Materia:

DISEÑO ELECTRÓNICO BASADO EN SISTEMAS EMBEBIDOS

Alumno:

Posadas Pérez Isaac Sayeg

Paniagua Rico Juan Julian

García Azzúa Jorge Roberto

Grado y grupo:

8°G

Profesor:

Garcia Ruiz Alejandro Humberto

Unidad 4 -Tarea 8:

API

API

Introducción

En el mundo del desarrollo de software, la capacidad de los sistemas para comunicarse e interactuar entre sí es fundamental. Para lograrlo, se utilizan las **APIs** (Application Programming Interfaces, o Interfaces de Programación de Aplicaciones). Una API es un conjunto de normas, protocolos y herramientas que permiten que distintos programas o componentes de software intercambien información y funcionalidades de forma estructurada y controlada.

Las APIs han revolucionado la forma en que se diseñan, implementan y extienden las aplicaciones modernas, especialmente en la era de la computación en la nube, la arquitectura basada en microservicios y el Internet de las Cosas (IoT).

Desarrollo

Una API actúa como un intermediario entre dos aplicaciones o sistemas, definiendo claramente cómo se pueden solicitar y enviar datos o comandos. Proporciona un conjunto de reglas y especificaciones que describen las operaciones disponibles, los formatos de datos permitidos y los protocolos de comunicación.

Tipos de APIs más comunes:

- **APIs RESTful:** Basadas en el estilo arquitectónico REST, utilizan HTTP y métodos como GET, POST, PUT y DELETE para operar sobre recursos identificados por URLs. Son ligeras, fáciles de usar y ampliamente adoptadas en servicios web y microservicios.

- **APIs SOAP:** Usan un protocolo estándar basado en XML para intercambiar mensajes estructurados. Son más rígidas y ofrecen características como seguridad y transacciones integradas, pero suelen ser más complejas que REST.

- **APIs de bibliotecas o frameworks:** Son conjuntos de funciones y métodos que permiten a los desarrolladores utilizar funcionalidades preconstruidas sin preocuparse por su implementación interna.

- **APIs de hardware:** Permiten que el software controle componentes físicos, como sensores, cámaras o actuadores, facilitando la interacción con

dispositivos embebidos.

Elementos clave de una API:

- **Endpoints:** Puntos de acceso donde se realizan las solicitudes.
- **Métodos:** Tipos de operaciones permitidas (leer, crear, actualizar, eliminar).
- **Formatos de datos:** JSON, XML u otros, para estructurar la información intercambiada.
- **Autenticación y autorización:** Mecanismos para controlar quién puede acceder a la API (tokens, OAuth).
- **Documentación:** Guías y especificaciones para que los desarrolladores puedan usarla correctamente.

Las APIs facilitan la interoperabilidad, permitiendo que aplicaciones heterogéneas, desarrolladas en diferentes lenguajes o plataformas, colaboren fácilmente.

Ejemplo:

Supongamos que una empresa desea integrar un sistema de pagos en su tienda en línea. En lugar de desarrollar un sistema completo para procesar tarjetas de crédito, decide usar la API de un proveedor externo de pagos (como Stripe o PayPal).

- La aplicación de la tienda envía una solicitud a la API del proveedor, enviando detalles como el monto a cobrar y los datos del cliente.
- La API procesa la transacción y devuelve una respuesta indicando si el pago fue aprobado o rechazado.
- La tienda, según la respuesta, continúa con la confirmación del pedido o muestra un error.

Este ejemplo ilustra cómo las APIs permiten aprovechar servicios especializados sin tener que reinventar la rueda, además de garantizar seguridad y cumplimiento normativo.

Conclusión

Las APIs son pilares fundamentales en el desarrollo de software actual, facilitando la creación de sistemas modulares, escalables y conectados. Su correcta implementación y documentación aseguran que aplicaciones diversas puedan comunicarse de forma eficiente y segura, habilitando ecosistemas tecnológicos flexibles. Además, con la adopción creciente de arquitecturas distribuidas, microservicios e IoT, las APIs se han convertido en la columna vertebral que soporta la innovación y la integración tecnológica.

Bibliografía

1. Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures* (Doctoral dissertation). University of California, Irvine.

2. Richardson, L., & Ruby, S. (2008). *RESTful Web Services*. O'Reilly Media.

3. Pautasso, C., Zimmermann, O., & Leymann, F. (2017). *RESTful Web Services vs. "Big" Web Services: Making the Right Architectural Decision*. ACM Transactions on Internet Technology.

4. O'Reilly Media. (2019). *API Design Patterns and Best Practices*.

5. Postman. (2023). *What is an API?* postman.com.