**UML - Diagramas de Comportamiento**

Diego Alejandro Aguilera Díaz 160004901

Andrés Felipe Carrasquilla Gutiérrez 160004911

Yohan Stevens Piñarte Diaz 160004926

Jhon Franklin Sandoval Segura 160004936

Juan Carlos Tobon Montoya 160004937

Jhon Matthew Vanegas Buitrago 160004940

***Resumen– UML (Unified Modeling Language) es un lenguaje estándar utilizado para modelar sistemas de software, creado en 1997 por Grady Booch, James Rumbaugh e Ivar Jacobson. Su propósito es ofrecer una notación común que facilite la comunicación entre los miembros de un equipo de desarrollo, utilizando diagramas visuales que representan tanto aspectos conceptuales como concretos de un sistema.***

1. **INTRODUCCIÓN**

Este documento describe los requerimientos del sistema de modelado y diseño de software utilizando UML en el desarrollo de aplicaciones. El objetivo es proporcionar un lenguaje visual común que facilite la creación, comunicación y comprensión de sistemas de software complejos. A través de diagramas como casos de uso, clases y actividades, el sistema debe permitir representar de manera clara tanto la estructura como el comportamiento del software, promoviendo la colaboración eficaz entre los miembros del equipo de desarrollo y garantizando la consistencia en la documentación del sistema.

1. **MARCO TEÓRICO**

El Unified Modeling Language (UML) es un lenguaje estándar de modelado gráfico utilizado para la especificación, visualización, construcción y documentación de los componentes de un sistema de software. Desarrollado en la década de 1990 por Grady Booch, James Rumbaugh e Ivar Jacobson, UML busca ofrecer una notación común que facilite la comunicación y comprensión de los diferentes aspectos de un sistema, desde su estructura hasta su comportamiento.

UML no es un lenguaje de programación, sino una herramienta para representar visualmente los modelos de un sistema a través de diversos tipos de diagramas. Estos diagramas están diseñados para abordar diferentes perspectivas del sistema, como la estructura estática, las interacciones dinámicas y los comportamientos de los componentes. Entre los diagramas más utilizados en UML se encuentran los diagramas de clases, diagramas de casos de uso, diagramas de secuencia y diagramas de actividades, cada uno con un propósito específico en la modelización del sistema.

El uso de UML proporciona varias ventajas en el desarrollo de software. Primero, permite una representación clara y comprensible de un sistema, lo que facilita la comunicación entre los desarrolladores, diseñadores y otras partes interesadas. Además, UML contribuye a la documentación estandarizada, lo que es fundamental para la evolución del sistema a lo largo del tiempo. La capacidad de representar sistemas complejos de manera visual también mejora la identificación de errores y la toma de decisiones durante las fases de diseño y desarrollo.

La evolución de UML ha permitido la incorporación de nuevos diagramas y características para adaptarse a las necesidades cambiantes de los proyectos de software. La versión 2.0, lanzada en 2001, introdujo mejoras significativas en la notación, haciendo que el lenguaje fuera más preciso y flexible. A lo largo de los años, se han incluido herramientas de personalización y extensión, como perfiles UML, que permiten adaptar el lenguaje a contextos específicos.

El uso de UML no solo se limita a equipos de desarrollo de software, sino que también es útil para otros actores del proceso de desarrollo, como gerentes de proyectos y analistas de negocio. A través de la representación visual, se facilita la comprensión del funcionamiento del sistema y se promueve una colaboración más eficaz.

En resumen, UML es un lenguaje poderoso que ha demostrado ser esencial en el diseño y desarrollo de sistemas de software. Su capacidad para representar de manera visual y estandarizada los diferentes aspectos de un sistema facilita la comunicación y contribuye al éxito de los proyectos de software.

1. **Objetivos del proyecto**

**Objetivo General:** Desarrollar un sistema de modelado y diseño de software utilizando UML, que permita representar de manera visual y comprensible los aspectos estructurales y comportamentales de un sistema de software, facilitando la comunicación y colaboración entre los miembros del equipo de desarrollo.

**Objetivos Específicos:**

Diseñar diagramas UML adecuados para representar diferentes perspectivas del sistema, como diagramas de clases, casos de uso y actividades, que reflejen la arquitectura y el comportamiento del software.

Establecer una metodología de trabajo basada en UML para que todos los miembros del equipo utilicen un lenguaje común en el proceso de diseño y desarrollo del software.

Fomentar la colaboración eficiente entre los desarrolladores y otras partes interesadas mediante el uso de diagramas visuales, mejorando la comprensión del sistema.

Implementar un sistema de documentación estandarizada que facilite la revisión, mantenimiento y evolución del software a lo largo del tiempo.

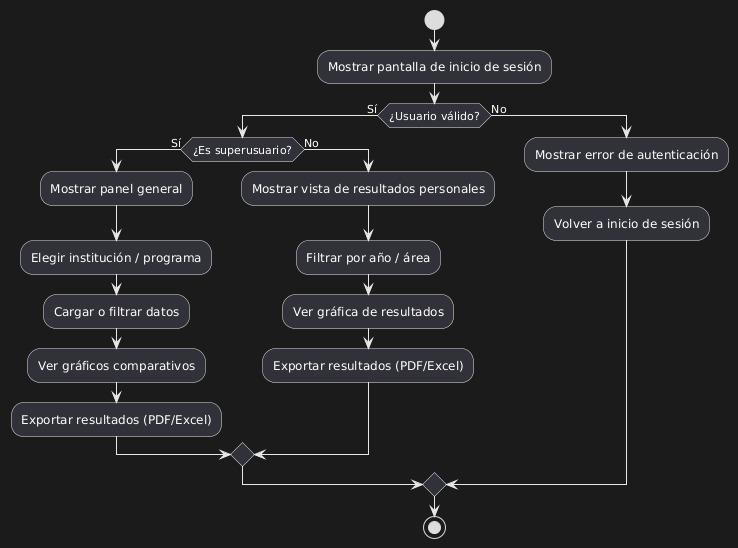
Proporcionar herramientas de personalización que permitan adaptar UML a las necesidades específicas del proyecto, asegurando la flexibilidad del sistema para futuros ajustes.

Validar la eficacia de UML como herramienta de modelado a través de su aplicación práctica en el desarrollo de un sistema real, midiendo la mejora en la comunicación, reducción de errores y eficiencia del proceso de desarrollo.

1. **Estado de resultados**

**Diagrama de Actividades**

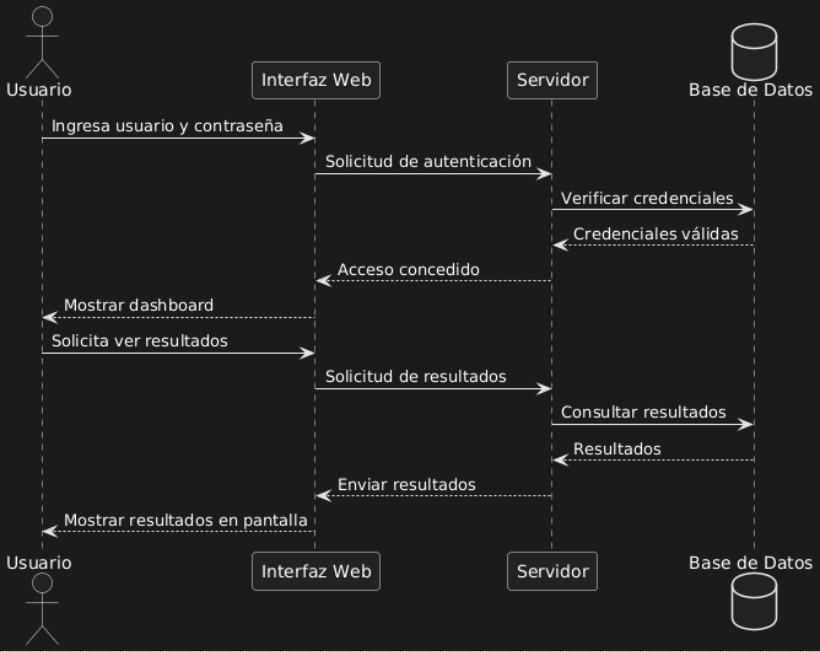
El diagrama de actividades es un tipo de diagrama de comportamiento utilizado en UML para representar el flujo de trabajo de un proceso o sistema. Este diagrama muestra la secuencia de acciones y decisiones que ocurren durante la ejecución de una actividad, permitiendo visualizar claramente cómo se desarrollan los procesos en el sistema.



**Figura 1**. Diagrama de actividades

**Diagrama de Secuencia**

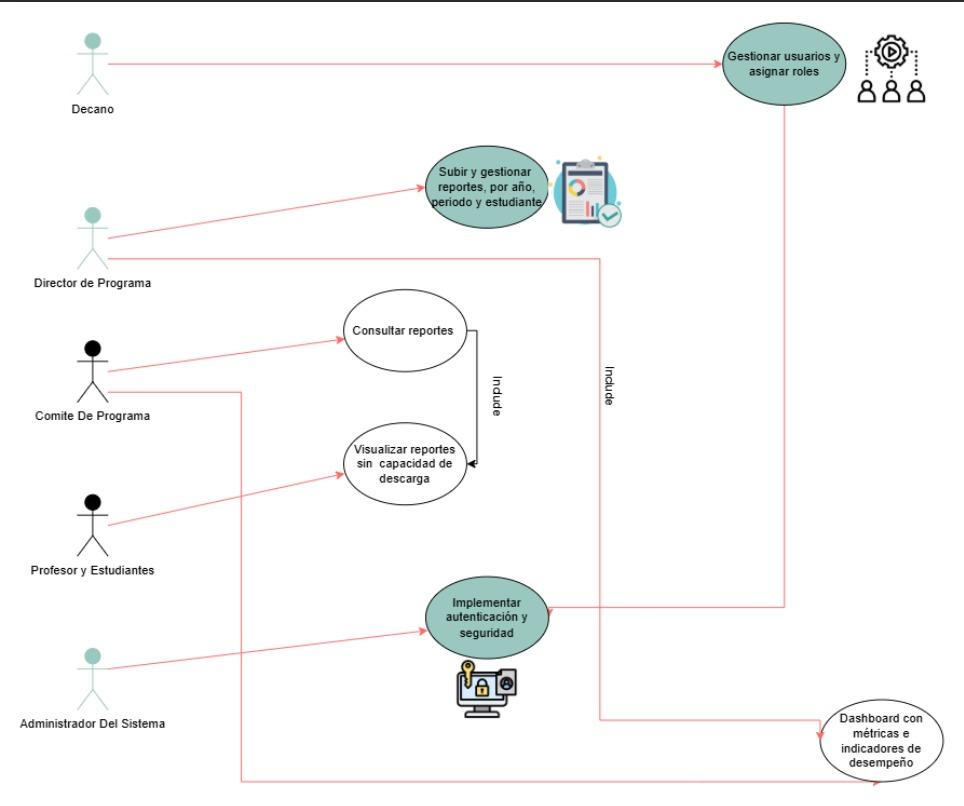
El diagrama de secuencia es un tipo de diagrama de comportamiento en UML que ilustra cómo interactúan los objetos en un sistema a lo largo del tiempo. Este diagrama se centra en el orden cronológico de los mensajes intercambiados entre los objetos o actores, lo cual permite visualizar claramente el flujo de ejecución de un proceso específico.



**Figura 2**. Diagrama de secuencia

**Diagrama de Casos de Uso**

En este proyecto, el diagrama de casos de uso muestra las diferentes interacciones que pueden tener los usuarios con el sistema de modelado, tales como crear modelos UML, editar diagramas, guardar cambios o generar reportes. También se identifican actores como el "Usuario", "Administrador" o "Desarrollador", quienes tienen distintos niveles de interacción con el sistema.

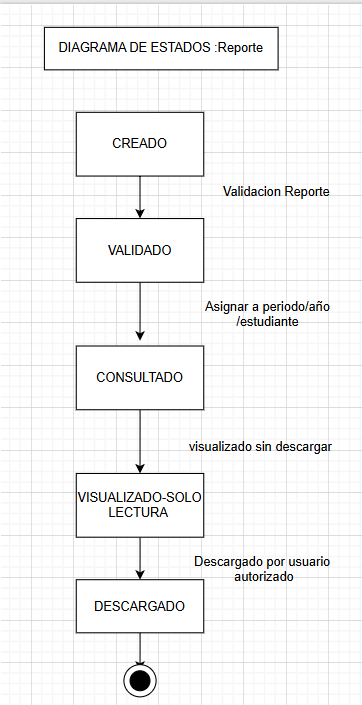


**Figura 3**. Diagrama de casos de uso

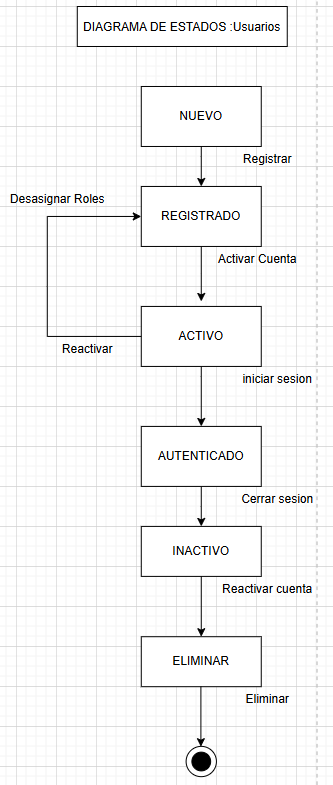
**Diagrama de Estados**

El diagrama de estados (o de máquina de estados) es un tipo de diagrama de comportamiento en UML que muestra los diferentes estados por los que pasa un objeto a lo largo de su ciclo de vida, así como los eventos o condiciones que causan la transición de un estado a otro.

En el contexto de este proyecto, el diagrama de estados se utiliza para representar cómo cambia el estado de un modelo UML desde su creación hasta su finalización. Por ejemplo, un modelo puede comenzar en estado “Borrador”, luego pasar a “En edición”, “Revisado”, y finalmente “Guardado” o “Eliminado”, dependiendo de las acciones realizadas por el usuario.



**Figura 4.** Diagrama de Estados : Reportes de la aplicación



**Figura 5**. Diagrama de Estados : Usuarios en la aplicación

1. **Conclusión**

El uso de UML en el desarrollo de software representa una gran ventaja al momento de planificar, diseñar y comunicar ideas dentro de un equipo. Gracias a su lenguaje visual y estandarizado, es posible representar procesos, estructuras y comportamientos del sistema de forma clara y entendible para todos los involucrados.

Durante este proyecto se evidenció cómo los diagramas, como el de actividades y el de secuencia, permiten organizar mejor las ideas, entender el flujo de trabajo y detectar posibles errores antes de escribir una sola línea de código. Además, al utilizar UML, se facilita la documentación del sistema y se mejora la colaboración entre los integrantes del equipo.

En resumen, UML no solo ayuda a crear mejores sistemas, sino que también promueve una forma más ordenada y profesional de desarrollar software, siendo una herramienta esencial para cualquier ingeniero en formación.

1. **Referencias bibliográficas**

* <https://app.diagrams.net/>
* Todos los diagramas UML. Teoría y ejemplos. (2025, abril20).DiagramasUML.com.https://diagramasuml.com/
* luis. (2025, abril 20). Diagrama de Comportamiento. Administrar proyectos. https://administrarproyectos.com/diagrama-de-comportamiento/