.take()` and `txn.rollback()`.[99]

Chapter 55

2.0

This section lists the new features, enhancements and fixed issues for 2.0 release.

55.0.13 New Features and Changes

The following are the new features and changes introduced with 2.0 release.

- New Elastic Memory(Enterprise Edition Only): By default, Hazelcast stores your distributed data (map entries, queue items) into Java heap which is subject to garbage collection. As your heap gets bigger, garbage collection might cause your application to pause tens of seconds, badly effecting your application performance and response times. Elastic Memory is Hazelcast with off-heap memory storage to avoid GC pauses. Even if you have terabytes of cache in-memory with lots of updates, GC will have almost no effect; resulting in more predictable latency and throughput.
- Security Framework(Enterprise Edition Only): Hazelcast Security is JAAS based pluggable security framework which can be used to authenticate both cluster members and clients and do access control checks on client operations. With the security framework, take control of who can be part of the cluster or connect as client and which operations are allowed or not.
- Native C# Client(Enterprise Edition Only): Just like our Native Java Client, it supports all map, multimap, queue, topic operations including listeners and queries.
- Distributed Backups: Data owned by a member will be evenly backed up by all the other members. In other word, every member takes equal responsibility to backup every other node. This leads to better memory usage and less influence in the cluster when you add/remove nodes. The new backup system makes it possible to form backup-groups so that you can have backups and owners fall into different groups.
- Parallel IO: Number of socket selector threads can be configured. You can have more IO threads, if you have good number of CPU/cores and high-throughput network.
- Connection Management: Hazelcast 2.0 is more tolerant to connection failures. On connection failure it tries to repair it before declaring the member as dead. So now it is ok to have short socket disconnections... No problem if your virtual server migrates to a new host.
- Listeners such as migration, membership and map indexes can be added with configuration.
- New Event Objects: Event Listeners for Queue/List/Set/Topic were delivering the item itself on event methods. That's why the items had to be deserialized by Hazelcast Threads before invoking the listeners. Sometimes this was causing class loader problems too. With 2.0, we have introduced new event containers for Queue/List/Set and Topic just like Map has EntryEvent. The new listeners now receive ItemEvent and Message objects respectively. The actual items are deserialized only if you call the appropriate get method on the event objects. This is where we brake the compatibility with the older versions of Hazelcast.
- ClientConfig API: We had too many of factory methods to instantiate a HazelcastClient. Now all we need `isHazelcastClient.newHazelcastClient(ClientConfig)`.
- SSL communication support among cluster nodes.
- Distributed MultiMap value collection can be either List or Set.
- SuperClient is renamed to LiteMember to avoid confusion. Be careful! It is a member, not a client.

- New `IMap.set(key, value, ttl, TimeUnit)` implementation, which is optimized `put(key, value)` operation as `Set` does not return the old value.
- `HazelcastInstance.getLifecycleService().kill()` will forcefully kill the node. Useful for testing.
- `forceUnlock`, to unlock the locked entry from any node and any thread regardless of the owner.
- Enum type query support: As an example; `new SqlPredicate ("level = Level.WARNING")`

55.0.14 Fixes

The following are the fixed issues and enhancements for 2.0 release.

- [459], [471], [567], [574], [582], [629], [632], [646], [666], [686], [669], [690], [692], [693], [695], [698], [705], [708], [711], [714], [715], [719], [721], [722], [727], [728], [729], [730], [731], [732], [733], [735], [738], [739], [740], [741], [742], [747], [751], [752], [758], [759], [760], [761], [765], [767], [770], [773], [779], [781], [782], [783], [787], [795], [796]

Chapter 56

1.9.4.9

This section lists the new features, enhancements and fixed issues for the releases starting from 1.9.4.9 to 1.0.

- Added WAN Replication (synchronization of separate active clusters).
- Added Data Affinity (co-location of related entries) feature.
- Added EC2 Auto Discovery for your Hazelcast cluster running on Amazon EC2 platform.
- Implemented Distributed CountDownLatch.
- Implemented Distributed Semaphore implementation.
- Hazelcast distribution now contains HTML and PDF documentation besides Javadoc.
- Better TCP/IP and multicast join support. Handling more edge cases like multiple nodes starting at the same time.
- Memcache protocol: Better integration between Java and Memcache clients. Put from memcache, get from Java client.
- Monitoring Tool is removed from the project.
- Re-implementation of distributed queue:
  - Configurable backup count and synchronous backup.
  - Persistence support based on backing MapStore.
  - Auto-recovery from backing MapStore on startup.
- Re-implementation of distributed list supporting index based operations.
- Optimized `IMap.putAll` for much faster bulk writes.
- Added `IMap.getAll` for bulk reads which is calling `MapLoader.loadAll` if necessary.
- Added `IMap.tryLockAndGet` and `IMap.putAndUnlock` methods.
- Added `IMap.putTransient` API for storing only in-memory.
- Added `IMap.addLocalEntryListener()` for listening locally owned entry events.
- Added `IMap.flush()` for flushing the dirty entries into MapStore.
- Added `MapLoader.getAllKeys` API for auto-pre-populating the map when cluster starts.
- Support for minimum initial cluster size to enable equally partitioned start.
- Introduced graceful shutdown.
- Faster dead-member detection.
- Memcache interface support. Memcache clients written in any language can access Hazelcast cluster.
- RESTful access support, e.g. http://:5701/hazelcast/rest/maps/mymap/key1.
- Added split-brain (network partitioning) handling.
- Added LifecycleService API to restart, pause Hazelcast instances and listen for the lifecycle events.
- Added asynchronous put and get support for IMap via `IMap.asyncPut()` and `IMap.asyncGet()`.
- Added AtomicNumber API; distributed implementation of `java.util.concurrent.atomic.AtomicLong`.
- Significant performance gain for multi-core servers. Higher CPU utilization and lower latency.
- Reduced the cost of map entries by 50 percent.
- Better thread management. No more idle threads.
- Added queue statistics API and the queue statistics panel on the Monitoring Tool.

- Monitoring Tool enhancements. More responsive and robust.
- Hazelcast distribution now contains `hazelcast-all-<version>.jar` to simplify the JAR dependency.
- Sorted index optimization for map queries.
- Added Hazelcast Cluster Monitoring Tool.
- Added Partition API. Partition and key owner, migration listeners.
- Added `IMap.lockMap()` method.
- Added Multicast and TCP/IP join feature. Try multicast first, if not found, try TCP/IP.
- Added `Hazelcast.getExecutorService(name)` API. You can have separate named executor services. Do not let your big tasks blocking your small ones.
- Added `Logging` API. Build your own logging. or simply use Log4j or get logs as LogEvents.
- Added `MapStatistics` API. Get statistics for your Map operations and entries.
- Hazelcast client now automatically updates the member list. There is no need to pass the list to all members.
- Added the ability to start the cluster members evenly partitioned. Hence, no migration.
- Added Java clients for accessing the cluster remotely.
- Added Distributed Query for maps. Both Criteria API and SQL are supported.
- Added near cache feature for distributed maps.
- Added TTL (time-to-live) property for each individual map entry.
- Improved the put operation: `IMap.put(key,value, ttl, timeunit)`.
- Introduced the method `IMap.putIfAbsent(key,value, ttl, timeunit)`.
- Now, you can have multiple Hazelcast members on the same JVM. Introduced `HazelcastInstance` API.
- Better API based configuration support.
- Smoother data migration enabling better response times during joins.
- Persistence via Loader/Store interface for distributed map.
- Added Socket level encryption feature. Both symmetric and asymmetric encryption are supported.
- Added support for JMX.
- Added support for Hibernate second level cache provider.
- Added instance events for getting notified when a data structure instance (map, queue, topic, etc.) is created or destroyed.
- Added eviction listener: `EntryListener.entryEvicted(EntryEvent)`.
- Hazelcast is now fully “maven”ized.
- Added support for synchronous backups and configurable backup-count for maps.
- Added eviction support: Timed eviction for queues. LRU, LFU and time based eviction for maps.
- Added support for statistics/history for entries: create/update time, number of hits, cost.
- Implemented MultiMap structure. Similar to `google-collections` and `apache-common-collections`, but distributed and thread-safe.
- Now, you can `destroy()` the data structures when not needed anymore.
- Now, you can shutdown the local member using `Hazelcast.shutdown()`.
- Now, you can get the list of all data structure instances via `Hazelcast.getInstances()`.
- Full implementation of `java.util.concurrent.BlockingQueue`. Now, queues can have configurable capacity limits.
- Introduced Super Clients (a.k.a LiteMember): Members with no storage. If `-Dhazelcast.super.client=true` JVM parameter is set, that JVM will join the cluster as a ‘super client’ which will not be a ‘data partition’ (no data on that node) but will have super fast access to the cluster just like any regular member does.
- Added HTTP Session sharing support for Hazelcast Web Manager. Different web applications can share the same sessions.
- Added the ability to separate clusters by creating groups.
- Added `java.util.logging` support.
- Added the support for adding, removing and updating events for queue, map, set and list data structures.
- Introduced Distributed Topic for pub/sub messaging.
- Added integration with J2EE transactions via JCA complaint resource adapter.
- Added `ExecutionCallback` interface for distributed tasks.
- Introduced cluster-wide unique ID generator.
- Implemented Transactional Distributed Queue, Map, Set and List.
- Implemented Distributed Executor Service.

- Added support for multi member executions.
- Implemented key based execution routing.
- Added task cancellation support.
- Implemented Session Clustering with Hazelcast Webapp Manager.
- Added full TCP/IP clustering support.
- Introduced distributed implementation of `java.util.{Queue,Map,Set,List}`.
- Introduced distributed implementation of `java.util.concurrency.Lock`.
- Added support for retrieving cluster membership events.
- 1000+ commits 100+ bug fixes and several other enhancements.