GridGain – Java Grid Computing Made Simple

Denis Kharlamov
www.gridgain.org

Kiev JUG
Ukraine

Содержание

- GridGain
  - Что такое Грид?
  - Почему Грид?
  - Краткий обзор GridGain
  - Ключевые понятия
- Примеры
  - Грид приложение за 15 минут

Что такое грид?

- Вычислительный грид
  - Параллельное выполнение кода
- Распределенный кеш данных
  - Паралельный доступ к данным
- Грид = вычислительный грид + кеш данных.
  - Выполнение кода там где находятся данные

Почему Грид?

- Спросите Google, Amazon, eBay
- Решение проблем зачастую не имеющих другого решения
  - У Google примерно 1 000 000 нодов
- Универсальная парадигма программирования
  - Масштабирование от 2 до 1 000 000 компьютеров.

Краткий обзор GridGain

Open Source Java Grid Computing

- Грид
  - Инновационный вычислительный грид
  - Интеграция с лидирующими кешами данных
- Java
  - Создан на Java и для Java
- Open Source
  - LGPL и Apache 2.0

Уникальная простота и широчайшие возможности

Профессиональный Open Source

- GridGain - Профессиональный Open Source
  - Свободный и основан на Open Source лицензиях: LGPL and Apache 2.0
  - Профессиональная поддержка, обучение и консультации.
- Лучная бизнес модель для масштабируемых серверов
- Подобно JBoss, Spring Source, Mule Source...

Статистика GridGain

За 9 месяцев с первого резиза:

- Более 15,000 скачиваний
- Запуск новой ноды каждые 60 секунд
- Более 2000 различных организаций, проектов и частных лиц

Наиболее быстро растущий Java Грид.

Ключевые понятия

- Map/Reduce
- Zero Deployment
- Масштабируемость
- Гарантированное выполнение
- Интеграция в стиле LEGO
- Выполнение без изменения кода
- Интеграция с ведущими кешами
- JMX мониторинг

MapReduce

Особенности:

- API для Map/Reduce
- Рспределенная сессия
- Выполнение с аннотациями
- Асинхронное выполнение
- Избыточные подзадачи
- Частичная и асинхронная обработка результата
- Адаптивное разбиение на подзадачи
- Сохранение промежуточных результатов.
- Балансировка нагрузки на всех этапах выполнения
- Выполнение кода там где находятся данные

1. Запрос на выполнение задачи
2. Разбиение на подзадачи
3. Результат выполнение подзадач
4. Сборка результата

Zero Deployment

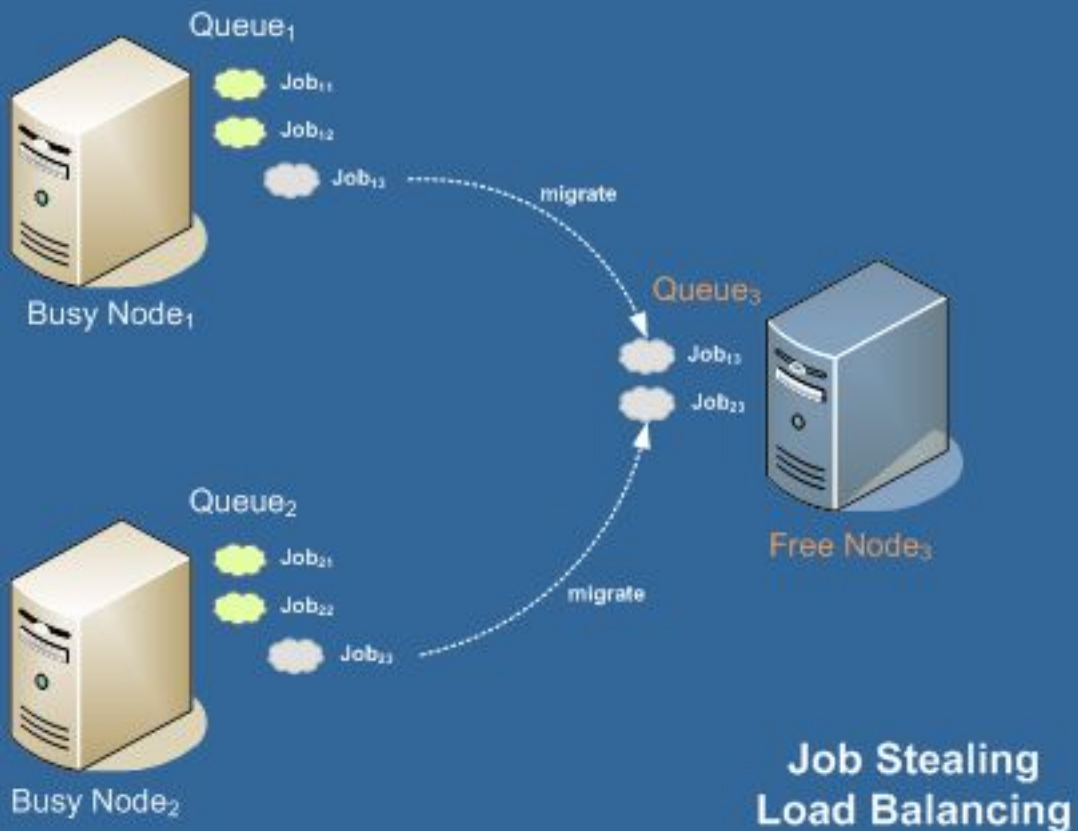
- Загрузка пользовательских классов по требованию
  - Никаких скриптов
  - Никаких установок по FTP
  - Никаких перезапусков нодов
- Разработка ТОЧНО так же как обычно
  - Изменяй->Компилируй->Выполняй на гриде
- Запускай несколько нод в
  - Той же JVM – отлаживай код локально (!)
  - Том же компьютере – Запускай грид на рабочей станции

Масштабируемость

- Балансировка нагрузки на всех этапах выполнения:
  - Оптимально при возникновении перегрузок на нодах.
- Load Balancing SPI
  - «Ранняя» балансировка
- Collision SPI
  - «Поздняя» балансировка

=> Максимальная поддержка масштабируемости

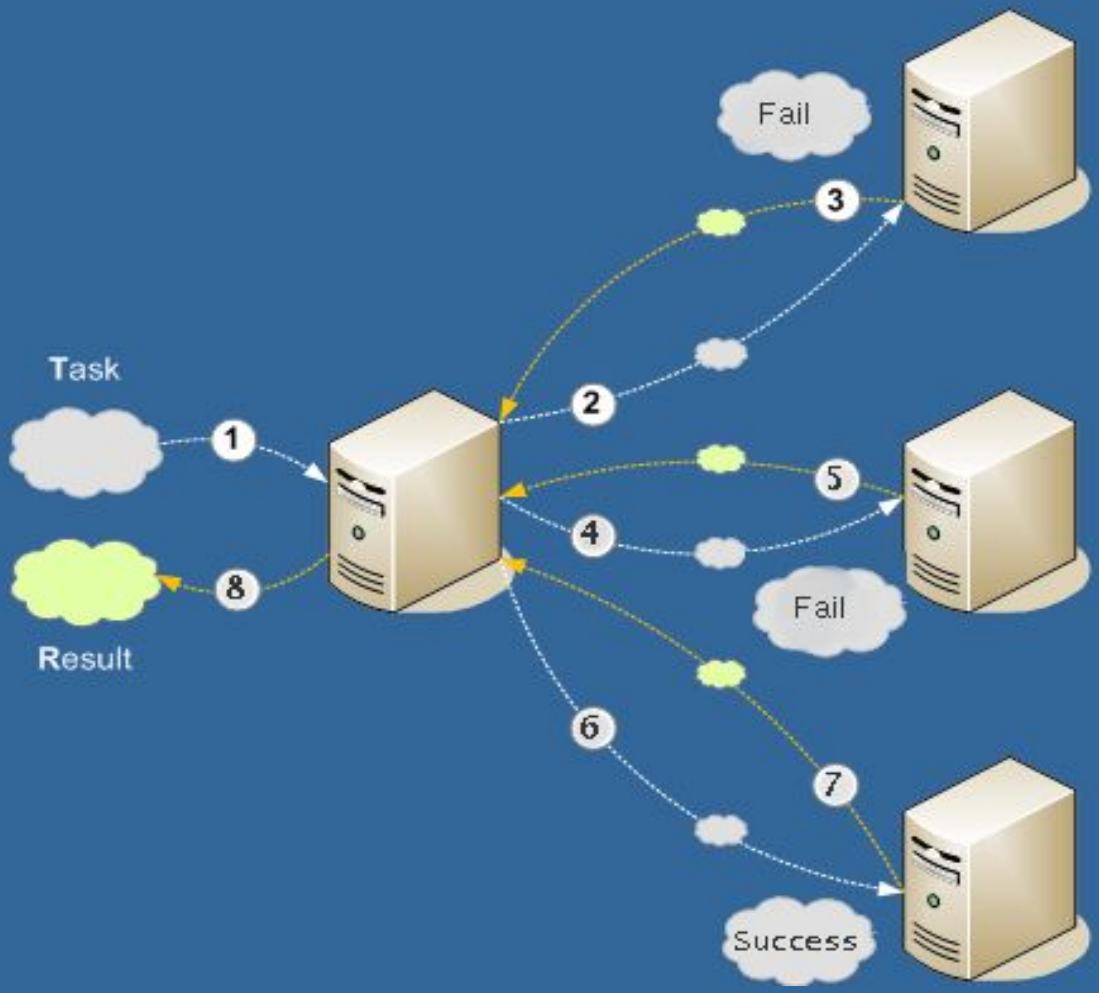
Масштабируемость



Гарантированное выполнение

- Ошибка тоже результат
- Продолжение выполнения в случае ошибки
  - Решение о продолжении выполнения основанное на “политиках”
- Асинхронная обработка результата
- Checkpoint для задач длящихся долго
  - “Умное” выполнение в случае ошибки
- **=>** Всеобъемлющая обработка ошибок

Гарантированное выполнение



Интеграция в стиле LEGO

- Service Provider Interface (SPI) архитектура
  - Подключать и настраивать практически любую функциональность грида так же просто как собрать LEGO.
- Следующая функциональность может быть изменена и сконфигурирована:
  - Communication
  - Discovery
  - Tracing
  - Startup
  - Event storage
  - Marshalling
  - OnDemand
  - Checkpoints
  - Failover
  - Collision Resolution
  - Topology management
  - Load balancing
  - Deployment

Интеграция в стиле LEGO

"Готовая" интеграция с:

Application Servers
- JBoss AS
- BEA Weblogic
- IBM Websphere
- Glassfish
- Tomcat

Data Grids
- JBoss Cache
- Coherence
- GigaSpaces

AOP
- JBoss AOP
- Spring AOP
- AspectJ

Messaging Middleware
- Mule
- JMS
  - ActiveMQ
  - SunMQ
- Jgroups
- Email
- TCP, IP-Multicast

Другое
- Spring
- Junit
- JXInsight

Выполнение без изменения кода

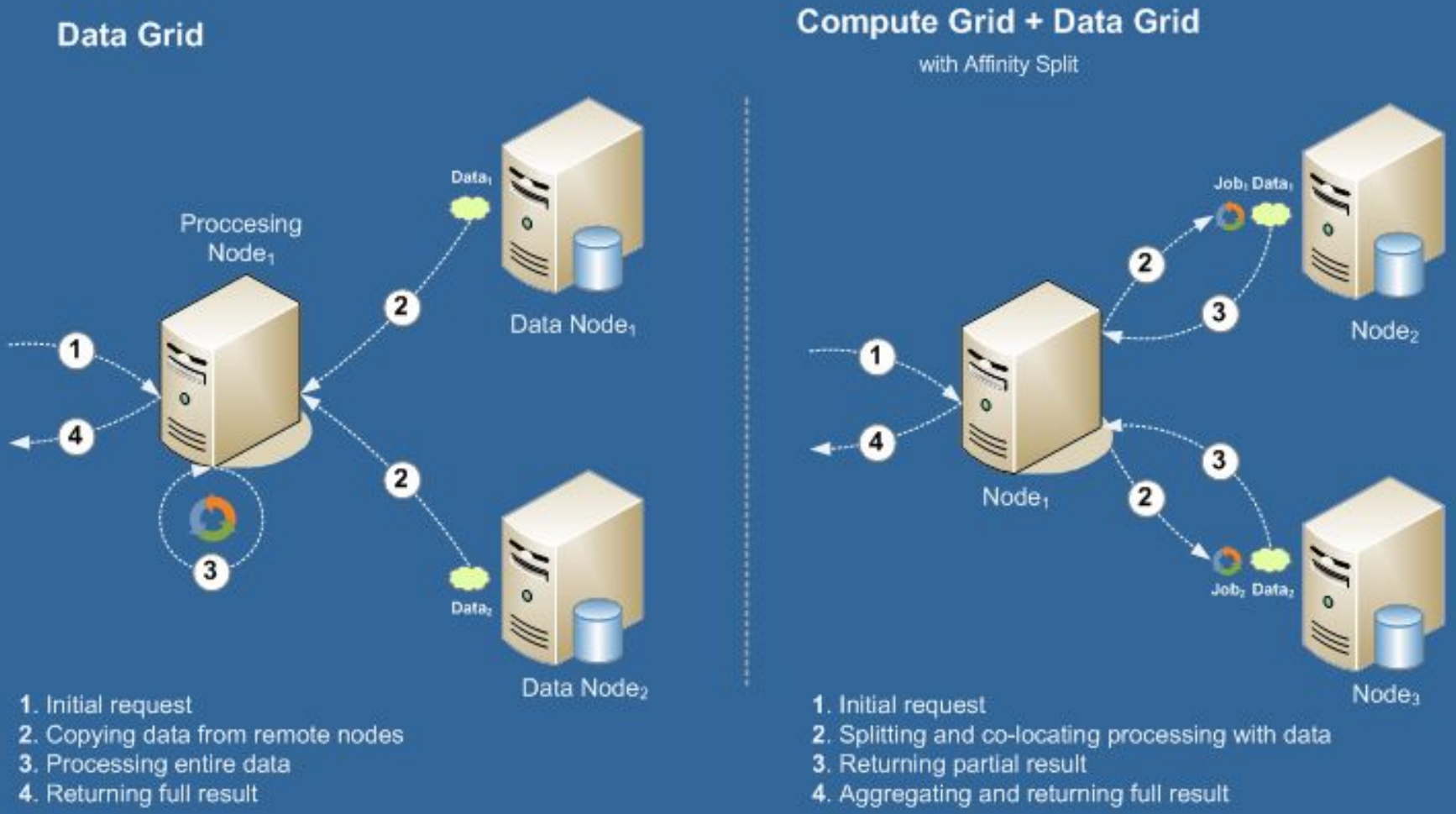
```
01 class BizLogic {
02   @Gridify(...)
03   public static Result process(String param) {
04     ...
05   }
06 }
07
08 class Caller {
09   public static void Main(String[] args) {
10    GridFactory.start();
11
12    try {
13      BizLogic.process(args[0]);
14    }
15    finally {
16      GridFactory.stop();
17    }
18  }
19 }
```

Execution of process() method will be performed on the grid

Интеграция с кешами данных

- Интеграция с кешами – ключ к беспрецедентной масштабируемости
- Affinity Map/Reduce – возможность выполнить код там где находятся данные
  - Минимизирует “избыточный” трафик
  - Оптимальная балансировка и производительность
- Готовая поддержка:
  - JBoss Cache
  - Oracle Coherence

Интеграция с кешами данных



JMX мониторинг

- JMX бины для
  - Каждого SPI
  - Ядра
  - Публичных APIs
- Гибкий доступ
  - Программный используя JMX API
  - Из GUI JMX консоли
    - Jboss Management
    - Hyperic
    - Jconsole/VisualVM

Что же дальше?

- GridGain 1.5 - Июль 2007
- GridGain 2.0 - Февраль 2008
- GridGain 3.0 - Q109
  - Мобильный грид: Google Android
  - Грид по требованию: Amazon EC2
  - Web 2.0 интеграция: REST + JSON
  - Консоль для управления и мониторинга

Пример

- Java 5/Eclipse 3.3/Linux
- GridGain 2.0

Q & A

Спасибо!

Денис Харламов:
dkharlamov@gridgain.com
GridGain: www.gridgain.org