GUIA PARA ELEGIR LA ARQUITECTURA DE UNA APLICACION

V.I. Quino Quino
7690-17-6433 Universidad Mariano Gálvez
Seminario de Tecnologías de Información
vquinoq@miumg.edu.gt

Resumen
Elegir la arquitectura de una aplicación es un proceso complicado y crítico que puede impactar su rendimiento, escalabilidad y mantenibilidad, por lo que no solo se trata de elegir una arquitectura popular. En este artículo trataremos de analizar los principales criterios que se debe considerar antes de seleccionar una arquitectura, el tipo de aplicación a desarrollar (web, móvil, escritorio o embebida, etc.), la cantidad de usuarios que la utilizarán, la escalabilidad, los recursos con que se cuenta y tampoco olvidar el tiempo que se tiene para entregar el software. Se exploran las arquitecturas más utilizadas, como monolítica, microservicios y basada en eventos, destacando sus ventajas y sus limitaciones describiendo casos prácticos que asocian tipos de aplicaciones con arquitecturas recomendadas. El objetivo es proporcionar una guía clara para los desarrolladores, los resultados muestran que no existe una arquitectura universal; por la cual la decisión debe basarse en un análisis detallado de las necesidades específicas de la aplicación.

Palabras claves: arquitectura de software, escalabilidad, microservicios, tipo de aplicación

Desarrollo del tema
La arquitectura de software es como el plano que guía a los desarrolladores a como construir el sistema, define como los componentes interactúan entre sí y que cumplan con los requisitos del sistema. Elegir la arquitectura adecuada no se simplemente al azar o se elige el más común, sino que se debe elegir en base a las necesidades de la aplicación que se va a desarrollar y también el tipo de aplicación (web, móvil, escritorio, etc.), esto implica que se debe realizar un análisis de requerimientos exhaustivo, evaluar cuales van hacer los requerimientos funcionales (que describen a detalle lo que la aplicación debe hacer) y los no funcionales (factores como rendimiento, seguridad y escalabilidad).

Tipos de aplicaciones y sus necesidades
Dependiendo del tipo de aplicación a desarrollar se tomará las decisiones para elegir la arquitectura, ya que cada tipo de aplicación tiene características únicas que influyen en la elección de la arquitectura.

Aplicaciones web: estas aplicaciones se ejecutan en un navegador y por lo general requieren alta disponibilidad, escalabilidad y accesibilidad multiplataforma. Un ejemplo de esto podría ser un sistema para realizar pagos de servicios, que en días específicos se llegue a manejar mucho tráfico.

Aplicaciones móviles: estas aplicaciones requieren que se optimicen los recursos, soporte offline y actualizaciones frecuentes para diferentes sistemas operativos. Como por ejemplo una aplicación de mensajería requiere sincronización y actualización con frecuencia.

Aplicaciones de escritorio: estas por lo regular se ejecutan localmente y se le da prioridad al rendimiento y la experiencia del usuario en un entorno controlado. Un ejemplo de esto podría ser un software diseño grafico que requiere un procesamiento rápido de gráficos. Las arquitecturas monolíticas o modulares se consideran los ideales, ya permiten un desarrollo sencillo y un acceso de forma eficiente a los recursos locales, la desventaja que podría presentar es que es menos escalables para sistemas distribuidos.

Arquitecturas comunes y su aplicación

Arquitectura monolítica: combina todos los componentes en una única unidad. Toda la lógica y funcionalidades están en un solo bloque. Es fácil de desarrollar y probar, ideal para aplicaciones pequeñas, prototipos rápidos o equipos con experiencia limitada. Sin embargo, puede ser difícil de escalar y mantener en sistemas grandes.
Microservicios: divide la aplicación en módulos independientes que se comunican entre sí mediante APIs. Es ideal su uso para aplicaciones grandes y con alta demanda de escalabilidad, que requieren actualizaciones frecuentes y equipo de desarrollo distribuidos. También son ideales para aplicaciones en la nube y para aquellas que se benefician de la independencia de componentes y la reutilización de código.
Arquitectura en capas: también conocida como multicapa, es un enfoque de diseño de software que divide una aplicación en diferentes niveles o capas, cada una con una función en especifica. Estas capas interactúan entre sí, permitiendo una estructura organizada y modular. Capas comunes como presentación, aplicación, lógica del negocio, acceso a datos y base de datos. Se utiliza principalmente en entornos empresariales por su claridad y facilidad de mantenimiento.
Serverless: permite ejecutar funciones sin gestionar servidores, ejecuta funciones bajo demanda en la nube, ideal para aplicaciones web con cargas variables, como sistemas de reservas. Reduce costos operativos, pero puede limitar el control sobre el entorno de ejecución.
Arquitectura basada en eventos: utiliza eventos para ejecutar acciones, esta arquitectura es adecuada para aplicaciones móviles y embebidas que manejan flujos de datos asíncronos, como sistemas IoT. Por ejemplo, un sistema de monitoreo de sensores puede usar Apache Kafka para procesar eventos en tiempo real.
Criterios sugeridos para la elección
Escalabilidad: si se espera un alto nivel de usuarios o alta carga, las arquitecturas basadas en microservicios o serveless permiten escalar componentes de forma independiente, esto garantiza un rendimiento estable.
Mantenibilidad: las arquitecturas modulares y de microservicios, estas arquitecturas permiten actualizar, probar o desplegar componentes individuales sin afectar al sistema completo, reduciendo riegos de errores y mejora la agilidad del desarrollo, esto es esencial para el ciclo de vida del software.
Rendimiento: en entornos locales o aplicaciones con menor carga, las arquitecturas monolíticas pueden ofrecer mejor desempeño debido a una menor latencia en la comunicación interna, ofreciendo tiempos de respuesta más rápidos en entornos locales o sistemas con pocos recursos.
Seguridad: todas las arquitecturas sin excepción alguna deben implementar medidas de seguridad robustas, como cifrado, autenticación y control de acceso. En el caso de los entornos distribuidos estas medidas deben aplicarse a cada componente de forma individual.
Costos: el presupuesto influye en la elección arquitectónica. Las soluciones serverless y monolíticas suelen ser rentables para proyectos pequeños o poca carga, reduciendo los costos operativos y la complejidad de despliegue, contrario a lo que son los microservicios requieren una fuerte inversión inicial en infraestructura, monitoreo y orquestación, al final es una buena inversión, ya que ofrecen beneficios a largo plazo en temas de escalabilidad.

Observaciones y comentarios
La elección de la arquitectura no es una decisión estática, implica un balance entre simplicidad y escalabilidad; debe evolucionar con los requisitos y el contexto de la aplicación. Los desarrolladores deben considerar la curva de aprendizaje de arquitecturas complejas, como los microservicios, y los costos asociados a herramientas de orquestación. Además, la falta de estandarización en algunos

entornos, como serverless, puede limitar la portabilidad.

Conclusiones

1. La elección de la arquitectura de una aplicación es un proceso que requiere un análisis previo y minuciosamente para que impacte en su rendimiento, mantenibilidad y adaptación a los cambios del mercado.
2. Tener en cuenta que no existe una arquitectura universal que sirva para todos los proyectos; cada tipo de aplicación posee sus necesidades únicas.

Bibliografía

- Ardalis. (n.d.). Arquitecturas de aplicaciones web comunes - .NET. Microsoft Learn. https://learn.microsoft.com/es-es/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures
- Schmidt, R. (2023, August 18). Cómo elegir la arquitectura de software adecuada para su proyecto. AppMaster - Ultimate All-in No-code Platform. https://appmaster.io/es/blog/como-elegir-la-arquitectura-de-software
- Torres, P. B. J. (2024, March 19). Patrones de Arquitectura de Software: cómo elegir una estructura escalable. https://www.st-computacion.com/2024/03/19/patrones-de-arquitectura-de-software-como-elegir-una-estructura-escalable/