Public

АСИНХРОННИ ОБРАБОТЧИЦИ НА СЪБИТИЯ (ASYNCHRONOUS EVENT PROCESSORS)

РЕАЛИЗАЦИЯ НА JAVA – VERT.X

Емил симеонов
Senior Product Manager & Development Architect
Bulpros Consulting AD

МРЕЖОВО ПРОГРАМИРАНЕ - НАБЛЮДЕНИЯ

- Независимо дали използваме java.io или java.nio се сблъскваме с трудности
  - Сами по себе си и двата пакета изискват работа на доста ниско ниво и изискват пълно разбиране на
    - OSI модела
    - TCP, UDP, IP протоколите
    - Сериозни познания в областта на операционните системи и тяхната еволюция
  - Управлението на жизнения цикъл на сървъри и клиенти не е лесно - ще са необходими и JVM hooks…
  - Не сме разглеждали конкурентна обработка на клиентски заявки – още една димeнсия на „Сложност“

Ще търсим друг подход.
Рамка (framework) от по-високо ниво би ни помогнала с тези роблеми.

ОТ КАКВО СЕ НУЖДАЕМ?

- Възможност за работа с TCP, UDP и IP протоколите по по-лесен начин, с по-малка възможност за грешка
- Поддръжка на неблокиращи и блокиращи сокети – производителността и скалируемостта са важни за нас
- Работата с подобен компонент трябва да ни СПЕСТЯВА УСИЛИЯ и да е ЗАБАВНА – основни изисквания за висока продуктивност
- Разбира се, търсим Java технология. Подобни концепции не са чужди и в JavaScript, C, .NET и др.

ЗАПОЗНАЙТЕ СЕ С VERT.X!

Асинхронен модел на разработка

- Vert.x е базиран на Netty.io
- Изцяло асинхронен
- Работата се основава на «излъчване» и «обработка» на събития

Хоризонтална скалируемост

- Клъстеризация на 1..N машини
- EventBus за комуникация на свързаните Vert.x “инстанции

Поддържа работа със следните протоколи

TCP, UDP, IP, HTTP

Модулярен

- Модулярна архитектура, вкл. сигурност
- Основа за Cloud micro services
- Интеграция с множество open source проекти

Вертикална скалируемост

- Ефикасна употреба на CPU ядрата
- Силно опростен многонишков модел на изпълнение – скрит

ПРИМЕР 1: ECHO TCP СЪРВЪР С VERT.X

```
public class EchoNonBlockingTCPServer extends AbstractVerticle {
    @Override
    public void start() throws Exception {
        Vertx.vertx().createNetServer().connectHandler(
                sock -> {
                    System.out.println("Processing incoming connection. Pumping out everything...");
                    Pump.pump(sock, sock).start();
                }
        ).listen(TCPConstants.DEFAULT_PORT);
        System.out.println("Simple Echo non-blocking TCP server is started.");
    }
    public static void main(String[] argv) throws Exception {
        new EchoNonBlockingTCPServer().start();
    }
}
```

ПРИМЕР 2: ПРОСТ TCP СЪРВЪР С VERT.X

```
public class SimpleNonBlockingTCPServer extends AbstractVerticle {
    public static void main(String[] argv) {
        Vertx.vertx().deployVerticle(new SimpleNonBlockingTCPServer());
    }

    @Override
    public void start() throws Exception {
        vertx.createNetServer().connectHandler(sock -> {
            sock.handler(in -> {
                String msg = in.getString(0, in.length());
                Buffer out = Buffer.buffer();
                out.appendString("You sent me this message: ").appendString(msg);
                sock.write(out);
            });
        }).listen(TCPConstants.DEFAULT_PORT);
        System.out.println("Simple non-blocking TCP server is started.");
    }
}
```

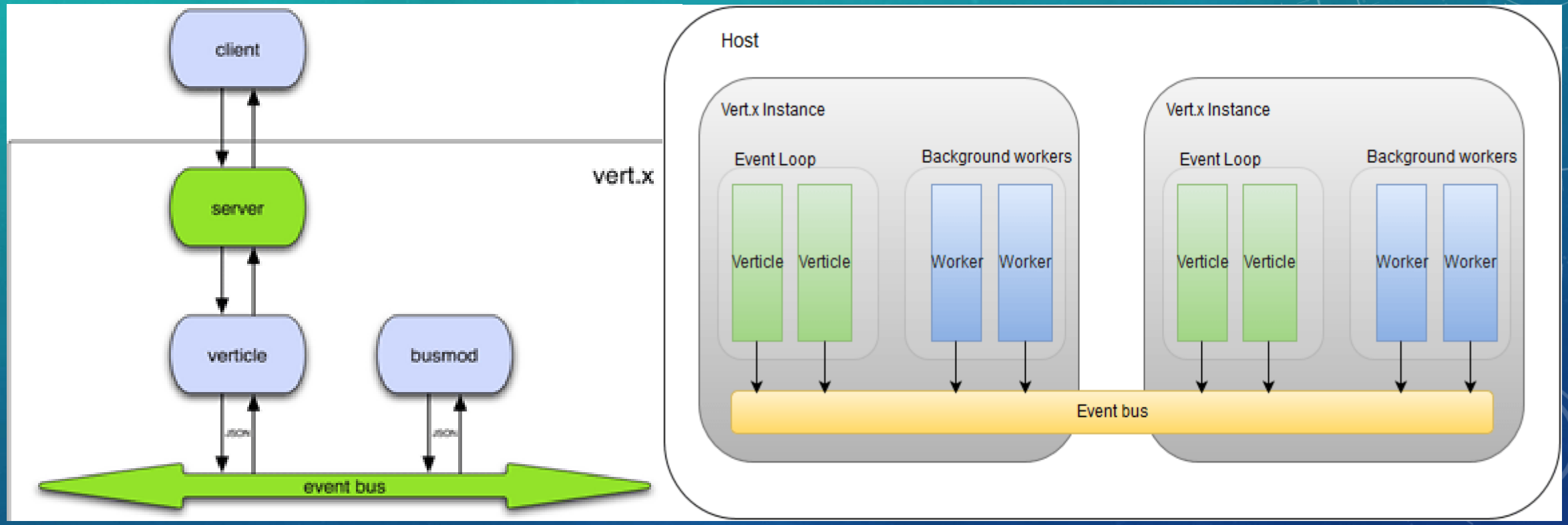
ВЪВЕДЕНИЕ ВЪВ VERT.X

- Vert.x разчита на асинхронни програмни модели на разработка и изпълнение
- Основните градивни елементи са, т. нар., „вертикали“, които
  - Енкапсулират бизнес логиката по даден сценарий
  - Могат да се добавят и премахват динамично (включително и от други вертикали)
  - Са „реактивни“ – те се активират, когато следва да обработят дадено събитие
  - Винаги се изпълняват в една и съща нишка (thread)
- Асинхронната комуникация между N на брой вертикали се подсигурява чрез шина за обмяна на съобщенията или още – Event Bus

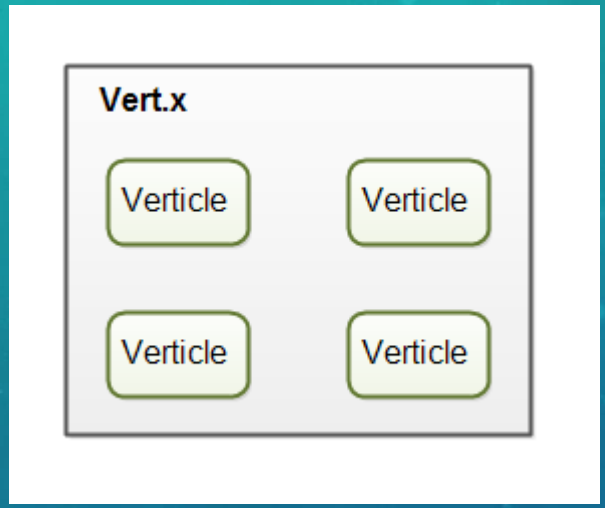
Vert.x е рамка (framework) за обработка на асинхронни събития или asynchronous event processor.

Друг изтъкнат представител на този клас софтуерни решения е node.js (JavaScript).

VERT.X – ПРИНЦИПНА АРХИТЕКТУРА

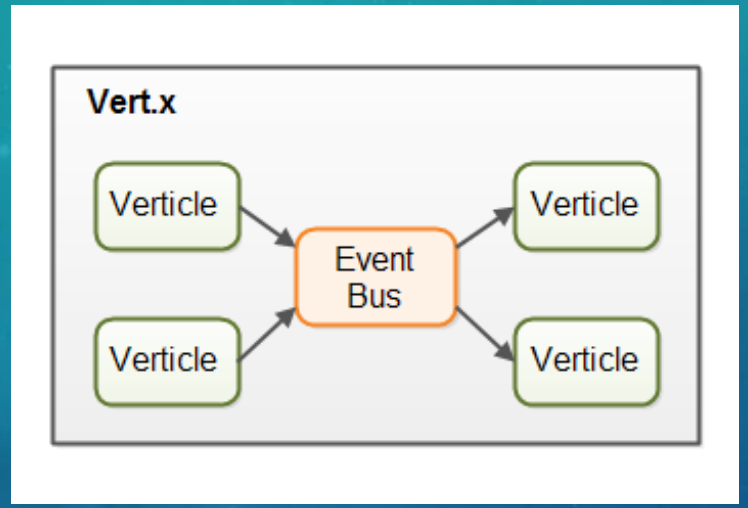


ОСНОВНИ КОНЦЕПЦИИ ВЪВ VERT.X



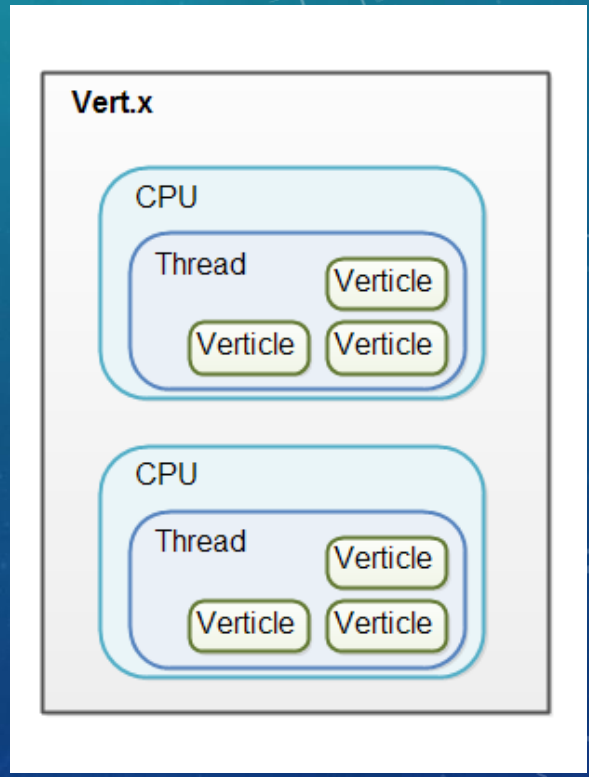
Вертикали

Могат да се добавят и премахват динамично. Всяка вертикала е самостоятелна услуга.



Шина за асинхронна комуникация (Event Bus)

Дава възможност за размяна на асинхронни съощения м/у вертикали чрез pub/sub и p2p механизми.



Ефикасна употреба на CPU ядра - еднонишкови вертикали

ПРИМЕР 3: КОМУНИКАЦИЯ ЧРЕЗ EVENT BUS

```
public class PubSubEventBusCommunication {
    public static void main(String[] argv) throws InterruptedException {
        Vertx vertx = Vertx.vertx();
        vertx.deployVerticle(new Subscriber());
        vertx.deployVerticle(new Subscriber());
        vertx.deployVerticle(new Publisher());
    }
    public static class Publisher extends AbstractVerticle {
        @Override
        public void start() throws Exception {
            EventBus eb = vertx.eventBus();
            vertx.setPeriodic(2000, lv -> eb.publish(EventBusConstants.TOPIC_NAME, "I am sending some news."));
        }
    }
    public static class Subscriber extends AbstractVerticle {
        @Override
        public void start() throws Exception {
            EventBus eb = vertx.eventBus();
            eb.consumer(EventBusConstants.TOPIC_NAME, msg -> {
                System.out.println(String.format("%d received news: \"%s\"", hashCode(), msg.body()));
            });
        }
    }
}
```

ДОПЪЛНИТЕЛНИ МАТЕРИАЛИ

- Примери от лекцията: https://github.com/emil-simeonov/edu/tree/master/tu-sofia/modern-java-technologies/vert.x/vertx-examples
- Въведение в асинхронния програмен модел: http://www.i-programmer.info/programming/theory/6040-what-is-asynchronous-programming.html
- Проекта Vert.x: http://vertx.io/
- Документация на Vert.x: http://vertx.io/docs/
- Vert.x Tutorial: http://tutorials.jenkov.com/vert.x/index.html
- Съпоставка на Vert.x с Node.js: http://www.cubrid.org/blog/dev-platform/inside-vertx-comparison-with-nodejs/
- Въведение в архитектурата на Vert.x: http://www.cubrid.org/blog/dev-platform/understanding-vertx-architecture-part-2/
- Netty 4.1 Javadoc: http://netty.io/4.1/api/index.html

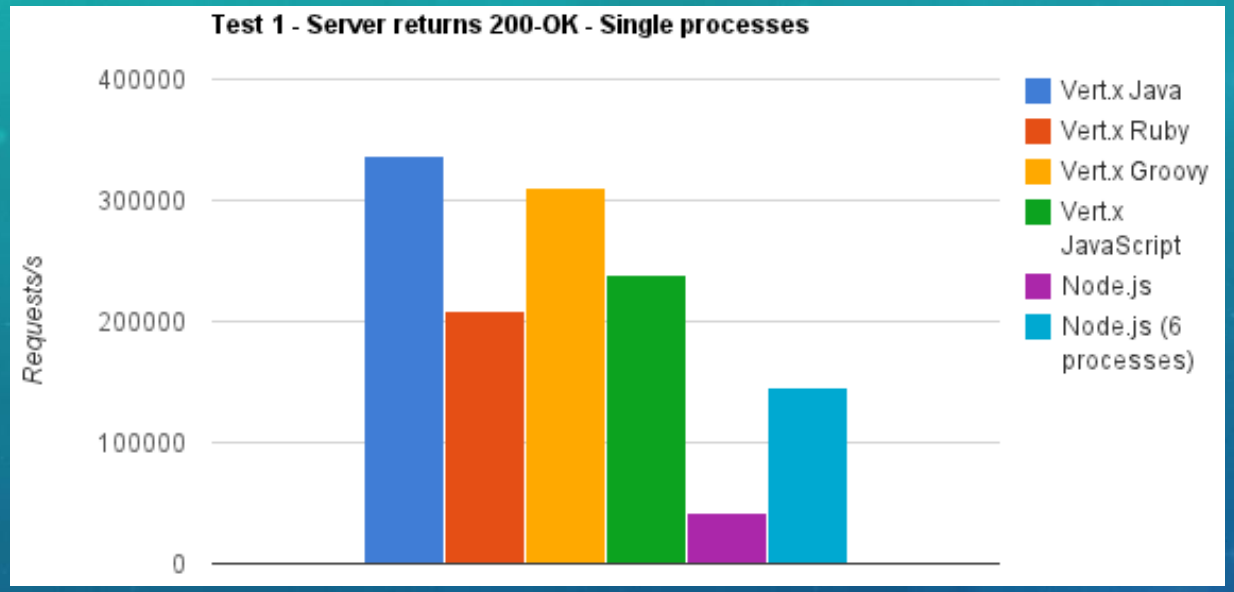
Благодаря Ви!

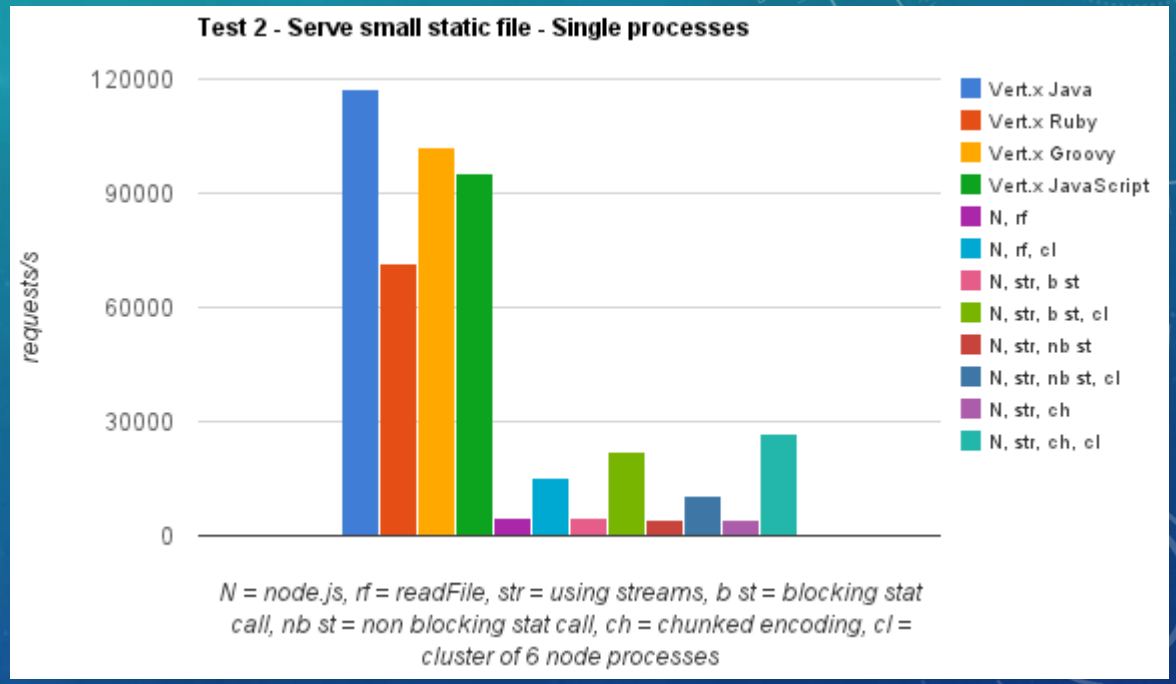
За контакти:

Емил Симеонов, emil.simeonov@bulpros.com

Senior Product Manager & Development Architect

VERT.X VS. NODE.JS - ПРОИЗВОДИТЕЛНОСТ





http://www.cubrid.org/blog/dev-platform/inside-vertx-comparison-with-nodejs/

BULPROS