Resumen de IS2:

# POO

Fundamentos:

* Escribir software fácilmente escalable y mantenible.
* Trasladar el dominio del software a un lenguaje mas natural.
* Escribir software reutilizable.

# UML

Es UN LENGUAJE, NO es una metodología. Es independiente del proceso, aunque lo agilize.

Viene a ayudarnos a entender mejor la arquitectura del software y facilitar la comunicación.

Es una tecnica de análisis orientado a objetos.

Expresa: Requisitos, Funcionalidades, comportamiento interno del sistema.

Otra forma de definirlo es: Representa el modelo funcional, de objetos y dinamico.

Muestran una abstracción del sistema, no todo.

Enfocado a distintos puntos de vista.

Permite la generación de código a través del diagrama.

Sirve para: Comunicarse entre devs, clientes y usar herramientas de generación automática de código.

Es estandarizado por **OMG**.

Es **extensible, flexible, y escalable.**

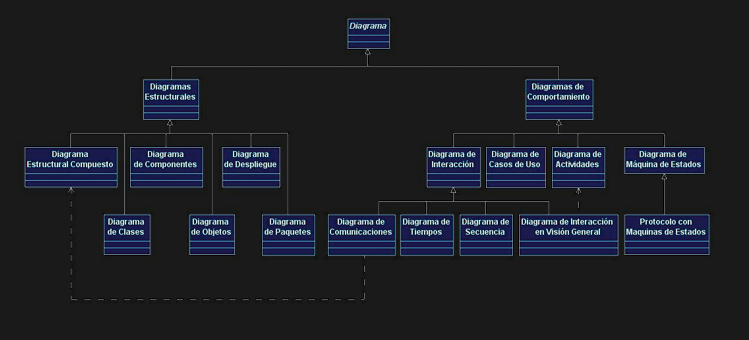
Podemos considerarlo como una forma visual de modelizar el negocio y sus reglas.

**Paralisis de análisis**: Cuando no se exagera el modelo hasta un punto que no es productivo.

Cada diagrama tiene un propósito especifico

Estan constituidos por la notación y los diagramas / objetos.

Arbol de diagramas:



Diagramas de Interaccion: Diagramas que engloban el intercambio de mensajes entre objetos.

Ahí podes encontrar el diagrama de Secuencia, comunicaciones, diagrama de interacción en visión general, de tiempos.

Diagramas de CU, Diagrama de actividades NO es parte del diagrama de interacción.

Diagrama de maquinas de estados = diagramas de estados ¿  
Diagramas de Comportamiento: Muestran como cambia el sistema con el tiempo.

Diagramas estructurales: Muestran la arquitectura del sistema.

# Abstraccion

Proceso de modelar un sistema complejo de forma sencilla y generalizada.

Solo se muestra una parte del sistema, no todo en el UML.

# Puntos de vista

Con esto se refiere a los distintos diagramas que tenemos, cada diagrama es un punto de vista distinto.

# Casos de Uso

Reflejan los requisitos funcionales, y como el usuario puede interactuar con el sistema.

Es una secuencia de interacciones. Se usa para mostrar los servicios que puede usar el usuario, y como los usa, pero NO como los implementa el sistema.

Estan mas relacionados con el Proceso Unificado de Desarrollo, no tanto con el UML.

Propone que la mejor forma de empezar a entender un sistema, es a partir de los servicios a los cuales pueden acceder los usuarios, y no tanto en los componentes internos.

Son agenos a la programación que se use y al método de disenio.

Es ideal hacerlo previo a diseniar el sistema estructurado.

Caracteristicas:

* Estan acotados a una funcionalidad diferenciada del sistema.
* Solo los puede iniciar un actor.
* El texto que lo refleja debe ser informal.
* El énfasis esta en la interacción.
* Son desde el punto de vista del actor.

Se compone del actor, de las funcionalidades y relaciones. Todo esto dentro del Sistema.

# Diagrama de clase

Es el mas común. Muestra la relación entre una clase y otra, pero no el que y como lo hacen.

El diagrama de clases muestra como se relacionan las clases entre si, que comportamientos tienen y de que se componen, pero no muestra información temporal.

Sus elementos básicos son las **clases** y las **relaciones**.

Tipos de objetos:

* Objetos entidad
* Boundary Objects (Objetos limite/Interfaz)
* Objetos de control

## Componentes raros

Visibilidades raras:

# protegido ~ paquete. (Protegido es ideal para la herencia)

Estereotipos: Nos permiten extender el lenguaje UML con cosas que no son propias de este, como los <<enum>> o las <<interfaces>>

Restricciones: Muestran limitaciones o condiciones de negocio, pre y postcondiciones, se pueden poner notas o en un mismo elemento. Se usa { } para denotarlas.

Realizacion: Implementacion de un estereotipo, muestra una implementacion

# Diagrama de Estados

Se conoce el numero de estados de antemano, nos centramos en el comportamiento.

Nos sirve para mostrar el comportamiento del sistema, ya sea por un caso de uso, como funciona una clase o el sistema entero.

El estado lleva en su interior ACTIVIDADES.

# Diagrama de secuencia

Muestra intercambio de mensajes entre objetos a lo largo del tiempo. Los objetos no tienen porque ser instancias de las clases, sino que tmb puede representar componentes del sistema.

Depende el nivel de abstracción se puede usar un lenguaje mas natural o mas cercano al a programación.

Mensaje llamada: Envia mensaje, espera respuesta.

Mensaje asíncrono: No espera respuesta

Automensaje: Mensaje a si mismo

Mensaje recursivo: Se usa recursión // no entiendo cual es la diferencia entre este y el anterior.

Mensaje encontrado: Mensaje que llega literalmente de un random.

Mensaje perdido: Mensaje que se va afuera, que no es parte del diagrama. (lo contrario al anterior)

# Diagrama de comunicación

Se enfoca mas en el orden en el que se van a dar los mensajes, que en el tiempo. Es decir, que sabremos el orden pero no cuando.

Es bastante parecido al de secuencia pero con ligeras diferencias.