**¿Qué es un modelo?**

Un modelo es una simplificación de la realidad

**¿Por qué modelamos?**

Construimos modelos para comprender mejor el sistema que estamos desarrollando

A través del modelado conseguimos:

* Los modelos nos ayudan a visualizar como es o queremos que sea un sistema
* Los modelos nos permiten especificar la estructura o el comportamiento de un sistema
* Los modelos nos proporcionan plantillas que nos guían en la construcción de un sistema
* Los modelos documentan las decisiones que hemos adoptado

Cuanto más grande es el sistema, el modelado se vuelve más importante, por una simple razón:

*Construimos modelos de sistemas complejos porque no podemos comprender el sistema en su totalidad*

A través del modelado reducimos el problema que se está estudiando, centrándonos en un solo aspecto cada vez. Este es el enfoque divide y vencerás de DIJKSTRA: al dividir el problema en una serie de sub problemas más pequeños que se puedan resolver

**Principios del modelado**

* ***La elección acerca de que modelos crear tiene una profunda influencia sobre cómo se acomete un problema y como se da forma a una solución***

En otras palabras hay que elegir bien los modelos, los modelos adecuados pueden arrojar mucha luz sobre los problemas de desarrollo más horribles

* ***Todo modelo puede ser expresado con diferentes niveles de precisión***

Los mejores tipos de modelos son aquellos que permiten elegir el grado de detalle, dependiendo de quién está viendo el sistema y porque necesita verlo.

* ***Los mejores modelos están ligados a la realidad***

Un modelo físico de un edificio que no responda de la misma forma que los materiales reales tan solo tiene un valor limitado. Es mejor tener modelos que tengan una clara conexión con la realidad y donde esta conexión sea débil saber exactamente como se apartan esos modelos del mundo real

* ***Un único modelo o vista no es suficiente. Cualquier sistema no trivial se aborda mejor a través de un pequeño conjunto de modelos casi independientes con múltiples puntos de vista***

La expresión clase es “casi independiente”. En este contexto significa tener modelos que podemos construir y estudiar separadamente, pero aun así están interrelacionados.

**UML**

El Lenguaje Unificado de Modelado, es un lenguaje estándar para escribir planos de software. UML puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema que involucre una gran cantidad de software

UML es un solo un lenguaje y por tanto es tan solo una parte de un método de desarrollo de software. UML es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en arquitectura e iterativo e incremental.

**Visión de UML**

UML es un lenguaje

Un lenguaje proporciona un vocabulario y las reglas para combinar palabras de ese vocabulario con el objetivo de posibilitar la comunicación. Un lenguaje de modelado es un lenguaje cuyo vocabulario se centra en la representación conceptual y física de un sistema. Un lenguaje de modelado estándar como UML es, por tanto, un lenguaje estándar para los planos del software.

El vocabulario y las reglas de un lenguaje como UML indican cómo crear y leer modelos bien formados, pero no dicen que modelos se deben crear ni cuando se debería crear

Es un lenguaje para:

* **Visualiza**

Al escribir modelos en UML, se supera el problema de la comunicación, ya que este lenguaje facilita la comunicación

UML es un lenguaje grafico que trasciende el lenguaje de programación

Detrás de cada símbolo en la notación de UML hay una semántica bien definida, de esta manera un desarrollador puede escribir un modelo en UML y otro desarrollador o incluso otra herramienta puede interpretar ese modelo sin ambigüedad.

* **Especificar**

Especificar significa construir modelos precisos, no ambiguos y completos. En particular UML cubre la especificación de todas las decisiones de análisis, diseño e implementación que debe realizarse al desarrollar y desplegar un sistema con gran cantidad de software

* **Construir**

UML no es un lenguaje de programación visual, pero sus modelos pueden conectarse de forma directa a una gran variedad de lenguajes de programación. Esto significa que se puede establecer correspondencia desde un modelo de UML a un lenguaje de programación, permitiendo así la ingeniería directa e inversa o ingeniería de ida y vuelta.

Por medio de herramientas case se puede generar el cogido y esto agiliza mucho el desarrollo del software

* **Documentar**

UML cubre la documentación de la arquitectura de un sistema y todos sus detalles. UML también proporciona un lenguaje para expresar requisitos y pruebas. Por último, UML proporciona un lenguaje para modelar las actividades de planificación de proyectos y gestión de versiones

**Vocabulario de UML**

El vocabulario de UML incluye tres clases de bloques básicos

* **Elementos**

Hay cuatro tipos de elementos

* + **Elementos estructurales**

Son los nombres de los modelos, en su mayoría son la parte estática de un modelos, colectivamente se los denomina *clasificadores*

* + **Elementos de comportamiento**

Son al parte dinámica de los modelos, estos son los verbos de un modelo y representan comportamiento en el tiempo y espacio

* + **Elementos de agrupación**

Son las partes organizativas de los modelos de UML, estos son las cajas en las que se puede descomponerse un modelo (Paquetes)

* + **Elementos de anotación**

Son las partes explicativas de los modelos UML, son comentarios que se pueden aplicar para describir, clarificar y hacer observaciones sobre cualquier elemento del modelo (notas)

* **Relaciones**

Hay cuatro tipos de relaciones

* + **Dependencia**

Es una relación semántica, en la cual un cambio en un elemento puede afectar a la semántica del otro elemento

* + **Asociación**

Es una relación estructural entre clases que describen un conjunto de enlaces, los cuales son conexiones entre objetos que son instancias de clases

* + **Generalización**

Es una relación de especialización/generalización en la cual el elemento especializado se base en la especificación del elemento generalizado

* + **Realización**

Es una relación semántica entre clasificadores, en donde un clasificador especifica un contrato con otro clasificador que garantiza que cumplirá

* **Diagramas**

UML incluye trece tipos de diagramas

* + **Diagrama de clases**

Muestra un conjunto de interfaces y colaboraciones, así como sus relaciones. Abarcan la vista de diseño estática de un sistema.

* + **Diagrama de objetos**

Muestran un conjunto de objetos y sus relaciones, estos diagramas cubren la vista de diseño estática o la vista de procesos estática de un sistema, como lo hacen los Diagrama de clases pero desde la perspectiva de casos reales o prototípicos.

* + **Diagrama de componentes**

Representan la encapsulación de una clase junto con sus interfaces, puertos y estructura interna la cual está formada por otros componentes anidados y conectores. Los diagramas de componentes cubren la vista de implementación estática del diseño.

* + **Diagrama de estructura compuesta**
  + **Diagrama de casos de uso**

Muestran un conjunto de casos de uso y actores y sus relaciones. Los diagramas de casos de uso cubren la vista de casis de uso estática de un sistema

* + **Diagrama de secuencia**

Al igual que los diagramas de comunicación es de tipo iteración. Un diagrama de iteración muestra una iteración que consta de un conjunto de objetos o roles, incluyendo los mensaje que pueden ser enviaos entre ellos. Un diagrama de secuencia es un diagrama de interacción que resalta la ordenación temporal de los mensajes; un diagrama de comunicación es un diagrama de interacción que resalta la organización estructural de los objetos o roles que envían y reciben mensajes

* + **Diagrama de estados**

Muestra una máquina de estados que consta de estados, transiciones, eventos y actividades. Un diagrama de estados muestra la vista dinámica de un objeto

* + **Diagrama de actividades**

Muestra la estructura de un proceso u otra computación como el flujo de control y datos paso a paso. Cubren la vista dinámica de un sistema

* + **Diagrama de despliegue**

Muestra la configuración de nodos de procesamiento en tiempo de ejecución y los artefactos que residen en ellos. Cubren la vista de despliegue estática de una arquitectura

* + **Diagrama de paquetes**

Muestra la descomposición del propio modelo en unidades organizativas y sus dependencias

* + **Diagrama de tiempo**

Es un diagrama de interacción que muestra los tiempos reales entre diferentes objetos o toles, oposición a la simple secuencia relativa de mensajes

* + **Diagrama de visión global de interacciones**

Es un hibrido entre un diagrama de actividades y un diagrama de secuencia

**Reglas de UML**

UML tiene las siguientes reglas sintácticas

* Nombre: como llamar a los elementos, relaciones y diagramas
* Alcance: el contexto que da un significado especifico a un nombre
* Visibilidad: Como se pueden ver y utilizar esos nombres por otros
* Integridad: Como se relacionan apropiada t consistentemente unos elementos con otros
* Ejecución: Que significa ejecutar o simular un modelo dinámico

**Mecanismos Comunes de UML**

UML tiene 4 mecanismos comunes:

* **Especificadores**

UML es algo más que un lenguaje gráfico, detrás de cada elemento de su notación grafica hay una especificación que proporciona una explicación textual de la sintaxis y semántica de ese bloque de construcción

* **Adornos**

Son los detalles de los elementos de UML, pueden ser detalles gráficos o textuales

* **Divisiones comunes**

Al modelar sistemas orientados a objetos el mundo a menudo se divide de varias formas

Clases y objetos, interfaz e implementación, tipo y rol

* **Mecanismos de extensibilidad**

UML es abierto-cerrado, siendo posible extender el lenguaje de manera controlada, los mecanismos de extensión comprende:

* + Estereotipos
  + Valores etiquetados
  + Restricciones