ORACLE

ORACLE

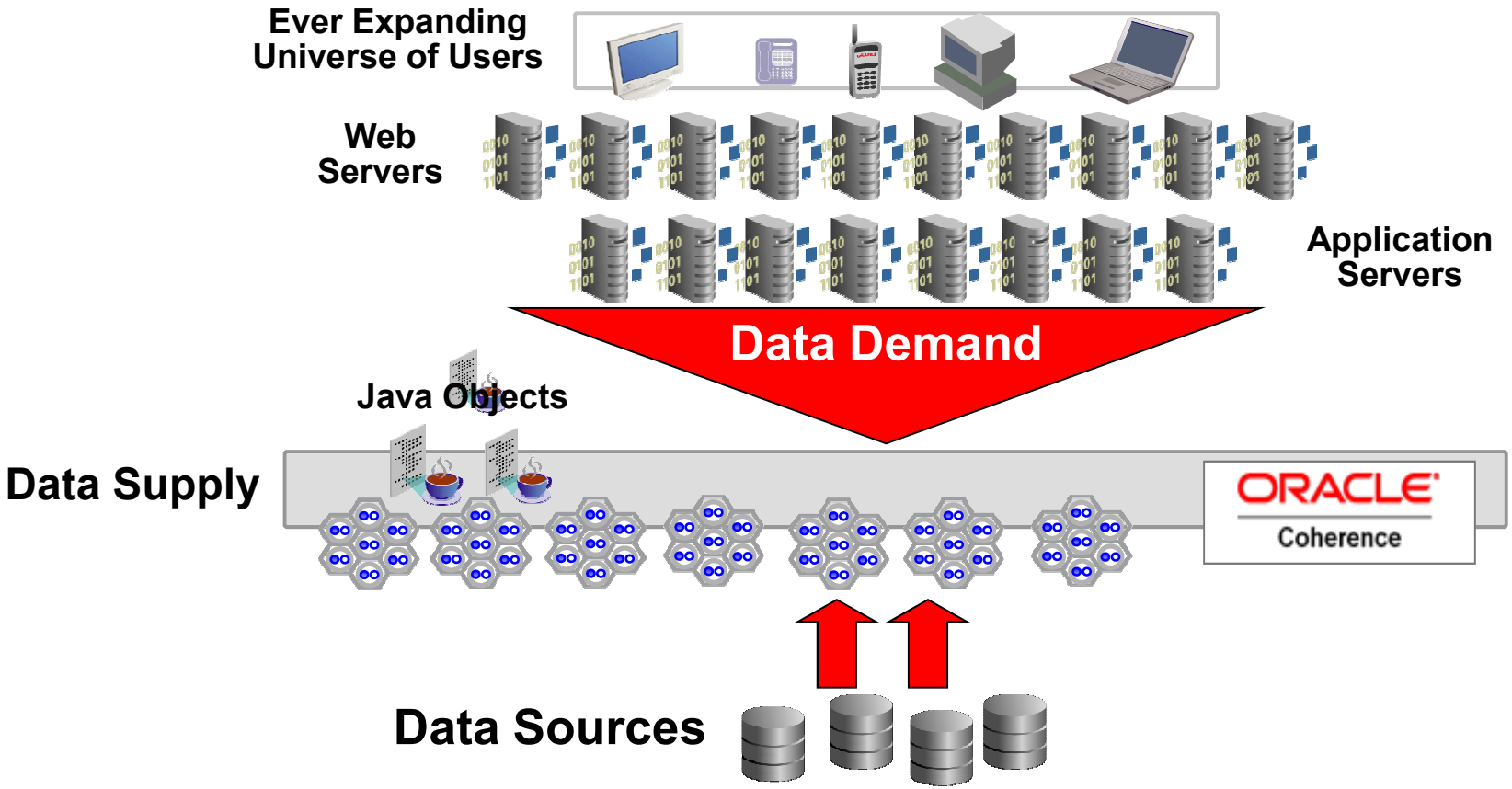
Oracle Coherence. Детали

Григорий Сень, Oracle
Ведущий консультант

Кластеризация данных

Oracle Coherence

- Доставки данных по запросу
- Работа на простом оборудовании
- Надежная обработка транзакций при любой загрузке системы



Oracle Coherence

Архитектура и топология развертывания

Высокоуровневая архитектура

- Oracle Coherence – это набор вычислительных узлов (кластер)
- Кластер управляет данными
- Кластер содержит данные (cache)
- Кластер обрабатывает данные
- Может иметь разные топологии развертывания

Высокоуровневая архитектура

Кластер

- Набор процессов (узлов), работающих вместе
- Все узлы имеют одинаковые полномочия*
  - для жизнеспособности кластера
- Нет статической/жесткой связи полномочий с оборудованием
- Нет главных и подчиненных узлов
- Нет централизованного реестра данных или сервисов
- Нет единой точки отказа
- Нет единой точки для узкого места

***узлы можно запросить выполнять персональные задания, например: Grid Agents**

Высокоуровневая архитектура

Данные

- Любой сериализуемый (serializable*) объект
  - Отсутствие внутренних классов для расширения**
  - POJOs и PONOs, а и POCOs
- Полная поддержка совместимости с Java, .NET и C++
- Никаких машинно-зависимых инструкций
- Нет необходимости использовать Relational Models, Object-Relational-Mapping, SQL и т.д.

***сериализация = запись в двоичную форму**

**** применение специальных интерфейсов увеличивает производительность**

Высокоуровневая архитектура

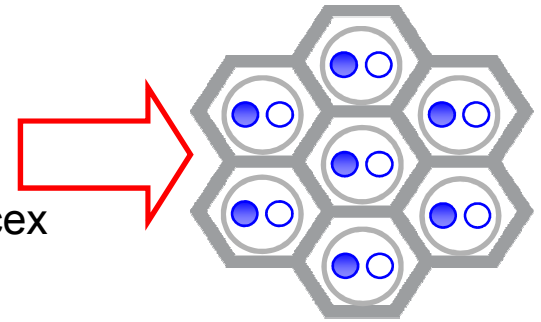
Управление и обработка данных

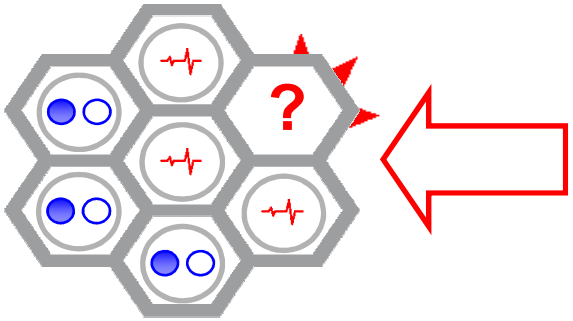
- Создание, удаление, изменение
- Простой API для всех данных, независимо от типов
- Обработка
  - Distributed Objects, Maps, Caching (распределенные объекты)
  - Real-Time Events, Listeners (события и листенеры)
  - Parallel Queries, Indexing (запросы и индексы)
  - Data Processing and Service Agents (Grid возможности)
  - Continuous Views (непротиворечивые представления)
  - Aggregation (статистические операции)
  - Persistence, Sessions (поддержка сессий и источников данных)

Высокоуровневая архитектура

Как работает Oracle Coherence

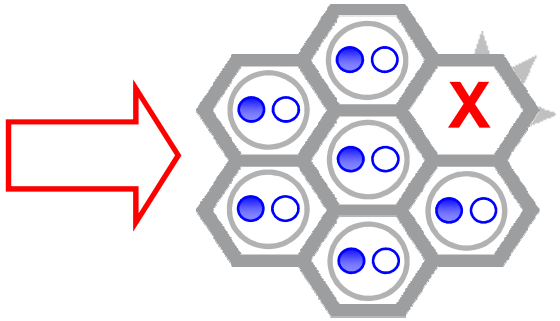
- Данные распределяются в памяти серверов кластера
- Данные автоматически и синхронно реплицируются с одного сервера на другой для постоянной доступности
- Единый взгляд на данные: Логическое представление всех данных на всех серверах





- Сервера контролируют работоспособность друг друга
- Если сервер падает или неправильно работает, другие сервера совместно диагностируют его состояние

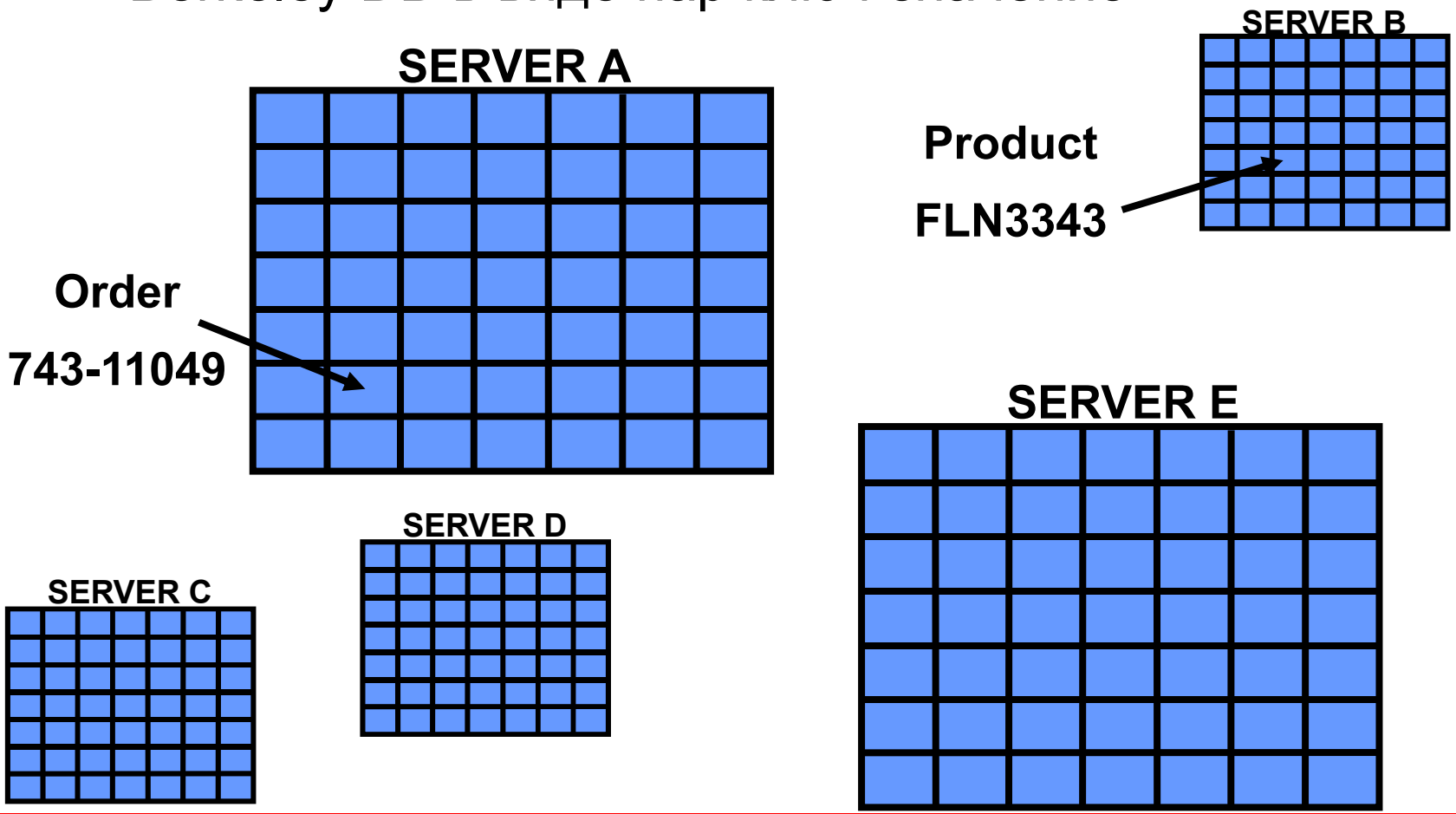
- Работоспособные сервера немедленно берут на себя обязанности упавшего сервера
- Непрерывность работы: В случае падения сервера не происходит прерывания работы или потери данных



Высокоуровневая архитектура

Как работает Oracle Coherence (продолжение)

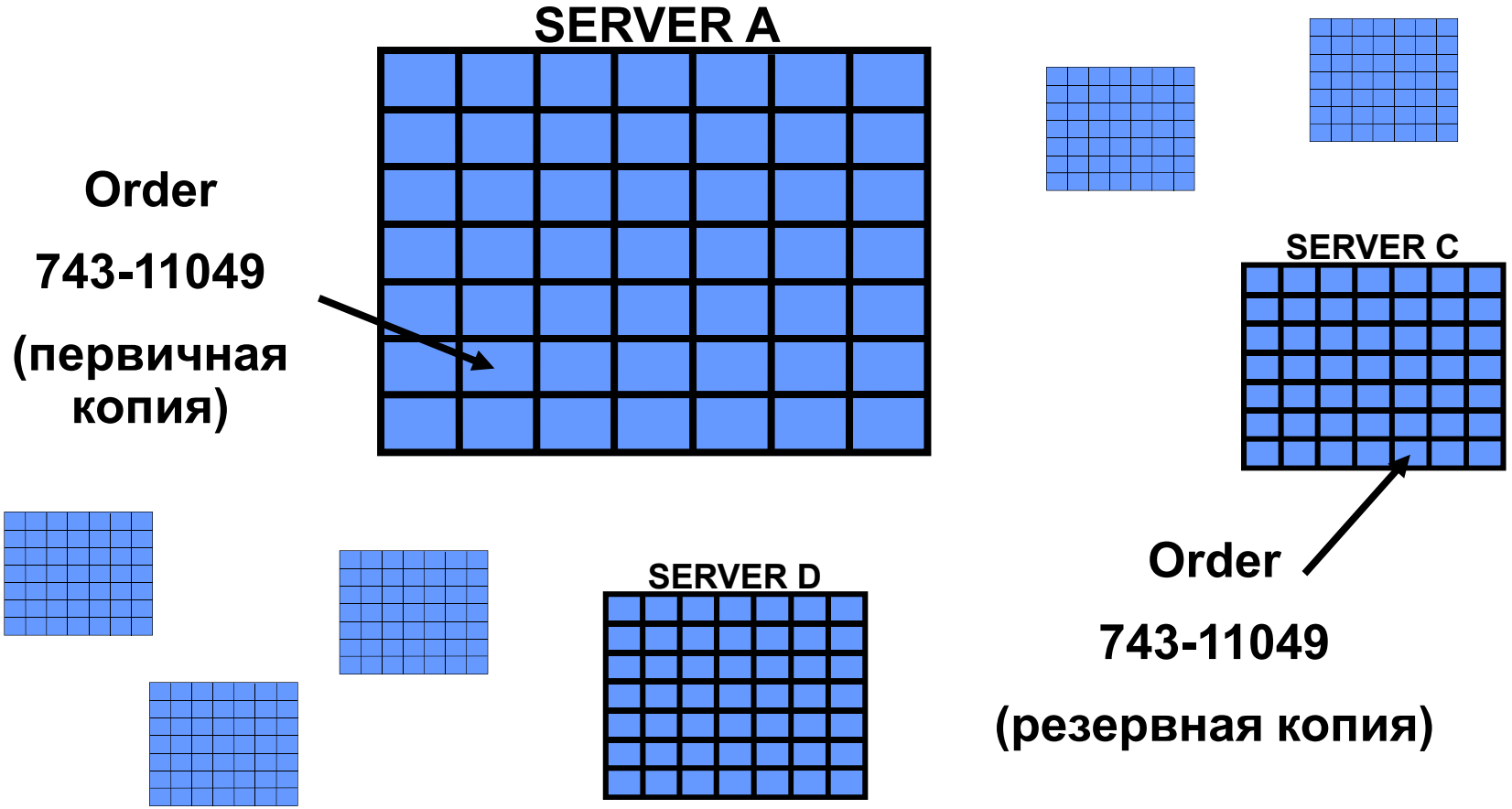
Coherence хранит данные подобно Berkeley DB в виде пар ключ-значение



Высокоуровневая архитектура

Как работает Oracle Coherence (продолжение)

Coherence использует избыточность для высокой надежности



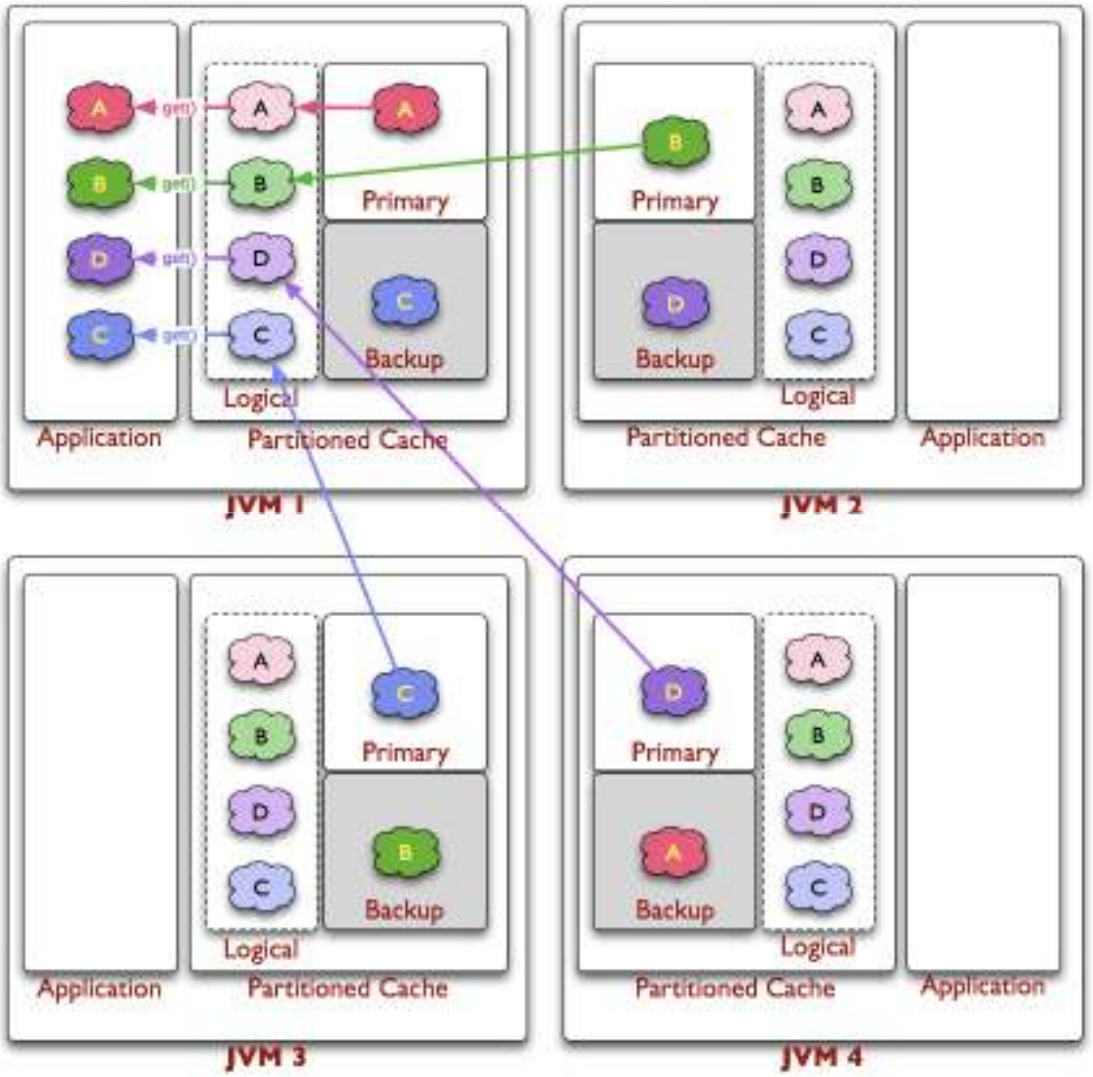
Технология разделения данных

Доступ к данным

- Oracle Coherence обеспечивает различные топологии для управления данными
- Local, Near, Replicated, Overview, Disk, Off-Heap, Extend (WAN), Extend (Clients)

<u>Топология разделения</u>

- Данные распространяются и резервируются между узлами
- Прозрачно для разработчиков
- Узлы имеют доступ ко всем данным

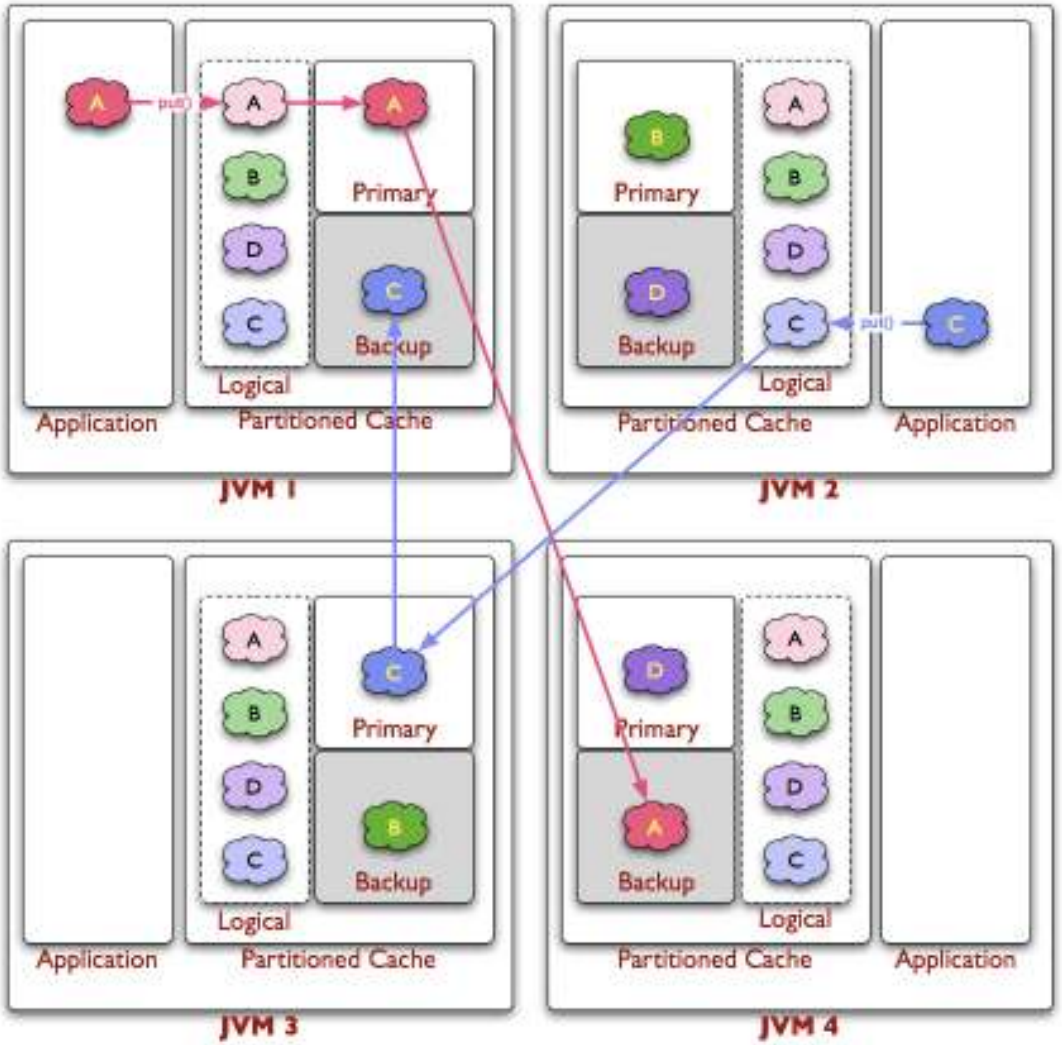


Технология разделения данных

Обновление данных

Топология разделения

- Синхронное обновление
- Предупреждение потенциальной потери и разрушения данных
- Предсказуемая производительность
- Резервные разделы выбираются далеко от первичных
- Не требуется настроек для установки первичных и резервных разделов
- Автоматическое и динамически управляемое

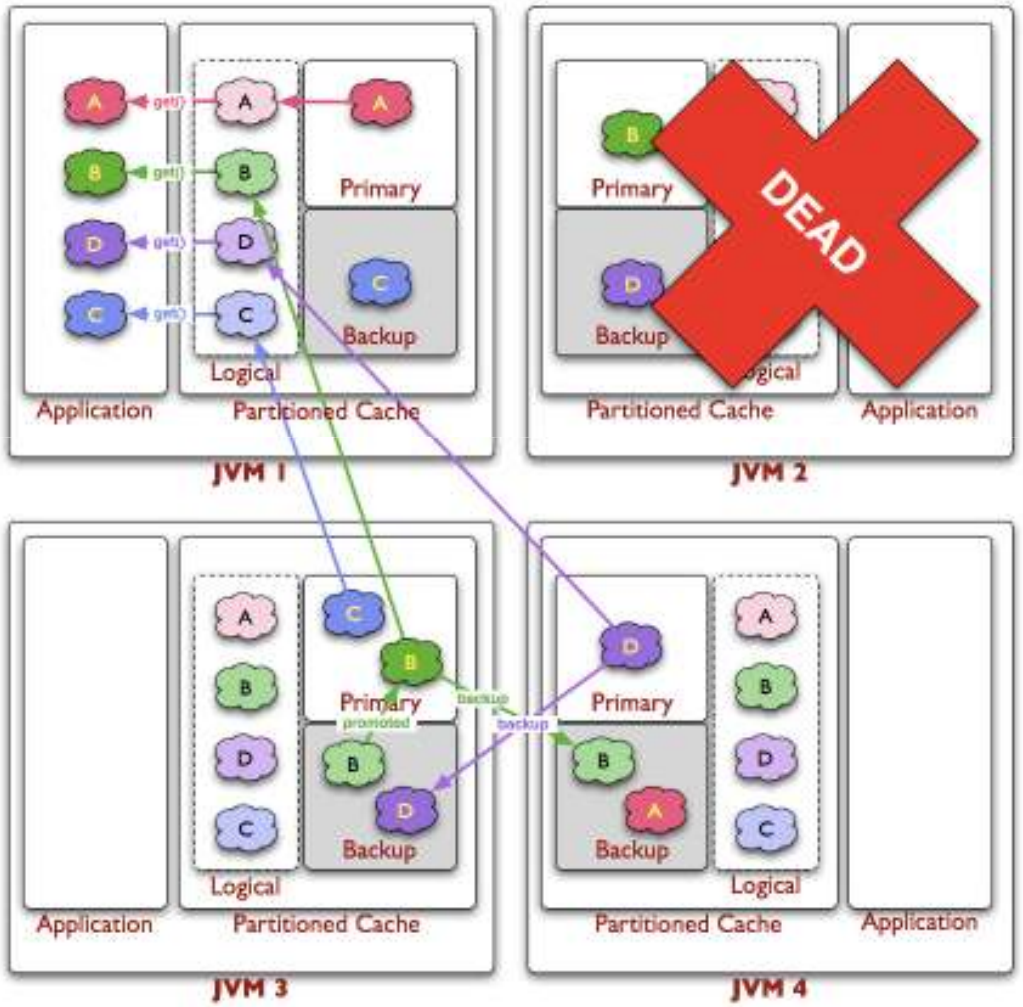


Технология разделения данных

Восстановление данных

Топология разделения

- Изменение состава узлов (новые узлы добавляются или старые покидают)
- Другие узлы параллельно, восстанавливаются / перераспределяются
- Запущенные операции продолжают выполнятся
- Бывают задержки (из-за более высокого приоритета восстановления)
- Надежность, Масштабируемость, Производительность – приоритетные задачи



Технология разделения данных

Ключевые возможности

- Постоянное время ожидания для доступа к данным и их обновления
- Линейная масштабируемость в соответствии с дизайном
  - Включая объемы
- Все операции обрабатываются в режиме точка-точка (без multi-cast)
- Не создаются/поддерживаются лишние TCP/IP соединения
- Не происходит потерь работающих операций в процессе перераспределения данных
- Не требуется перезагружать кластер для
  - восстановления узлов после падения
  - добавления новых узлов
  - добавления новых кэшей
- Не требуется обрабатывать исключения при перераспределении данных

Топология развертывания

- Механизм для организации данных часто называют Топология (Topology) или более в общем случае Схема (Scheme)
- Coherence поставляется с несколькими стандартными схемами
  - Можно сконфигурировать / перенастроить / создать свою собственную!
- Все настройки Coherence осуществляются программно или через конфигурационные файлы
  - Файлы, содержащие конфигурацию по умолчанию
  - Файлы, переопределяющие параметры по умолчнию

Топология развертывания

Схемы

Существуют базовые и составные схемы

- Базовые схемы
  - Local, Replicated, Distributed, Extend
- Составные схемы
  - Near, Overflow (запись на диск)
  - Позволяют подключать другие схемы

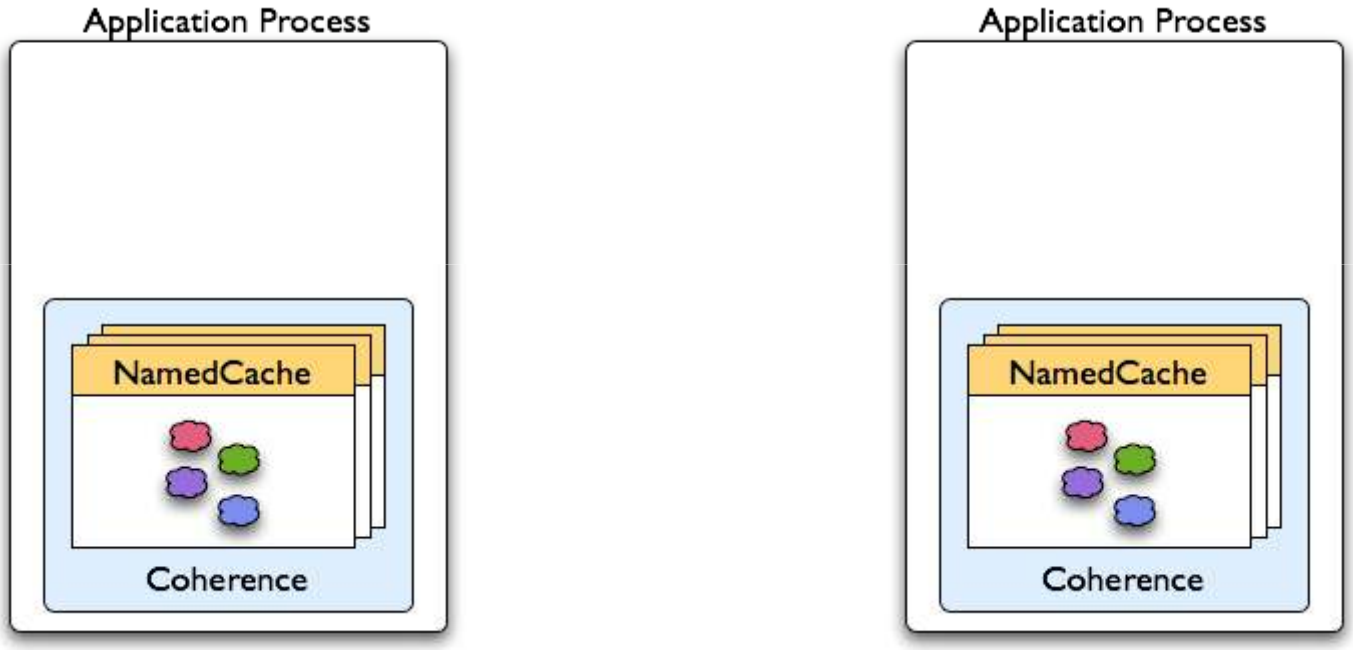
Топология развертывания

Локальная схема

- Некластеризуемый локальный кэш
  - Содержит локальные ссылки POJO в «куче» памяти приложения
- Зачем нужна:
  - Заменяет внутренние реализации кэша
  - Совместим с другими схемами Coherence
- Как работает:
  - Кэш на объекте SafeHashMap (высокая производительность, безопасность потоков)
  - Размер ограничен (если определен)
- Настраиваемые политики истечения срока действия:
  - LFU (Least Frequently Used), LRU (Least Recently Used), гибрид (LFU+LRU), По времени, Никогда, Подключаемая

Топология развертывания

Локальная схема



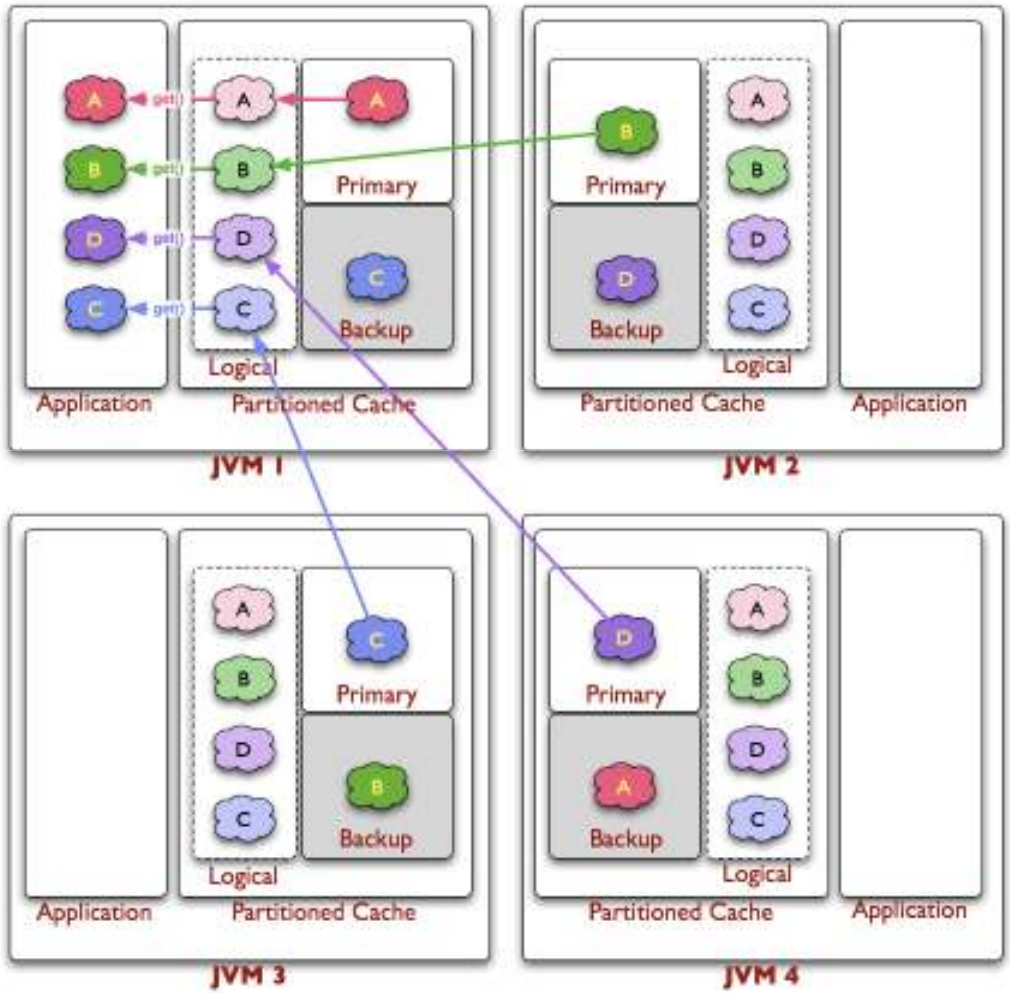
Топология развертывания

Распределенная (distributed) схема

- Реализует сложный подход для кластерного кэша
- Зачем нужна:
  - Разработана для экстремально высокой масштабируемости
- Как работает:
  - Прозрачное разделение, распространение и резервирование сущностей кэша между узлами
  - Часто называют «Топологией разделения»
- Настраиваемые политики истечения срока действия:
  - LFU, LRU, гибрид (LFU+LRU), По времени, Никогда, Подключаемая

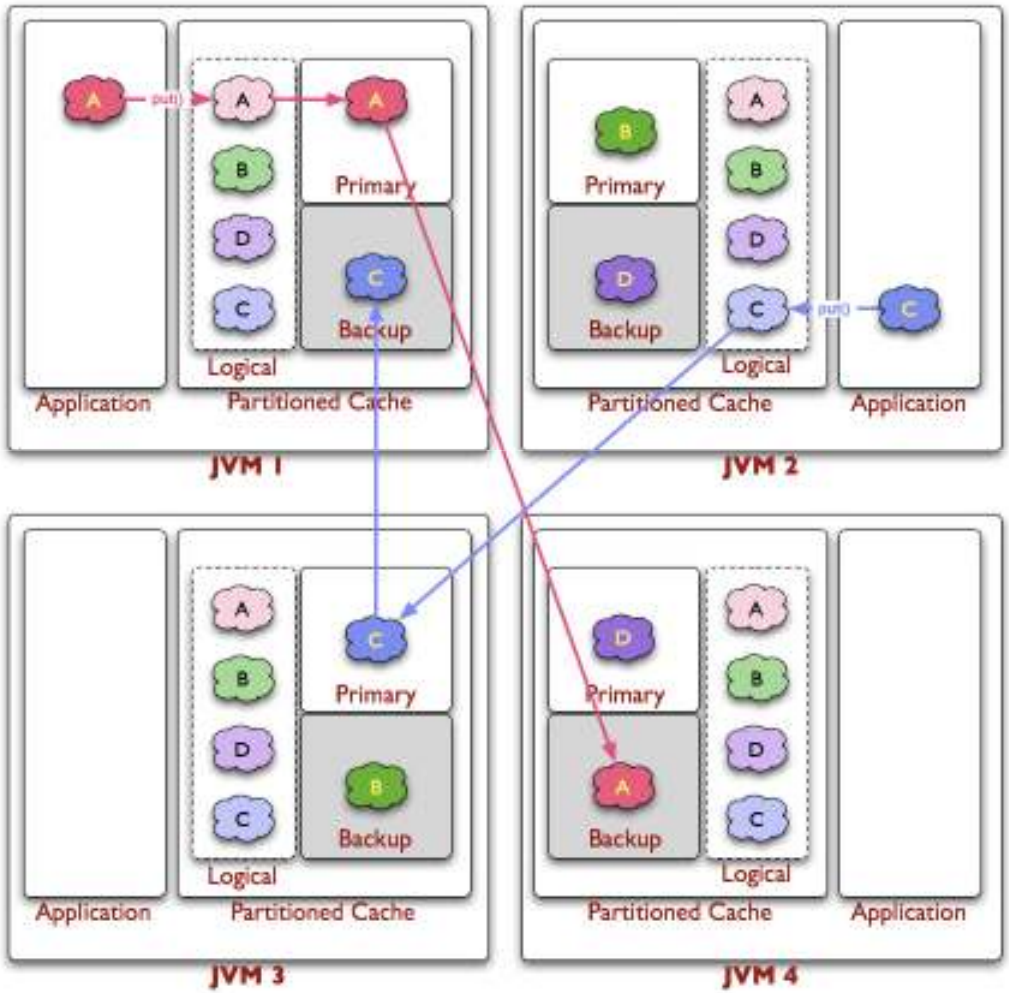
Распределенная схема

Чтение



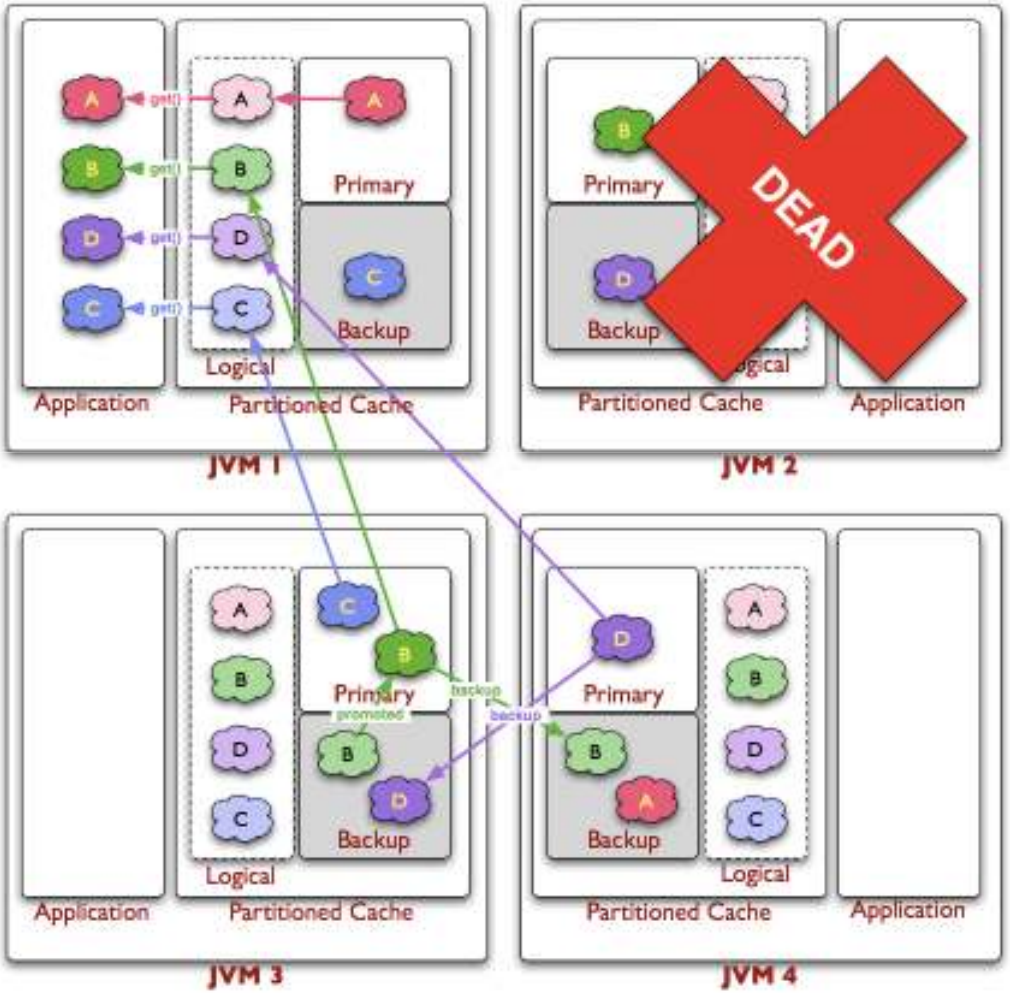
Распределенная схема

Запись



Распределенная схема

Удаление узла



Распределенная схема

Восстановление

- Плавный Failover и Failback
  - Резервные данные становятся первичными
  - Первичные делают новые резервные на другом узле
- Прозрачно для приложения
  - Кроме некоторых задержек при восстановлении записей
- Восстановление происходит параллельно
- Любой узел может упасть без дальнейшей потери данных
- Настраиваемое количество резервных копий
- Без вмешательства в разработку или инфраструктуру

Распределенная схема

Особенности

- Каждый узел имеет логический доступ ко всем записям
  - Не более одного сетевого обращения для операции доступа
  - Независим от размера кластера
- Линейная масштабируемость
  - Объем кэша увеличивается с размером кластера
  - Coherence выравнивает нагрузку разделения через кластер
  - Взаимодействие точка-точка
  - Multicast не требуется (иногда вообще не возможен)

Топология развертывания

Составные схемы

- Схемы могут быть оставлены, чтобы соответствовать специфическим системным требованиям или SLA.
  - Например, необходимо хранить наиболее используемые данные в памяти, а остальные на диске
- Базовые схемы
  - Local, Replicated, Distributed, Extend*
- Составные схемы:
  - Near, Overflow (запись на диск)
  - Позволяют подключать другие схемы

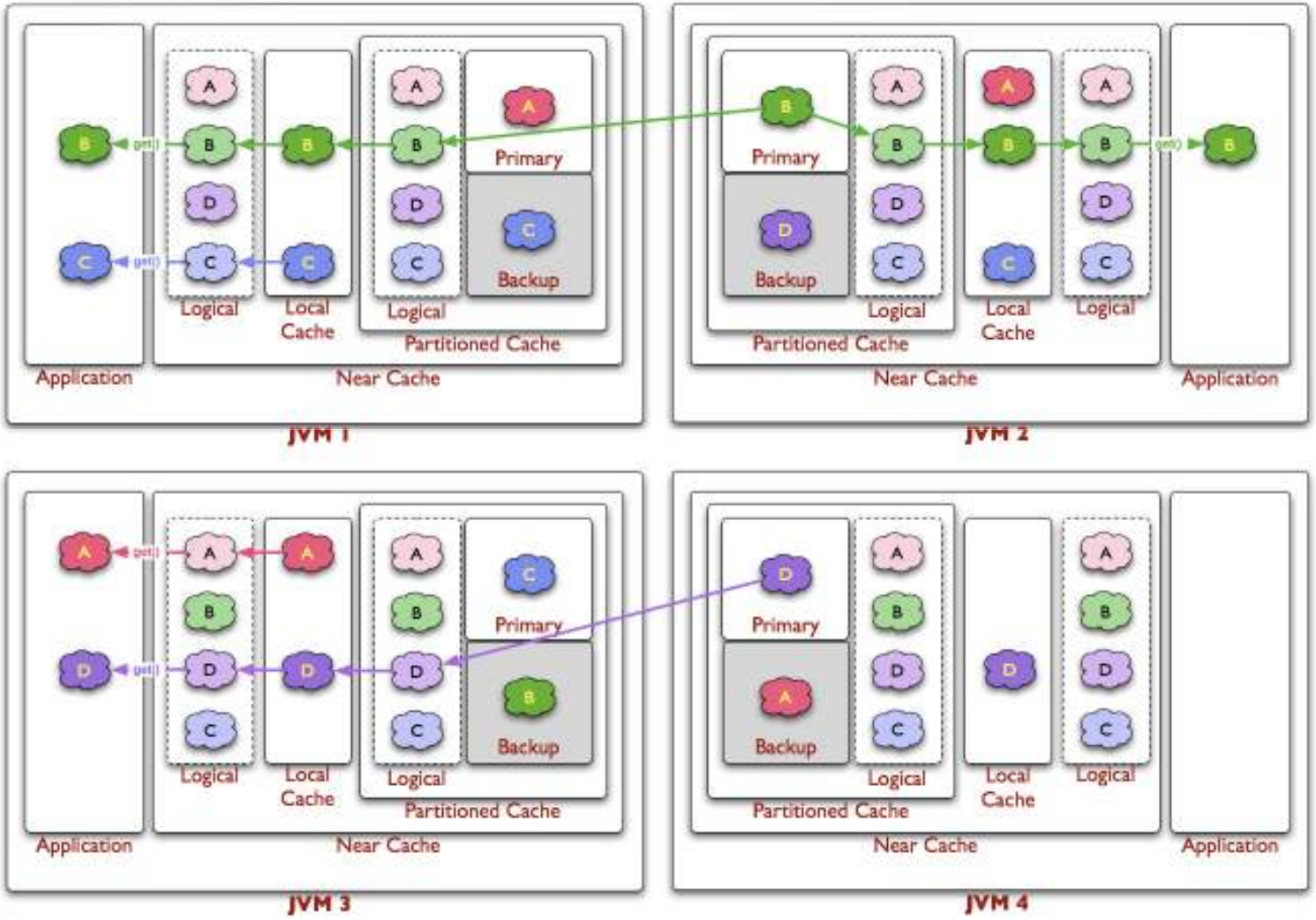
Топология развертывания

Ближняя схема (Near)

- Композиция подключаемых схем Front и Back
  - Обеспечивает кэширование уровня L1 и L2 (кэш кэшей)
- Зачем нужна:
  - Разделенная топология всегда передает данные по сети
  - Необходимость локального кэша (L1) через распределенную схему (L2)
  - Наилучшая опция для масштабируемой производительности!
- Как работает:
  - Настройка «Front» и «Back» топологий
- Настраиваемые политики истечения срока действия:
  - LFU, LRU, гибрид (LFU+LRU), По времени, Никогда, Подключаемая

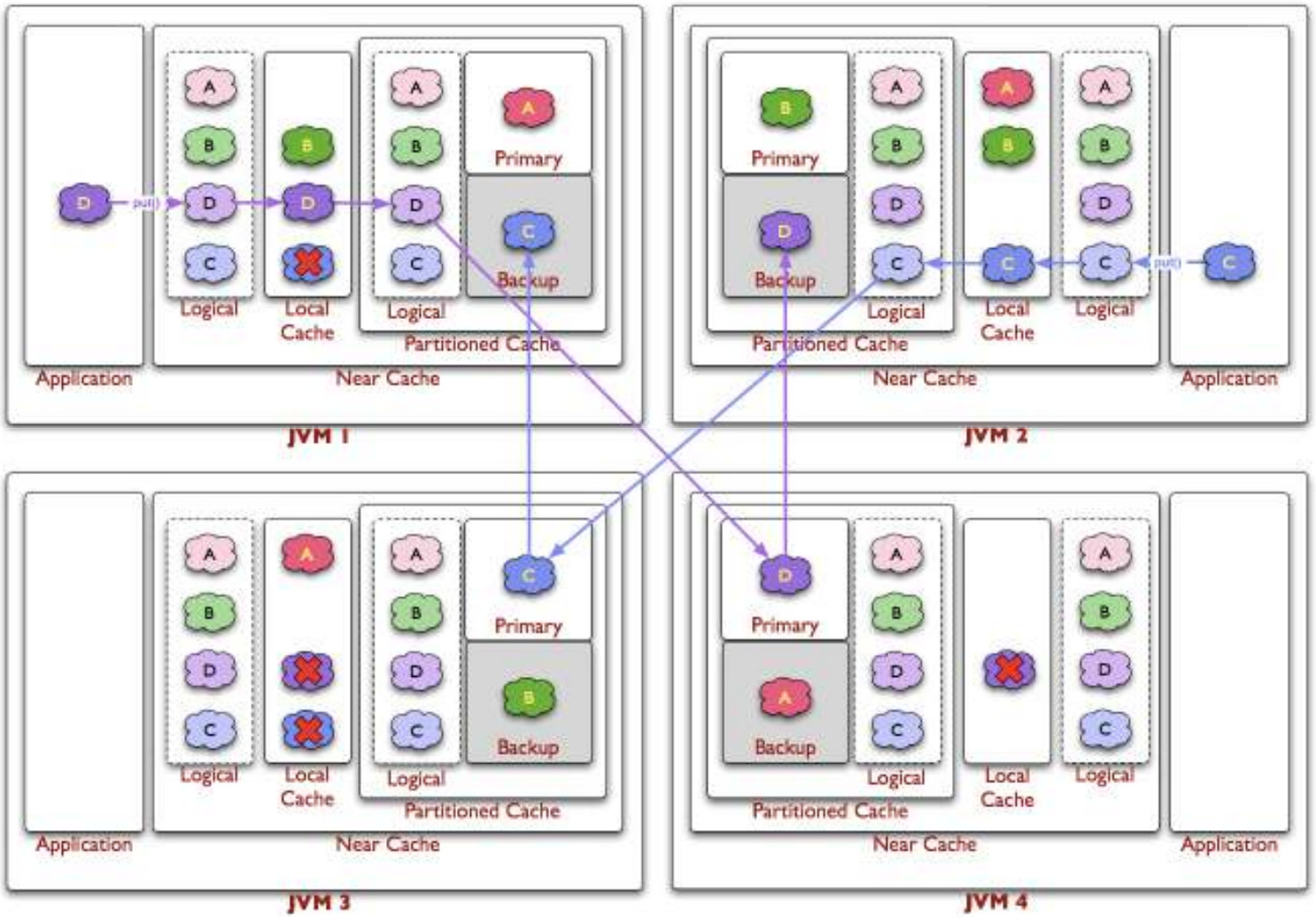
Ближняя схема

Чтение



Ближняя схема

Запись



Oracle Coherence

Доступ к данным

Доступ к данным

Механизм хранения данных

- Coherence (так же как и Berkeley DB) управляет данными парами ключ-значение (key-value) или ключ-объект (key-object)
  - Хранилища кэша: JVM Heap, NIO RAM, NIO Disk

  http://download.oracle.com/docs/cd/E14526_01/coh.350/e14510/overviewimplementors.htm#CIHDDDIH
- Ключ и объект могут быть любым типом данных
  - Может быть простой строковый тип, например String
  - Часто, это пользовательские типы данных или пользовательские объекты
- Пример вставок /запросов данных (эквивалент SQL)
  - `put(1, "Hello") = INSERT INTO table VALUES(1,'Hello');`
  - `get(1) = SELECT * FROM table WHERE id = 1;`
- Нет служебной схемы или словаря
  - Пользователь только должен определить объекты и типы данных
  - Приложение должно знать об этих структурах
- Нет SQL или другого языка запросов
  - Запросы пишутся внутри приложения, использую классы и их методы

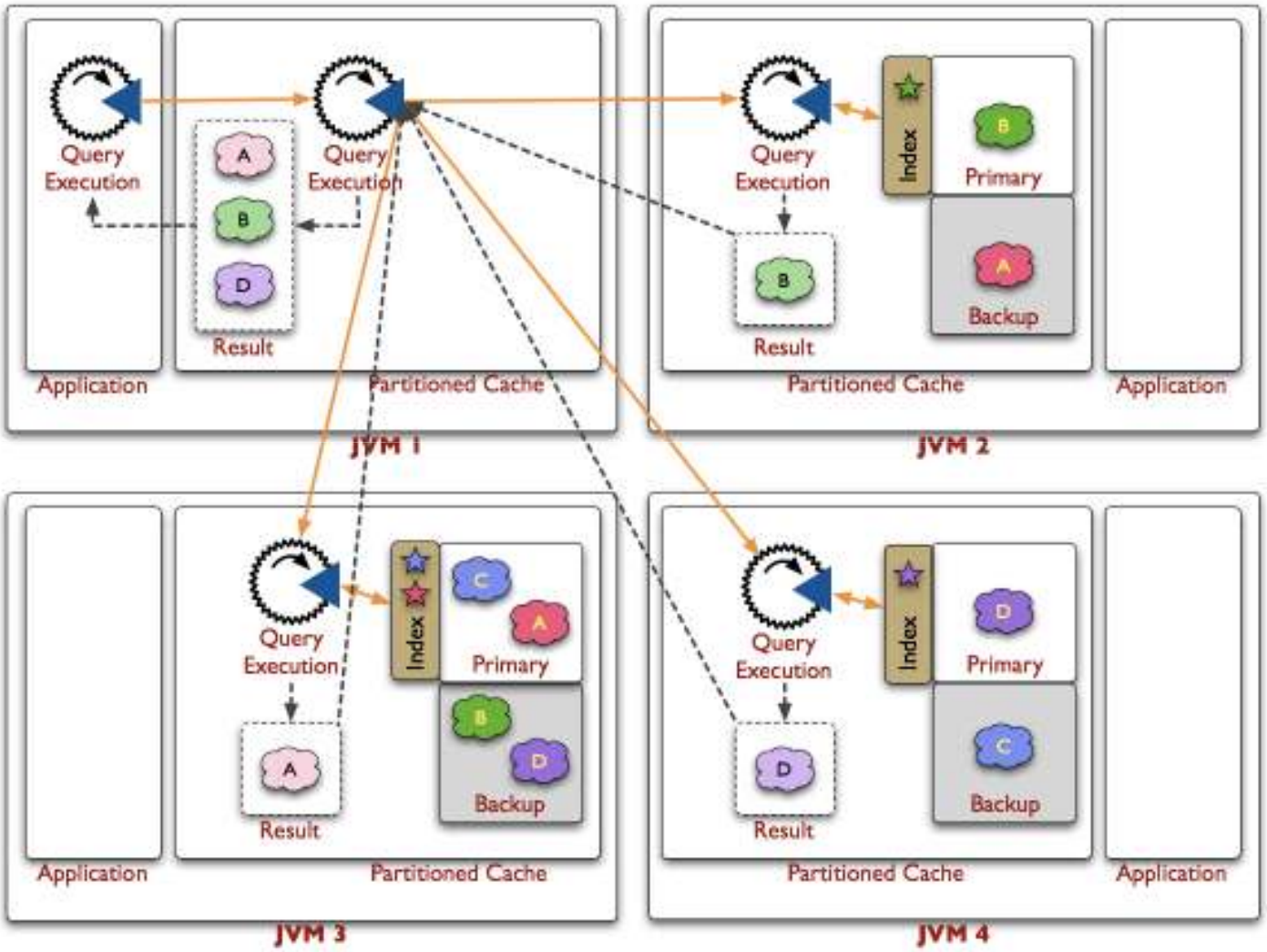
Доступ к данным

Класс NamedCache

- Разработчики на Java используют класс Coherence, который называется *NamedCache* (в .NET – называется *INamedCache*) для управления данными
  - NamedCache может использовать распределенный (Distributed) кэш или другую топологию
- NamedCache
  - Реализует класс `java.util.Map`
  - Логически эквивалентен таблице в базе данных
  - В приложении может быть сотни или тысячи различных экземпляров
  - Может создаваться динамически
  - Может содержать любой сериализуемый объект (нет необходимости настраивать схему для этого)

Запросы к кэшу

Интерфейс QueryMap



Запросы к кэшу

Интерфейс QueryMap

- Возвращает все записи из кэша «person», у которых атрибут «Gender» равен «MALE»

```
NamedCache person = CacheFactory.getCache(“person”);
Set males = person.entrySet(new
  EqualsFilter(“getGender", Person.MALE));
```

- Возвращает все записи, у которых атрибут «Age» больше или равен 35

```
Set personOver35 = person.entrySet( new
  GreaterEqualsFilter("getAge", 35));
```

- Возвращает все записи из кэша, у которых атрибут «Gender» равен «MALE» и атрибут «Age» больше или равен 35

```
Set malesOver35 = person.keySet(
  new AndFilter(
        new EqualsFilter("getGender", Person.MALE),
        new GreaterEqualsFilter("getAge", 35)));
```

Запросы к кэшу

Index, InvocableMap, Aggregator, Filter

- Есть возможность создать индексы (глобальны для кэша)

```
trades.addIndex(new
    ReflectionExtractor("getPortfolio"),
          true,  /* sort */
          null); /* optional comparator */
```

- Исполнение обработчиков в контексте записей, коллекций или фильтров
- Исполнение происходит параллельно (Grid-style)
- Нет специальных "демонов" для управления!
- Обработчики могут возвращать любой результат

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

- Агрегаторы записей основаны на фильтрах

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

Дополнения для

Oracle Coherence

Дополнения для Oracle Coherence

Coherence*Web

Распределенное управление HTTP-сессиями

- «Бесшовное» разделение сессий между приложениями
- Разделение сессий между серверами приложений WebLogic, OAS, WebSphere, JBoss*
- Обработка объемных сессий - хранение большого количества информации в сессии (возможно это ошибка проектирования приложения)

* http://download.oracle.com/docs/cd/E14526_01/coh.350/e14536/start.htm#CIAFFFEF

Coherence*Web

Когда и почему

- Отделение функции управления сессиями от Web-контейнера
  - Обработка большего числа пользователей без добавления новых серверов приложений
  - Перезагрузка/обслуживание приложений/контейнеров без потери состояния сессий
- Обработка очень больших сессий происходит очень эффективно
- Хранение данных в сессии надежно при высокой нагрузке

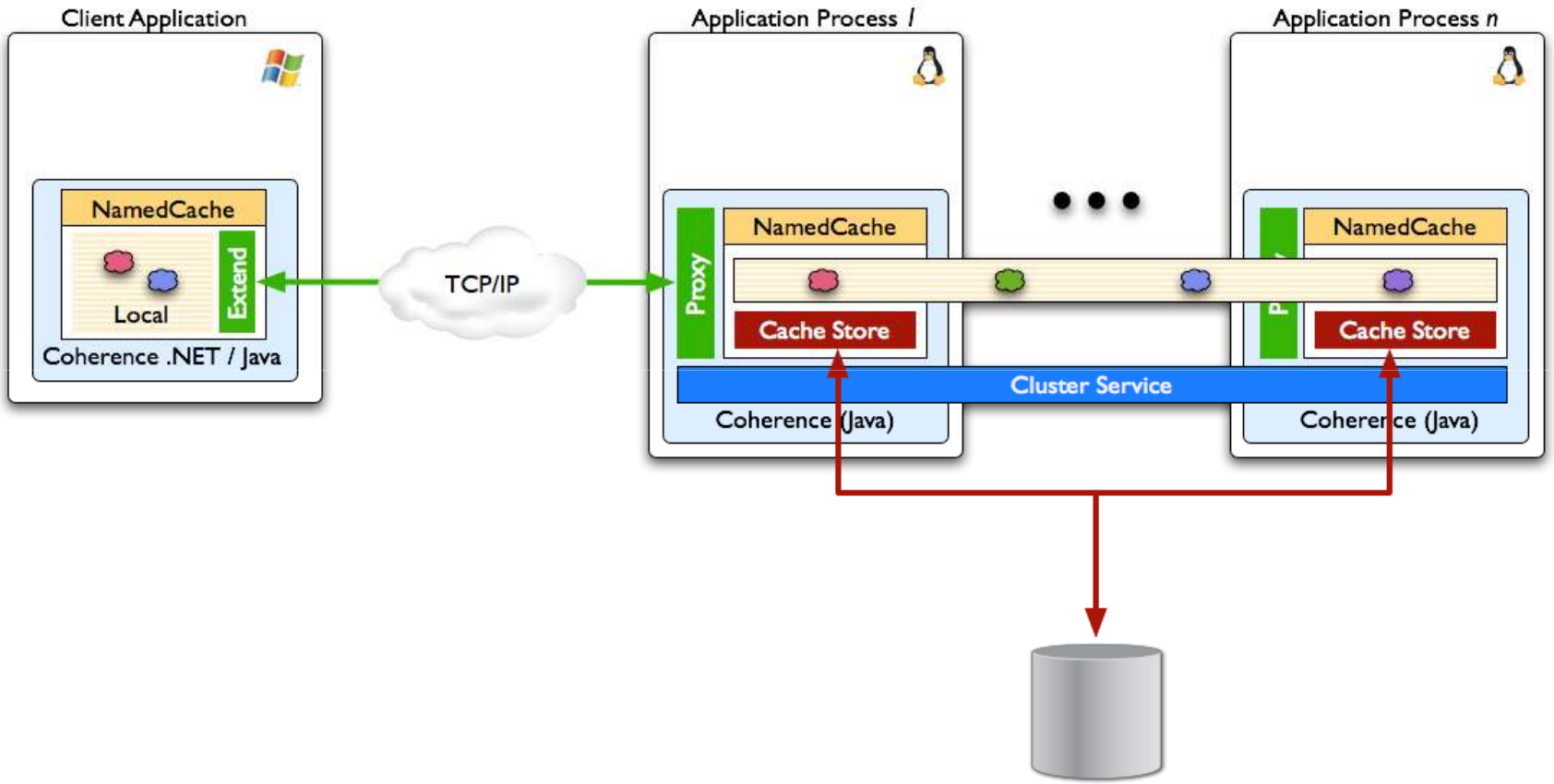
Дополнения для Oracle Coherence

Coherence*Extend

- Расширение Coherence*Extend позволяет клиентским приложениям присоединяться к кластеру по протоколу TCP/IP
- Это похоже на присоединения к базе данных в режиме клиент-сервера
  - Coherence включает в себя встроенный proxy server
  - Proxy server управляет TCP/IP соединениями
  - Клиент может быть настроен таким образом, что в случае отказа сервера, его запросы будут переброшены на другой IP адрес
- Используйте *Extend когда приложение …
  - Соединяется через WAN
  - Между приложением и кластером есть фаервол
  - Приложение написано не на Java (поддерживается C#, .NET, и C++)
- Так же необходимо для репликации одного кластера в другой
  - Может использоваться для аварийных сайтов
  - Требует высокой полосы пропускания сети, в зависимости от транзакционной загрузки системы

Coherence*Extend

Доступ к удаленному кластеру



Дополнения для Oracle Coherence

Coherence.NET

- Клиентская библиотека .NET 1.1 и 2.0 для Coherence*Extend
- «Родная» реализация .NET – нет необходимости запускать JVM через JNI
- Полный функциональный эквивалент клиенту на Java
- Так же включает в себя реализацию ASP.NET Session-State Store Provider для хранения объектов ASP.NET HTTP-сессий в Data Grid

Monsanto R&D IT

Oracle Coherence Strategy

Oct 1, 2012

Oracle Open World

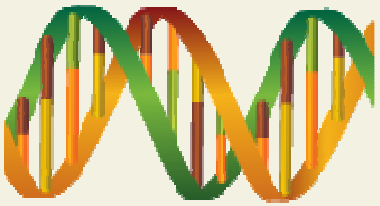
Our Approach to Driving Yield

A System of Agriculture Working Together to Boost Productivity

BREEDING

The art and science of combining genetic material to produce a new seed

BIOTECHNOLOGY



The science of improving plants by inserting genes into their DNA

AGRONOMICS

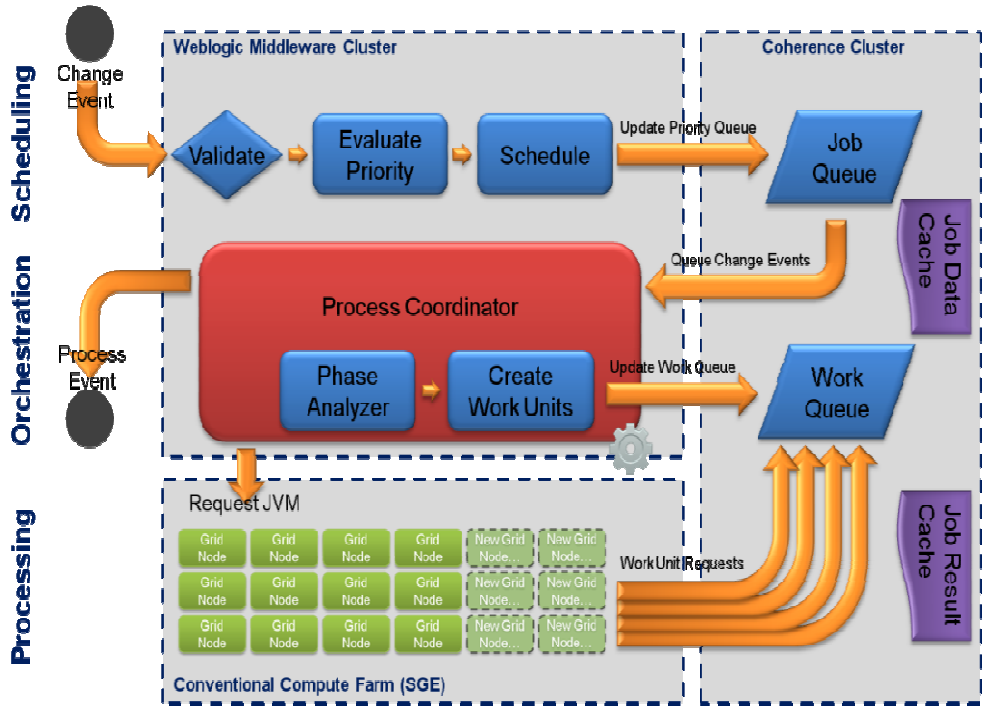
The farm management practices involved in growing plants

ORACLE

Distributed computation & event orchestration

Delivering high-throughput innovative solutions for R&D Harvest operations

- Technical Objectives
  - Elastic compute across 1000+ compute nodes
  - Fault tolerant work management
  - Database offload



- Business Value
  - Improve efficiency
    - Eliminate manual steps
    - ~30K hours man-time savings annually
  - Data availability for decision applications
    - Immediate analysis results
    - Reduce data turnaround time
      - 5-10 days to 1-2 days
    - Improves data traceability
    - Standardized analysis results

Copyright © 2011, Oracle and/or its affiliates. All rights reserved.
Confidential – Oracle Internal

Заключение

- Oracle Coherence мощное, кроссплатформенное, гибконастраиваемое и масштабируемое средство распределенных вычислений на основе технологии Data Grid
- Позволяет данным быть надежными, доступными, непротиворечивыми
- Имеет дополнения в виде Coherence*Web, Coherence*Extend и Coherence.NET

Информация

Где найти продукт
http://www.oracle.com/us/products/middleware/coherence/index.html

Oracle Partner Academy
http://www.oracle.com/ru/education/academy/index.html

mailto:grigoriy.sen@oracle.com

ORACLE IS THE INFORMATION COMPANY