
ORACLE®

ORACLE®

Coherence 3.3 Введение

Игорь Лукьянов
Oracle Presales Principal Consultant

2 июня 2007 года

Основные вопросы для обсуждения:

- Для чего необходим продукт?
- Что такое Coherence?
- Как работает Coherence?

Основные вопросы для обсуждения:

- Где используется Coherence?
- Преимущества использования продукта
- Взаимодействие Coherence и других продуктов

Основные вопросы для обсуждения:

- Внедрение продукта, потенциальные заказчики, цены
- Вопросы

Раздел

Для чего необходим продукт Coherence?

Мотивация использования продукта – решение задачи масштабируемости прикладных систем и приложений

Общие правила, упрощающие создание масштабируемых систем, эмпирически получены из опыта эксплуатации и программирования прикладных систем:

- **Создание stateless-приложений (Приложения stateful тяжело масштабировать)**
- **Использование паттернов программирования (пример: Master/Worker, MVC, и др.), позволяющих координировать топологическую привязку исполнения работы приложений к элементам масштабируемой системы**
- **Использование систем передачи сообщений для координации работы элементов масштабируемой системы**

Из общих наблюдений за работой масштабируемых приложений можно выделить основные факторы, влияющие на их работу:

Факторы – характеристики работы информационных систем.
Взгляд с точки зрения характеристик работы приложений.

- **Правило 3 элементов: RAS – Reliability, Availability, Serviceability (Надежность, Готовность, Готовность к обслуживанию)**
- **Правило 4 элементов: RASP - Reliability, Availability, Scalability, Perfomance (Надежность, Готовность, Масштабируемость, Производительность)**
  - **+ Manageability (Управляемость).**
- **RAS связан с uptime системы – временем работы (оригинально появился под влиянием производителей mainframe систем)**
- **RASP связан и с временем работы системы, а также с Scalability Perfomance системы и характерен для распределенных систем.**

Reliability Надежность

Надежность исчисляется процентом времени от общего времени работы приложения, в течение которого приложение в состоянии правильно исполнится (Приложение может быть доступно и все же ненадежно, если оно не в состоянии правильно исполнить прикладную обработку данных. Пример – сети мобильных телефонов).

Availability Готовность

Готовность вычисляется процентом времени, в течение которого приложение работает к линейному времени. 100 % готовность – безостановочная работа приложения в режиме 24x7.

Scalability Масштабируемость

Масштабируемость определяется способностью приложения обработать большие нагрузки. Приложение показывает линейную масштабируемость, если максимальное количество загрузки, которое приложение может выдержать, непосредственно пропорционально ресурсам аппаратных средств на которых исполняется приложение. (Пример: линейная масштабируемость – 2 сервера с приложением обрабатывают 2000 запросов в секунду, 10 серверов – 10000 запросов).

Perfomance Производительность

Производительность – величина обратная времени ожидания исполнения приложения (latency time). Чем больше время ожидания исполнения, тем меньше производительность. Производительность приложения конечна и уменьшается, если нам необходимо учитывать другие факторы исполнения приложения (Пример – наличие инфраструктуры приложения, обеспечивающей высокую готовность и надежность приложения).

Serviceability Готовность к обслуживанию (эксплуатационная надежность)

Эксплуатационная надежность определяется удобством и возможностью вносить оперативные изменения в работу приложения без серьезного влияния на готовность приложения. (Пример: синхронизация состояния entity объекта и базы данных).

Введение failover-конфигураций прикладных систем существенно снижает влияние данного фактора.

Manageability Управляемость

Управляемость определяется возможностью достигать нужных параметров работы приложения в связи с информацией, которую приложение или прикладная система предоставляет для этих целей.

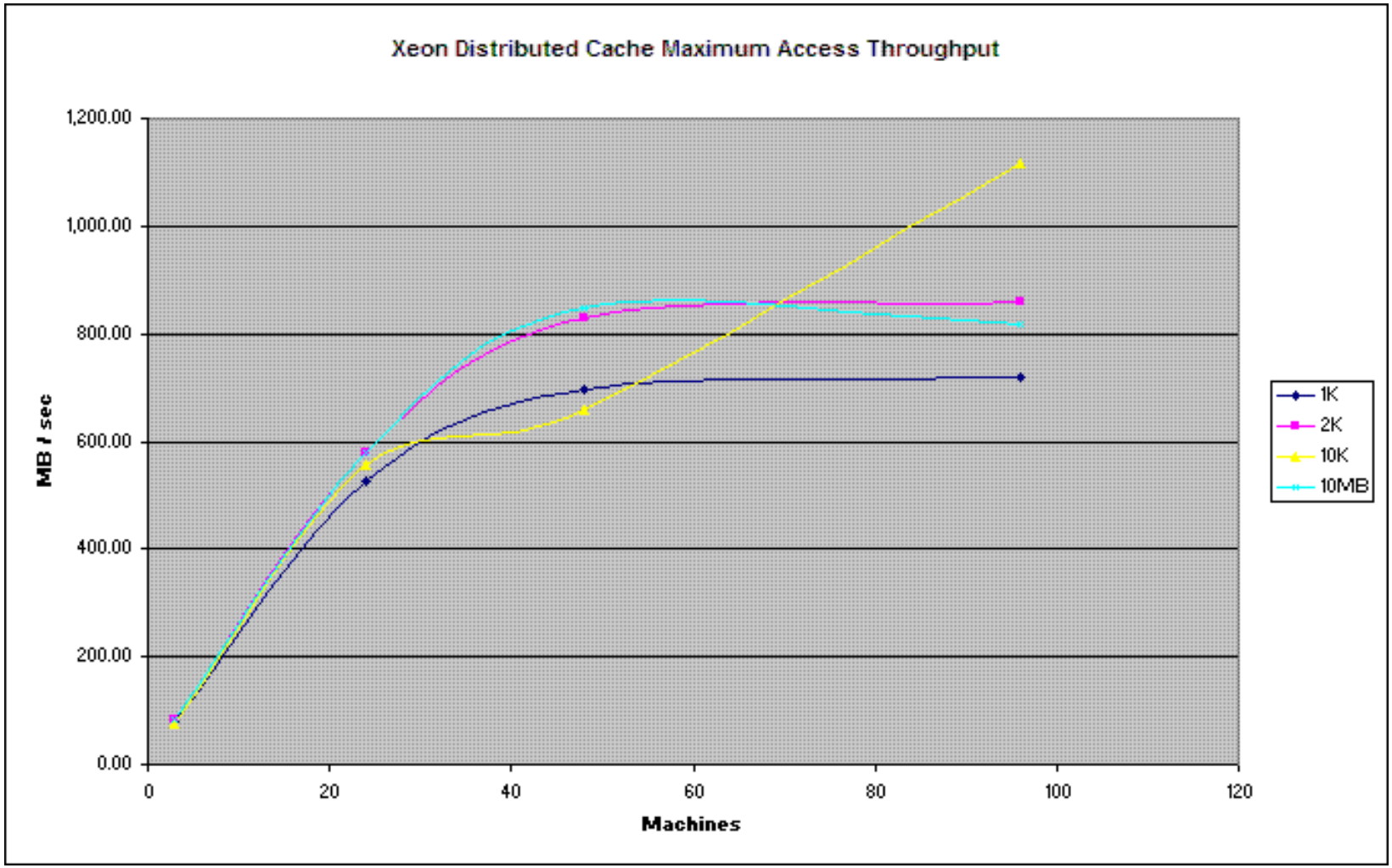
Из общих наблюдений за работой масштабируемых приложений можно выделить основные факторы, влияющие на их работу:

- Откуда прибывают данные к рабочему процессу (процессам) приложения
- Требует ли работа приложения копирования или дублирования данных на время исполнения из источников данных
- Процессируются ли данные в одном месте или в разных местах при использовании приложения
- Используется ли кэширование данных для ускорения работы приложения
- Существуют ли SPOF (single points of failure) или bottlenecks для работы масштабируемого приложения
- Приложение может быть простым и нересурсоемким, а данные слишком востребованными
- Ручное управление данными/службами может быть очень ресурсоемким

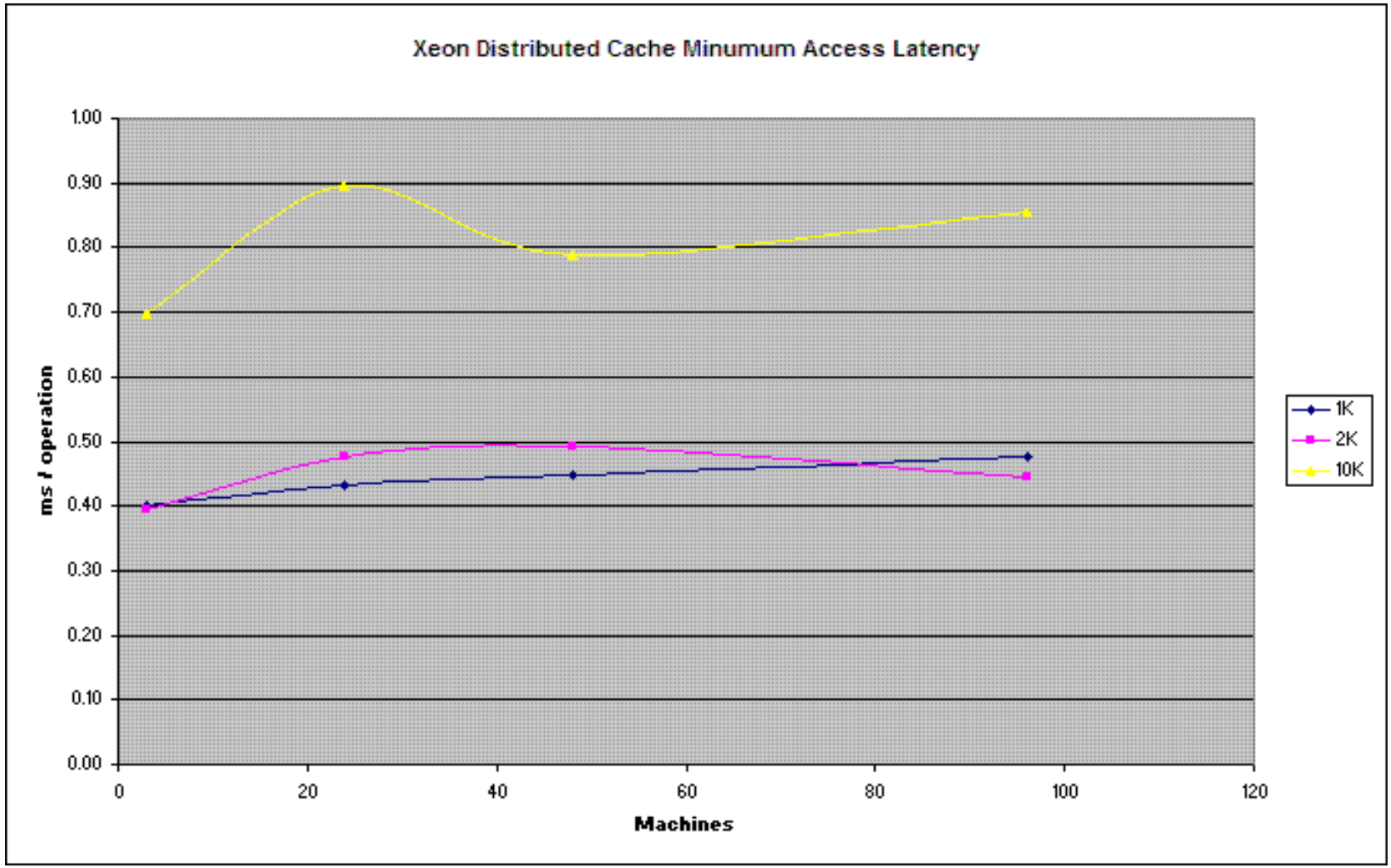
Coherence, как продукт необходим и предназначен для

- Для решения задач связанных с проблемами указанными выше и увеличения Scalability-Perfomance приложений в многозвенной архитектуре и предназначен продукт Coherence.
- Увеличения жизни источников данных и архитектур многозвенных систем
- Для создания XPS (extreme processing systems) и XTS (extreme transactional systems) систем обработки данных и приложений

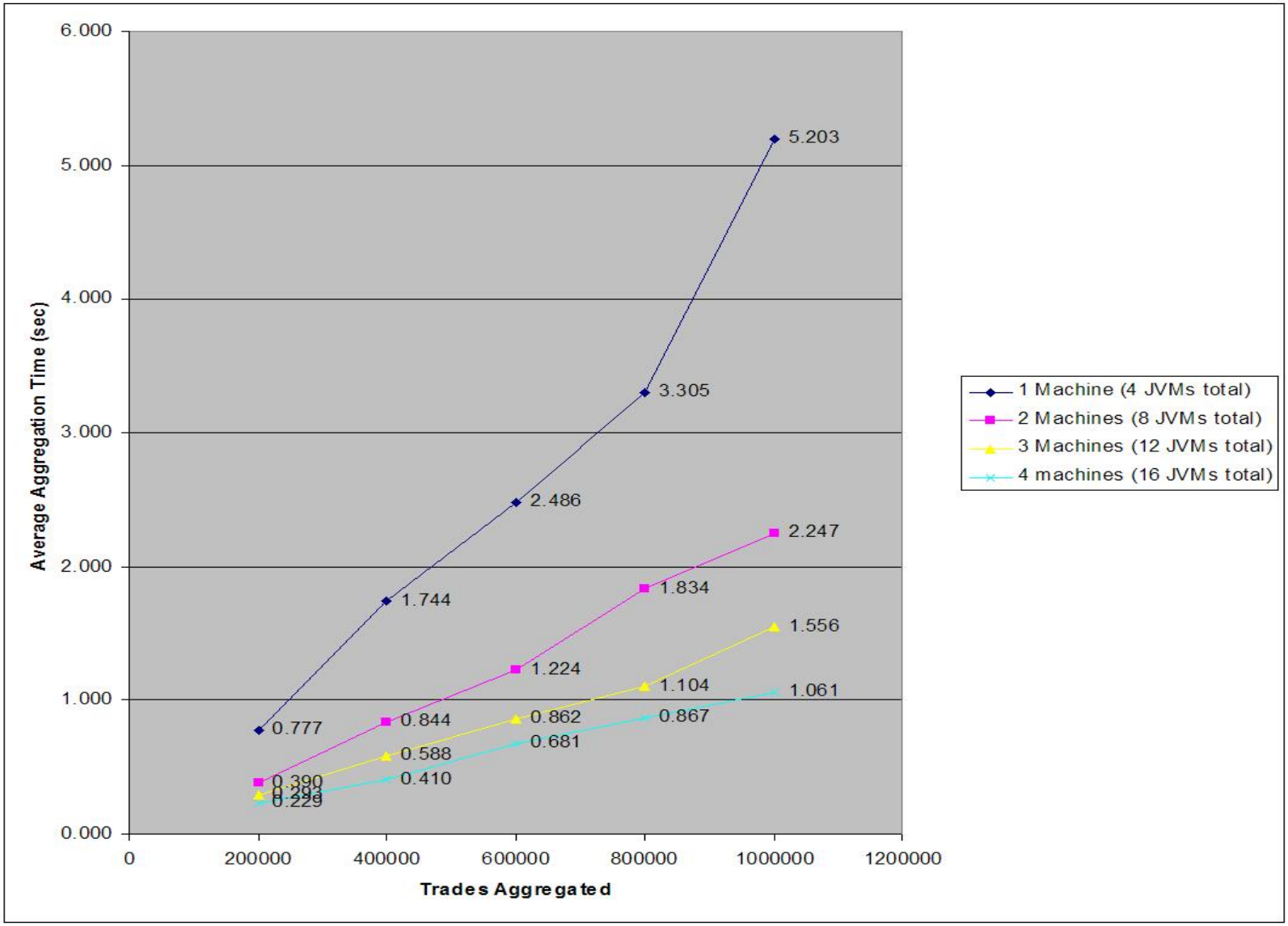
При использовании Coherence



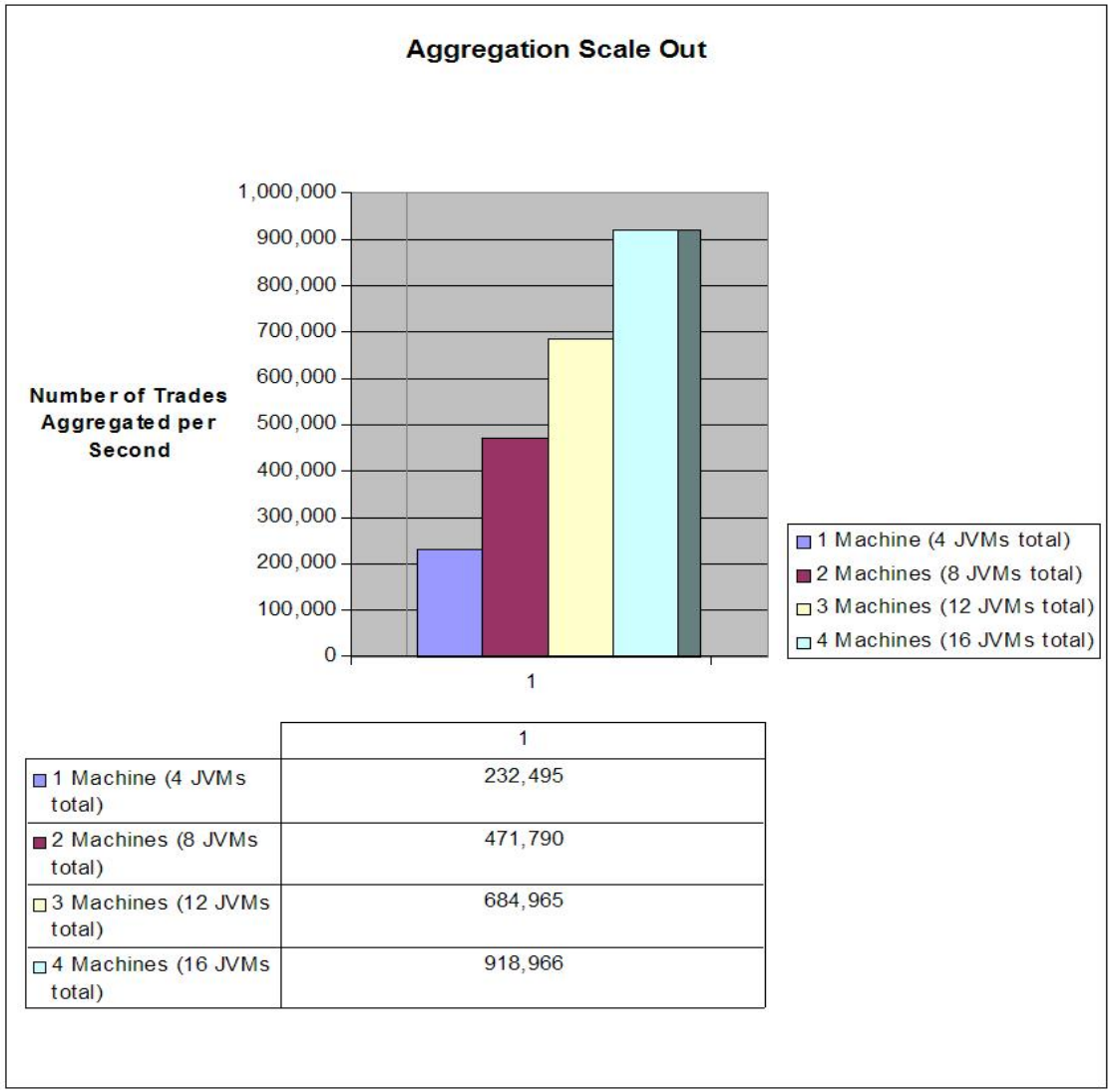
При использовании Coherence



При использовании Coherence



При использовании Coherence



	1
1 Machine (4 JVMs total)	232,495
2 Machines (8 JVMs total)	471,790
3 Machines (12 JVMs total)	684,965
4 Machines (16 JVMs total)	918,966

Раздел

Что такое Coherence?

Персоналии:

Cameron Purdy

- **Президент и основатель Tangosol, Incorporated (Coherence – продукт компании Tangosol).**
- **Автор ряда Java и XML спецификаций.**
- **Лидер создания спецификации JCache (JSR-107) и Work Manager (JSR-237).**

From Oracle – Tangosol FAQ paper:

”On March 23, 2007, Oracle announced its plan to acquire Tangosol, a leading provider of reliable in-memory data grid software. Tangosol will be the latest addition to the Oracle Fusion Middleware product family.”

Сделка завершилась в конце апреля.

Персоналии:

Shaun Smith

- **Principal Product Manager of Oracle TopLink в компании Oracle Corporation.**
- **Свыше 10 лет работы в сфере объектно-ориентированного программирования, объектно-реляционных технологий и объект-XML технологий мапирования.**
- **Лидер проекта Eclipse Dali JPA Tools**
- **Лидер проекта разработки экосистемы на платформе Eclipse Java Persistence (Eclipse Link).**

В результате приобретения Tangosol Oracle расширил свою платформу Fusion Middleware компонентом Coherence – основным компонентом, предназначенным для создания надежных масштабируемых кластеризованных приложений. Термин кластеризация в данном случае используется в том смысле, что приложение исполняется более чем на одном сервере для целей масштабируемости и надежности. Coherence обеспечивает работу приложений в режиме кластеризации для достижения максимального высокого уровня RASP (reliability, availability, scalability, perfomance) - надежности, готовности, масштабируемости и производительности. Это достигается за счет возможности кластеризации объектов и данных в используемом приложении. В самом простом смысле это достигается за счет того, что объекты и данные используемые в приложениях доступны всем серверам на которых они исполняются.

Coherence - Cluster-based Data Management Solution for Applications - кластер-ориентированное решение управления данными для приложений

Или.....

… Distributed Memory Data Management Solution (aka: Data Grid) – решение управления данными, работающее на основе распределенной памяти (Data Grid).

Используя Coherence компоненты Oracle Fusion Middleware и прикладные системы (приложения), работающие на этой платформе приобретают возможность их эксплуатации в оптимальном соотношении с точки зрения характеристик исполнения приложений RASP, Serviceability, Manageability.

Архитектура Coherence базируется на концепции:

Failures are always about to occur!

С этой точки зрения Coherence, как продукт хорошо согласуется с концепцией Failover кластеров, к которым относятся и кластера на основе Oracle Application Server 10G для Java объектов.

Coherence – это кластеризация приложений. Кластеризация в Coherence имеет специфические специализированные черты (таким образом это кластеризация в узком понимании, ориентированная на приложения):

Набор процессов (участников), работающих вместе. Нет masters/workers разделения обязанностей.

Участники полностью равнозначны и имеют одинаковые функции и задачи. Все-таки участники могут быть запрошены для выполнения специфических задач (являться Grid агентами).

Не существует никаких централизованных служб или registry для регистрации служб или данных

Не существует SPOF (single points of failures) в архитектуре

Не существует bottlenecks в архитектуре

Данными для манипуляции в Coherence могут являться:

Любые сериализуемые объекты (которые можно записать или сохранить в бинарной форме).

Не расширяются за счет proprietary классов.

Полная возможность взаимодействия данных Coherence с Java и .Net (вскоре с C++).

Нет байт-код инструкций по манипуляции с данными

Не применяется к объектам реляционной модели, объектно-реляционному мапированию, SQL и т.д.

Объекты: POJO и PONO (вскоре POCO) – plain objects для Java, .Net, C++

Различные топологии для Ваших данных

Простой API (Application Programme Interface) для всех данных в независимости от топологий.

Что Вы можете делать с данными:

Распределять объекты, мапировать их, кэшировать их.

Управлять событиями, связанными с изменением данных в реальном времени, создавать listener’ы

Использовать параллельные запросы к данным и индексирование данных

Использовать процессирование данных и служебные агенты (встроенные черты для Grid)

Обозревать данные

Агрегировать данные

Отслеживать целостность данных, сессии по работе с данными

Coherence - Cluster-based Data Management Solution for Applications - кластер-ориентированное решение управления данными для приложений

Или.....

… Distributed Memory Data Management Solution (aka: Data Grid) – решение управления данными, работающее на основе распределенной памяти (Data Grid).

А также.....

Решение для управления данными.....

Coherence как решение для управления данными:

Отвечает за кластеризацию и управление службами и данными в информационной системе, включая и разделяемый доступ к данным (partitioning).

В идеале инженеры, разработчики и администраторы не должны в распределенном приложении:

Создавать дизайн, специфицировать и кодировать в приложении, как осуществляется разделяемый доступ к данным.

Обрабатывать исключительные ситуации, связанные с управлением (это задача приложения или системы).

Управлять кластером вручную или на уровне кодирования его работы.

Останавливать работу приложения для добавления новых ресурсов

или изменения схемы разделенного доступа к данным.

Использовать “консоли” для восстановления или масштабирования системы.

Если количество элементов управления растет к бесконечности – стоимость управления на элемент должна стремиться к нулю и при этом логирование работы элементов системы/приложения не должно страдать.

...и

Сама технология кластеризации должна быть невидима и незаметна в вашем приложении (аналог – использование кэши второго уровня в процессорном модуле)

Coherence - Cluster-based Data Management Solution for Applications - кластер-ориентированное решение управления данными для приложений

Или.....

… Distributed Memory Data Management Solution (aka: Data Grid) – решение управления данными, работающее на основе распределенной памяти (Data Grid).

А также.....

Решение для управления данными.....

И еще....

Coherence ориентирован на приложения.

Coherence ориентирован на приложения:

Coherence не требует для своей работы наличия контейнера или сервера

С точки зрения компоновки является простой библиотекой. Все-таки для интеграции с конкретными продуктами требует добавления необходимых библиотек (для примера: spring, web, hipernate, jta и т.д.)

Coherence не имеет зависимостей от внешнего программного обеспечения или от Open Source.

Coherence практически аморфен к работе класслоадера для JAR файлов (или процесса загрузки для DLL).

Может быть встроен в другое программное обеспечение или работать автономно как отдельное приложение

Работает, где Java SE/EE или .Net работает

Не имеет привязки к конкретным архитектурным паттернам программирования

Раздел

Как работает Coherence?

Как работает Coherence? Взгляд со стороны приложения....

Закластеризуем Hello World….

Обычный класс.....

```
package dongdaemoon;
public class HelloWorld {
        public static void main(String[ ] args) {
       System.out.println("Hello World");
    }
}
```

Работаем с Coherence..... Входит....

```
public static void main(String[] args)
throws IOException {
NamedCache nc = CacheFactory.getCache("test");
nc.put("message", "Hello World");
System.out.println(nc.get("message"));
System.in.read();
}
```

Работаем с Coherence..... Входит....

```
public static void main(String[] args)
throws IOException {
```

Joins / Establishes a cluster

```
NamedCache nc = CacheFactory.getCache("test");
```

Places an Entry (key, value) into the Cache "test"

```
nc.put("message", "Hello World");
```

Retrieves the Entry from the Cache. Displays it.

```
System.out.println(nc.get("message"));
```

Keep the application (and Cluster) from terminating

```
System.in.read();
}
```

Работаем с Coherence..... И выходит....

```
   public static void main(String[] args) throws
IOException {
NamedCache nc = CacheFactory.getCache("test");
System.out.println(nc.get("message"));
}
```

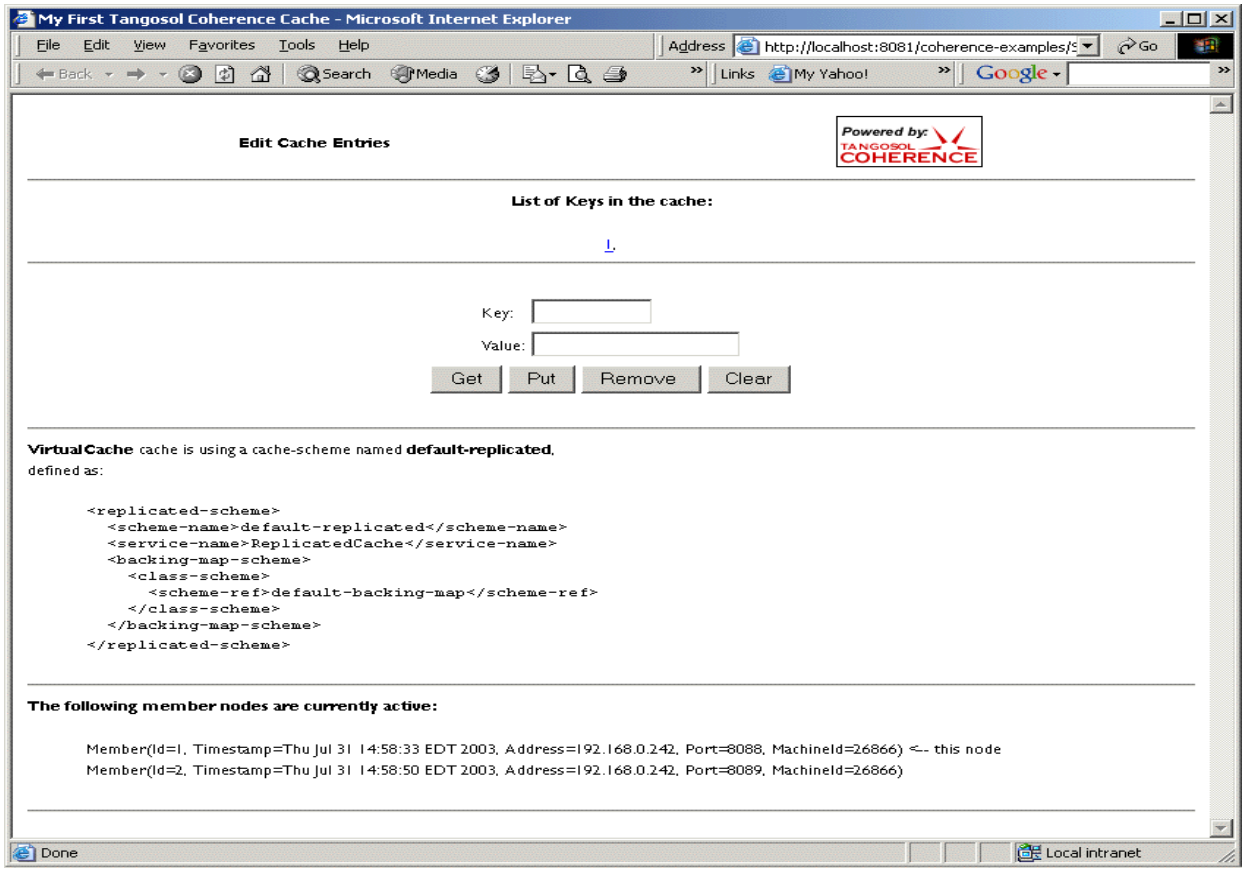
Работаем с Coherence..... И выходит....

```
   public static void main(String[] args) throws
IOException {
Joins / Establishes a cluster
NamedCache nc = CacheFactory.getCache("test");
Retrieves the Entry from the Cache. Displays it
System.out.println(nc.get("message"));
}
```

... Запускаем столько приложений сколько хотим и с разных инстансов.Они будут использовать кластеризуемые данные из общей Jcache кэши.

Как работает Coherence? Взгляд со стороны приложения....

Вызов Coherence из jsp-файла: через создание стандартного web-приложения и деплоймента на сервер приложений.



Запуск тестового режима Coherence

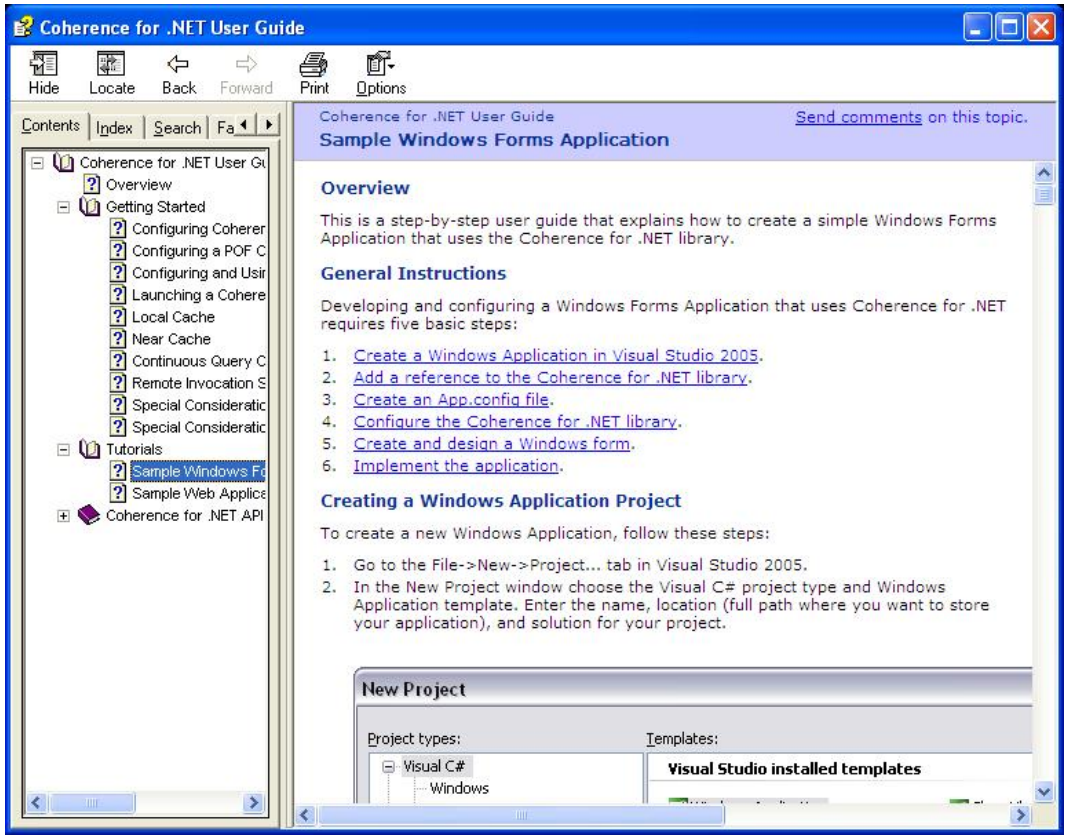
```
java -cp ./lib/coherence.jar;./lib/tangosol.jar;./examples/java
-Dtangosol.coherence.cacheconfig=./examples/config/explore-config.xml
com.tangosol.examples.explore.SimpleCacheExplorer
```

Запуск тестового режима Coherence

- Oracle Coherence Version 3.3/387 Grid Edition: Development mode Copyright (c) 2000-2007 Oracle. All rights reserved. 2007-05-23 11:53:40.390 Oracle Coherence GE 3.3/387 (thread=main, member=n/a): Loaded cache configuration from file "C:\coherence\examples\config\explore-config.xml" 2007-05-23 11:53:40.890 Oracle Coherence GE 3.3/387 (thread=Cluster, member=n/a): Service Cluster joined the cluster with senior service member n/a 2007-05-23 11:53:44.140 Oracle Coherence GE 3.3/387 (thread=Cluster, member=n/a): Created a new cluster with Member(Id=1, Timestamp=2007-05-23 11:53:40.687, Address=192.168.0.204:8088, MachineId=26828, Location=process:3976@localhost, Edition=Grid Edition, Mode=Development, CpuCount=2, SocketCount=1) UID=0xC0A800CC00000112B9A4BE4F68CC1F98 2007-05-23 11:53:44.203 Oracle Coherence GE 3.3/387 (thread=ReplicatedCache, member=1): Service ReplicatedCache joined the cluster with senior service member 1 Command: help clear get keys info put quit remove

Как работает Coherence? Взгляд со стороны приложения....

Coherence 3.3 для .Net



Чем Coherence не является?

Это не работающая в оперативной памяти база данных, как TimesTen, хотя также используется для реализации систем eXtreme Transaction Processing (XTP).

… Однако... Вы можете:

Осуществлять запросы в параллельном режиме к данным в Coherence, естественно не в реляционном стиле запросов (это не RDBMS)

Использовать запросы подобные SQL запросам

Осуществлять индексирование данных (как в базе данных)

Исполнять вещи, подобные сохраненным процедурам в базе данных

Использовать real-time view, подобные Materialized Views в базе данных.

Чем Coherence не является?

Это не служба обмена сообщениями, как JMS API …
Однако... Вы можете:

Использовать Events и Listeners для вставки, обновления и стирания данных.

Использовать Agents для контроля за изменением данных

Использовать Filters для фильтрации событий, связанных с изменением данных.

Чем Coherence не является?

Это не просто объектная кэш...

Данные в кэши устаревают или их можно обнулить.

Потребители (приложения) часто отключают кэш.

В Coherence этого сделать нельзя. Данные в Coherence обновляются все время на всех инстансах, где локализованы объекты.

В Coherence организация данных основана на надежной кластеризующей технологии. Это устойчиво, надежно и очень доступно.

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Надежная эластичная масштабируемая технология кластера в Coherence дает возможность разработчикам легко кластеризовать stateful приложения таким образом, что они могут динамически и надежно совместным образом использовать данные, обеспечивать обслуживание и отвечать на события, чтобы масштабированные решения могли удовлетворять деловой спрос.

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Кластеризация в Coherence отлична от других архитектурных реализаций.

Поддерживается симметризация объектов на инстансах все время.

Симметризация объектов осуществляется с максимально возможной скоростью.

Процесс кластеризации не имеет SPOF (single points of failure).

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Механизм кластеризации не использует никакого механизма или общей точки регистрации объектов и инстансов объектной кэши.

Гарантировано, что все элементы симметризуются и имеют одинаковый уровень значимости в кластере и сообща поддерживают синхронность обновления других элементов.

Гарантировано, что не происходит никакого выбора приоритета обновления элементов и выбора принадлежности элемента к кластеру.

Как работает Coherence? Взгляд со стороны архитектурной реализации....

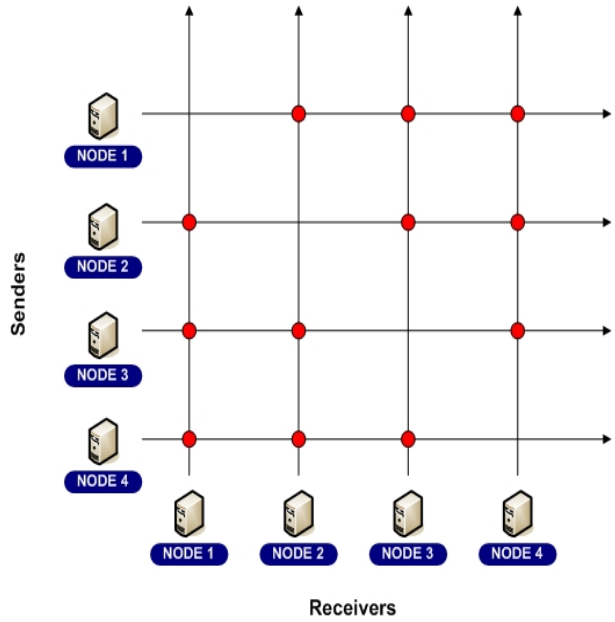
В 2002 году Oracle Corporation предложил концепцию data fabric.

Используя эту концепцию в Coherence реализована служба разделяемого управления данными.

Forrester Research именовал комбинацию виртуализации данных, прозрачной и распределенной интеграции информационной системы, основанную на службе разделяемого управления данными Coherence, как information fabric.

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Термин fabric (ткань) происходит из схемы 2-мерной иллюстрации реализации соединительных проводов, получившей название switched fabric.



Цель архитектуры fabric – все точки в пределах fabric имеют прямые соединительные провода со всеми другими точками.

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Information fabric, или как более частный упрощенный случай data fabric, названный data grid использует концепцию switched fabric, как основу для управления данными в распределенной информационной среде. Фактически, в общем виде data grid является основой для реализации динамической архитектуры информационной сети – DMA (Dynamic Mesh Architecture).

Coherence автоматически и динамически формирует надежную, все более эластичную switched fabric, составленную из любого числа серверов в пределах среды сетки (data grid).

Как работает Coherence? Взгляд со стороны архитектурной реализации....

При использования данной архитектуры:

Составная пропускная способность data grid прямо пропорциональна числу серверов

Емкость данных в памяти и возможность индексирования данных прямо пропорциональны числу серверов в сетке

Составная пропускная способность ввода-вывода дисков и накопителей в сетке прямо пропорциональны количеству серверов в сетке

Количество дисковой памяти и вероятность ее переполнения прямо пропорциональны количеству серверов в сетке

Устойчивость data fabric увеличивается при расширении сетки, поскольку каждая из точек ответсвенна за 1/n failover data grid

Если data fabric обслуживает клиентов типа систем торговли, то агрегированное число клиентов прямо пропорционально количеству узлов в data grid.

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Механизм кластеризации в Coherence – это об установлении консенсуса между элементами, это о согласии.

Если каждый элемент кластера всегда знает о существовании всех других элементов:

Мы можем осуществлять partitioning и load balance данных и служб в кластере.

Мы экономим ресурсы системы поскольку у нас нет необходимости держать все установленные TCP/IP коннекции в системе открытыми.

Любой элемент кластера взаимодействует с любым другим элементом кластера (peer-to-peer взаимодействие)

Кластер в течение его активного состояния может динамически масштабироваться к любому размеру.

Типы используемой кэши

Numerical Terms:

JVMs = number of JVMs

DataSize = total size of cached data (measured without redundancy)

Redundancy = number of copies of data maintained

LocalCache = size of local cache (for near caches)

Типы используемой кэши

	Replicated Cache	Optimistic Cache	Partitioned Cache	Near Cache backed by partitioned cache	VersionedNearCache backed by partitioned cache	LocalCache not clustered
Topology	Replicated	Replicated	Partitioned Cache	Local Caches + Partitioned Cache	Local Caches + Partitioned Cache	Local Cache
Fault Tolerance	Extremely High	Extremely High	Configurable [4] Zero to Extremely High	Configurable [4] Zero to Extremely High	Configurable [4] Zero to Extremely High	Zero
Read Performance	Instant [5]	Instant [5]	Locally cached: instant [5] Remote: network speed [1]	Locally cached: instant [5] Remote: network speed [1]	Locally cached: instant [5] Remote: network speed [1]	Instant [5]
Write Performance	Fast [2]	Fast [2]	Extremely fast [3]	Extremely fast [3]	Extremely fast [3]	Instant [5]
Memory Usage (Per JVM)	DataSize	DataSize	DataSize/JVMs x Redundancy	LocalCache + [DataSize / JVMs]	LocalCache + [DataSize/JVMs]	DataSize
Memory Usage (Total)	JVMs x DataSize	JVMs x DataSize	Redundancy x DataSize	[Redundancy x DataSize] + [JVMs x LocalCache]	[Redundancy x DataSize] + [JVMs x LocalCache]	n/a
Coherency	fully coherent	fully coherent	fully coherent	fully coherent [6]	fully coherent	n/a
Locking	fully transactional	none	fully transactional	fully transactional	fully transactional	fully transactional
Typical Uses	Metadata	n/a (see Near Cache)	Read-write caches	Read-heavy caches w/ access affinity	n/a (see Near Cache)	Local data

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Coherence Management Framework.

Существует возможность доступа к Coherence Mbeans через JMX-консоль:

java -cp jmxri.jar;jmxtools.jar;coherence.jar

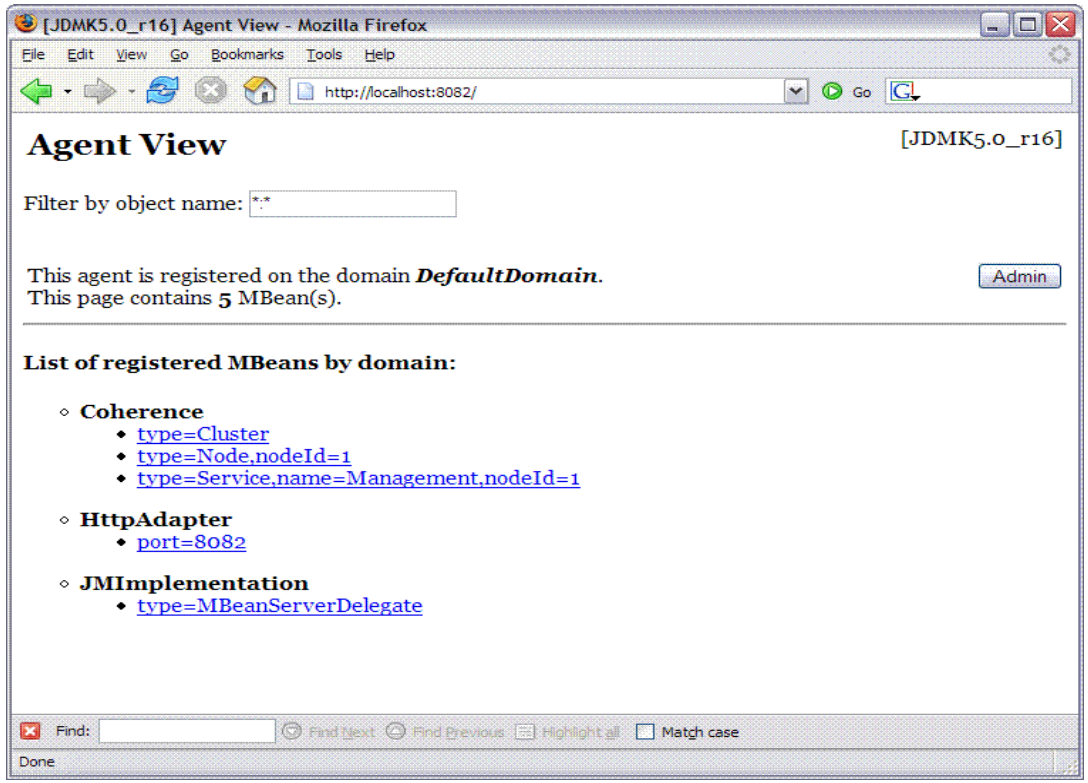
-Dtangosol.coherence.management=all

-Dtangosol.coherence.management.remote=true

com.tangosol.net.CacheFactory

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Существует возможность доступа к Coherence Mbeans через JMX-консоль:



Как работает Coherence? Взгляд со стороны архитектурной реализации....

Или через Jconsole J2SE:

java -Dcom.sun.management.jmxremote

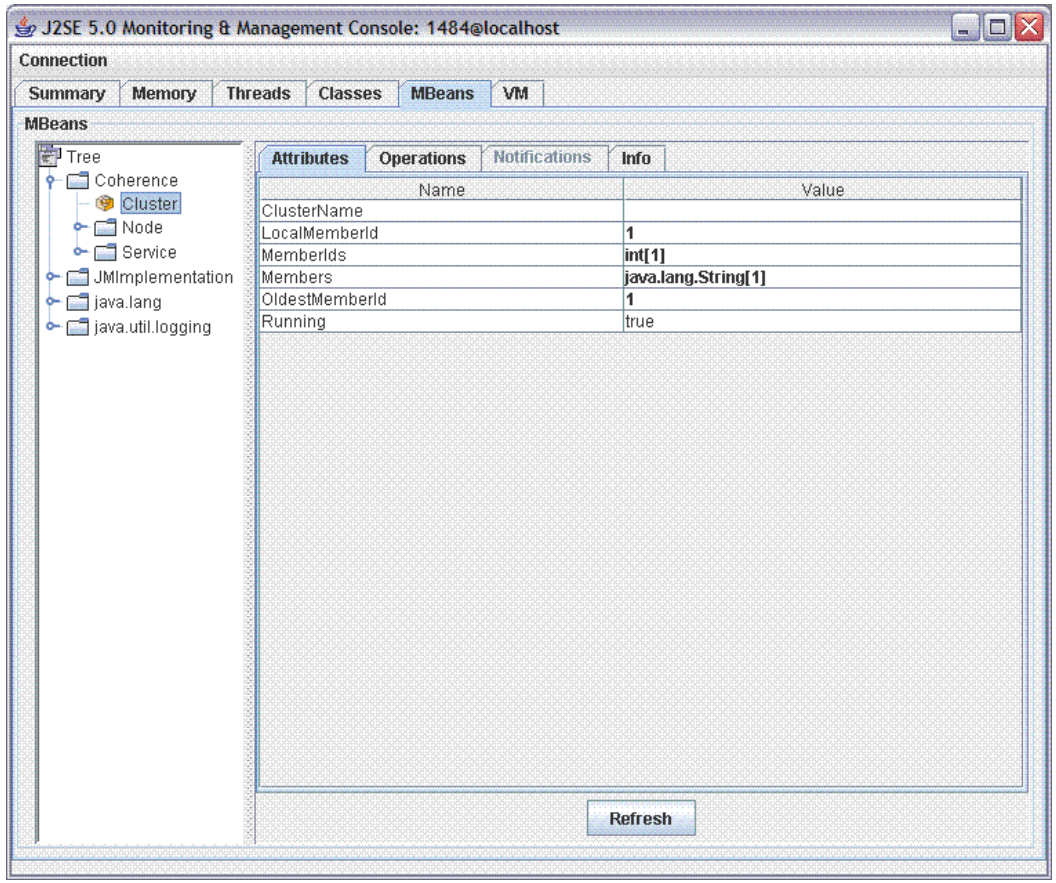
-Dtangosol.coherence.management=all

-Dtangosol.coherence.management.remote=true

-jar coherence.jar

Как работает Coherence? Взгляд со стороны архитектурной реализации....

Или через Jconsole J2SE:



Coherence*Web Session Management Module: Supported Web Containers

Application Server	Server Type Alias*
Apache Tomcat 4.1.x	Tomcat/4.1.x
Apache Tomcat 5.0.x	Tomcat/5.0.x
Apache Tomcat 5.5.x	Tomcat/5.5.x
BEA™ WebLogic™ 8.x	WebLogic/8.x
BEA™ WebLogic™ 9.x	WebLogic/9.x
BEA™ WebLogic™ Portal 8.1.6+	WebLogic/Portal/8.1.6+
Caucho Resin® 3.0.x	Resin/3.0.x
IronFlare Orion 2.0.x	Orion/2.0.x
IBM® WebSphere™ 5.x	WebSphere/5.x
IBM® WebSphere™ 6.x	WebSphere/6.x
JBoss Application Server	Jetty/4.2+ *or* Tomcat/*
Jetty 4.2.14 and later	Jetty/4.2+
New Atlanta ServletExec® 5.0	ServletExec/5.x
Oracle® OC4J 10.1.2.x	Oracle/10.1.2.x
Oracle® OC4J 10.1.3.x	Oracle/10.1.3.x
Sun™ ONE 6.1 and 7	SunONE

Разделяемая топология кэши: Доступ к данным

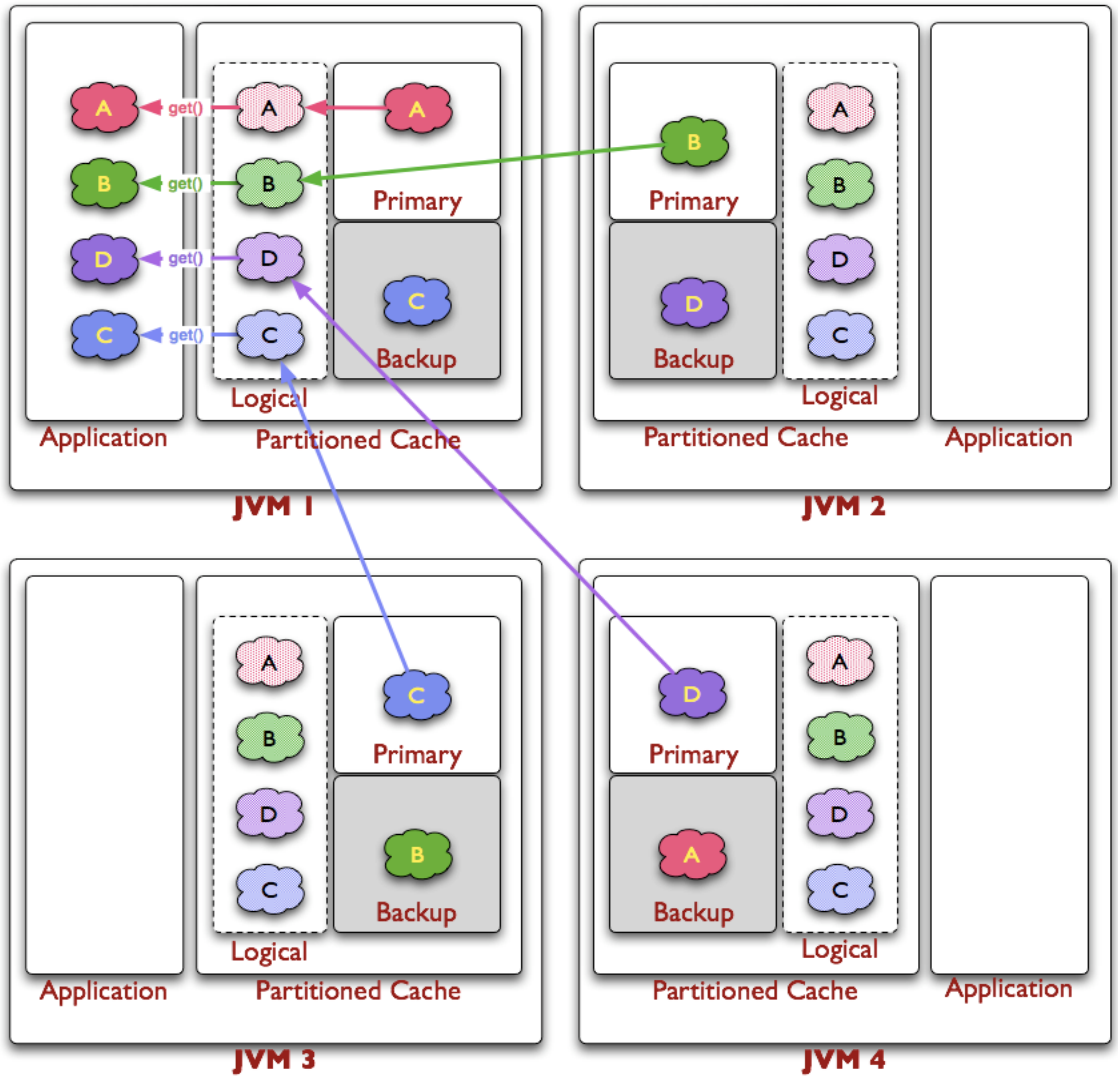
Data Access Topologies

•Coherence provides many Topologies for Data Management

• Local, Near, Replicated, Overview, Disk, Off-Heap, Extend (WAN), Extend (Clients)

Partitioned Topology

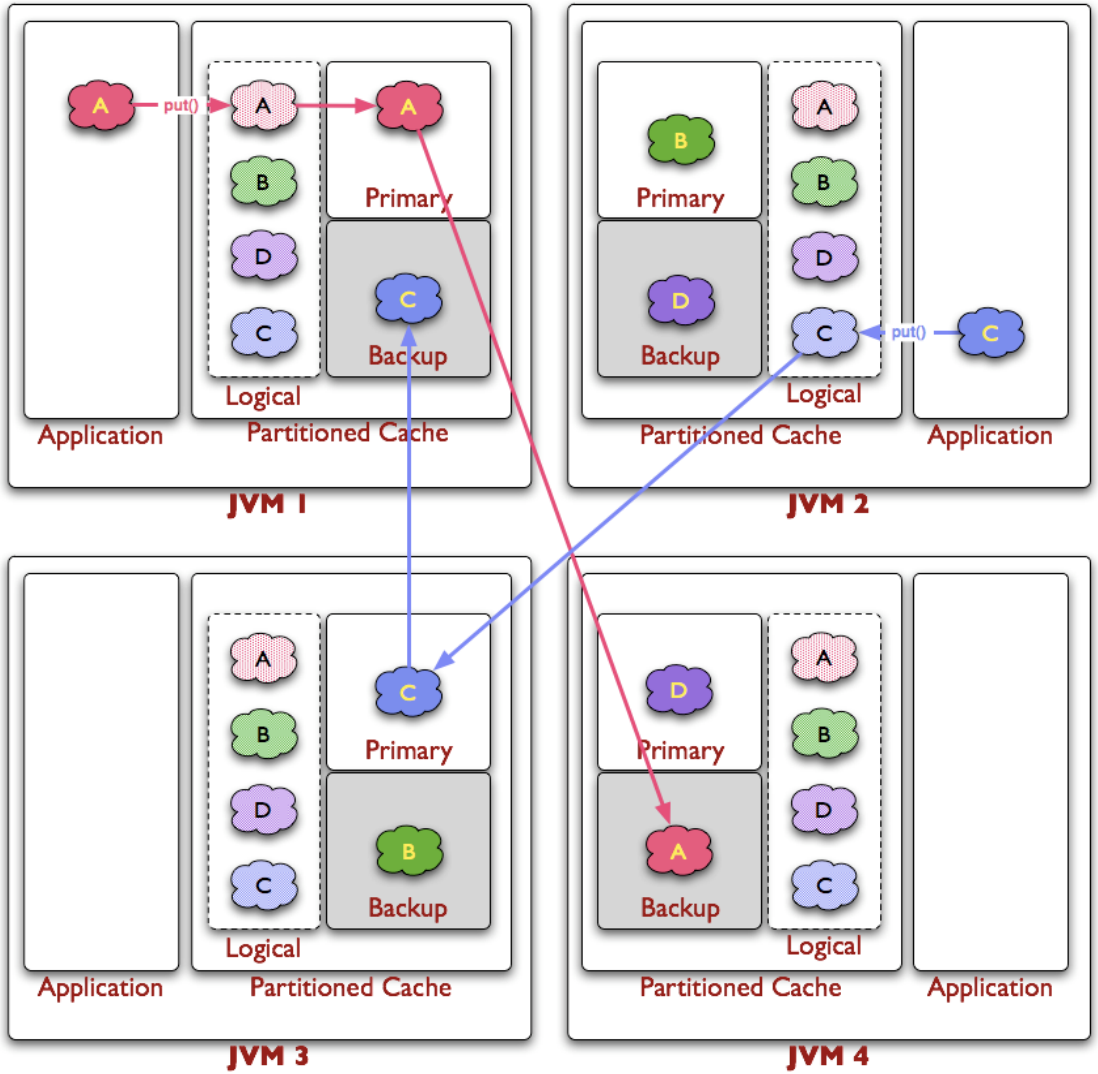
• Data spread and backed up across Members
• Transparent to developer
• Members have access to all Data
• All Data locations are known – no lookup & no registry!



Разделяемая топология кэши: Обновление данных

Partitioned Topology

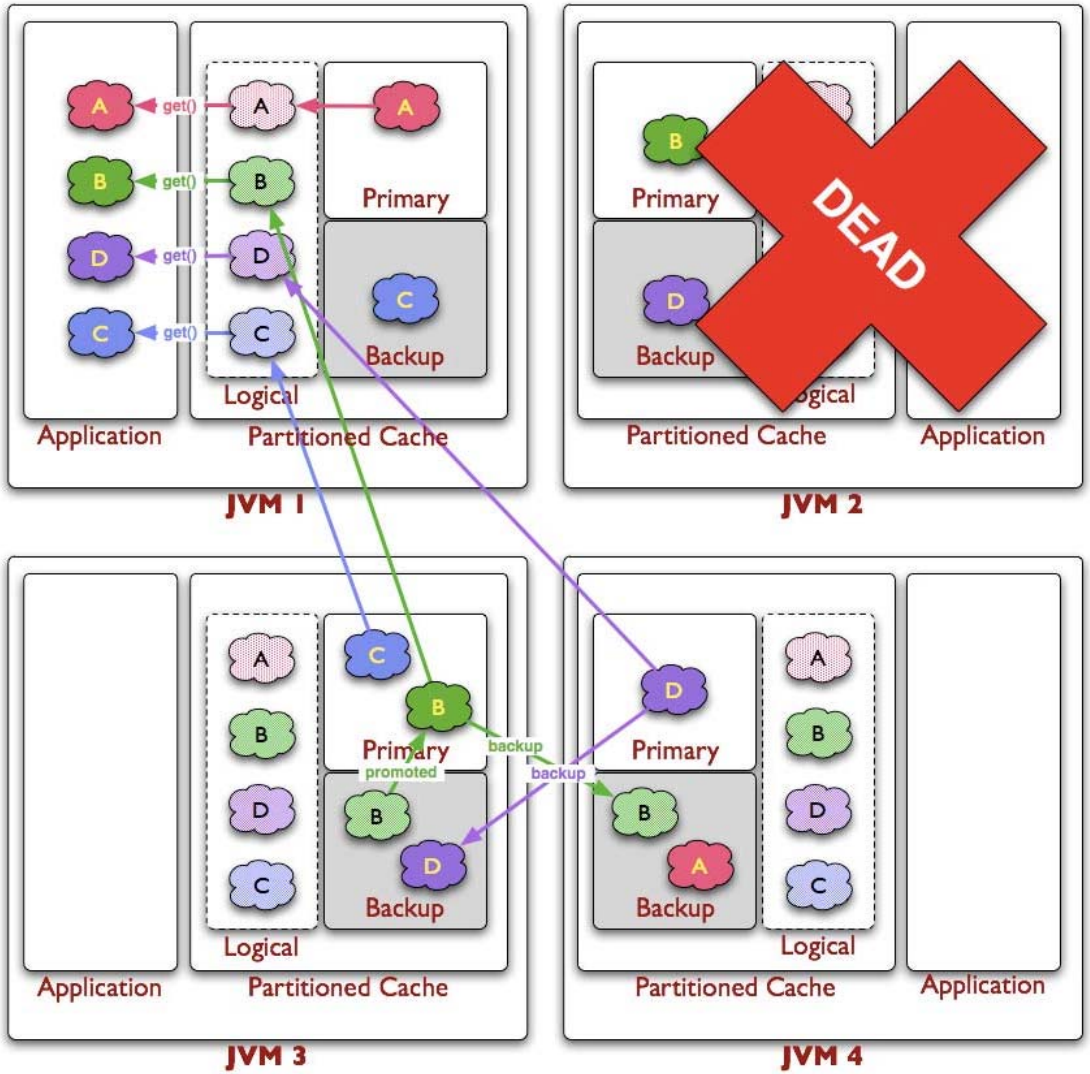
- Synchronous Update
- Avoids potential Data Loss & Corruption
- Predictable Performance
- Backup Partitions are partitioned away from Primaries for resilience
- No engineering requirement to setup Primaries or Backups
- Automatically and Dynamically Managed



Разделяемая топология кэши: Восстановление данных

Partitioned Topology

- Membership changes (new members added or members leaving)
- Other members, in parallel, recover / repartition
- No in-flight operations lost
- Some latencies (due to higher priority of recovery)
- Reliability, Availability, Scalability, Performance are the priority
- Degrade performance of some requests



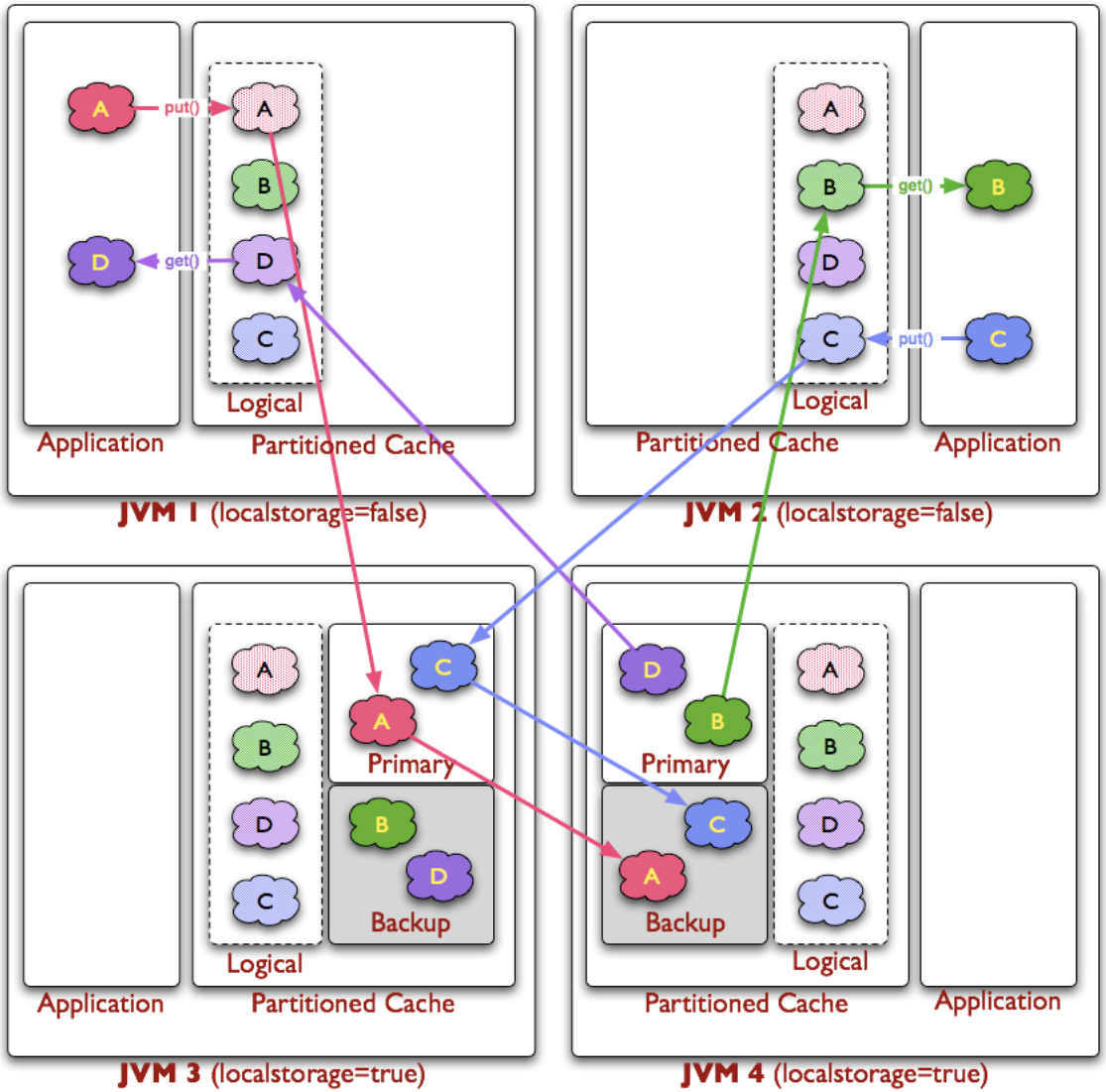
Разделяемая топология кэши:

- Детерминированное время для доступа к данным и их обновления.
- Практически линейное масштабирование согласно дизайну архитектуры.
- Все операции point-to-point (без multi-cast).
- Нет необходимости использования входящих TCP/IP коннекций для создания и поддержки элементов в архитектуре.
- Отсутствие потерянных операций в течение repartitioning.
- Нет необходимости остановки кластера в течение восстановления при разрушении элемента кластера, добавлении новых элементов, добавлении новых объектов кэши.
- Отсутствие необходимости перехвата сетевых исключительных ситуаций в течение repartitioning
- Динамическое выделение разделов кластера означает автоматическое масштабирование кластера по требованию - SOOD (“scale-out on demand”).

Разделяемая топология кэши: : Local Storage

Partitioned Topology

- Some members are temporary in a cluster
- They should not cause repartitioning
- Repartitioning means work for the other members (and network traffic)
- So turn off storage!



Составные топологии : Near Topology

- Coherence разрешает использовать составные топологии
- Базовые топологии

Local, Replicated, Partitioned / Distributed, Extend

- Составные топологии

Near…

- Near Topology

Объединение внешней и внутренней кэши в одну топологию.

Разрешение использования кэшей первого и второго уровня- L1 и L2.

Типы внутренней и внешней кэши могут быть абсолютно различны.

Оба типа кэши могут иметь различные expiry Policies

- Expiry Policies

LRU, LFU, Hybrid, Seppuku, Custom…

Составные топологии : Near Topology

- **Expiry Policies**

LRU – expiry policy, основанное на последнем времени обращения к объекту.

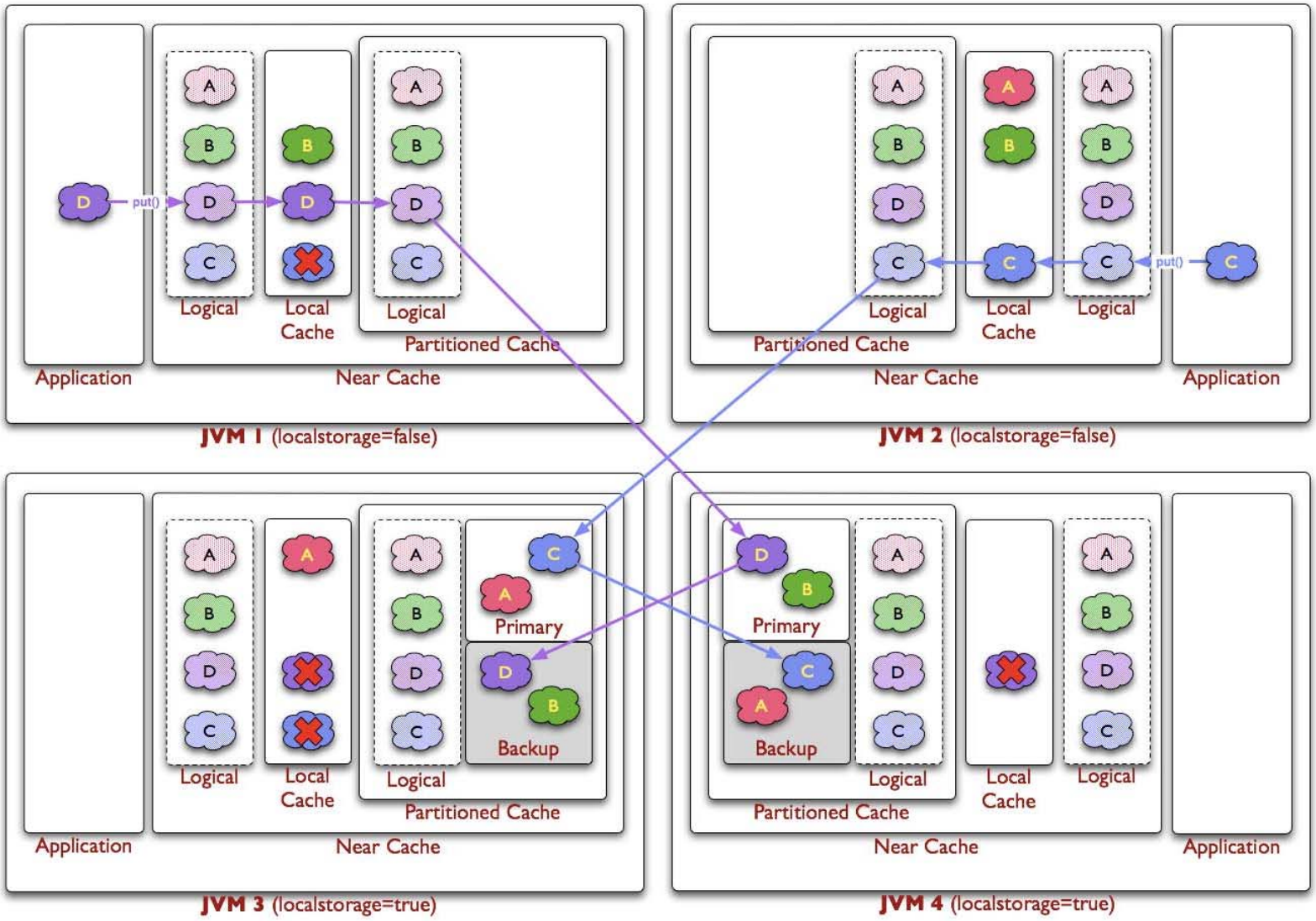
LFU – expiry policy, очень редко используемое, базируется на частоте обращения к объекту.

Hybrid – expiry policy, на основе взвешенного счета, как часто и недавно обращались к объекту.

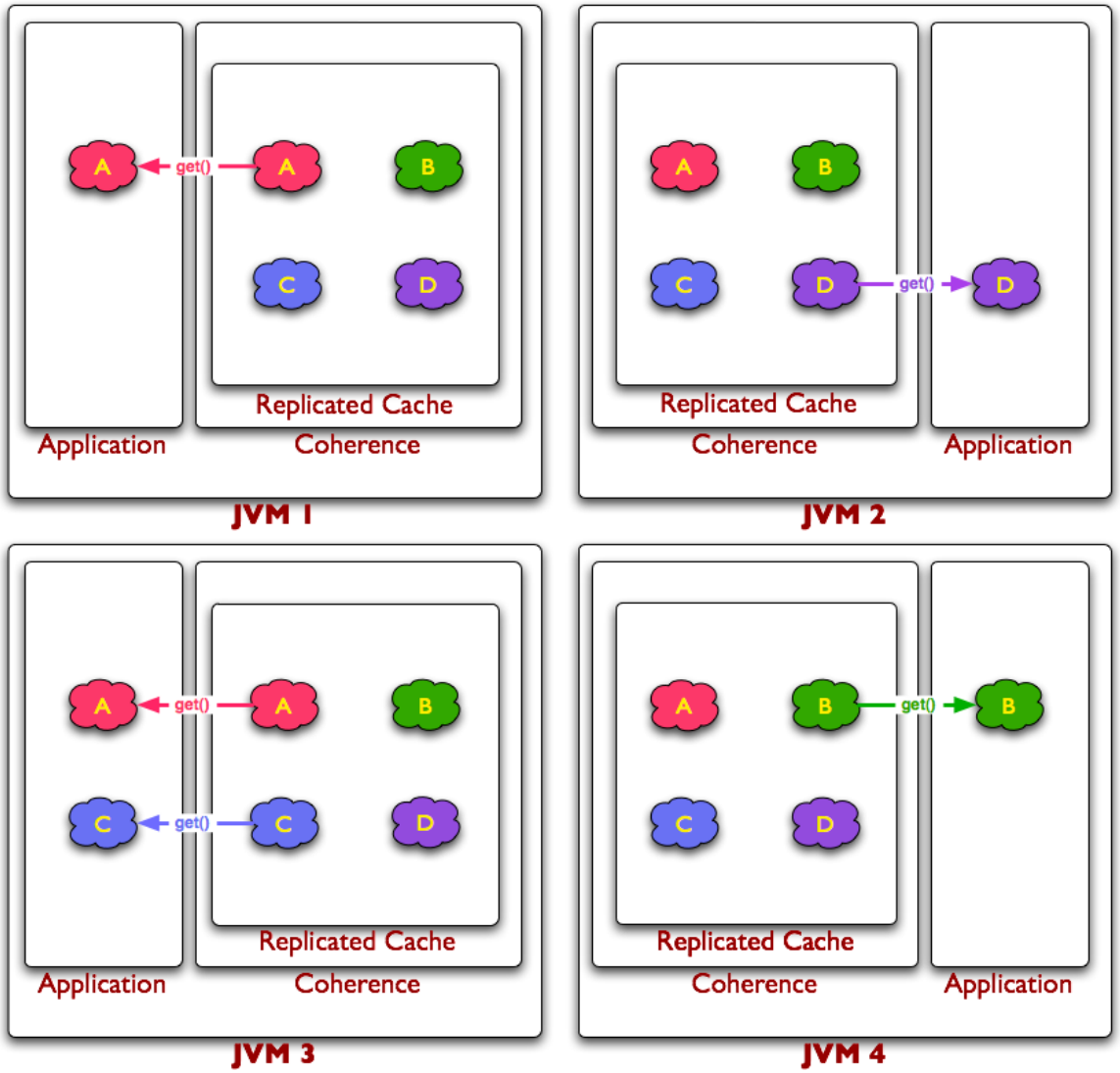
Seppuku – expiry policy, основанное на первичном удалении локальной кэши.

Custom – х-м-м-м...., для любителей сакрального....

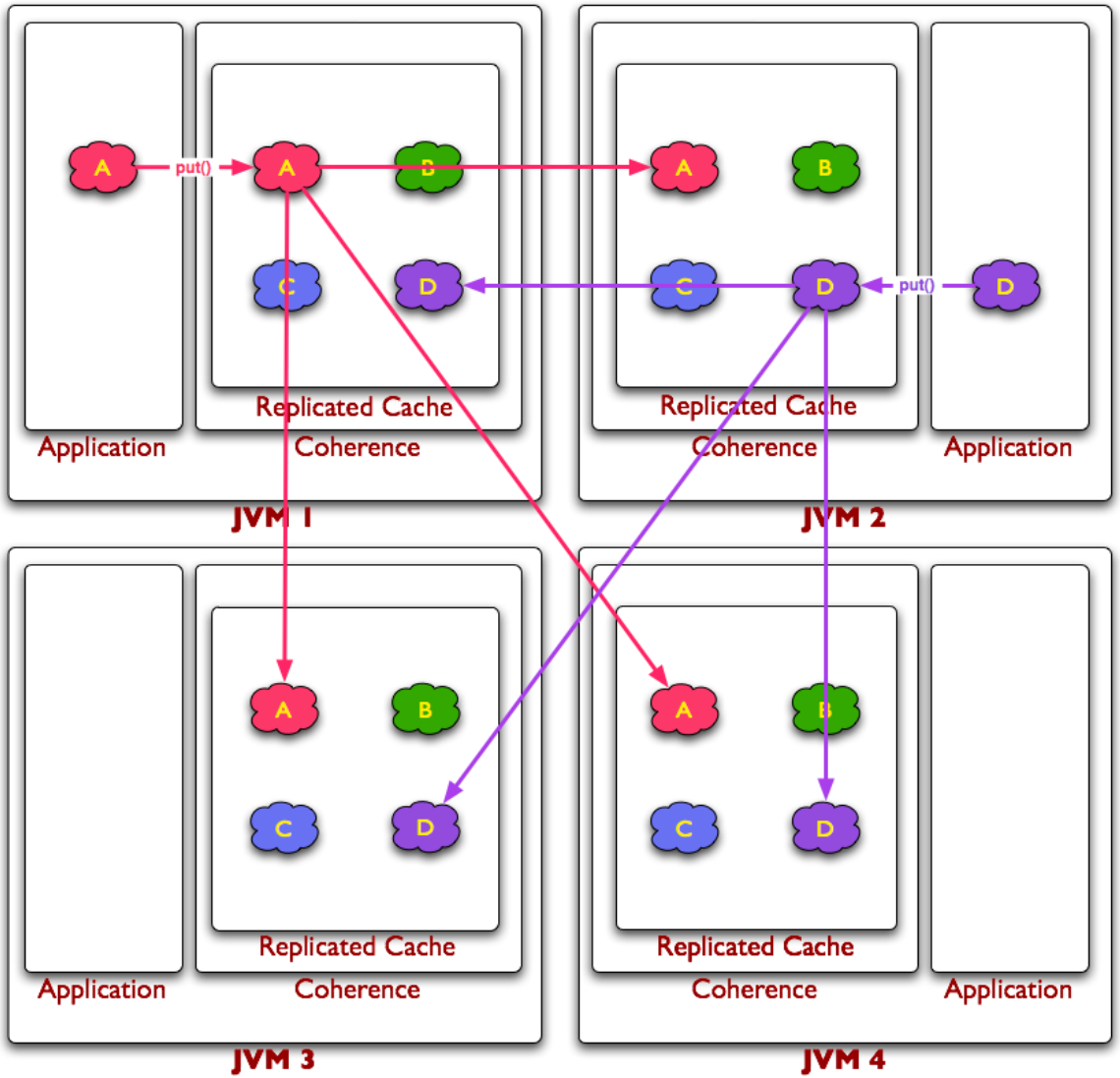
Составные топологии : Near Topology



Топология репликации : Доступ к данным



Топология репликации : Обновление данных



Традиционные возможности, интегрированные в Coherence

- Имплементация Map interface
- Drop in replacement. Full concurrency control. Multi- threaded. Scalable.

get, put, putAll, size, clear, lock, unlock…

- Имплементация JCache interface
- Расширенная поддержка для множества expiration policies, включая отсутствие expiration!
- Больше, чем просто кэш. Больше, чем просто мапирование.

Фильтры : Observable Interface

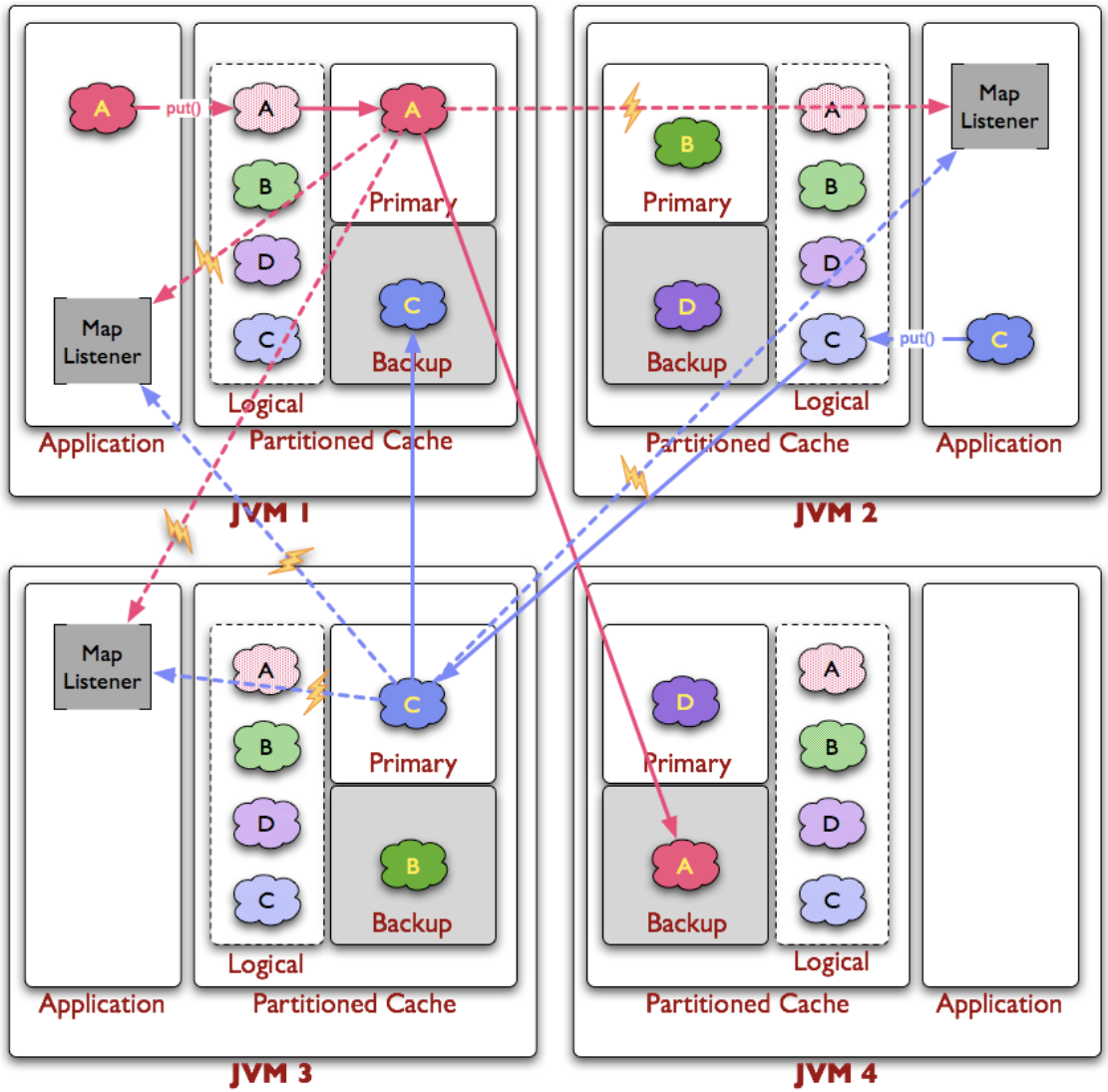
- Фильтрация событий в реальном времени для операций вставки, обновления, стирания данных.
- Возможность параллельного применения фильтров (в Grid Edition)
- Фильтры полностью расширяемы.
- Большое количество фильтров поставляемых с продуктом:

All, Always, And, Any, Array, Between, Class, Comparison, ContainsAll, ContainsAny, Contains, Equals, GreaterEquals, Greater, In, InKeySet, IsNotNull, IsNull, LessEquals, Less, Like, Limit, Never, NotEquals, Not, Or, Present, Xor…

- События могут быть синхронными:

```
trades.addMapListener(
new StockEventFilter("ORCL"),
```

Фильтры : Observable Interface



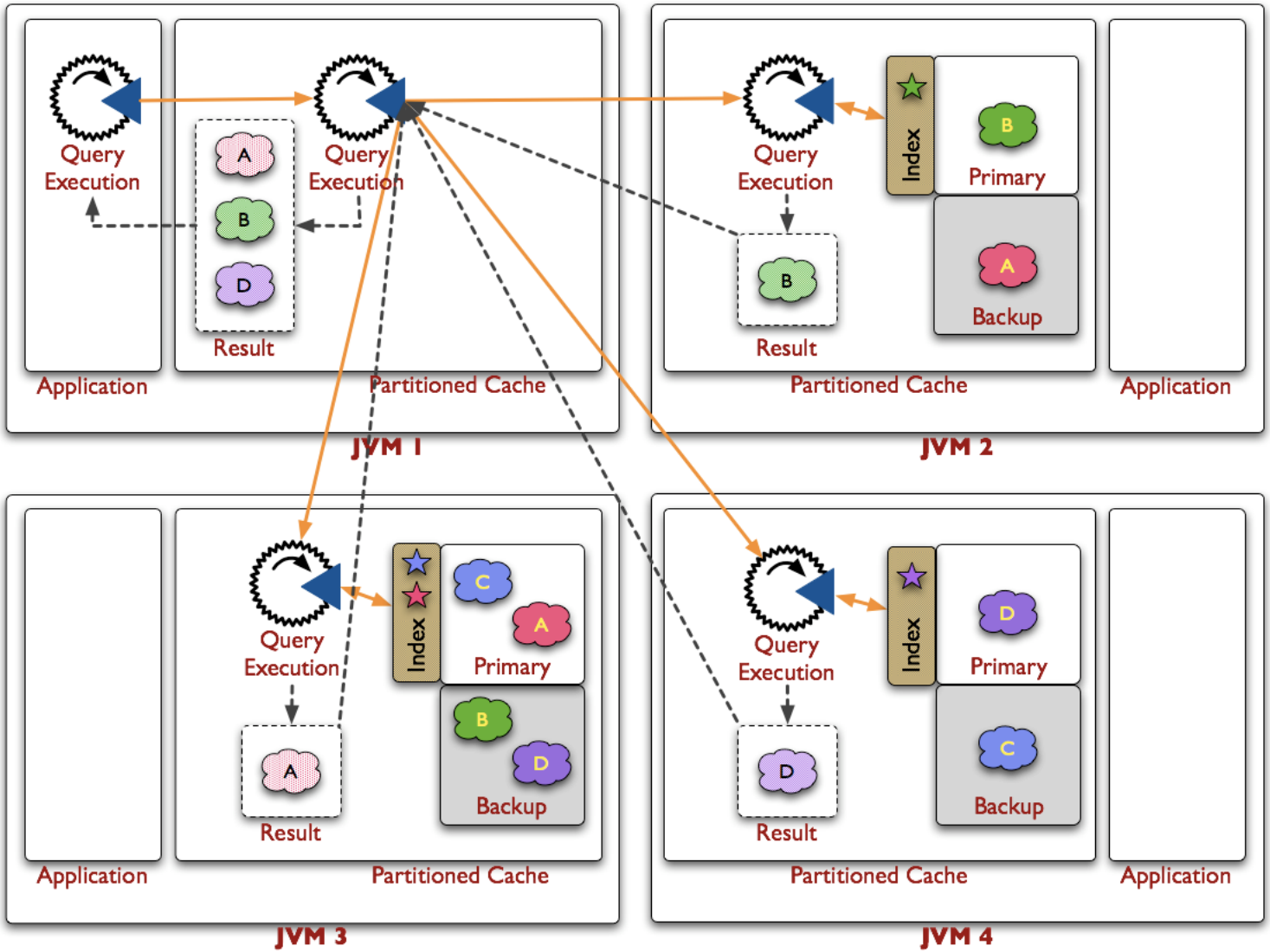
Фильтры : QueryMap Interface

- Нахождение ключей или значений, которые удовлетворяют фильтру.

entrySet(…), keySet(…)

- Определение индексов (на лету) для извлечения и индексирования любой части ввода данных.
- Исполнение в параллельном режиме
- Создание непрерывных представлений используемых запросов, основанных на совпадении с работой фильтра в реальном режиме времени.
- Удобно использовать для клиентских приложений, анализирующих и наблюдающих за данными.

Фильтры : QueryMap Interface



Фильтры : InvocableMap Interface

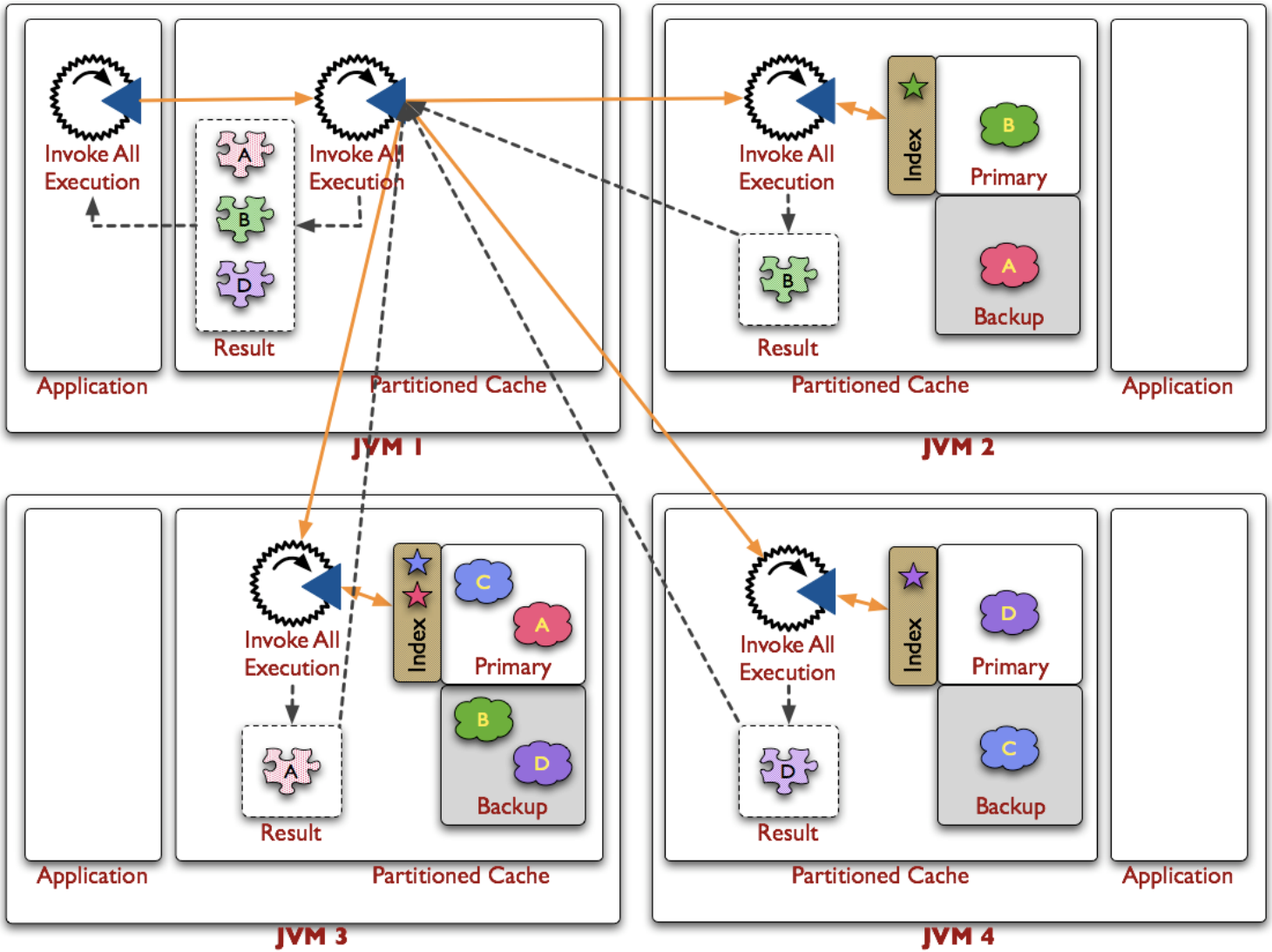
- Выполнение процессов с привязкой к вводу данных, их сбору или фильтрации.
- Выполнение процессов в параллельном режиме (aka: Grid-style)
- Отсутствие необходимости управление в распределенной конфигурации
- Процессы могут возвращать любые значения

```
trades.invokeAll(
new EqualsFilter("getSecurity","ORCL"), new StockSplit(2.0));
```

- Агрегирование ввода данных базируется на фильтрации

```
positions.aggregate(
new EqualsFilter("getSecurity","ORCL"), new SumFilter("amount"));
```

Фильтры : InvocableMap Interface



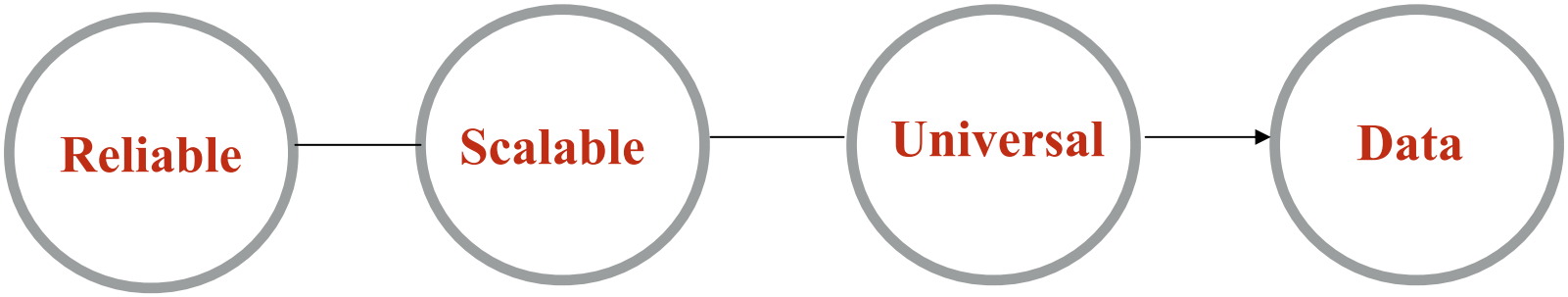
Раздел

Преимущества использования продукта.

Преимущества использования Coherence

- Возможность масштабирования stateful приложений.
- “Если Вам необходимо business agility в системе”
- Возможность экономии ресурсов
- Отказ от управления кластерами приложений
- Отказ от дизайна систем и приложений вокруг специализированных кластер-мастеров.
- Отказ от ручного кодирования для разделения data and service partitioning
- Если Вы хотите предсказуемой оценки затрат на масштабирование системы без ее продолжительного перекодирования или переконфигурирования.
- Если Вы хотите использовать природную поддержку языка программирования без дополнительных оберток или встроенных сторонних библиотек.

Преимущества использования Coherence (согласно Tangosol).



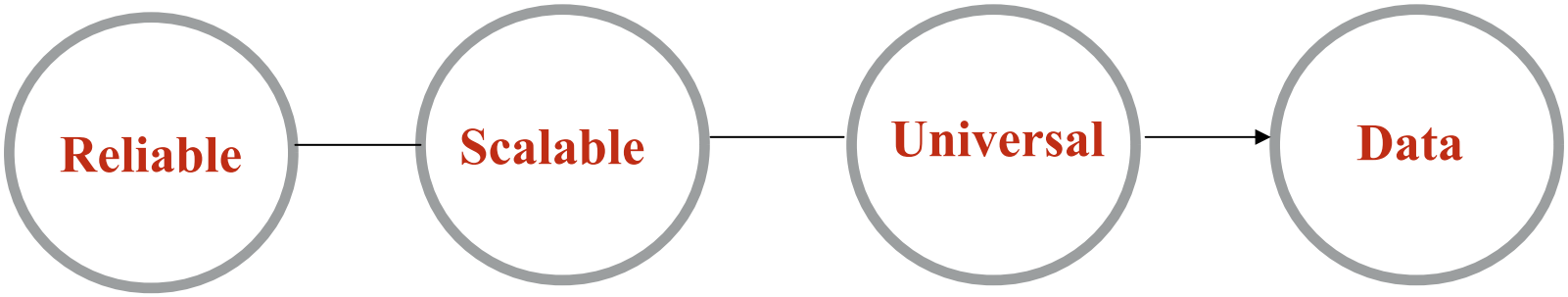
- Built for continuous operation
- Data Fault Tolerance
- Self-Diagnosis and Healing
- “Once and Only Once” Processing

- Dynamically Expandable
- No data loss at any volume
- No interruption of service
- Leverage Commodity Hardware
- Cost Effective

- Single view of data
- Single management view
- Simple programming model
- Any Application
- Any Data Source

- Data Caching
- Analytics
- Transaction Processing
- Event Processing

Преимущества использования Coherence (согласно Tangosol).



- Построен для непрерывной работы
- Целостность данных
- Самодиагностика и заживление
- “Однажды и Только Однажды” обработка

- Динамическая Расширяемость
- Отсутсвие потерь данных в любом элементе
- Отсутствие прерываний для обслуживания
- Экономия ресурсов аппаратных средств
- Экономия затрат

- Единственное представление данных
- Единственное представление управления
- Простая модель для программирования
- Любое Приложение
- Любой Источник данных

- Кэширование данных
- Аналитика
- Транзакционная Jбработка -XTP
- Обработка событий

Раздел

Взаимодействие Coherence и других продуктов

Возможности архитектурной интеграции:

Интеграция с различными Data Sources.

Read Through: Чтение из CacheLoader когда данные не в grid

Write Through: Запись в CacheStore когда данные вставляются, обновляются или удаляются в grid

Write Behind: Асинхронные и объединенные обновления к CacheStore когда данные вставляются, обновляются или удаляются в grid

Управление сессиями (Coherence – Web development).

Возможности архитектурной интеграции:

Встроенные возможности:

Интеграция со Spring Framework.

Приложения, созданные с применением Spring Framework становятся полностью кластеризованными, включая регистрацию изменений, связанных с изменением количества элементов в кластере.

Интеграция с Hibernate Framework.

Интеграция с TopLink JPA.

Управление HTTP-сессиями для серверов приложений

Интеграция с BEA WebLogic порталом.

Возможности архитектурной интеграции:

Обеспечивается с применением:

Push/Pull модели данных основанной на подписке и уведомлении о происходящих событиях

Client/Data Grid модели, где клиенты подсоединяются к Coherence для взаимодействия с данными и службами

Раздел

Внедрение продукта, потенциальные заказчики, цены.

- Внедрение продукта – потенциально везде, где существуют прикладные OOP системы для придания им кластерных черт. Новые системы на .Net и Java.
- В будущем – Oracle Fusion Applications.
- В будущем – полная интеграция в платформу Oracle Fusion Middleware.

- Потенциальные заказчики – так как не существует глубокой привязки к типу Data Source,базе данных, типу используемых frameworks выбор заказчика может быть широк.
- Все-таки, для внедрения продукта требуется предконсалтинговая и возможно консалтинговая проработка . Следовательно это не SMB клиенты, если только Coherence не интегрируется в тиражируемый продукт.

Предложение продукта Coherence в Tangosol (взято за основу в Oracle):

		Standard Edition *(Formerly known as Caching Edition)* Application caching solution	**Enterprise Edition** *(Formerly known as Application Edition)* Application data management	**Grid Edition** *(Formerly known as Data Grid Edition)* Enterprise-wide data management
Connectivity	Embedded Data Client and Real Time Client functionality [1]	●	●	●
	TCMP cluster technology [3], [8]	●	●	●
	Support for cross-platform Data Clients	●	●	●
	Multicast-free operation (WKA)		●	●
Security	Network traffic encryption	●	●	●
	Java Authentication & Authorization Service (JAAS)	●	●	●
Management & Monitoring	Management host [3]	●	●	●
	Manageable via clustered JMX		●	●
Caching	Local cache, Near cache, continuous query cache, real-time events	●	●	●
	Fully replicated data management	●	●	●
	Partitioned data management	●	●	●
	Data source integration via read-through/write-through caching	●	●	●
Integration	Hibernate integration	●	●	●
	HTTP session management for application servers [4]		●	●
	BEA Portal "p13n cache" integration		●	●
Analytics	Parallel InvocableMap and QueryMap [5]		●	●
Transactions	Write-behind caching		●	●
	J2CA Resource Adapter		●	●
Compute Grid	InvocationService		●	●
	WorkManager		●	●
Enterprise Data Grid	WAN support [6]			●
	Support for cross-platform Real Time Clients			●

Предложение продукта Coherence в Tangosol (взято за основу в Oracle):

		Data Client	Real Time Client
		Data Grid client for use anywhere	Real time desktop client
Client API	Data transformation (PIFPOF / ExternalizableLite / XmlBean)	●	●
	InvocationService	●[7]	●[7]
	NamedCache (core)	●	●
	NamedCache (with ObservableMap real time events)		●
Connectivity	Coherence*Extend client [9]	●	●
	Multicast-free operation	●[9]	●[9]
Security	Network traffic encryption	●	●
Caching	Local cache		●
	Near cache		●
	Continuous query cache		●

Предложение продукта Coherence в Oracle:

Предложение продукта Coherence в Oracle:

Section II

Oracle Fusion Middleware

Prices in USA (Dollar)

Internet Application Server Products	Named User Plus	Software Update License & Support	Processor License	Software Update License & Support	Notes
Coherence Standard Edition	80	17.60	4,000	880.00	1
Coherence Enterprise Edition	200	44.00	10,000	2,200.00	1
Coherence Grid Edition	400	88.00	20,000	4,400.00	1
Coherence Real Time Client	100	22.00	-	-	47

Oracle Technology Footnotes

[1] If licensing by Named User Plus, the minimum is 10 Named User Plus licenses per Processor.

[47] The Named User Plus minimums for this program are 25 Named User Plus licenses.

Раздел

Вопросы.

За дополнительной информацией обращайтесь:

Oracle СНГ:

Отдел предпродажного консалтинга,

Adress: 119435, Russia, Moscow, Savvinskaya nab., 15

Phone: +7(495) 641-1400

Fax: +7(495) 641-1414

E-Mail: oracle_ru@oracle.com


ORACLE®

ORACLE IS THE INFORMATION COMPANY