



Serverless con OpenFaaS & Java

Ing. Carlos Camacho
PUCMM, Santiago, República Dominicana,
29/06/2019
JCONF Dominicana 2019

Acerca de mí (un poco personal)

- Profesor por vocación.
- Pertenezco al lado oscuro de la fuerza (Dark Side).
- Aldea de la Hoja (Konoha).
- Evangelista de Java.
- Apasionado de Linux.
- Ideología conservadora.
- Nacido en la era Nintendo. Amante de los RPG (Chrono Trigger) y juegos cooperativos (The Division, Monster Hunter, entre otros).
- Rock, Merengue, Bachata, Balada, Típico, Salsa, Son, Dembow, Reggaeton.
(Pirata musical)

Acerca de mí (un poco más serio)

- Ingeniero Telemático.
- Magíster en Tecnología Educativa.
- Profesor de la Escuela de Ingeniería en Computación y Telecomunicaciones, PUCMM.
- Director Académico de la Escuela de Ingeniería en Computación y Telecomunicaciones, Santiago PUCMM.
- Más de 10 años de experiencia desarrollo de sistemas en tecnología Java (JSE y JEE).
- 8 años de implementación en sistemas basados en Groovy y Grails.
- Fundación Código Libre Santiago - Miembro Fundador.
- Gerente AvatharTech EIRL.
- Co-Fundador del Java User Group Java Dominicano

Asuntos Legales

Los conceptos y juicios de valor emitidos en esta presentación es responsabilidad personal y no se puede entenderse como una posición oficial de alguna empresa con la que hemos tenido relación laboral.

Todas las marcas registradas, así como todos los logotipos, imágenes, fotografías, audio y vídeos mostrados en esta presentación son propiedad de sus respectivos propietarios.

Su utilización es solamente para fines ilustrativos y no pretendemos dar a entender cualquier afiliación con esas empresas.

Agenda

- Evolución a la Nube.
- ¿Qué son los Serverless?.
 - Plataformas Disponibles en la Nube
 - Ventajas y Desventajas
- OpenFaaS
 - Diseño
 - Hola Mundo OpenFaaS
 - Creado Función Java
- Demostración SPA con OpenFaaS
- Preguntas

Evolución a la Nube

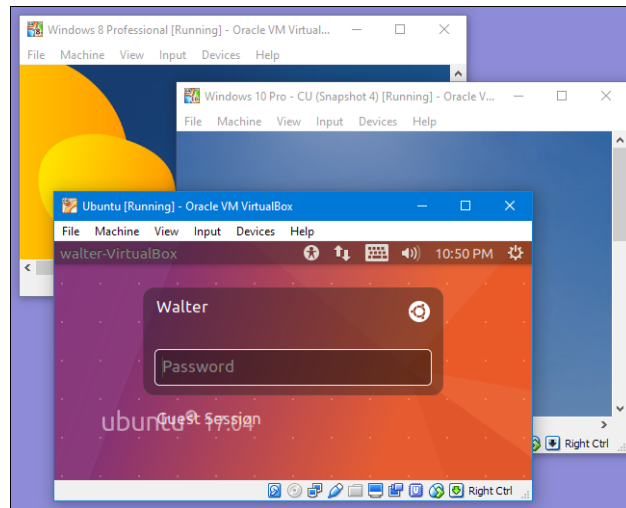
Servidor Físico

- Pensados para soportar aplicaciones en producción (cualquier ambiente).
- Necesitan un cuidado especial y mantenimientos.
- Las configuraciones y sus implementaciones son lentas, dependen de otros factores (espacios físicos, capacidad energética, capacidad en climatización, entre otros).
- **Pensados para durar años.**



Máquinas Virtuales

- Permiten entornos compartidos entre los recursos (Multitenancy).
- Permiten aislamiento entre las máquinas virtuales.
- Su implementación se puede desarrollar en minutos.
- **El tiempo de vida ronda las semanas o meses.**

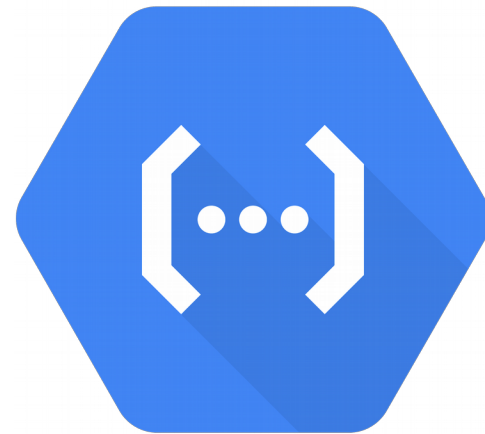
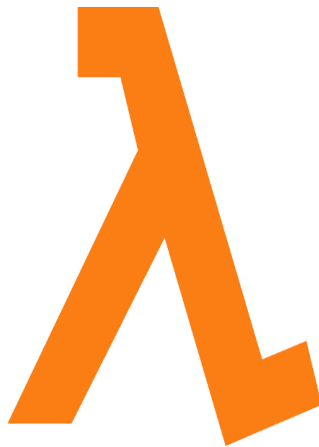


Contenedores

- Entornos de alta disponibilidad.
- Aislamiento de procesos.
- Evitar el consumo de recursos en memoria y espacio generados por las máquinas virtuales.
- Pensados para una rápida implementación de recursos.
- **El tiempo de vida ronda las horas o días.**



Lo nuevo en el ambiente Serverless



Serverless

“Es un diseño de aplicaciones donde dependen de manera significativa de servicios de terceros conocidos como **Backend as a Services (BaaS)** o ejecutan un bloque de instrucciones en un contenedor efímero, conocidos como **Function as a Services (Faas)**”, Mike Roberts, traducción libre.

Arquitectura Serverless

- **Backend as Services (BaaS):** Nace como un concepto para integrar las aplicaciones web con las aplicaciones móviles, utilizando API para esos fines.
- **Function as Services (FaaS):** Están más relacionado a no administrar los aspectos de infraestructura. (Estaremos abordando en la presentación)



Serverless

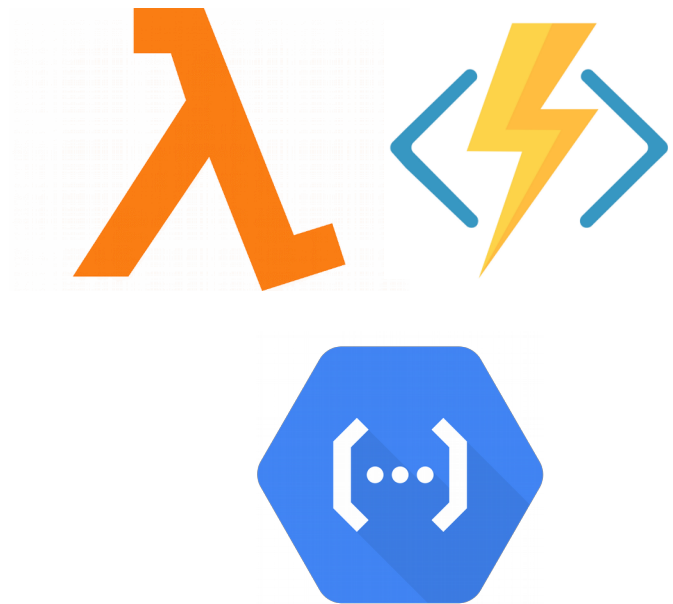
Una función es como la unidad lógica más pequeña de una aplicación

¿Cómo funcionan?

- Cada invocación de la función crea una nueva instancia.
- Esa invocación es realizada por eventos que están conectado con la función para ese fin.
- Pensadas para aplicaciones que no requieran estados.

¿Entonces los Serverless?....

- Son pequeñas unidades de cómputo.
- Pensados para escalar.
- Aislamiento entre procesos.
- Rápida implementación.
- Permite trabajar en múltiples lenguajes.
- El tiempo de vida ronda los segundos.



Proveedores en La Nube

- AWS Lambda.
- Google Cloud Functions.
- Microsoft Functions.
- IBM Openwhisk.
- IronFunctions



¿Cuáles son los Beneficios?

- Pagamos por lo que usamos, **el tiempo muerto no importa.**
- Diseñados para la escalar de forma fácil.
- No tenemos que configurar servidores.
- Implementación fácil.
- Seguridad.

¿Cuáles son las desventajas?

- Dependencia de las librerías de la plataforma utilizada.
- La depuración y el monitoreo dependen de las herramientas disponibles por el proveedor.
- Limitaciones establecidas por políticas de rendimiento por los proveedores.
- Limitaciones de plataforma de programación.
- Podemos tener problemas de latencia.
- Prueba locales son más complejas.
- Pérdida de control de la aplicación.
- Según la demanda pueden dejar de ser costo-efectivas.

¿Tengo limitaciones para usar
proveedores en La Nube?



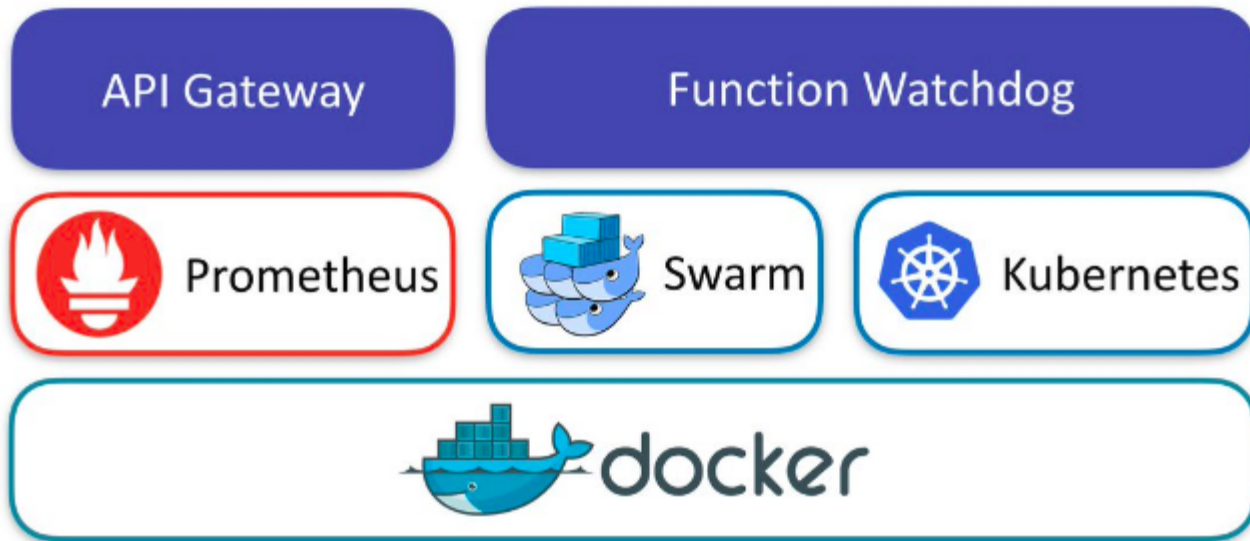
OpenFaaS

OpenFaaS

- Es un framework basado en imágenes Docker.
- Puede implementar cualquier tipo plataforma de programación siempre y cuando podamos “Dockerizar”.
- Puede ser utilizado en nubes privadas y públicas utilizando Kubernetes o Docker Swarm permitiendo la portabilidad.
- Fácil de utilizar gracias al portal de gestión.
- Dispone de una CLI para definir las funciones mediante el uso de plantillas.
- Auto escalable mediante se incrementa la demanda.
- Más de 14k estrellas en Github y más de 100 contribuyentes

Diseño Conceptual de OpenFaaS

Functions as a Service



Diseño Conceptual de OpenFaaS

- **Función Watchdog:** Es un utilitario que es incluido en cualquier imagen Docker como un punto de entrada que maneja el protocolo HTTP desde la UI.
- **API Gateway:** Es la puerta externa a las funciones recopilando estadísticas mediante Prometheus (software de monitoreo). Se encarga de escalar mediante Docker Swarm o Kubernetes.
- **Orquestadores de Contenedores:** Puede ser implementado bajo Docker Swarm o Kubernetes permitiendo la escalabilidad.
- **Docker:** Contenedor de aplicaciones base.

Hola Mundo en OpenFaaS

- Vamos a utilizar Play With Docker:

<https://labs.play-with-docker.com/>

- Inicializamos Docker Swarm:

```
docker swarm init --advertise-addr eth0
```

- Clonamos e inicializamos:



```
git clone https://github.com/openfaas/faas && cd faas && ./deploy_stack.sh
```

- Guardar el token generado para el usuario admin por la consola.


Ventana Play With Docker

03:52:08

CLOSE SESSION

Instances  

+ ADD NEW INSTANCE




192.168.0.33
node1

bkbhj2lt_bkbhj45tcgkg00amrtr0

IP 192.168.0.33 8080 9090

Memory 13.25% (530.1MiB / 3.906GiB) CPU 39.64%

SSH ssh ip172-18-0-27-bkbhj2ltcgkg00amrtqg@direct.labs.play-with-do 

DELETE EDITOR

```

Resolving deltas: 100% (8143/8143), done.
Attempting to create credentials for gateway..
aty4lrnnqac1t0v4ewy57khv
mrdwog00pbexapk998jckfk4y
[Credentials]\n username: admin \n password: ba2b3b96ca372f407e116c73b936e88aea064d6882\n echo -n ba2b3b96ca372f407e116c73b936e88aea06454c4b222b0adeafc80f214d6882 |
name=admin --password-stdin

Enabling basic authentication for gateway..

Deploying OpenFaaS core services
Creating network func_functions
Creating config func_prometheus_rules
Creating config func_alertmanager_config
Creating config func_prometheus_config
Creating service func_alertmanager
Creating service func_gateway
Creating service func_auth-plugin

```



Agregando una Función



The screenshot shows a web browser window with the OpenFaaS Portal interface. The browser's address bar shows a URL from 'direct.labs.play-with-docker.com'. The page has a dark blue header with the 'OpenFaaS Portal' logo and a sidebar with a 'Deploy New Function' button. The main content area is light blue and contains the text: 'You have no functions in OpenFaaS. Start by deploying a new function.' Below this is a blue button labeled 'DEPLOY NEW FUNCTION'. Further down, it says 'Or use **faas-cli** to build and deploy functions:' followed by a terminal command in a light yellow box: `$ curl -sSL https://cli.openfaas.com | sudo sh`. A mouse cursor is visible over the main content area.

Uso del CLI de OpenFaaS

- Instalación del Cliente:

```
curl -sSL https://cli.openfaas.com | sudo -E sh
```

- Realizar login con el key generado:

```
faas-cli login -u admin -p KEY_GENERADO
```

- Carga de las plantillas oficiales:

```
faas-cli template pull
```

- Creación Función Java:

```
faas-cli new --lang java8 hola-mundo-java
```

Estructura del Proyecto

```
[node1] (local) root@192.168.0.33 ~/hola-mundo-java  
$ tree
```

```
.  
├── build.gradle  
├── gradle  
│   └── wrapper  
│       ├── gradle-wrapper.jar  
│       └── gradle-wrapper.properties  
├── settings.gradle  
└── src  
    ├── main  
    │   └── java  
    │       ├── com  
    │       │   └── openfaas  
    │       │       ├── function  
    │       │       └── Handler.java  
    └── test  
        └── java  
            └── HandlerTest.java
```

```
10 directories, 6 files
```

Archivo Handler.java

```
1 package com.openfaas.function;  
2  
3 import com.openfaas.model.IHandler;  
4 import com.openfaas.model.IResponse;  
5 import com.openfaas.model.IRequest;  
6 import com.openfaas.model.Response;  
7  
8 public class Handler implements com.openfaas.model.IHandler {  
9  
10     public IResponse Handle(IRequest req) {  
11         Response res = new Response();  
12         res.setBody("Hola Mundo en Java con OpenFaaS :-D");  
13  
14         return res;  
15     }  
16 }
```

```
java/com/openfaas/function/Handler.java" 16L, 407C 1,1
```

Publicar la Función en OpenFaaS

- Para Compilar, Publicar y Desplegar una función utilizamos el comando:

```
faas-cli up -f archivo_funcion.yml
```

- Puede utilizarse de forma individual cada proceso:
 - `faas-cli build -f archivo_funcion.yml`
 - `faas-cli push -f archivo_funcion.yml`
 - `faas-cli deploy -f archivo_funcion.yml`

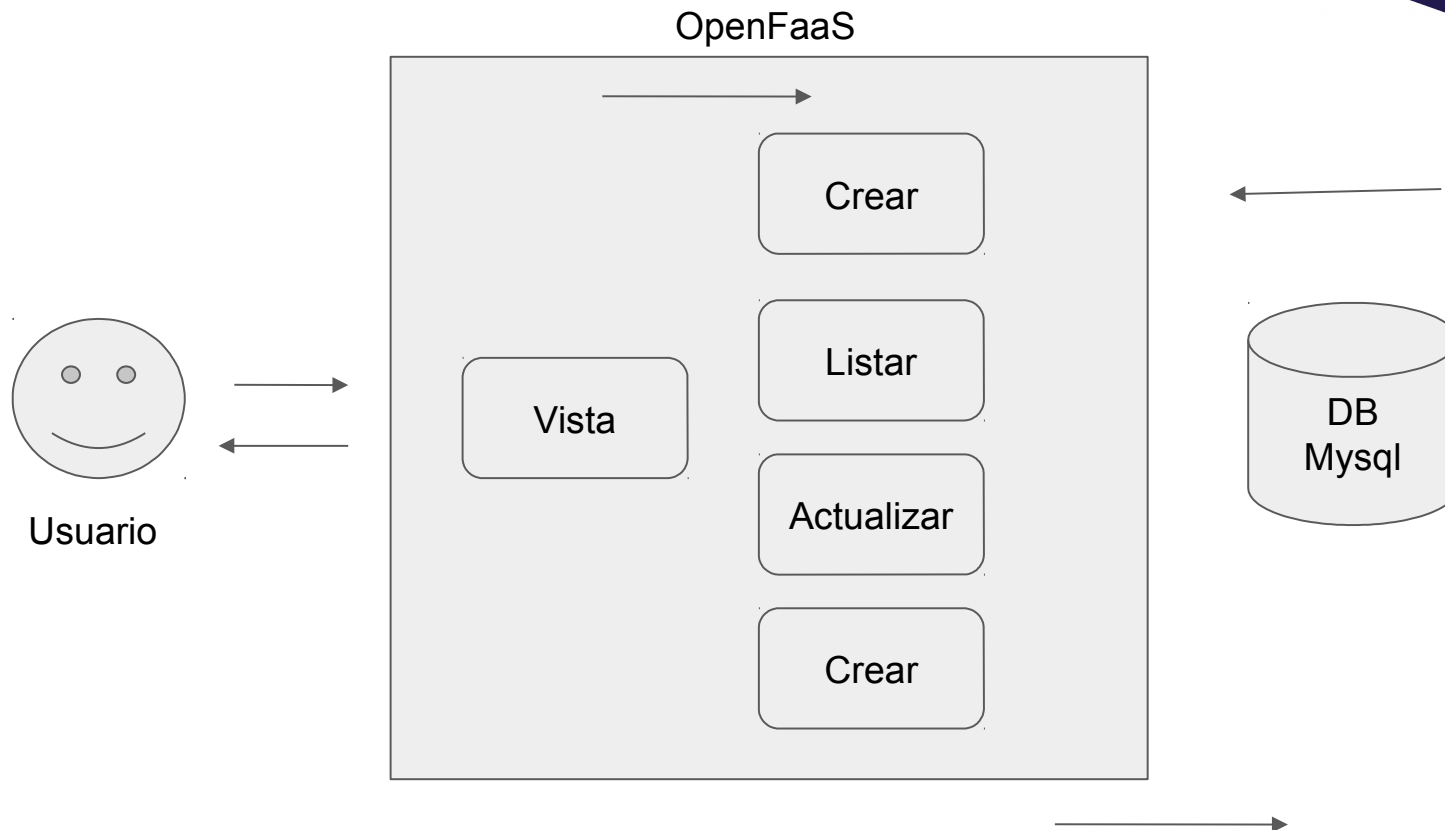
Publicar la Función en OpenFaaS



```
Archivo Editar Ver Buscar Terminar Ayuda  
[node1] (local) root@192.168.0.33 ~  
$
```

Demostración:
Single Page Application (SPA) con
OpenFaaS

Esquema de SPA



Funciones Publicadas



OpenFaaS Portal

Deploy New Function

Search for Function

listar-estudiante-openfaas

eliminar-estudiante-openfaas

crud-estudiante-openfaas

crear-estudiante-openfaas

actualizar-estudiante-openfaas

crud-estudiante-openfaas

Status	Replicas	Invocation count
Ready	1	24

Image	URL
vacax/crud-estudiante-openfaas:latest	http://192.168.77.10:8080/function/crud-estudiante-openfaas

Invoke function

INVOKE

☒ Text ☐ JSON ☐ Download

Request body

Response status	Round-trip (s)
-----------------	----------------

Response body

Demo SPA - OpenFaaS



192.168.77.10:8080/ui/

Aplicaciones Instagram Facebook YouTube Twitter Facebook http://voip.avati Wads Virtual Co www.nebrija.es/ Otros favoritos

OpenFaaS Portal

Deploy New Function

Search for Function

listar-estudiante-openfaas

eliminar-estudiante-openfaas

crud-estudiante-openfaas

crear-estudiante-openfaas

actualizar-estudiante-openfaas

crud-estudiante-openfaas

Status	Replicas	Invocation count
Ready	1	24

Image URL

vacax/crud-estudiante-openfaas:latest <http://192.168.77.10:8080/function/crud-estudiante-openfaas>

Invoke function

INVOKE

☒ Text ☐ JSON ☐ Download

Request body

Response status Round-trip (s)

Response body

Fuente Demo

<https://github.com/vacax/demo-serverless-openfaas>

Otros Frameworks Open Source

- Apache OpenWhisk.
- Fission.
- IronFunctions.
- Fn Project.
- Open Lambda.

¿Preguntas?

Contacto

Email: carlosalfredocamacho@gmail.com /
ccamachog@avathartech.com / ca.camacho@pucmm.edu.do
/ ccamachog@javadominicano.org

Twitter: [ccamachog](https://twitter.com/ccamachog)

Github: [vacax](https://github.com/vacax)

LinkedIn: [ccamachog](https://www.linkedin.com/company/ccamachog)

Muchas Gracias :-D

