Centro de Enseñanza Técnica Industrial

Plantel Colomos

Ingeniería en Desarrollo de Software

Nombre Alumno: Ruiz Gudiño José Rafael

Registro: 20110374

Arquitectura de Software

Actividad 2 - UML

5°P

T/M

20/02/2022

¿Qué es UML?

El Lenguaje Unificado de Modelado (UML) fue creado para forjar un lenguaje de modelado visual común y semántica y sintácticamente rico para la arquitectura, el diseño y la implementación de sistemas de software complejos, tanto en estructura como en comportamiento. UML tiene aplicaciones más allá del desarrollo de software, p. ej., en el flujo de procesos en la fabricación.

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

Es comparable a los planos usados en otros campos y consiste en diferentes tipos de diagramas. En general, los diagramas UML describen los límites, la estructura y el comportamiento del sistema y los objetos que contiene.

UML no es un lenguaje de programación, pero existen herramientas que se pueden usar para generar código en diversos lenguajes usando los diagramas UML. UML guarda una relación directa con el análisis y el diseño orientados a objetos.

La finalidad de UML según OMG (Object Management Group)

Brindar a arquitectos de sistemas, ingenieros y desarrolladores de software las herramientas para el análisis, el diseño y la implementación de sistemas basados en software, así como para el modelado de procesos de negocios y similares.

Hacer progresar el estado de la industria permitiendo la interoperabilidad de herramientas de modelado visual de objetos. No obstante, para habilitar un intercambio significativo de información de modelos entre herramientas, se requiere de un acuerdo con respecto a la semántica y notación.

UML cumple con los siguientes requerimientos:

- Establecer una definición formal de un metamodelo común basado en el estándar MOF (Meta-Object Facility) que especifique la sintaxis abstracta del UML. La sintaxis abstracta define el conjunto de conceptos de modelado UML, sus atributos y sus relaciones, así como las reglas de combinación de estos conceptos para construir modelos UML parciales o completos.
- Brindar una explicación detallada de la semántica de cada concepto de modelado UML. La semántica define, de manera independiente a la tecnología, cómo los conceptos UML se habrán de desarrollar por las computadoras.
- Especificar los elementos de notación de lectura humana para representar los conceptos individuales de modelado UML, así como las reglas para

combinarlos en una variedad de diferentes tipos de diagramas que corresponden a diferentes aspectos de los sistemas modelados.

- Definir formas que permitan hacer que las herramientas UML cumplan con esta especificación. Esto se apoya (en una especificación independiente) con una especificación basada en XML de formatos de intercambio de modelos correspondientes (XMI) que deben ser concretados por herramientas compatibles.

Ventajas

Los principales beneficios de UML son:

- Mejores tiempos totales de desarrollo (de 50 % o más).
- Modelar sistemas (y no sólo de software) utilizando conceptos orientados a objetos.
- Establecer conceptos y artefactos ejecutables.
- Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- Mejor soporte a la planeación y al control de proyectos.
- Alta reutilización y minimización de costos

Tipos de diagramas UML

UML usa elementos y los asocia de diferentes formas para formar diagramas que representan aspectos estáticos o estructurales de un sistema, y diagramas de comportamiento, que captan los aspectos dinámicos de un sistema.

Diagramas UML estructurales

Diagrama de clases: El diagrama UML más comúnmente usado, y la base principal de toda solución orientada a objetos. Las clases dentro de un sistema, atributos y operaciones, y la relación entre cada clase. Las clases se agrupan para crear diagramas de clases al crear diagramas de sistemas grandes.

Diagrama de componentes: Muestra la relación estructural de los elementos del sistema de software, muy frecuentemente empleados al trabajar con sistemas complejos con componentes múltiples. Los componentes se comunican por medio de interfaces.

Diagrama de estructura compuesta: Los diagramas de estructura compuesta se usan para mostrar la estructura interna de una clase.

Diagrama de implementación: Ilustra el hardware del sistema y su software. Útil cuando se implementa una solución de software en múltiples máquinas con configuraciones únicas.

Diagrama de objetos: Muestra la relación entre objetos por medio de ejemplos del mundo real e ilustra cómo se verá un sistema en un momento dado. Dado que los datos están disponibles dentro de los objetos, estos pueden usarse para clarificar relaciones entre objetos.

Diagrama de paquetes: Hay dos tipos especiales de dependencias que se definen entre paquetes: la importación de paquetes y la fusión de paquetes. Los paquetes pueden representar los diferentes niveles de un sistema para revelar la arquitectura. Se pueden marcar las dependencias de paquetes para mostrar el mecanismo de comunicación entre niveles.

Diagramas UML de comportamiento

Diagramas de actividades: Flujos de trabajo de negocios u operativos representados gráficamente para mostrar la actividad de alguna parte o componente del sistema. Los diagramas de actividades se usan como una alternativa a los diagramas de máquina de estados.

Diagrama de comunicación: Similar a los diagramas de secuencia, pero el enfoque está en los mensajes que se pasan entre objetos. La misma información se puede representar usando un diagrama de secuencia y objetos diferentes.

Diagrama de panorama de interacciones: Hay siete tipos de diagramas de interacciones. Este diagrama muestra la secuencia en la cual actúan.

Diagrama de secuencia: Muestra cómo los objetos interactúan entre sí y el orden de la ocurrencia. Representan interacciones para un escenario concreto.

Diagrama de máquina de estados: Similar a los diagramas de actividades, describen el comportamiento de objetos que se comportan de diversas formas en su estado actual.

Diagrama de temporización: Al igual que en los diagramas de secuencia, se representa el comportamiento de los objetos en un período de tiempo dado. Si hay un solo objeto, el diagrama es simple. Si hay más de un objeto, las interacciones de los objetos se muestran durante ese período de tiempo particular.

Diagrama de caso de uso: Representa una funcionalidad particular de un sistema. Se crea para ilustrar cómo se relacionan las funcionalidades con sus controladores (actores) internos/externos.

Conclusiones

UML es un lenguaje de gráfico de modelado, que como su nombre lo dice ayuda a planificar, diseñar, estructurar y modelar sistemas de software complejos para que de esta manera exista una mejor gestión de un proyecto de software y optimizar costos de producción. En lo que respecta en la arquitectura de software, UML nos proporciona una gran ayuda ya que nos da las herramientas necesarias para poder plasmar visualmente la estructura, modelo, comportamiento y la implementación de esto a un proyecto de software con el que se vaya a trabajar.

Bibliografía

Lucid Software Inc. (s. f.). *Qué es el lenguaje unificado de modelado (UML)*. Lucidchart. Recuperado 20 de febrero de 2022, de https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml/#section_7

Román Zamitiz, C. A. (s. f.). *El Lenguaje Unificado de Modelado (UML)*. UNAM. Recuperado 20 de febrero de 2022, de http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html