Construir microservicios en Python. v3.0

Expectativas

VS

REALIDAD

Microservicios en el mundo real

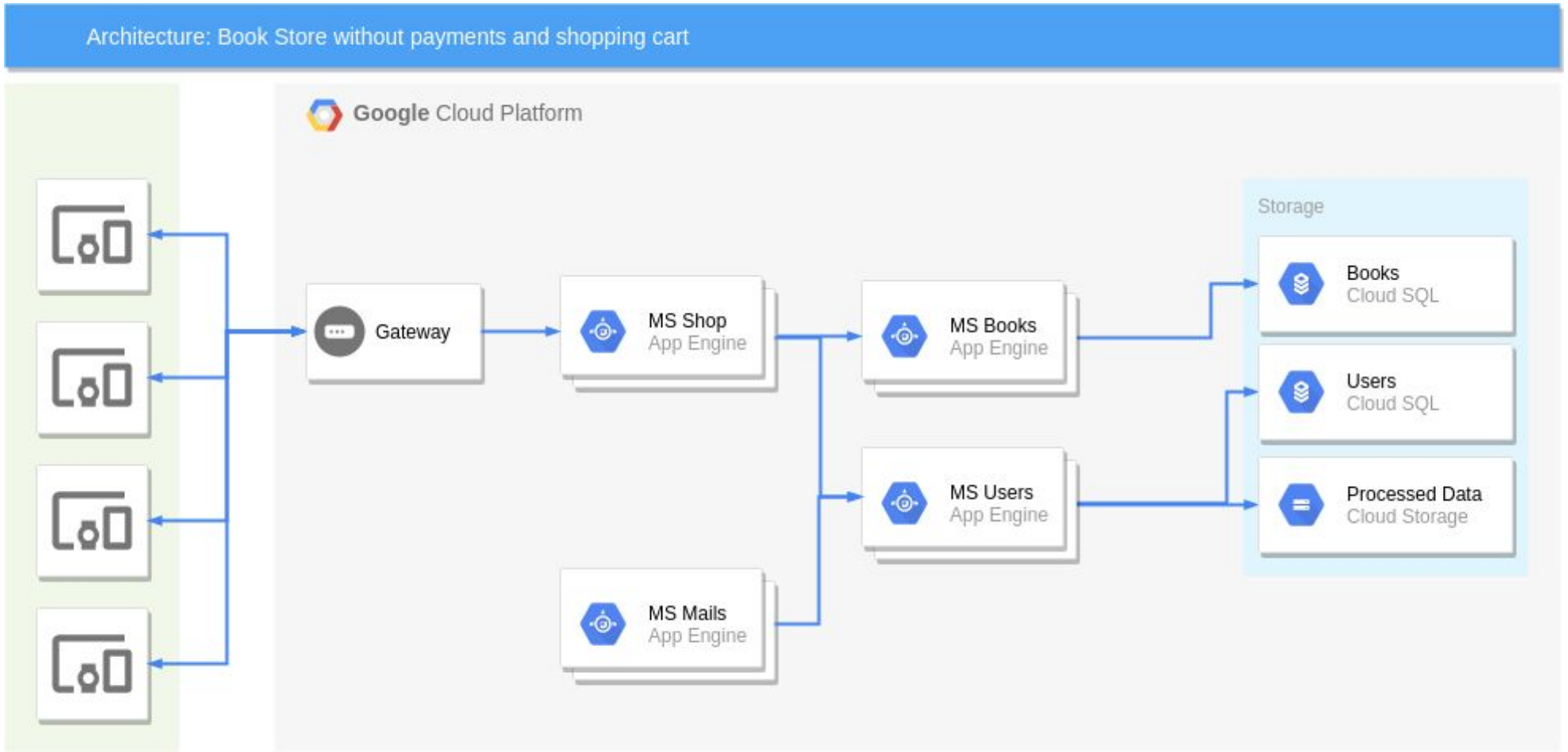
Blogs y tutoriales de internet al buscar “Build microservice in XXX”

- **Añade una ruta GET: /users/**
- **Añade otra ruta PUT: /users/**
- **devuelve un JSON:**

```
[{
        “id”: 1,
        “name”: “ImTheBoss”
}]
```

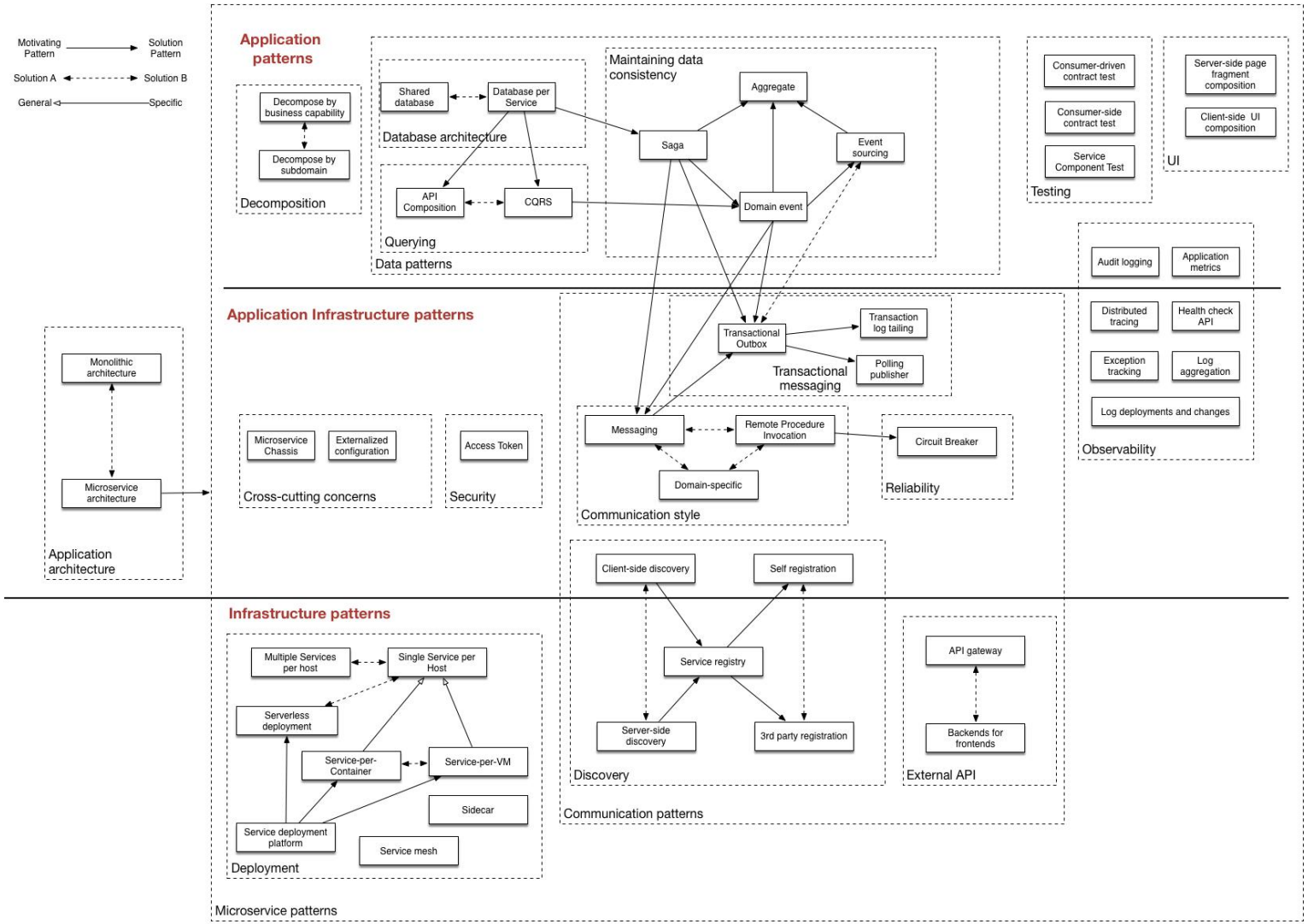
- **Actualiza Linkedin con “Microservice Architect”**
- **Wait...**

El cliente que te pide hacer esto:



El cliente que te pide hacer esto:

Patrones de microservicios



Patrón: Microservice chassis

- Documentación
- Configuración externalizada
- Trazabilidad de peticiones
- logging
- Health checks
- Metrics

https://microservices.io/patterns/microservice-chassis.html

github.com/python-microservices

PyMS, la librería construida y basada en Flask que unifica todas las librerías necesarias para construir un microservicio. Entre ellas:

- Flask (obvio).
- Connexion y Swagger.
- Prometheus.
- Opentracing y Jaegger (WIP: Opentelemetry)
- Anyconfig.
- Consul.
- Cryptography.

Más literatura de cómo llegué a esto (DRY):
paradigmadigital.com/dev/como-construir-microservicios-en-python-1-2/
paradigmadigital.com/techbiz/microservice-chassis-pattern-python-2-2/

Librería y arquetipos: cómo contribuir

Librería Patrón Chasis para Microservicios:
github.com/python-microservices/pyms

Arquetipo en el que nos hemos basado:
github.com/python-microservices/microservices-scaffold

Template con Cookiecutter:
github.com/python-microservices/cookiecutter-pyms

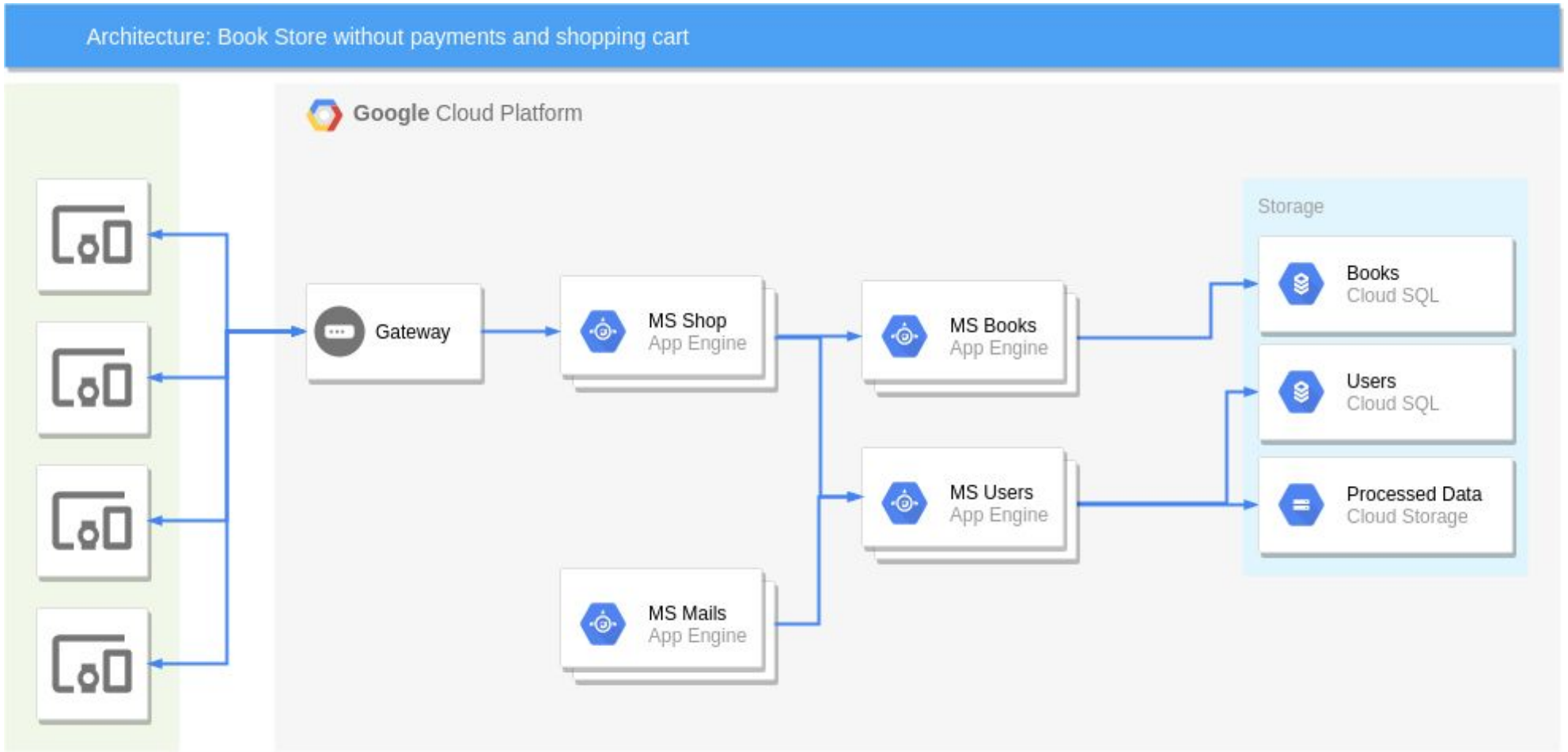
Ejemplos

Ejemplos sencillos:
github.com/python-microservices/pyms/tree/master/examples

Ejemplo con Kubernetes:
github.com/python-microservices/microservices-chat

Ejemplo con Docker Compose:
github.com/avara1986/pivoandcode-2019-11-15

El cliente que te pide hacer esto:



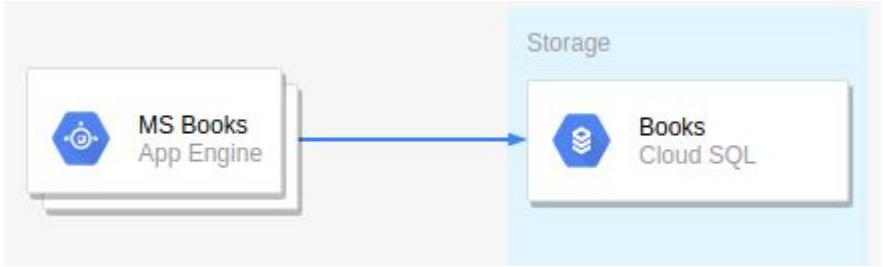
Problema


Gateway
MS Shop
App Engine
MS Books
App Engine
Storage
Books
Cloud SQL

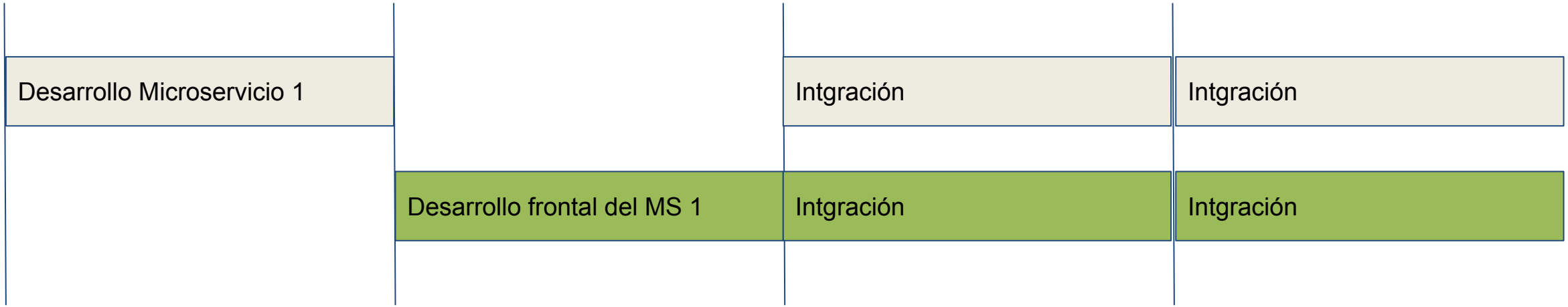

MS Books
App Engine
Storage
Books
Cloud SQL


MS Shop
App Engine
MS Books
App Engine

Problema: Documentación



Problema: Documentación



Problema: Documentación

OpenAPI

Estándar para la descripción de APIS

Swagger

Tecnología más popular para las documentaciones de API

~~API First~~ Spec First

Documentar primero antes de trabajar

Problema: Documentación

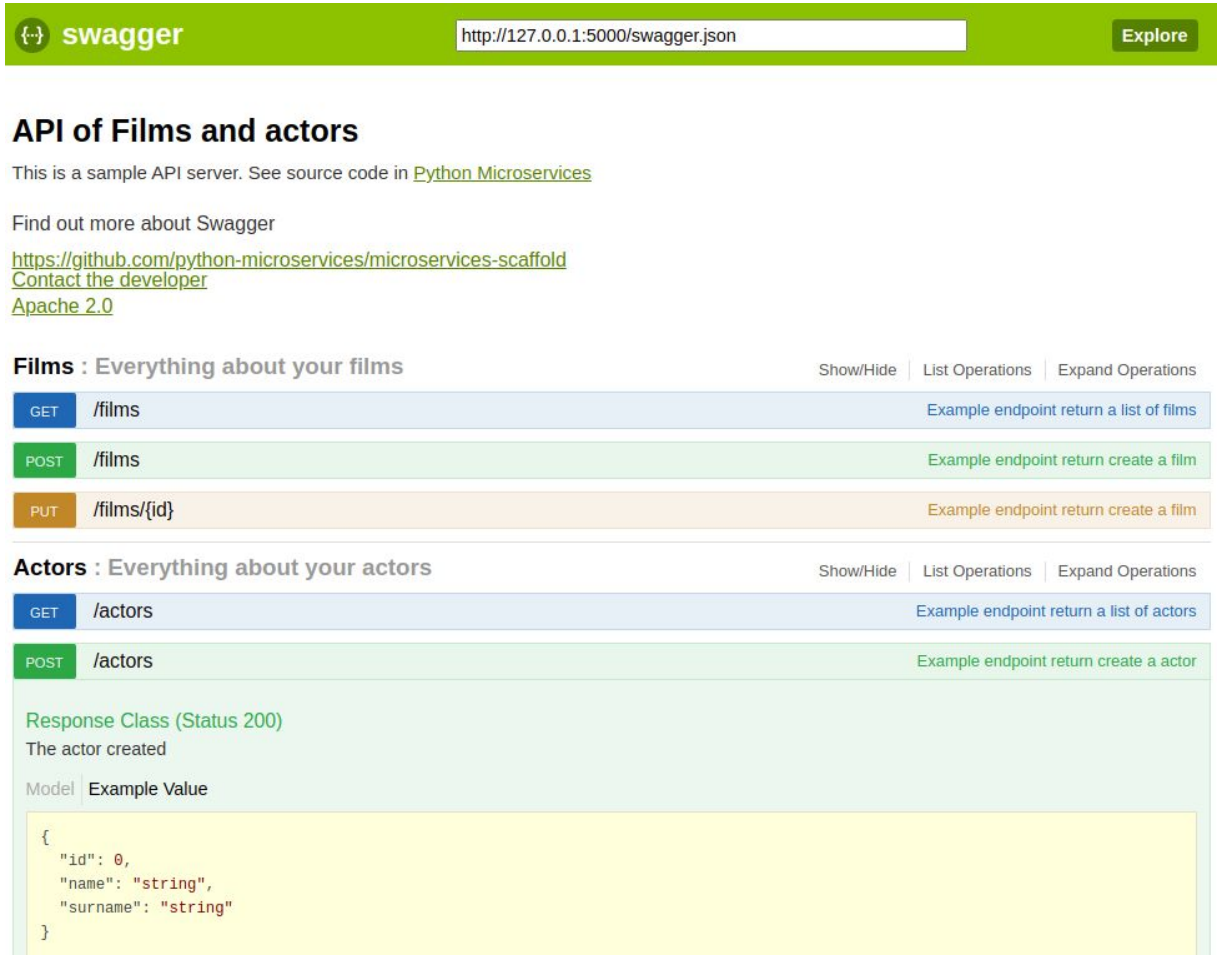
OpenAPI + Swagger

```yaml
    get:
      tags:
      - "Actors"
      summary: "Example endpoint return a list of actors"
      description: ""
      consumes:
      - "application/json"
      produces:
      - "application/json"
      responses:
        "200":
          description: "A list of actors"
          schema:
            $ref: '#/definitions/Actors'
        "400":
          description: "Invalid ID supplied"
        "404":
          description: "Actor not found"
        "405":
          description: "Validation exception"
```

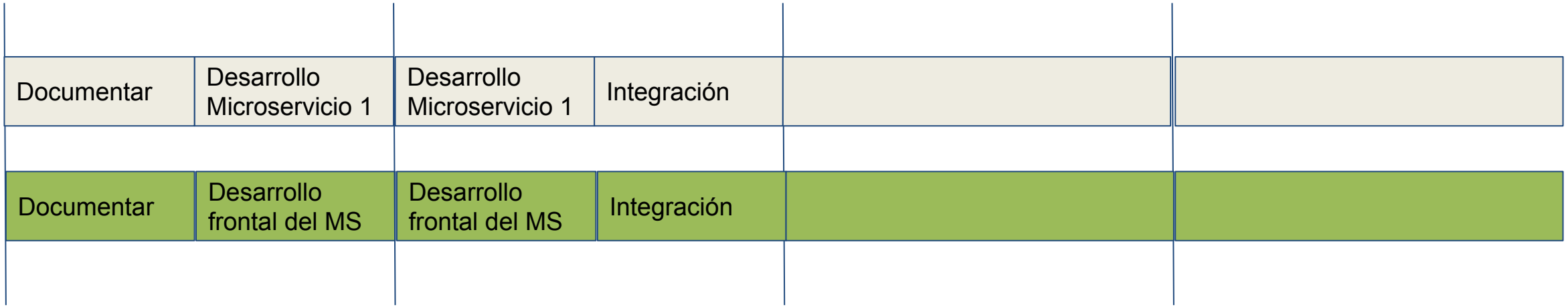
```yaml
    post:
      tags:
      - "messages"
      summary: "Example endpoint return create a messages"
      description: ""
      operationId: "create_view"
      consumes:
      - "application/json"
      produces:
      - "application/json"
      parameters:
      - name: user_id
        in: formData
        type: string
        required: true
        description: "Id of the user"
      - name: username
        in: formData
        type: string
```

Problema: Documentación

OpenAPI + Swagger



Problema: Documentación



Problema: Documentación

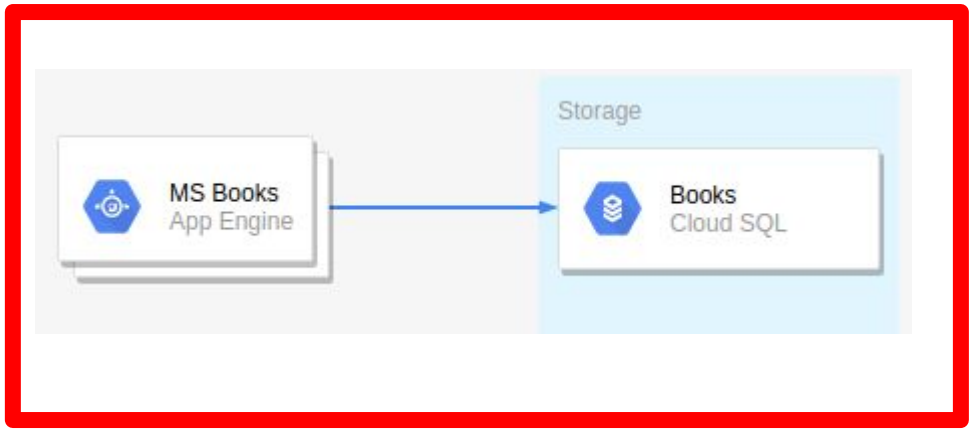
Documentar	Desarrollo Microservicio 1	Desarrollo Microservicio 1	Integración	
Documentar	Desarrollo frontal del MS	Desarrollo frontal del MS	Integración	

Problema: Documentación

Proyectos y librerías OpenAPI + Swagger

- Connexion connexion.readthedocs.io
- django-rest-framework www.django-rest-framework.org
- flask-restplus flask-restplus.readthedocs.io
- flask-restful flask-restful.readthedocs.io
- FastAPI fastapi.tiangolo.com

Problema: Configuración y desarrollo en local



Problema: Configuración y desarrollo en local

Problema: Configuración y desarrollo en local

```
class Config:
    DEBUG = False
    TESTING = False
    APP_NAME = "Template"
    SQLALCHEMY_DATABASE_URI = "sqlite:/..."

class DevConfig(Config):

class TestConfig(Config):

class PreConfig(Config):

class ProConfig(Config):
```

Problema: Configuración y desarrollo en local

Uso de variables de entorno. config.py

```
class Config:
    DEBUG = False
    TESTING = False
    APP_NAME = "Template"
    SQLALCHEMY_DATABASE_URI = os.env.get("DB")
```

Configuración externalizada. config.yaml

```
ms:

 DEBUG: false

 TESTING: false

 APP_NAME: Template

 SQLALCHEMY_DATABASE_URI : sqlite:/…
```

Problema: Configuración y desarrollo en local

Docker compose

```
services:

  postgresql:

    image: postgres

  books:

    image:

    depends_on:

      - mysql

  shop:

    image:

    depends_on:

      - books
```

Kubernetes (Minikube) y Helm

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: {{ include "chat_db.fullname" . }}
 labels:
spec:
 replicas: {{ .Values.replicaCount }}
 template:
   metadata:
     labels:
       app.kubernetes.io/name: {{ include "chat_db.name" . }}
   spec:
     containers:
       - name: {{ .Chart.Name }}
         image: "{{ .Values.image.repository }}:{{
.Values.image.tag }}"
         ports:
           - name: http
             containerPort: 8080
             protocol: TCP
```

Problema: Configuración y desarrollo en local

Proyectos y librerías

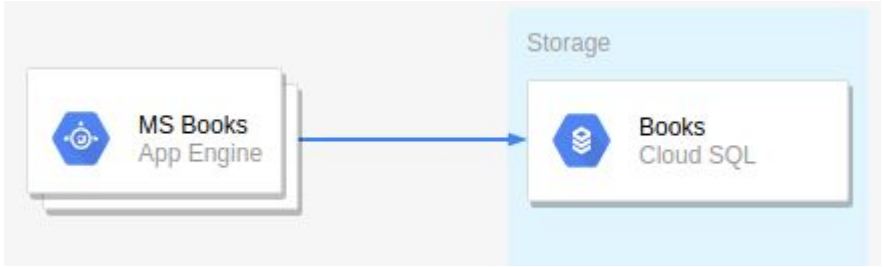
- PyYAML https://pyyaml.org/
- anyconfig github.com/ssato/python-anyconfig
- Vault github.com/hvac/hvac
- Kubernetes Secrets kubernetes.io/es/docs/concepts/configuration/secret/

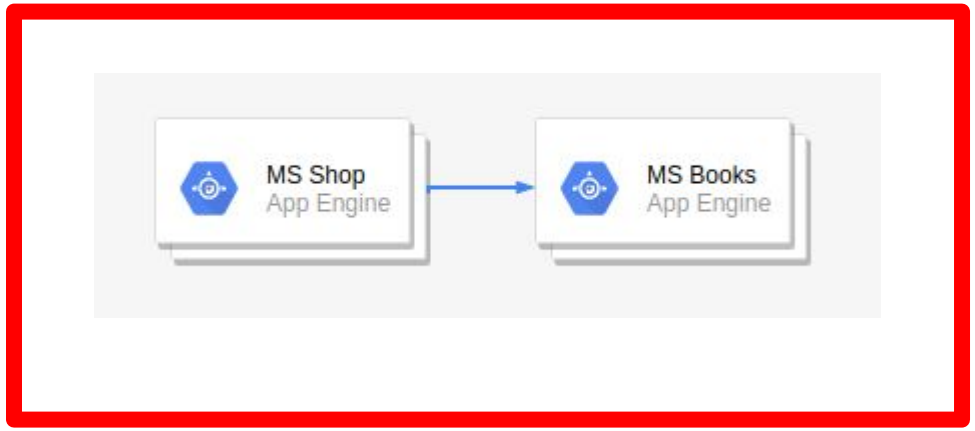
Ejemplos

Ejemplo con Kubernetes:
github.com/python-microservices/microservices-chat

Ejemplo con Docker Compose:
github.com/avara1986/pivoandcode-2019-11-15

Problema: Trazabilidad y Logging





Trazabilidad Distribuida

En una frase: poder hacer seguimiento de una request a través de los diferentes microservicios por donde pasa

Pero la trazabilidad engloba muchísimo más

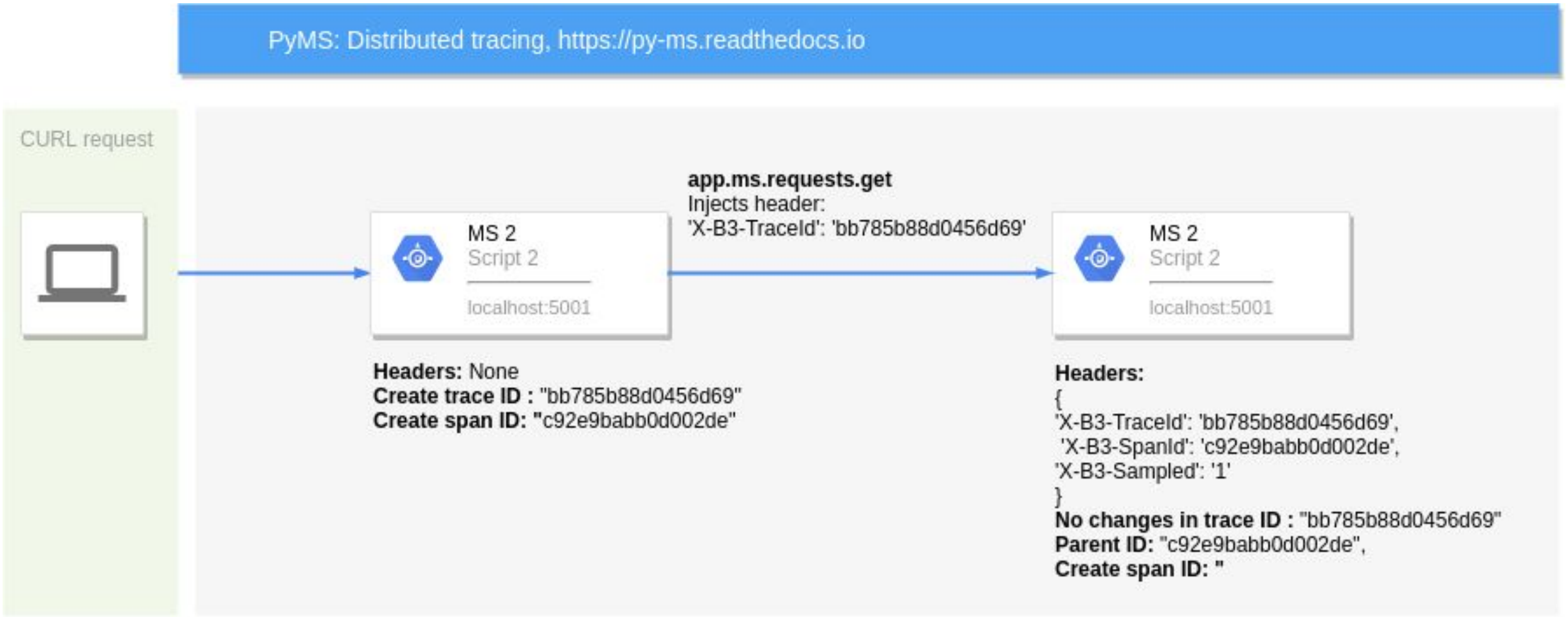
Más Info:

Trazabilidad distribuida paradigmadigital.com/dev/trazabilidad-distribuida-con-opentracing-y-jaeger/

Patrón de Trazabilidad distribuida microservices.io/patterns/observability/distributed-tracing

Ejemplo de trazabilidad: python-microservices.github.io/tutorials/tutorial_propagate_traces/

Trazabilidad Distribuida



Jaeger

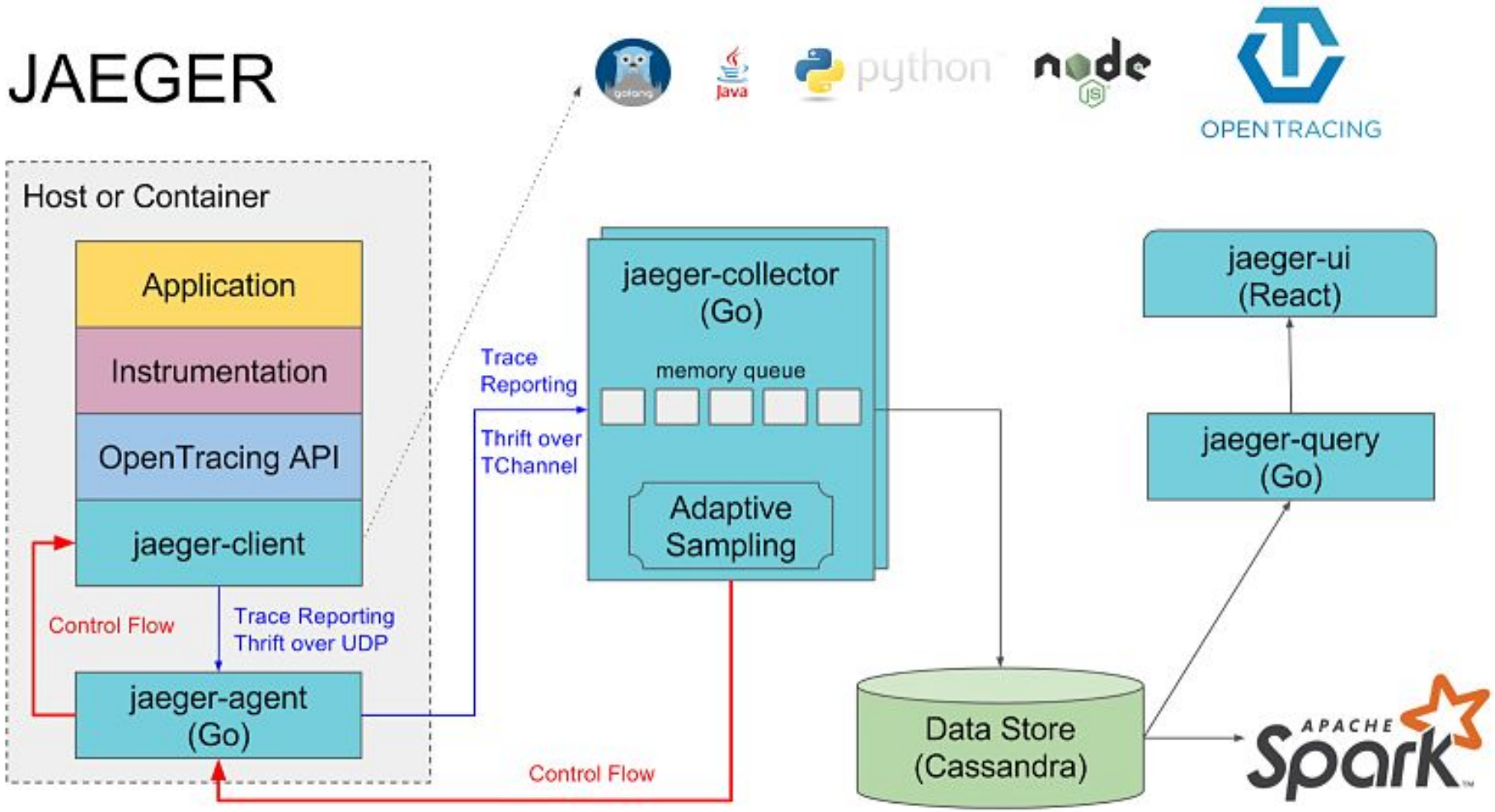


Imagen de paradigmadigital.com/dev/trazabilidad-distribuida-con-opentracing-y-jaeger/

Problema: Trazabilidad

Clientes

- jaeger jaegertracing.io
- LightStep lightstep.com
- Datadog datadoghq.com
- ElasticAPM elastic.co
- Opentracing opentracing.io
- OpenTelemetry opentelemetry.io

Healthcheck y Metrics

Metrics

http://localhost:8000/metrics

```
# HELP go_gc_duration_seconds A summary of the GC
invocation durations.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 3.291e-05
go_gc_duration_seconds{quantile="0.25"} 4.3849e-05
go_gc_duration_seconds{quantile="0.5"} 6.2452e-05
go_gc_duration_seconds{quantile="0.75"} 9.8154e-05
go_gc_duration_seconds{quantile="1"} 0.011689149
go_gc_duration_seconds_sum 3.451780079
go_gc_duration_seconds_count 13118
```

Healthcheck

http://localhost:8000/healthcheck

```
curl -IX GET "http://localhost:8000/healtcheck"
HTTP/1.1 200
Content-Type: text/html; charset=utf-8
Date: Sun, 15 Sep 2019 12:32:22 GMT
```

Healthcheck y Metrics

Proyectos y librerías

- Prometheus github.com/prometheus/client_python
- Graphana grafana.com

¡GRACIAS POR VUESTRO TIEMPO!

github.com/avara1986/pycones-2021

github.com/avara1986

linkedin.com/in/albertovara/

twitter.com/avara1986

a.vara.1986@gmail.com

Hdiv

¡Hdiv estamos contratando!
careers@hdivsecurity.com