Casos de uso

Introducción

En el mundo del desarrollo de software, es esencial tener una comprensión clara y completa de los requerimientos del sistema antes de comenzar a trabajar en él. Para ello, una técnica de modelado de requerimientos ampliamente utilizada es la creación de diagramas de casos de uso. Los diagramas de casos de uso son una forma visual de representar los diferentes actores, tareas y flujos de interacción que ocurren dentro de un sistema.

Si eres nuevo en el mundo del modelado de requerimientos y diagramas de casos de uso, puede parecer abrumador al principio. Sin embargo, hay muchos recursos excelentes disponibles para ayudarte a entender los conceptos y técnicas detrás de esta práctica esencial de desarrollo de software.

Uno de los libros más populares sobre este tema es "UML Gota a Gota" de Martin Fowler. Este libro proporciona una introducción clara y completa a UML (Lenguaje de Modelado Unificado) y diagramas de casos de uso, incluyendo un análisis detallado de cada elemento en un diagrama de casos de uso y cómo se relacionan entre sí.

Otro libro útil es "Aprendiendo UML en 24 horas" de Joseph Schmuller, que se centra en enseñar los conceptos y técnicas de UML y diagramas de casos de uso de manera fácil de entender en un corto período de tiempo. Este libro es especialmente útil si deseas una introducción rápida a los conceptos básicos de UML y diagramas de casos de uso.

El "Manual de UML" de Paul Kimmel es otro recurso valioso que ofrece una comprensión detallada de UML y los diferentes tipos de diagramas, incluidos los diagramas de casos de uso. El enfoque del libro es aprender UML mediante la creación de diagramas y ejemplos paso a paso.

Por último, "Diseñar y Programar, Todo es Empezar: Una Introducción a la Programación Orientada a Objetos Usando UML y Java" de José Serrano y otros, es una excelente opción para aquellos que deseen aprender sobre UML y diagramas de casos de uso en el contexto de la programación orientada a objetos. Este libro también proporciona un tutorial completo sobre Java y su aplicación en el diseño y desarrollo de software.

Como vemos, hay muchos recursos excelentes disponibles para aprender sobre UML y diagramas de casos de uso, desde libros más detallados hasta guías rápidas. Tomarse el tiempo para aprender sobre esta técnica vital de modelado de requerimientos puede ayudar a mejorar la comprensión y el éxito en el desarrollo de software.

Todos los libros tienen en su título “UML”!

UML (Lenguaje de Modelado Unificado) es una herramienta utilizada para la visualización, especificación, construcción y documentación de sistemas complejos. Fue desarrollada en los años 90 por un grupo de expertos en lenguajes de programación y análisis de sistemas.

El lector puede leer –;D-, la sección "¿Qué es UML?", se explican los fundamentos del lenguaje, sus elementos principales (como clases, objetos, relaciones, diagramas, entre otros), y cómo estos elementos se utilizan en la modelación de sistemas. También se mencionan las diferentes versiones de UML y su evolución a lo largo del tiempo.

En la sección "Cómo llegamos hasta aquí", se hace una breve reseña histórica sobre la evolución de los lenguajes de modelado de sistemas y cómo UML se convirtió en uno de los más populares y utilizados en la actualidad. Se mencionan los antecedentes de UML, como los lenguajes OMT, Booch y OOSE, y cómo estos influyeron en el desarrollo del lenguaje actual.

La creación de diagramas UML es esencial para el desarrollo de software orientado a objetos. Si bien el manual de UML de Paul Kimmel recomienda el uso de Visio para crear estos diagramas, existen alternativas gratuitas en línea que pueden ser de gran ayuda, como Lucidchart, Creately y *Draw.io*. Si no

tienes acceso a Visio o prefieres no usarlo, estas herramientas pueden ser un excelente sustituto para crear diagramas UML de manera fácil y rápida.

No todo es sobre UML:

La programación orientada a objetos es uno de los pilares fundamentales de la programación moderna, permitiendo la creación de programas modulares y escalables. Si bien ya hemos estudiado los conceptos principales de la POO, siempre es importante repasarlos para consolidar nuestro conocimiento. En este sentido, la sección 1.2 del libro "Diseñar y programar, todo es empezar: una introducción a la Programación Orientada a Objetos usando UML y Java" de José Serrano y otros es una excelente herramienta para recordar los conceptos básicos de la POO de manera sencilla y con ejemplos concretos.

En el capítulo 8.2 del libro "Diseñar y programar, todo es empezar: una introducción a la Programación Orientada a Objetos usando UML y Java" de José Serrano y otros, se presentan algunos principios útiles en Programación Orientada a Objetos (POO), como el principio de responsabilidad única, el principio de sustitución de Liskov, el principio de segregación de interfaces, el principio de inversión de dependencias y el principio de Open/Closed. Además, se recomienda la lectura de otros libros clásicos como "Design Patterns" de Erich Gamma, "Refactoring" de Martin Fowler y "Clean Code" de Robert C. Martin, que son fundamentales en el desarrollo de software con POO. Definitivamente es un capítulo que vale la pena revisar como lectura transversal al curso.

Identificación de Casos de Uso

La identificación de casos de uso es el proceso de identificar las funcionalidades del sistema a través de la definición de escenarios de uso del sistema por parte de los usuarios. Un caso de uso es una descripción detallada de cómo el sistema responde a una acción particular del usuario.

Para identificar los casos de uso, se debe analizar cómo los usuarios interactúan con el sistema, lo que les permite definir los escenarios que conforman el funcionamiento del sistema. Es importante tener en cuenta que la identificación de casos de uso debe involucrar a todos los stakeholders, ya que cada uno tiene diferentes necesidades y expectativas del sistema.

Un ejemplo de caso de uso es el proceso de realizar una compra en línea. El escenario comienza con el usuario ingresando a la página web del sitio de compras y navegando por los productos disponibles. Luego, el usuario selecciona los productos que desea comprar y los agrega a su carrito de compras. Después, el usuario procede a la página de pago y proporciona los detalles de su tarjeta de crédito. Finalmente, el sistema procesa la transacción y confirma la compra al usuario.

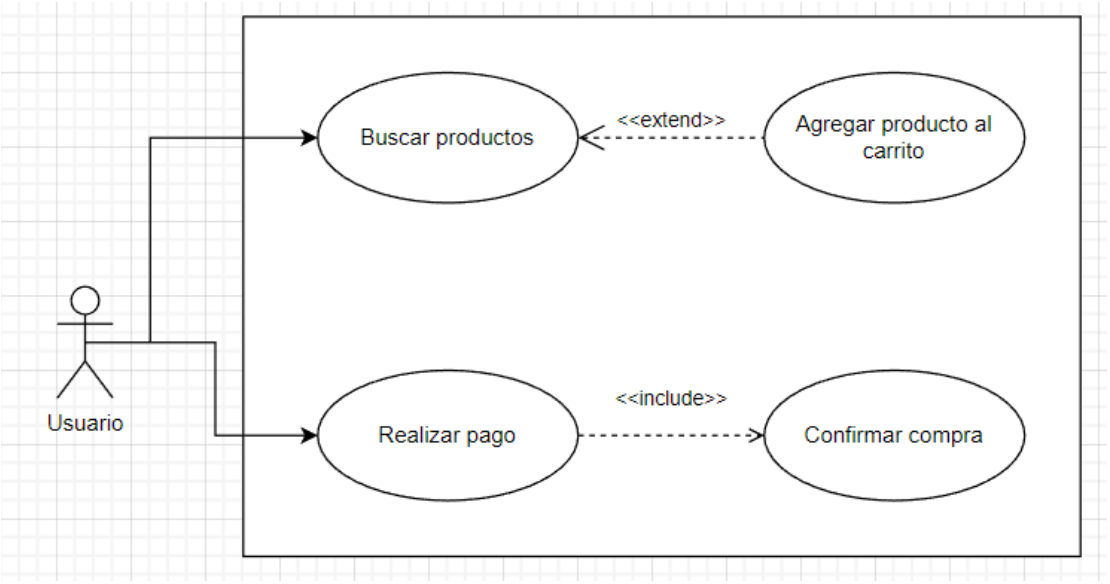
Identificar los casos de uso es importante porque ayuda a definir los requisitos del sistema y a garantizar que se cumplan las expectativas del usuario. Además, permite una mejor comprensión de las funcionalidades del sistema y ayuda a los desarrolladores a crear un diseño y una arquitectura que cumpla con las necesidades del usuario.

Relaciones entre Casos de Uso

Las relaciones entre casos de uso son importantes para entender cómo se relacionan los distintos actores y casos de uso. En el ejemplo que utilizamos anteriormente, el caso de uso "Comprar producto"

se relaciona directamente con el actor "Cliente", ya que es el cliente quien realiza la compra. Sin embargo, también puede haber relaciones entre casos de uso, como por ejemplo entre "Buscar producto" y "Comprar producto", ya que para comprar un producto es necesario primero buscarlo en el catálogo. Estas relaciones se representan mediante flechas que indican la dirección de la relación y el tipo de relación que existe entre los casos de uso. Es importante identificar y entender estas relaciones para poder modelar correctamente el sistema.

Diagrama de Casos de Uso



El diagrama de casos de uso es una herramienta gráfica que permite representar los diferentes casos de uso de un sistema y las relaciones entre ellos. Cada caso de uso se representa mediante un óvalo en el diagrama, mientras que los actores que interactúan con el sistema se representan como elementos externos a él. Las relaciones entre los casos de uso y los actores se representan mediante flechas, y cada flecha debe tener una etiqueta que indique el tipo de relación.

Tomando como ejemplo el proceso de realizar una compra en línea, podemos ver cómo el actor "Usuario" se relaciona con los casos de uso "Buscar productos" y "Realizar pago". La relación entre el actor "Usuario" y el caso de uso "Buscar productos" es una relación de inclusión (include), ya que el usuario debe buscar los productos antes de agregarlos al carrito y realizar el pago. Es decir, el caso de uso "Buscar productos" es necesario para que el usuario pueda realizar la compra.

Por otro lado, la relación entre los casos de uso "Agregar producto al carrito" y "Realizar pago" es una relación de extensión (extend), ya que el caso de uso "Agregar producto al carrito" es opcional y no siempre sucede antes de realizar el pago. La relación de extensión se utiliza para casos de uso opcionales o que agregan funcionalidad adicional al sistema. En este caso, el usuario puede decidir agregar o no productos al carrito antes de realizar el pago, por lo que la relación de extensión es la adecuada.

En el diagrama de casos de uso, también se puede mostrar cómo el sistema interactúa con otros sistemas externos, como un sistema de pago, para procesar la transacción y confirmar la compra al usuario. La relación entre el caso de uso "Realizar pago" y el sistema de pago podría ser una relación

de inclusión (include), ya que el caso de uso "Realizar pago" depende del sistema de pago para procesar la transacción.

Escenarios

Los escenarios son una herramienta importante en el desarrollo de sistemas para entender cómo los casos de uso se relacionan entre sí y cómo interactúan con los actores del sistema. En esencia, los escenarios describen situaciones hipotéticas que permiten a los desarrolladores comprender cómo un usuario interactúa con el sistema en diferentes etapas del proceso.

En el contexto de la compra en línea, los escenarios pueden incluir desde la búsqueda de productos hasta la confirmación de la compra. Por ejemplo, un escenario posible podría ser el siguiente:

- El usuario ingresa al sitio web de compras y navega por los diferentes productos disponibles.
- El usuario selecciona un producto y lo agrega a su carrito de compras.
- El usuario decide agregar otro producto y lo agrega a su carrito de compras.
- El usuario decide eliminar un producto de su carrito de compras.
- El usuario procede a la página de pago y proporciona los detalles de su tarjeta de crédito.
- El sistema procesa la transacción y muestra una página de confirmación de compra al usuario.
- El usuario confirma la compra y el sistema muestra un mensaje de agradecimiento.

En este escenario, podemos identificar varios casos de uso, como "Buscar productos", "Agregar producto al carrito", "Eliminar producto del carrito", "Realizar pago" y "Confirmar compra". Además, podemos identificar al actor "Usuario", que interactúa con el sistema en varias etapas del proceso.

Los escenarios permiten a los desarrolladores comprender cómo un usuario interactúa con el sistema en diferentes etapas del proceso. Por lo tanto, es importante que los escenarios sean precisos y detallados. Esto ayuda a los desarrolladores a identificar casos de uso adicionales que pueden ser necesarios para mejorar la experiencia del usuario.

Otro beneficio de los escenarios es que pueden ayudar a los desarrolladores a identificar problemas potenciales en el sistema. Por ejemplo, en el escenario anterior, si un usuario intenta agregar un producto que ya está en su carrito de compras, el sistema debe mostrar un mensaje de error. Si este problema no se identifica en el escenario, podría causar confusión y frustración al usuario.

Además, los escenarios pueden ser útiles para el diseño de pruebas de usuario y para asegurarse de que el sistema funciona como se espera en diferentes situaciones.

Los escenarios más importantes deben ser escritos, ya que proporcionan información valiosa para el diseño y desarrollo del sistema. Estos escenarios clave ayudan a asegurar que los requisitos esenciales del usuario estén cubiertos y se minimicen los errores.

Es importante recordar que los escenarios no representan una secuencia fija de eventos, sino que son una descripción general de las diferentes interacciones posibles entre los casos de uso y los actores del sistema. Por lo tanto, los escenarios pueden variar según las necesidades específicas del sistema y de los usuarios.

Ejemplo:
Escenario importante para el caso de uso de Compra en línea:

El usuario ingresa al sitio web de compras. El sistema muestra los diferentes productos disponibles. El usuario navega por los productos y selecciona uno para agregarlo a su carrito de compras. El sistema añade el producto al carrito y muestra el contenido actualizado del carrito. El usuario decide proceder al pago. El sistema solicita al usuario que proporcione los detalles de su tarjeta de crédito. El usuario ingresa la información requerida. El sistema verifica la validez de la tarjeta de crédito y procesa la transacción. Tras la aprobación del pago, el sistema muestra una página de confirmación de compra al usuario. El usuario confirma la compra y el sistema muestra un mensaje de agradecimiento. Finalmente, el sistema envía un correo electrónico de confirmación al usuario y termina el caso de uso.

Desarrollo de Casos de Uso

ADVERTENCIA: Esto no es parte de UML, no es una notación estándar.
Aunque el desarrollo de casos de uso no es parte del Lenguaje Unificado de Modelado (UML), sigue siendo una herramienta valiosa en el proceso de diseño de software. A continuación, se presenta un formulario para el desarrollo del caso de uso "Compra en línea":

Ítem	Descripción
Nombre del caso de uso	Compra en línea
Actores involucrados	Usuario, Sistema
Prioridad	Alta
Estado del desarrollo	En progreso
Puntos de extensión/inclusión	Registro e inicio de sesión de usuario Aplicación de cupones de descuento Elección de dirección de envío
Precondiciones	El usuario ha ingresado al sitio web de compras
Postcondiciones	El usuario ha completado una compra y ha recibido un correo electrónico de confirmación
Flujos de eventos	1. El usuario navega por los productos 2. El usuario selecciona un producto y lo agrega al carrito 3. El usuario procede al pago 4. El usuario proporciona detalles de tarjeta de crédito 5. El sistema verifica la tarjeta de crédito y procesa la transacción 6. El usuario confirma la compra y recibe un mensaje de agradecimiento 7. El sistema envía un correo electrónico de confirmación al usuario
Restricciones de performance	El tiempo de respuesta del sistema debe ser rápido para garantizar una experiencia de compra fluida y sin interrupciones
Frecuencia de ejecución	Múltiples veces al día, según la cantidad de usuarios que realicen compras en línea

Este formulario proporciona una descripción detallada del caso de uso "Compra en línea" e incluye información relevante para el diseño y desarrollo del sistema.

Clasificación de Casos de Uso

La clasificación de casos de uso es una práctica útil en el análisis y diseño de sistemas, ya que ayuda a los desarrolladores a priorizar los requerimientos y enfocarse en las funcionalidades más importantes. Un enfoque popular es clasificar los casos de uso según su importancia. El libro "UML Gota a Gota" de Martin Fowler es una referencia común en el mundo de la ingeniería de software y aborda este tema en el contexto del Lenguaje Unificado de Modelado (UML).

Según este enfoque, los casos de uso se pueden clasificar en tres categorías principales:

Imprescindibles: Estos casos de uso son esenciales para el funcionamiento básico del sistema y deben ser implementados para que el sistema sea operativo. Sin ellos, el sistema no cumpliría con su propósito principal y, por lo tanto, no sería útil para los usuarios.

Ejemplo concreto: En un sistema de comercio electrónico, un caso de uso imprescindible sería "Realizar compra", ya que sin esta funcionalidad, los usuarios no podrían adquirir los productos ofrecidos en el sitio web.

Importantes pero no inmediatos: Estos casos de uso son importantes para mejorar la experiencia del usuario y proporcionar funcionalidades adicionales, pero no son críticos para el funcionamiento básico del sistema. Por lo tanto, pueden ser implementados en una etapa posterior del desarrollo.

Ejemplo concreto: En un sistema de comercio electrónico, un caso de uso importante pero no inmediato podría ser "Recomendaciones personalizadas", ya que, aunque esta funcionalidad mejora la experiencia del usuario al sugerir productos relevantes, no es crítica para el funcionamiento básico del sistema.

Complementarios: Estos casos de uso son útiles para agregar funcionalidades extra y mejorar la experiencia del usuario, pero no son esenciales para el sistema. Pueden ser implementados si el tiempo y los recursos lo permiten, pero no deben ser la prioridad principal.

Ejemplo concreto: En un sistema de comercio electrónico, un caso de uso complementario podría ser "Chat en vivo con soporte al cliente", ya que, aunque esta funcionalidad proporciona un canal adicional de atención al cliente, no es crucial para el funcionamiento básico del sistema.

Al clasificar los casos de uso según su importancia, los desarrolladores pueden enfocarse en implementar primero las funcionalidades esenciales, asegurando que el sistema cumpla con su propósito principal. Posteriormente, se pueden agregar funcionalidades adicionales para mejorar la experiencia del usuario y enriquecer el sistema.

> *"...demasiados casos de uso pueden acabar siendo abrumadores. Por el momento no creo que haya una respuesta correcta, de modo que le recomiendo que sea flexible y trabaje con lo que le resulte más cómodo"*
>
> *Martin Fowler*

Actividades

1. *Tareas:* Las presentes tareas deben realizarse en orden -o no- prestando atención a la relación entre las lecturas realizadas.
    a. Leer el Prefacio de UML Gota a Gota
    b. Leer la sección “Cómo llegamos hasta aquí” de UML Gota a Gota y crea un documento que expanda lo escrito hasta nuestros días.
    c. Aprende UML utilizando Visual Paradigm https://www.visual-paradigm.com/guide/ y crea un resumen conciso de los conceptos más relevantes trabajados en clase. Explora los tutoriales y guías disponibles en el sitio web, enfocándote en los temas que se relacionan con los conceptos discutidos en clase. A medida que avances, toma notas y organiza un resumen estructurado de los conceptos clave de UML. Finalmente, comparte y discute tus hallazgos con tus compañeros de clase para mejorar la comprensión de UML y su aplicación en proyectos futuros.

2. Considere el ejercicio 2 del teórico 2.1:

    Se pide:

    ~~a) Identifique las clases y objetos candidatos presentes en el documento anterior.~~

    ~~b) Elija algunos compañeros y suponga que los docentes del curso son los usuarios del sistema del comercio. Consúltelos a efectos de elaborar un glosario de todos aquellos términos que usted considere ambiguos o necesario precisar.~~

    c) Identifique los casos de uso del sistema y elabore un diagrama de casos de uso.

    d) Escriba el escenario principal correspondiente a cada uno de ellos.

    e) Escriba el desarrollo expandido para cada caso de uso del sistema (o sea, para cada uno de ellos, escriba una ficha que detalle actores, puntos de extensión y de inclusión, precondiciones, poscondiciones, flujo principal y flujos alternativos).

Próximamente >> Análisis Orientado a Objetos

- *Análisis Orientado a Objetos*
- *Lo que compete al análisis*
- *Modelo Conceptual de UML para el análisis*

Bibliografía

Fowler, M. (1999). UML Gota a Gota. Madrid: Addison-Wesley Iberoamericana.