Trabajo Práctico 1

Parte 1

Objetivo

Comprender y diferenciar los conceptos fundamentales de arquitectura de software, incluyendo estilos, patrones, vistas y aspectos transversales, y analizar cómo se relacionan con los atributos de calidad

Actividad 3.1

Arquitectura de Software vs Diseño de Software

La **arquitectura de software** se centra en la visión global del sistema, definiendo su estructura de alto nivel, los componentes principales y la forma en que estos se relacionan para garantizar escalabilidad, seguridad y mantenimiento a largo plazo. En cambio, el **diseño de software** aborda un nivel más detallado, actuando como un plan que traduce esa visión arquitectónica en soluciones concretas, asegurando que cada módulo cumpla su función de manera eficiente y coherente con los objetivos del sistema. Mientras la arquitectura responde a la pregunta de "qué forma tendrá el sistema y cómo crecerá, el diseño se enfoca en cómo se construyen las piezas y cómo trabajan juntas en la práctica".

El diseño arquitectural es el proceso de identificar subsistemas que componen un sistema y establecer un marco para su control y comunicación. Es la manera en que resolvemos las estructuras de código dentro del programa de software.

Estilos arquitectónicos

Un estilo arquitectónico es un conjunto de principios que proporciona un marco abstracto para entender familias de soluciones o sistemas. Los estilos dictan el vocabulario de los componentes y conectores que pueden usarse en las instancias de cada estilo definido, así como las restricciones y la forma en que se pueden combinar.
El propósito es definir un lenguaje de diseño, que puede ser visto como plantillas de alto nivel que influyen en las decisiones de diseño.

Patrones arquitectónicos

Los patrones de arquitectura del software son soluciones concretas a problemas comunes en el diseño del software a nivel de arquitectura. Son descripciones o plantillas, para resolver un problema que puede usarse en diferentes situaciones.
Los patrones ofrecen una solución a un problema dentro de un contexto particular, incluyendo las consideraciones que afectan a la decisión de diseño.

Vistas de arquitectura (Modelos: 4+1; Siemens; SEI; C4)

Las vistas de arquitectura son representaciones estructuradas que facilitan la compresión y la comunicación de la arquitectura de una solución. Todos los métodos y técnicas actuales para crear y documentar arquitecturas se basan en vistas relevantes.

El diseño y la estructura de un software se representan mediante diferentes planos
y vistas.

Existen varios modelos de vistas, como:

- **4+1**
  - **Consta de 5 vistas:** lógica, despliegue, física, de procesos, de escenarios o casos de uso
  - Proporciona una estructura integral y comprensible para documentar y comunicar la arquitectura de software.
  - 

  - **Siemens:** Es otra propuesta relevante en la arquitectura de software, se enfoca en diferentes aspectos de la arquitectura tanto del software como del hardware. TIene vistas: **conceptual, módulos, código, ejecución**
- **SEI**
  - Este modelo comprende de viewtypes: módulos - componentes y conectores - asignación o alojamiento.
- **C4**
  - Es simple y trazable entre las distintas vistas posibles.
  - Tenemos diagramas de contextos

Aspectos transversales (seguridad; gestión operativa; comunicación)

Funcionalidades o requisitos que no pertenecen únicamente a un módulo específico, sino que atraviesan varias partes del sistema y afectan a múltiples componentes.

Se dividen en 3 areas:

- Seguridad:
    - Autenticación: El objetivo es asegurar que sólo los usuarios que están autorizados pueden acceder al sistema.
    - Autorización: Los usuarios tienen ciertas autorizaciones para determinadas funciones. Es decir, además de acceder a la aplicación, este acceso permite el uso de sólo algunas acciones dependiendo justamente de la autorización pertinente.
    - Auditoría: Rastreo y trazabilidad de las acciones que realizan los usuarios.
- Gestión Operativa:
    - Gestión de roles y perfiles: Todo tiene una gestión centralizada a través de perfiles y roles que permiten administrar las configuraciones de la seguridad a nivel de usuario y asegurando que cada uno tenga la experiencia y los permisos que correspondan
    - Gestión de excepciones: Identificación o el aseguramiento de que al aparecer los errores se manejan de una manera eficiente
- Comunicación:
    - Instrumentación
    - Interoperabilidad
    - Mailing y notificaciones

Actividad 3.2

¿Por qué se dice que los atributos de calidad son los que definen la arquitectura de software?

Podemos decir que los atributos de calidad definen la arquitectura de un sistema influye directamente en la performance, seguridad, disponibilidad y mantenibilidad. Constantemente vemos una relación, ya que podemos ver que los conflictos y los beneficios van de la mano de los atributos de calidad, es decir, podemos obtener mejor rendimiento en la arquitectura pero a costa de reducir disponibilidad.

¿Cómo se relacionan los escenarios de calidad, las tácticas de diseño y los patrones de arquitectura?

Los escenarios de calidad describen situaciones específicas que ponen a prueba un atributo de calidad, las tacticas de diseño son decisiones concretas de los arquitectos para lograr mejorar un atributo de calidad en esos escenarios, y los patrones de arquitectura son soluciones más generales y probadas que encapsulan varias tácticas y permiten alcanzar atributos de calidad de manera más estructurada.

¿Qué diferencias existen entre estilos arquitectónicos y patrones arquitectónicos? ¿Y entre patrones arquitectónicos y patrones de diseño?

Entre estilos arquitectónicos y patrones arquitectónicos la diferencia se da en que los estilos arquitectónicos representan una visión más general y abstracta de la arquitectura de sistema, mientras que los patrones arquitectónicos ofrecen soluciones recurrentes a problemas específicos dentro de esa arquitectura global.

¿Qué ventajas aporta documentar una arquitectura usando un modelo de vistas?

Facilitan la comprensión y la comunicación de la arquitectura de una solución, permite que todos los involucrados, cada uno con su perspectiva particular, puedan entender y evaluar si la visión cubre sus expectativas, preocupaciones e intereses.

¿Qué diferencias encuentran entre el modelo 4+1 y el modelo C4?

El modelo 4+1 está pensado como una manera de documentar la arquitectura desde múltiples perspectivas para distintos interesados, mientras que el modelo C4 propone una visualización estructurada y jerárquica que facilita la comunicación y comprensión progresiva del sistema.

¿Por qué los aspectos transversales (ej. autenticación, interoperabilidad) son críticos para la robustez de un sistema?

Son críticos ya que impactan en todo el sistema, no se limitan a un módulo puntual, sino que impactan en la experiencia global.
Definen la robustez ya que si un sistema falla en seguridad, interoperabilidad o monitoreo, no será confiable en producción.
Además dificultan el mantenimiento debido a que su incorporación es costosa y propensa a errores, por lo que se las debe considerar desde el diseño.

Actividad 3.3

1. Estilo arquitectónico Cliente-Servidor

- **Características**: separación entre cliente (solicita servicios) y servidor (los provee). Comunicación mediante red.

- **Cuándo usarlo**: aplicaciones distribuidas tradicionales (ej.: web apps, correo electrónico).

2. Patrón arquitectónico en Capas

- **Características**: organización jerárquica del sistema (presentación, negocio, datos). Cada capa se apoya en la inferior.

- **Cuándo usarlo**: sistemas grandes donde se prioriza mantenibilidad y separación de responsabilidades (ej.: software empresarial).

3. Patrón arquitectónico Microservicios

- **Características**: división del sistema en servicios pequeños e independientes, que se comunican por APIs. Escalables y desplegables por separado.

- **Cuándo usarlo**: sistemas complejos y de alta demanda que requieren escalabilidad y despliegues continuos (ej.: plataformas tipo Netflix, Amazon).

4. Patrón de diseño MVC (Modelo–Vista–Controlador)

- **Características**: separación en tres componentes: Modelo (datos/lógica), Vista (interfaz), Controlador (coordina interacción).

- **Cuándo usarlo**: aplicaciones con interfaces de usuario, donde se busca independencia entre lógica y presentación (ej.: frameworks web).

5. Vista de arquitectura Modelo 4+1

- **Características**: describe la arquitectura desde 5 perspectivas: lógica, procesos, desarrollo, física y escenarios (+1).

- **Cuándo usarlo**: documentación y comunicación de arquitecturas complejas, asegurando que se contemplen aspectos técnicos y funcionales.

Conclusión

El concepto que nos resultó más desafiante fue el de vistas de arquitectura, porque implica comprender cómo un mismo sistema puede representarse desde diferentes perspectivas y niveles de abstracción. A diferencia de los estilos o patrones, que son más concretos y fáciles de ejemplificar, las vistas requieren integrar la mirada técnica con la comunicacional y la organizativa.