Diseño de API

con OpenAPI

METADEV

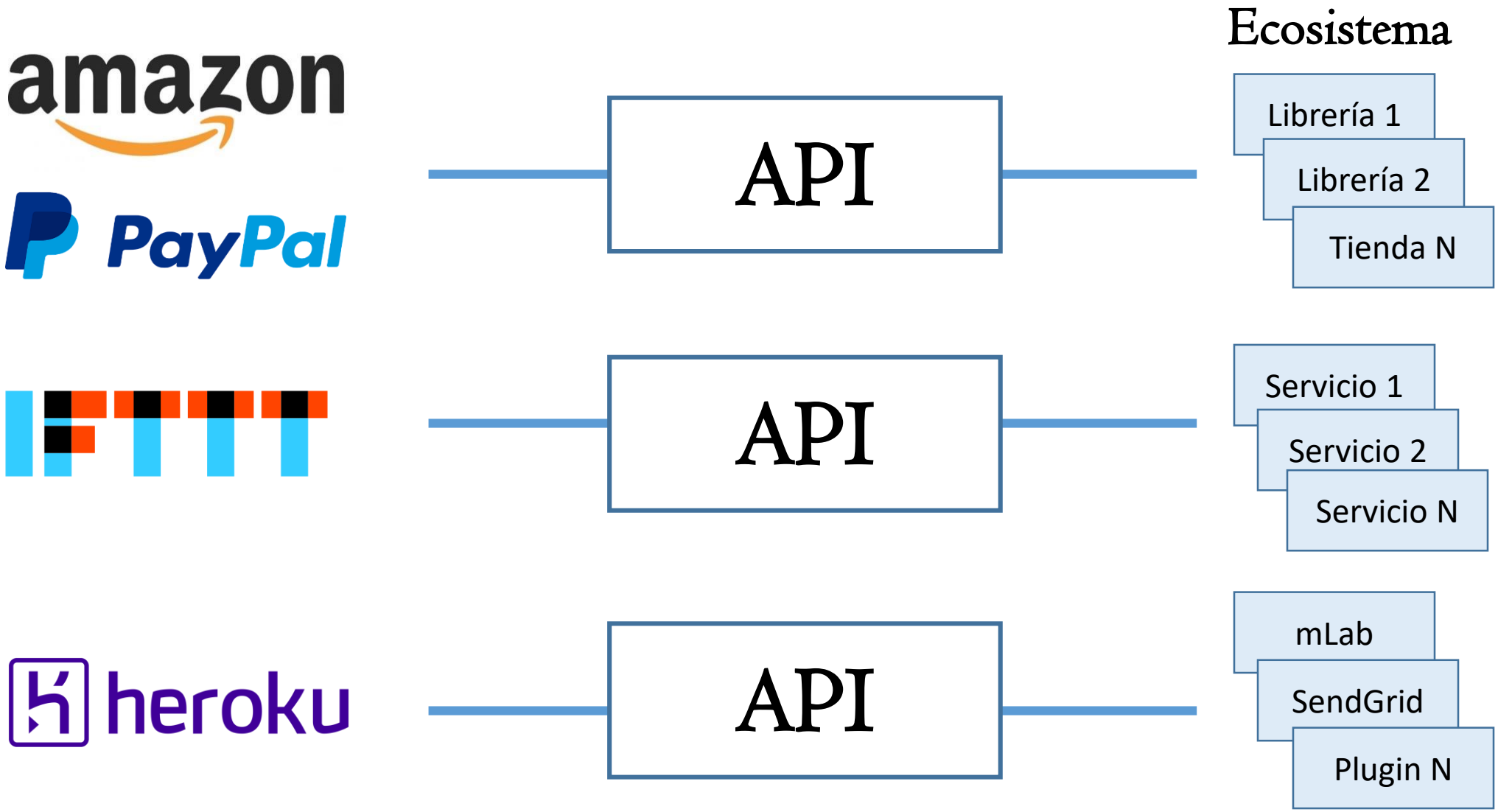
https://metadev.pro

Pedro J. Molina

https://pjmolina.com

@pmolinam

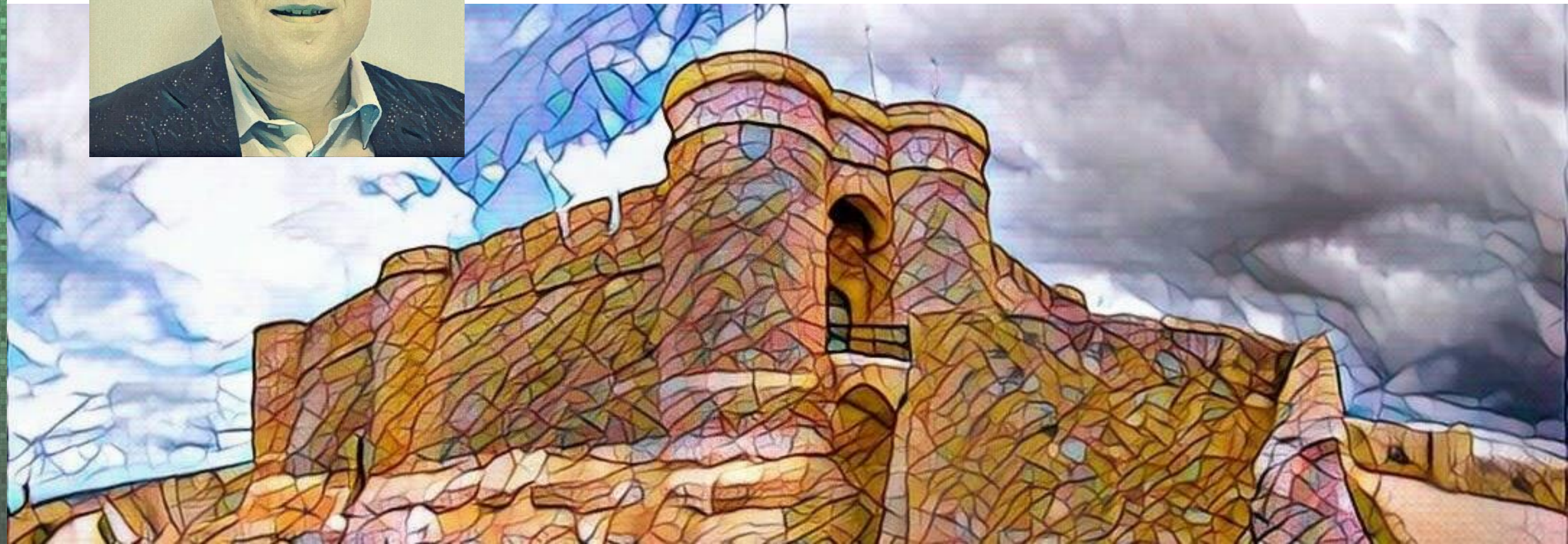
¿Qué tienen en común estos modelos de negocios?



Pedro J. Molina

@pmolinam

METADEV



Agenda

- API *Economy*
- OpenAPI
- Casos de uso
- Versionado
- Futuro

API

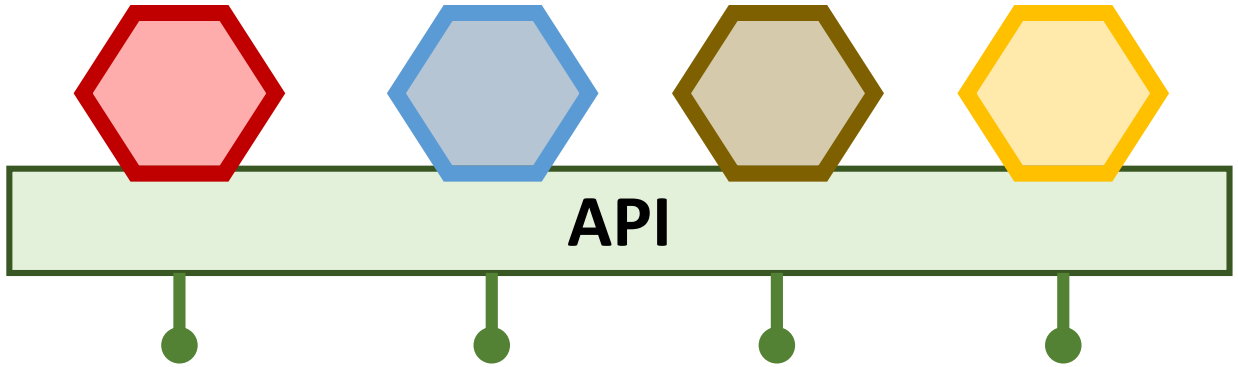
Application **P**rogrammer **I**nterface

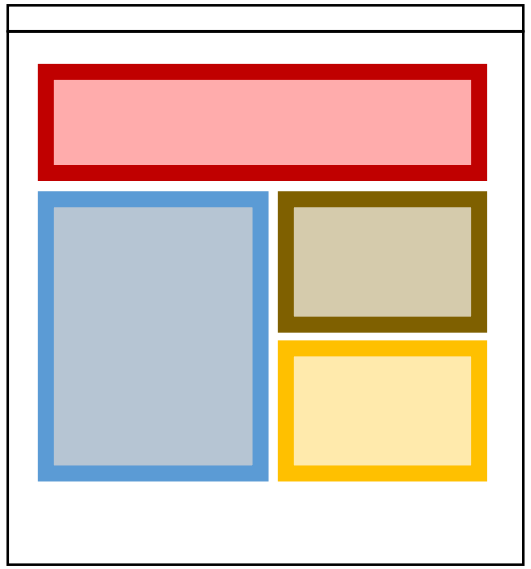
- Servicio públicos para que 3ºs puedan consumirlos
- Descripción técnica (orientado a devs)
- Promueve la integración de sistemas mediante contratos claros y perdurables en el tiempo

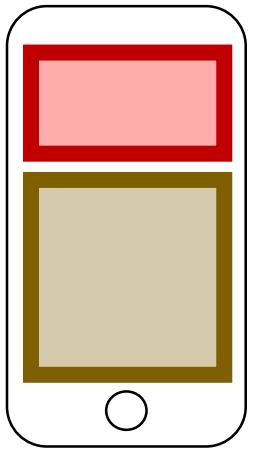
API Economy

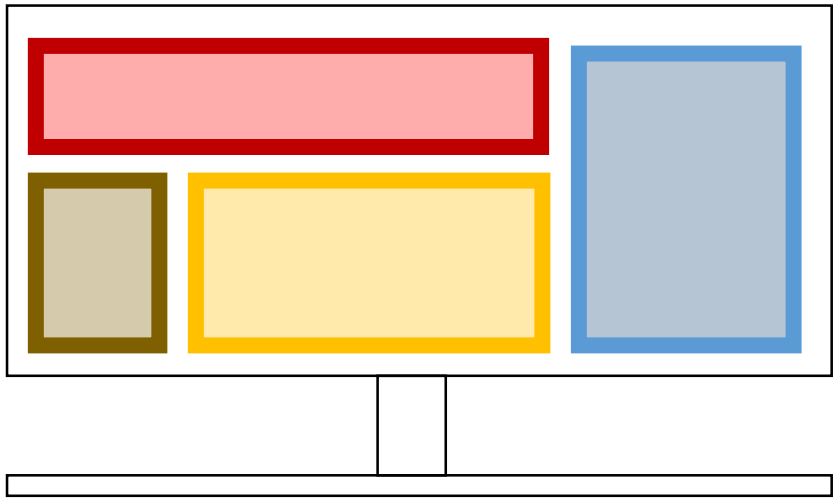
- Las APIs definen **plataformas**.
- Twitter, Twilio, Google Maps son ejemplos de APIs abiertas que permiter a 3ºs integrarse con sus sistemas.
- No puedes ganar sin **ecosistema**.
- No puedes tener ecosistema sin **API**.
- El primero que gana la **cuota de mercado** → gana el juego.

API como contrato con clientes

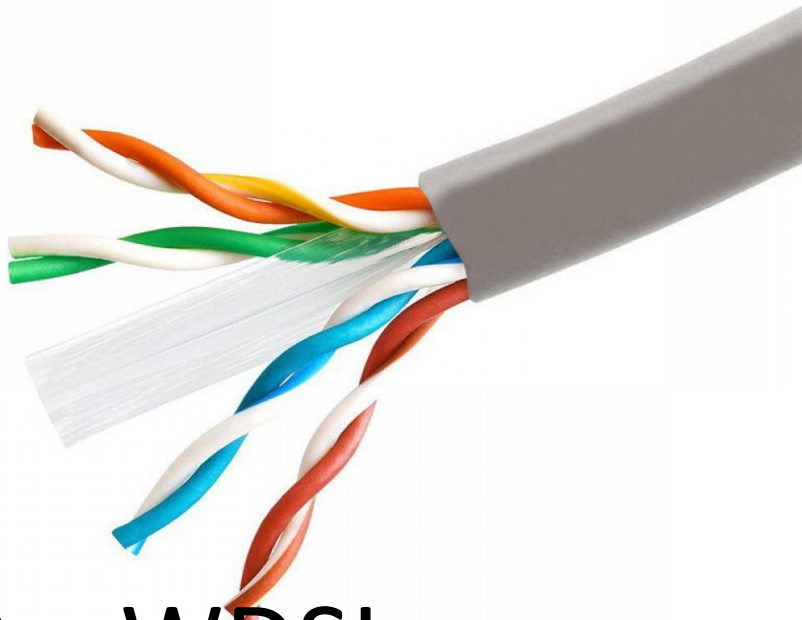




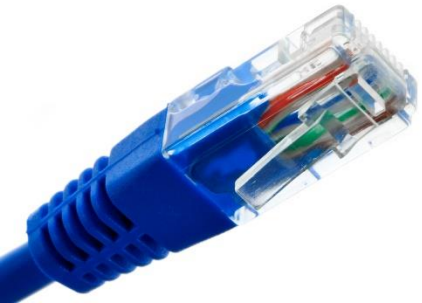




APIs agnósticas de lenguaje



1. CORBA >> C + IDL
2. SOA >> XML + SOAP + WDSL …
3. REST >> JSON + HTTP

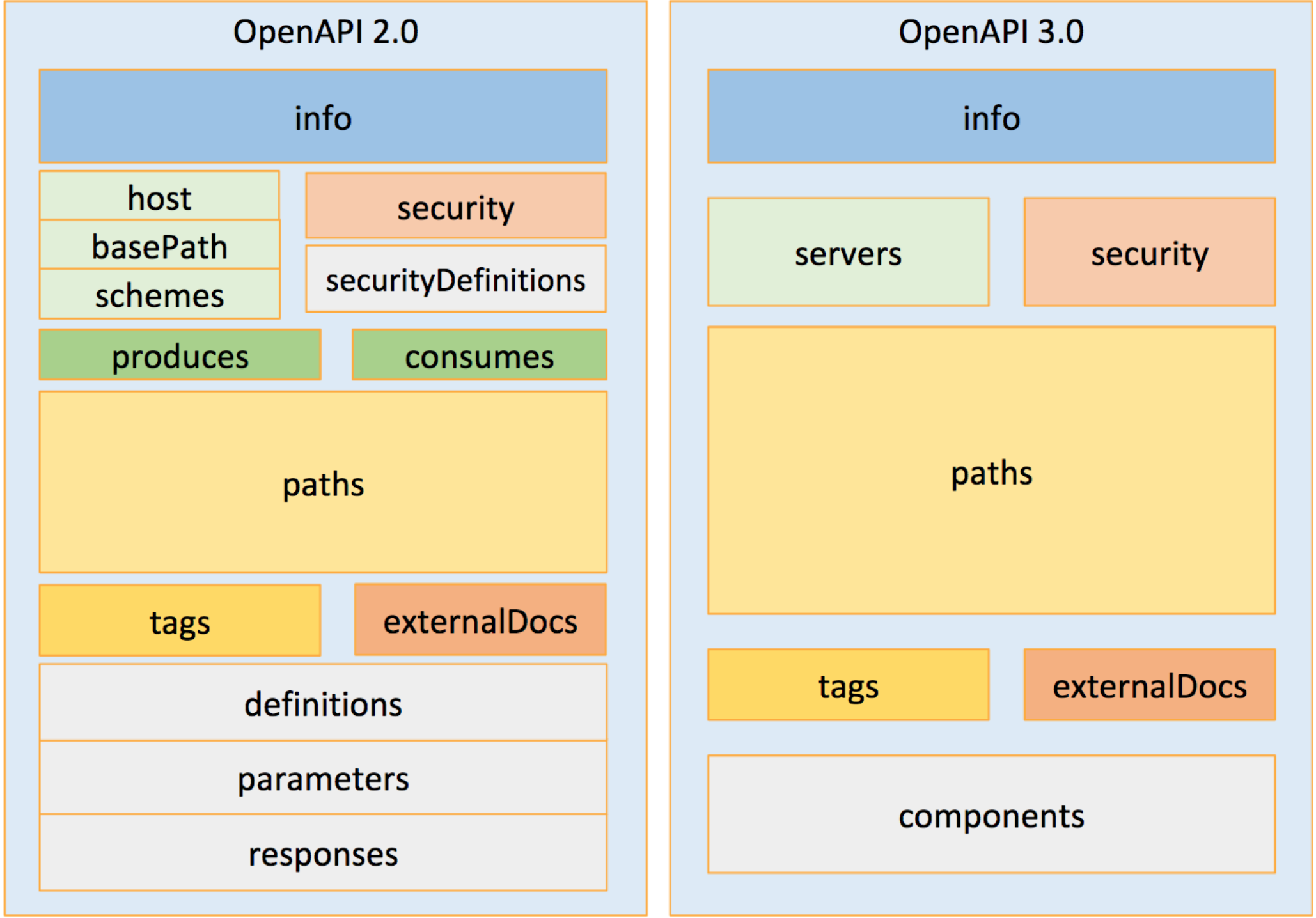


OpenAPI Initiative

- Estándar de facto (Swagger)
- Linux Foundation https://www.openapis.org

- Descripción formal del contrato de un API REST consumible por maquinas y humanos.

- JSON o YAML


OpenAPI 2.0
info
host
basePath
schemes
security
securityDefinitions
produces
consumes
paths
tags
externalDocs
definitions
parameters
responses
OpenAPI 3.0
info
servers
security
paths
tags
externalDocs
components

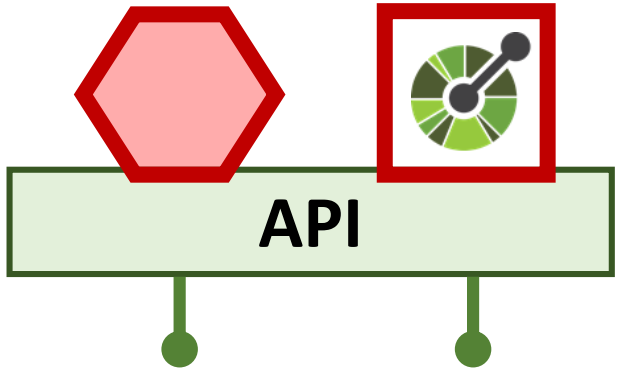
OpenAPI Initiative

- Herramientas
  - Editor http://editor.swagger.io
  - Explorador de APIs http://petstore.swagger.io
  - Validador https://online.swagger.io/validator
  - Generadores opensource para
    - *skeletons* para backends
    - *proxies* para clientes o front-end
    - http://swagger.io/swagger-codegen

Casos de uso

1. API Legada
2. Contrato primero
3. Dirigida por el servicio

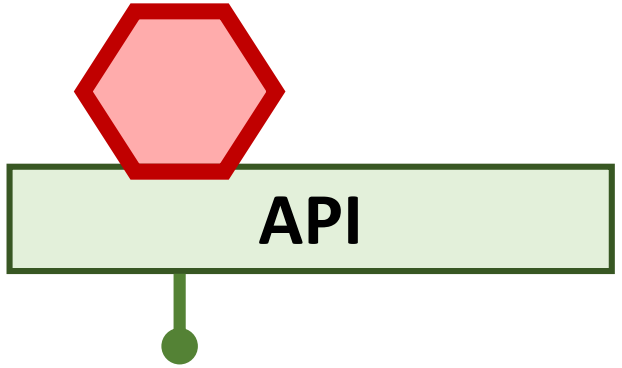
I. API Legada



- Documentar un API existente
- Construcción del contrato http://editor.swagger.io
- Validación

- Resultados:
  - API documentada
  - Generación de SDKs para cliente

I. API Legada. Ejemplo



¿Es **Nieves** un nombre de hombre o de mujer?

Servicio web para descubrirlo

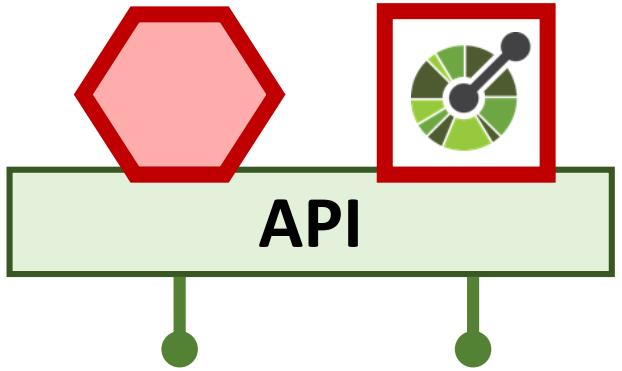
http://www.jerolba.com/mujeres-y-hombres-y-serverless

GET https://us-central1-hombre-o-mujer.cloudfunctions.net/gender?name=nieves

Créditos: Jerónimo López @jerolba

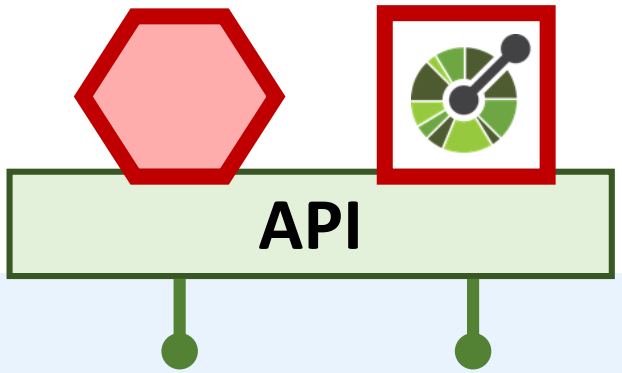
I. API Legada. Contrato

http://bit.ly/genero-openapi



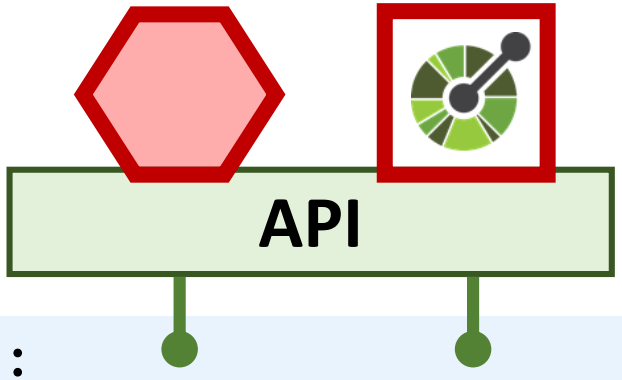

```
swagger: '2.0'
info:
  version: "1.0.0"
  title: Hombre o mujer.
host: us-central1-hombre-o-mujer.cloudfunctions.net
schemes:
  - https
consumes:
  - application/json
produces:
  - application/json
tags:
  - name: Gender
    description: API para no meter la pata con el género.
```

I. API Legada. Contrato



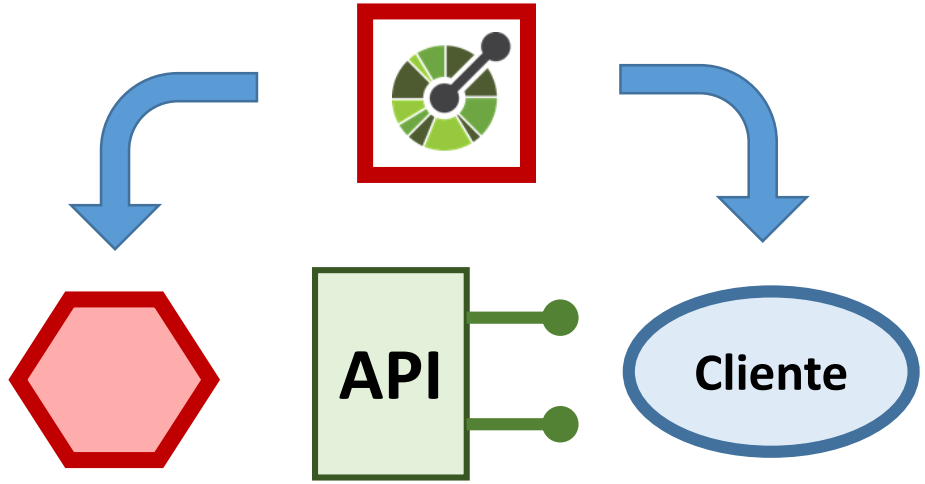

```yaml
paths:
  /gender:
    get:
      description: |
        Devuelve la probabilidad de que el nombre indicado como parámetro sea de mujer u
hombre.
      parameters:
        - name: name
          in: query
          description: Nombre de la persona
          required: true
          type: string
      responses:
        # Response code
        200:
          description: Respuesta con exito
          schema:
            $ref: "#/definitions/GenderResponse"
        404:
          description: No encontrado
          schema:
            $ref: "#/definitions/GenderNotFoundResponse"
```

I. API Legada. Contrato




```yaml
definitions:
  GenderResponse:
    required:
      - name
      - gender
      - probability
      - totalMale
      - totalFemale
    properties:
      name:
        type: string
      gender:
        type: string
      probability:
        type: number
        format: float
      totalMale:
        type: number
        format: int32
      totalFemale:
        type: number
        format: int32
  GenderNotFoundResponse:
    required:
      - name
      - gender
    properties:
      name:
        type: string
      gender:
        type: string
```

2. Contrato Primero



- La especificación se escribe en primer lugar

  http://editor.swagger.io

- Puede generarse:
  - un *skeleton* para el backend
  - Un *proxy* o SDK para el cliente
- Permite paralelizar el trabajo en backend y frontend.
- Los cambios al contrato pueden versionarse adecuadamente.

2. Contrato Primero. Servidores disponibles

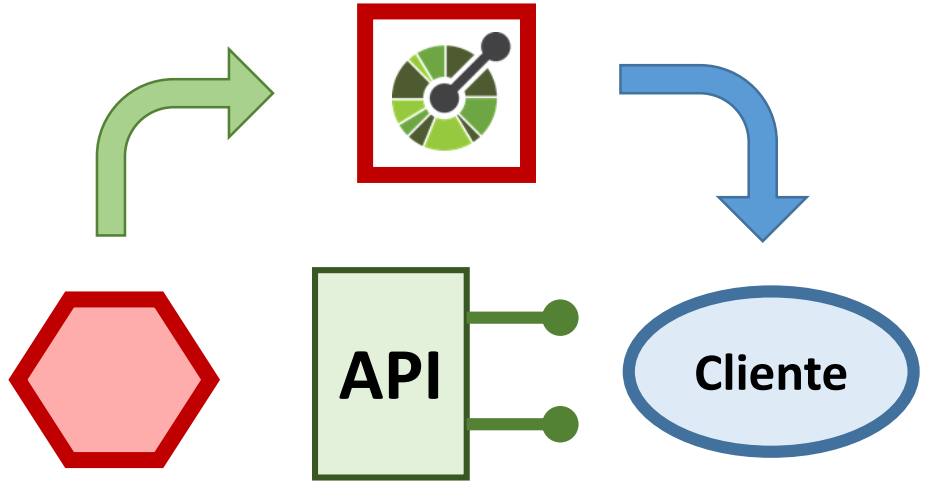
Generate Server ▾			
aspnet5	aspnetcore	erlang-server	finch
go-server	haskell	inflector	jaxrs
jaxrs-cxf	jaxrs-cxf-cdi	jaxrs-resteasy	jaxrs-resteasy-eap
jaxrs-spec	lumen	msf4j	nancyfx
nodejs-server	python-flask	rails5	scalatra
silex-PHP	sinatra	slim	spring
undertow		ze-ph	

2. Contrato Primero. Clientes disponibles

Generate Client ▾

akka-scala	android	async-scala	bash
clojure	cpprest	csharp	CsharpDotNet2
cwiki	dart	dynamic-html	elixir
flash	go	groovy	html
html2	java	javascript	javascript-closure-angular
jaxrs-cxf-client	jmeter	objc	perl
php	python	qt5cpp	ruby
scala	swagger	swagger-yaml	swift
swift3	tizen	typescript-angular	typescript-angular2
typescript-fetch		typescript-node	

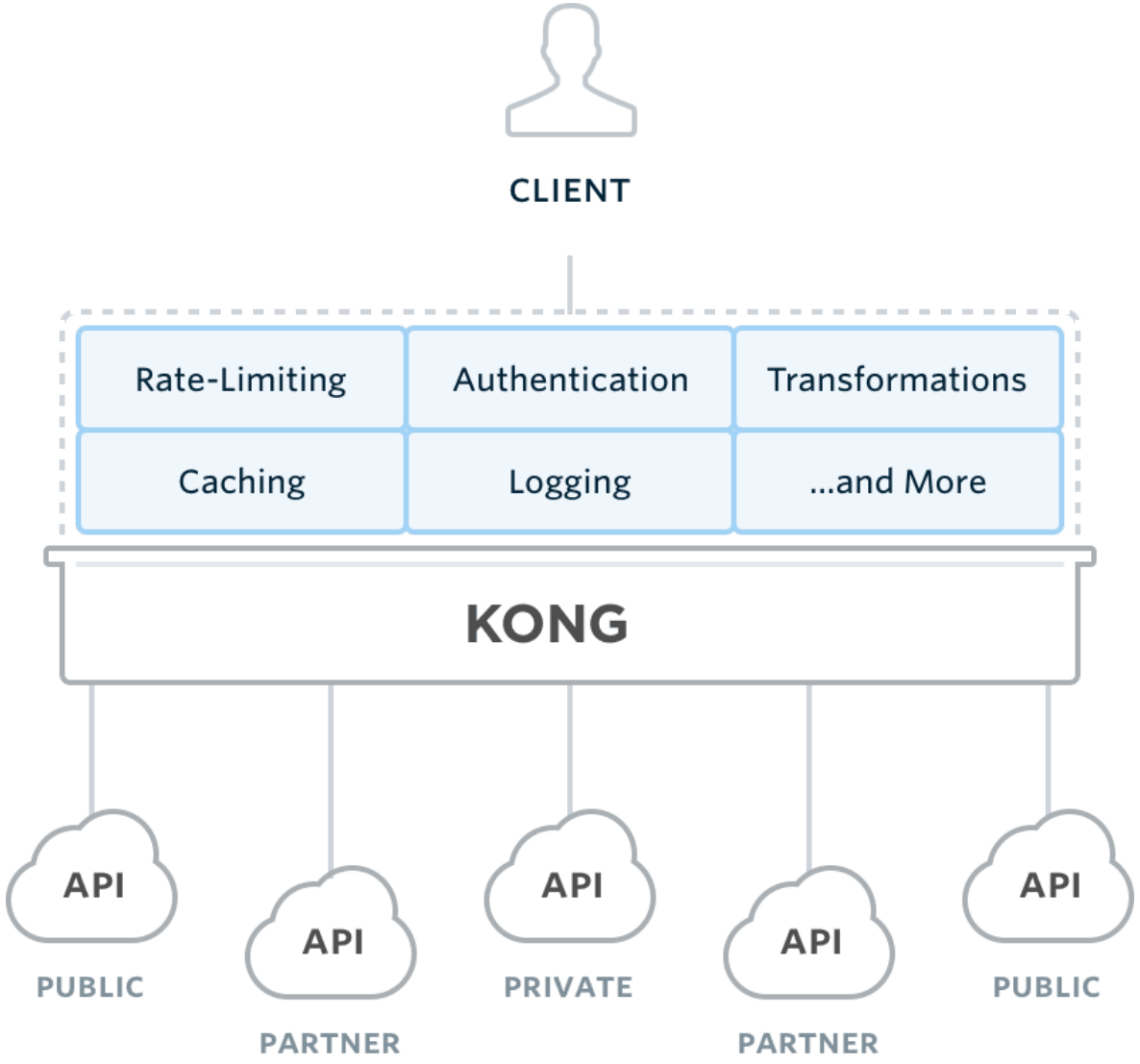
3. Dirigida por el servicio



- El servicio define el contrato
- La especificación del API en formato OpenAPI se genera por una librería que hace reflexión sobre el servicio.
- Requiere cuidado para no romper la compatibilidad del API.

- Ejemplo: https://openapi3.herokuapp.com
- Fuente: https://github.com/pjmolina/event-backend

API Management Tools



API Management Tools

- Aportar una capa que se coloca por delante del API
- Gestionada por 3ºs
- Aporta:
  - Autenticación, Autorización
  - Seguridad basada en roles
  - Protección frente a ataques DOS
  - Monetización: cobro por
  - Escalado
  - Auditoría
  - Métricas de uso, analíticas

Ejemplos

- 3scale
- Apigee
- Mashape Kong
- CA 7Layers
- Azure API Management
- IBM Bluemix API Management
- WSO

Versionado de APIs

- Versionado en la URL

```
GET /v1/restaurants?location=SVQ
```

```
GET /v2/restaurants?location=SVQ&limit=30
```

- Versionado en Parámetros

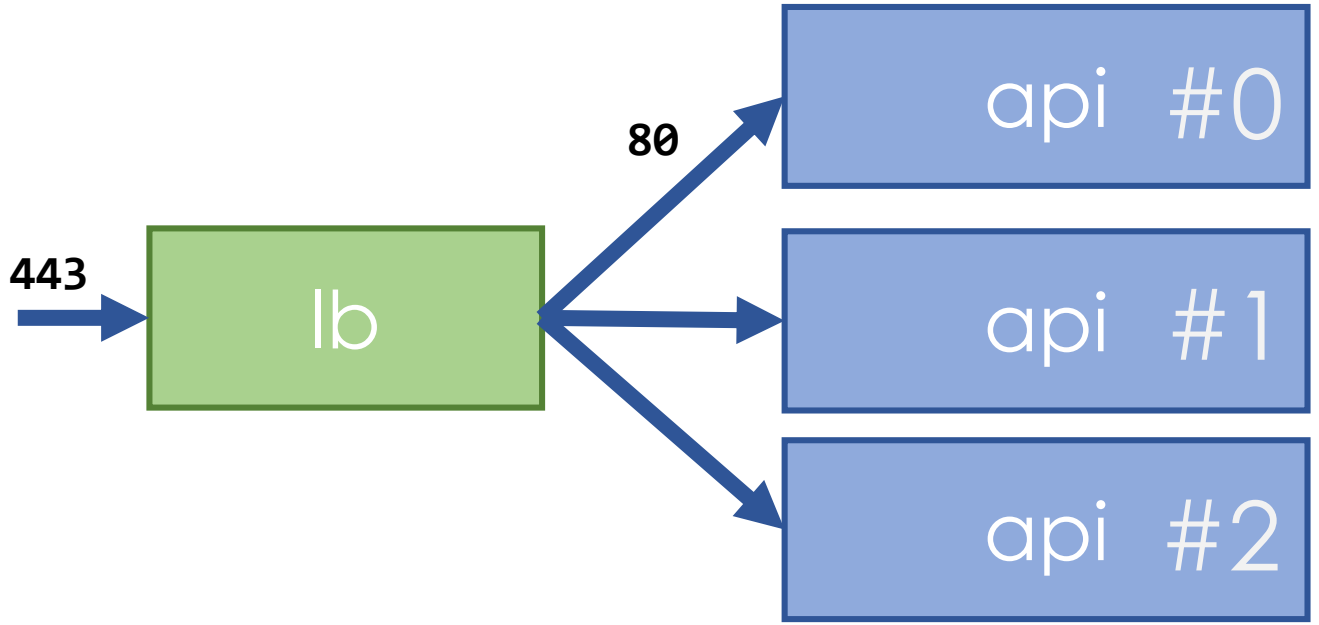
```
GET /restaurants?location=SVQ&limit=30&v=2
```

- Versionado en cabecera

```
GET /restaurants?location=SVQ&limit=30
Version: 2
```

Escalabilidad en APIs

- API **sin estado**
- Con un balanceador de carga en el frontal (como p.e. nginx o ha-proxy)
- Que distribuye el trafico a N (con N>=2) servidores
- DNS, SSL y certificados se configuran solo en el balanceador



Conclusiones

- OpenAPI es un estándar de facto para gestión de APIs
- Simplifica el consumo y la integración de APIs
- Futuro:
  - Versión 3.0 en Junio/Julio de 2017
  - Convergencia con el estándar **gRPC** de Google en curso

¡Gracias!

@pmolinam

Anexos

REST

- **RE**presentational **S**tate **T**ransfer
- Protocolo **sin estado**
- URIs únicas para cada recurso
- Semántica asociada a operaciones
  - **GET/PUT/POST/DELETE** sobre HTTP
- Hipermedia (navegable)

```
GET /actors/42
Accept: application/json
```

```
200 OK
Content-Type: application/json

{ "id": 42
  "name": "Jessica"
  "lastname": "Alba"
  "filmography": "/films/42"
}
```

Tipos MIME

- Declaran el tipo de codificación a emplear
- Los más frecuentes:
  - JSON `application/json`
  - XML `text/xml`
  - HTML `text/html`
  - Texto plano `text/plain`
  - CSV `text/csv`

```
GET /actors/42
Accept: text/xml
```

```
200 OK
Content-Type: text/xml

<actor id="42">
    <name>Jessica</name>
    <lastname>Alba</lastname>
    <filmography
        url= "/films/42" />
</actor>
```

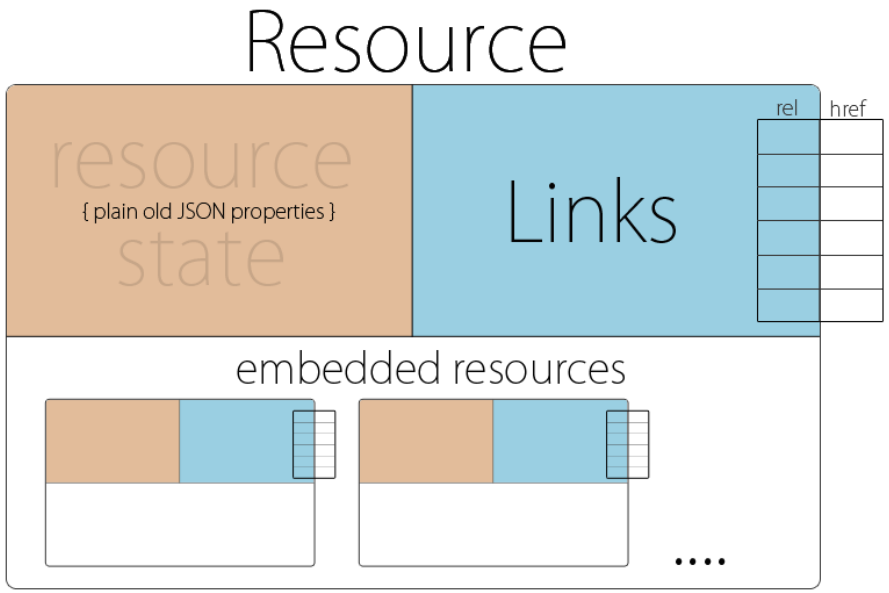
Niveles REST

Richarson Maturity Model
http://martinfowler.com/articles/richardsonMaturityModel.html

- **Nivel 0.** HTTP y nada mas (RPC bajo HTTP)
- **Nivel 1.** Recursos: `GET /factura/217`
- **Nivel 2.** Verbos y códigos de error HTTP `POST /factura/ → 201 Created`
  https://i.stack.imgur.com/whhD1.png
  https://httpstatuses.com
- **Nivel 3.** Controles Hipermedia

```
<link rel="lineas"
      uri="/factura/217/lineas"
/>
```

HAL



- Estándar en Hipermedia para Recursos

http://stateless.co/hal_specification.html

```
{
  “id”: 1234
  “name”: “Alice in Wonderland”
  “_links”: {
      “self”: { “href”: “/book/10”},
      “prev”: { “href”: “/book/9”},
      “next”: { “href”: “/book/11”},
      “action-delete”: {
          “verb”: “DELETE”,
          “href”: “/book/10”
      }
  }
}
```