“MicroProfile en Arquitecturas de MicroServicios”

VI Semana de la Informática 2020

10/02/2020

https://pixabay.com/es/contenedor-puerto-carga-apilados-3121310/

Luis Roldán – Digital Operations Lead – Altran Digital

cloudhispano

I consider myself a proactive person with strong technical skills (DevOps, BigData, Cloud, Java, Hybrid Apps), as well as team management, always focused on Agile methodology.

Passionate about the implementation of new software architectures and / or methodologies, trying to keep abreast of the trends in software development, but always trying to make the best decisions to solve the problems raised in the different projects and / or clients.

Interests:

Mobility

Always on | Anytime | Anywhere

BYOD | IoT | Native | Hybrid

Social

Everybody | Everything | Every channel

Digital Content | Gamification | Engagement| Personalization

Cloud

Availability | Scalability | Elasticity

Open Source | APIs | I/P/C/S aaS | Hybrid

Data

Velocity | Variety | Volume | Value

Big Data | Fast Data | SQL | NoSQL | Open Data



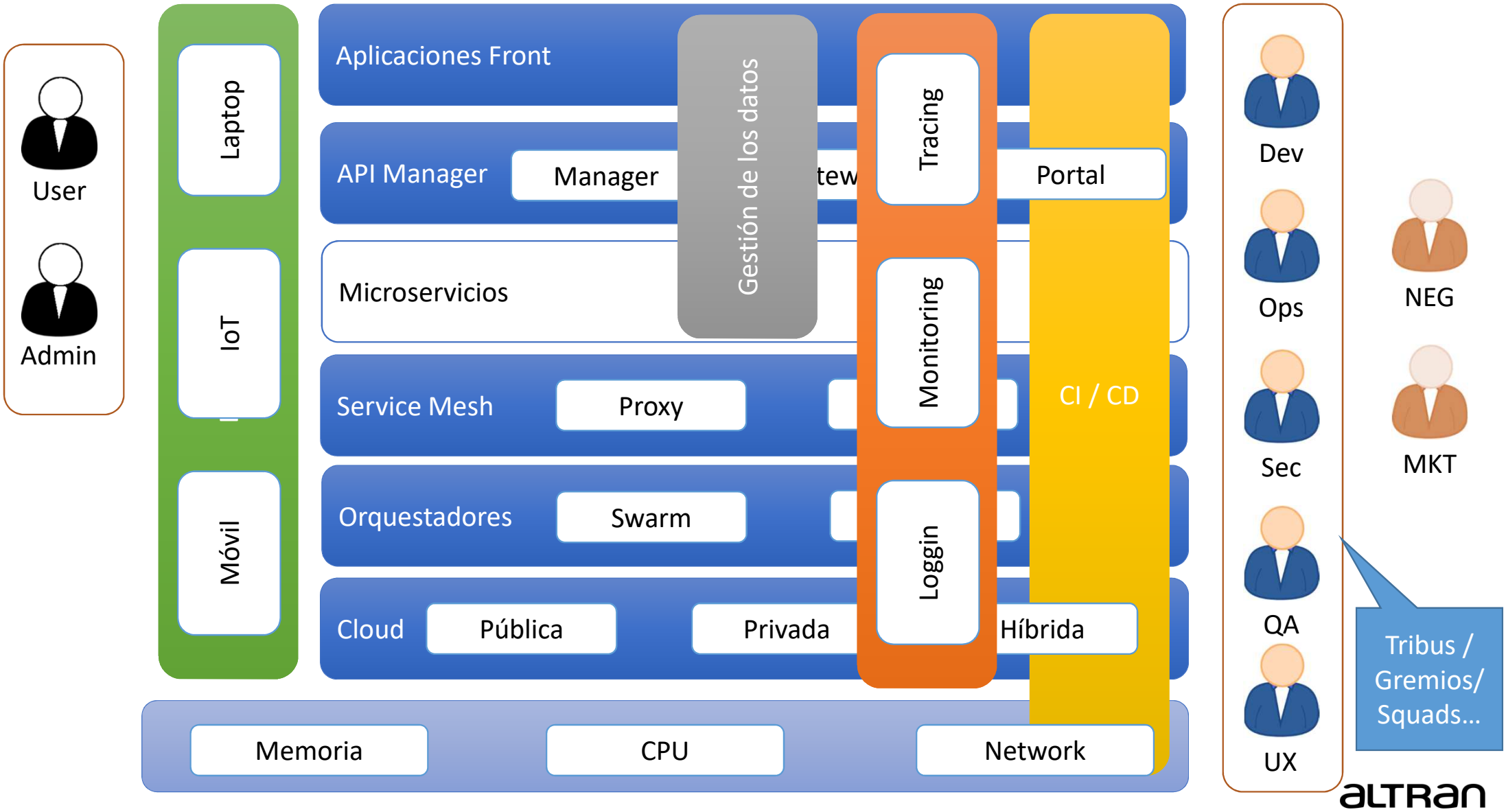

ALTRAN

Introducción

ALTRAN

Componentes

ALTRAN


User
Admin
Laptop
IoT
Móvil
Aplicaciones Front
API Manager
Manager
Portal
Microservicios
Service Mesh
Proxy
Orquestadores
Swarm
Cloud
Pública
Privada
Híbrida
Memoria
CPU
Network
Gestión de los datos
Tracing
Monitoring
Loggin
CI / CD
Dev
Ops
Sec
QA
UX
NEG
MKT
Tribus / Gremios/ Squads...
ALTRAN

Twelve Factors

I. Código base (Codebase)
Un código base sobre el que hacer el control de versiones y multiples despliegues

II. Dependencias
Declarar y aislar explícitamente las dependencias

III. Configuraciones
Guardar la configuración en el entorno

IV. Backing services
Tratar a los “backing services” como recursos conectables

V. Construir, desplegar, ejecutar
Separar completamente la etapa de construcción de la etapa de ejecución

VI. Procesos
Ejecutar la aplicación como uno o más procesos sin estado

VII. Asignación de puertos
Publicar servicios mediante asignación de puertos

VIII. Concurrencia
Escalar mediante el modelo de procesos

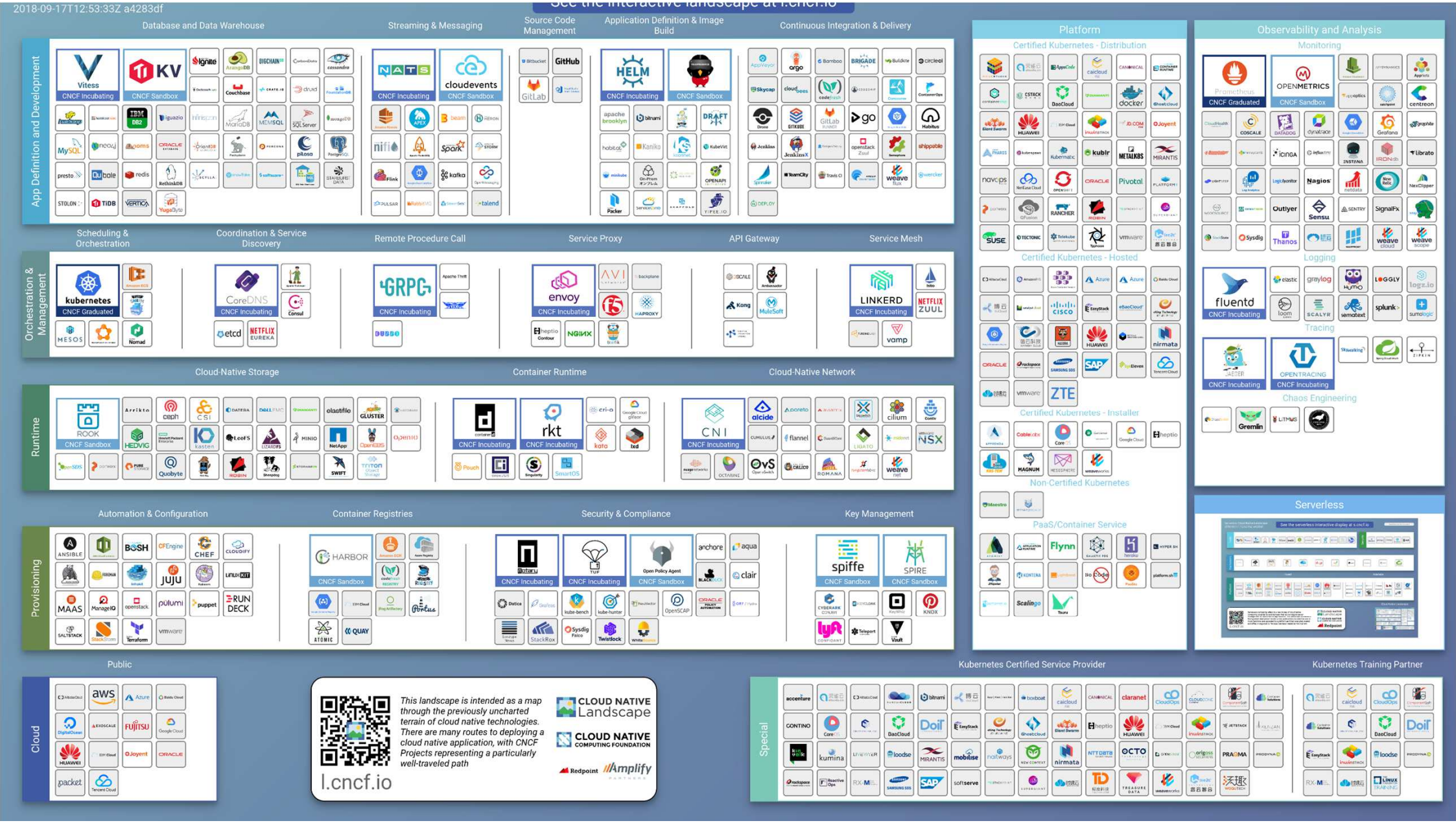
IX. Desechabilidad
Hacer el sistema más robusto intentando conseguir inicios rápidos y finalizaciones seguras

X. Paridad en desarrollo y producción
Mantener desarrollo, preproducción y producción tan parecidos como sea posible

XI. Historiales
Tratar los historiales como una transmisión de eventos

XII. Administración de procesos
Ejecutar las tareas de gestión/administración como procesos que solo se ejecutan una vez

ALTRAN


2018-09-17T12:53:33Z a4283df
Database and Data Warehouse
Streaming & Messaging
Source Code Management
Application Definition & Image Build
Continuous Integration & Delivery
App Definition and Development
Vitess
CNCF Incubating
KV
CNCF Sandbox
NATS
CNCF Incubating
cloudevents
CNCF Sandbox
HELM
CNCF Incubating
BRIGADE
CNCF Sandbox
Scheduling & Orchestration
Coordination & Service Discovery
Remote Procedure Call
Service Proxy
API Gateway
Service Mesh
Orchestration & Management
kubernetes
CNCF Graduated
CoreDNS
CNCF Incubating
GRPC
CNCF Incubating
envoy
CNCF Incubating
LINKERD
CNCF Incubating
Cloud-Native Storage
Container Runtime
Cloud-Native Network
Runtime
ROOK
CNCF Sandbox
rkt
CNCF Incubating
CNI
CNCF Incubating
Automation & Configuration
Container Registries
Security & Compliance
Key Management
Provisioning
HARBOR
CNCF Sandbox
spiffe
CNCF Sandbox
SPIRE
CNCF Sandbox
Public
Cloud
aws
Azure
Platform
Certified Kubernetes - Distribution
Certified Kubernetes - Hosted
Certified Kubernetes - Installer
Non-Certified Kubernetes
PaaS/Container Service
Observability and Analysis
Monitoring
Prometheus
CNCF Graduated
OPENMETRICS
CNCF Sandbox
Logging
fluentd
CNCF Incubating
Tracing
JAEGER
CNCF Incubating
OPENTRACING
CNCF Incubating
Chaos Engineering
Serverless
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path
l.cncf.io
CLOUD NATIVE Landscape
CLOUD NATIVE COMPUTING FOUNDATION
Redpoint
Amplify
Kubernetes Certified Service Provider
Kubernetes Training Partner
Special

Microprofile

ALTRAN

ROADMAP

MicroProfile especifica una colección de APIs y tecnologías Java EE que juntas forman la línea de base para crear Microservicios, que tiene como objetivo proporcionar la portabilidad de la aplicación en múltiples entornos de ejecución.

MicroProfile
3.2
(11/11/2019)

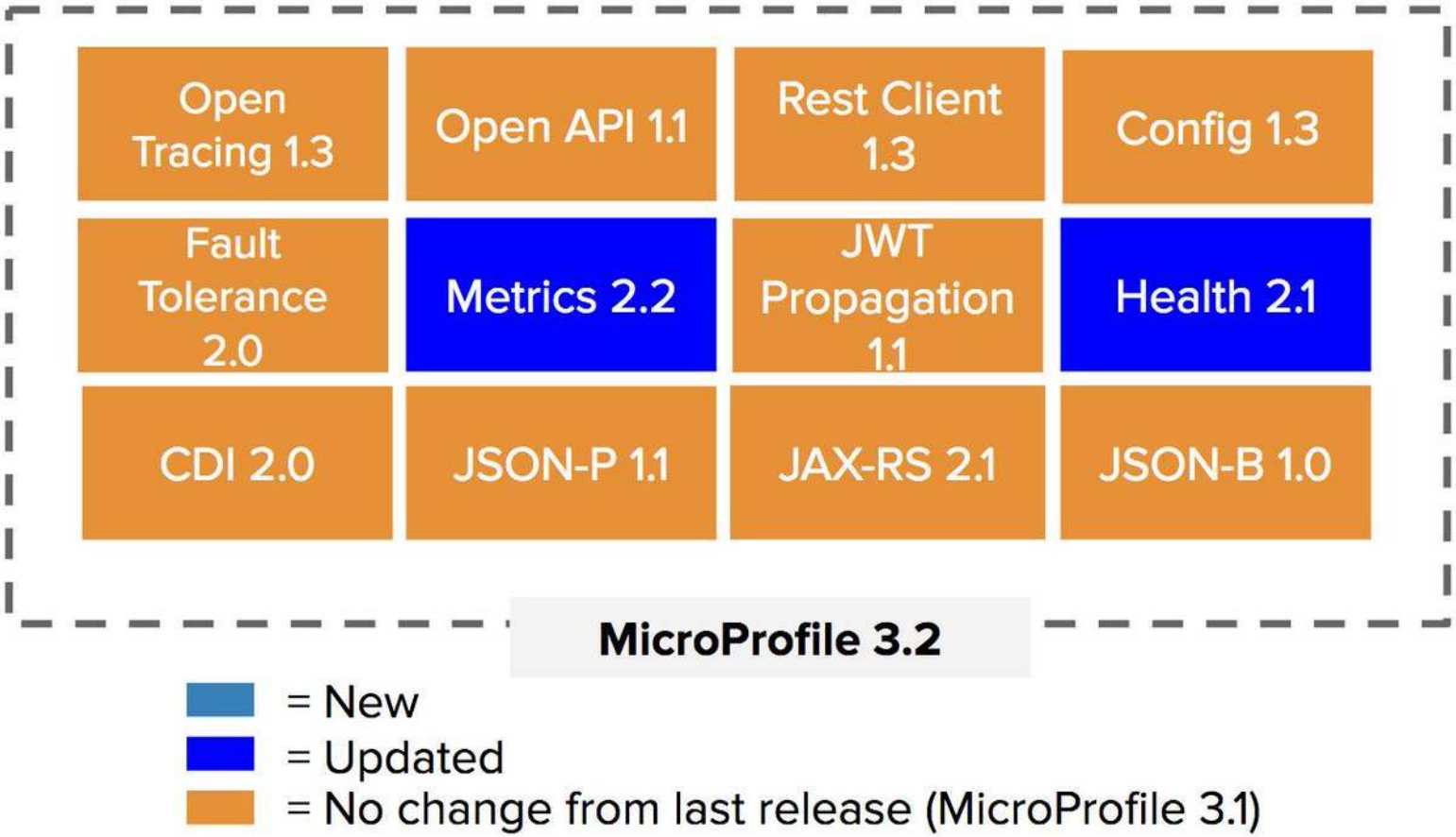
https://microprofile.io/

ALTRAN

MicroProfile implementations & open source projects

- Red Hat - Thorntail
- Red Hat - Red Hat Application Runtimes
- IBM - WebSphere Liberty
- IBM - Open Liberty
- Payara Foundation - Payara Micro
- Payara Foundation - Payara Server
- Tomitribe - TomEE
- Oracle - Helidon
- Fujitsu - Launcher
- SmallRye
- Hammock
- KumuluzEE

PROYECTOS


Open Tracing 1.3
Open API 1.1
Rest Client 1.3
Config 1.3
Fault Tolerance 2.0
Metrics 2.2
JWT Propagation 1.1
Health 2.1
CDI 2.0
JSON-P 1.1
JAX-RS 2.1
JSON-B 1.0
MicroProfile 3.2
= New
= Updated
= No change from last release (MicroProfile 3.1)

CDI

- Contexto e Inyección de dependencias
- Ciclo de Vida y Inyección de Tipos
- Productores
- Interceptores
- Observadores

JSON-P

- Parseado, generación, transformación, búsquedas sobre JSON
- Streaming API
- Object Model API

JAX-RS

- Servicios web REST
- Basado en anotaciones
- HTTP

```
3  import javax.ws.rs.GET;
4
5  public interface TestServ {
6          @GET
7          public String test1();
8  }
```

JSON-B

- Convierte un objeto Java en json y viceversa

```
public void printJson() {
    Item item = new Item(1, "calculator");
    Jsonb jsonb = JsonBuilder.create();
    String jsonString = jsonb.toJson(item);
    System.out.println(jsonString);
   }
```

CONFIG

Gestión de variable de configuración:

- META-INF/microprofile-config.prop
- Variables de entorno
- Propiedades de sistema

```java
import javax.enterprise.context.ApplicationScoped;
import javax.inject.Inject;
import javax.ws.rs.GET;
import javax.ws.rs.Path;

import org.eclipse.microprofile.config.inject.ConfigProperty;
@ApplicationScoped
@Path("config")
public class ConfigTest {
        @Inject
        @ConfigProperty(defaultValue="Hola" ,name = "SALUDO")
        String saludo;

        @Inject
        @ConfigProperty(name = "NOMBRE")
        String nombre;

        @GET
        public String getSaludo() {
                return saludo + ", " +nombre;
        }
```

FAULT TOLERANCE

La propuesta de Fault Tolerance continene los aspectos: TimeOut, RetryPolicy, Fallback, Bulkhead y CircuitBreaker.

- TimeOut: define una duración para el tiempo de espera
- RetryPolicy: define un criterio sobre cuándo reintentar
- Fallback: proporciona una solución alternativa para una ejecución fallida.
- Bulkhead: limita el numero de llamadas concurrentes a un servicio.
- CircuitBreaker: ofrece una manera de fallar rápidamente fallando automáticamente en la ejecución para evitar la sobrecarga del sistema y la espera indefinida o el tiempo de espera de los clientes.

METRICS

Diferentes tipos de métricas:

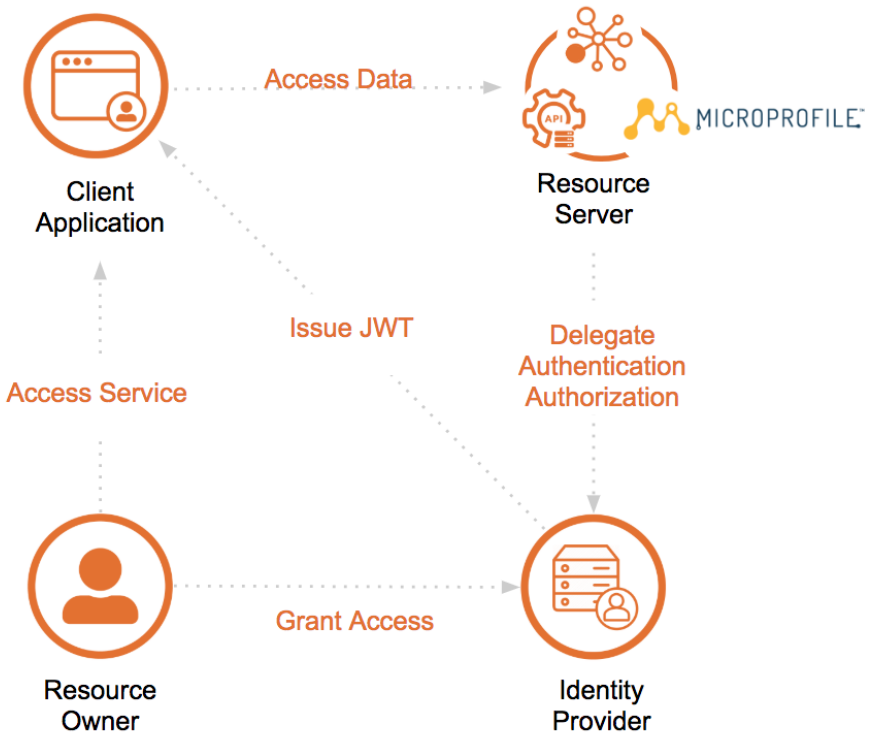
- Base
- Vendor
- Aplicación

```java
import org.eclipse.microprofile.metrics.MetricUnits;
import org.eclipse.microprofile.metrics.annotation.Counted;
import org.eclipse.microprofile.metrics.annotation.Gauge;
import org.eclipse.microprofile.metrics.annotation.Metered;

@ApplicationScoped
@Path("test")
public class Test1 {

	@Inject
	MetricsTest mt;

	@Metered(name = "itemsSold", unit = MetricUnits.MINUTES, description = "Metrics to monitor sold method.",
	@GET
	public String test1() {
		mt.getRandomGauge();
		return "Test1! " + mt.getRandomGauge() + " " + System.currentTimeMillis();
	}

	@Gauge(unit = "USD", name = "itemsPrice", absolute = true)
	public long getPrice() {
		return 4;
	}

	@Counted(unit = MetricUnits.NONE, name = "itemsCheckedOut", absolute = true, monotonic = true, displayNam
			"checkout=items" })
	@GET
	@Path("/counter")
	public Valor checkoutItems() {
		return new Valor(1,"nuevo");
	}
```

JWT PROPAGATION

JWT se compone de:

- Header: el encabezado contiene metadatos como el tipo de algoritmo utilizado para firmar el token (HS256, RS256, ES256,etc), el tipo de token (OpenID Connect, OAuth2, Microprofile JWT), etc.
- Claim: la información que desea almacenar en el token.
- Firma



https://jwt.io/

ALTRAN

HEALTH CHECK

Health checks se utiliza para probar el estado de un servicio desde otro servicio (por ejemplo, el controlador de servicio de Kubernetes).

```
5  import org.eclipse.microprofile.health.Health;
6  import org.eclipse.microprofile.health.HealthCheck;
7  import org.eclipse.microprofile.health.HealthCheckResponse;
8
9  @Health
10 @ApplicationScoped
11 public class HealthTest implements HealthCheck {
12
13         public HealthCheckResponse call() {
14                 // TODO Auto-generated method stub
15                  return HealthCheckResponse.named("successful-check").up().build();
16         }
17
18 }
```

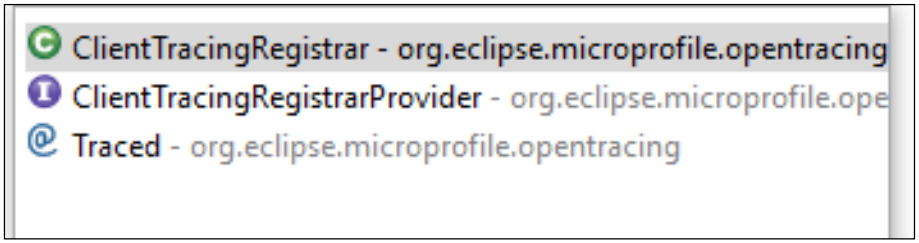
ALTRAN

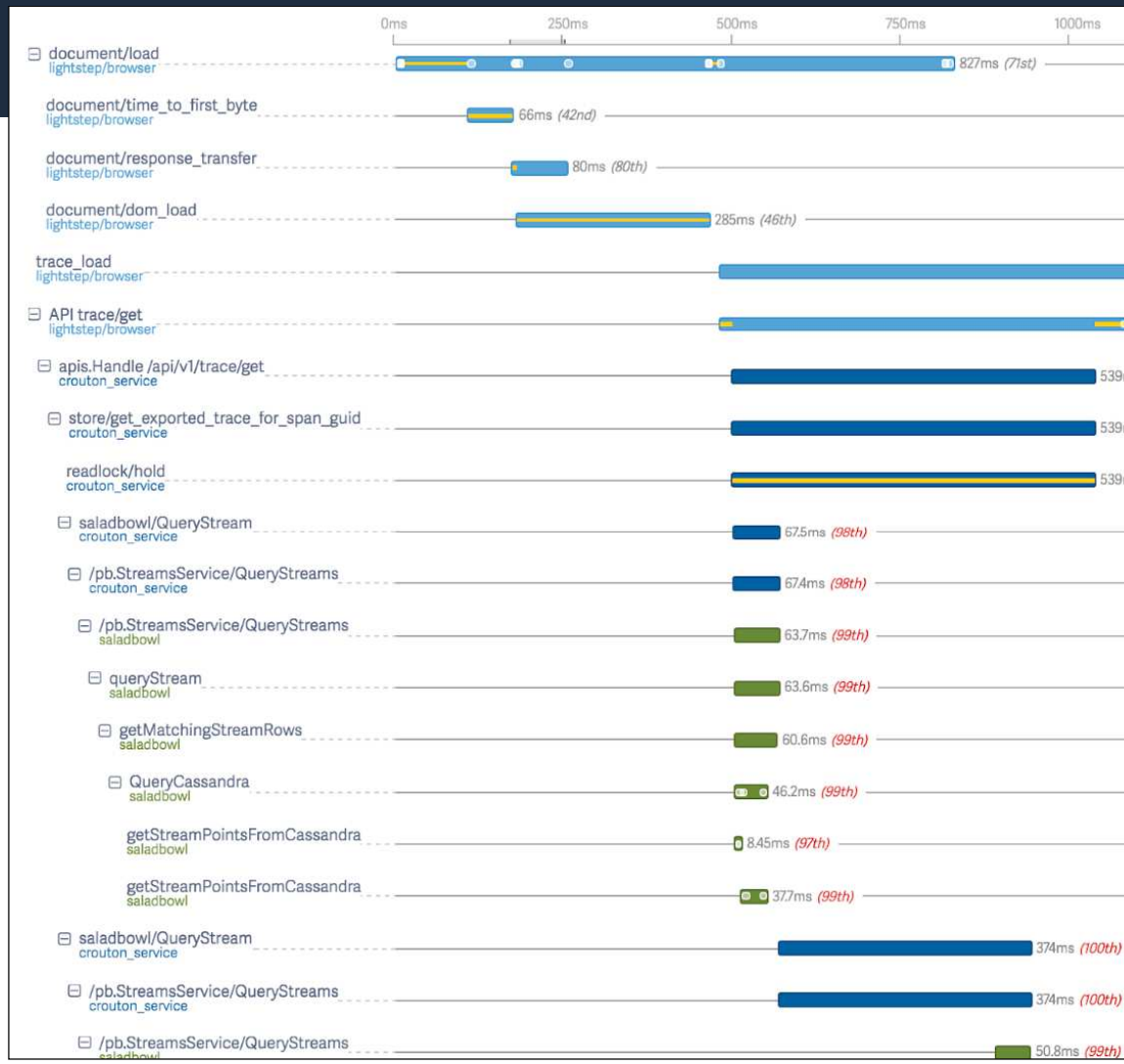
OPEN TRACING

- Permite gestionar la trazabilidad

de las peticiones entre microservicios

- Basado en anotaciones

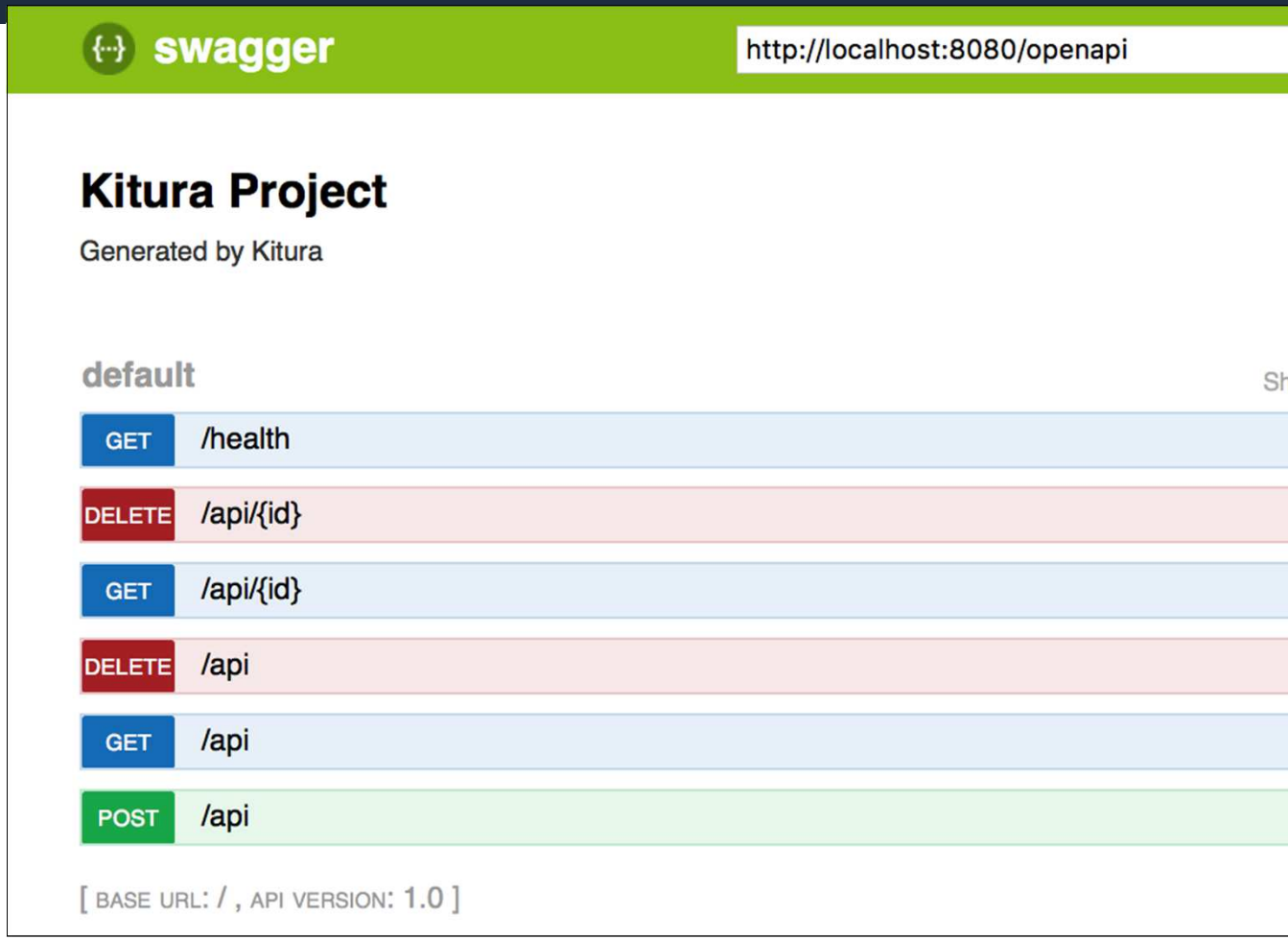




ALTRAN

OPEN API

- Open API v3 esta basado en Swagger v2
- Basado en anotaciones
- Incluido en el servidor



REST CLIENT

- Cliente de Servicios web REST

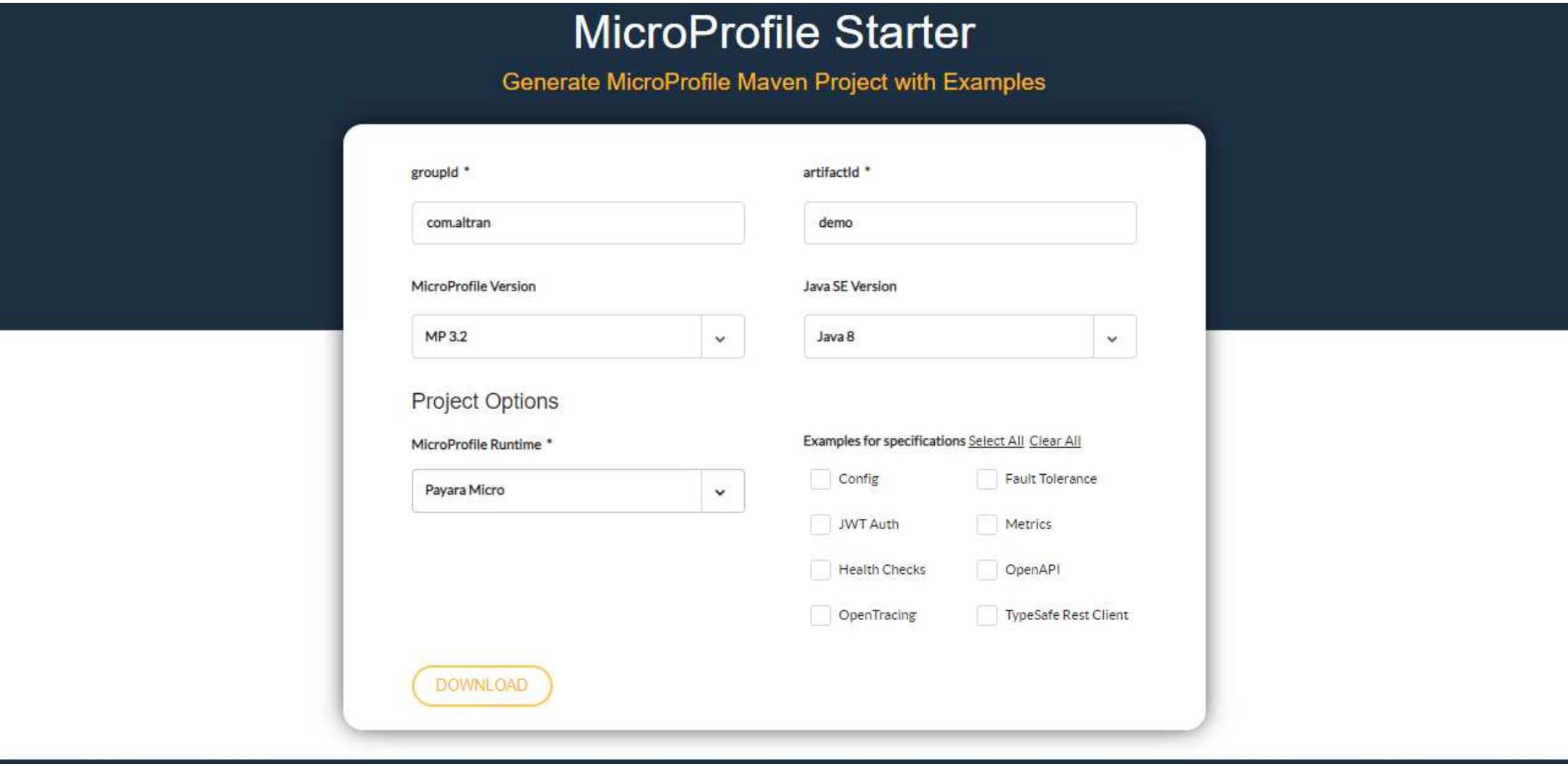
- Basado en tipos (type-safe)

```
9
10  import org.eclipse.microprofile.rest.client.RestClientBuilder;
11
12  @ApplicationScoped
13  @Path("cliente")
14  public class HttpClient {
15
16          @GET
17          public String getValor() throws URISyntaxException {
18                  URI apiUri = new URI("http://172.31.121.177:8080/mp/v1/test");
19                  TestServ valorServ = RestClientBuilder.newBuilder().baseUri(apiUri).build(TestServ.class);
20                  return valorServ.test1();
21                  //return "valor";
22          }
23
24  }
```

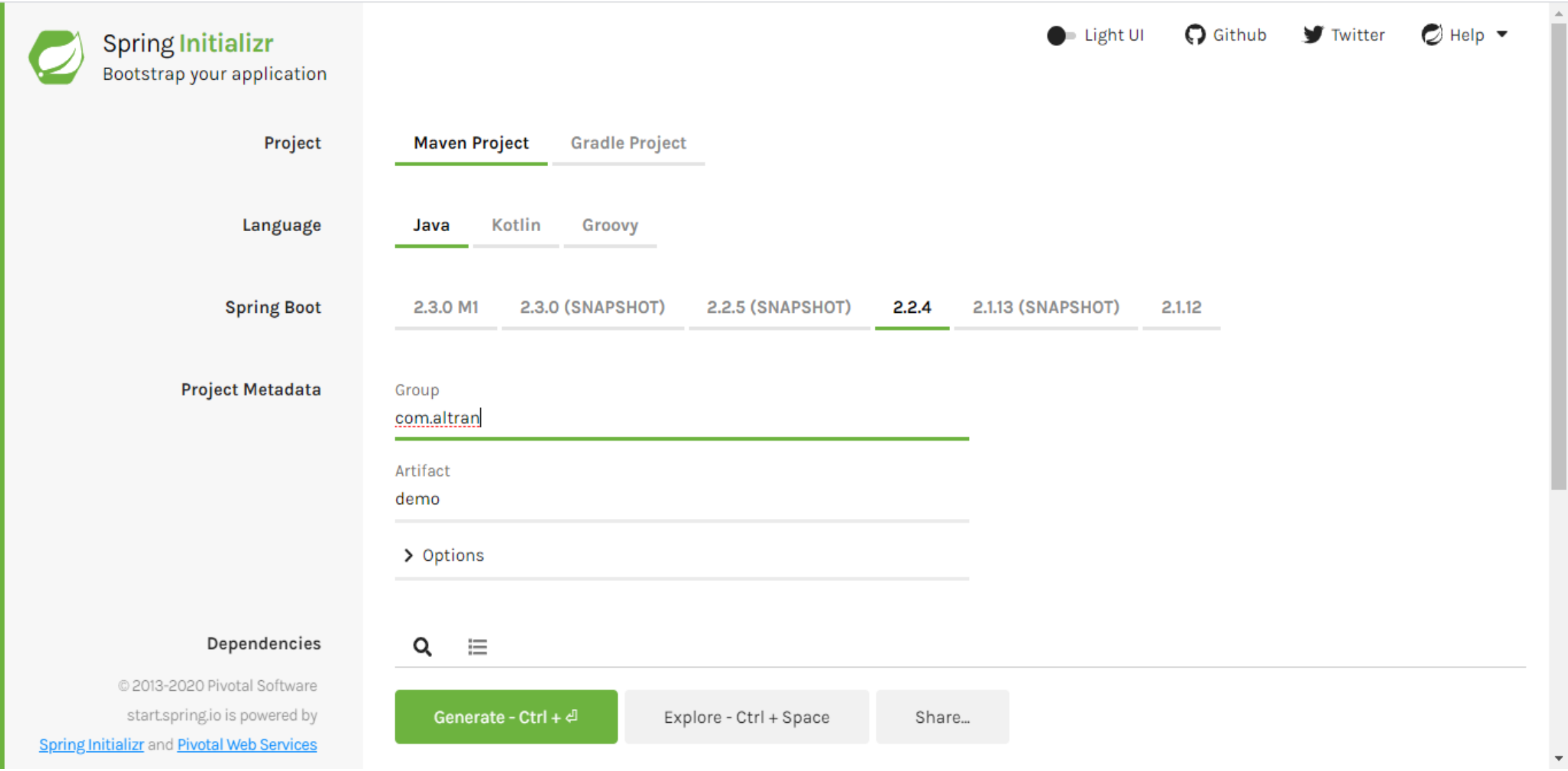
DEMO

ALTRAN

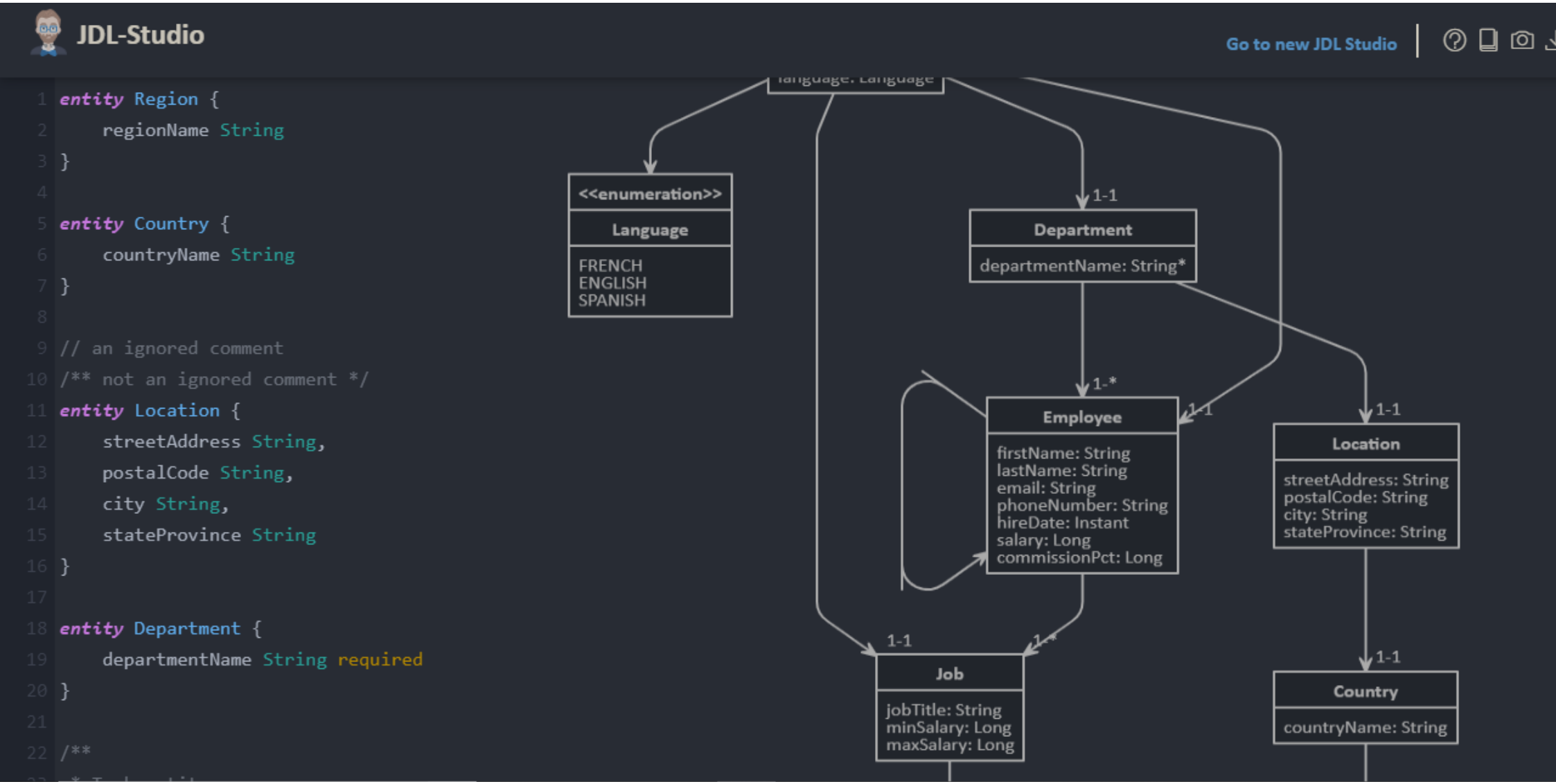
https://start.microprofile.io/



https://start.spring.io/



https://www.jhipster.tech/jdl-studio/



¿Preguntas?

Conclusiones

ALTRAN

PLAN DE INTEGRACIÓN

Programa BK2!

UNIRTE AL PROGRAMA DE BECAS BK2! ES

1. ***Sumergirte en Tecnología***
2. ***Hacer amigos***
3. ***Aprender***
4. ***Colaborar y participar***

BK2! es el **plan de Iniciación profesional** que proporciona Altran a **estudiantes** universitarios, de master, recién titulados y de Formación Profesional.

El **objetivo** principal del Programa BK2! es ofrecer una visión profesional y práctica de la formación que reciben los estudiantes o recién graduados, durante sus estudios, con el fin de **favorecer su inserción profesional** en el mercado laboral.

PROGRAMA BK2!

Un lugar donde ***SER EQUIPO***

Intégrate y comparte aficiones con tus compañeros a través de nuestras Hobby Pills, Alternates, Technitives...

#ThisIsJustTheBeginning

ALTRAN


ALTRAN