Análisis y Diseño de Software - EIN081B

1. **Introducción**

Contexto de la Ingeniería en Informática y su enfoque en procesos empresariales, gestión de proyectos y principios fundamentales como el pensamiento sistémico, liderazgo y administración. Los sistemas de información requieren infraestructura sólida como redes internas y servicios web seguros.

## **Ingeniería de Requisitos**

La ingeniería de requisitos se centra en identificar, analizar, documentar y validar los requerimientos de un sistema:

* + Elicitación: recopilación y análisis de requisitos mediante técnicas como Joint Application Design.
  + Clasificación: requisitos funcionales y no funcionales (según ISO 25010).
  + Validación: asegurar que los requisitos sean consistentes, completos y verificables.
  + Especificación: formalización de requisitos para evitar errores en el desarrollo.

## **Orientación a Objetos (OO)**

La orientación a objetos es un paradigma que modela sistemas basados en entidades denominadas objetos.

* + Claves: encapsulación, herencia, polimorfismo y abstracción.
  + Ventajas: reutilización de código, flexibilidad y modelado del mundo real.
  + Relaciona objetos a través de clases, métodos y atributos.
  + Ejemplo: diseño de sistemas hospitalarios con relación entre entidades como hospitales, médicos y pacientes.

**Análisis orientado a objetos:**

Enfoque en identificar y definir los objetos, sus interacciones, sus características y responsabilidades.

## **Unified Modeling Language (UML)**

UML es un lenguaje gráfico desarrollado por el Object Management Group (OMG) que permite visualizar, especificar, construir y documentar los artefactos de un sistema de software. Recomiendo el uso de <https://diagramasuml.com/>, a mí me sirvió muchísimo.

Sus principales diagramas incluyen:

* + **ESTRUCTURALES:**
    - [Clases](https://diagramasuml.com/diagrama-de-clases/): definen estructuras y relaciones estáticas.
    - [Componentes](https://diagramasuml.com/componentes/), [Paquetes](https://diagramasuml.com/paquetes/) y [Despliegue](https://diagramasuml.com/despliegue/): muestran módulos de software y hardware.
  + **DE COMPORTAMIENTO:**
    - [Casos de Uso](https://diagramasuml.com/casos-de-uso/): muestran interacciones entre usuarios y el sistema.
    - [Secuencia](https://diagramasuml.com/secuencia/): representan el flujo de mensajes en el tiempo.
    - [Actividades](https://diagramasuml.com/actividades/): ilustran procesos dinámicos de un sistema.
    - [Máquina de Estados](https://diagramasuml.com/estados/): representan estados y transiciones de los componentes.
    - [Comunicación](https://diagramasuml.com/comunicacion/): muestra las interacciones entre objetos y/o actores utilizando mensajes.

## **Patrones y Calidad**

Los patrones de diseño ayudan a resolver problemas recurrentes y mejorar la calidad del software.

* Patrones arquitectónicos: Cliente-Servidor, Microservicios, MVC.
* Patrones de diseño (GoF): estructurales, creacionales y de comportamiento.
* GRASP: principios para asignar responsabilidades en el diseño.
* Calidad: pruebas estáticas y dinámicas, principios del ISTQB para asegurar robustez y fiabilidad.