

Introducción a las APIs

Carlos Alberto Medina González

Septiembre 2021

Contenido del Curso

01

Introducción y
Conceptos Básicos

02

¿Qué son las
Web-APIs?

03

API REST

04

APIs
Documentacion

05

APIs
Seguridad, Performance y
Monitorizacion

06

APIs
Arquitectura Técnica de
Componentes

07

APIs
Ciclo de Vida

08

Conclusiones

APIs

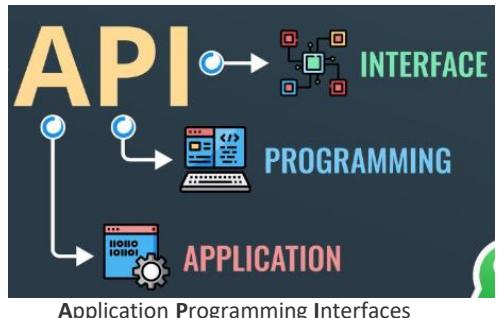
Conceptos Básicos

- ¿Qué es un API?
- La relevancia de las APIs en el Nuevo Ecosistema
- ¿Qué se necesita para que una API tenga éxito?
- Tipos de API



CONCEPTOS BÁSICOS

¿Qué es un API?



¿Qué es un API?

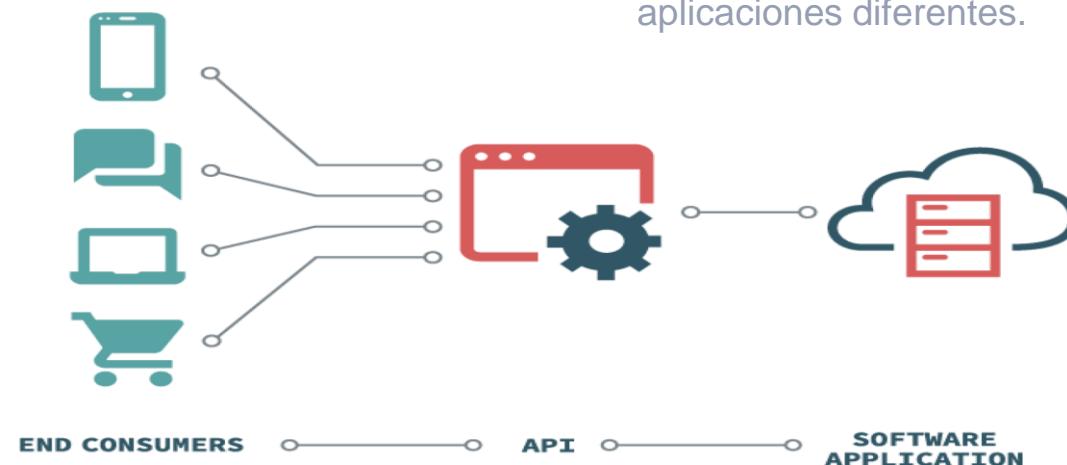
Una API facilita la relación entre dos aplicaciones para el intercambio de mensajes o datos mediante un contrato prefijado.

Definición 2

Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, **permitiendo la comunicación entre dos aplicaciones** de software a través de un conjunto de reglas.

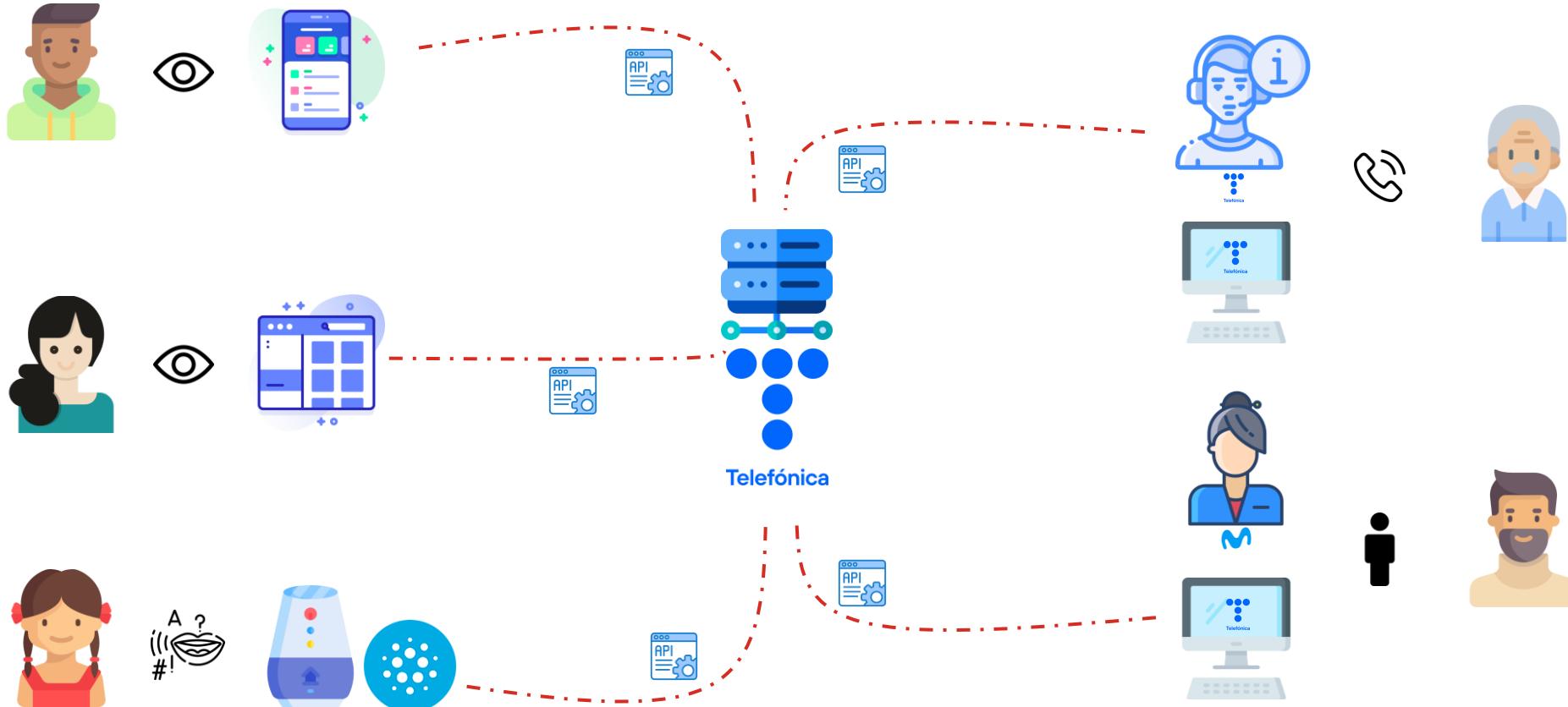
Definición 3

En términos genéricos establece **un contrato entre dos partes** en el que **una se compromete a proveer a la otra de un servicio determinado, siguiendo unas reglas de comunicación concretas**. Esas dos partes pueden pertenecer a la misma aplicación o tratarse de dos aplicaciones diferentes.



CONCEPTOS BÁSICOS

¿Para que sirve un API?



Relevancia de las APIs

Pero las APIs ya existían,
¿por qué se han vuelto ahora tan relevantes?

El Entorno

- Nuestro entorno demanda cada vez **más cambios y más rápidos**.
- Es necesario dar una **experiencia completa** a nuestros clientes, lo que nos hace requerir muchos **servicios de terceros**.
- Todo está **conectado y todo en tiempo real**



La Necesidad

Es necesario disponer de un modelo **ágil** y **eficiente** para poder **integrar** todos los elementos del nuevo ecosistema digital.

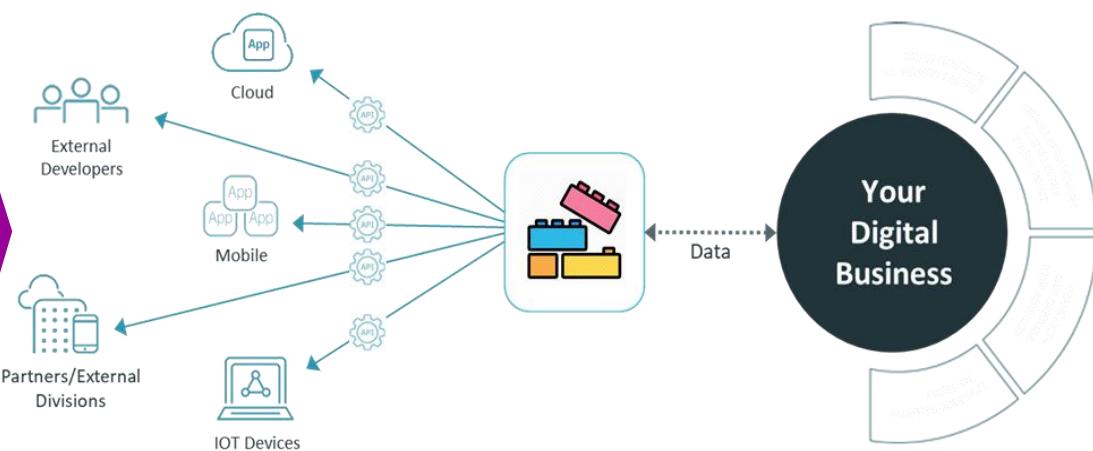


En un mundo donde todos interactúan con todos y en la mayoría de los casos en modo automático, dos cosas son esenciales: **flexibilidad y estandarización**.

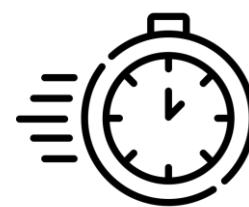
Relevancia de las APIs

Las APIs se han convertido en los Building Blocks del nuevo ecosistema digital.

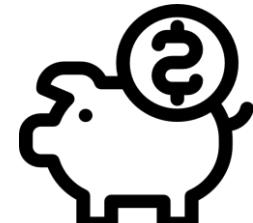
Habilitador



Las APIs permiten un increíble ahorro de tiempo y de recursos en el sector industrial, ya que permite reutilizar, funciones y software ya existentes en nuevas plataformas, sin necesidad de volver a diseñarlas ni crearlas, solo usarlas



Velocidad



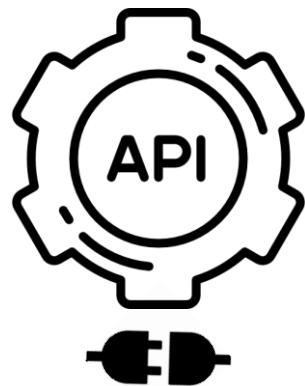
Ahorro



Fiabilidad

APIs: El Habilitador

Habilitador

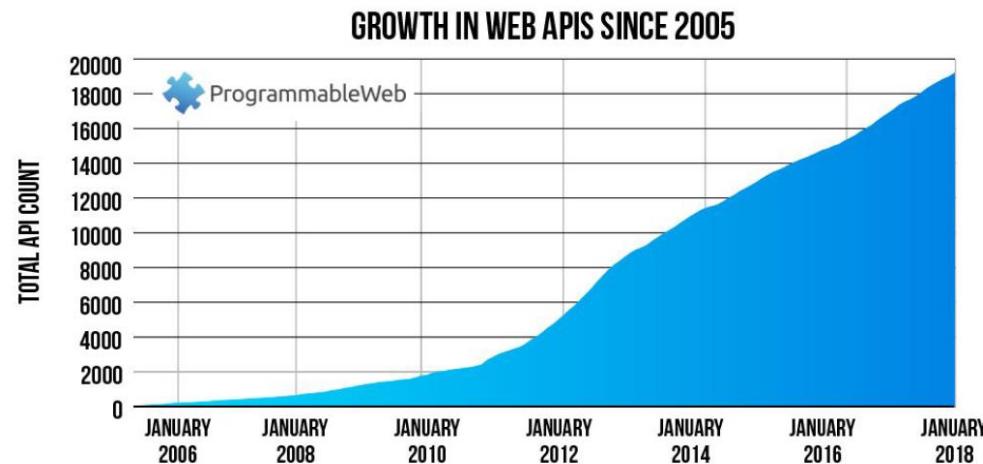


- 1. Automatización:** las APIs potencian la automatización de las actividades al compartir información entre sistemas se puede evitar la intervención manual por parte de personas de diferentes organizaciones.
- 2. Aplicación:** al habilitar el acceso de las APIs a los componentes de las *apps* se proporciona flexibilidad en la entrega de servicios e información, propia y de terceros
- 3. Más alcance:** con una API se puede crear una capa de aplicación que se puede utilizar para la distribución de información y servicios a nuevas audiencias que se pueden personalizar para crear experiencias de usuario a la carta.
- 4. Democratización de la información:** las APIs facilitan la compartición de la información y hacerla mas accesible a colectivos que no tenían anteriormente esa facilidad de acceso.
- 5. Eficiencia:** cuando se proporciona acceso a una API, el contenido que se genera se puede publicar automáticamente y está disponible para todos los canales. Permite que se comparta y se distribuya más fácilmente.
- 6. Integración:** las APIs permiten que los contenidos se puedan embeber desde cualquier *site* o aplicación con mayor facilidad. Lo que garantiza una entrega de información fluida y una experiencia de usuario integrada.
- 7. Personalización:** a través las APIs cualquier usuario o empresa pueden personalizar los contenidos y servicios que más utilizan.
- 8. Adaptación:** las necesidades cambian con el tiempo y las APIs ayudan a anticiparse a los cambios. Cuando se trabaja sobre esta tecnología se puede soportar mejor la migración de datos, a la vez que se ofrece una mejor revisión de la información. En definitiva, las APIs proporcionan una mayor flexibilizar en la prestación de servicios.

CONCEPTOS BÁSICOS

Evolución del Uso de APIs

A medida que el paradigma de conectividad se expande y nos adentramos en un entorno dominado por los datos, **las APIs se convierten en una piedra angular para cualquier negocio**. Desde las redes sociales y medios de comunicación a múltiples sectores como la banca o el retail, las compañías ven facilitada su gestión a través de estas herramientas que impulsan la transformación digital mediante la integración de servicios.



El Mercado de las APIs registra un crecimiento anual del **32,9%**

Se estima que pasará de mover 1.200 millones de dólares en 2018 a **5.100 millones en 2023**.

McKinsey



El 68% de las compañías industriales españolas están en un estadio de digitalización **medio o bajo**

El 32% en un nivel avanzado, solo un **5% pueden considerarse** como **digital champions**.

PwC

Claves del Éxito

¿Qué se necesita para que una API tenga éxito?

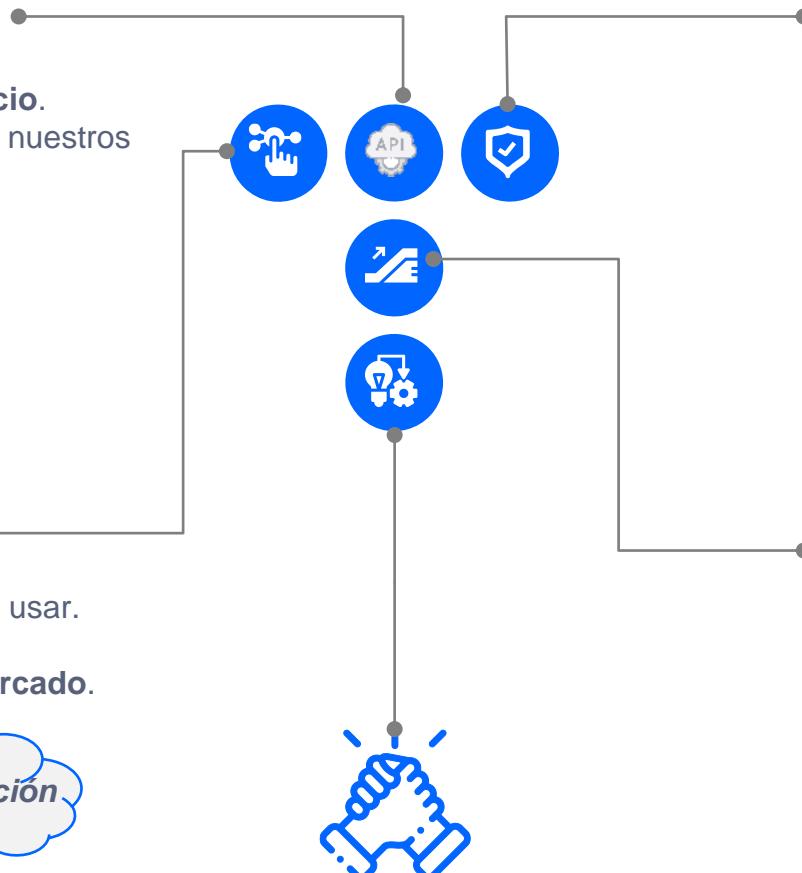
Experiencia Cliente

- Respuesta a las necesidades de **Negocio**.
- Potenciar la **agilidad** en la relación con nuestros clientes.



Implementación

- **Sencillas** de implementar y **fáciles** de usar.
- **Bien documentadas**.
- Desarrolladas bajo **estándares de mercado**.



Fiabilidad

- Criterios de **tokenización** y **autenticación** Auth 2.0.
- **Trazabilidad end-to-end** de todas las capas de Arquitectura para garantizar el *performance*.
- **Estabilidad** de los sistemas de back-end.
- Implementación y normalización de **códigos de error**.



Escalabilidad

- Definición de **roadmap** de evolución de mas a menos.
- Aplicación de criterios de versionado intuitivos y cronológicos.



CONCEPTOS BÁSICOS

Claves del Éxito

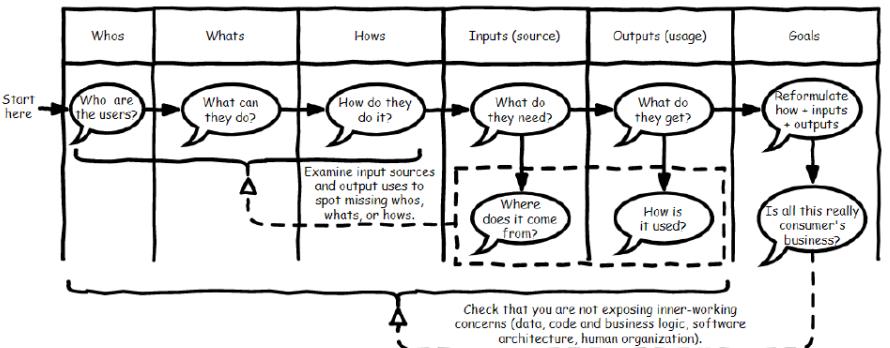
Experiencia Cliente



Dar respuesta a las necesidades de Negocio



Es hacerse una serie de preguntas básicas que se resumen en la API goals canvas.



Whos	Whats	Hows	Inputs (source)	Outputs (usage)	Goals
Admin users	Manage catalog	Add product	Catalog (API), Product info (user)	Added product (get, update, delete, replace)	Add product to catalog
		Get product's information	Product (search, add)	Product info (user)	Get product
		Update product's information	Product (get, search, add), Updated info (user)		Update product
		Replace product	Product (get, search, add), New product info (user)		Replace product
		Delete product	Product (get, search, add)		Delete product
		Search for products	Catalog (API), free query (user)	Products matching query (get, update, delete, replace)	Search for products in catalog using a free query

Fiabilidad



Seguridad



Estrategias de Prevención y Mitigación



Evitar Ataques de denegación de Servicio

Modificación de datos de entrada

Ataques de intermediario

Suplantación de Identidad



Estabilidad



Estrategias de Prevención



Adopción de nuevas arquitecturas técnicas que nos permitan disponibilidad 24x7 (Cloud, Microservices, etc...)



Estrategias de Prevención



Estrategias de Monitorización Proactiva

Desarrollo de nuevas arquitecturas escalables (Ej. FastData)

CONCEPTOS BÁSICOS

Claves del Exito

Implementación

Estrictamente la API es **solamente la interfaz** que expone un determinado software



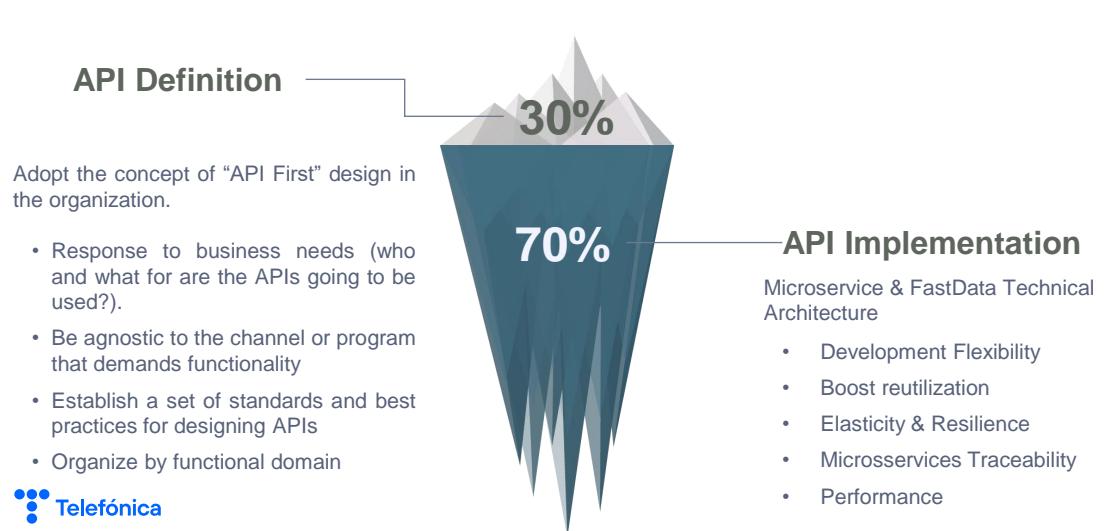
Pero no se puede tener una buena API sin una buena implementación ambas partes son necesarias

API = Interfaz + Implementación

Escalabilidad

Creación agonísticas al canal que va a requerir la información, las APIs deben cubrir una demanda funcional de proceso.

Aplicación de una clara política de versionado y evolución de las APIs



**Las APIs deben ser
Útiles, Sencillas, Fiables
y**

API = Interfaz + Implementación

CONCEPTOS BÁSICOS

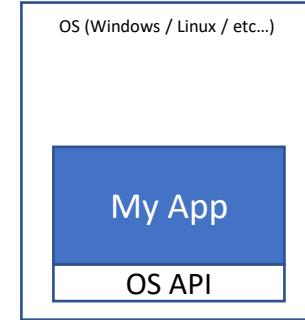
Tipos de APIs

Operating System API

Los programas de software están continuamente interactuando con los sistemas operativos.

La forma en la que lo hacen es a través de APIs. Sistemas operativos como Windows disponen de APIs que permiten esa comunicación entre programas y el OS. (ej. Win32 API);

BIOS interrupt calls: permiten acceder a las funcionalidades más básicas de una máquina de arquitectura x86

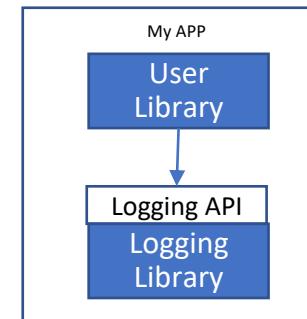


- File System
- Network Device
- User Interface Elements
- etc...

APIs de Librerías y basadas en Clases API

Este tipo de APIs son las que permiten que una aplicación importe una biblioteca de otro software para hacer el intercambio de información.

Este tipo de interfaces de desarrollo de aplicaciones permite la conexión con los datos en torno a las clases, como es habitual en programación orientada a objetos (POO) con java



- Class Library
- .Net
 - Java
 - Python
 - node.js

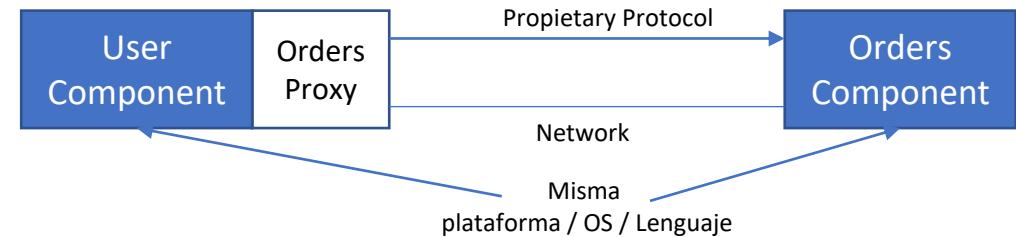
CONCEPTOS BÁSICOS

Tipos de APIs

Remote API

Este tipo de APIs requieren un recurso que está ejecutándose en otro proceso y posiblemente en otro PC para ello necesita comunicarse a través de la RED y creando un protocolo.

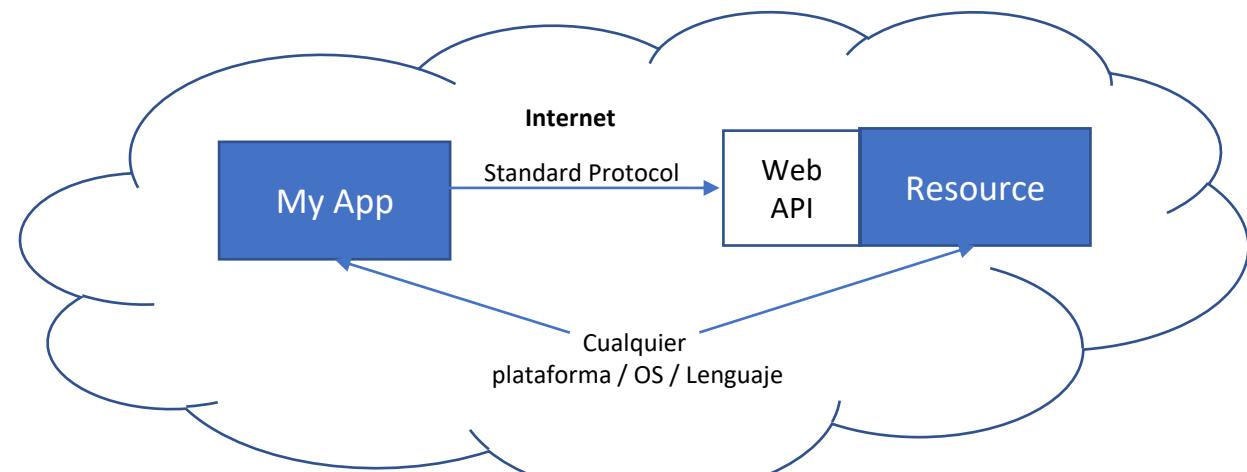
Pero tiene una limitante importante que **ambos programas debe estar bajo la misma plataforma, ósea deben estar los dos desarrollados en el mismo lenguaje y sistema operativo** (ej. Java RMI, .NET Remoting, DCOM)



Web API

Este tipo de APIs exponen recursos a través de Internet, por lo que el canal de comunicación será WEB y cualquier aplicación puede acceder a obtener esos recursos siempre que tenga los permisos y derechos.

Es muy flexible ya que a través del protocolo http se independiza de las plataformas, sistemas operativos y lenguajes de programación entre la APP y el proveedor de recursos



Web - APIs

- ¿Qué es un Web-API?
- Tecnologías del Web API

02

¿Que es un Web APIs?

Las APIs web son una tipología de APIs, en las que la **comunicación se lleva a cabo a través de Internet**, haciendo uso de **protocolos web como puede ser HTTP**.

En este “contrato” del que hablábamos al principio de las APIs, **una API web define una serie de endpoints o URLs de acceso**, la forma de las peticiones y la estructura de las respuestas.

Características comunes del Web-API

- Agnóstico a las plataformas y programas de programación
- Uso de protocolos estándar -> (http el más frecuente)
- Usan el modelo de Request/Response

Características que determinan los tipos de Web-API

- | | |
|-------------------------|--------------------------|
| • Formato del Request | • Formato del Response |
| • Contenido del Request | • Contenido del Response |

Evolución de las WebAPIs

01**RPC- XML****(1998) - Dave Winer**

- XML-RPC es una implementación de RPC que usa XML como formato de datos y HTTP como protocolo de comunicación
- XML-RPC es una implementación muy ligera de RPC. Es muy sencillo de desarrollar y depurar.
- Limitaciones para usar ciertos tipos de datos complejos
- Process Driven

02**SOAP****(1998) - Microsoft**

- Es un protocolo define el intercambio de información estructurada (Protocolo de Acceso a Objetos Simples)
- Ser Considera el sucesor de RPC-XML y SOAP introduce el uso de WSDL o Web Services Description
- Requiere un mayor ancho de banda y recursos que resto de alternativas (exceso de verbosidad)
- Process Driven, requiere un mayor acoplamiento con el servidor, lo que complica el desarrollo de nuevas funcionalidades

03**REST****(2000) - Roy Fielding**

- Representational State Transfer (No usa protocolo es un estilo de arquitectura)
- REST tiene mejor rendimiento y escalabilidad que SOAP
- Son el tipo de API mas extendido dentro de las Web-API
- Driven by Data (Es el recurso el que vertebral las APIs REST)

04**GraphQL****(2012) - Facebook**

- Trabaja de una forma similar a las REST
- Muy flexible y se obtiene solo los datos que se requieren (eficiente en el numero de llamadas)
- Driven by Data
- Requiere especial atención optimización del Performance
- Permite gestión de Cache pero no de forma nativa

05**gRPC****(2015) - Google**

- Basada en el protocolo http/2
- Soporta mensajería en streaming y bi-direccional
- Mejor performance
- Requiere librerías específicas en los dos lados de la comunicación

Web APIs: Tipos de WebAPIs (Comparativa)

	XML-RPC	SOAP	REST	GraphQL	gRPC
Tipo	Protocolo	Protocolo	Arquitectura	Lenguaje de consultas	Protocolo (buffers)
Formato de datos	XML	XML	XML, JSON, HTML, Texto Plano	JSON	<u>Protobuf (pequeño, binario)</u>
Protocolo de comunicación	HTTP	TCP, HTTP STMP	HTTP	HTTP	HTTP/2
Tipado de datos	Débil	Fuerte	Débil	Fuerte	Débil
Definición de datos	Servidor	Servidor	Servidor	Cliente	
Streaming	Cliente - Servidor	Cliente - Servidor	Cliente - Servidor	Cliente - Servidor	Cliente – Servidor (bidireccional)
Endpoints	Único	Único	Múltiples	Único	
Caché	No	No	Si	No (de forma nativa)	Si
Versionado	Si	Si	Si	No	Si

API REST

- ¿Qué es un API Rest?
- Conceptos Básicos del API REST
- REST vs RESTful

03

APIs REST

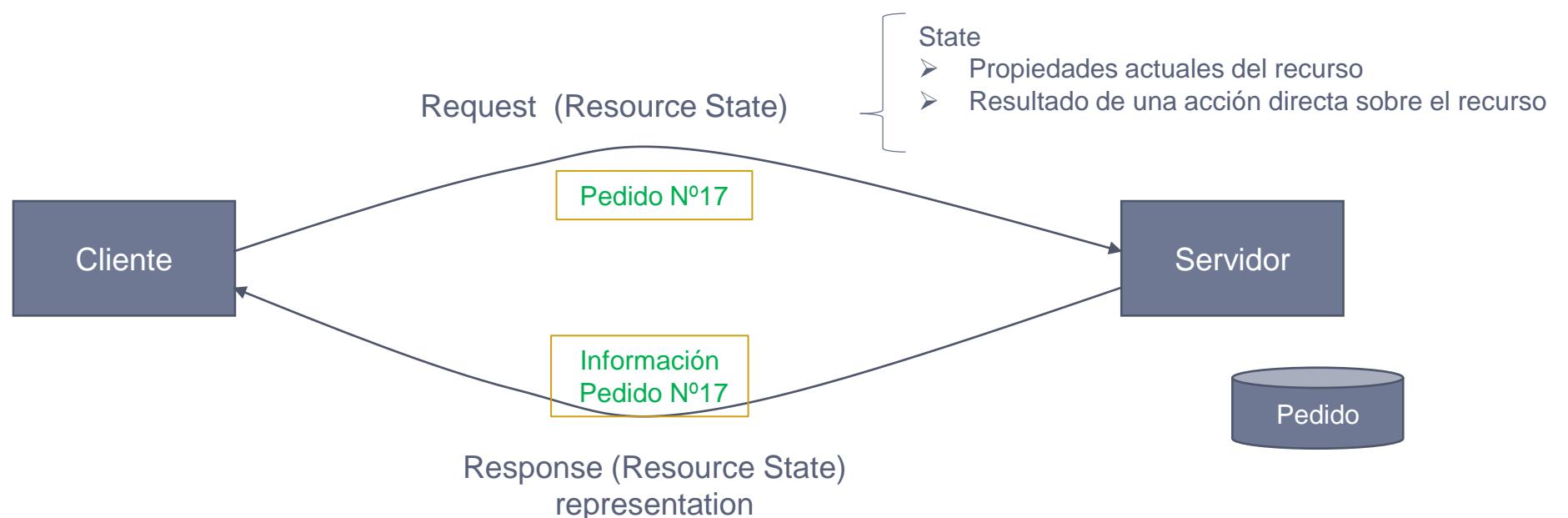


REST deriva de “REpresentational State Transfer”, que traducido significaría “Transferencia de representación de estado”.

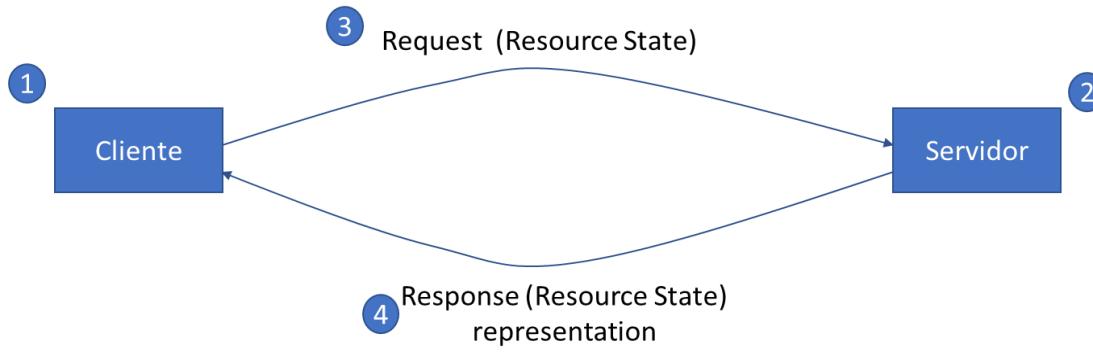


La API REST su objetivo es realizar una acción/petición sobre un recurso, y siempre quiere saber cual es el resultado de la acción o la situación del recurso

Mejor con un ejemplo



Estructura de las APIs REST



Tanto el Request como el Response se codifican bajo el protocolo http

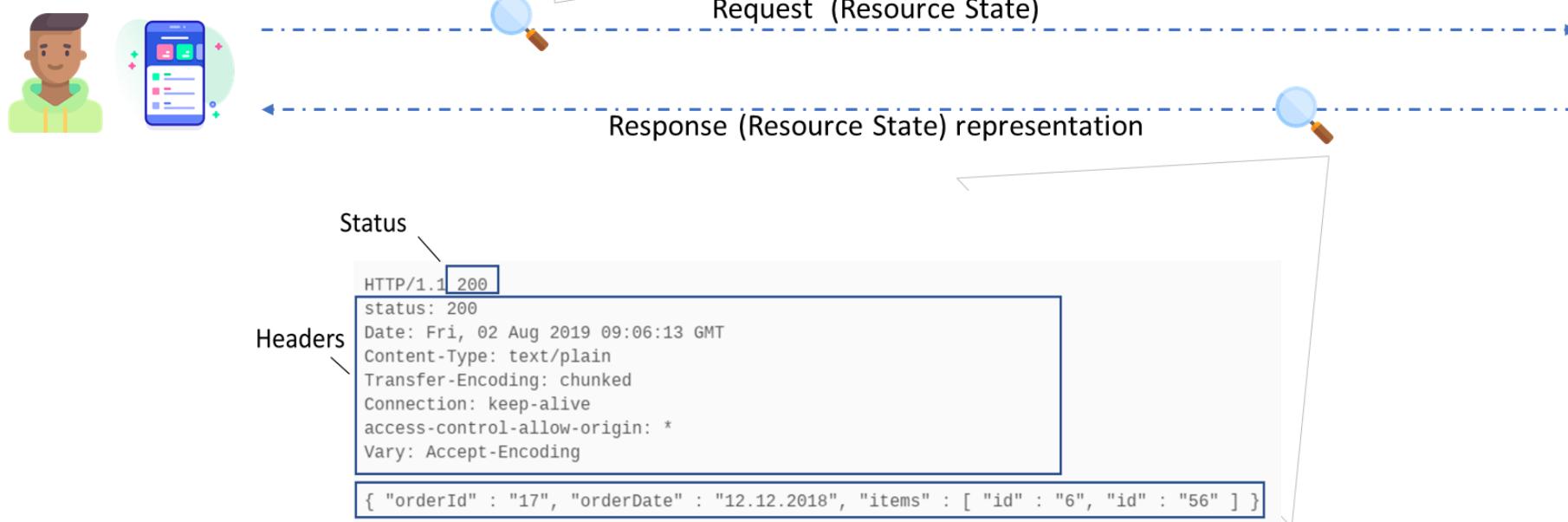
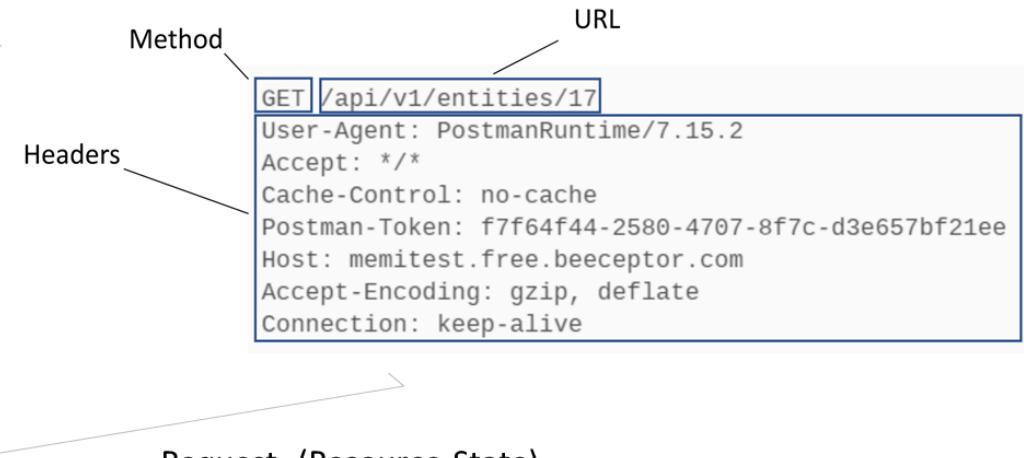
3 Request

- | | | |
|----------|---|---|
| Método | → | HTTP Verb: Determina la acción que se quiere realizar (GET, POST, PUT, DELETE, ...) |
| Endpoint | → | Localización del recurso sobre el que aplicar la acción + parámetros (parametros no es obligatorio) |
| Headers | → | Zona de Metadatos se pasa el token, el usuario, etc...) |
| Body | → | Contenido del Request (opcional) |

4 Response

- | | | |
|-------------|---|---|
| Status Code | → | Es el código que representa el resultado (éxito o fracaso) y esta codificado para indicar tipo de fracaso |
| Headers | → | Zona de Metadatos del response (Content Type ...) |
| Body | → | Contenido de la respuesta (opcional) |

Estructura de las APIs REST (Ejemplo)



Body: En la mayoría de los casos el Body se expone a través de un Fichero JSON, pero REST soporta otros tipos de formato como XML, Texto Plano.



Telefónica

Detalle de la Estructura del API (Request)

1

HTTP Verbs

Determina la acción que se quiere realizar el cliente en un determinado recurso, los verbos permiten identificar de forma muy rápida la acción que quiere realizar la API



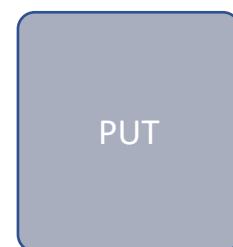
- Se usa para recuperar recursos (información)
- NUNCA se use para agregar / actualizar / eliminar recursos
- Es el verbo predeterminado de la barra de direcciones del explorador
- Por lo general, se combina con parámetros
- No debe incluir cuerpo

Excepciones

A veces GET no se puede utilizar para la recuperación, para los casos en los que se requiere muchos parámetros, en esos casos se usa el método POST



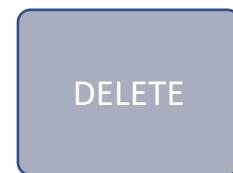
- Se usa para añadir recursos
- Debe contener body donde se especifique el recurso que se va a añadir
- No debe contener parámetros de cadena de consulta



- Se usa para modificar recursos que ya existen
- Debe contener body donde se especifique el recurso que se va a modificar
- No debe contener parámetros de cadena de consulta
- El método PUT es Idempotente

Excepciones

Si el elemento que se quiere modificar no existe el método PUT puede crearlo



- Se usa para eliminar recursos
- NUNCA se use para agregar / actualizar / recuperar recursos
- Casi siempre combinado con parámetros

Detalle de la Estructura del API

Resumen Verbos CRUD (Create, Read, Update, Delete)

Verb	Role	Body?	Params In...
GET	Retrieve resource(s)	No	URL
POST	Add resource(s)	Yes	Body
PUT	Modify resource(s)	Yes	Body
DELETE	Delete resource(s)	No	URL

Otros verbos menos usados

PATCH

- Similar a PUT pero para modificaciones parciales

HEAD

- Lo mismo que GET pero sin la estructura BODY en el respond

OPTIONS

- Determina los verbos que están disponibles para una determinada URL

Detalle de la Estructura del API (Request)

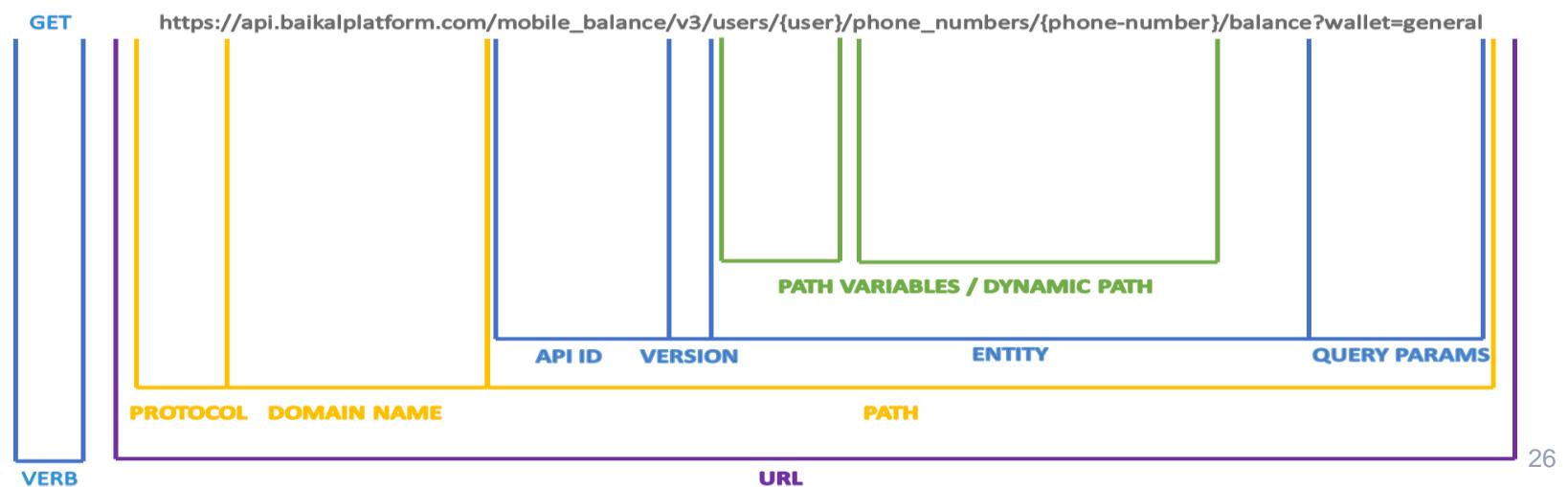
2

Endpoint (URL)

- Es la parte más importante de la API
- **El endpoint contiene la (URI) que indica dónde y cómo encontrar el recurso en Internet,**
- **Actúa como una dirección web completa**, donde se determina qué datos debe contener, en qué orden y cuál es la importancia
- Para la estructura de la URL se suele definir una template para que todas las APIs sean consistentes y tengan un comportamiento igual y facilite en entendimiento y trabajo a los desarrolladores u usuarios de la API.
- La URL debe ser autoexplicativa, ya que al leerla debe poder identificarse el propósito de la misma, aunque este documentada, el objetivo de las APIS es tener una URL lo mas explicativa posible.

Conceptos que conforman la URL

- Protocol
- Domain Name
- API ID
- Version
- Entity
- Dynamic Path Variables
- ID Parameter
- Query Parameters



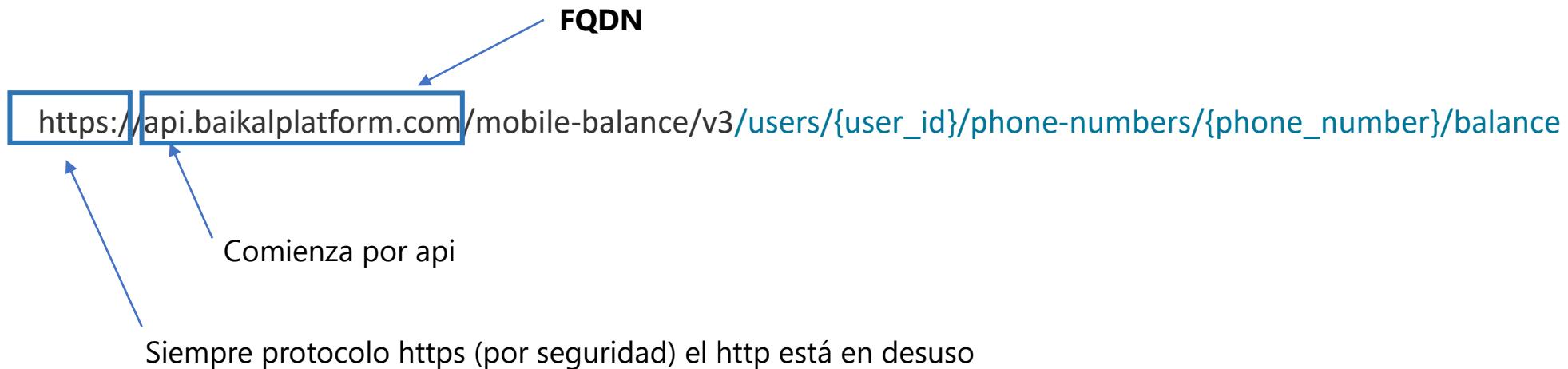
Detalle de la Estructura del API (Request)

2

Endpoint (URL)

Protocol &
Domain Name

- Determina la dirección del servidor donde se disponibiliza el recurso
- Es el primer componente de la URL
- Normalmente contiene FQDN (Fully Qualified Domain Name) o el nombre del servidor. (Cuando son del tipo FQDN suele tener el nombre de api)



Detalle de la Estructura del API (Request)

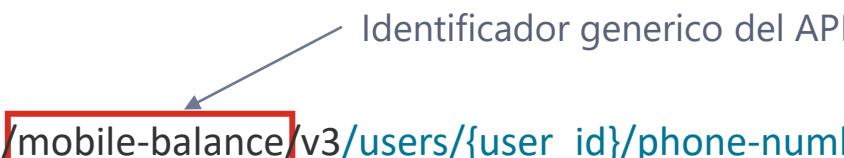
2

Endpoint (URL)

API ID

- Determina el API a la cual pertenece el método que se va a consumir

GET https://api.baikalplatform.com/mobile-balance/v3/users/{user_id}/phone-numbers/{phone_number}/balance



Identificador generico del API

Detalle de la Estructura del API (Request)

2

Endpoint (URL)

Versión

- Determina la versión que se quiere usar de la API
- Siempre es un numero natural (positive y entero) y suele llevar el prefijo 'V'
- Es el primer componente de la URL

https://api.baikalplatform.com/mobile-balance/v3/users/{user_id}/phone-numbers/{phone_number}/balance



Es la version 3 del API

Comienza por V

Detalle de la Estructura del API (Request)

2

Endpoint (URL)

Entity

- Determina la entidad sobre la que vamos a realizar la acción que determine el verbo
- Las SubEntity es consistente con la jerarquia de los recursos

`https://api.baikalplatform.com/mobile-balance/v3/users/{user_id}/phone-numbers/{phone_number}/balance`

Entidad de la que vamos a obtener la información
del balance de la línea del cliente.

Detalle de la Estructura del API (Request)

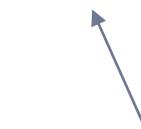
2

Endpoint (URL)

Dynamic Path Variables

- Son variables que nos describir un “Path” donde se acotar la información de una determinada entidad, la cual pertenece a una determinada jerarquía

`https://api.baikalplatform.com/mobile-balance/v3/users/{user_id}/phone-numbers/{phone_number}/balance`



Aplica una jerarquía a través del path de la URL
(1 Usuario; 2 Numero de Telefono del Cliente)

- `users/{user_id}/`
- `phone-numbers/{phone_number}/`

Se posiciona en un Usuario concreto

Se posiciona en un Numero de Telefono concreto del usuario
previamente posicionado

Detalle de la Estructura del API (Request)

2

Endpoint (URL)

ID Parameters

- Son parámetros requeridos para realizar la consulta sobre un determinado recurso (Se usan para el GET)

A partir de "?" Se informan los query parameter

`https://api.baikalplatform.com/mobile_balance/v3/users/{user}/phone_numbers/{phone-number}/balance?wallet=general`

Cada Query parámetro lleva su clave-valor y se concadenan los parámetros con el símbolo "&"

Detalle de la Estructura del API (Request)

3

Header

- El Header almacenar información relevante tanto para el cliente como para el servidor.
- Principalmente, los Headers proporcionan datos de autenticación, como una clave de API, el nombre o la dirección IP del equipo donde está instalado el servidor y la información sobre el formato de respuesta.

Headers

GET/api/V1/entities/17

```
User-Agent: PostmanRuntime/7.15.2
Accept: /*
Cache-Control: no-cache
Postman-Token: f7f64f44-2580-4707-8f7c-d3e657bf21ee
Host: memitest.free.beeceptor.com
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

Detalle de la Estructura del API (Request)

4

Body

- El Body se utiliza para transmitir información adicional al servidor.
- Por ejemplo, puede ser un fragmento de datos que desea agregar, reemplazar o solicitar.

```
{  
    "phone_number": "+34612345678",  
    "wallets": [  
        {  
            "id": "db5e8962-a190-44ac-a384-44d63db5b3dd",  
            "name": "Credito recargas",  
            "description": "Tu crédito de recargas bajo demanda",  
            "amount": {  
                "value": 26.88,  
                "currency": "EUR",  
                "tax_included": true  
            },  
            "expiration_date": "2017-01-18T00:00:00Z",  
            "type": "general"  
        }  
    ]  
}
```

Detalle de la Estructura del API (Response)

1

Response Codes

Notificar a los clientes sobre el resultado de la solicitud

- ¿Lo logró?
- ¿Fracasó? ¿Por qué?
- ¿Qué tipo de error?

¿Debido a la importancia de los códigos de error estos están estandarizados y organizados por grupos

- Son cinco grupos
- Cada grupo representa un tipo de respuesta específico (éxito, error, etc.)
- Cada grupo consta de un código de estado de 3 dígitos

¿El porqué de la importancia de los códigos de error?

- La mayoría de los clientes comprueban el código de respuesta y actúan en consecuencia
- Las herramientas de monitoreo comprueban los códigos de respuesta y lo informan
- Hace que la API sea más fácil de usar y entender
- Muy fácil de implementar, a menudo pasada por alto

Grupos de Errores

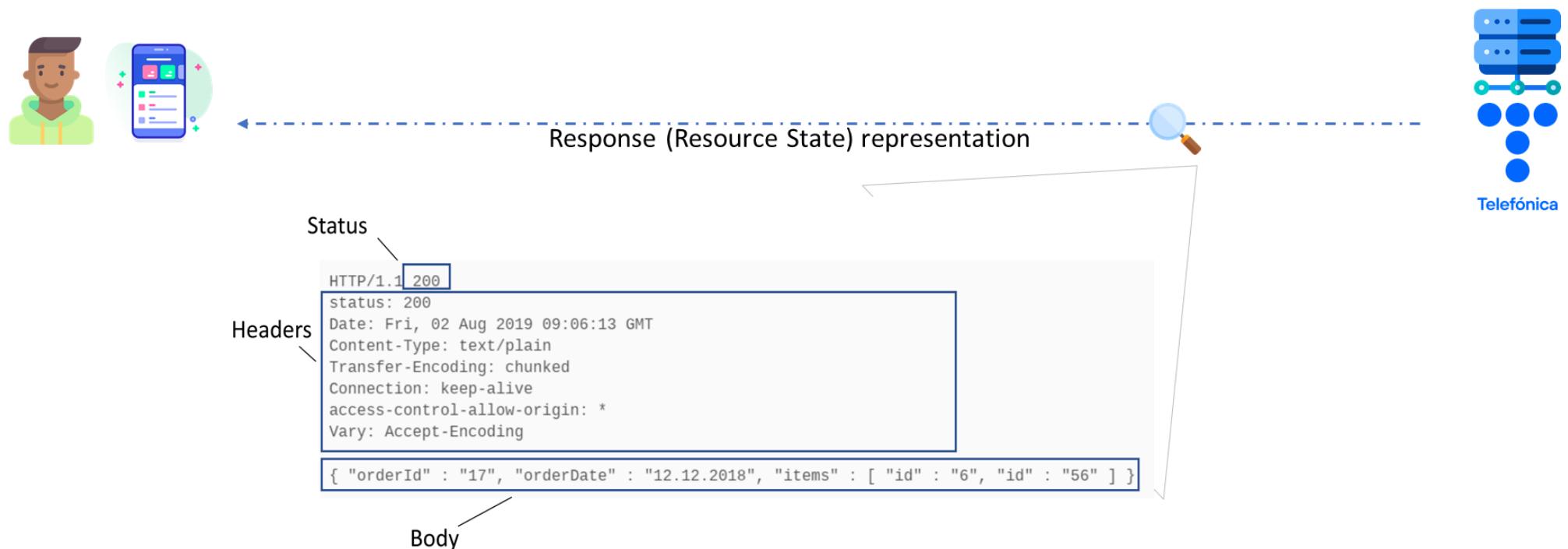
- 1xx – Respuesta informativa
- 2xx – Éxito
- 3xx – Redirección
- 4xx – Errores de cliente
- 5xx – Errores del servidor

Detalle de la Estructura del API (Response)

2

Headers

- El Header almacenar información relevante tanto para el cliente como para el servidor.
- Principalmente, los Headers proporcionan datos de autenticación, como una clave de API, el nombre o la dirección IP del equipo donde está instalado el servidor y la información sobre el formato de respuesta.



Detalle de la Estructura del API (Response)

3

Body

- El Body se utiliza recibir la información requerida al servidor

```
{  
  "phone_number": "+34612345678",  
  "wallets": [  
    {  
      "id": "db5e8962-a190-44ac-a384-44d63db5b3dd",  
      "name": "Credito recargas",  
      "description": "Tu crédito de recargas bajo demanda",  
      "amount": {  
        "value": 26.88,  
        "currency": "EUR",  
        "tax_included": true  
      },  
      "expiration_date": "2017-01-18T00:00:00Z",  
      "type": "general"  
    },  
    {  
      "id": "db5e8962-a190-44ac-a384-44d63db5b3dd",  
      "name": "Credito inicial",  
      "description": "El credito inicial que te regalamos",  
      "amount": {  
        "value": 5,  
        "currency": "EUR",  
        "tax_included": true  
      },  
      "expiration_date": "2017-01-01T00:00:00Z",  
      "type": "general"  
    }  
  ]  
}
```

Ficheros JSON vs XML

Es un lenguaje usado para el intercambio de datos entre sistemas, está basado en la notación de los literales de objeto de Javascript.

La utilidad de JSON es la de intercambiar datos, por eso se conoce como lenguaje de intercambio de información o lenguaje de transporte

Características de JSON

- JSON es un lenguaje de modelador de datos
- Consiste en pares "clave - valor"
- Los valores pueden cadenas, números o booleanos, así como otros objetos JSON, con cualquier nivel de anidación
- Es un formato flexible, ligero y fácilmente transferible a través de las redes

Ventajas de JSON

- La lectura del código resulta de fácil lectura y la información es suficientemente expresiva para poder ser leída por personas.
- El tamaño de los archivos que se transfieren es ligero.
- El código está basado en el lenguaje Javascript, lo que es ideal para las aplicaciones web.
- Todos los lenguajes disponen de funciones para interpretar cadenas JSON y convertir datos en cadenas JSON válidas.

JSON vs XML

JSON es Javascript y parsearlo para que pueda ser introducido en la memoria del navegador, como un objeto nativo Javascript es prácticamente inmediato, no requiere coste alguno para el intérprete de Javascript sin embargo en el caso de XML si se necesitan algoritmos para recorrer los arboles de etiquetas

```
{
  "marcadores": [
    {
      "latitude": 40.416875,
      "longitude": -3.703308,
      "city": "Madrid",
      "description": "Puerta del Sol"
    },
    {
      "latitude": 40.417438,
      "longitude": -3.693363,
      "city": "Madrid",
      "description": "Paseo del Prado"
    }
  ]
}
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
  <marcadores>
    <latitude>40.416875</latitude>
    <longitude>-3.703308</longitude>
    <city>Madrid</city>
    <description>Puerta del Sol</description>
  </marcadores>
  <marcadores>
    <latitude>40.417438</latitude>
    <longitude>-3.693363</longitude>
    <city>Madrid</city>
    <description>Paseo del Prado</description>
  </marcadores>
</root>
```

API Documentation

- Documentación
- Introduction to OpenAPI
- API Swagger Format

04

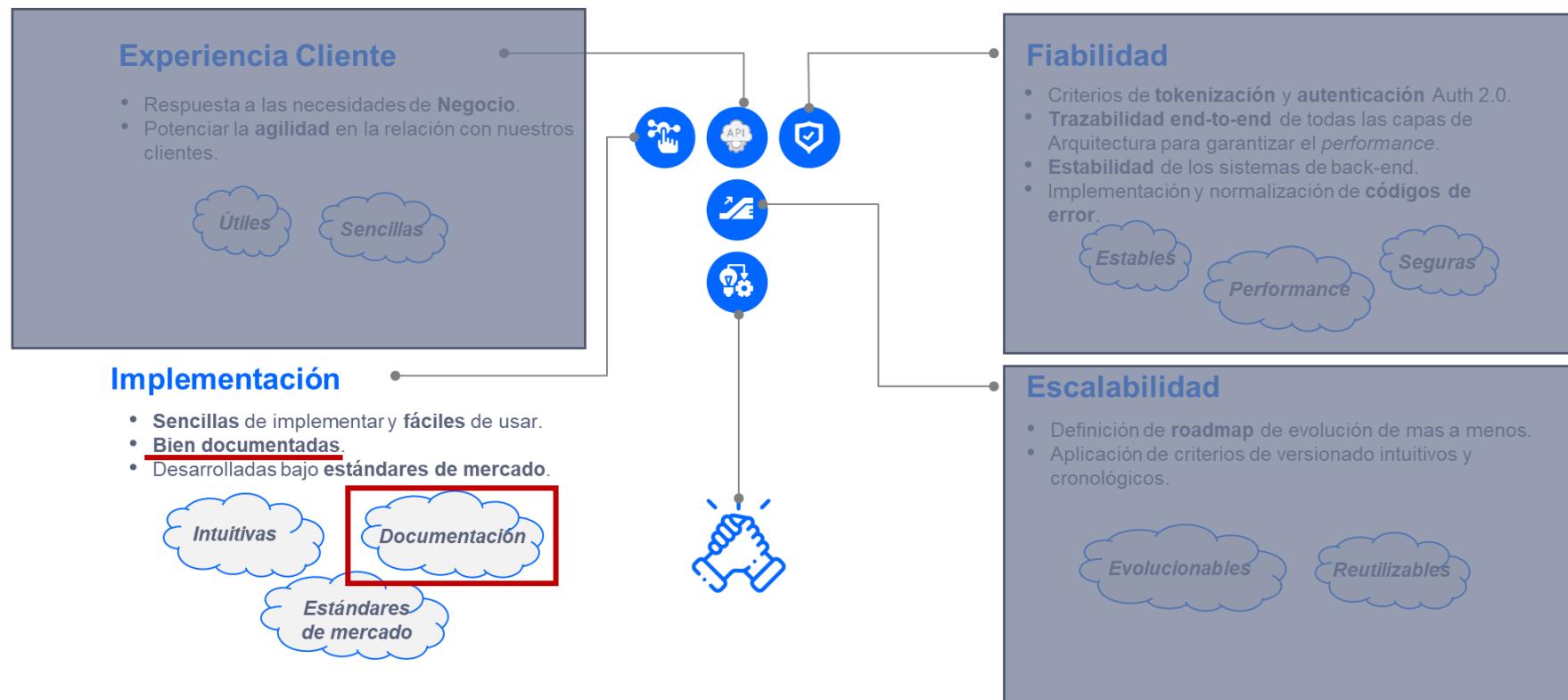
Muchas veces cuando hablamos de documentación nos encontramos algo como esto...



APIS: Documentation

No podemos pretender que nuestras APIs sean usadas si no están bien documentadas

Sin documentación no sabe como implementar un API ¿Para que vale? ¿Qué inputs se necesita? ¿Qué Salidas?



Importancia de la Documentación en el ecosistema de las API Web

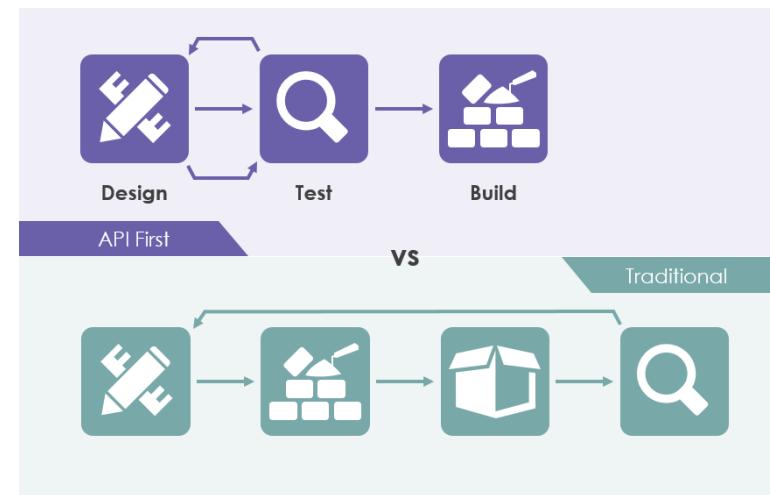
- Critico para la implementación de la API
- Acelera los tiempos de implementación
- Atrae a los desarrolladores (Potencia se Divulgación)
- ¡Mejora tu imagen de la compañía o desarrollador!

APIS: Documentation

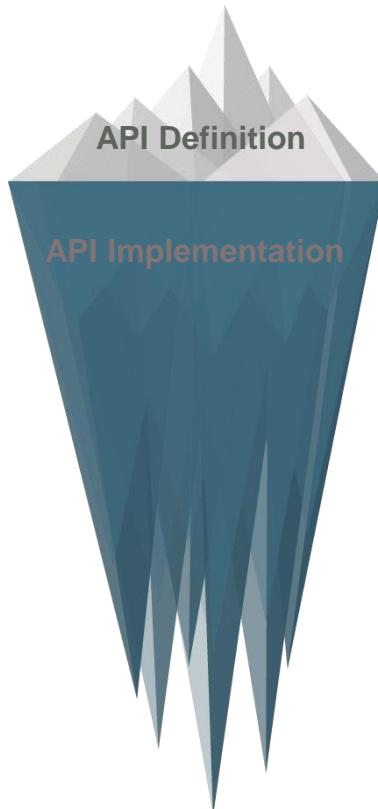
OpenAPI



- OpenAPI es el estándar para documentar un API
(Es el más extendido entre la red de desarrolladores)
- Realiza una descripción formal de todos los componentes que conforman un API REST (Endpoints, Parámetros, Autenticación, e información a cerca del API)
- Open API es conocido formalmente como Swagger
- Es el impulsor del concepto (Design-First)



APIS: Documentation



Durante el proceso de diseño y definición del contrato de la interfaz se lleva a cabo la creación de documentación.

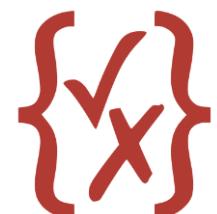
Este proceso utiliza herramientas y lenguajes específicos dedicados y enfocados a definir contratos de interfaces programáticas de Aplicación.

Existen numerosos lenguajes y herramientas diferentes, los más usados en este mercado son JSON SCHEMA como lenguaje, SWAGGER Editor como herramienta de diseño y SWAGGER Code Generator como generador de código de implementación.



- Documentamos mientras diseñamos
- Nos permite tener al día la documentación

Lenguaje



Editor



Code Gen



APIS Documentation



Json Schema está por supuesto descrito en notación JSON o YAML y éste se compone de varios bloques requeridos para un diseño estándar y completo:

BasePath: En este campo simple se indica cual será el identificador el API en todos los path de sus operaciones seguido de el identificador de versión mayor del API.

host: api.baikalplatform.com

Definitions: Este objeto contiene absolutamente todas las definiciones de objetos que se van a referenciar durante el diseño, ya sean en requests, responses o headers.

```
definitions:
  Balance:
    type: object
    description: Data of the balance
    required:
      - phone_number
      - wallets
    properties:
      phone_number:
        type: string
        description: >-
          Phone number for which the balance information applies.
          +
          format.
      wallets:
        type: array
        items:
          $ref: '#/definitions/Wallet'
  Wallet:
    type: object
    description: >-
      A wallet is a collection of funds held by a user.
```

```
expiration_date:
  type: string
  format: date-time
  description: >-
    Time when the current balance will expire,in ISO-8601 extended
    local
    date format. Time-offset from UTC may be used to match local OB
    time.
```

APIS Documentation



Host: Este campo simple contiene el FQDN donde el API se alojará.

host: api.baikalplatform.com

Info: Se trata del objeto donde se encuentra detallado el título o nombre del API, su versión completa (Major.Minor.Patch) y la descripción del propósito de dicha interfaz.

```
info:  
  description: >-  
    Retrieve monetary mobile balance for prepaid or control lines.  
  
  # Relevant Definitions and concepts  
  
  - Balance: Money available in a prepaid or control line. A control  
    line  
    is one that has both postpaid service (usages and services are charged in  
    a  
    bill) and prepay service (usages and services are charged against a  
    previously acquired/assigned money, i.e.: the balance); In this case, the  
    prepay service of the control line is having the balance.  
  
  version: 3.2.0  
  title: Mobile balance
```

APIS Documentation



Paths: Este campo contiene el detalle de todas y cada una de las operaciones disponibles en el API, dentro de estas se detallarán los verbos que aplican, headers, request, responses, expected codes...

```
paths:
  /users/{user_id}/phone-numbers/{phone_number}/balance:
    get:
      security:
        - three_legged:
            - mobile-balance-read
      description: Get info of monetary balance for the phone number
                   matching the query
      tags:
        - balance
      operationId: getBalance
      parameters:
        - $ref: '#/parameters/UserId'
        - $ref: '#/parameters/PhoneNumber'
        - in: header
          name: x-correlator
          type: string
          required: false
          description: Correlation id for the different services
      summary: Retrieve phone-number balance
      responses:
        '200':
          schema:
            $ref: '#/definitions/Balance'
          headers:
            x-correlator:
              description: Correlation id for the different services
              type: string
```

APIS Documentation



Schemes: El array de esquemas indica los protocolos que soportan la definición, normalmente en las WebApis aparecerá siempre https.

```
schemes:
```

```
- https
```

Swagger/openapi: En este campo se detalla la versión del swagger en la que estamos diseñando. Existen dos posibilidades actualmente: Swagger 2.0 o OpenAPI (También conocido como Swagger 3.0, esta versión sufre cambios de identificador en los campos descritos)

```
swagger: '2.0'
```

```
openapi: 3.0.1
```

Tags: Por último pero no menos importante, el array de tags agrupa todas las etiquetas de categorización de operaciones dentro del modelo, es decir, en un API con numerosas operaciones podemos dividirlas en bloques identificados mediante estos Tag.

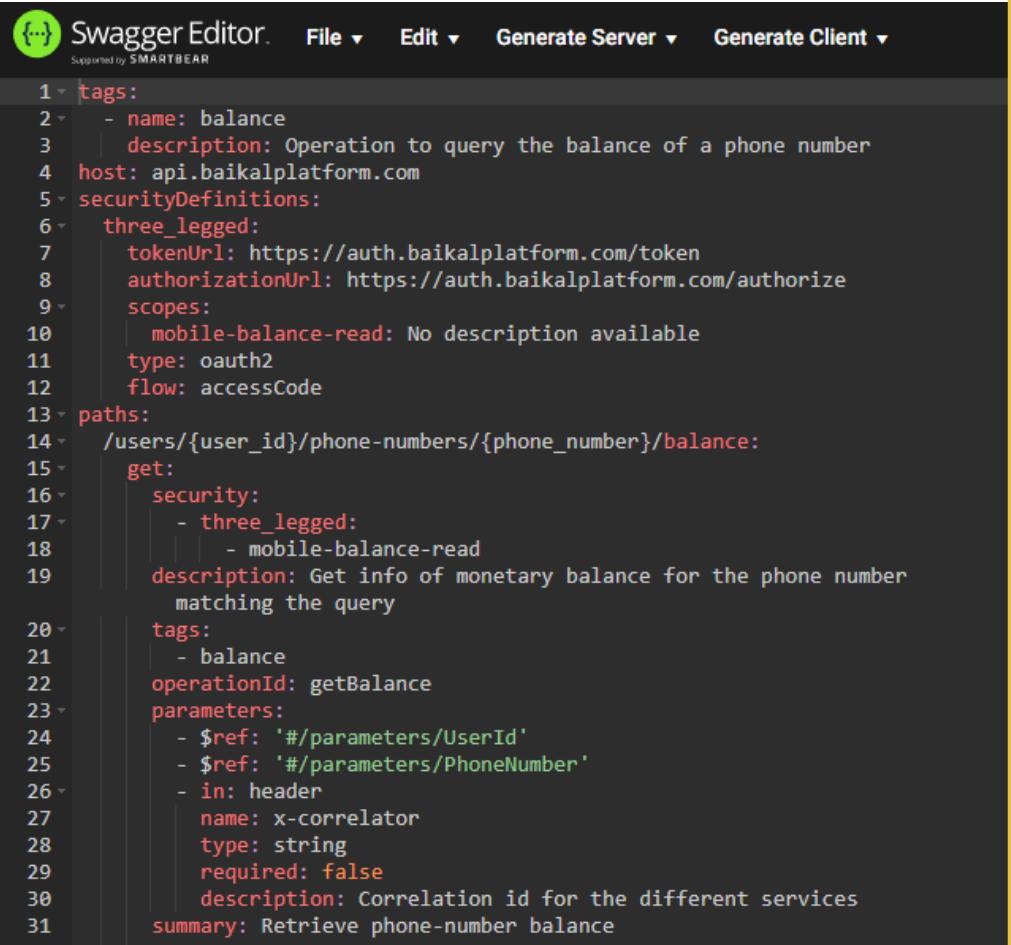
```
tags:
```

```
- name: balance
```

```
  | description: Operation to query the balance of a phone number
```

APIS Documentation

Code ←



```

1 tags:
2   - name: balance
3     description: Operation to query the balance of a phone number
4 host: api.baikalplatform.com
5 securityDefinitions:
6   three_legged:
7     tokenUrl: https://auth.baikalplatform.com/token
8     authorizationUrl: https://auth.baikalplatform.com/authorize
9     scopes:
10       mobile-balance-read: No description available
11     type: oauth2
12     flow: accessCode
13 paths:
14   /users/{user_id}/phone-numbers/{phone_number}/balance:
15     get:
16       security:
17         - three_legged:
18           - mobile-balance-read
19         description: Get info of monetary balance for the phone number
20         matching the query
21       tags:
22         - balance
23       operationId: getBalance
24       parameters:
25         - $ref: '#/parameters/UserId'
26         - $ref: '#/parameters/PhoneNumber'
27         - in: header
28           name: x-correlator
29           type: string
30           required: false
31         description: Correlation id for the different services
32       summary: Retrieve phone-number balance
  
```



Mobile balance 3.2.0

[Base URL: api.baikalplatform.com/mobile-balance/v3]

Retrieve monetary mobile balance for prepaid or control lines.

Relevant Definitions and concepts

- **Balance**: Money available in a prepaid or control line. A control line is one that has both postpaid service (usages and services are charged in a bill) and prepay service (usages and services are charged against a previously acquired/assigned money, i.e.: the balance); In this case, the prepay service can be used while the control line is having the balance.
- **Wallet**: Where the phone-number balance is placed. A phone-number may have its balance placed in different wallets, which may have different conditions in terms of balance expiration and/or what can be done with the balance. For example a wallet for promotional balance which can be used only for calls and SMSs while in home network, or a wallet valid for communication services but not for purchases.

API Functionality

This API has the single functionality of querying the balance of a mobile phone-number of a user.

Resources and Operations overview

→ Real Time Renderer

APIS Documentation

The screenshot shows the Swagger UI interface. At the top, there's a dropdown menu for selecting API version (3.0.0) and tabs for 'CLIENT CODE' and 'SERVER CODE'. Below the tabs, there's a 'download' button and a list of supported languages/frameworks:

- Bash
- C++
- Qt C++ & QT5
- C++ & Tizen
- C# & .net 2.0
- C# & .net >= 3.5
- Clojure
- Dart
- ELM
- Ada
- C++ & Pistache
- C++ & RestBed
- C# & ASP.net Core
- C# & Nancy FX
- CWiki
- Elixir
- Erlang
- GO



Lenguaje



Editor



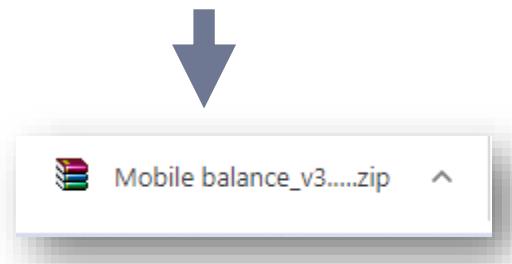
Code Gen



CodeGen de Swagger permite que a partir de un esquema de interfaz ya definido se genere el código de consumidor o expositor de forma automática en numerosos lenguajes disponibles en función de tus propósitos y necesidades.

Los lenguajes disponibles en Code Gen varía en función de dos factores:

- Versión del Esquema (2.0 u OpenAPI)
- Propósito (Servidor o Cliente)



Seguridad, Performance y Monitorización de las APIs

- **Introducción a Autenticación and Autorización**
 - OAuth2.0 Protocol
 - JWT and JWT in REST API (JWT.IO)
- **Introducción a Performance**
 - Data streamins & Async Operations
 - Caching
 - Rate Limit & Quota
- **Monitoring**
 - Métricas de Monitorización
 - Implementar un proceso de Monitoring & Tools

05

APIS: Seguridad

La seguridad es un factor crítico en las APIs

el nivel de seguridad se aplican en función de la Sensibilidad del Dato que se expone

Pero todas deben ser seguras

APIs Públicas



- Son APIs que suelen exponer información o servicios no sensibles, por lo que suelen ser de fácil acceso, y suelen tener métodos de seguridad más relajados.

APIs de Terceros



- Son APIs que suelen exponer información sensible entre terceros de confianza, por lo cual requieren de proceso de autenticación y autorización, ya que se exponen datos y transacciones muy sensibles entre organizaciones.

APIs Privadas



- Son APIs que suelen exponer información sensible entre los backend de las organizaciones y solo son accesibles por los dueños de las APIs (Organización), es necesario que tengan un alto grado de seguridad al igual que la de terceros

Principales Vulnerabilidades

1 Inyección



Los errores causados por inyección muy completo (como inyección SQL) se producen cuando datos maliciosos son enviados al servidor y pueden ser interpretados como comandos o consultas que pueden habilitar acciones indeseadas

Amenazas	Dificultad del ataque	Prevalencia del riesgo	Detección del riesgo	Impacto técnico
	Fácil	Común	Fácil	Severo

2 Rotura Autenticación



Las funciones relacionadas con la autenticación y gestión de sesiones son implementadas incorrectamente permitiendo a los atacantes comprometer usuarios y contraseñas toques de sesión o explorar otros errores de implementación para asumir la identidad de otros usuarios

Amenazas	Dificultad del ataque	Prevalencia del riesgo	Detección del riesgo	Impacto técnico
	Fácil	Común	Media	Severo

3 Exposición Datos Sensibles



Los atacantes pueden robar o modificar información financiera, de salud, de índole personal que están protegidos inadecuadamente para llevar a cabo fraude con tarjetas de crédito robos de identidad u otros delitos los datos sensibles requieren métodos de protección adicionales como el cifrado en almacenamiento y tránsito

Amenazas	Dificultad del ataque	Prevalencia del riesgo	Detección del riesgo	Impacto técnico
	Media	Generalizada	Media	Severo

Identidad Digital: Triple A

La **identidad digital** es el mecanismo mediante el que un usuario puede demostrar su identidad en el ámbito digital, para poder acceder a un sistema/servicio online y realizar gestiones en el mismo.

Está compuesta por:



Proporciona al usuario el acceso a todos los servicios contratados o habilitados en Telefónica o de terceros de manera transparente

Protocolo de Autenticación - Auth2.0



Access Token – JWT

JSON Web Token. Contiene toda la información necesaria para autenticar y autorizar al usuario.

Componentes:

- *Header – Contiene el tipo de Algoritmo que se usa*
- *Payload – Contiene la información del usuario que el Resorce Server necesita para proporcionar los recursos*
- *Signature – El JWT está firmado electrónicamente*

Todas las partes del JWT están concatenadas y codificadas bajo algoritmos de Base 64

Autenticación y Autorización

La **identidad digital única** del cliente con **single-sign-on** en todos los canales consta de los siguientes **componentes**



1 API Gateway – Securiza el acceso a las APIs dependiendo del usuario, la aplicación cliente y los permisos de ambos



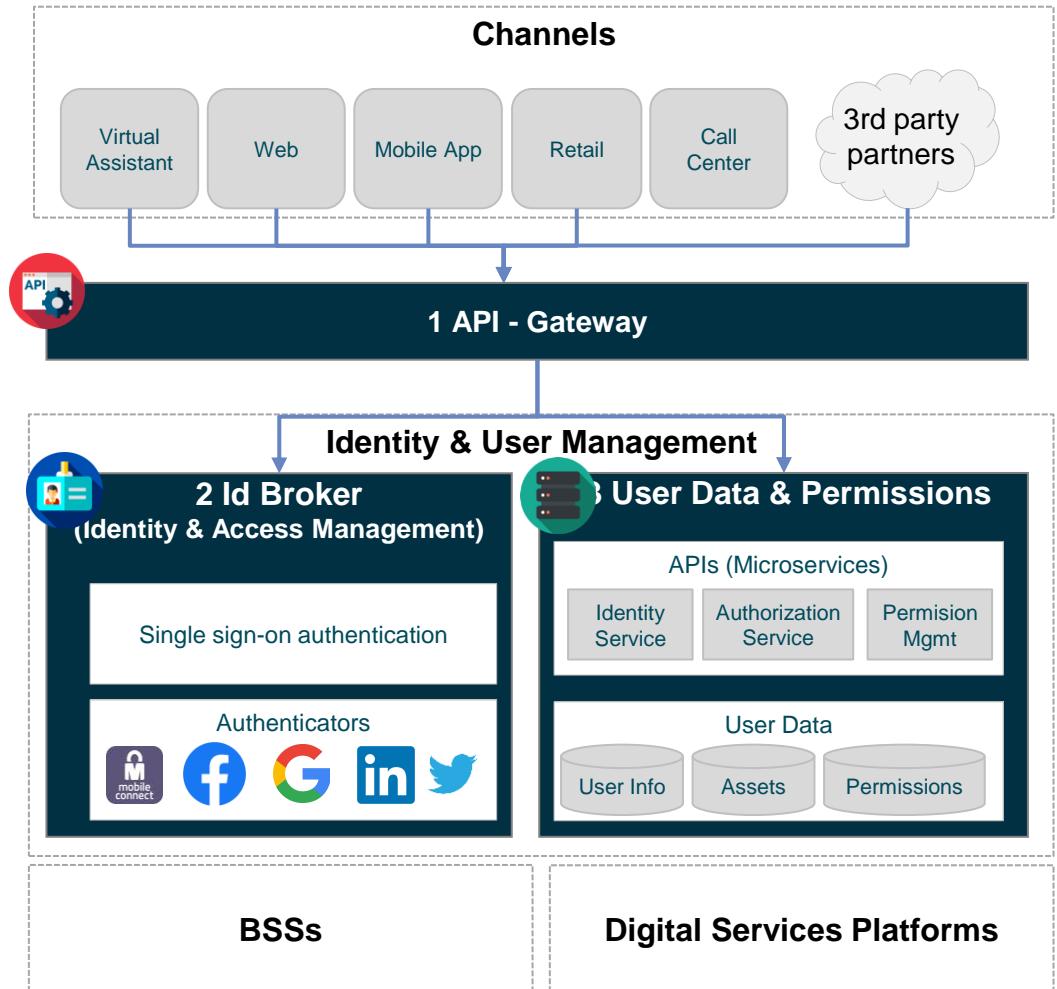
2 Id Broker – Realiza la actividad de autenticación, permitiendo acreditar que el cliente es quien dice ser. Principales características:

1. Único punto de acceso a los autenticadores
2. Registra la relación de la identidad del usuario en los IDPs (LDAP, mobile connect, Facebook, etc.)
3. Devuelve a la App u otros canales la identidad digital única del cliente en Telefónica (UserID).

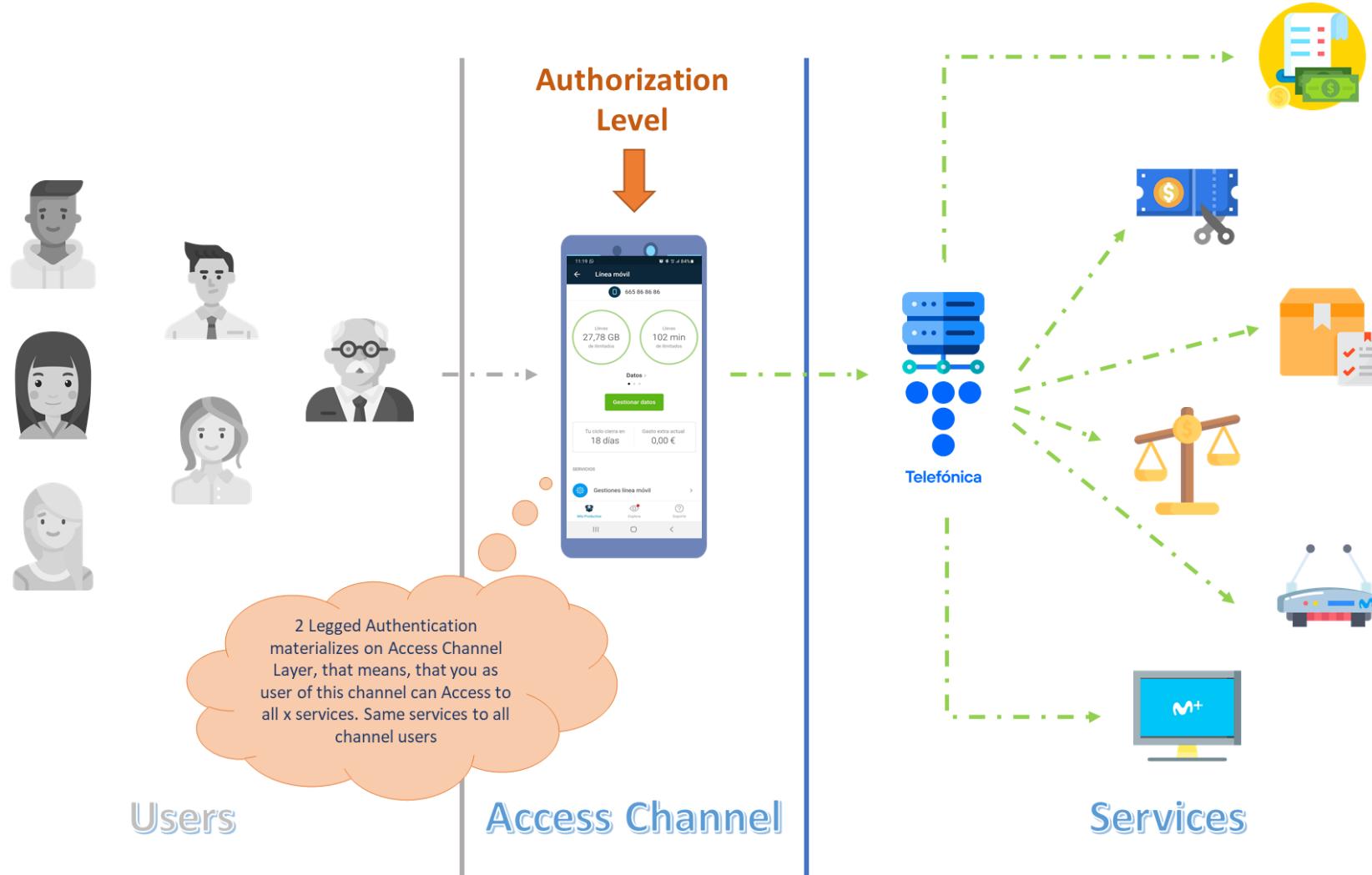


3 User Data – Implementada bajo tecnología FastData, realiza las siguientes acciones:

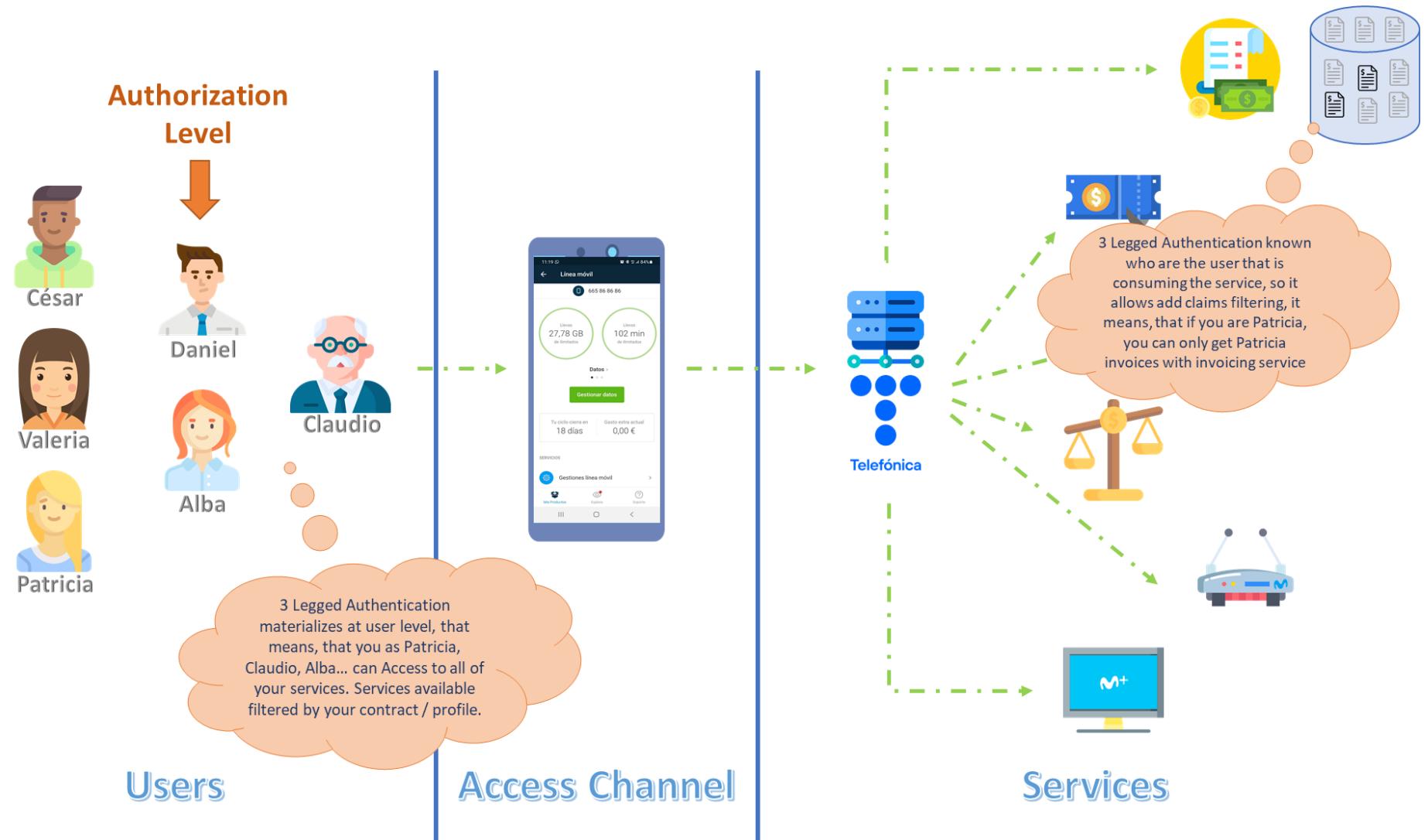
1. Habilita la información del cliente (Perfil) con independencia del canal
2. Establece diferentes niveles de confianza en función del método de autenticación usado previamente por el cliente
3. Establece y gestiona los permisos de los que dispone el cliente para los diferentes activos a los que tiene derecho.



Authorization - 2 Legged



Authorization - 3 Legged



Autorización – Aplicado a Telefónica



1st Level – 3 Legged

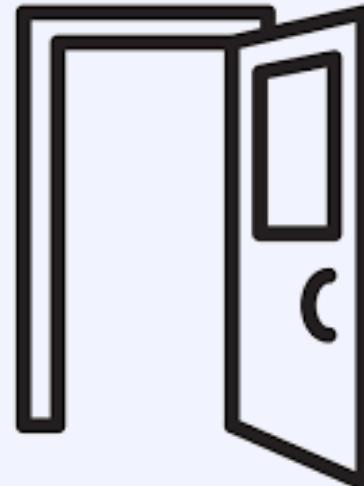


2nd Level – 2 Legged

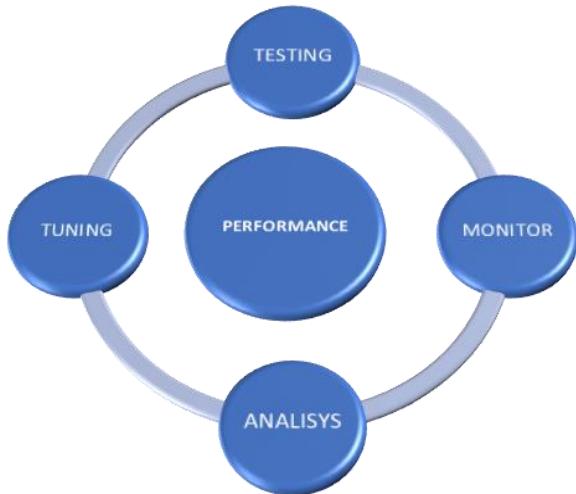


Services

Sin Seguridad no hay APIs



Performance



API = Interfaz + Implementación

El objetivo del performance, es **garantizar** que la implementación de la APIs, con las capacidades de infraestructura disponibles, **proporciona el tiempo de respuesta requerido** para la concurrencia de peticiones esperada.

Está regulado, que el **tiempo máximo** de una operación de cliente en Canal, para **garantizar la Experiencia de cliente**, debe ser **inferior a 2 segundos**, siendo **1 segundo**, el establecido como **máximo para las APIs**.

Con la idea de simplificar estas pruebas y detectar desde las fases tempranas los posibles puntos de mejora en las capas que intervienen en la ejecución de APIs, se plantean dos tipos de pruebas:

- **Análisis proactivo.**
- **Pruebas de performance.**

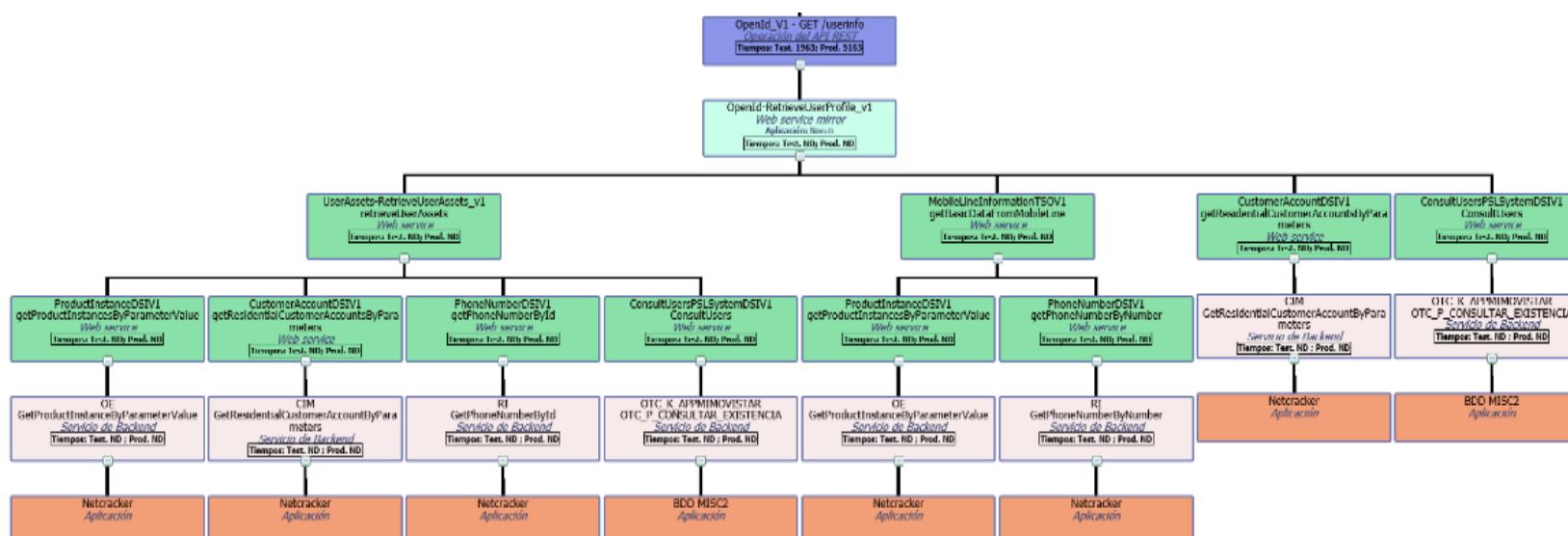
Performance: Análisis Proactivo



El análisis proactivo es un análisis de estimación de tiempo de respuesta, **partiendo del conocimiento de los componentes utilizados.**

Se realiza en las fases tempranas del diseño, donde se identifican los servicios / Apis de los sistemas que proveen la información.

El resultado de este análisis, permite actuar en la mejora de los tiempos de respuesta de los sistemas que peor rendimiento den o cambiar el diseño e incluso Arquitectura de implementación – Caching (FastData), Modelos Asíncronos, Establecer de Quotas y Rates -



Performance: Pruebas de Performance

HORA/FEC	01-Ago	02-Ago	03-Ago	04-Ago	05-Ago	06-Ago	07-Ago	08-Ago	09-Ago	10-Ago	11-Ago	12-Ago	13-Ago	14-Ago	15-Ago	16-Ago	17-Ago
HA	2402	2775	2171	0	2581	0	2340	3245	2464	2818	2864	8790	0	2545	2184	2277	2794
00	1081	1187	1166	0	1560	0	954	1309	1279	1369	1691	3786	0	1195	1068	1080	1246
01	564	674	589	0	1001	0	540	718	611	820	967	2445	439	649	563	669	665
02	432	571	428	0	702	400	414	484	555	607	666	1486	579	442	444	447	554
03	473	562	543	0	595	587	464	741	557	607	666	1292	599	498	495	533	580
04	1120	1517	1190	744	617	1121	1105	1734	1195	1010	914	1679	1407	1201	1153	1181	1137
05	2921	4165	2767	2336	1521	3045	3252	4413	3035	2170	2233	4244	3844	3312	3076	2970	3163
06	4583	5895	4546	3952	2994	5166	5254	6393	4906	3598	3699	6903	6153	5085	4839	4808	4609
07	5696	6084	5873	5215	4313	6468	7056	5698	4605	5049	8438	8239	7344	6367	6644	1029	0
08	6308	6171	6982	5983	4685	6470	6603	7097	6081	5294	5695	8820	9695	9794	7262	7581	0
09	6357	6402	6887	6009	5071	7130	6986	6902	6349	5928	5876	8238	8946	7823	6802	7620	0
10	6203	6423	7419	6138	5078	7177	6950	6786	6390	6052	6016	7765	7912	7315	6510	7914	0
11	6816	6561	7512	6066	5206	7269	7024	6985	6876	7153	5981	7222	7766	7305	6814	8260	0
12	7572	6772	10112	6087	5308	7201	7398	7332	7229	7850	6005	6565	7897	7682	6944	8294	0
13	7489	6556	11794	6158	5427	7121	6860	7576	6805	8525	5872	6818	7824	7264	7286	9067	0
14	7382	6327	8320	5616	5100	6974	6887	7167	6267	7045	6021	6345	7222	6860	7921	9277	0
15	7132	6117	7436	5514	4945	7702	7916	6886	6777	6848	5934	6154	7843	6452	7868	10498	0
16	7479	7273	7834	5866	5121	7794	7562	9251	7397	6617	5934	6457	8261	6637	8438	11417	0
17	7742	7154	8755	6180	5653	7364	7698	11482	7528	6669	6647	6596	8309	7676	7491	10119	0
18	7893	7478	8328	6370	6131	8054	7746	11611	7357	7379	7002	7117	8523	7801	8164	9393	0
19	7815	7053	7585	6206	6377	8001	7824	9693	7236	7008	7204	7522	7948	7534	8415	8381	0
20	7310	6885	7141	6174	6025	7251	7450	8122	6581	6401	6852	7523	7535	6711	7238	7625	0
21	5845	5885	6122	5148	5117	6069	6438	6436	5658	5633	6163	6423	4807	6054	6359	5516	0
22	3945	3732	467	3820	2740	4125	4364	4234	3963	4213	5094	2750	3806	3902	3965	4191	0

Mapa de calor de la APP / servicio Actual

Las pruebas de performance, son pruebas de carga en un entorno similar al de producción o con capacidad conocida y extrapolable al entorno de producción.

El objetivo es conocer los tiempos de respuesta y la concurrencia máxima hasta la degradación de los tiempos o se producen errores.

Para realizar las pruebas, es importante conocer:

- Análisis de comportamiento en tiempo de APP / Servicios actual. Mapa de calor de ejecuciones por Día / hora.
- Estimación de concurrencia de la nueva API y estimación de crecimiento a corto / medio.
- Definición de pruebas de carga y modelo de ejecución, teniendo en cuenta estimaciones para analizar degradación.
- Análisis de resultados de tiempos por capas de la Arquitectura (API-Gateway, OSB, sistemas Backend).
- Acciones correctivas.
- Nuevas pruebas.

Los valores significativos de las pruebas y válidos, son principalmente el **percentil 95**, siendo **la media, el mínimo y el máximo valores de contexto**.

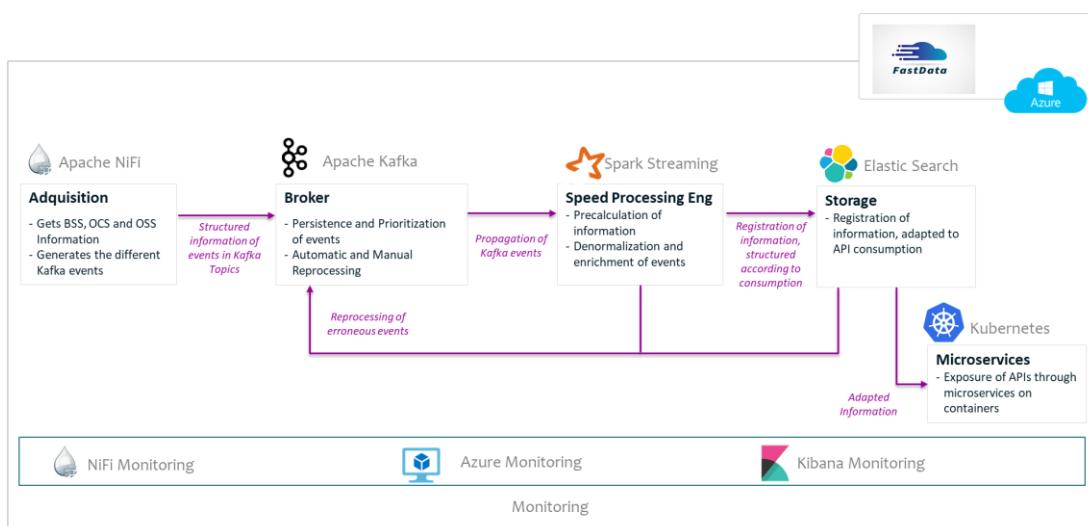
Performance: Acciones de Mejora

1

Caching - FastData

Es una solución Caching, maneja una gran cantidad de información de BSS y OSS que es preprocesada, agrupada y organizada de la manera más eficiente para ser expuesta por API, con actualización en tiempo real.

+ 90%	35x	24 x 7
Faster APIs response	Increase Concurrency	Information Availability
+ 60%	- 30%	- 60%
Faster APIs implementation	BSS& OSS Backend Cost Reduction	APIs Deployment Cost Reduction



2

Limit - rate

Se limita el número de llamadas que puede tener una API de forma concurrente, se limita el número de transacciones que se puede servir a nivel de API, no de usuario.

3

Quota

Limits the number of consumption to APIs for a specific User, limiting the number of use that a user can make in a range of time (e.g. 20 queries x second x IP Address)

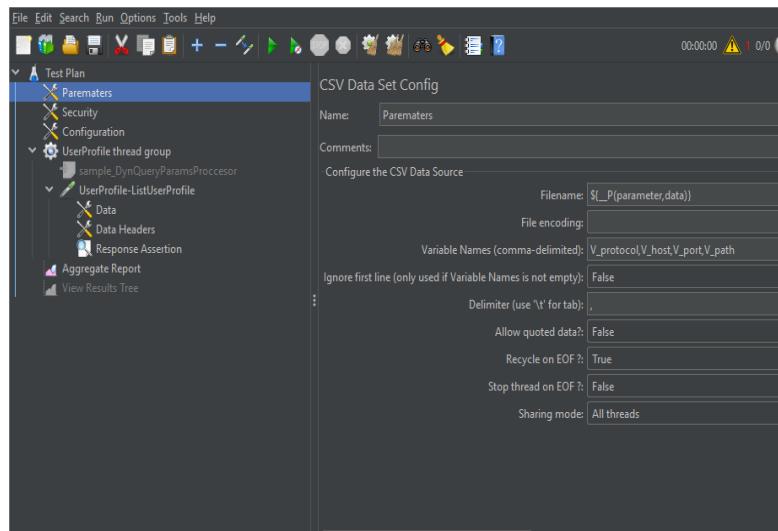
4

Async Operations

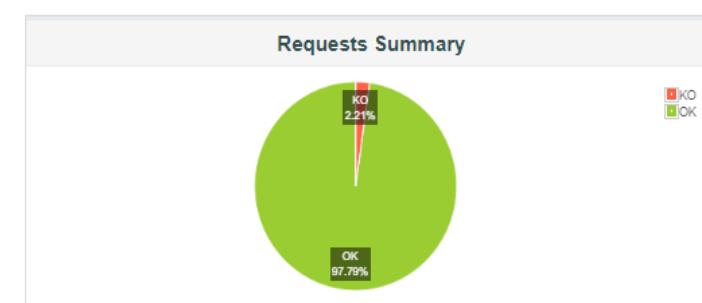
Este tipo de acción se usa para los casos en los que se puede solicitar una acción y no se requiere esperar a su finalización, se suele usar para proceso con generación de ficheros, acceso pesado a bases de datos (Históricos) donde no se requiere una entrega inmediata de la información..

Performance: Herramientas de Performance

Jmeter, es la propuesta OPEN que desde CTIO hacemos para su ejecución, pero vale cualquier otra, siempre que facilite la realización de las pruebas y reporte esperado.



APDEX (Application Performance Index)			
Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.000	500 ms	1 sec 500 ms	Total
0.000	500 ms	1 sec 500 ms	HTTP Request



Statistics

Requests	Executions				Response Times (ms)							Throughput		Network (KB/sec)		
	Label	#Samples	KO	Error %	Average	Min	Max	90th pct	95th pct	99th pct	Transactions/s	Received	Sent			
Total	1903	42	2.21%	3163.78	2399	9895	3389.00	3916.80	7589.12	6.20	7.31	2.13				
HTTP Request	1903	42	2.21%	3163.78	2399	9895	3389.00	3916.80	7589.12	6.20	7.31	2.13				

Monitoring

Monitorización Funcional



- Verificar y controlar parámetros de las APIs que aportan un valor funcional

Usuarios

Sesiones

Distribución Geográfica

Patrones de consulta por tipo cliente

Monitorización Técnica



- Verificar y controlar parámetros de las APIs que aportan un valor técnico

Peticiones por Segundo

y % Fallos

Consumo CPU

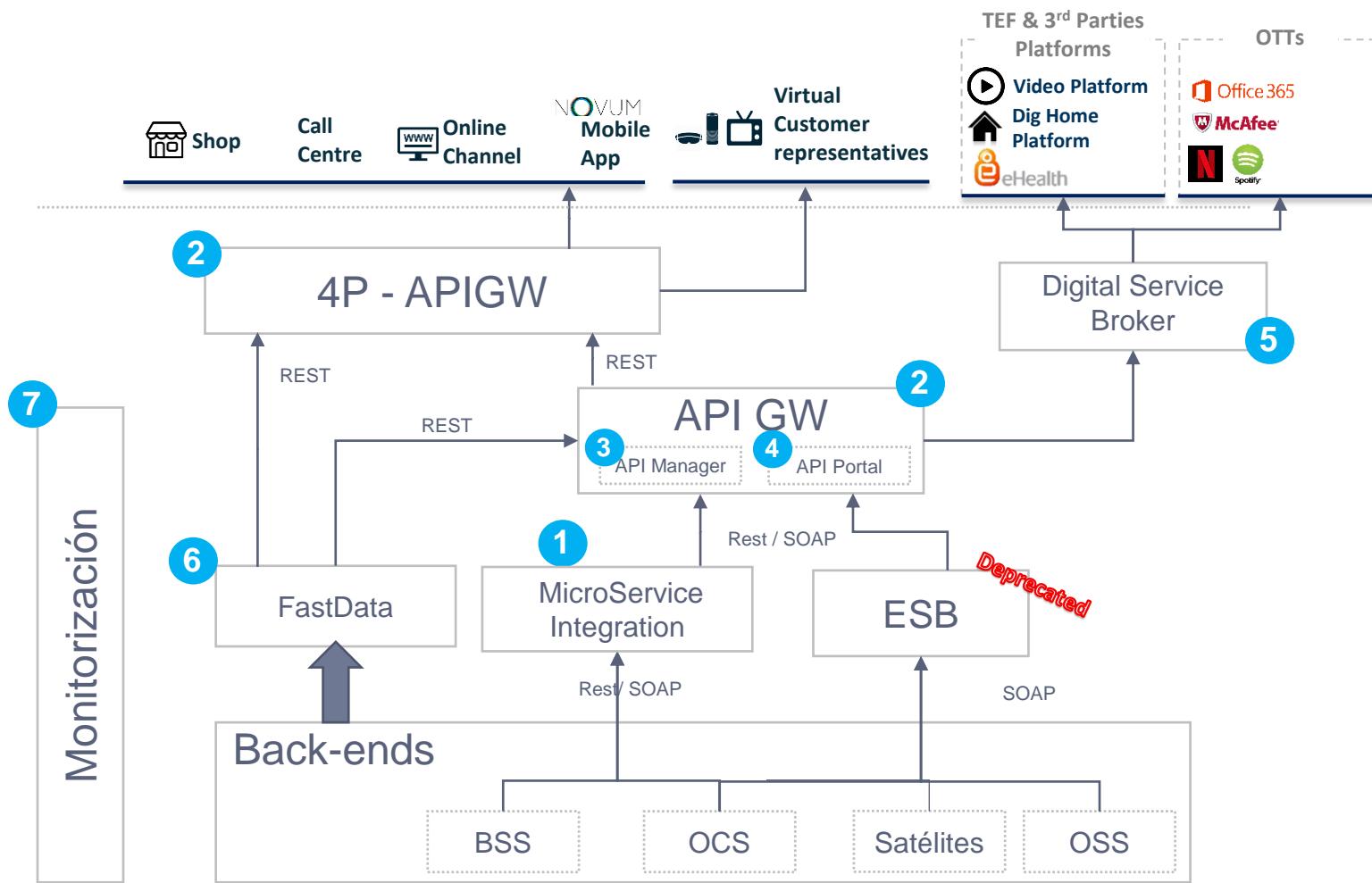
Consumo RAM y Network

Arquitectura Técnica de Componentes

- Arquitectura de referencia



Arquitectura Técnica



1 Microservice Integration & ESB:

- Orquestación de servicios de back-end.

2 API Gateway:

- Enrutamiento de mensajes.
- Autenticación, autorización y cifrado.
- Monitorización y SLAs.

3 API Manager:

- Gestión de ciclo de vida de las APIs.
- Publicación de APIs y end-points.

4 API Portal:

- Inventario de APIs.
- Diseño de servicios.

5 Digital Service Broker:

- Enrutamiento de peticiones digitales.

6 FastData:

- Agregado de información para consumo y acceso rápido.

7 Perform Monitoring (ELK):

- Monitorización end-to-end de las capas de Arquitectura para análisis de performance.

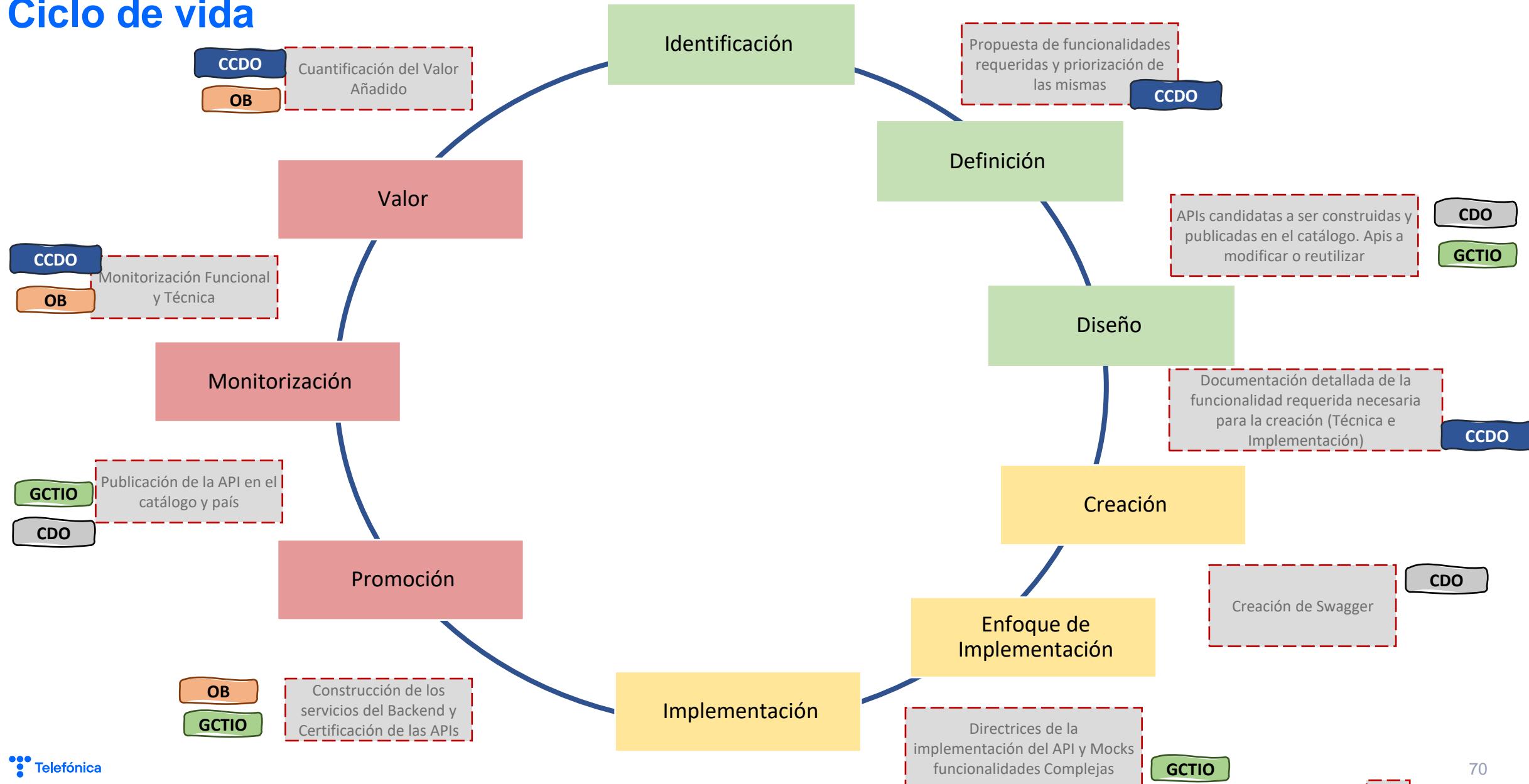
Ciclo de Vida del API

- Criterios de Versionado
- Herramientas (API Telefónica Catalog, Code Generator)



CICLO DE VIDA DEL API

Ciclo de vida



CICLO DE VIDA DEL API

Ciclo de vida

Fase de Diseño : Engloban la identificación de la API, su objetivo funcional y diseño de la interfaz.

Identificación

En la fase de Identificación, el equipo de negocio realiza un enfoque Top-down, donde se hace una propuesta de identificación de las funcionalidades que deben ser resultas a través de APIs.

- Especificación de las necesidades funcionales requeridas para los programas globales.
- Determina la prioridad que requiere cada una de las necesidades identificadas
- Las funcionalidades deben ser estructuradas por programa

CCDO

Entregable

Propuesta de funcionalidades requeridas (acompañadas de un definición funcional de las mismas)

Priorización de las funcionalidades y registro en la herramienta Redmine

Potencial lista de APIs que soportar la funcionalidad demandada

Definición

En este proceso se **analizan las necesidades de negocio para obtener un conjunto de APIs que puedan aportar valor a dicho negocio**. En esta fase se realiza el enfoque bottom-up haciendo una identificación proactiva desde el punto de vista técnico.

- Guiar a los responsables de negocio para filtrar aquellas APIs que verdaderamente aporten valor y que puedan ser implementables.
- Determina para cada proyectos la estructuración de las APIs candidatas.
- Identificar si esas APIs ya existen parcial o totalmente en la compañía

CDO

GCTIO

Entregable

Determinación de la tecnología que va a exponer las necesidades funcionales (APIs, Ficheros, etc..)

APIs candidatas a ser construidas y publicadas en el catálogo.

APIs a modificar o reutilizar

Diseño

En esta fase, desde negocio, se define la **funcionalidad detallada requerida que se persigue con la API, a quién va dirigida y los objetivos que se pretenden conseguir** con su implementación, y una definición de detalle funcional considerando las casuísticas globales y específicas de cada OB.

En esta fase es clave la captura de conocimiento por parte de las personas de gobierno junto con los expertos funcionales y negocio en caso de que aplique, para asegurar que el diseño cumple con la funcionalidad definida.

CCDO

Entregable

Diseño detallado de las funcionalidades requeridas, con la especificación del tratamiento de las casuísticas globales y locales

El equipo de CDO y GCTIO supervisará que la definición está suficientemente bien documentada para poder ser revisada con la OB y establecer el enfoque de implementación.

GCTIO

CDO

CICLO DE VIDA DEL API

Ciclo de vida

La fase de desarrollo : se crea la definición de la API siguiendo el diseño de la fase anterior, la implementación del servicio backend y la configuración del mismo dentro de la plataforma de gestión de APIs si la hubiera incorporando la configuración de seguridad, límite de accesos, planes de consumo, etc..

Creación

Este proceso considera la definición técnica del API donde se establece y definen todos los recursos, tanto a nivel de **endpoints, campos de entrada y salida, formatos de respuesta y códigos HTTP**.

Una vez definida la interfaz de la API en el documento, en esta fase se realiza esa definición en el lenguaje de definición de APIs: Swagger por parte del *API designer*.

CDO-CTO

Entregable

Diseño completo de la interfaz en un documento de análisis.

CDO-CTO

Creación del Swagger del API

Validación del **Swagger** para corroborar que cumple lo definido en la fase de diseño, que sigue los estándares y buenas prácticas y que está suficientemente bien documentada como para ser expuesta.

CCDO

GCTIO

Enfoque de Implementación

El *API implementation Lead* define las directrices de la implementación del API, con el objetivo de garantizar un modelo standard en todas las operadoras con el mismo Full Stack y sistemas backend

Para verificar una correcta funcionamiento se desarrollaran mocks para las APIs más complejas, la certificación de las APIs se realizará tomando como base los Mocks generados.

GCTIO

Las OB debe realizar el diseño técnico de la implementación aplicando los estándares de arquitectura y las directrices definidas por el API Implementation Lead, y registrará la información en la herramienta TER

OB

Entregable

Diseño técnico de la implantación del API, documentado en la Herramienta (TER)

OB

Verificación que el diseño técnico de implementación se ha realizado siguiendo las directrices de arquitectura de integración.

GCTIO

Verificación de los Mocks creados por parte de los equipos de CDO y CCDO, el MOCK será el componente que se siga como directriz para la certificación

CCDO

CDO-CTO

Implementación

Implica el desarrollo de la lógica de negocio que hay detrás de la API partiendo de la definición de la interfaz y el diseño funciona.

Además el enfoque funciona debe también considerarse un enfoque técnico en cuyo ámbito entra la medición del rendimiento/performance de la API, tiempo de ejecución.

OB

Se realiza la certificación de API, asegurando el correcto funcionamiento a nivel funcional y técnico.

El *API implementation Lead* realiza la actividad de aclarar a los equipo de implementación las dudas técnicas e como implementar para obtener la funcionalidad deseada. También hace de nexo con los responsables funcionales para solucionar las dudas que surjan y corregir lo que sea necesario.

GCTIO

Entregables

Se crea el SW requerido para la implementacion del API

OB

Se establecen los scripts de medición del performance de las APIs, el cual será ejecutado por el equipo del Technical Lead con el objetivo de que se consigan los tiempos definidos por el *Product Owner*.

OB

Certificación funcional y técnica del API

GCTIO

CICLO DE VIDA DEL API

Ciclo de vida

Fase de ejecución: En esta fase se incluyen todas las tareas necesarias para la publicación, monitorización y evolución de las APIs.

Promoción

Esta fase se inicia tras la implementación de los servicios backend que tienen la lógica de negocio de la API, una vez certificada la API y publica en el catálogo, se pone a disposición de los programas para que empiecen a utilizarla.



Resulta una tarea básica a la hora de publicar una API su catalogación, indicando en qué países está disponible la API para cualquier consumidor.



Se debe revisar la Documentación de la API y actualizarla en función de los últimos ajustes, para disponer de una documentación actualizada.



Entregables

- Versiona y publica las APIs en el catálogo de la organización y garantiza que la calidad de la documentación es la adecuada.
- Consolida el feedback de su uso y funcionalidad.
- Socialización del uso de las APIs ya publicadas en el catálogo.



Monitorización el Uso

Esta actividad tiene dos objetivos:

- Uno funcional más orientado a negocio en el cual se generan informes para cuantificar el cumplimiento de los SLAs y objetivos definidos, cuantificar el uso, la reutilización, la calidad de la documentación .
- Un enfoque técnico en cuyo ámbito entra la medición del rendimiento de la API, tiempo de ejecución, tiempo de espera a los servicios invocados, para ver potenciales degradaciones o acciones de mejora. etc



Gobierno

El equipo de CCDO interviene en esta etapa para definir los indicadores que ofrezcan mayor información para la monitorización de la API y son los interlocutores para obtener el feedback de los usuarios.



Cuantificación del Valor

Esta fase tiene como objetivo es identificar el beneficio del uso de las APIs en los diferentes programas Globales /Locales.

Medición del uso y grado de satisfacción final del cliente



Cierre

- Conclusiones

08

Preguntas o Comentarios

M Mis consumos - Mi Movistar x

movistar.es/mimovistar-cliente/es-es/c_particulares/cclivr/consumos.html

Aplicaciones Trabajo Telefonica Formacion TV casa Gmail YouTube Maps Traducir Carreras Alfonso Viajes teatro CAMG Otros marcadores Lista de lectura

Productos y servicios Mis facturas Mis productos Mis consumos Soporte Mi perfil

MIN de ilimitados de 500 min de 50 min

DETALLE CONFIGURA TU ROUTER

629 188 449
Fusión+ 2
18/08 - 17/09 / Se emite el 01/10. 18 días para cerrar la factura 0,00 €

Esta línea tiene ahora incluidos datos ilimitados.

datos voz sms
0 MB 0 min 0 sms de ilimitados de ilimitados de ilimitados

DETALLE GESTIONAR LÍNEA SERVICIOS PUK DATOS COMPARTIDOS

676 782 292
Fusión+ 2/4 Incluida
18/08 - 17/09 / Se emite el 01/10. 18 días para cerrar la factura 0,00 €

Esta línea tiene ahora incluidos datos ilimitados.

?

9:55 94% ← Línea móvil

Alfonso Medina Olmedo

Llevas 366 MB de ilimitados

Llevas 0 min de ilimitados

Minutos >

Gestionar datos

Tu ciclo cierra en 16 días Gasto extra actual 0,00 €

SERVICIOS

Gestiones línea móvil >

Conexión Segura >

Mis Productos Explora Soporte



Mis consumos - Mi Movistar

movistar.es/mimovistar-cliente/es-es/c_particulares/cclvr/consumos.html

Aplicaciones Trabajo Telefónica Formación TV casa Gmail YouTube Maps Traducir Carreras Alfonso Viajes teatro CAMG Otros marcadores Lista de lectura

Productos y servicios Mis facturas Mis productos Mis consumos Soporte Mi perfil

629 188 449 Fusión+ 2 18/08-17/09 / Se emite el 01/10. 16 días para cerrar la factura 0,00 €

Esta línea tiene ahora incluidos datos ilimitados.

datos: 365,60 MB de ilimitados

voz: 0 min de ilimitados

sms: 1 sms de ilimitados

DETALLE GESTIONAR LÍNEA SERVICIOS PUK DATOS COMPARTIDOS

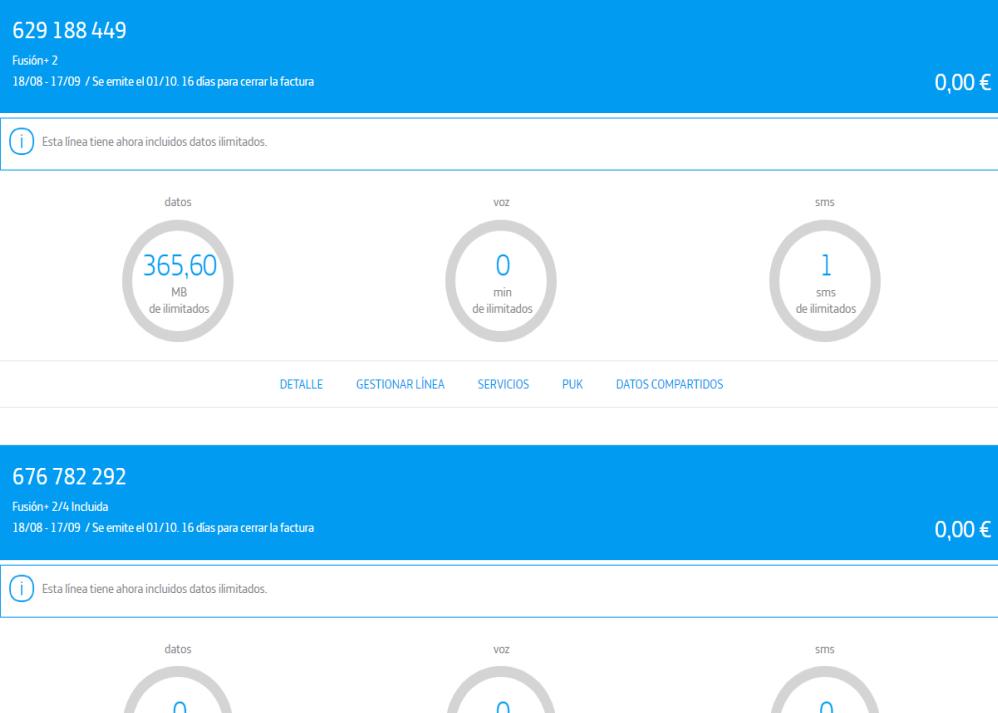
676 782 292 Fusión+ 2/4 Incluida 18/08-17/09 / Se emite el 01/10. 16 días para cerrar la factura 0,00 €

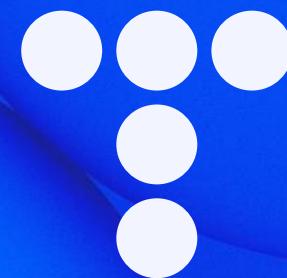
Esta línea tiene ahora incluidos datos ilimitados.

datos: 0

voz: 0

sms: 0





Telefónica

***IT&NETWORK
ACADEMY***

Conectando conocimientos