DigitalOnUs
Accelerating Digital Business

FIRST MICROSERVICE USING SPRING BOOT – JAVA DEV DAY 2019

ALEJANDRO CARDENAS

DigitalOnUs
Accelerating Digital Business

OBJETIVO

- Entender a nivel básico una arquitectura basada en microservicios
- Poder crear un microservicio utilizando Consul como discovery service y KV store

DigitalOnUs
Accelerating Digital Business

AGENDA

- Aplicación Monolitica
- Microservicios
- Consul
- Ejemplos
- Q&A

DigitalOnUs
Accelerating Digital Business

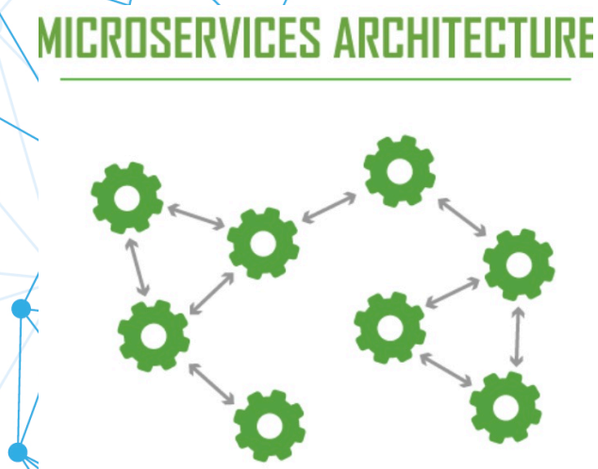
APLICACIÓN MONOLITICA

- Solo un código base
- Solo un Build System
- Solo un artefacto para hacer deploy
- Puede ser código muy grande miles de clases y paquetes.
- Escalar la aplicación es todo o nada

DigitalOnUs
Accelerating Digital Business

QUE SON LOS MICROSERVICIOS?

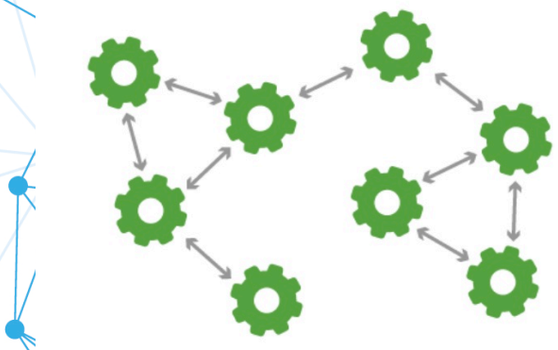
- Pequeños servicios dirigidos
- Cada servicio tiene su propio repositorio
- Cada servicio esta separado, permitiendo hacer deploy de cada uno sin depender o modificar otro.
- Típicamente cada servicio tiene su propia base de datos



DigitalOnUs
Accelerating Digital Business

MICROSERVICIOS – BENEFICIOS

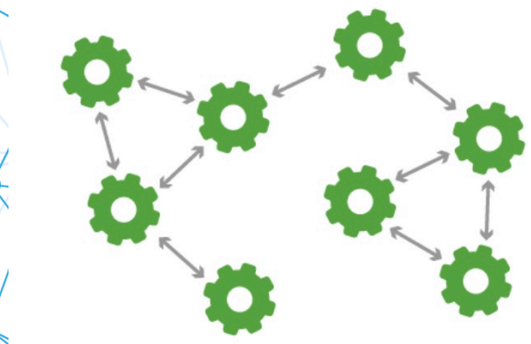
- Fáciles de entender y desarrollar
- Calidad de Software
- Escalabilidad
- Agnostico al stack de tecnologia



DigitalOnUs
Accelerating Digital Business

MICROSERVICIOS – DIFICULTADES

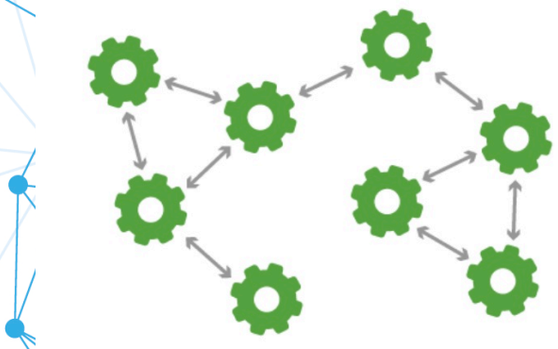
- Las pruebas de integración se complican
- Deploys de una app completa se vuelve mas complejo
- Costos de Infraestructura



DigitalOnUs
Accelerating Digital Business

MICROSERVICIOS – COMO DEFINIRLOS

- Debe tener solo un endpoint?
- Debe tener solo CRUD de un dominio?
- Debe tener muchos endpoints referentes a la unidad de negocio?
- Jeff Bezos - Two Pizza Rule



DigitalOnUs
Accelerating Digital Business

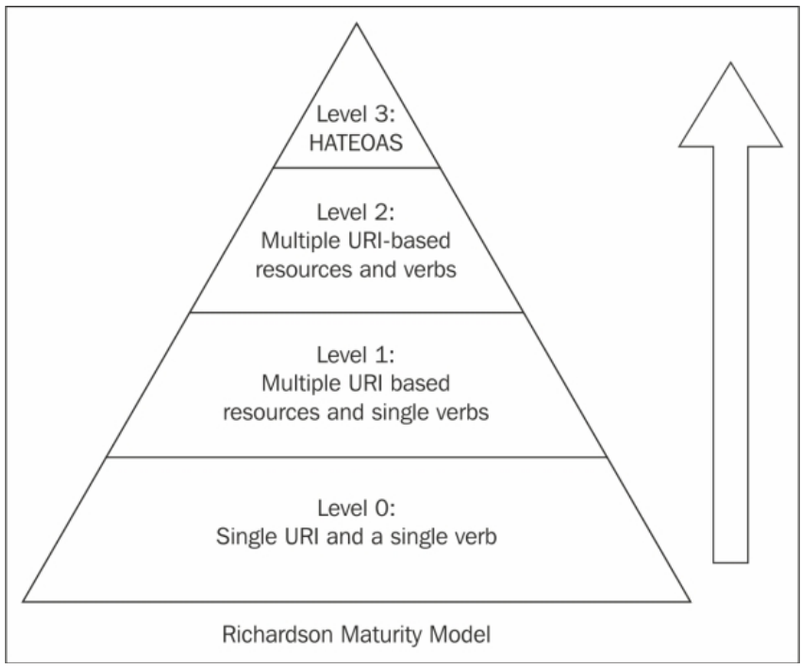
SINGLE RESPONSIBILITY PRINCIPLE

- Aplica a Microservicios ?

DigitalOnUs
Accelerating Digital Business

BREVE REPASO REST

- HTTP Request Methods
- HTTP Status Code
- Richardson Maturity Model



Richardson Maturity Model

DigitalOnUs
Accelerating Digital Business

CONSUL

Consul es una herramienta de Hashicorp, cuya función principal es facilitar la comunicación entre servicios productores y consumidores, llevando a cabo el registro y descubrimiento de los mismos. Sus características más importantes son:

- Descubrimiento de servicios.
- Health checking a nivel de servicio o de nodo del cluster consul.
- Almacenamiento clave-valor, lo que ofrece la posibilidad de configuración centralizada.
- Multidatacenter: No hay que preocuparse de incluir capas para unir diferentes regiones.

El concepto fundamental de Consul es Agente. Un agente es un proceso de larga duración que se ejecuta en cada nodo que forma parte del sistema Consul. Es el responsable de surtir de información de sí mismo y recoger información de los demás agentes. Se comunica vía HTTP y, además, integra un servidor DNS. Puede ejecutarse en dos modos: cliente o servidor

HashiCorp
Consul

DigitalOnUs
Accelerating Digital Business

PORQUE CONSUL EN LUGAR EUREKA

El cluster de Consul tiene un servidor que actúa como líder y responde las peticiones.

En caso de caída se elige uno nuevo y este cambio es transparente para las aplicaciones cliente

Eureka, en cambio, si un servidor se cae, son las aplicaciones cliente las que activamente comunican con el siguiente servidor Eureka configurado en ellas.

Además, ofrece interfaz DNS para comunicarse, algo que Eureka no por lo que no estamos atados a comunicación REST

Contamos con almacenamiento KV.

HashiCorp
Consul

NETFLIX
OSS
Eureka

DigitalOnUs
Accelerating Digital Business

DIFERENCIAS ENTRE ZUUL SPRING CLOUD GATEWAY

- Zuul se basa en servlet 2.5 (funciona con 3.x), usando API de bloqueo y no admite conexiones de larga duración, como websockets.

- Spring Cloud Gateway esta basado en Spring Framework 5, Project Reactor (WebFlux) y Spring Boot 2 utilizando API sin bloqueo. Esto hace que sea compatible con conexiones de larga duración (Websockets) y tiene una mejor integración con Spring.

- Spring Cloud Gateway tiene un rendimiento superior en comparación con el de Zuul. Gracias a eventloop de Netty.

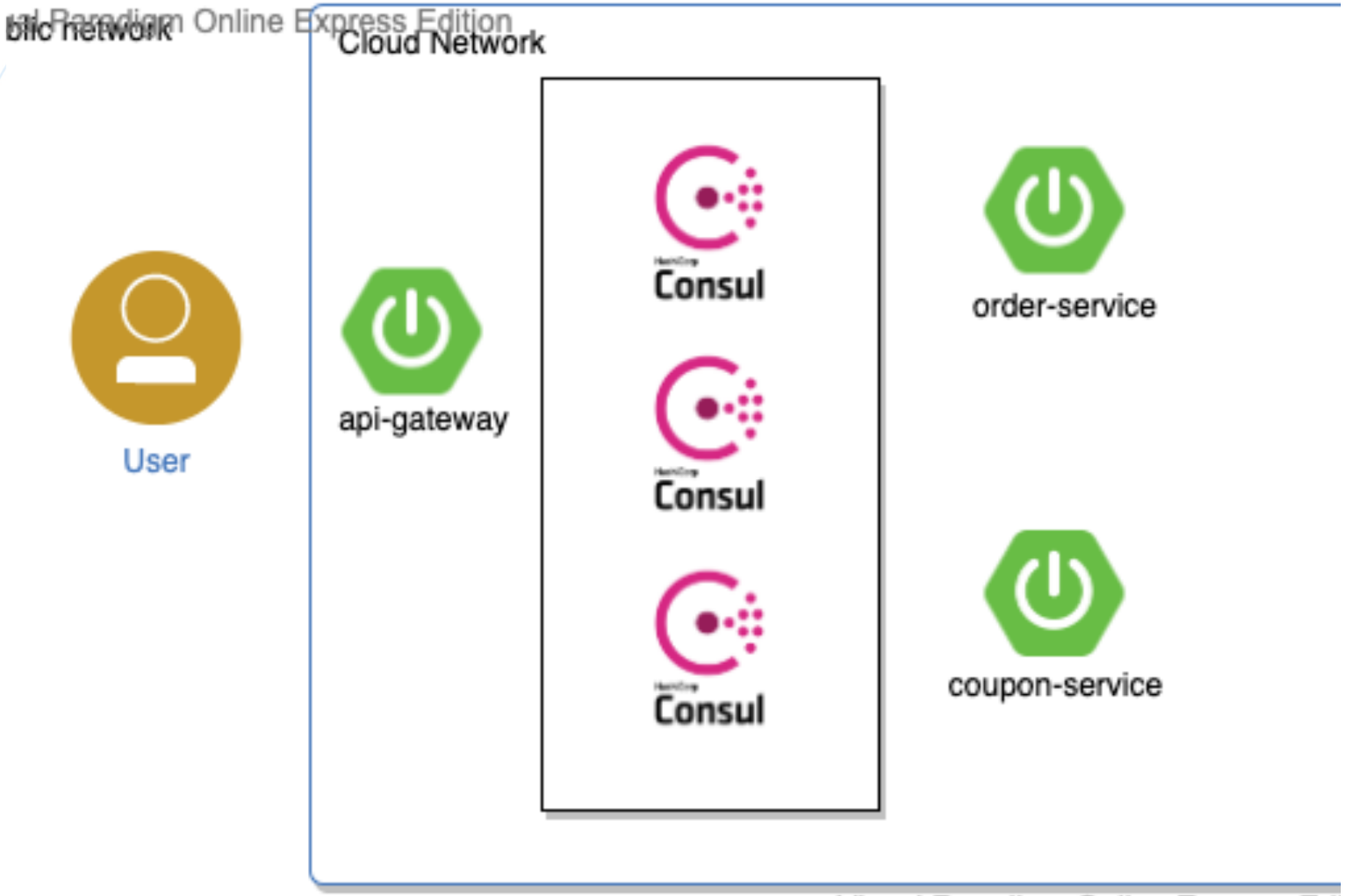
DigitalOnUs
Accelerating Digital Business

SPRING INITIALIZR

- https://start.spring.io/

DigitalOnUs
Accelerating Digital Business

EJEMPLO



DigitalOnUs
Accelerating Digital Business

EJEMPLO – PASOS

- Levantamos Consul (Utilizamos Docker)
- Creamos el servicio “order-service”
- Creamos el servicio “coupon-service”
- Hacemos que podamos registrar varias instancias del mismo servicio
- Activamos Load Balancing para hablar de order a coupon
- Agregamos KV en consul y los leemos en el servicio y hacemos que se refresquen (‘/config/application/<nombre-propiedad>’)
- Creamos api-gateway.

DigitalOnUs
Accelerating Digital Business

PUNTOS INTERESANTES

- Cuando usamos DiscoveryClient para traer los servicios disponibles , por default nos trae los que están un healthy, para hacer que solo regrese los healthy, se requiere el property: **spring.cloud.consul.discovery.query-passing=true**

- Al igual que Cloud Config mantiene una jerarquía de priorización de configuraciones. En caso de tener repetida una clave en varios paths, será la de mayor prioridad el valor asociado al nombre de la aplicación y que disponga de un perfil igual al indicado al arranque:

config/<**spring.application.name**>,<**spring.profile.active**>/key

config/<**spring.application.name**>/key

config/application,<**spring.profile.active**>/key

config/application/key

DigitalOnUs
Accelerating Digital Business

PUNTOS INTERESANTES

- Podemos cambiar la dirección del consul que queremos usar:

```
spring.cloud.consul.discovery.tags=javaDevDay
spring.cloud.consul.discovery.prefer-ip-address=true
spring.cloud.consul.host=localhost
spring.cloud.consul.port=8500
```

- Para registrar un servicio muchas veces, ahi que crear instancias con diferente nombre:

```
spring.cloud.consul.discovery.instance-id=$
{spring.application.name}:${random.value}
```

DigitalOnUs
Accelerating Digital Business

Q&A

DigitalOnUs
Accelerating Digital Business

THANK YOU

CONTACT INFORMATION

- eMails : alejandro.cardenas@digitalonus.com

  cardenas.alejandro@gmail.com

- Gitlab: https://gitlab.com/cardenas.alejandro

- GitHub: https://github.com/acardenasnet

- Linkedin: https://www.linkedin.com/in/acardenasnet/