ENTORNOS DE DESARROLLO UD-3

UML

DIAGRAMAS UML

ÍNDICE

[INTRODUCCIÓN 3](#_Toc125361012)

[FAMILIARIZÁNDONOS CON LA NOTACIÓN UML 3](#_Toc125361013)

[A. NOTACIÓN 3](#_Toc125361014)

[B. MODELOS Y HERRAMIENTAS 3](#_Toc125361015)

[C. MÉTODOS 3](#_Toc125361016)

[TIPOS DE DIAGRAMAS UML 4](#_Toc125361017)

[HERRAMIENTAS PARA LA ELABORACIÓN DE DIAGRAMAS UML 5](#_Toc125361018)

[INGENIERÍA INVERSA 5](#_Toc125361019)

# INTRODUCCIÓN

UML (*Unified Modeling Language* o Lenguaje Unificado de Modelado) permite modular y documentar los elementos que forman un sistema software orientado a objetos.

Se ha convertido en el estándar de facto de la industria.

Nos permite visualizar diferentes vistas de un mismo producto, es decir, nosotros vamos a desarrollar nuestro proyecto y UML nos permite ver diferentes enfoques de una misma aplicación.

Permite plasmar fácilmente lo que a lo mejor no todo el mundo conoce, por ejemplo, un código Java no lo sabe interpretar todo el mundo pero un diagrama UML es más fácil de entender.

Además, el lenguaje UML nos permite conectarlo a lenguajes de programación mediante ingeniería directa e inversa mediante un plugin.

# FAMILIARIZÁNDONOS CON LA NOTACIÓN UML

## NOTACIÓN

Una notación es un conjunto de símbolos para combinarlos y que representen algo de forma entendible.

En UML hay varios tipos de diagramas, sin embargo, la mayoría de los desarrolladores tienden a utilizar uno o ninguno, algo que no debe realizarse. Lo mejor es utilizar dos.

En UML el lenguaje utilizado debe ser independiente del lenguaje de programación que se utilice.

Existen diferentes notaciones en UML.

## MODELOS Y HERRAMIENTAS

Un modelo captura una vista de un sistema del mundo real. Es una abstracción de dicho sistema, considerando un cierto propósito.

## MÉTODOS

Un método es un proceso disciplinado para generar uno/varios modelos que describen aspectos de un sistema de software en desarrollo, utilizando alguna notación bien definida.

# TIPOS DE DIAGRAMAS UML

Tenemos dos principalmente que se dividen en varios a su vez:

* **Diagramas estructurales.** Representan la visión estática del sistema. Especifican clases y objetos y como se distribuyen físicamente en el sistema.
  + **Diagrama de clases.** Conjunto de clases y sus relaciones.
  + **Diagrama de objetos.** Conjunto de objetos y sus relaciones.
  + **Diagrama de componentes.** Conjunto de componentes y sus relaciones. Así como la organización lógica de la implementación.
  + **Diagrama de despliegue.** Configuración del sistema en tiempo de ejecución.
  + **Diagrama de estructuras compuesta.** La estructura interna de una clase.
  + **Diagrama de paquetes.** Define cómo los elementos del modelo se organizan en paquetes.
* **Diagramas de comportamiento.** Muestran la conducta en tiempo de ejecución del sistema, tanto desde el punto de vista del sistema completo como de las instancias u objetos que lo integran.
  + **Diagrama de casos de uso.** Representa las acciones a realizar en el sistema desde el punto de vista de los usuarios.
  + **Diagrama de estado de la máquina.** Describe el comportamiento de un sistema dirigido por eventos.
  + **Diagrama de actividades.** Muestra el orden en el que se van realizando las tareas.
  + **Diagramas de interacción:**
    - **Diagrama de secuencia.** Representa la ordenación temporal en el paso de mensajes.
    - **Diagrama de comunicación/colaboración.** Resalta la organización estructural de los objetos que se pasan mensajes.
    - **Diagrama de interacción.** Muestra un conjunto de objetos y sus relaciones junto con los mensajes que se envían entre ellos.
    - **Diagrama de tiempos.** Muestra el cambio en un estado o una condición de una instancia o un rol a través del tiempo.

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

# HERRAMIENTAS PARA LA ELABORACIÓN DE DIAGRAMAS UML

Podemos encontrar, entre otras, las siguientes herramientas:

* **Rational Systems Developer de IBM:** herramienta propietaria que permite el desarrollo de proyectos software basados en la metodología UML. Desarrollada en origen por los creadores de UML ha sido absorbida por IBM.
* **UMLet:** herramienta UML de código abierto y libre distribución. Dispone de un interfaz de usuario sencillo de utilizar.
* **Visual Paradigm for UML (VP-UML):** incluye una versión para uso no comercial que se distribuye libremente sin más que registrarse para obtener un archivo de licencia. Dispone de diferentes módulos para realizar desarrollo UML, diseñar bases de datos, realizar actividades de ingeniería inversa y diseñar con Agile. Es compatible con los IDEs Eclipse, Visual Studio .net, IntelliJ IDEA y NetBeans.
* **ArgoUML:** se distribuye bajo licencia Eclipse. Soporta los diagramas de UML 1.4, y genera código para java y C++. Para poder ejecutarlo se necesita la plataforma java. Admite ingeniería directa e inversa.

# INGENIERÍA INVERSA

La ingeniería inversa se define como el proceso de analizar código, documentación y comportamiento de una aplicación para extraer información del diseño.

Existen varios tipos:

* **Ingeniería inversa de datos.** Se aplica sobre el código de bases de datos para obtener el modelo entidad-relación.
* **Ingeniería inversa de lógica o de proceso.** Se aplica sobre el código de un programa para averiguar su lógica, o sobre cualquier documento de diseño para obtener documentos de análisis o de requisitos.
* **Ingeniería inversa de interfaces de usuario.** Consiste en estudiar la lógica interna de las interfaces para obtener los modelos y especificaciones que sirvieron para su construcción, con objeto de tomarlas como puntos de partida en procesos de ingeniería directa que permitan su actualización.