SERVICIO NACIONAL DE APRENDIZAJE SENA

Centro: Centro de Diseño e Innovación Tecnológica Industrial

Ficha: 3235906

Programa de Formación: TECNOLOGÍA EN ANÁLISIS Y DESARROLLO DE SOFTWARE

Nombre del Aprendiz: Laura Barona Saavedra

Nombre del Instructor: Felmaber Garzón Muñoz

Fecha: 22 de febrero 2026

INTRODUCCION

El desarrollo de software requiere herramientas que permitan organizar, visualizar y estructurar los sistemas antes de su implementación. El Lenguaje Unificado de Modelado (UML) surge como un estándar que facilita la representación gráfica de sistemas, permitiendo comprender su arquitectura, comportamiento e interacción entre sus componentes.

En este informe se presentan las bases conceptuales del UML, su importancia en el desarrollo de software, un resumen elaborado con palabras propias y un glosario con la terminología más relevante. Asimismo, se aborda de manera general la relación entre UML y los patrones de diseño, como complemento en la construcción de soluciones de software eficientes.

BASES CONCEPTUALES DEL LENGUAJE UNIFICADO DE MODELADO (UML)

¿Qué es UML?

UML (Unified Modeling Language) es un lenguaje gráfico estandarizado que se utiliza para modelar sistemas de software. Permite especificar, visualizar, construir y documentar los diferentes componentes de un sistema mediante diagramas estructurados.

No es un lenguaje de programación, sino una herramienta visual que facilita el análisis y diseño antes de escribir código.

Objetivos principales de UML

- Visualizar el diseño de un sistema.
- Especificar la estructura y comportamiento.
- Documentar decisiones técnicas.
- Facilitar la comunicación entre desarrolladores y stakeholders.
- Reducir errores antes de la implementación.

Tipos de Diagramas UML

UML se divide principalmente en diagramas estructurales y diagramas de comportamiento.

A. Diagramas Estructurales

Representan la estructura estática del sistema.

- Diagrama de clases
- Diagrama de objetos
- Diagrama de componentes
- Diagrama de despliegue
- Diagrama de paquetes

B. Diagramas de Comportamiento

Representan el comportamiento dinámico del sistema.

- Diagrama de casos de uso
- Diagrama de secuencia
- Diagrama de actividades
- Diagrama de estados
- Diagrama de comunicación

Elementos básicos de UML

Los principales elementos que componen UML son:

1. Elementos estructurales

Representan las partes estáticas del sistema, como clases, interfaces y componentes.

2. Elementos de comportamiento

Representan procesos o interacciones, como actividades y casos de uso.

3. Relaciones

Conectan los elementos entre sí. Las principales relaciones son:

- Asociación
- Agregación
- Composición
- Herencia
- Dependencia

UML Y PATRONES DE DISEÑO

Los patrones de diseño son soluciones reutilizables a problemas comunes en el desarrollo de software. UML permite representar gráficamente estos patrones para facilitar su comprensión e implementación.

Por ejemplo:

- El patrón Singleton puede representarse mediante un diagrama de clases.
- El patrón Observer se modela mostrando relaciones de dependencia entre objetos.
- El patrón MVC puede visualizarse mediante diagramas de componentes o clases.

UML no define patrones, pero sí permite documentarlos y modelarlos correctamente dentro de una arquitectura de software.

RESUMEN SOBRE UML

UML es una herramienta visual que ayuda a organizar las ideas antes de desarrollar un sistema de software. Permite representar gráficamente cómo está estructurado el sistema y cómo interactúan sus partes. Gracias a sus diagramas, los desarrolladores pueden analizar mejor los requisitos, evitar errores y comunicar de forma clara el diseño del proyecto.

Considero que UML es fundamental en la etapa de análisis y diseño porque permite entender el sistema antes de programarlo. Además, facilita el trabajo en equipo, ya que todos pueden interpretar los diagramas como un lenguaje común.

GLOSARIO DE TERMINOLOGÍA UML

Actor: Entidad externa que interactúa con el sistema (usuario u otro sistema).

Asociación: Relación entre dos clases que indica que están conectadas.

Agregación: Tipo de relación donde un objeto contiene a otros, pero estos pueden existir independientemente.

Composición: Relación fuerte donde los objetos contenidos dependen completamente del objeto principal.

Clase: Representación de un conjunto de objetos con atributos y métodos similares.

Caso de uso: Representación de una funcionalidad del sistema desde la perspectiva del usuario.

Diagrama de clases: Representa la estructura del sistema mostrando clases y relaciones.

Diagrama de secuencia: Muestra cómo interactúan los objetos en el tiempo.

Herencia: Relación donde una clase hija hereda atributos y métodos de una clase padre.

Interfaz: Define un conjunto de métodos que una clase debe implementar.

Modelo: Representación simplificada de un sistema real.

Objeto: Instancia de una clase.

Patrón de diseño: Solución reutilizable a un problema común en el diseño de software.

UML: Lenguaje gráfico estandarizado para modelar sistemas.

CONCLUSIÓN

UML es una herramienta esencial en el desarrollo de software, ya que permite estructurar y representar de forma clara los sistemas antes de su implementación. Su uso mejora la comunicación entre los miembros del equipo y reduce errores en etapas tempranas del proyecto.

Además, UML complementa el uso de patrones de diseño, facilitando su representación y documentación. En conclusión, dominar UML fortalece las competencias en análisis y diseño de software, aportando mayor organización y calidad en los proyectos desarrollados.