|  |  |  |
| --- | --- | --- |
| de | Carátula para entrega de prácticas | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Ing. Jorge Ángel Hernández López |
| *Asignatura:* | Programación Orientada a Objetos |
| *Grupo:* | 04 |
| *No. de práctica(s):* | 09 |
| *Integrante(s):* | Barrera Treviño José Gerardo  Muñoz San Agustin Victoria Monserrat  Olvera Bravo Cynthia Carolina  Leocadio Chávez Rodrigo  Velasco Garcia Santiago |
| *No. de lista o brigada:* | Brigada 01 |
| *Semestre:* | 2024-1 |
| *Fecha de entrega:* | 30 de octubre de 2023 |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_\_\_

# **Introducción**

La práctica se titula utilerías y clases de uso general donde se abordan los siguientes temas:

* Diagramas estáticos o de comportamiento:

Un gráfico que ilustra la disposición fija de un modelo, destacando los elementos presentes, como clases y tipos, la organización interna de dichos elementos y sus conexiones mutuas. Estos gráficos pueden emplearse para elaborar representaciones conceptuales de las ideas del mundo real y sus interconexiones, o diagramas de clases que desglosan un sistema de software en sus componentes fundamentales.

* Diagramas dinámicos:

El modelo dinámico engloba los aspectos temporales y los cambios que ocurren en un sistema, tanto en los objetos como en sus relaciones a lo largo del tiempo. Este modelo se utiliza para describir el control en el sistema, es decir, las secuencias de acciones que se desencadenan en respuesta a estímulos externos, sin profundizar en los detalles de lo que realizan esas acciones, en qué objeto operan o cómo se llevan a cabo. Los objetos se comunican entre sí a través del intercambio de mensajes, y cuando un conjunto de objetos se comunica para cumplir una función específica, se denomina interacción.

* Diagrama de casos de uso:

El diagrama de casos de uso es un tipo de diagrama de comportamiento en el marco del Lenguaje de Modelado Unificado (UML, por sus siglas en inglés), y se utiliza para representar tanto procesos empresariales como sistemas y procesos de programación orientada a objetos. Es importante destacar que UML no es un lenguaje de programación, sino más bien un lenguaje de modelado, que proporciona un enfoque estandarizado para representar sistemas ya planificados o en existencia. En este tipo de diagrama, se organiza y establecen las relaciones entre todos los objetos involucrados en el sistema.

* Diagrama de clase:

Los diagramas de clases son altamente beneficiosos dentro de UML, ya que representan de manera evidente la estructura de un sistema específico al definir sus clases, atributos, operaciones y las relaciones entre los objetos. A través de nuestro software de generación de diagramas UML, la tarea de crear estos diagramas no resulta tan compleja como podría pensarse. La siguiente guía te orientará en la comprensión, planificación y creación de tu propio diagrama de clases.

* Diagrama de objetos:

Un diagrama de objetos en UML representa una instancia específica de un diagrama de clases en un punto preciso en el tiempo. Visualmente, comparte muchas similitudes con el diagrama de clases.

El diagrama de objetos se centra en los atributos de un conjunto de objetos y en cómo estos objetos están interconectados. Por ejemplo, en el siguiente diagrama de objetos, las tres cuentas bancarias están asociadas al mismo banco. Los títulos de clase muestran los tipos de cuentas (ahorro, corriente y tarjeta de crédito) que un cliente podría poseer en ese banco en particular. Los atributos de clase varían según el tipo de cuenta; por ejemplo, el objeto de tarjeta de crédito tiene un límite de crédito, mientras que las cuentas de ahorro y corriente tienen tasas de interés diferentes.

* Diagrama de estados:

Un diagrama de estado UML, también conocido como diagrama de estado, diagrama de transición de estados o diagrama de máquina de estados, muestra los estados que atraviesa una máquina de estados finitos. Esta máquina de estados finitos es un modelo de comportamiento que incluye acciones, estados y transiciones hacia otros estados. El diagrama representa un estado inicial y un estado final, además de al menos un estado intermedio para cada objeto. De esta manera, el diagrama de estado es una herramienta que permite representar el ciclo de vida completo de sistemas, subsistemas o componentes, como por ejemplo, una máquina de café, un lector de libros electrónicos o un componente tecnológico de un vehículo.

Este tipo de diagrama de estado es uno de los 14 diagramas definidos en el Lenguaje de Modelado Unificado (UML) y en el Lenguaje de Modelado de Sistemas (SysML). El concepto se originó en un artículo de David Harel en 1987 llamado "Statecharts: A Visual Formalism for Complex Systems". Otros tipos de diagramas UML incluyen el diagrama de casos de uso y el diagrama de componentes, entre otros.

* Diagrama de actividades:

El Lenguaje Unificado de Modelado (UML) abarca varios subconjuntos de diagramas, entre los que se incluyen los diagramas de estructuras, los diagramas de interacción y los diagramas de comportamiento. Los diagramas de actividades, junto con los diagramas de casos de uso y los diagramas de máquina de estados, se clasifican como diagramas de comportamiento, ya que se encargan de describir lo que debe ocurrir en el sistema que se está modelando.

Dado que las partes interesadas gestionan numerosos asuntos, la comunicación clara y concisa resulta fundamental. Los diagramas de actividades facilitan que las personas en las áreas de negocios y desarrollo de una organización trabajen juntas para comprender un mismo proceso y su comportamiento. Para crear un diagrama de actividades, se utilizan una serie de símbolos especializados, que incluyen elementos para representar pasos iniciales, finales, fusiones y recepciones en el flujo. Estos conceptos se abordarán con mayor detalle en esta guía sobre diagramas de actividades.

* Diagrama de interacción:

Como su nombre indica, un diagrama de interacción es un tipo de diagrama UML diseñado para capturar el comportamiento interactivo de un sistema. Estos diagramas se centran en describir el flujo de mensajes en un sistema y proporcionan contexto para una o más "líneas de vida" dentro del sistema. Además, los diagramas de interacción pueden utilizarse para representar secuencias ordenadas de eventos dentro de un sistema y funcionan como una herramienta para visualizar datos en tiempo real a través de UML.

* Diagrama de secuencia:

Para comprender qué es un diagrama de secuencia, es fundamental tener en cuenta la función del Lenguaje Unificado de Modelado, comúnmente conocido como UML. El UML es un conjunto de herramientas de modelado que proporciona orientación para la creación y notación de diversos tipos de diagramas, incluyendo diagramas de comportamiento, diagramas de interacción y diagramas de estructuras.

Un diagrama de secuencia pertenece a la categoría de diagramas de interacción, ya que detalla cómo un grupo de objetos trabaja juntos y en qué secuencia. Tanto los desarrolladores de software como los profesionales de negocios utilizan estos diagramas para comprender los requisitos de un sistema nuevo o documentar un proceso existente. A veces, estos diagramas se conocen como diagramas de eventos o escenarios de eventos. Es importante señalar que existen dos tipos de diagramas de secuencia: los diagramas UML y los diagramas basados en código. Los últimos se derivan directamente del código de programación y no se abordarán en esta guía. El software de diagramas UML de Lucidchart incluye todas las figuras y funciones necesarias para crear ambos tipos de diagramas de secuencia.

* Diagrama de comunicación:

Se trata de un diagrama de clases que incorpora tanto roles de clasificador como roles de asociación en lugar de limitarse a clasificadores y asociaciones. En este tipo de diagrama, las instancias, las colaboraciones y los objetos se vinculan a los roles de clasificador, mientras que los enlaces se relacionan con los roles de asociación. Los roles de asociación también pueden ser desempeñados por diversos tipos de enlaces temporales, como argumentos de procedimientos o variables locales de un procedimiento. Los símbolos de los enlaces pueden llevar estereotipos para indicar enlaces temporales (por ejemplo, <<parameter>> o <<local>>) o llamadas al mismo objeto (<<self>>).

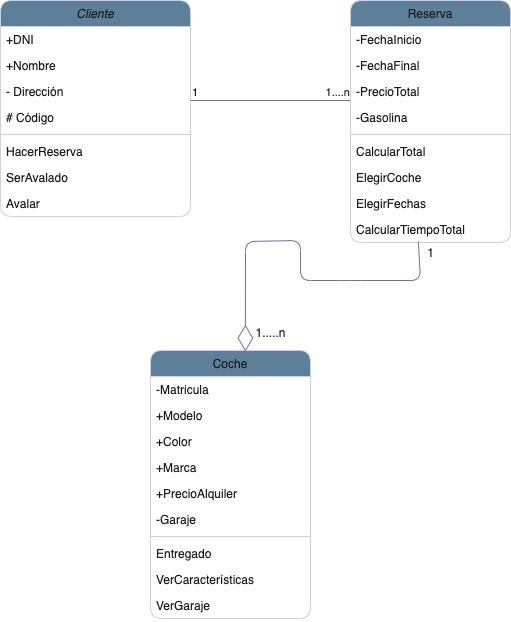
Este tipo de diagrama tiene la intención de representar, desde una perspectiva tanto estática como dinámica, los objetos involucrados en la implementación de una función de una aplicación. A diferencia de los diagramas de secuencia, estos diagramas pueden mostrar el contexto de la operación, incluyendo qué objetos son atributos, cuáles son temporales, entre otros, así como los ciclos en la ejecución del sistema.

# **Desarrollo:**

1. Se requiere diseñar un sistema que me permita llevar el control de la renta de automóviles.

* Un cliente puede tener en un momento dado hechas varias reservas.
* De cada cliente se desean almacenar su DNI, nombre, dirección y teléfono. Además, dos clientes se diferencian por un código único.
* Cada cliente puede ser avalado por otro cliente de la empresa.
* Una reserva la realiza un único cliente, pero puede involucrar varios coches.
* Es importante registrar la fecha de inicio y final de la reserva, el precio del alquiler de cada uno de los coches, los litros de gasolina en el depósito en el momento de realizar la reserva, el precio total de la reserva y un indicador de si el coche o los coches han sido entregados.
* Todo coche tiene siempre asignado un determinado garaje que no puede cambiar. De cada coche se requiere la matricula, el modelo el color y la marca.

**Solución:**

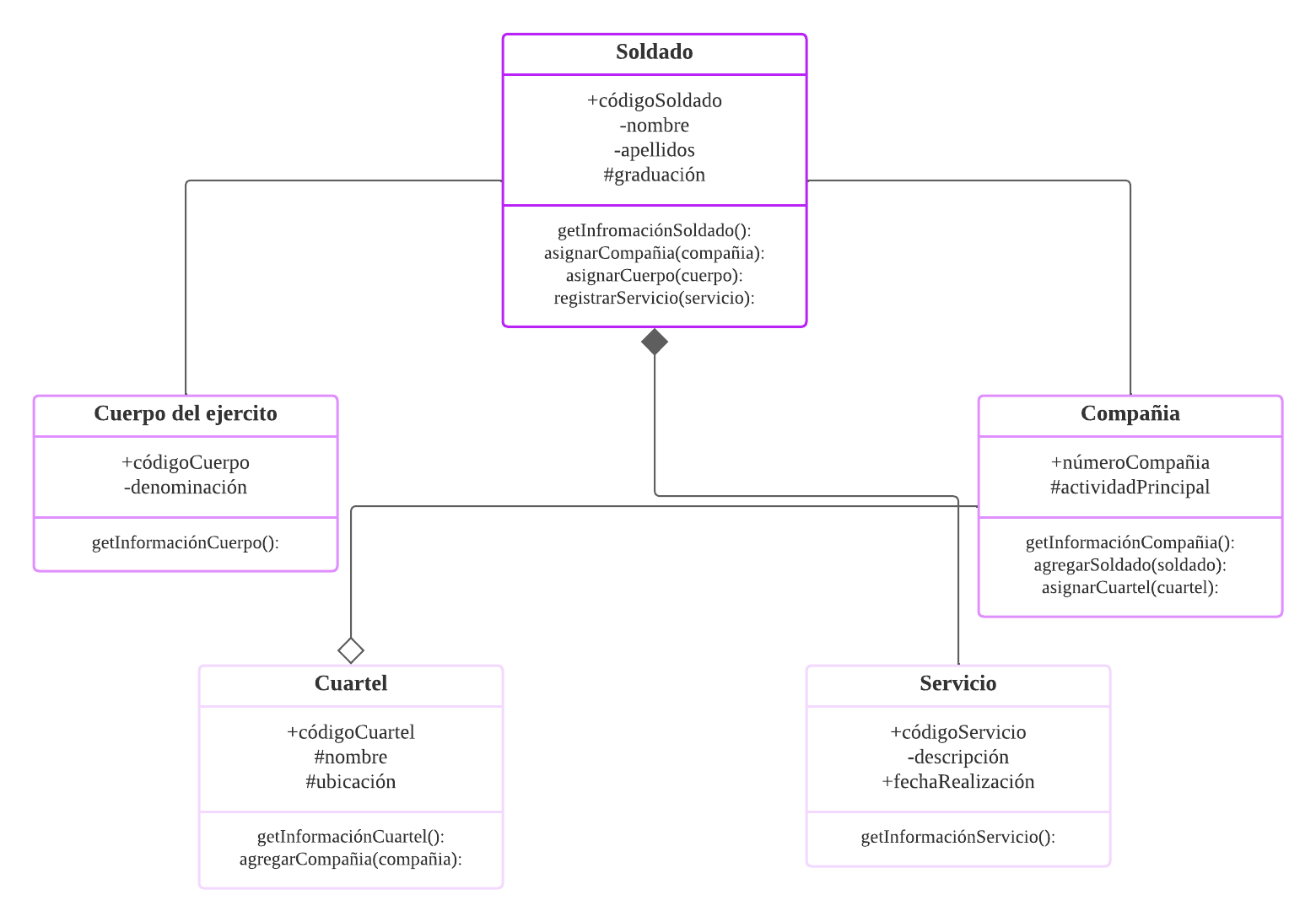


1. El colegio militar desea diseñar un sistema para llevar un cierto control de los soldados que realizan el servicio militar, generar el diagrama de clases considerando lo siguiente:

* Un soldado se define por su código de soldado (único), su nombre y apellidos, y su graduación.
* Existen varios cuarteles, cada uno se define por su código de cuartel, nombre y ubicación.
* • Hay que tener en cuenta que existen diferentes Cuerpos del Ejército (Infantería, Artillería, Armada, …), y cada uno se define por un código de Cuerpo y denominación.
* Los soldados están agrupados en compañías, siendo significativa para cada una de éstas, el número de compañía y la actividad principal que realiza.
* Se desea controlar los servicios que realizan los soldados (guardias, imaginarias, cuarteleros, ...), y se definen por el código de servicio y descripción.

Consideraciones adicionales de diseño:

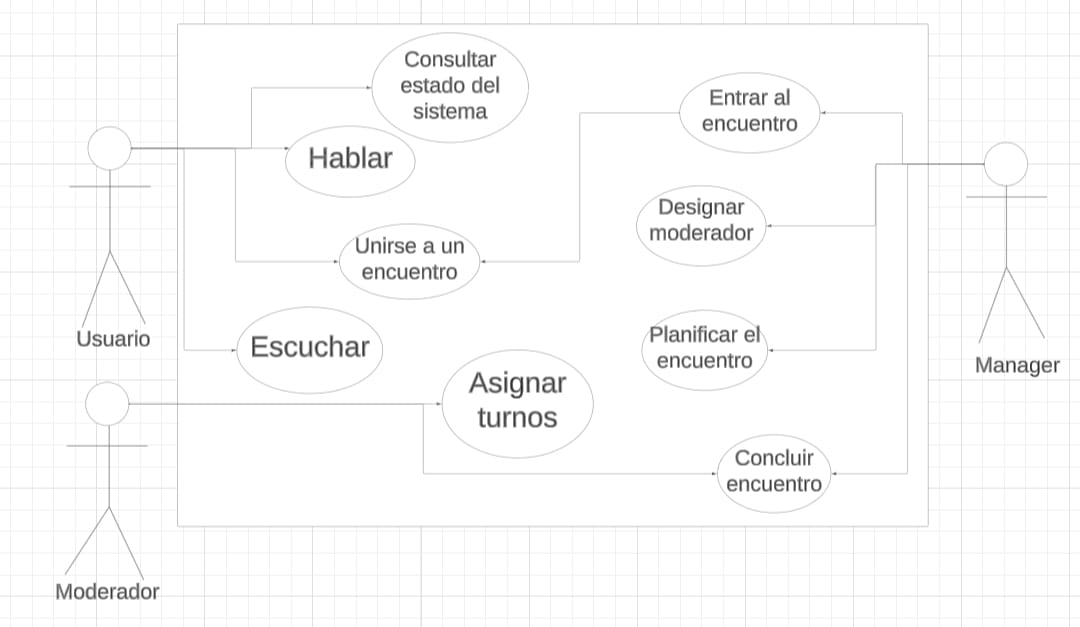
* Un soldado pertenece a un único cuerpo y a una única compañía, durante todo el servicio militar. A una compañía pueden pertenecer soldados de diferentes cuerpos, no habiendo relación directa entre compañías y cuerpos.
* Los soldados de una misma compañía pueden estar destinados en diferentes cuarteles, es decir, una compañía puede estar ubicada en varios cuarteles, y en un cuartel puede haber varias compañías. Eso sí, un soldado sólo está en un cuartel.
* Un soldado realiza varios servicios a lo largo de la milicia. Un mismo servicio puede ser realizado por más de un soldado (con independencia de la compañía), siendo significativa la fecha de realización.

**Solución:**

1. Se desea desarrollar un sistema de encuentros virtuales (parecido a un chat), generar el diagrama de casos de uso, teniendo en cuenta lo siguiente:

* Cuando se conecta al servidor, un usuario puede entrar o salir de un encuentro.
* Cada encuentro tiene un manager.
* El manager es el usuario que ha planificado el encuentro (el nombre del encuentro, la agenda del encuentro y el moderador del encuentro).
* Cada encuentro puede tener también un moderador designado por el manager.
* La misión del moderador es asignar los turnos de palabra para que los usuarios hablen.
* El moderador también podrá dar por concluido el encuentro en cualquier momento.
* En cualquier momento un usuario puede consultar el estado del sistema, por ejemplo, los encuentros planeados y su información.

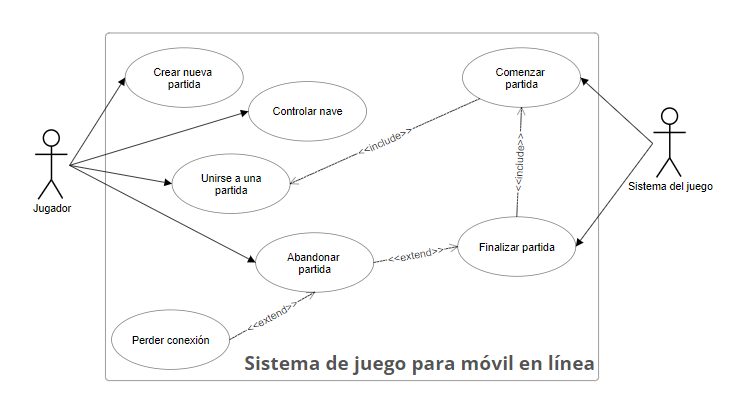
**Solución:**



4. Realizar el diagrama de casos de uso para el siguiente sistema:

* Se trata de un juego de teléfono móvil dónde participan dos jugadores cada uno con su propia terminal.
* Cuando dos jugadores desean jugar, uno de ellos crea una nueva partida y el otro se conecta.
* El objetivo del juego es manejar una nave y disparar, al contrario. Si uno de los dos jugadores acierta, la partida termina.
* Si uno de los dos jugadores deja la partida (o se pierde la conexión) la partida termina.

**Solución:**



# Conclusiones:

Barrera Treviño José Gerardo:

En conclusión, los diagramas desempeñan un papel fundamental en el Lenguaje Unificado de Modelado (UML) para representar distintos aspectos de sistemas y procesos, ya sea en términos de estructura, interacción o comportamiento. Los diagramas de clases se enfocan en la estructura, mientras que los diagramas de interacción describen cómo los objetos interactúan entre sí. Los diagramas de secuencia capturan el flujo de eventos y mensajes en un sistema, y los diagramas de actividades representan procesos y comportamientos. Además, existen diagramas que incorporan roles de clasificador y roles de asociación, brindando una perspectiva tanto estática como dinámica de los objetos en un sistema.

En conjunto, estos diagramas constituyen herramientas poderosas para la comunicación, el diseño y la documentación de sistemas complejos, permitiendo a desarrolladores y profesionales de negocios comprender y representar de manera efectiva la estructura, el comportamiento y las interacciones en el ámbito de la modelización de sistemas. La elección del tipo de diagrama depende de los aspectos específicos que se deseen abordar, ya sea la estructura, el flujo de eventos, la secuencia de acciones o la interacción entre objetos.

Muñoz San Agustín Victoria Monserrat: En esta práctica reforcé los conocimientos de UML, ya que no quedaba muy claro la manera en que actuaban las flecas y como se dividían los métodos y atributos dentro de las clases. Además, logre comprender de mejor manera como se van jerarquizando las clases, desde la superclase, hasta las subclases. Otro punto a tocar son los diferentes señaladores de las relaciones entre clases, desde la composición que señala que *no puede darse la existencia de una clase sin la clase principal,* o la herencia que habla más de las jerarquías ya mencionadas.

Finalmente, considero que realizar el UML en plataformas digitales y no a mano facilito mejor la comprensión de las relaciones y distribuciones de las diversas características que conforman a una clase. Además de que la cuenta institucional me permitió trabajar libremente con lucidchart, que es una plataforma que además de ser completa te permite acceder a soluciones teóricas y prácticas de lo que conlleva un UML.

Olvera Bravo Cynthia Carolina:

En esta práctica realizamos diagramas UML para implementarlos en la programación orientada a objetos. Con ellos pude entender la estructura para el programa que realizaremos y así pude entender la importancia de estos para organizar y poder visualizar la implementación.

En realidad, puede llegar a ser un reto poder ordenar de manera eficiente los diagramas y lograr hacer un buen programa, pero con la práctica podemos aprender a diferenciar entre lo que realmente vale la pena implementar y lo que no nos aporta realmente para la implementación. Con esta práctica logré tener más práctica y así realizar diagramas para tener programas más eficientes y de mayor calidad

Velasco Velasco Santiago:

Durante la práctica, experimentamos de primera mano la importancia de UML en el desarrollo de software basado en programación orientada a objetos. Hemos observado cómo UML proporciona una representación visual efectiva para modelar sistemas de software, lo que facilita la comunicación entre miembros del equipo de desarrollo y partes interesadas, así como el diseño eficiente y la documentación. Además, hemos comprendido cómo UML fomenta la reutilización de componentes y la identificación temprana de problemas de diseño, lo que contribuye a un desarrollo más eficiente y sistemas de mayor calidad. Gracias a la estandarización del diseño de sistemas que ofrece esta herramienta, se facilita el proceso de desarrollo de software y la colaboración en proyectos de programación orientada a objetos, y hemos podido implementar diagramas de clase y diagramas de casos de uso satisfactoriamente, atendiendo a la notación vista en clase, con algunas dudas en el uso de <<extend>> e <<implement>> durante el proceso pero sin ningún problema significativo.

Leocadio Chávez Rodrigo:

El desarrollo de esta práctica no fue complicado, pues lo realizado no conlleva gran dificultad, sin embargo, considero que me quedaron unas pocas dudas respecto a los <include> en los diagramas de casos de uso pues no supe cómo es su funcionamiento. Me parece importante realizar este tipo de prácticas pues reflejan buenos hábitos al programar, ya que facilita el proceso al realizar los programas

Referencias:

s.a. (s.f.). Diagrama de Comunicación UML 2.0. Recuperado el 29 de octubre de 2023 de <http://www.javier8a.com/itc/bd1/comunicacion.pdf>

Lucidchart. (2023). UML. Recuperado el 29 de octubre de 2023 de <https://www.lucidchart.com/pages/es>

Digital Guide. (2023). UML. Recuperado el 29 de octubre de 2023 de <https://www.ionos.mx/digitalguide/>