Diseño y Pruebas II

WIS ARCHITECTURE REPORT

Grupo: C1.02.07
Miembros:

- Javier Nunes Ruiz - **javnunrui@alum.us.es**
- Manuel Palacios Pineda - **manpalpin@alum.us.es**
- Manuel Carnero Vergel - **correo corporativo**
- Pablo Martínez Valladares - **pabmarval@alum.us.e**
- Julio Navarro Rodríguez - **julnavrod@alum.us.es**

Repositorio: https://github.com/DP2-C1-02-07/Acme-L3

1. Índice

2. Tabla de versiones

Versión	Fecha	Descripción
1.0.0	11/02/2023	Documento creado y completadas las secciones 3 y 4

3. Resumen del documento

Somos el grupo C1.02.07 de la asignatura DP2 de la universidad de Sevilla, en el siguiente informe expondremos nuestro conocimiento sobre tipos de arquitectura de sistemas de información así como qué información es actualmente conocida por el grupo acerca de la arquitectura a usar en el proyecto de la asignatura.

4. Introducción

A lo largo de nuestro aprendizaje en la carrera hemos cursado varias asignaturas en las cuales se ha hablado de distintas formas sobre arquitecturas software en los sistemas de información. Desde ISSII donde se mencionan de una forma más general hasta AISS o DP1 donde se abordaron de una forma más exhaustiva.

Dadas las circunstancias del proyecto de DP1, probablemente el modelo más conocido y dominado por el grupo es el MVC, Modelo Vista Controlador, el cual expondremos más adelante detalladamente y el cual coincide nuevamente con la arquitectura de este nuevo proyecto.

5. Contenidos

5.1. Arquitecturas y patrones

En primer lugar, entendemos por arquitectura software una definición general de las relaciones entre los grandes elementos software, así como restricciones en un marco de alto nivel de aplicación, usados al igual que los patrones arquitectónicos para resolver problemas de características determinadas (de contexto y solución generalmente comunes) que ya han sido aplacados de forma anterior y efectiva con una solución que quedaría recogida como arquitectura.

Así pues, existen arquitecturas y patrones arquitectónicos especializados en resolver determinados tipos de problemas así como hay arquitecturas incompatibles con otros marcos, por ejemplo no es recomendable usar una arquitectura de 3 capas en un sistema en tiempo real.

Tenemos conocimiento de diferentes arquitecturas y patrones como:

- **Por capas:** Se agrupan funcionalidades comunes en capas que confían en la funcionalidad de la capa inmediatamente inferior y suministran interfaces a la capa superior, además reduciendo el acoplamiento ya que no se puede llamar a cualquier método desde cualquier lugar.

- **Microservicios:** El sistema es desarrollado como un conjunto de pequeños servicios débilmente acoplados entre sí que se comunican a menudo mediante mensajes HTTP. Es modular, de débil cohesión, separa las funciones y oculta la información no necesaria entre servicios, aunque de comunicaciones lentas y complejas.

- **SPA (Simple Page Aplication):** Una sola página que se carga inicialmente en el navegador, y que según interactúa con el usuario se reescribe a sí misma sin necesidad de interactuar con el servidor, o en caso de hacerlo, sólo son datos en formato JSON que se actualizan en la aplicación sin necesidad de recargar. Es posible establecer un sistema en tiempo real, siempre y cuando se sacrifique el tiempo en la carga inicial, además de que suele ser necesario trabajar con frameworks especializados.

- **Cliente-Servidor:** La arquitectura cliente-servidor es un modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta.

- **Patrón Front Controller:** Se trata de un patrón donde existe un controlador principal que maneja todas las peticiones entrantes del navegador y las reconduce a los controladores adecuados.

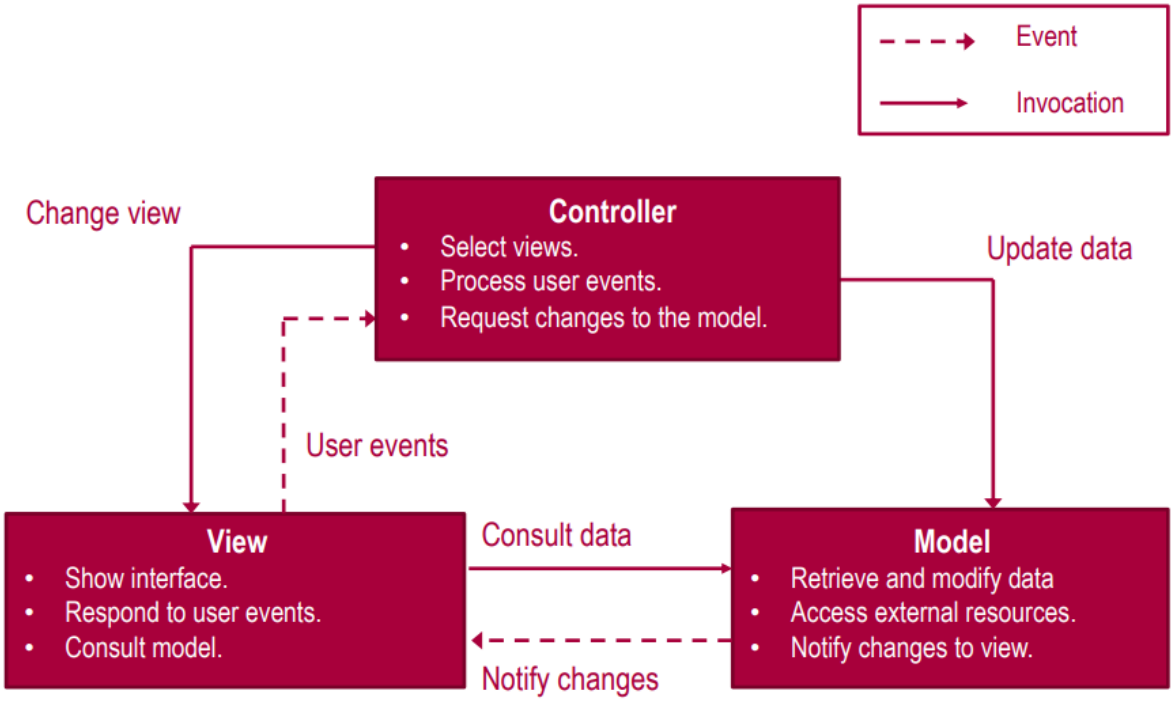
5.2. MVC

Si hemos trabajado con diferencia un patrón arquitectónico, este es el patrón MVC, Modelo-Vista-Controlador, junto a Spring Boot y el patrón front-controller dentro del marco de arquitectura Cliente-Servidor.
Centrándonos en el primero, como dice su nombre se divide en 3 partes bien diferenciadas con estos nombres:

- **Modelo:** Es la representación específica de la información con la que se opera, donde se junta la lógica de negocio con el repositorio de datos y generalmente brindan la información en forma de servicios (API)

- **Vista:** Se le ofrecen los datos del Modelo al cliente de forma usable a través de una interfaz de usuario por lo general.

- **Controlador:** Responde a los eventos que se producen en la Vista (peticiones HTTP) y provoca cambios en el modelo y por extensión, en la Vista.



(Esquema simple de relaciones del MVC) - "Architectural design of software applications" DP1 2021/2022

6. Conclusiones

Por lo observado y aprendido en el corto período en la asignatura, el proyecto a desarrollar tiene ciertas similitudes respecto a la tecnología ya abordada, ésto es por ejemplo el aspecto de la arquitectura y los patrones arquitectónicos principales usados en el proyecto. Sin embargo, tenemos un cambio de framework a descubrir, así como nuevas metodologías de trabajo y organización interna que con toda probabilidad supondrán uno de los retos principales en el aspecto de adaptación.

7. Bibliografía

- Wikipedia: Definición de arquitectura cliente-servidor
- Presentaciones del Tema 2 de DP1 2021/2022 *“Architectural design of software applications”*, Universidad de Sevilla