**UNIVERSIDAD LA SALLE**

**CARRERA DE INGENIERÍA DE SISTEMAS**



**INVESTIGACIÓN: ARQUITECTURAS DE SOFTWARE**

**Nombre:**

Rosalía Reynaga Funes

La Paz - Bolivia

2023

**Definición**

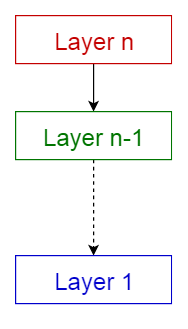
La arquitectura de software es una disciplina fundamental en la ingeniería de software que se enfoca en la concepción y diseño estructural de sistemas de software. En términos simples, se trata de la planificación y organización de un sistema informático a nivel de alto nivel, definiendo la estructura de sus componentes, sus interacciones, y los principios que guiarán su construcción y evolución a lo largo del tiempo. Esta disciplina busca equilibrar múltiples factores, como la eficiencia, la escalabilidad, la seguridad, la mantenibilidad y la modularidad, para garantizar que el sistema cumpla con los requisitos funcionales y no funcionales establecidos.

De acuerdo con el Software Engineering Institute (SEI), la Arquitectura de Software se refiere a “las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos.”

**Arquitecturas mas usadas en la actualidad**

1. **En capas:** La arquitectura en capas es un enfoque en el que una aplicación se divide en varias capas lógicas, cada una responsable de una funcionalidad específica. Las capas están ordenadas jerárquicamente y se comunican entre sí a través de interfaces bien definidas. Está arquitectura permite una mayor modularidad y facilita la separación de preocupaciones. Cada capa se puede desarrollar y mantener de forma independiente, lo que facilita las actualizaciones y mejoras en la aplicación. Sin embargo, puede generar un mayor acoplamiento entre las capas y puede ser menos eficiente en términos de rendimiento debido a las comunicaciones entre ellas.

Las 4 capas más comúnmente encontradas de un sistema de información general son las siguientes.

* Capa de presentación (también conocida como capa UI )
* Capa de aplicación (también conocida como capa de servicio )
* Capa de lógica de negocios (también conocida como capa de dominio )
* Capa de acceso a datos (también conocida como capa de persistencia)

Suele utilizarse en aplicaciones de escritorio generales o aplicaciones web de comercio electrónico.

1. **Arquitectura sin servidor (serverless):** Es un enfoque en el que el proveedor de servicios en la nube se encarga de gestionar la infraestructura necesaria para ejecutar y escalar las aplicaciones. Esto libera a los desarrolladores de la gestión de los servidores y les permite centrarse en la lógica de negocio de la aplicación.

Esta arquitectura ofrece una alta escalabilidad, ya que los recursos se asignan dinámicamente según la demanda. Sin embargo, puede tener limitaciones en cuanto a la latencia y el tiempo de ejecución, y puede generar dependencia del proveedor de servicios en la nube.

**Ventajas:** Alta escalabilidad con asignación dinámica de recursos y liberación de la gestión de servidores.

**Desventajas:** Limitaciones en la latencia y el tiempo de ejecución y dependencia del proveedor de servicios en la nube.

1. **Arquitectura de microservicios:** Esta arquitectura una aplicación se divide en un conjunto de servicios pequeños e independientes, cada uno enfocado en una funcionalidad específica. Estos servicios se comunican entre sí a través de interfaces bien definidas, generalmente utilizando protocolos HTTP.

Cada microservicio tiene su propia responsabilidad y los equipos pueden desarrollarlos independientemente de otros microservicios. La única dependencia entre ellos es la comunicación.

Esta arquitectura permite una mayor modularidad y escalabilidad, ya que cada servicio puede ser desarrollado, desplegado y escalado de forma independiente. Sin embargo, también implica una mayor complejidad en la gestión de los servicios y las interacciones entre ellos.

Suele usar en Sitios web con pequeños componentes, centros de datos corporativos con límites bien definidos.

1. **Arquitectura orientada a servicios (SOA):** Permite la creación de aplicaciones a través de la integración de diferentes servicios independientes entre sí. Estos servicios se comunican a través de interfaces bien definidas utilizando protocolos estándar, lo que facilita la interoperabilidad y la reutilización de componentes.

La ventaja de esta arquitectura es su capacidad para adaptarse a cambios en los requerimientos de negocio y permitir la integración de sistemas heterogéneos. Sin embargo, puede haber un mayor acoplamiento debido a la dependencia de los servicios externos y puede requerir un esfuerzo adicional en la gestión del ciclo de vida de los servicios.

1. **Arquitectura de microkernel:** Es un enfoque de diseño que se utiliza en la construcción de sistemas de software, especialmente sistemas operativos, donde la funcionalidad principal se mantiene lo más pequeña y simple posible, mientras que las características adicionales se implementan como módulos separados que se ejecutan fuera del núcleo del sistema. Sus componentes principales son:

**Microkernel:** El microkernel es el núcleo mínimo del sistema operativo que proporciona servicios básicos como la gestión de la memoria, la programación de tareas y la comunicación entre procesos. Este núcleo es pequeño, rápido y esencialmente responsable de la gestión de recursos.

**Servicios del Sistema:** Las funciones que normalmente se consideran parte del núcleo en sistemas operativos monolíticos, como la gestión de archivos, la administración de dispositivos y las redes, se implementan como servicios independientes fuera del microkernel. Estos servicios se ejecutan en espacio de usuario y pueden comunicarse con otros a través de mecanismos de IPC (Inter-Process Communication).

Se utilizan en sistemas operativos, sistemas embebidos críticos, istemas distribuidos, sistemas de tiempo real,sistemas de seguridad y virtualización.

1. **Arquitectura basada en eventos (Event-based architecture o EBA):** Se centra en la comunicación y coordinación de componentes de software a través de eventos. En esta arquitectura, los eventos son cambios de estado o acontecimientos significativos que ocurren en el sistema y que desencadenan acciones o respuestas específicas por parte de los componentes. Estos son sus principales componentes

**Productores de Eventos**: Estos componentes generan eventos cuando ocurren cambios o situaciones relevantes en el sistema. Los productores de eventos publican estos eventos en un canal o un bus de eventos.

**Canal o Bus de Eventos**: Es el mecanismo de comunicación central donde se publican los eventos. Los eventos se transmiten a través de este canal para que otros componentes interesados puedan suscribirse y recibir notificaciones.

**Consumidores de Eventos**: Estos componentes se suscriben a eventos específicos que son de interés para ellos. Cuando un evento se publica en el canal de eventos, los consumidores relevantes lo reciben y pueden tomar acciones basadas en ese evento.

Se utiliza en sistemas de monitorización en tiempo real, aplicaciones de comercio electrónico, aplicaciones de internet de las cosas (IoT),

sistemas de procesamiento de datos en tiempo real, aplicaciones de mensajería y notificaciones, sistemas de automatización industrial,

aplicaciones de juegos en línea y multimedia, aplicaciones de finanzas,

sistemas de control de tráfico y logística