“Universidad Autónoma del Estado de México”

(UAEM)

PLANTEL: Unidad Académica Profesional (UAP) Tianguistenco.

Tarea 2

Análisis y diseño de software.

Investigación UML y Herramientas Case

Equipo:

Alanis Agular Yohualy Alejandra

Cejudo Tovar Alejandro

Fuentes Esquivel Kevin Brian

Villana Rueda Efren Jair.

Fecha de entrega: 10 de agosto.

Ciclo escolar 2022B

Mtra. Roció Elizabeth Pulido Alba

Introducción:

El propósito de este trabajo es presentar la información obtenida de la investigación sobre UML y Case Tools, e identificar cada uno de los datos generales y sus principales características que los diferencian del resto de métodos de creación de sistemas. Se utilizaron las diversas fuentes de información disponibles en Internet (libros, artículos, blogs, videos, etc.) para obtener información relevante y validada mutuamente.

Por lo tanto, a continuación, se dará la definición, estructura, parámetros y parte de trabajo de las dos metodologías, identificando los puntos más relevantes para su consideración, donde se deben resaltar los méritos de las dos metodologías y su uso para los requerimientos y conformación del sistema.

Calidad que responde eficazmente a las necesidades específicas. Además de describir el trabajo de cada parte interesada específicamente para comprender su relación con el equipo de desarrollo y su contribución al producto final.

Finalmente, cabe señalar que el material fue creado en colaboración con un equipo de 4 miembros que previamente realizaron su propia investigación durante las horas de clase para recopilar todos los datos y crear este material complementario, eliminando la única información redundante y organizándola en un forma lógica y comprensible. Cabe destacar que este documento, en definitiva, contiene la información más relevante sobre metodologías de creación de sistemas.

UML

¿Qué es UML?

Lenguaje de Modelado Unificado (Unified Modeling Language) 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 comparable a los planos usados en otros campos y consiste en diferentes tipos de diagramas, 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.

¿Qué se puede modelar con UML?

UML 2.0 define trece tipos de diagramas, divididos en tres categorías: seis tipos de diagramas representan una estructura de aplicación estática; tres representan tipos generales de comportamiento; y cuatro representan diferentes aspectos de las interacciones:

- **Los diagramas de estructura** incluyen el diagrama de clases, el diagrama de objetos, el diagrama de componentes, el diagrama de estructura compuesta, el diagrama de paquetes y el diagrama de implementación.
- **Los diagramas de comportamiento** incluyen el diagrama de casos de uso (utilizado por algunas metodologías durante la recopilación de requisitos); Diagrama de actividad y Diagrama de máquina de estados.
- **Los diagramas de interacción**, todos derivados del diagrama de comportamiento más general, incluyen el diagrama de secuencia, el diagrama de comunicación, el diagrama de tiempo y el diagrama general de interacción.

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 de Estructura:

- **Diagrama de clases:** Este diagrama, el más común en el desarrollo de software, se usa para representar el diseño lógico y físico de un sistema, y muestra sus clases. Tiene un aspecto similar al del diagrama de flujo porque las clases se representan con cuadros. Este diagrama ofrece una imagen de las diferentes clases y la forma en la que se interrelacionan, y cada clase posee tres compartimientos:
    - Sección superior: nombre de clase
    - Sección central: atributos de clase
    - Sección inferior: métodos u operaciones de clase

- **Diagrama de objetos:** A menudo, este diagrama se usa como una forma de comprobar la revisión de un diagrama de clases para fines de precisión. En otras palabras, ¿funcionará en la práctica? Muestra los objetos de un sistema y sus relaciones, y ofrece una mejor visión de los potenciales defectos de diseño que necesitan reparación.

- **Diagrama de componentes:** También conocido como diagrama de flujo de componentes, muestra agrupaciones lógicas de elementos y sus relaciones. En otras palabras, ofrece una vista más simplificada de un sistema complejo al desglosarlo en componentes más pequeños. Cada una de las piezas se muestra con una caja rectangular, que tiene su nombre escrito dentro. Los conectores definen la relación/las dependencias entre los diferentes componentes.

- **Diagrama de estructura compuesta:** Este lo utilizan rara vez las personas externas al campo de desarrollo de software. ¿Por qué? Aunque es similar a un diagrama de clases, adopta un enfoque más profundo, que describe la estructura interna de múltiples clases y muestra las interacciones entre ellas. Salvo que usted sea desarrollador, la vista de nivel superior probablemente le entregará información suficiente.

- **Diagrama de paquetes:** Este se utiliza para representar las dependencias entre los paquetes que componen un modelo. Su objetivo principal es mostrar la relación entre los diversos componentes grandes que forman un sistema complejo.

- **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.

Diagramas de Comportamiento:

- **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.

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

Diagramas de interacción:

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

- **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 Tiempo:** 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.

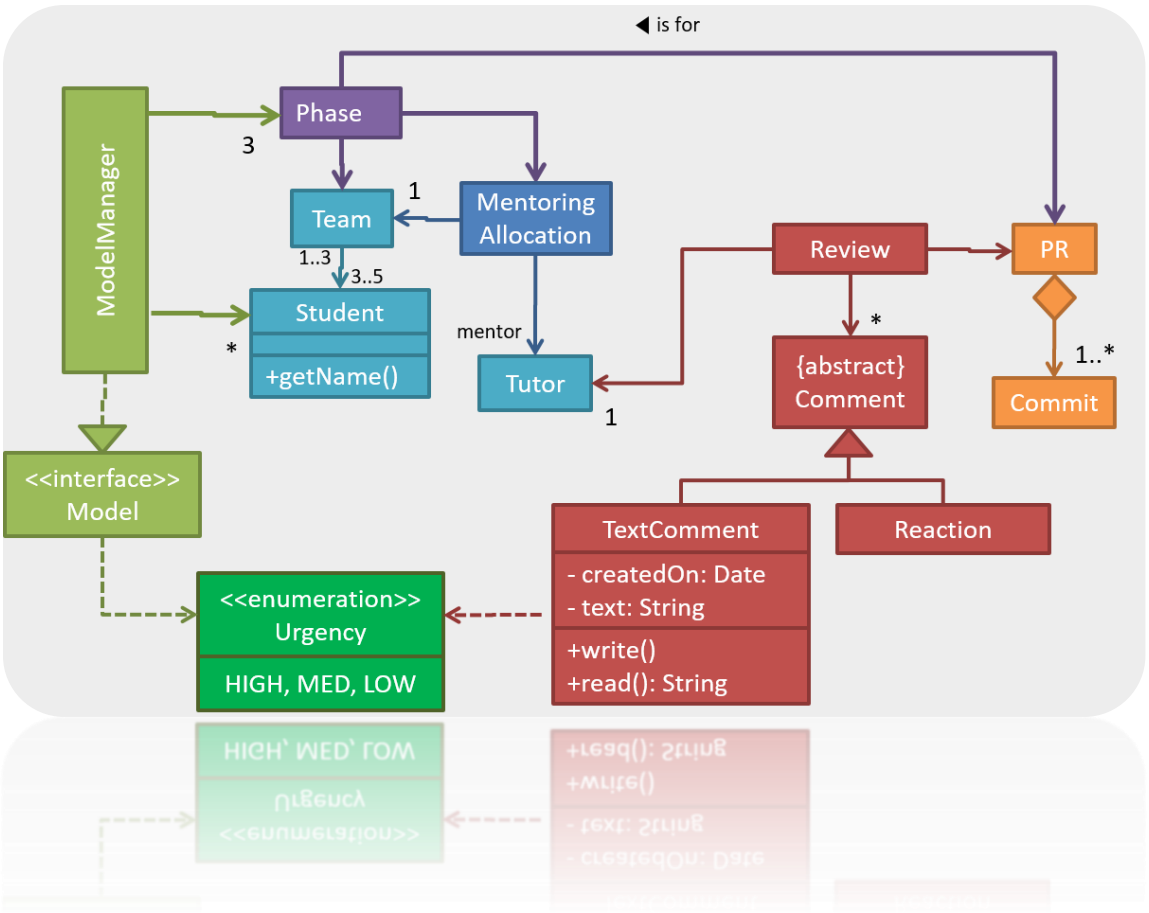
Sintetizando un poco la información de arriba en una tabla pera mejor entendimiento.

Categoría	Tipo de diagrama	Aplicación
Estructura	Diagrama de clases	Visualizar clases
	Diagrama de objetos	Estado del sistema en un momento dado
	Diagrama de componentes	Estructurar componentes y mostrar relaciones
	Diagrama de estructura compositiva	Divide los componentes o clases en sus componentes y aclara sus relaciones.
	Diagrama de paquete	Agrupa las clases en paquetes, muestra la jerarquía y la estructura de los paquetes.

	Diagrama de distribución	Distribución de componentes a los nodos informáticos
Comportamiento	Gráfica de perfil	Ilustra contextos de uso a través de estereotipos, condiciones límite, etc.
	Diagrama de casos de uso	Representa varias aplicaciones
	Diagrama de actividades	Describe el comportamiento de diferentes procesos (paralelos) en un sistema.
Comportamiento: interacción	Diagrama de máquina de estados	Documenta cómo un objeto es movido de un estado a otro por un evento.
	Diagrama secuencial	Secuencia temporal de las interacciones entre objetos
	Diagrama de comunicación	Distribución de roles de los objetos dentro de una interacción
	Diagrama de tiempos	Limitación de tiempo para los acontecimientos que conducen a un cambio de estado
	Diagrama de interacción	Secuencias y actividades interactivas

Ventajas del UML

- Simplifica las complejidades
- Mantiene abiertas las líneas de comunicación
- Automatiza la producción de software y los procesos
- Ayuda a resolver los problemas arquitectónicos constantes
- Aumenta la calidad del trabajo
- Reduce los costos y el tiempo de comercialización


is for
ModelManager
Phase
3
Team
1
Mentoring Allocation
1..3
3..5
Student
+getName()
*
mentor
Tutor
1
Review
PR
*
1..*
{abstract} Comment
Commit
<<interface>> Model
TextComment
- createdOn: Date
- text: String
+write()
+read(): String
Reaction
<<enumeration>> Urgency
HIGH, MED, LOW

HERRAMIENTAS CASE

Las siglas 'CASE' se refieren a Computer Aided Software Engineering (Ingeniería de Software Asistida por Computadora). Por tanto, se refiere al desarrollo y mantenimiento de proyectos de Software con la ayuda de varias herramientas automatizadas.

Son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de estas en términos de tiempo y de dinero, por lo que pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el diseño de proyectos, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

Las herramientas CASE se pueden dividir en las siguientes partes en base a a su uso en una etapa concreta del SDLC:

- Depósito central - Las herramientas CASE requieren un Depósito central, el cual nos puede servir como fuente de común, consistente e integrada información. Es un lugar central de almacenamiento, donde los requisitos del producto, los documentos requeridos, los informes y diagramas relacionados, y otra información útil sobre la gestión se almacena. El Depósito central también sirve como Diccionario de datos.
- Herramientas Upper CASE - Las Herramientas Upper CASE se usan en las etapas de planificación, análisis y diseño del SDLC.
- Herramientas Lower CASE - Las Herramientas Lower CASE se usan en la implementación, las pruebas y en el mantenimiento.
- Herramientas Integrated CASE - Las Herramientas Integrated CASE son de utilidad en todas las fases del SDLC, desde la obtención de requisitos y las pruebas hasta la documentación

La Herramientas CASE se pueden agrupar todas juntas si tienen una funcionalidad similar, y procesa actividades y la capacidad de integrarse con otras Herramientas.

Alcance de las herramientas CASE recorre el SDLC.

Tipos de Herramientas CASE

Ahora veremos de manera breve varios casos de herramientas CASE

Herramienta CASE Diagrama

Estas herramientas se usan para representar componentes del sistema, datos, y a controlar la fluidez de varios componentes y estructura del software de manera gráfica.

Herramientas para modelado de procesos

El modelado de procesos es un método para crear modelos de proceso de software y se usa para desarrollar el software. Las herramientas para el modelado de procesos ayudan a los directores a escoger un modelo de proceso o para modificarlo según los requerimientos del producto software.

Herramientas de administración de procesos

Estas herramientas se usan para la planificación del proyecto, el coste y esfuerzo estimados, la temporalización y la organización de los recursos. Los Directivos deben coordinar de manera muy estricta la ejecución del proyecto con cada uno de los pasos mencionados con anterioridad para la buena gestión del proyecto software. Herramientas de administración de procesos ayudan a almacenar y a compartir información sobre el proyecto en tiempo real durante su organización.

Herramientas de documentación

La documentación de un proyecto de software empieza antes que el proceso de software, pasa por todas las fases del SDLC y se concluye con la terminación del proyecto.

Las Herramientas de documentación generan documentos tanto para el consumidor final como para consumidores de soporte técnico. Estos últimos son en su mayoría profesionales internos del equipo de desarrollo que consultan manuales de sistemas, manuales de referencia, manuales de formación, de instalación, etc. El consumidor final describe el funcionamiento e instrucciones del sistema como por ejemplo el manual para el usuario.

Herramientas de análisis

Estas herramientas ayudan a cumplir con los requisitos, de manera automática examinan si hay alguna inconsistencia, o informaciones no acuradas en los diagramas, buscan posibles redundancias o omisiones erróneas.

Herramientas de diseño

Estas herramientas ayudan a los diseñadores de software a crear la estructura de los programas, la cual se puede más adelante desglosar en pequeños módulos usando técnicas de perfeccionamiento. Estas herramientas aportan los detalles de cada módulo y la interconexión presente entre estos.

Herramientas para la gestión de la Configuración

Un ejemplo de software se lanza al mercado en una versión. Las Herramientas para la gestión de la Configuración se ocupa de ello –

- Control de versiones
- Línea base
- Gestión del control de cambios

Las herramientas CASE ayudan en esto usando un rastreo automático, control de versiones y gestión de versiones.

Herramientas de control de cambios

Estas herramientas son consideradas como una parte de la configuración en la gestión de herramientas. Se ocupan de los cambios hechos en el software después de que se haya fijado su línea de base, o cuando el software se lanza por primera vez al mercado. Las herramientas CASE automatizan la opción 'resaltar cambios', la gestión de archivos, la gestión del código, entre otros. También ayuda a ejecutar el cambio de principios en que se basa la organización.

Programming Tools

These tools consist of programming environments like IDE (Integrated Development Environment), in-built modules library and simulation tools. These tools provide comprehensive aid i n building software product and include features for simulation and testing.

Herramientas de desarrollo de software

El modelo de prototipo en Ingeniería de software, es una versión simulada del producto software que se intenta conseguir. Este prototipo da una idea inicial del producto y simula algunos aspectos del producto real.

Las Herramientas de modelos de prototipo CASEP, esencialmente vienen con bibliotecas gráficas. Pueden crear interfaces de usuario independientes del hardware y diseño. Estas herramientas nos ayudan a construir prototipos rápidos basados en información ya existente. Además, producen prototipos de simulación de software.

Herramientas de desarrollo Web

Estas herramientas ayudan en el diseño de páginas Web con todos los elementos relacionados como impresos, textos, secuencias de comando, gráficos y demás. Las herramientas Web también producen una vista preliminar en directo de lo que se está desarrollando y cómo será una vez terminado.

Herramientas de Aseguramiento de la calidad

El aseguramiento de la calidad de una organización de Software es la supervisión del proceso de Ingeniería y de los métodos adoptados para desarrollar el producto software con tal de asegurar conformidad con la calidad según los estándares organizativos. Las herramientas de Aseguramiento de la calidad constan de

herramientas de control de cambios y configuración y de herramientas para pruebas de software.

Herramientas de mantenimiento

El mantenimiento del Software incluye modificaciones en el producto software después de ser distribuido. Algunas de las herramientas CASE que ayudan en la organización y la fase de mantenimiento del software del SDLC son las técnicas de inicio automático y de reporte de error, producción automática de etiqueta de error y de Análisis de Causa Raíz (ACR o RCA en sus siglas en inglés).

Características que se plantea el ser una buena herramienta CASE:

- Permitir la aplicación práctica de metodologías estructuradas, las cuales al ser realizadas con una herramienta conseguimos agilizar el trabajo.

- Facilitar la realización de prototipos y el desarrollo conjunto de aplicaciones.

- Simplificar el mantenimiento de los programas.

- Mejorar y estandarizar la documentación.

- Aumentar la portabilidad de las aplicaciones.

- Facilitar la reutilización de componentes software.

- Permitir un desarrollo y un refinamiento visual de las aplicaciones, mediante la utilización de gráficos.

Conclusiones

Alejandra: A partir de esta investigación nos permitió abrirnos al tema de UML y Herramientas CASE, darnos cuenta de los múltiples usos, así mismo pudimos investigar los diagramas, su clasificación y significado de cada uno de ellos, buscando entender y tener un conocimiento base para no llegar en ceros a clase, a su vez tenemos idea de las herramientas CASE, sus usos y donde podemos encontrar cada una de estas, así como una definición de los que es cada uno.

Efrén: La presente investigación ha permitido llevar a cabo un repaso sobre el Lenguaje de Modelado Unificado (UML), y por lo tanto obtener una mejor comprensión de su funcionamiento, características, tipos y ventajas de uso a proyectos de software. Comprendiendo su relevancia dentro de este campo laboral y porque es importante aprender a utilizarlo correctamente para futuros proyectos que llevemos a cabo.

También se identificó el concepto de herramientas CASE y el papel que juegan dentro del desarrollo de software, así como sus características que debe de plantearse a manera de objetivos con la finalidad de realmente mejorar la calidad y la productividad de los sistemas de información.

Alejandro: Realmente son las herramientas CASE el mejor método para el análisis y soluciones de software, ya que han venido a mejorar los aspectos claves en el desarrollo de los sistemas de información, las CASE han sido creadas para la automatización de procesos de análisis, diseño e implementación, brindándonos una un sin número de componentes que hacen que los proyectos sean cada día más eficientes para los usuarios finales.

Las Herramientas Case han venido a revolucionar la forma de automatizar los aspectos clave del mejor desarrollo del ciclo de vida del Software, el mejoramiento en la calidad y su productividad de información. Para conseguir estos objetivos es conveniente contar con una organización y una metodología de trabajo además de la propia herramienta.

La mejora de calidad se consigue reduciendo sustancialmente muchos de los problemas de análisis y diseño, inherentes a los proyectos de mediano y gran tamaño (lógica del diseño, coherencia, consolidación, etc.).

La mejora de productividad se consigue a través de la automatización de determinadas tareas como la generación de código y la reutilización de objetos o módulos.

Kevin: El objetivo de UML es ser capaz de describir el comportamiento de un sistema, subsistema u operación particular mediante un diagrama de secuencia el cual muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso, esto facilita como se distribuyen las tareas entre los componentes.

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

Fuentes bibliográficas:

-Equipo del Centro de crecimiento. (2019). La guía sencilla para la diagramación de UML y el modelado de la base de datos. 09/08/2022, de Microsoft Sitio web: https://www.microsoft.com/es-mx/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling#:~:text=El%20Lenguaje%20Unificado%20de%20Modelado,de%20un%20sistema%20o%20proceso.

-Anónimo. (2015). Herramienta CASE. 09708/2022, de EcuRed Sitio web: https://www.ecured.cu/Herramienta_CASE.

tutorialspoint. (-). Software - CASE Herramientas. 08/Agosto/2022, de tutorialspoint.com Sitio web: https://www.tutorialspoint.com/es/software_engineering/case_tools_overview.htm

Lucidchart. (-). Qué es el lenguaje unificado de modelado (UML) ¿Para qué necesitas crear un diagrama UML?. 09/Agosto/2022, de lucidchart.com Sitio web: https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml

Microsoft 365 Team. (2019). La guía sencilla para la diagramación de UML y el modelado de la base de datos. 09/Agosto/2022, de Microsoft.com Sitio web: https://www.microsoft.com/es-mx/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling

UML. (2005). ¿QUÉ ES UML?. 09/Agosto/2022, de www.uml.org Sitio web: https://www.uml.org/what-is-uml.htm