

# Ciberseguridad

## Grado en Ingeniería Informática

### M5 - Desarrollo seguro de aplicaciones

#### Seguridad en la toma de requisitos

Oscar Delgado  
[oscar.delgado@uam.es](mailto:oscar.delgado@uam.es)

Álvaro Ortigosa (Coord.)  
[alvaro.ortigosa@uam.es](mailto:alvaro.ortigosa@uam.es)

Objetivo: sistematizar la seguridad en el desarrollo

- Incluir la seguridad en TODAS las etapas de la metodología de desarrollo
- **Fase 0:** sensibilización y formación

# Sensibilización

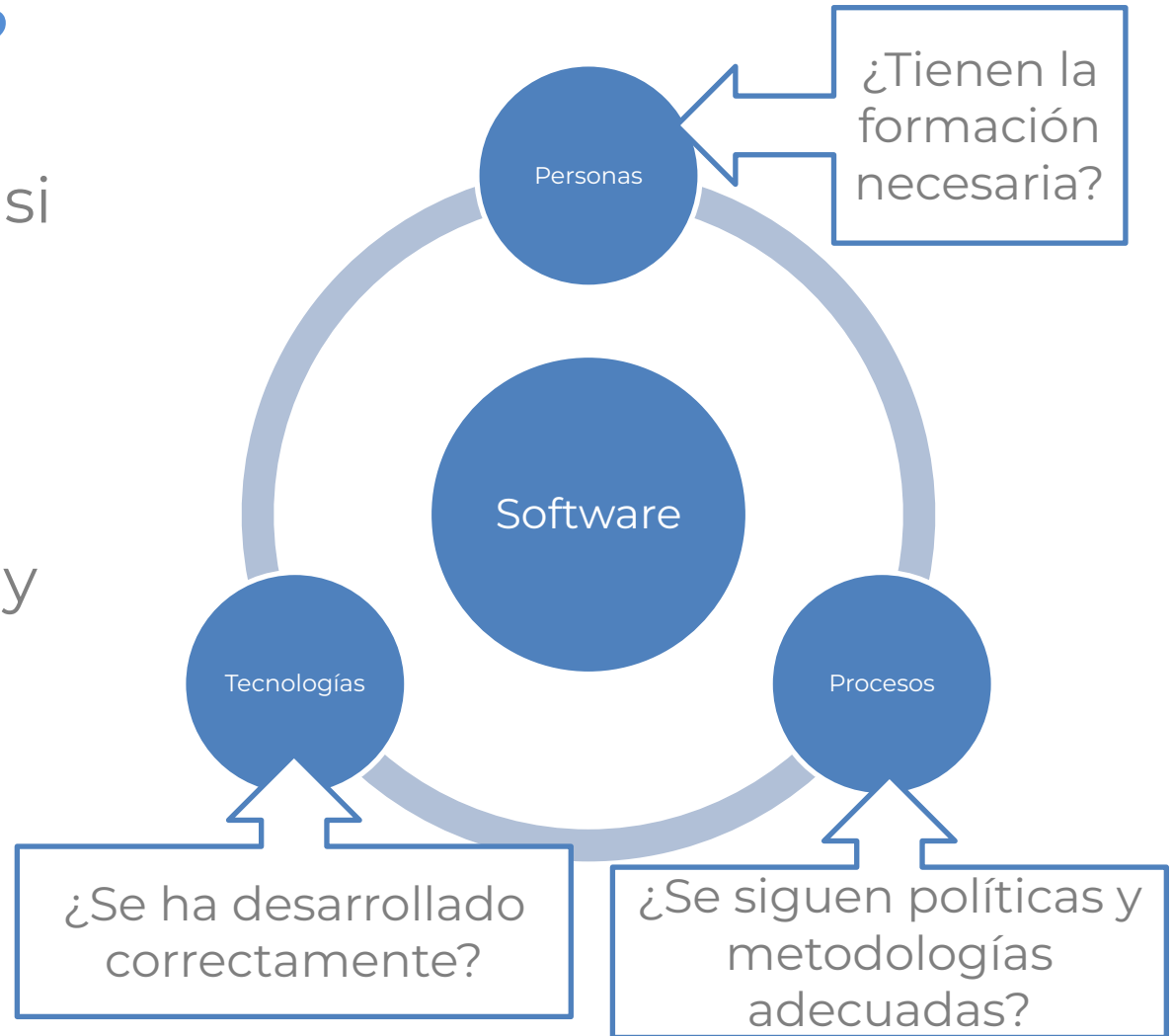
- Obtener el apoyo (real) de la dirección:
  - Insistir en que la seguridad NO implica más recursos, temporales o humanos o económicos
  - Material útil: kit de concienciación

## ¿Qué probar?


- El software lo hacen **personas**, a través de **procesos** y utilizando una **tecnología...**
- ...pero solo se prueba el software en sí

# ¿Qué probar?

- Solo se prueba, si se hace, el resultado
- Es necesario un enfoque global y sistemático



# ¿Cuándo probar?



¡El coste de no hacerlo desde el primer momento crece exponencialmente!

DEPLOY

# ¿Cuándo probar?

- Si la seguridad no se incluye desde ANTES de la fase de diseño, el resultado será:
  - Más **caro**: arreglar las vulnerabilidades es siempre la opción más cara
  - Más **lento**: el proyecto se retrasará inevitablemente
  - **Peor**: habrá aspectos de diseño que no podrán cambiarse, y se convertirán en vulnerabilidades estructurales

# El verdadero coste es...

## El desorbitado Mars Climate (1998)



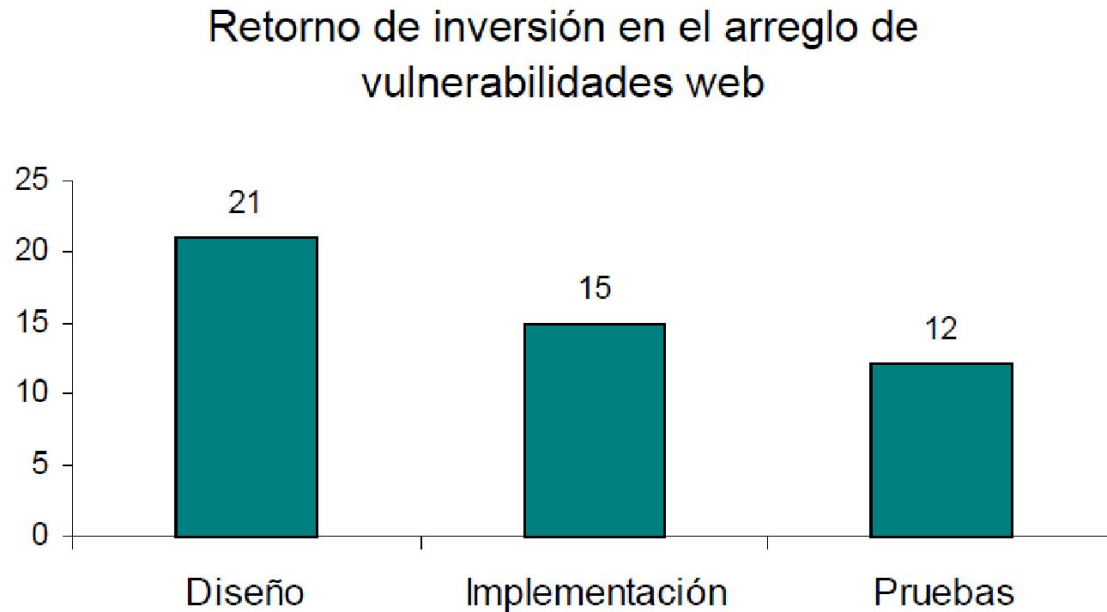
**Coste:** 125 millones de dólares.

**Desastre:** Después de un viaje de 286 días desde la tierra, la nave "Mars Climate Orbiter" encendió sus motores para ponerse en órbita alrededor de Marte. Los motores arrancaron, pero el ingenio entró demasiado en la atmósfera del planeta, provocando que se estrellara en su superficie.

**Causa:** El software que controlaba los propulsores del Mars Orbiter usaban unidades imperiales (libras de fuerza) en lugar de unidades métricas (Newtons), como especificaba la NASA.



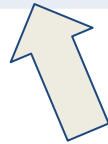
# El verdadero coste es...



Fuente: "Tangible ROI through software engineering" SBQ, vol 1, nº 2

# Tipos de pruebas de seguridad

	Objetivo	Metodología	Características	Duración
<b>Test de penetración</b>	Demostrar inseguridad	Trabajan en línea "recta", hasta encontrar un problema	Se suelen utilizar para demostrar la necesidad de una auditoría	Semanas
<b>Auditoría</b>	Encontrar todos los problemas	Utilizan una metodología sistemática	Pueden ser de <b>caja blanca o negra</b>	Meses



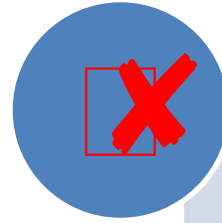
Tantos como sea posible

# Tests de intrusión



Rápidos

No  
requieren  
acceso al  
código



Sólo  
demuestran  
que existe  
algún fallo  
importante

No  
encuentran  
todos los  
problemas

Dependen  
mucho de la  
habilidad del  
experto

# Frameworks de pruebas y seguridad Web

OWASP

# OWASP: Open Web Application Security Project

- Proyecto libre creado en 2001, que aporta:
  - **Documentación:** estándares, publicaciones, artículos, conferencias
  - **Software** de entrenamiento y pruebas
  - **Asociaciones** locales

Documentación

**OWASP:** Open Web Application Security Project

- OWASP Top 10
- OWASP Guía de desarrollo de Aplicaciones Web Seguras (v4)

## OWASP Top 10

- Lista de las 10 vulnerabilidades más comunes y graves
- Se actualiza anualmente
- Cada vez más conocida y aceptada – cuasi-estándar

Documentación

# OWASP Top 10

A1. Entrada sin  
validar

A2. Controles de  
acceso  
inseguros

A3. Gestión de  
sesión y  
autenticación  
inseguros

A4. Cross Site  
Scripting

A5.  
Desbordamiento  
de buffers

A6. Problemas  
de inyección

A7. Gestión  
incorrecta de  
errores

A8.  
Almacenamiento  
inseguro

A9. Denegación  
de servicio

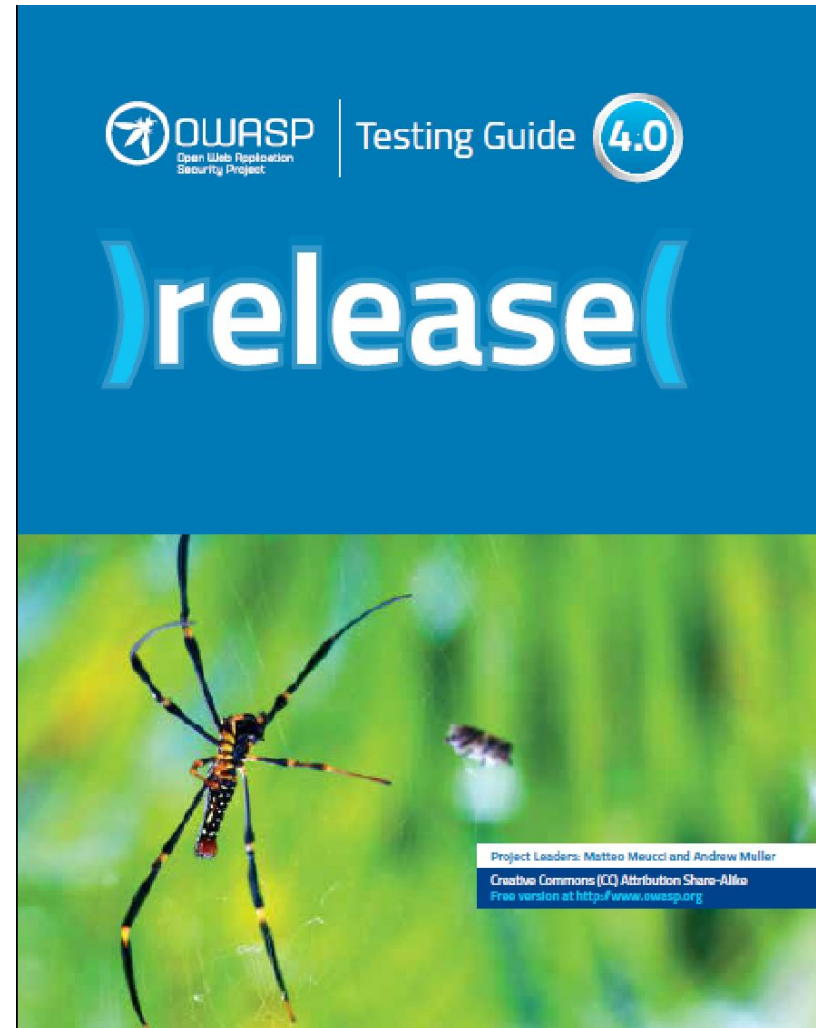
A10. Gestión  
insegura de la  
configuración



# OWASP Guía de desarrollo seguro

- Proporciona una buena base sobre el desarrollo seguro:
  - Introducción a la seguridad en general
  - Introducción a la seguridad en nivel de aplicación
  - Áreas claves: arquitectura, autenticación, gestión de sesiones, controles de acceso, validación de entrada y logging

# OWASP Guía de desarrollo seguro (v4)



# Uso de herramientas automáticas

- Proporciona una buena base sobre el desarrollo seguro:
  - Introducción a la seguridad en general
  - Introducción a la seguridad en nivel de aplicación
  - Áreas claves: arquitectura, autenticación, gestión de sesiones, controles de acceso, validación de entrada y logging

# Fase 1: Antes del desarrollo

- Incluir específicamente la seguridad en nuestra metodología de desarrollo
- Comprobar que existen políticas y estándares de seguridad conocidos por el equipo de desarrollo
- Tratar de definir métricas: problema abierto
- La mejor salvaguarda es que no se produzca el fallo: ¡concienciación y formación!

# Fase 1: Antes del desarrollo

- Designar una persona como responsable de la seguridad en el código, que se responsabilice específicamente de esta tarea
- Tratar de mantener el código tan simple como sea posible. La seguridad es siempre inversamente proporcional a la complejidad

## Fase 2: Durante el diseño

- Revisión de los requisitos de seguridad: grandes áreas anteriores: ASVS es un buen punto de partida
- Desde un punto de visto específico de la seguridad revisión de:
  - La arquitectura
  - El diseño, incluyendo modelos UML
  - Modelos de amenaza: crear escenarios realistas de ataques y atacantes

## Fase 3: Durante el desarrollo

- Revisión del código por el equipo encargado de la seguridad:
  - **1º fase:** explicación de la arquitectura y decisiones de alto nivel. Se realiza conjuntamente por ambos equipos.
  - **2º fase:** revisión profunda del código. Se realiza por el equipo de seguridad, diferente del de desarrollo.

## Fase 4: Durante el despliegue

- Antes del paso a producción, un test de penetración por un equipo especializado (interno o externo).
- Atención específica a la gestión de la configuración:
  - Usuarios admin “olvidados”, privilegios excesivos.
  - Securización también de la plataforma (SO, servidor Web, etc.)



## Fase 5: Operación y mantenimiento

- Foco en la gestión del cambio.
- Todo nuevo pase a producción debería ser aprobado por el equipo de seguridad.

# Frameworks de pruebas y seguridad Web

ASVS

## ASVS: Application Security Verification Standard

- Proyecto OWASP que pretende establecer un estándar sobre los requisitos y controles de seguridad más comunes
- Requisitos no-funcionales

# ASVS: Application Security Verification Standard



Dirigido a las aplicaciones más críticas: procesan transacciones económicas, registros médicos, etc...

Dirigido a aplicaciones que gestionan datos personales o sensibles

Dirigido a cualquier software

Dependiente de cada empresa

# ASVS: Application Security Verification Standard

## V3 : Requisitos de Gestión de Sesiones

#	Description	1	2	3	Since
3.1	Verify that there is no custom session manager, or that the custom session manager is resistant against all common session management attacks.	✓	✓	✓	1.0
3.2	Verify that sessions are invalidated when the user logs out.	✓	✓	✓	1.0
3.3	Verify that sessions timeout after a specified period of inactivity.	✓	✓	✓	1.0
3.4	Verify that sessions timeout after an administratively-configurable maximum time period regardless of activity (an absolute timeout).		✓	✓	1.0
3.5	Verify that all pages that require authentication have easy and visible access to logout functionality.	✓	✓	✓	1.0

Elicitación

# PIZZAS E IDEAS

# Problema

Una cadena de pizzerías quiere desarrollar un sistema de venta online de sus productos. Como suele ser habitual, el cliente quiere que el sistema sea innovador y haga que sus clientes se decanten por ellos en vez de por la competencia.

Para clarificar las ideas y requisitos sobre la aplicación el Ingeniero de Requisitos ha convocado una reunión con Negocio y Desarrollo donde se aplicará la técnica de Brainstorming.



Especificación

**¿REQUISITO, BIEN O MAL?**



## REQ 1

- Para reconocer al usuario tendrá que identificarse antes de entrar en la aplicación. La pantalla de identificación se ajustará según se trate de un móvil, tablet o PC.

# REQ 1 Respuesta

- MAL:
  1. No es un requisito, son varios.
  2. No es concreto:
    - ¿Qué tipo de identificación/autenticación se requerirá?
    - Identificación y autenticación son dos operaciones diferentes

## REQ 1 Corregido

- El usuario se autenticará con credenciales usuario/contraseña. ~~La pantalla de identificación se~~ ¿ajustará? ~~según se trate de un móvil, tablet o PC.~~

¿Justificación del requisito?

¿Por qué es necesario identificarse?

La autenticación es necesaria para limitar las acciones del usuario en el sistema.

## REQ 2, 3 y 4

- El sistema será lo más seguro posible.
- El sistema proporcionará una respuesta rápida.
- El sistema se recuperará automáticamente tras producirse un fallo.

## REQ 2,3,4 Respuesta

- MAL:
  1. No son requisitos, son objetivos generales.
  2. No son concretos ni **mensurables**
  3. No son claros

## REQ 2,3,4 Corregido

- El sistema se desarrollará con metodología X, y será objeto de auditorías antes de su paso a producción
- El sistema responderá en menos de 2 segundos.
- La plataforma dispondrá de un sistema espejo que se levantará automáticamente en caso de fallo

## REQ 5

- Con 100 usuarios concurrentes, el tiempo de respuesta será inferior o igual a 2 segundos.

## REQ 5 Respuesta

- BIEN:
  1. Claro
  2. Concreto
  3. Conciso
  4. Completo
  5. **Verificable**



## REQ 6, 7

- Los usuarios de la aplicación tienen que autenticarse antes de entrar por medio de un formulario que les solicitará usuario/contraseña para poder efectuar cualquier acción.
- Los usuarios de la aplicación tienen que autenticarse antes de entrar por medio del DNle para poder efectuar cualquier acción.

## REQ 6, 7 Respuesta

- BIEN:
  1. Claro
  2. Concreto
  3. Conciso
  4. Completo
  5. **Verificable**
- MAL:
  1. No son consistentes entre los dos.

Análisis y Especificación

# **GENERACIÓN DE REQUISITOS**

# REQ General

La **seguridad** empieza en los requisitos:

**CLIENTE:** “Necesitamos identificar a los usuarios, porque tienen distintos permisos”

**INGENIERO/A:** “Estos van a ser muchos sub-requisitos”

## REQ 1 (Reformulado)

Los usuarios de la aplicación tienen que autenticarse antes de entrar por medio de un formulario que les solicitará email/contraseña para poder efectuar cualquier acción.

## REQ 1.1

Los usuarios de la aplicación tienen que poder solicitar un cambio de contraseña si se les olvida. Se les enviará un enlace al correo que les redirigirá a un formulario para introducir la nueva contraseña.

## REQ 1.2

Toda comunicación entre cliente servidor usará un protocolo seguro.

## REQ 1.3

Las contraseñas de los usuarios NO se almacenarán en crudo en el sistema. Se guardan cifradas.



## REQ 1.4

El tiempo de respuesta para la autenticación del usuario tiene que ser menor o igual a 5 segundos.

# API REST

## Introducción

# Cambio de paradigma

- Las API REST no son ya soluciones de “startup”
- El futuro es interoperación, y esto solo es posible con estándares sencillos

# Basadas en HTTP

```
GET /doc/test.html HTTP/1.1
```

```
Host: www.test101.com
```

```
Accept: image/gif, image/jpeg, */*
```

```
Accept-Language: en-us
```

```
Accept-Encoding: gzip, deflate
```

```
User-Agent: Mozilla/4.0
```

```
Content-Length: 35
```

```
bookId=12345&author=Tan+Ah+Teck
```

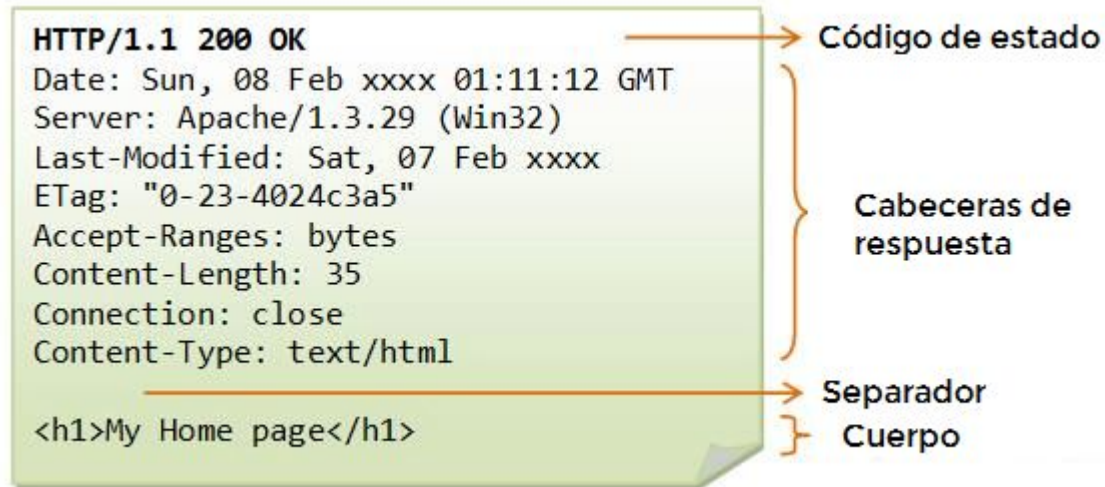
Petición

Cabeceras

Línea en blanco (\r\n)

Cuerpo

# Basadas en HTTP



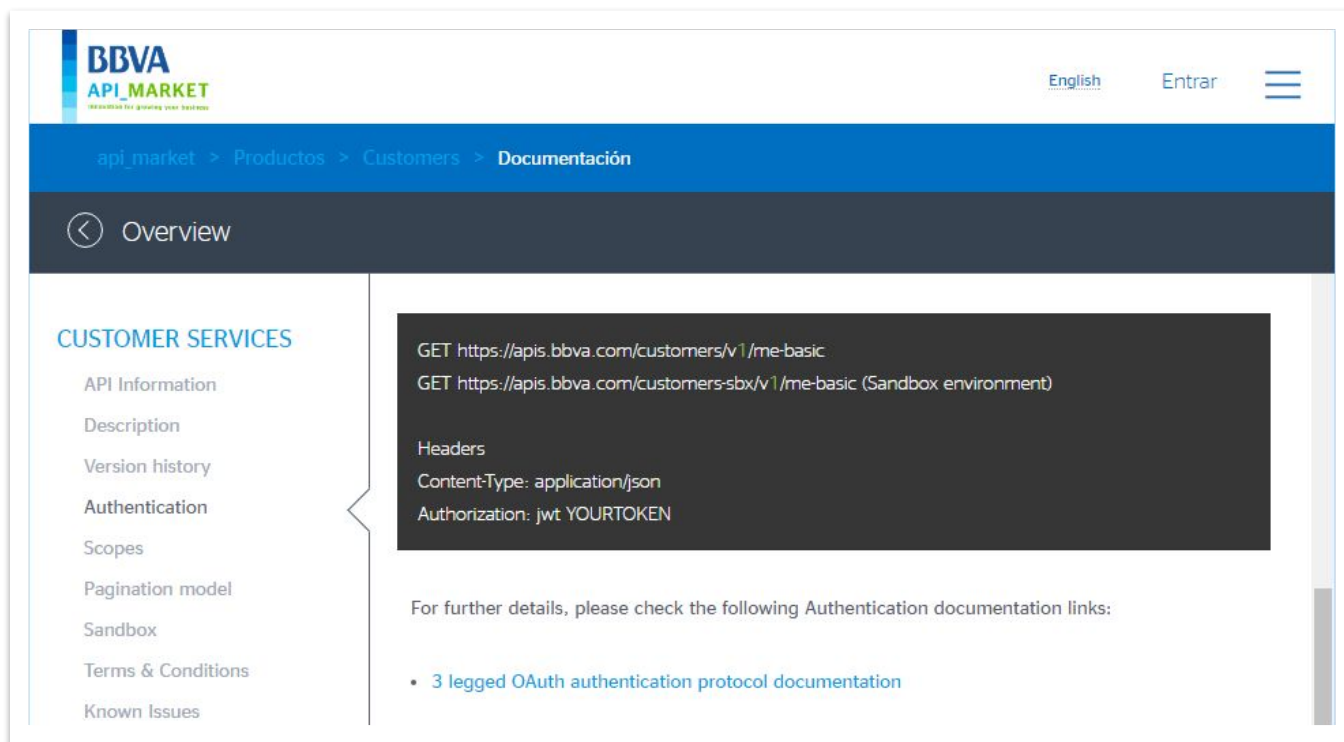
# Características

Propiedad	Descripción
Sin estado	Cada petición contiene toda la información necesaria para su ejecución: <ul style="list-style-type: none"><li>• Algunas peticiones se pueden cachear</li><li>• ¿Cómo mantenemos el estado?</li><li>• ¿Y la autenticación y/o autorización?</li></ul>
Verbos estándar	POST (crear), GET (leer), PUT (editar) y DELETE (eliminar)
Separación cliente/servidor	Separación entre interfaz y almacenamiento de datos en el servidor: mejora portabilidad, aumenta escalabilidad y permite desarrollo paralelo de diferentes componentes

# Características

Propiedad	Descripción
Sin estado	Cada petición contiene toda la información necesaria para su ejecución: <ul style="list-style-type: none"><li>• Algunas peticiones se pueden cachear</li><li>• ¿Cómo mantenemos el estado?</li><li>• ¿Y la autenticación y/o autorización?</li></ul>
Verbos estándar	POST (crear), GET (leer), PUT (editar) y DELETE (eliminar)
Separación cliente/servidor	Separación entre interfaz y almacenamiento de datos en el servidor: mejora portabilidad, aumenta escalabilidad y permite desarrollo el paralelo de diferentes componentes en diferentes lenguajes

# Ejemplo: el futuro de la banca



The screenshot shows the BBVA API Market website. The header includes the BBVA logo, 'API MARKET' tagline, language selector (English), and a login button (Entrar). A breadcrumb trail reads 'api\_market > Productos > Customers > Documentación'. Below this is a dark blue bar with a back arrow and the word 'Overview'. The left sidebar, titled 'CUSTOMER SERVICES', lists various links: API Information, Description, Version history, Authentication, Scopes, Pagination model, Sandbox, Terms & Conditions, and Known Issues. The main content area displays two GET requests: 'GET https://apis.bbva.com/customers/v1/me-basic' and 'GET https://apis.bbva.com/customers-sbx/v1/me-basic (Sandbox environment)'. It also shows headers: 'Content-Type: application/json' and 'Authorization: jwt YOURTOKEN'. At the bottom, it provides a link to '3 legged OAuth authentication protocol documentation'.

BBVA  
API MARKET  
We're here for guiding your business

English Entrar

api\_market > Productos > Customers > Documentación

< Overview

**CUSTOMER SERVICES**

- API Information
- Description
- Version history
- Authentication
- Scopes
- Pagination model
- Sandbox
- Terms & Conditions
- Known Issues

GET `https://apis.bbva.com/customers/v1/me-basic`  
GET `https://apis.bbva.com/customers-sbx/v1/me-basic` (Sandbox environment)

**Headers**

Content-Type: `application/json`  
Authorization: `jwt YOURTOKEN`

For further details, please check the following Authentication documentation links:

- [3 legged OAuth authentication protocol documentation](#)



# Cambio de paradigma

	Tradicional	Nuevo
Transporte	HTTP	HTTP
Datos	XML	JSON
Autenticación	Básica, X509, Kerberos, SAML	OAuth, JWT (!)
Confidencialidad e integridad	WS-Security	TLS

API REST

# Simplicidad

## SOAP + WS-\*

Complejo

Muy estandarizado

XML

## API REST

Sencillas

“Informales”

JSON