**Comparación de Seis Arquitecturas de Software**

# **Introducción**

La arquitectura de software define la estructura y el comportamiento de una aplicación. Existen diversos enfoques arquitectónicos, cada uno con ventajas y desventajas según el contexto en el que se apliquen. A continuación, se presentan seis arquitecturas comunes en el desarrollo de software.

# **1. Arquitectura Monolítica**

## **Definición**

La arquitectura monolítica es un modelo tradicional en el que todos los componentes de la aplicación están integrados en una única estructura de código.

## **Ventajas**

* Desarrollo y despliegue sencillos.
* Menos sobrecarga operativa.
* Comunicación interna rápida.

## **Desventajas**

* Dificultad para escalar de manera independiente.
* Mantenimiento complicado en proyectos grandes.
* Un fallo puede afectar a toda la aplicación.

# **2. Arquitectura de Microservicios**

## **Definición**

Se basa en dividir la aplicación en servicios independientes que se comunican entre sí mediante APIs.

## **Ventajas**

* Escalabilidad independiente de los servicios.
* Mayor tolerancia a fallos.
* Flexibilidad tecnológica.

## **Desventajas**

* Complejidad en la gestión y despliegue.
* Mayor necesidad de comunicación entre módulos.
* Costos operativos elevados.

# **3. Arquitectura en Capas**

## **Definición**

Organiza la aplicación en capas lógicas (por ejemplo, presentación, negocio, acceso a datos) para modularidad y mantenimiento.

## **Ventajas**

* Facilita el mantenimiento y la evolución.
* Reutilización de componentes.
* Separación clara de responsabilidades.

## **Desventajas**

* Puede generar dependencias innecesarias.
* Latencia entre capas.
* Complejidad en aplicaciones grandes.

# **4. Arquitectura Basada en Eventos**

## **Definición**

Los componentes se comunican mediante eventos, en lugar de llamadas directas, permitiendo mayor flexibilidad y asincronía.

## **Ventajas**

* Alta escalabilidad y flexibilidad.
* Desacoplamiento de componentes.
* Manejo eficiente de eventos en tiempo real.

## **Desventajas**

* Mayor complejidad en la gestión de eventos.
* Dificultad para rastrear flujos de datos.
* Depuración más complicada.

# **5. Arquitectura Orientada a Servicios (SOA)**

## **Definición**

Los sistemas se dividen en servicios reutilizables que interactúan a través de protocolos estandarizados.

## **Ventajas**

* Integración con otros sistemas.
* Modularidad y reutilización de servicios.
* Mejora la escalabilidad y flexibilidad.

## **Desventajas**

* Complejidad en la gestión de servicios.
* Requiere una infraestructura robusta.
* Overhead en la comunicación entre servicios.

# **6. Arquitectura de Servidor sin Estado (Serverless)**

## **Definición**

Las aplicaciones se ejecutan en entornos en la nube sin necesidad de gestionar servidores, respondiendo a eventos bajo demanda.

## **Ventajas**

* Escalabilidad automática.
* Costos reducidos por uso bajo demanda.
* Reducción de gestión de infraestructura.

## **Desventajas**

* Dependencia de proveedores en la nube.
* Latencia en la ejecución inicial.
* Restricciones en tiempo de ejecución y almacenamiento.

# **Conclusión**

Cada tipo de arquitectura tiene sus propias fortalezas y debilidades. La elección depende de factores como escalabilidad, mantenimiento, costos y requerimientos específicos del proyecto.

# **Fuentes Consultadas**

1. https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith
2. <https://click-it.es/arquitectura-monolitica-vs-arquitectura-de-microservicios-cual-debo-elegir/>
3. <https://medium.com/hengky-sanjaya-blog/monolith-vs-microservices-b3953650dfd>
4. https://martinfowler.com/articles/serverless.html
5. <https://aws.amazon.com/serverless/>
6. https://www.redhat.com/en/topics/cloud-native-apps/what-is-event-driven-architecture