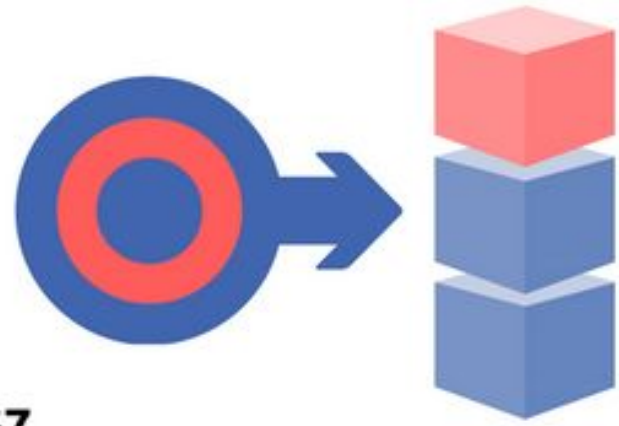
VIGOJUG PRESENTA:

DESARROLLO DE MICROSERVICIOS CON SPRING BOOT

Ponente

VITOR FERNANDEZ

Senior Analyst at Optare Solutions



INSCRICIÓN

HTTPS://TINYURL.COM/VIGOJUG7

Vítor Fernández

Senior Analyst at Optare Solutions

Optare Solutions • Universidad de Vigo

Ourense Area, Spain

@vfdiaz

UNIVERSIDADE DE VIGO

ESCOLA SUPERIOR DE ENXEÑARÍA INFORMÁTICA

EXPERIENCES

Honeywell

Hochschule Albstadt-Sigmaringen

Albstadt-Sigmaringen University

2008

CTAG

2009-2010

IBM

vodafone

MÁSMÓVIL

Optare Solutions

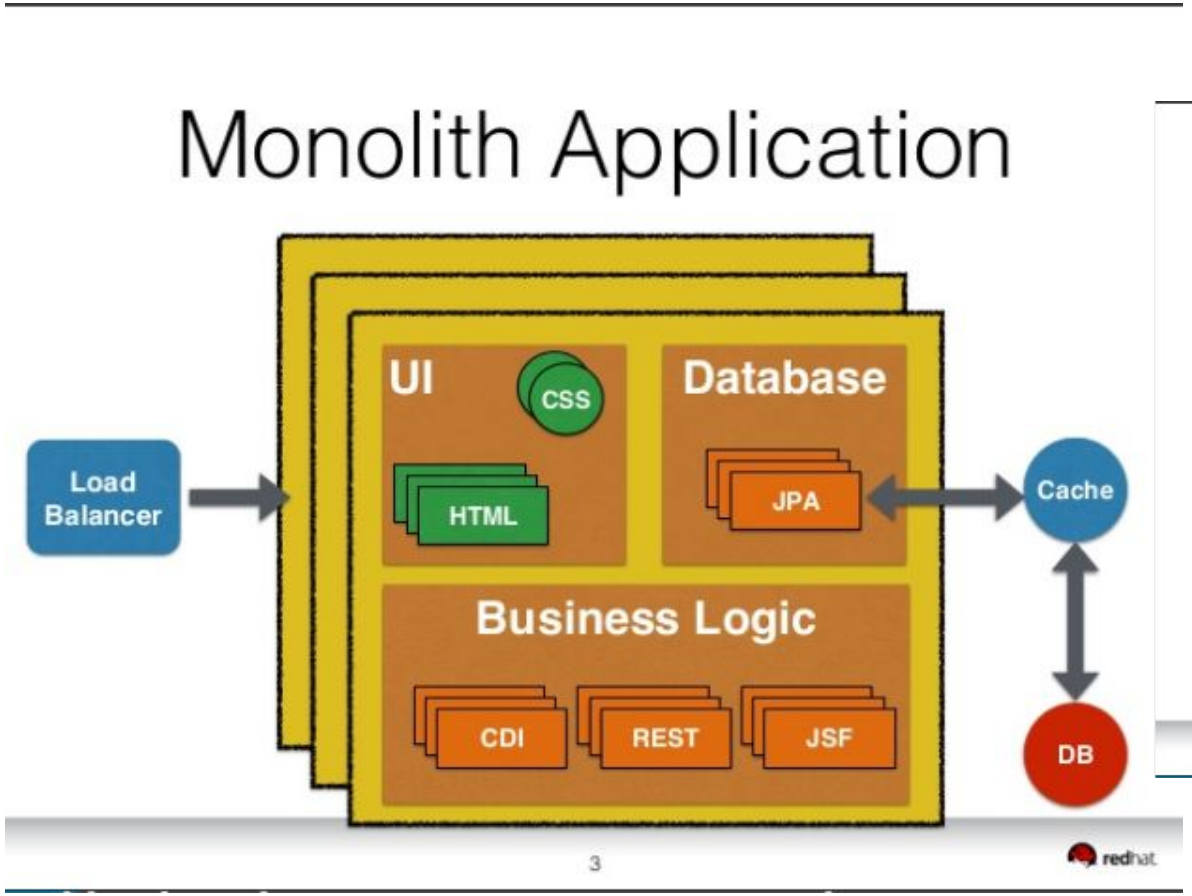
2011-...

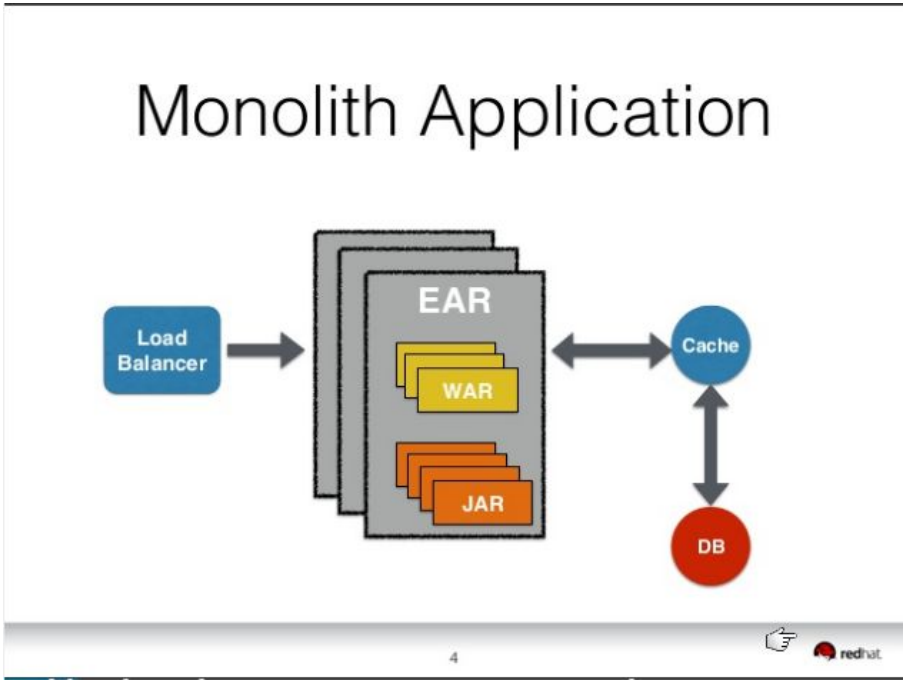
Desarrollo de Microservicios con Spring Boot

- ¿De dónde venimos?
- Conceptos sobre Microservicios
- Introducción a Spring Boot
- Práctica
- ¿Por qué Spring Boot?

¿De dónde venimos?

Hace ¿mucho? tiempo, en un entorno muy muy lejano...



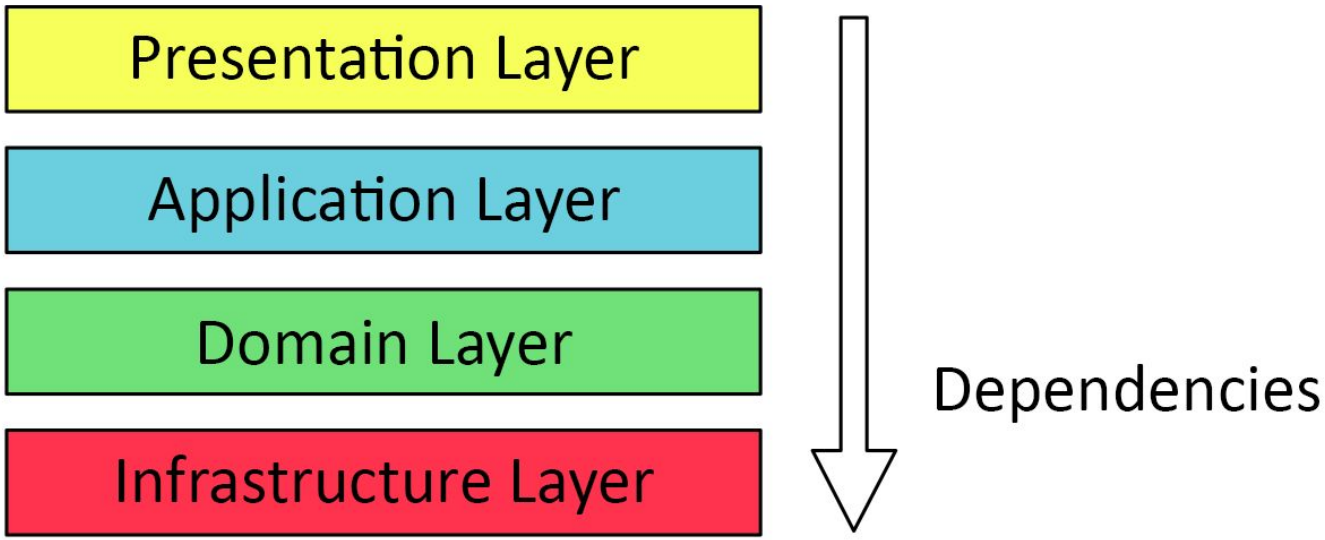


https://www.slideshare.net/Codemotion/refactor-microservices

Aplicaciones “Monolito”

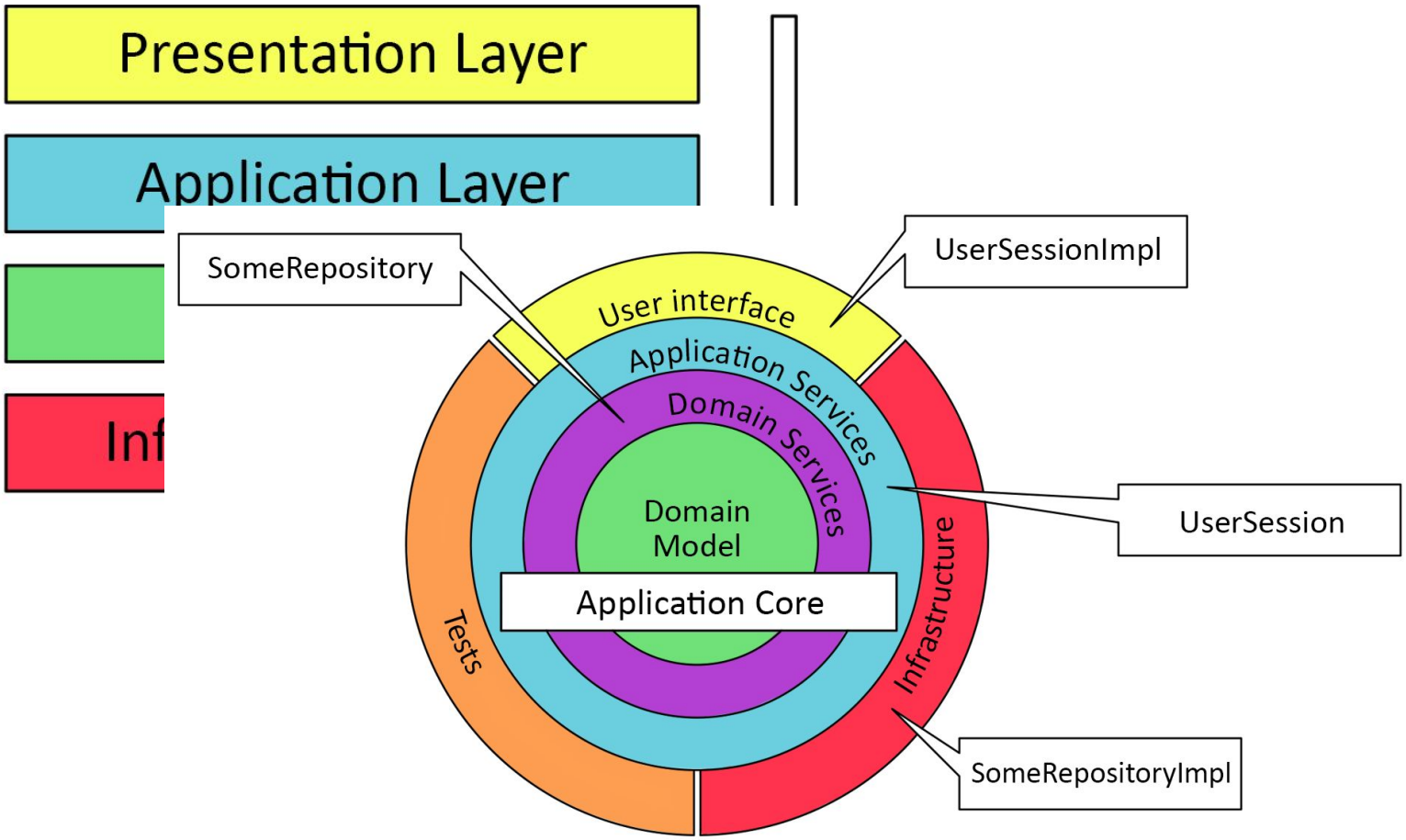
@vfdiaz

Tipos de arquitectura: Layered



Hay muchas “arquitecturas”

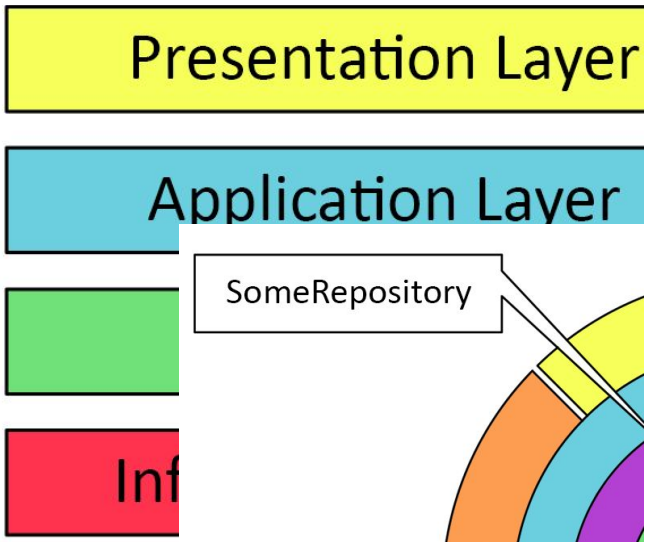
Tipos de arquitectura: Onion

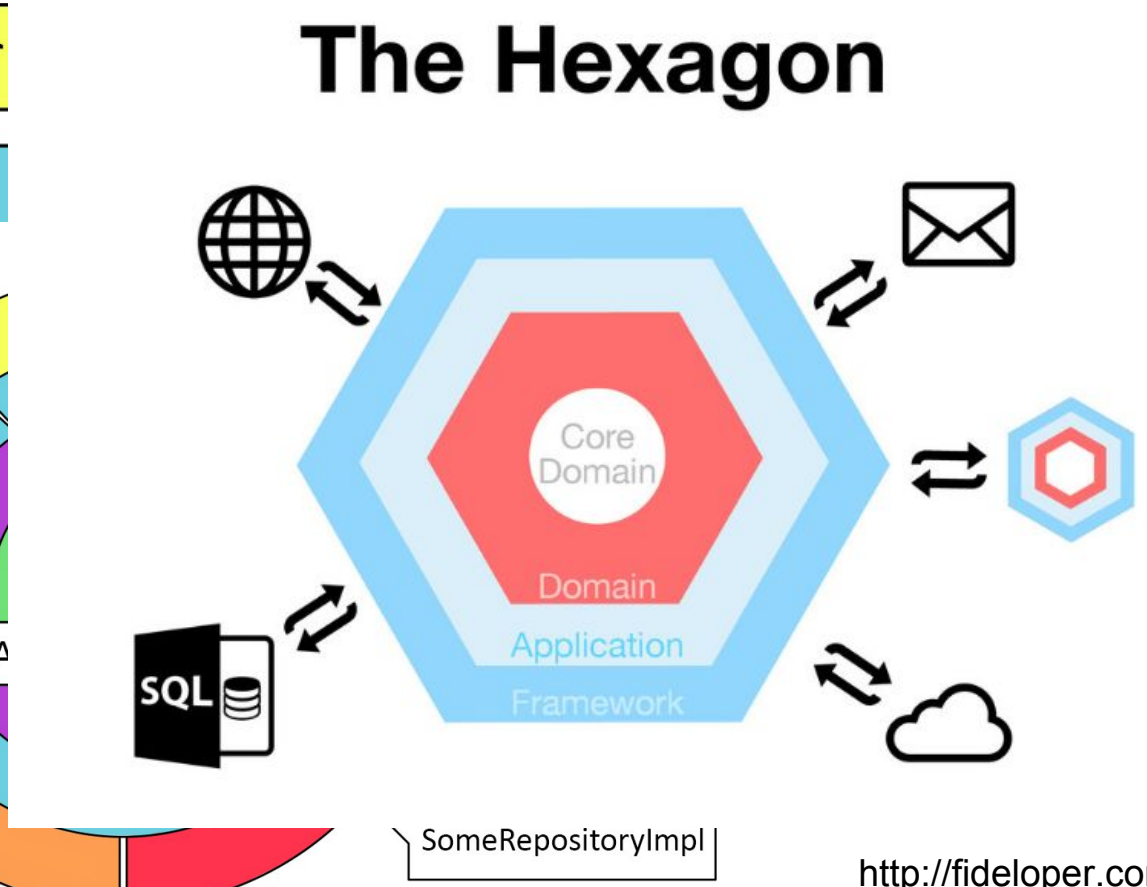


Hay muchas “arquitecturas”

@vfdiaz

Tipos de arquitectura: Hexagon



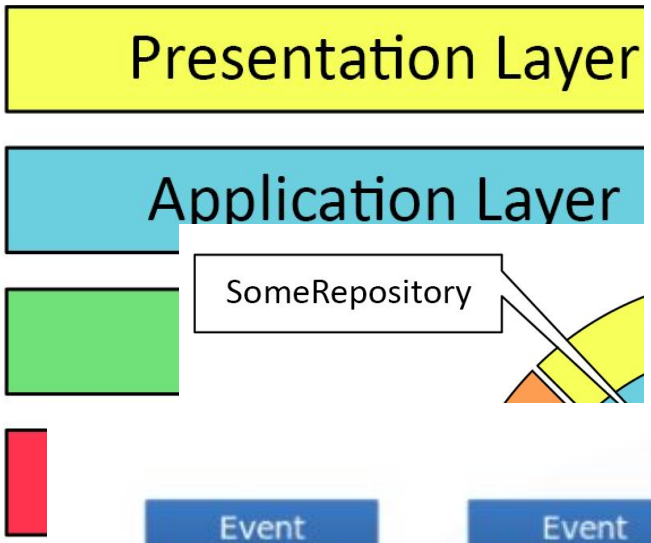


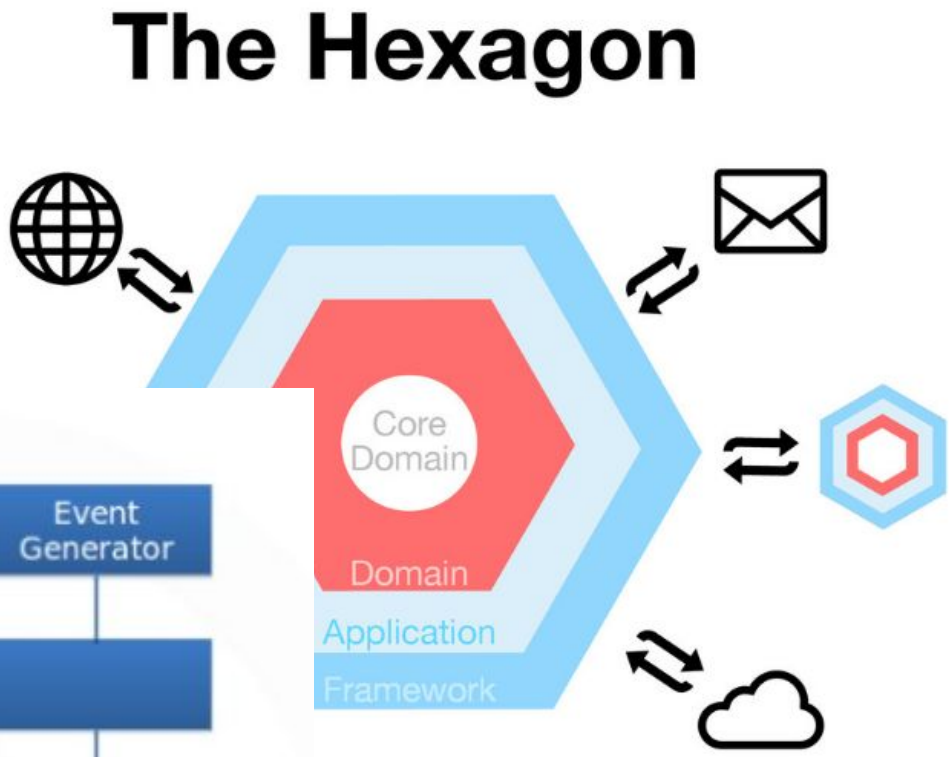
http://fideloper.com

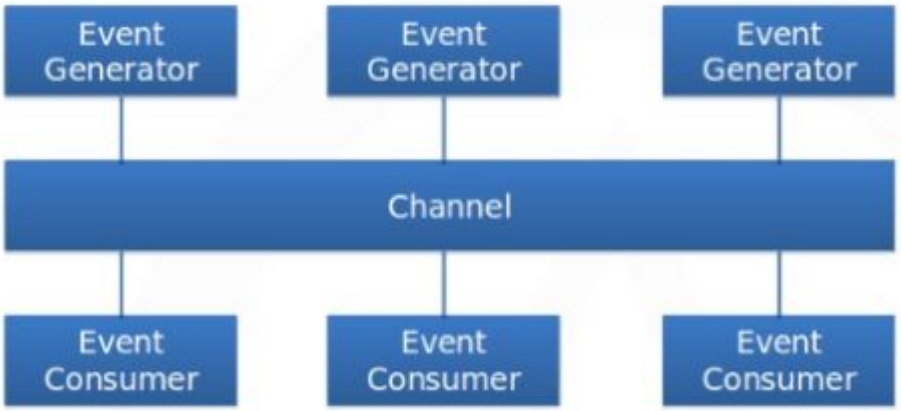
Hay muchas “arquitecturas”

@vfdiaz

Tipos de arquitectura: EDA (Event-Driven Arch.)

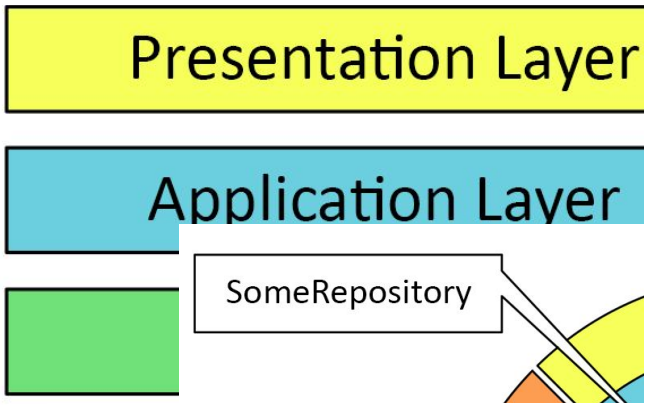


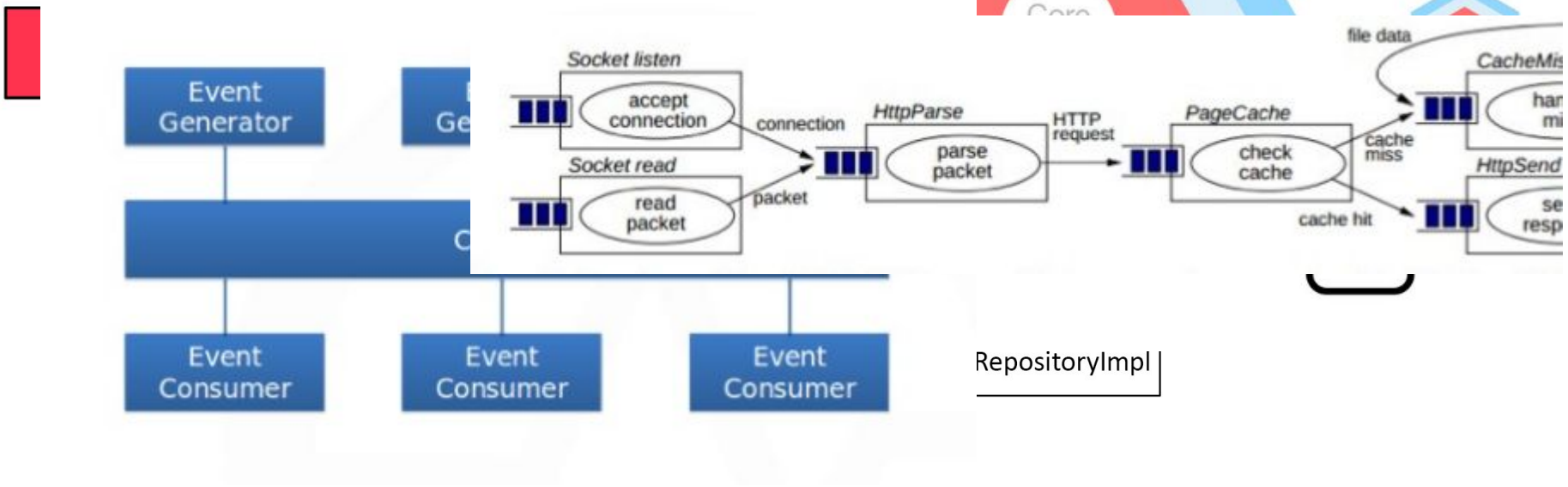




Hay muchas “arquitecturas”

Tipos de arquitectura: SEDA (Staged EDA)

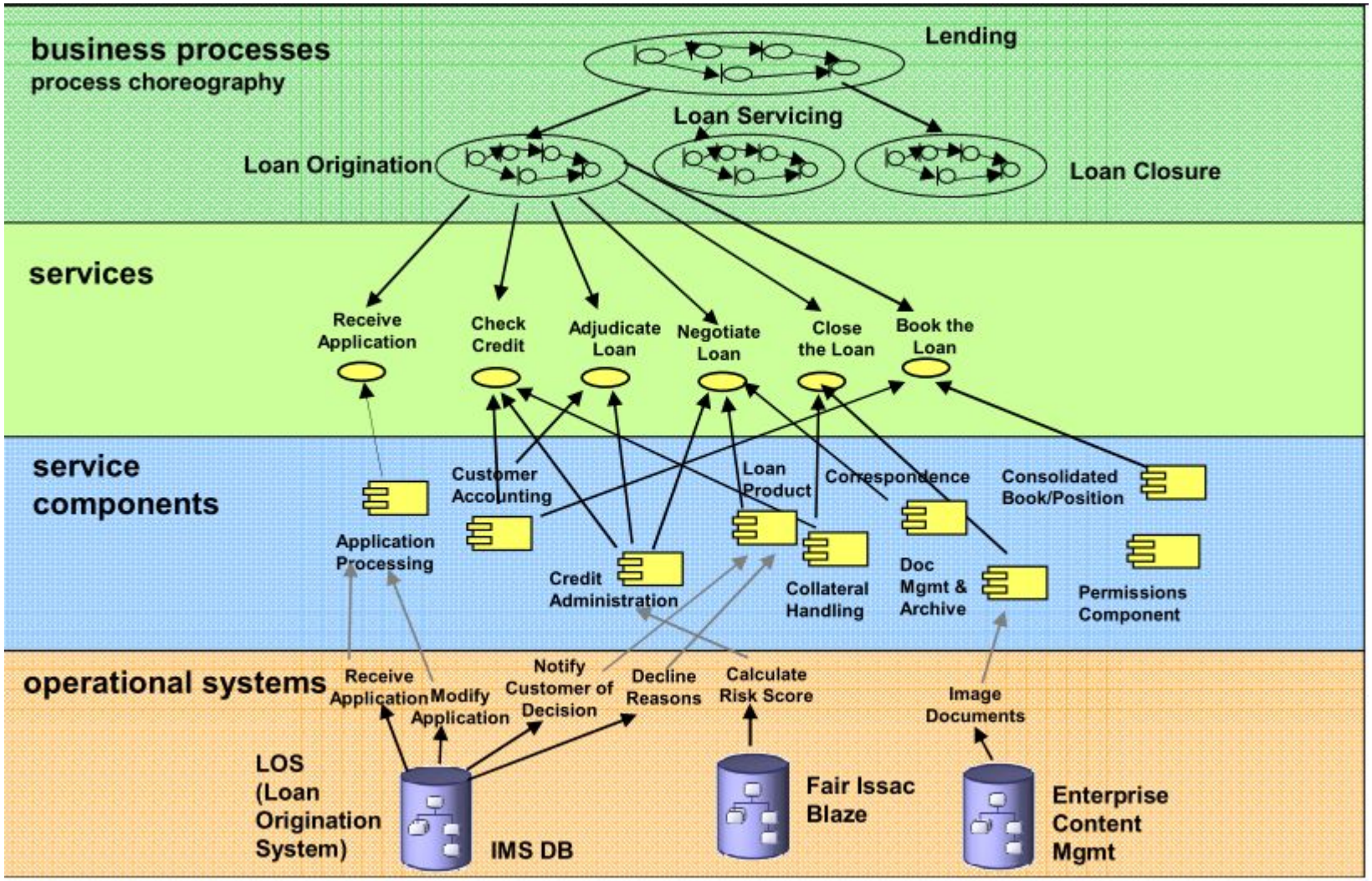




Hay muchas “arquitecturas”

@vfdiaz

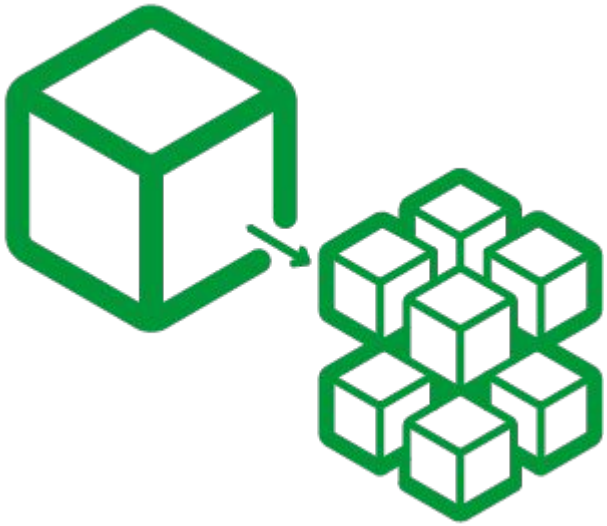
Tipos de arquitectura: SOA



Vamos acercándonos...

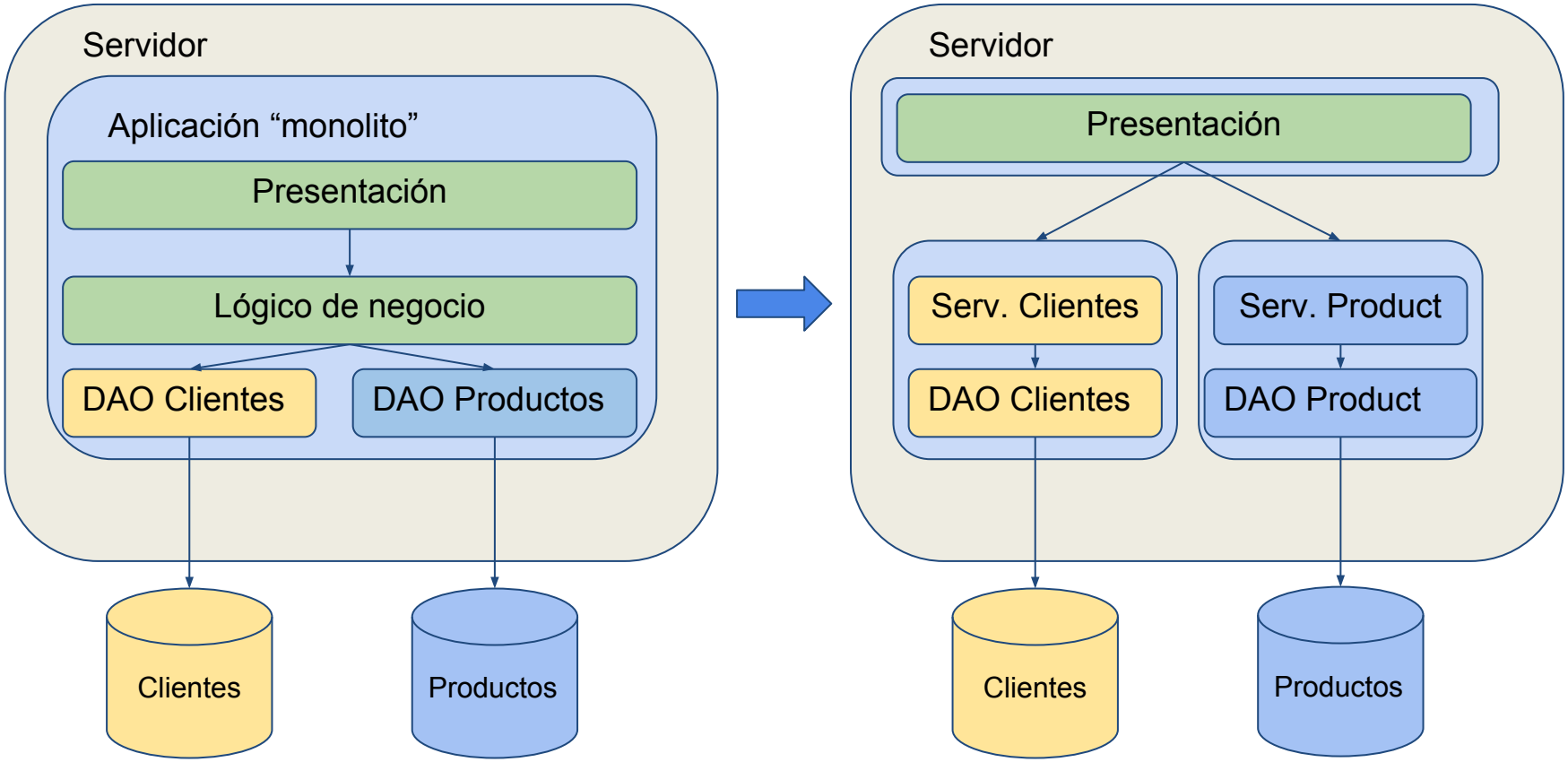
@vfdiaz

Conceptos sobre Microservicios



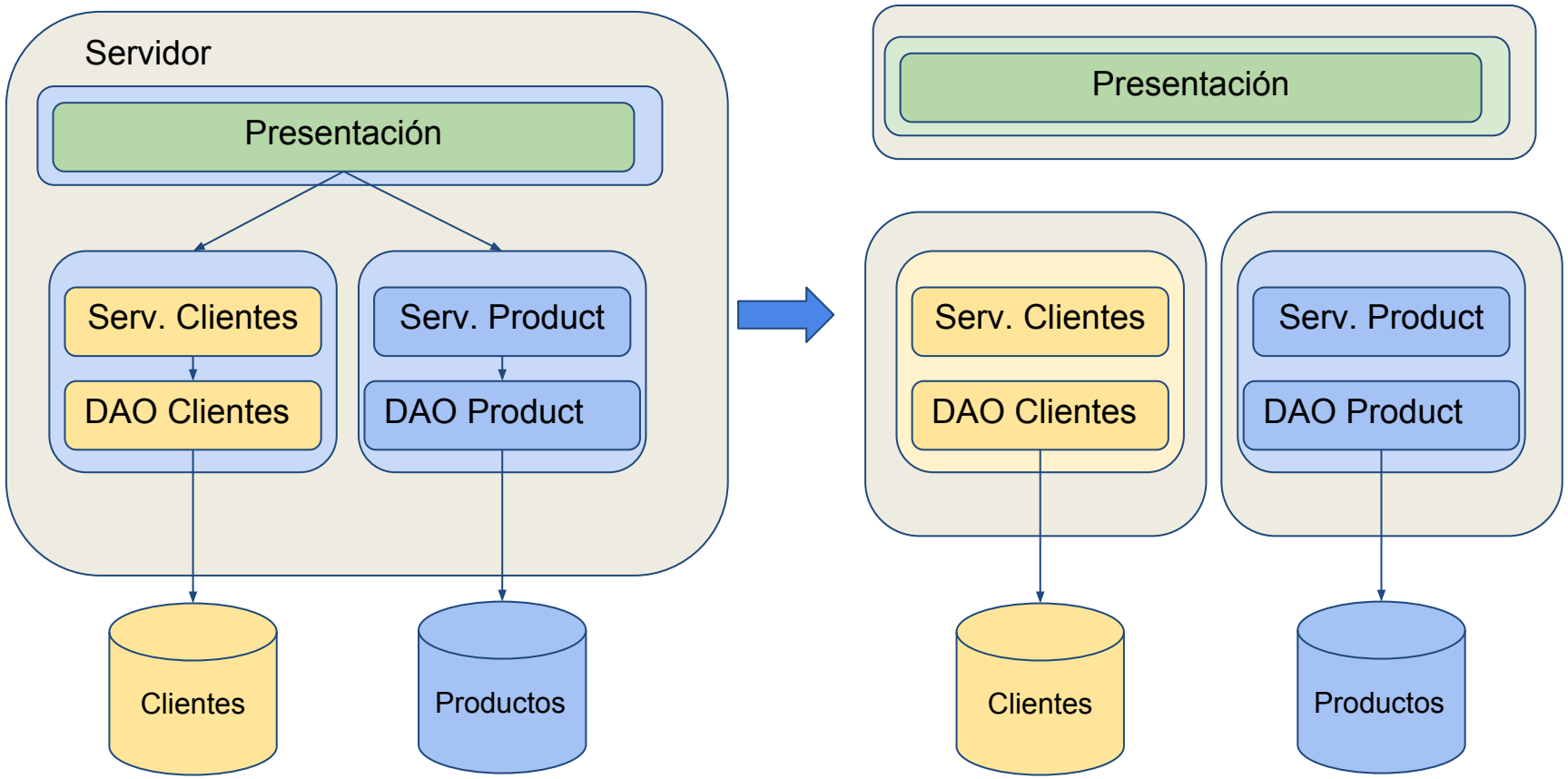
Arquitectura de Microservicios

- Aplicación como conjunto de pequeños servicios
- Cada servicio implementa una parte de funcionalidad



Arquitectura de Microservicios

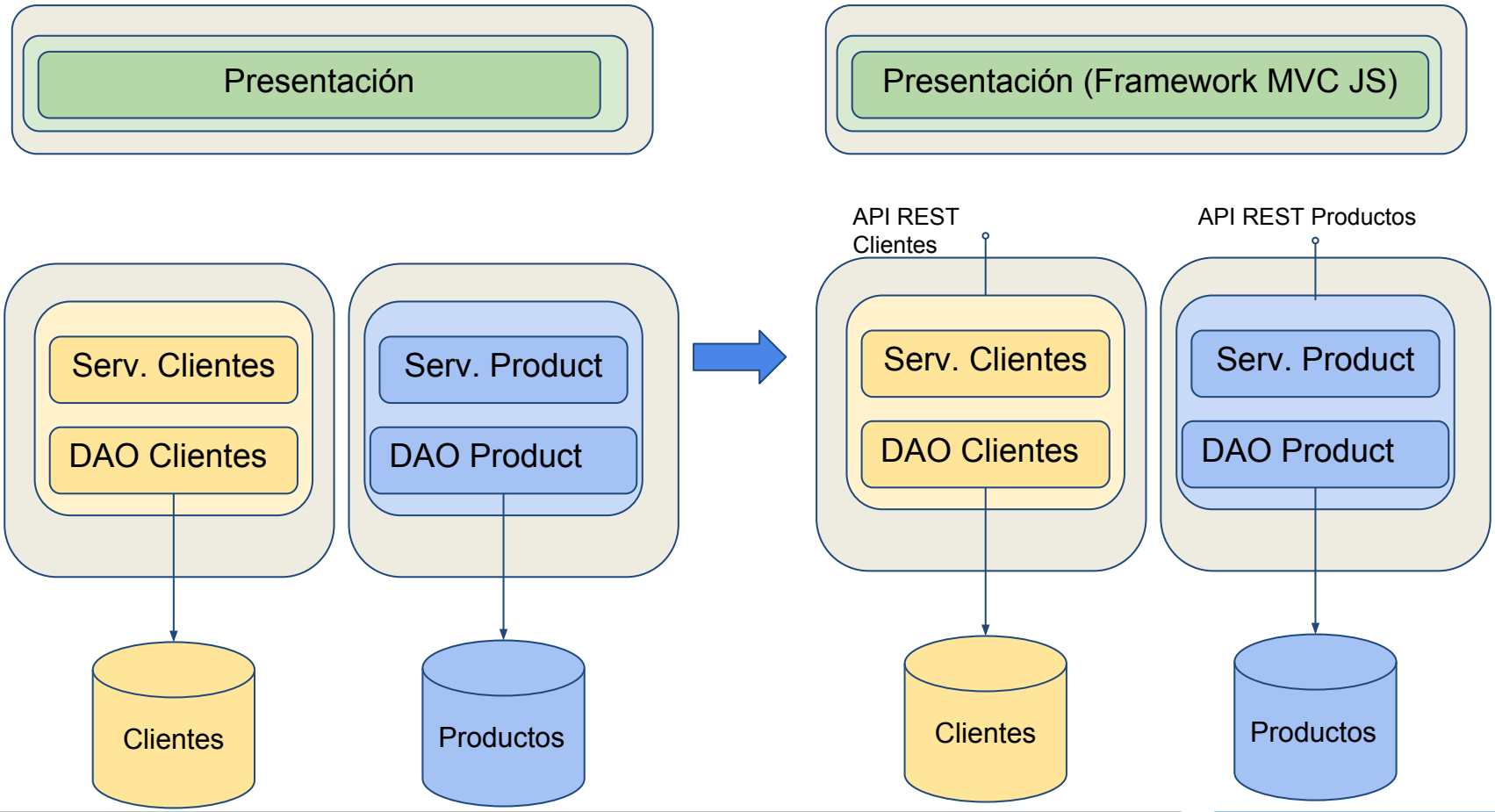
- Cada servicio se ejecuta en su propio proceso
- Despliegue independiente



Una aplicación, un servidor/proceso

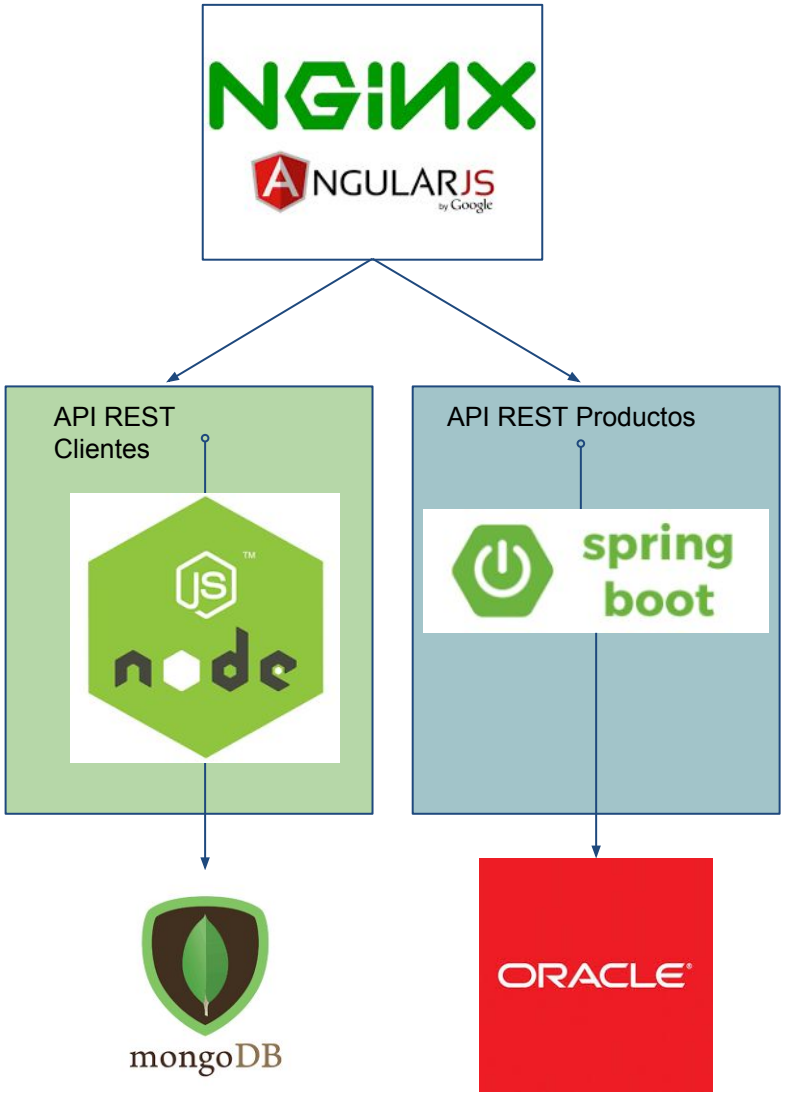
Arquitectura de Microservicios

- **Comunicación ligera (HTTP, Queues)**



Exponer servicios como APIs REST

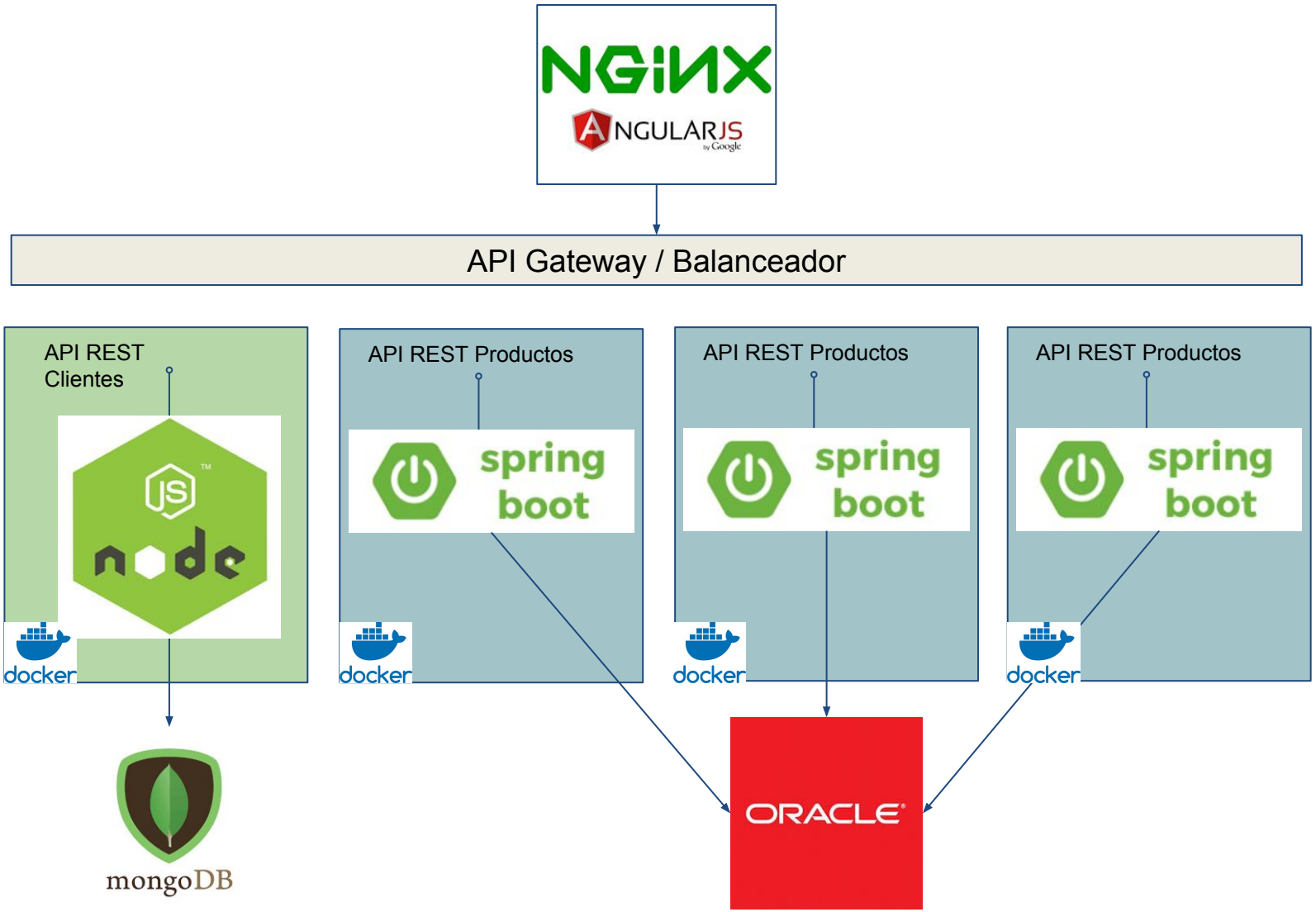
Arquitectura de Microservicios



Cada microservicio, un equipo, tecnologías heterogéneas

@vfdiaz

Arquitectura de Microservicios



Escalabilidad

@vfdiaz

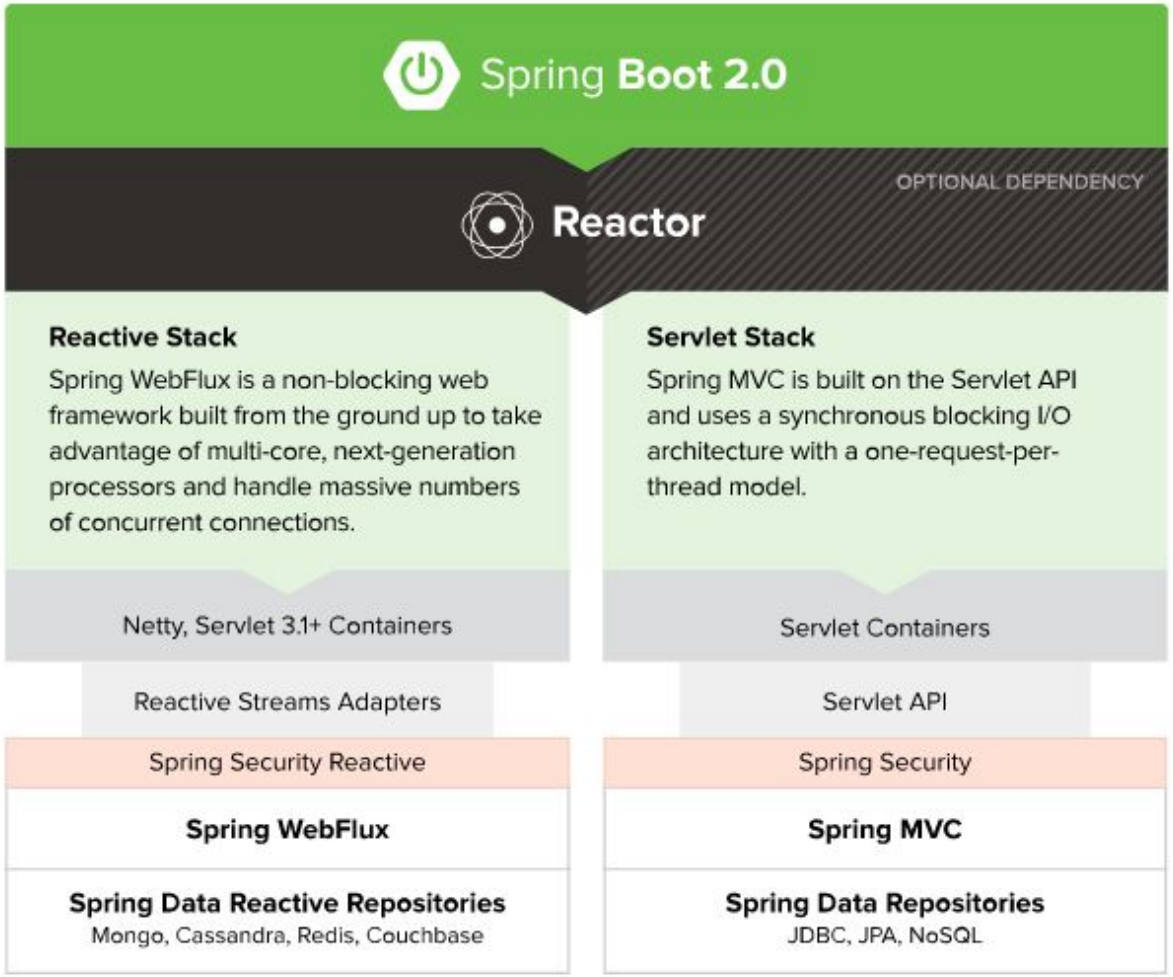
Introducción a Spring Boot

Introducción a Spring Boot

- Create stand-alone Spring applications
- Embed Tomcat, Jetty or Undertow directly (no need to deploy WAR files)
- Provide opinionated 'starter' POMs to simplify your Maven configuration
- Automatically configure Spring whenever possible
- Provide production-ready features such as metrics, health checks and externalized configuration
- Absolutely no code generation and no requirement for XML configuration

https://projects.spring.io/spring-boot/

Introducción a Spring Boot

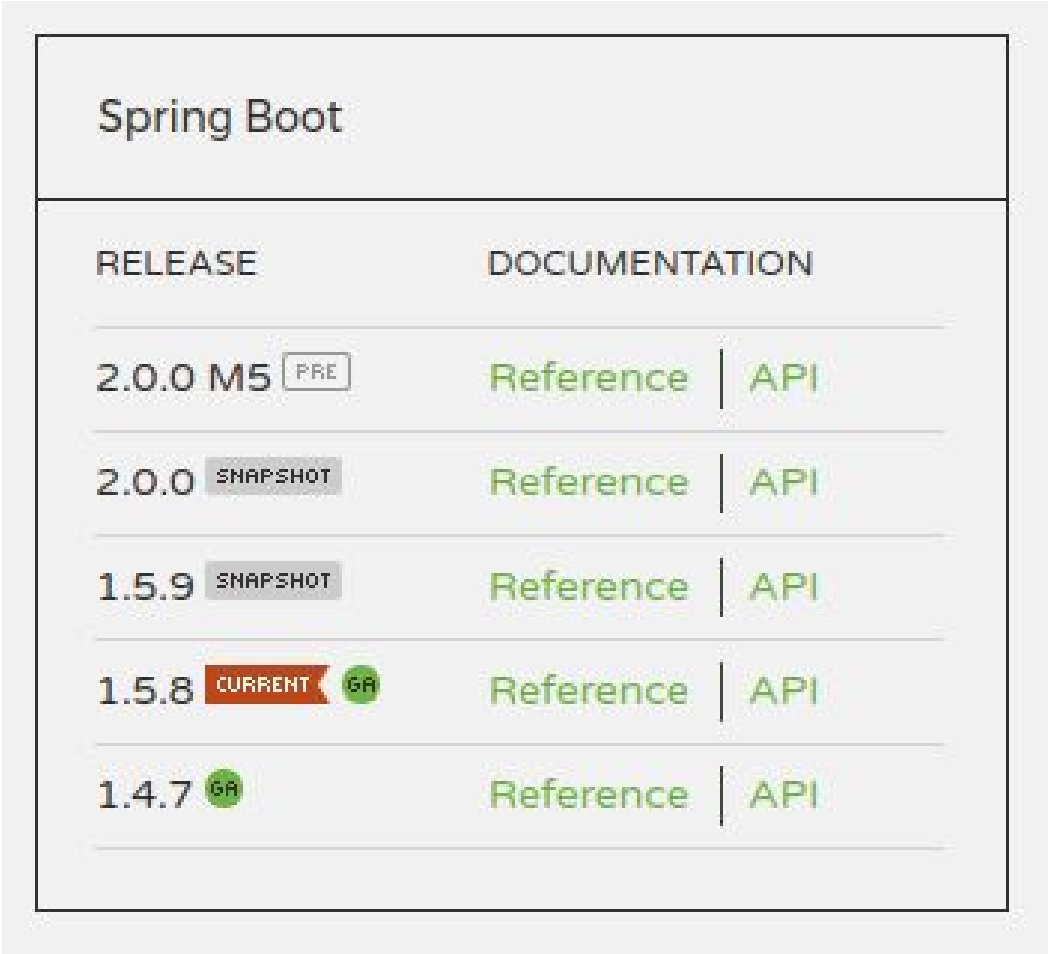


https://spring.io/

Spring Boot como base del nuevo Spring Framework

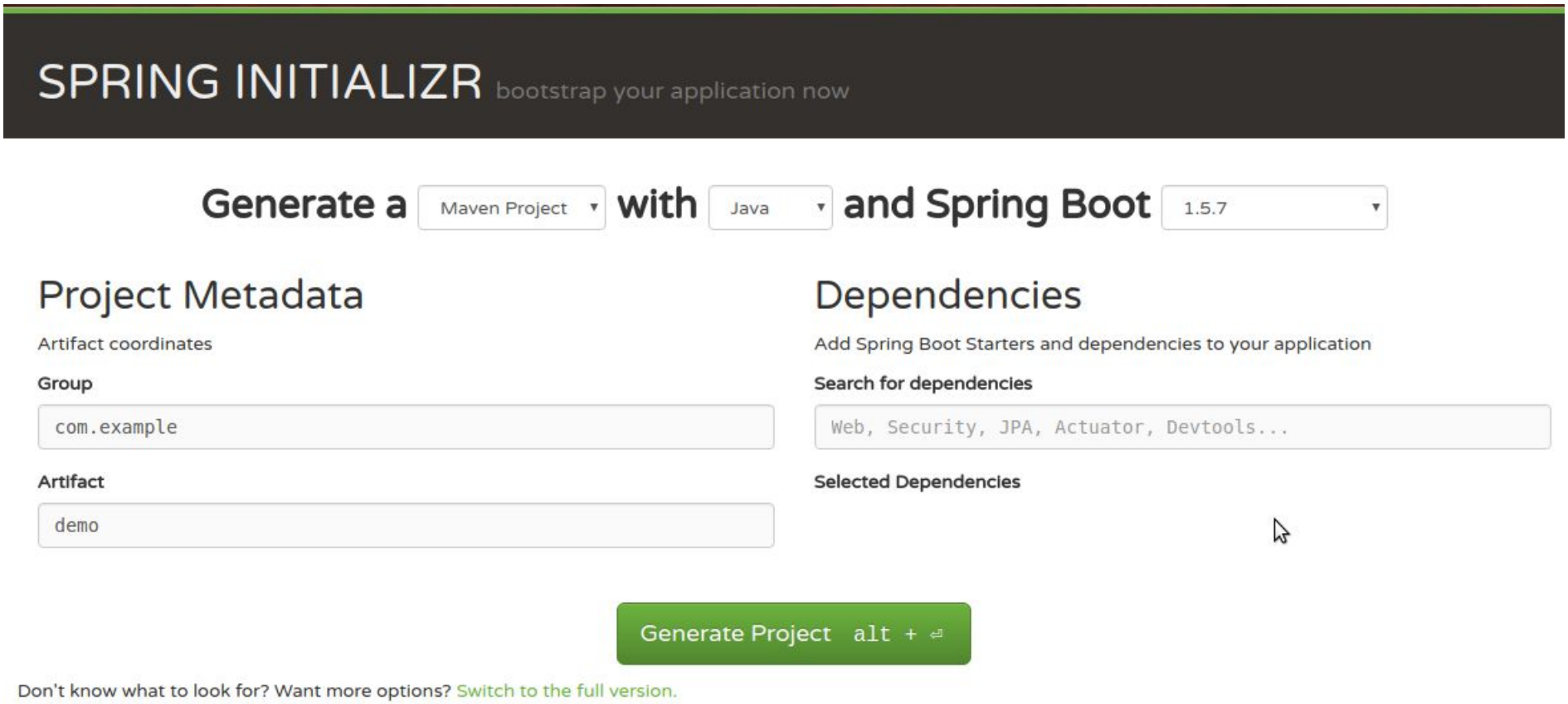
@vfdiaz

Introducción a Spring Boot



https://projects.spring.io/spring-boot/

Spring Boot Initializr



https://projects.spring.io/spring-boot/

Spring Boot Starters

- Utilidad para montar proyectos rápidamente
- Conjuntos de dependencias

spring-boot-starter-*

- compile('org.springframework.boot:spring-boot-starter')
- compile('org.springframework.boot:spring-boot-starter-**data-jpa**')
- compile('org.springframework.boot:spring-boot-starter-**data-rest**')
- compile('org.springframework.boot:spring-boot-starter-**web**')
- compile('org.springframework.boot:spring-boot-starter-**actuator**')

https://docs.spring.io/spring-boot/docs/current/reference/html/using-boot-build-systems.html#using-boot-starter

Hello World

Main.java

```java
package helloWorld;

public class Main {

    public static void main(String[] args) {
        System.out.println("Hello World!");
    }

}
```

JavaSE
vs
SpringBoot

SpringbootHelloworldRestApplication.java

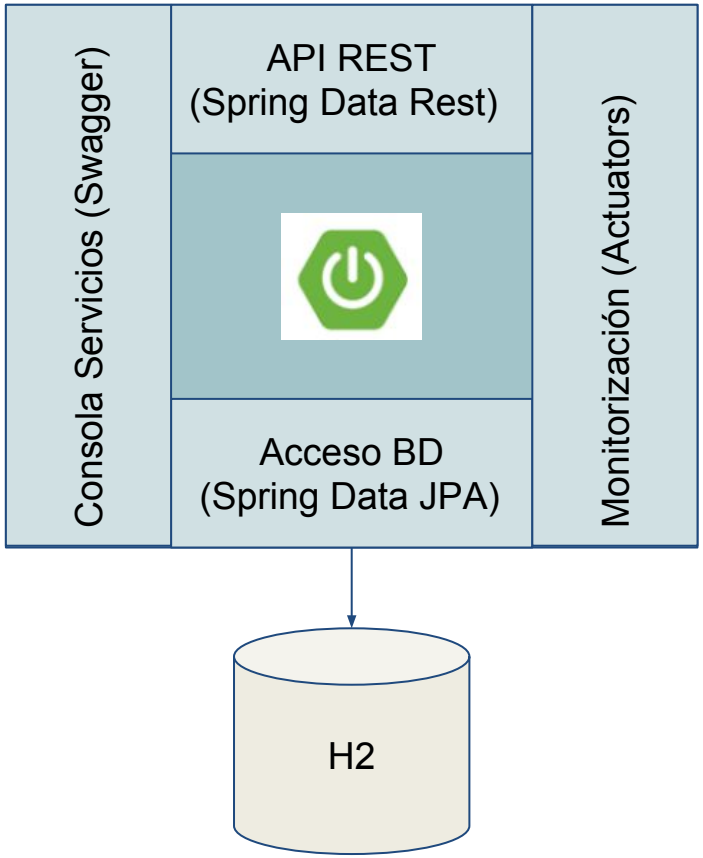
```java
package com.vfdiaz.springboot.springboothelloworldrest;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SpringbootHelloworldRestApplication {

    public static void main(String[] args) {
        SpringApplication.run(SpringbootHelloworldRestApplication.class, args);
    }
}
```

https://docs.spring.io/spring-boot/docs/current/reference/html/using-boot-build-systems.html#using-boot-starter

Desarrollo de Microservicios con Spring Boot

Práctica

JPA + APIs REST en 10 minutos

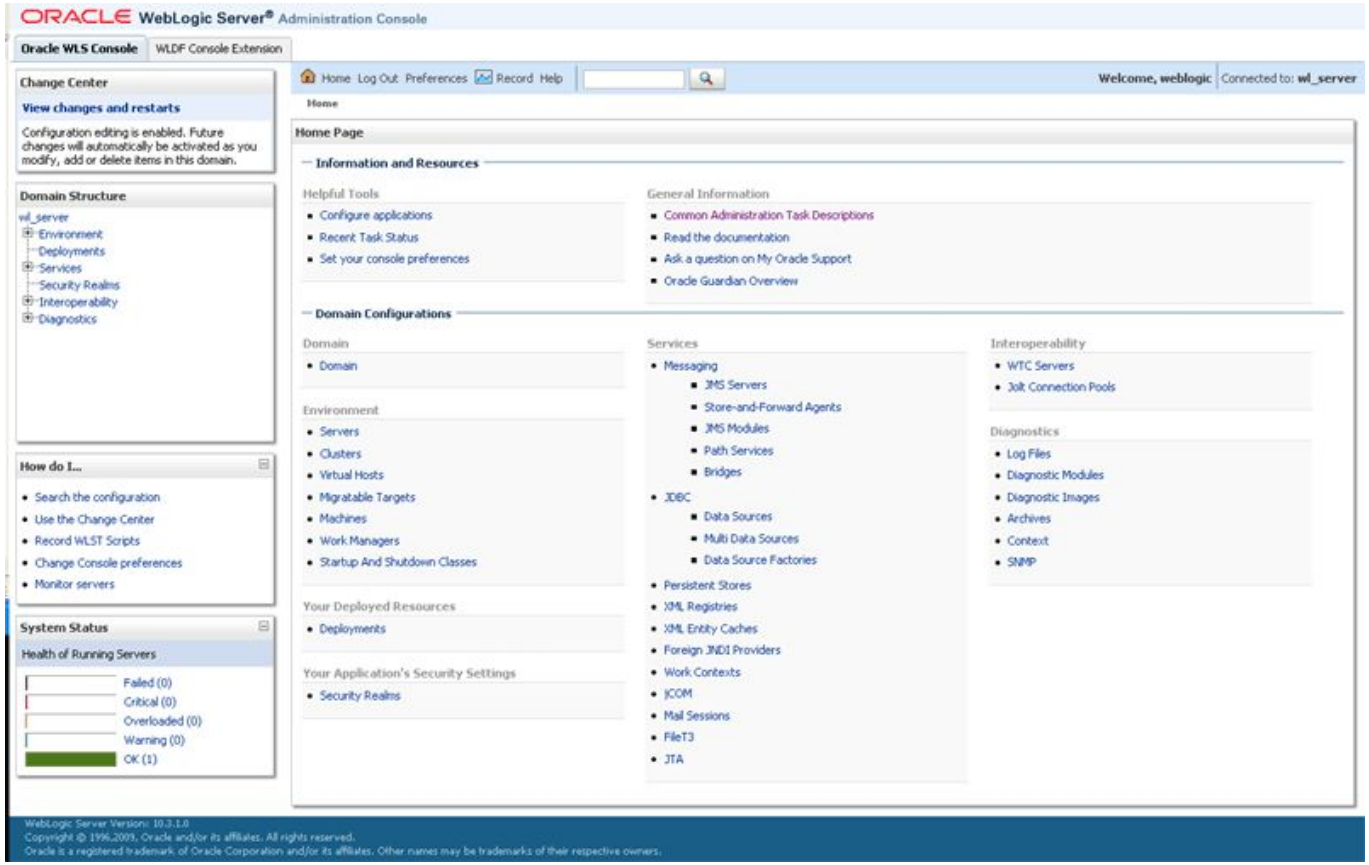


Probando..

@vfdiaz

¿Por qué Spring Boot?

¿Por qué Spring Boot?



vs

```
$ java -jar myapp.jar
```

No necesito mantener servidores

@vfdiaz

¿Por qué Spring Boot?

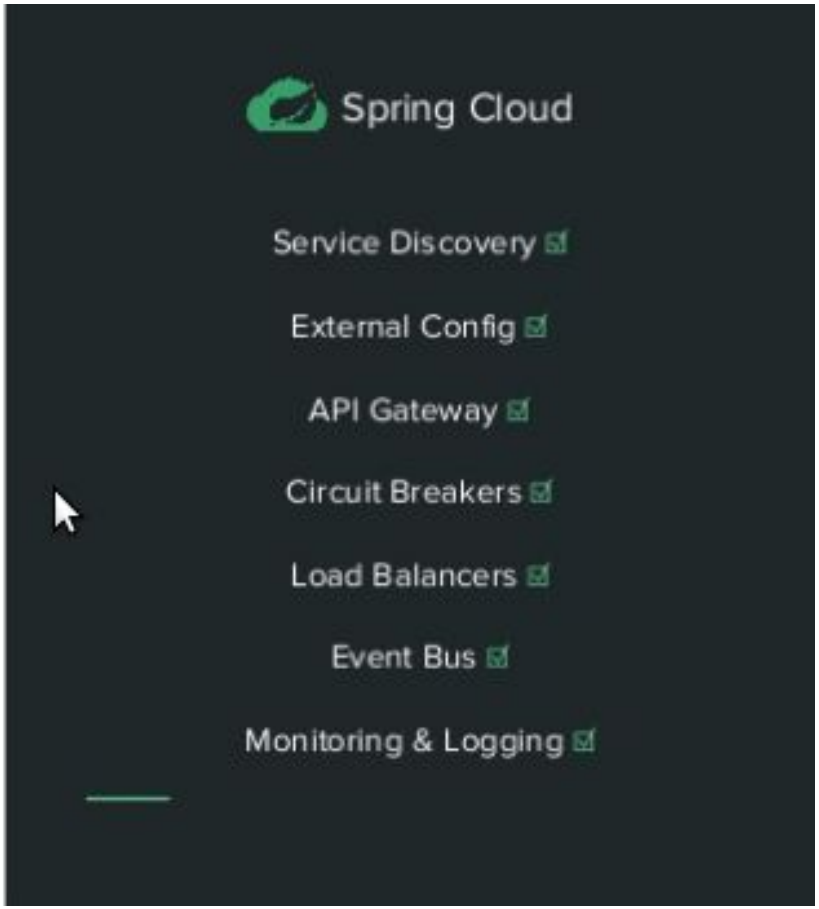
1. Choose a JEE Application Server with the right mix of features, standards support, etc. (weeks*)
2. Install the JEE application server on the target machine
3. Add the vendors JEE JARs to your Maven Repository
4. Create a new WAR project (mvn archetype)
5. Add the Jar dependencies you need to your POM (i. e. the vendors JEE jar)
6. Open the project in your IDE
7. Create a new Controller class
8. Add @Path to your class
9. Add @GET to your method
10. Add your servlet-mapping url-pattern instructions to your servlet.xml file
11. Build and package your WAR file
12. Take your WAR file to the application server
13. Install it in the correct folder on the app server
14. Reboot / restart the app server
15. Check the log / console to see if your particular service has started successfully
16. Call localhost to check your message is returned

Congratulations, your service is up!
Elapsed Time 2-4 hours*

1. Go to **start.spring.io** and download a starter project
2. Open the project in your IDE
3. Create a new Controller class
4. Add @RestController to your class
5. Add @RequestMappping to your method
6. Build your JAR `(mvn package)`
7. Run `java -jar <filename>`
8. Check your message is returned

Congratulations, your service is up!
Elapsed Time **15** mins

JPA + APIs REST en 10 minutos



https://www.slideshare.net/BenWilcock1/microservices-java-ee-vs-spring-boot-and-spring-cloud

Desarrollo de Microservicios con Spring Boot