

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Architecture of WIS




Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas 2.

Curso 2024 – 2025

Grupo de prácticas: C1.001		
Autores por orden alfabético	Rol	Descripción del rol
Cantalejo Gómez, Olga - *027*3***	Developer, Analyst, Tester	Persona encargada de desarrollar código.
Escobar Sanchez, Alberto - 7**4*4**J	Project Manager, Developer, Analyst, Tester	Persona encargada de tomar decisiones de diseño y vigilar el correcto desarrollo
González Lucena, Juan Antonio - **86*0**A	Developer, Analyst, Tester	Persona encargada de desarrollar el código.
Paradas Borrego, Álvaro - 44**38***	Developer, Analyst, Tester	Persona encargada de desarrollar el código.
Suárez Coronel, Celia - 4**3**7*M	Developer, Analyst, Tester	Persona encargada de desarrollar el código.


	Diseño y Pruebas 2 Architecture of WIS
---	---

Control de Versiones

Fecha	Versión	Descripción
19/02/2025	v1.0.0	Desarrollo de la primera versión.

Índice de contenido


1. Resumen ejecutivo	2
2. Contenido	2
3. Conclusiones	3

	Diseño y Pruebas 2 Architecture of WIS

1. Resumen ejecutivo

Durante toda la etapa de aprendizaje en la ETSII hemos estudiado multitud de arquitecturas que explican cómo cada una de ellas solucionan diversos problemas recurrentes, dando mucha importancia a cómo estas facilitan la planificación y diseño del software siguiendo un patrón.

La arquitectura nos permite planificar a priori nuestro desarrollo y elegir el mejor conjunto de herramientas para llevar a cabo nuestros proyectos. Así conseguimos que nuestros proyectos mejoren en diferentes aspectos principalmente en escalabilidad, seguridad, rendimiento, mantenibilidad e integración con otros sistemas.

	<p>Diseño y Pruebas 2 Architecture of WIS</p>
---	---

2. Contenido

La arquitectura de software es un criterio fundamental en la creación de un sistema software.


Determina la organización, estructura y relaciones entre los componentes de un sistema, generando unos cimientos consolidados para su diseño, implementación y evolución.

Las diferentes arquitecturas que existen pueden mejorar diferentes aspectos y características que se consideran muy importantes, aunque pueden provocar desventajas sobre otros aspectos. Algunas de estas características son:

- **Escalabilidad:** Es la capacidad de un sistema software para expandirse para ampliar sus funcionalidades y alcance.
- **Seguridad:** Es la capacidad de un sistema software para soportar ataques externos.
- **Rendimiento:** Mide la eficacia en la que un sistema software cumple las funcionalidades para lo que estaba pensado.
- **Mantenibilidad:** Es la capacidad de un sistema software para cambiar y realizar correcciones.
- **Integración:** Es la capacidad de un sistema software de conectarse o fusionarse con otros sistemas

Cada estilo arquitectónico tiene sus ventajas y desventajas, y cada uno se aplica en un contexto y problema específico. Algunas de las arquitecturas más comunes y el contexto donde resultan idóneos son:

- **Arquitectura de capas:** Aplicaciones donde haya que desarrollar y mantener partes del sistema independientemente. Organiza el sistema en capas o niveles, donde cada capa tiene una función específica y se comunica con las capas adyacentes.
- **Arquitectura cliente-servidor:** Una parte conocida como la parte del cliente, la interfaz de usuario; y una parte llamada parte del servidor, que contiene la lógica de negocio y almacenamiento.

	Diseño y Pruebas 2 Architecture of WIS
---	---

- Arquitectura orientada a servicios: Consiste en la creación de servicios independientes y reutilizables que pueden ser utilizados por diferentes aplicaciones.
- Patrón arquitectónico MVC: Patrón arquitectónico que separa datos de una aplicación, interfaz de usuario y la lógica del negocio en tres componentes distintos.

3. Conclusiones

La arquitectura de un sistema de información web nos permite tomar decisiones de diseño en una etapa temprana del desarrollo y solucionar problemas recurrentes. Para ello conseguimos mejorar características como, por ejemplo, la escalabilidad y la integración. No obstante, puede provocar algunas desventajas en otros aspectos como el rendimiento, mantenimiento, seguridad...