Oracle Fusion Middleware 12c

Cloud Application Foundation

YouTube Video Series

ORACLE

Coherence Live Events

Harvey Raja

Consulting Member Technical Staff, Cloud Application Foundation

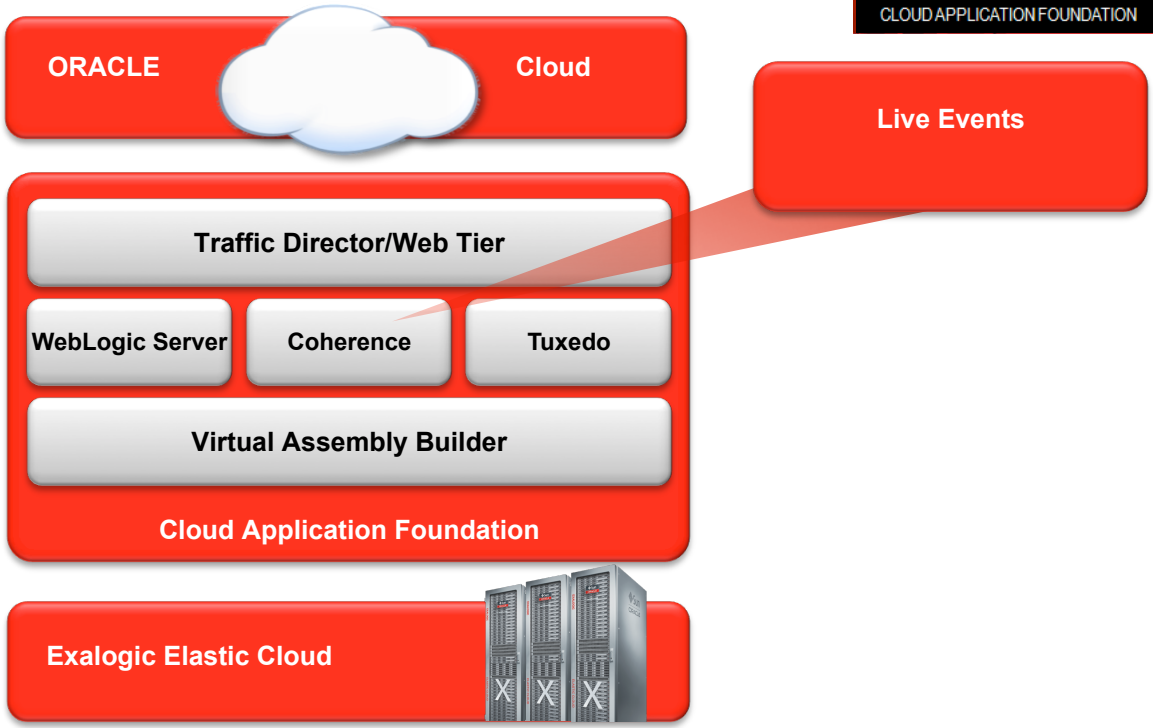
Oracle Coherence

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract.
It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Cloud Application Foundation

Coherence 12c Demonstration – Live Events

- Complete
- Open
- Integrated
- Best in Class
- On Premise – Private Cloud
- Public Cloud



Coherence Eventing Mechanisms

Existing

- MapListener
  - Client-side “post”-events
- Triggers
  - Server-side “pre”-events
- Backing Map Listeners
  - Server-side “post”-events
- Others
  - PartitionListener, MemberListener, ServiceListener

Coherence Eventing Mechanisms

Disadvantages

- Distinct registration mechanism per event type
- No commonality for client interfaces or events
- Are the events synchronously or asynchronously dispatched?

Live Event Model

ORACLE
CLOUD APPLICATION FOUNDATION

PartitionedCache

- EntryEvent
  - Data related
- EntryProcessorEvent
  - EntryProcessor invocation

PartitionedService

- TransferEvent
  - Partition redistribution
- TransactionEvent
  - Cross cache event

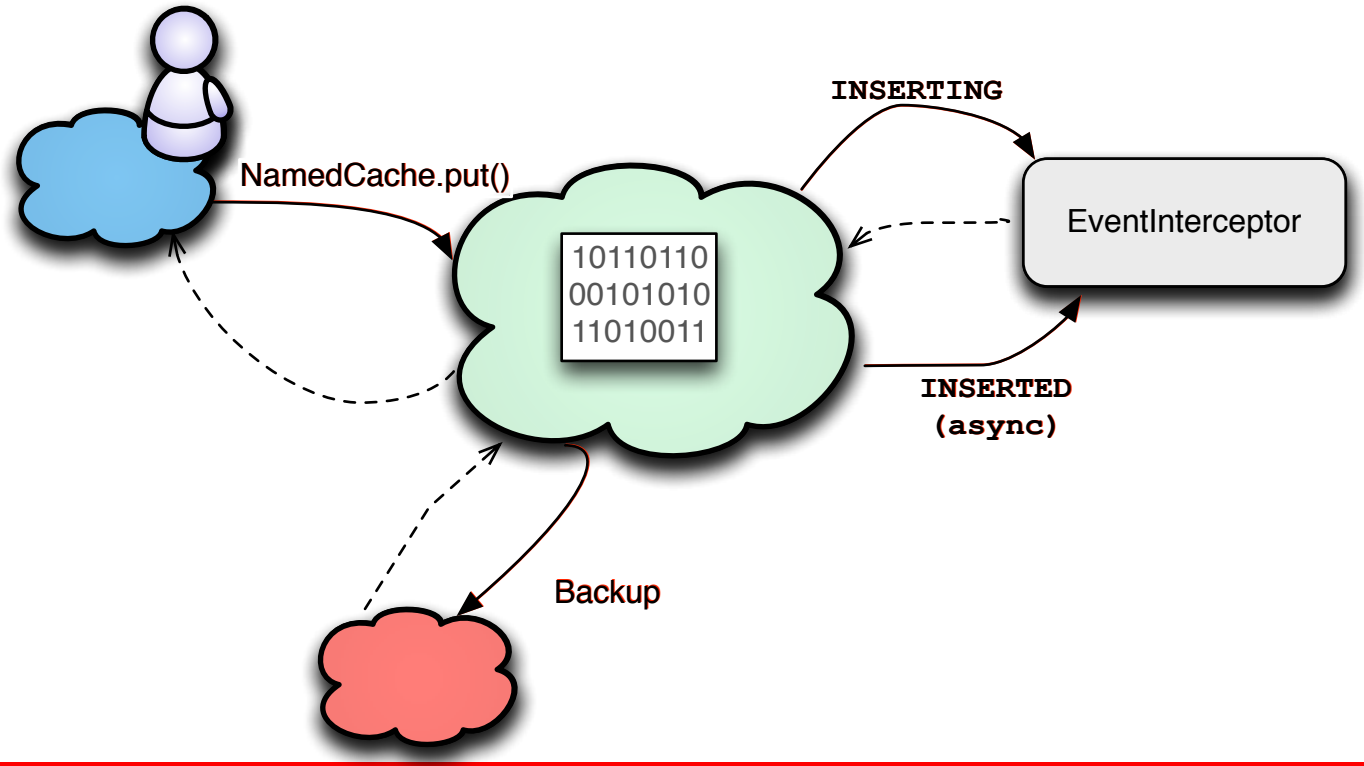
Data-Related Events

- BinaryEntry-based

```
public interface EntryEvent
        extends Event<EntryEvent.Type>
    {
    /**
     * Return the immutable Set of {@link BinaryEntry entries} on which the
     * action represented by this {@link EntryEvent} occurred.
     *
     * @return the Set of entries represented by this event
     */
    public Set<BinaryEntry> getEntrySet();


    public static enum Type
        {
        INSERTING, INSERTED, UPDATING, UPDATED, REMOVING, REMOVED
        }
    }
```

Data-Related Events



EntryProcessor-Related Events

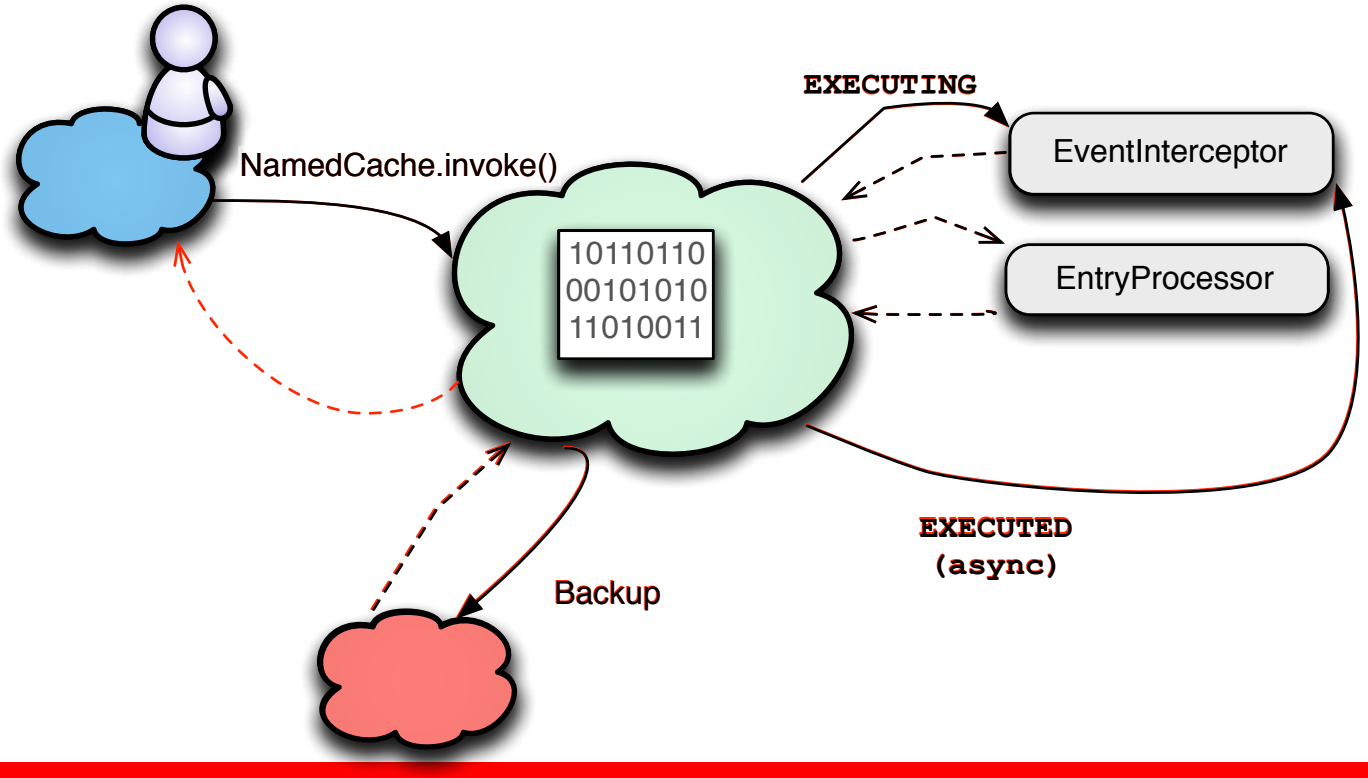
- BinaryEntry-based

```
public interface EntryProcessorEvent
        extends Event<EntryEvent.Type>
    {
    public Set<BinaryEntry> getEntrySet();

    /**
     * Return the {@link EntryProcessor} associated with this {@link
     * EntryProcessorEvent}.
     *
     * @return the entry processor associated with this event
     */
    public EntryProcessor getProcessor();

    public static enum Type
        {
        EXECUTING, EXECUTED
        }
    }
```

EntryProcessor-Related Events



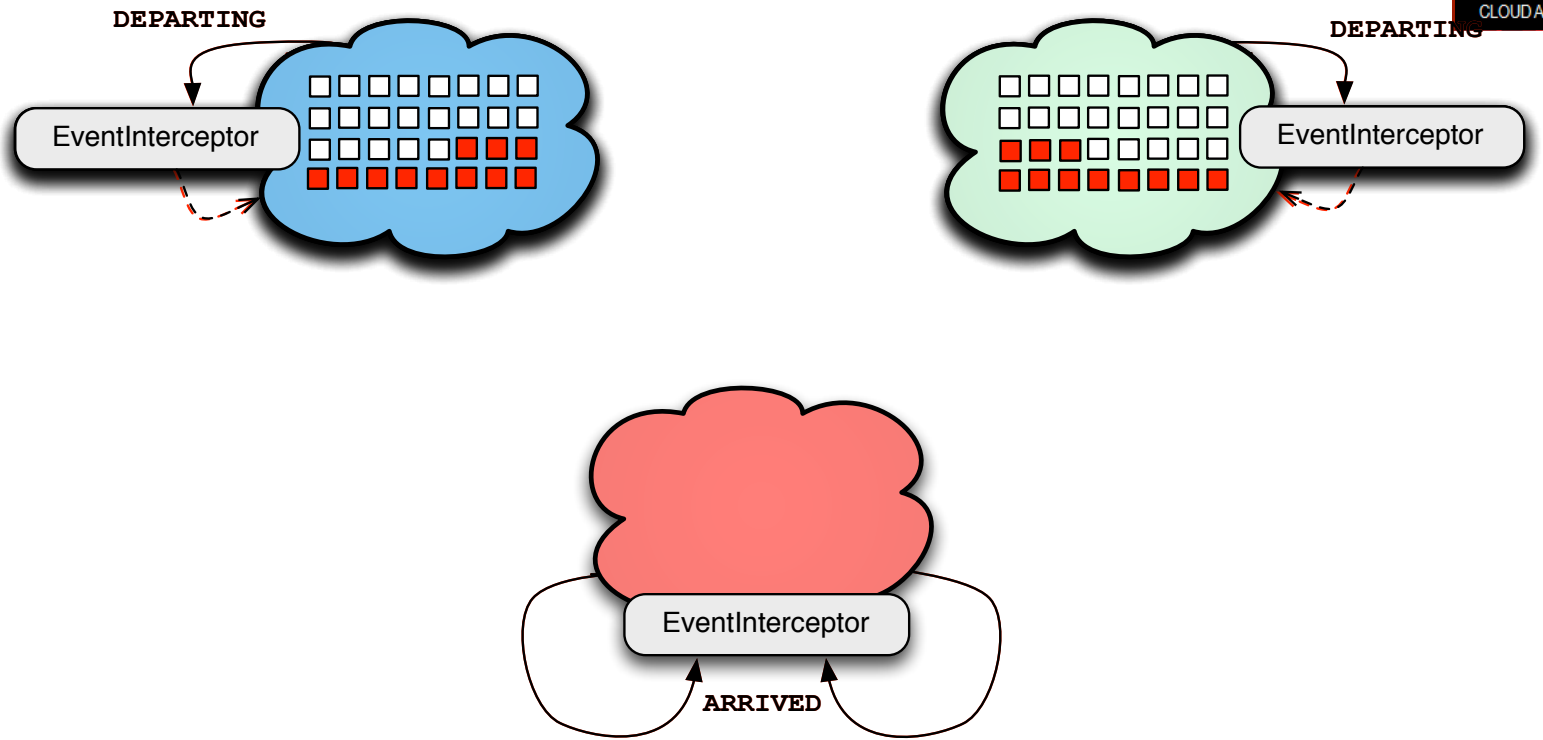
Transfer-Related Events

- BinaryEntry-based

```
public interface TransferEvent
        extends Event<EntryEvent.Type>
    {
    public int    getPartitionId();
    public Member getLocalMember();
    public Member getRemoteMember();
    /**
     * Return a map of cache names and associated set of read-only {@link
     * BinaryEntry entries} encapsulated in this {@link TransferEvent}. The
     * returned map and contained sets are immutable.
     *
     * @return a map of cache names and associated set of entries
     */
    public Map<String, Set<BinaryEntry>> getEntries();

    public static enum Type
        {
        DEPARTING, ARRIVED
        }
    }
```

Transfer-Related Events



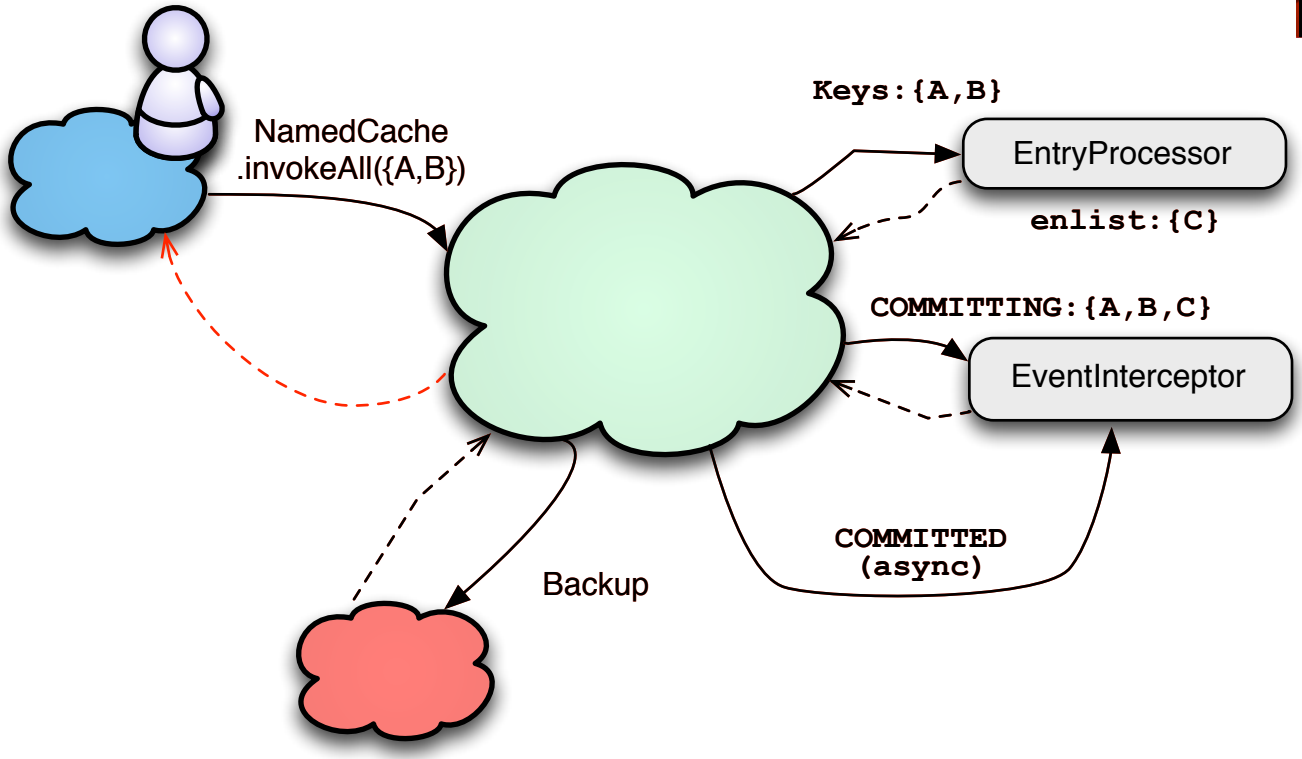
Transaction-Related Events

- All entries enlisted within a partition local transaction
- BinaryEntry-based

```
public interface TransactionEvent
        extends Event<TransactionEvent.Type>
    {
    /**
     * A set of {@link BinaryEntry entries} enlisted within this
     * transaction.
     *
     * @return a set of entries enlisted within this transaction
     */
    public Set<BinaryEntry> getEntrySet();

    public static enum Type
        {
        COMMITTING, COMMITTED
        }
    }
```

Transaction-Related Events



Registration

Declarative – Cache Configuration

- distributed-scheme
  - PartitionedService level interceptors: TransferEvent & TransactionEvent
  - PartitionedCache events for all caches
- cache-scheme-mapping
  - PartitionedCache level interceptors: EntryEvent & EntryProcessorEvent
  - Ties to caches that conform to the cache-name pattern
- Use Annotation & Generics to filter events

Registration

Declarative – Cache Configuration

```
<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
              xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-cache-config coherence-cache-config.xsd">
  <caching-scheme-mapping>
    <cache-mapping>
        <cache-name>vetod-events</cache-name>
        <scheme-name>ExamplesPartitionedPofScheme</scheme-name>
        <interceptors>
            <interceptor>
                <instance>
                    <class-name>com.tangosol.examples.events.CantankerousInterceptor</class-name>
                </instance>
            </interceptor>
        </interceptors>
    </cache-mapping>
  </caching-scheme-mapping>
  <caching-schemes>
    <distributed-scheme>
      <scheme-name>ExamplesPartitionedPofScheme</scheme-name>
      <interceptors>
          <interceptor>
              <instance>
                  <class-name>com.tangosol.examples.events.RedistributionInterceptor</class-name>
              </instance>
          </interceptor>
      </interceptors>
    </distributed-scheme>
  </caching-schemes>
</cache-config>
```

```
@Interceptor(identifier = "cantankerous",
        entryEvents = {Type.INSERTING, Type.INSERTED, Type.UPDATING, Type.UPDATED})
public class CantankerousInterceptor
        implements EventInterceptor<EntryEvent>
```

Registration

Programmatic

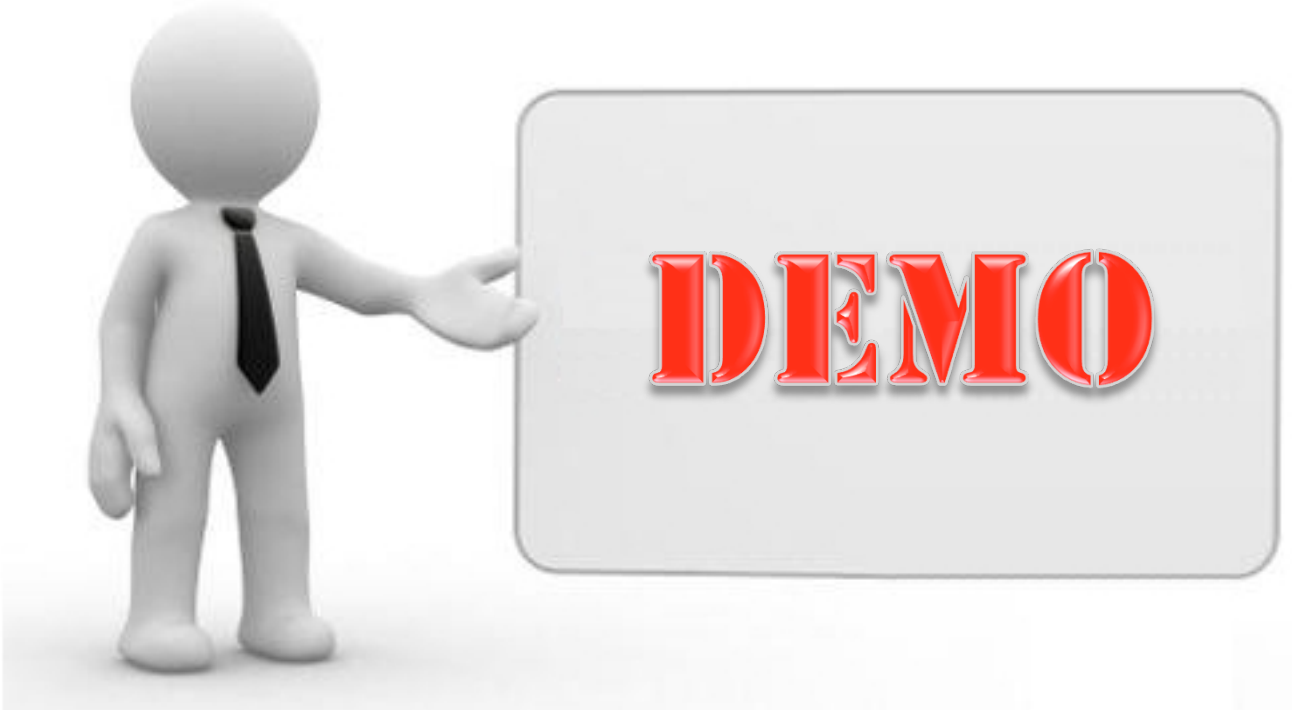
- ConfigurableCacheFactory holds an InterceptorRegistry:

```
ConfigurableCacheFactory ccf = CacheFactory.getConfigurableCacheFactory();
InterceptorRegistry      reg = ccf.getInterceptorRegistry();

reg.registerEventInterceptor("argumentative", new CantankerousInterceptor());
```

- Annotation and Generics on interceptor are honored

- Services and caches can be filtered by implementing:
  EventDispatcherAwareInterceptor



Join the Coherence Community

http://coherence.oracle.com

@OracleCoherence

/OracleCoherence

blogs.oracle.com/OracleCoherence

Group: Oracle Coherence Users

/OracleCoherence

coherence.oracle.com/display/CSIG
Coherence Special Interest Group

Oracle Fusion Middleware 12c

Cloud Application Foundation

YouTube Video Series

ORACLE

Coherence Live Events

Harvey Raja
Principal Member Technical Staff, Cloud Application Foundation
Oracle Coherence