Actividad 1. Leer y subrayar.

UML (LENGUAJE DE UNIFICADO DE MODELADO)

El Lenguaje Unificado de Modelado (UML) es un lenguaje estándar para escribir planos de software.

¿Qué es, entonces, un **modelo**?

- Un modelo es una simplificación de la realidad.

¿**Por qué** modelamos?

- Para comprender mejor el sistema que estamos desarrollando.

¿**Qué conseguimos** a través del modelado?

1. Visualizar cómo queremos que sea un sistema.
2. Especificar la estructura o el comportamiento de un sistema.
3. Proporciona plantillas que nos guían en la construcción de un sistema.
4. Documentan las decisiones que hemos adoptado.

¿Dónde puede **utilizarse** UML?

- Sistemas de información empresarial.
- Bancos y servicios financieros.
- Telecomunicaciones
- Transportes (Aéreo, Terrestre, Marítimo)
- Defensa/industria aeroespacial
- Comercio ((E-commerce, Retail))
- Electrónica médica
- Ámbito científico
- Servicios distribuidos basados en la web
- Inteligencia Artificial (IA) y Machine Learning
- Blockchain y Criptomonedas
- Cloud Computing y Microservicios
- DevOps y CI/CD
- Realidad Virtual (VR) y Metaverso
- IoT (Internet de las Cosas)
- Ciberseguridad
- Videojuegos (GameDev)

¿Cuáles son los **más usados en la práctica**?

1. **Diagrama de Clases** (para diseño POO y Bases de Datos).
2. **Diagrama de Secuencia** (para flujos de procesos críticos).
3. **Diagrama de Casos de Uso** (para capturar **requerimientos**).
4. **Diagrama de Actividades** (para workflows empresariales).
5. **Diagrama de Estados** (para sistemas reactivos como IoT).

El modelado es importante, por una simple razón:

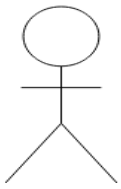
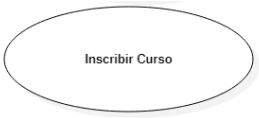
“Construimos modelos de sistemas porque no podemos comprender el sistema en su totalidad”.

DIAGRAMAS DE CASOS DE USO

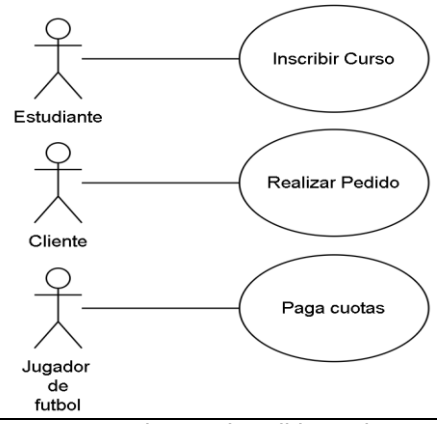
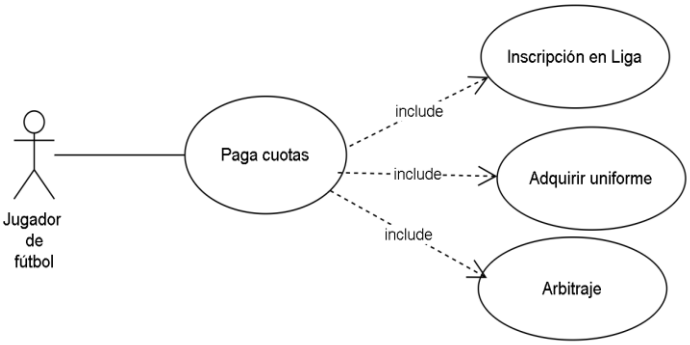
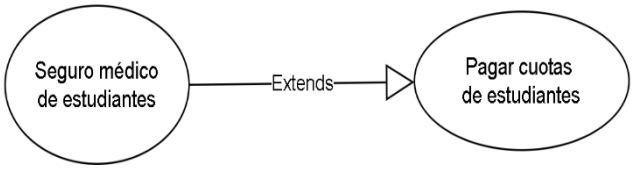
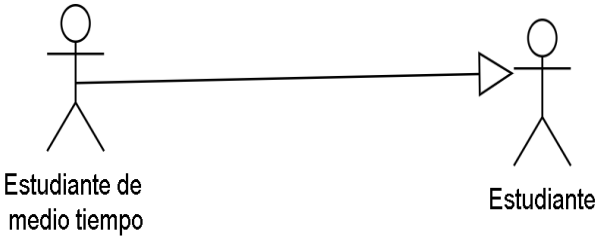
Muestra un conjunto de casos de uso y actores y sus relaciones, son usados durante el **análisis de un proyecto** para identificar la funcionalidad del sistema. Los casos de uso describen **qué hace** un sistema, pero **no cómo lo hace**.

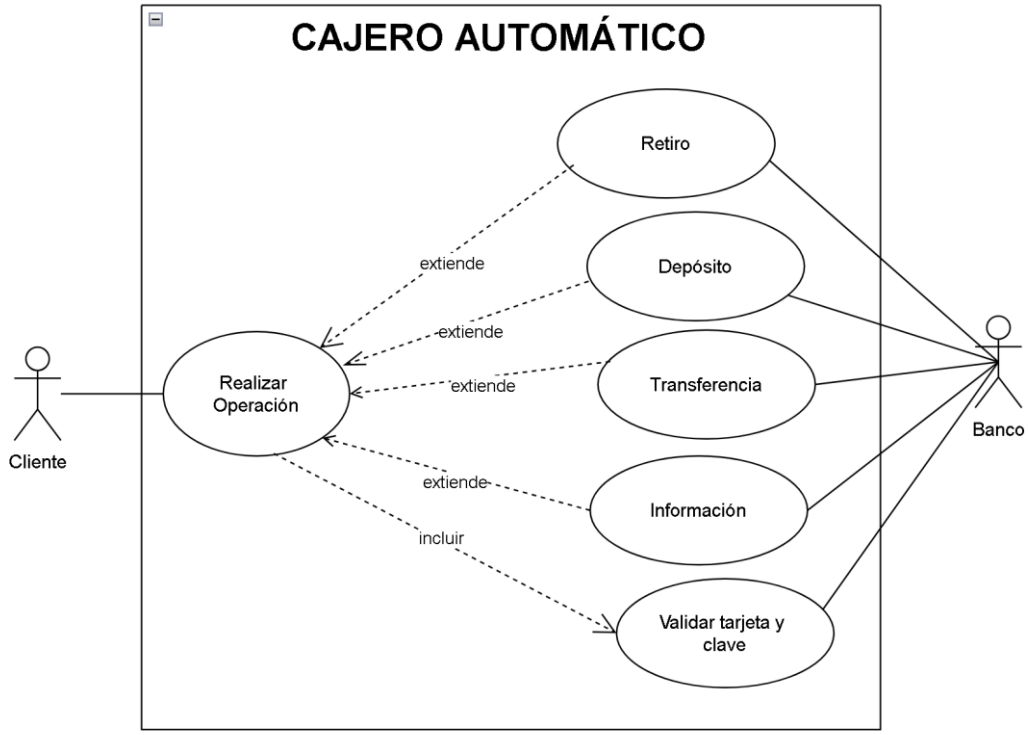
Modelar casos de uso:

- Se debe identificar los diferentes tipos de personas<actores> que usan el sistema de forma externa. Dibujar una línea alrededor de todo el sistema y asegurar qué actores quedan fuera del sistema e interactúan con él.
- Especificar qué debería hacer el sistema (desde el punto de vista externo), independientemente de cómo se haga.

Símbolos de los Diagramas de Caso de Uso	
Símbolo	**Significado**
Actor Se representa con un muñeco de palo  	Los actores son entidades externas; **existen fuera del sistema** e interactúa con éste de una manera específica y directa. • Actor se refiere al rol específico de un *usuario del sistema*. • Por ejemplo, alumno, empleado o cliente. • Los actores se pueden dividir en grupos. ○ **Actores principales** suministran datos o reciben información del sistema. ○ **Actores de soporte o secundarios**, ayudan a mantener el sistema en funcionamiento o a proveer otros servicios; soporte técnico, analistas, programadores, etcétera.
Casos de uso Se representa con una elipse  	• Un caso de uso es una secuencia de transacciones en un sistema. • El **nombre de un caso de uso** consta de un *verbo* y un *sustantivo*.
Relaciones Comunica ——————— Incluye - ▶ Extiende ◀ - Generaliza ——————— ▶	**COMUNICACIÓN:** ♣ conecta un actor con un caso de uso, mediante una línea sin puntas de flecha, ♣ conexión básica entre un **actor** y un **caso de uso** **INCLUSIÓN («include»)** ♣ Indica que un caso de uso **DEBE ejecutar** otro caso de uso. ♣ Usar cuándo: repetir funcionalidades comunes. ♣ **Ejem:** Realizar Pedido «include» Validar Pago . *(El caso "Realizar Pedido" **siempre** incluye "Validar Pago").* **EXTENSIÓN («extend»)** ♣ Un caso de uso **opcional** que se ejecuta bajo ciertas condiciones. ♣ Usar cuando: funcionalidades condicionales o excepciones. ♣ La flecha apunta del caso de uso extendido al básico. ♣ **Ejem:** Realizar Pedido «extend» Aplicar Descuento. **GENERALIZACIÓN (▷):** ♣ Esta relación puede existir entre dos actores o dos casos de uso. La flecha apunta a la "cosa" general. ♣ Relación de **herencia** entre actores o casos de uso.

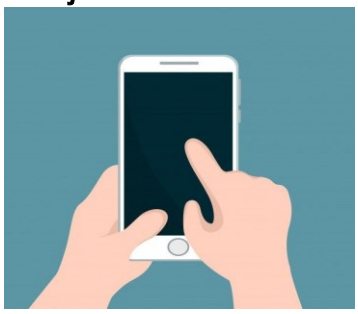
Ejemplos

RELACIÓN DE COMUNICACIÓN	RELACIÓN DE INCLUSIÓN
	
Un Estudiante se comunica con Inscribir en el curso. *Un "Cliente" interactúa con el caso de uso "Realizar Pedido"* Un jugador de futból paga cuotas.	Pagar cuotas de JUGADOR que se incluye en Inscribir en Liga, adquirir uniforme y arbitraje, ya que en estos casos los jugadores deben pagar sus cuotas.(**INLUYE SIEMPRE COMÚN**)
RELACIÓN DE EXTENSIÓN	**RELACIÓN DE GENERALIZACIÓN**
	
Seguro médico de estudiantes extiende el caso de uso básico Pagar cuotas de estudiantes. La flecha va del caso de uso extendido al caso de uso básico.(**EXTIENDE EXCEPCIONES**).	Un Estudiante de medio tiempo generaliza a un Estudiante. La flecha apunta a la cosa general.



DIAGRAMAS DE CLASES

- Muestra un conjunto de **clases**, **interfaces** y **colaboraciones**, así como sus **relaciones**.

Termonología básica de Diagramas de Clases	
Objetos	**Ejemplos**
Los objetos pueden ser algo físico o algo conceptual. Los nombres de los objetos comúnmente suelen ser sustantivos, un cliente, una persona, un producto. <<*instanciar*, crear un objeto a partir de una Clase>>	○ **Físico** como algo que podemos tocar, algo que sí es tangible. ***Por ejemplo***, una persona: Cliente, producto. ○ **Conceptual** es algo que no es tangible, algo que solamente puede existir en nuestra mente o conceptualmente o áreas en la pantalla. **Por ejemplo,** cuenta de un cliente, código de un producto.
Los objetos poseen sus propios: ○ **Atributos**, o sea sus características. ○ **Métodos** o sea funcionalidades.	○ Los **atributos** también son sustantivos como peso, precio, color, etcétera. ○ Los **métodos,** comportamientos o las funcionalidades de los objetos son verbos, por ejemplo, enviar pedido, mostrar, imprimir, comprar, etcétera.
EJEMPLO	
El **objeto telefono** tiene: 	**atributos** • id • marca • modelo **métodos** • llamar • colgar

Para UML una **Clase** es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, métodos, relaciones, y semántica. En POO(Programación Orientada a Objetos), la clase se representa una **plantilla,** que nos permite generar más objetos:

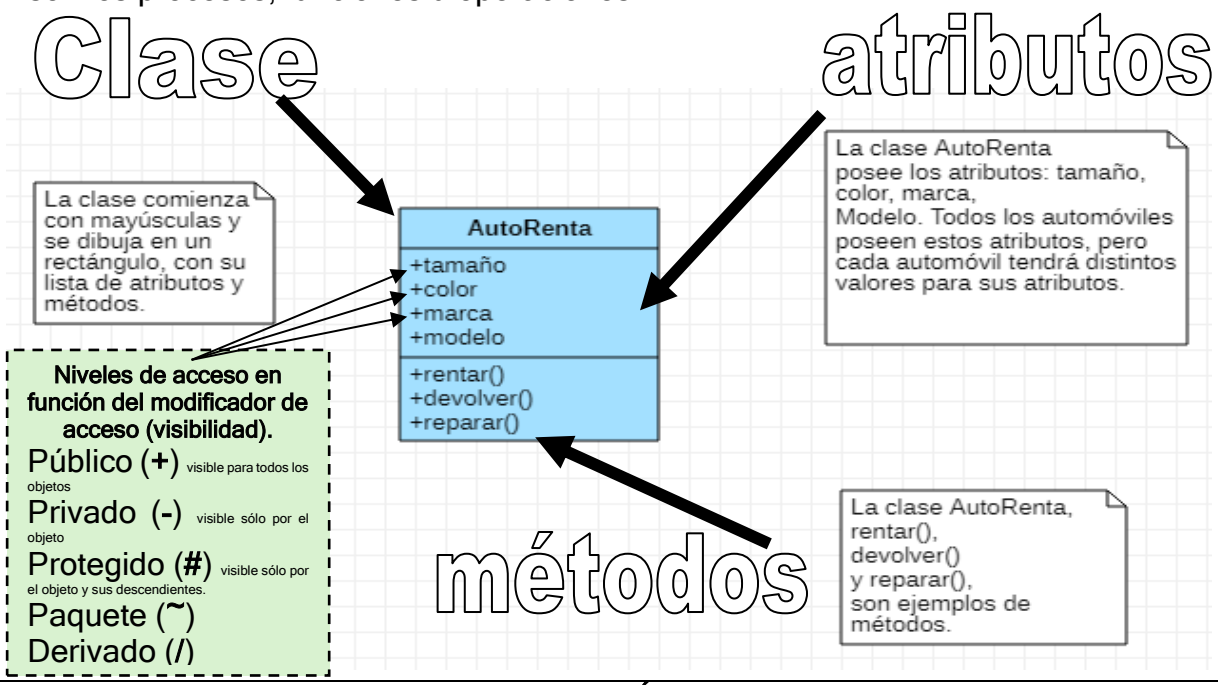
- Una **Clase** es la forma en cómo defines un objeto para generar más objetos.
- Una **Clase** define el conjunto de atributos compartidos y comportamientos que se encuentran en cada objeto de la clase.

Ejemplos: Los nombres de las clases son **sustantivos** y empiezan con **MAYÚSCULAS**.

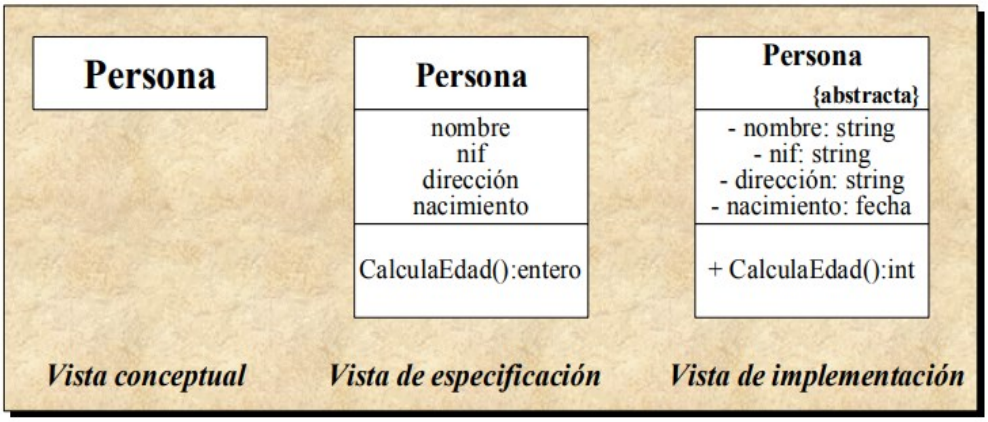
- Alumnos
- Producto
- Vehículo
- Departamento
- Curso
- Cliente

Representación de la Clase en *UML*

- **Clase** se dibuja como un **rectángulo**, contiene: lista de atributos y una lista de métodos.
- **Atributo** describe cierta **propiedad** que poseen todos los objetos de la clase.
- **Método** es una **acción** que se puede solicitar de cualquier objeto de la clase. Los métodos son los procesos, funciones u operaciones.



- Al especificar las **Clases**, la primera letra va con ***MAYÚSCULA y son nombres cortos***.
- Al especificar los **atributos**, la primera letra va ***con minúscula***.
- Al especificar los **métodos**, la primera letra va ***con minúscula***.



La posibilidad de especificar con un mayor o menor grado de detalle los atributos y las operaciones de una clase, permite utilizar un diagrama de clase de UML desde una perspectiva conceptual, de especificación o de implementación, como se puede apreciar en la Figura:

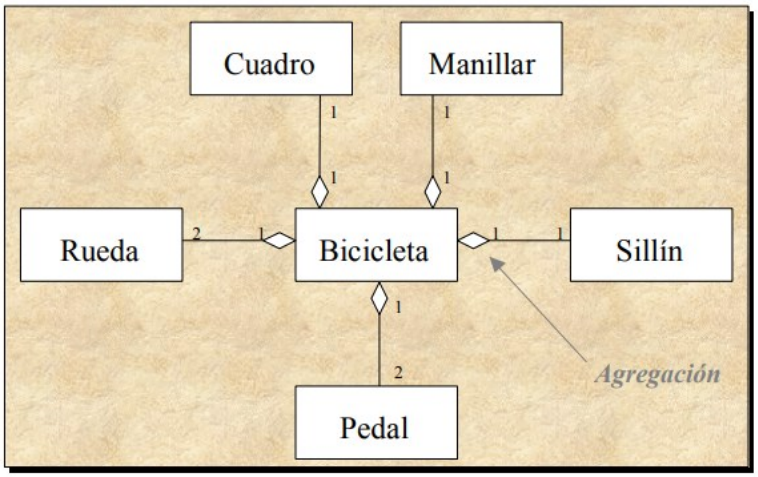
Asociaciones

- Las **asociaciones** representan las **relaciones entre clases**.
- **La multiplicidad o cardinalidad**: indica el grado y nivel de dependencia de las clases, se anotan en cada extremo de a relación.
- Las asociaciones son por defecto **bidireccionales**, debe indicarse de forma explícita el sentido de la asociación mediante una punta de flecha al final de la línea de asociación.
  - **Ejemplo**: Muchos Pedidos estan asociados a un Cliente.

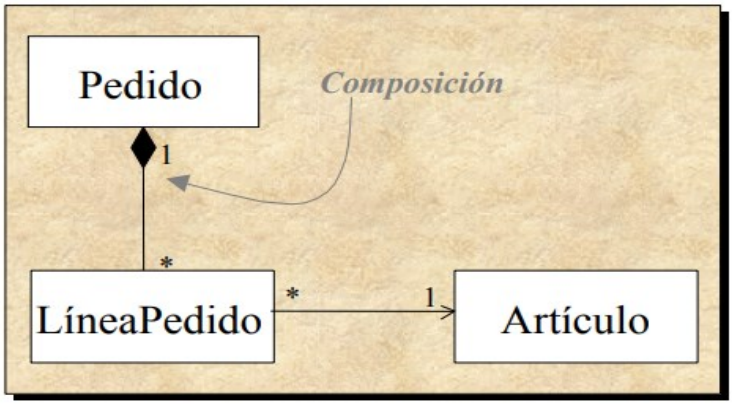
*	= Cero, uno ó n.
0,1	= Cero a uno.
1..*	= Uno o más
1	= Exactamente uno.
1..5	= Entre uno y cinco

Agregación y composición

- La **agregación** (diamante blanco) es la relación parte-de, que presenta a una entidad como un agregado de partes.

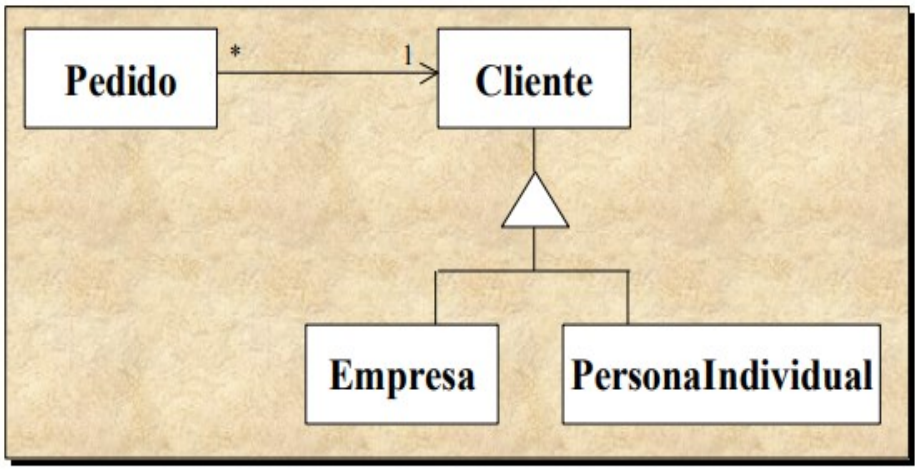


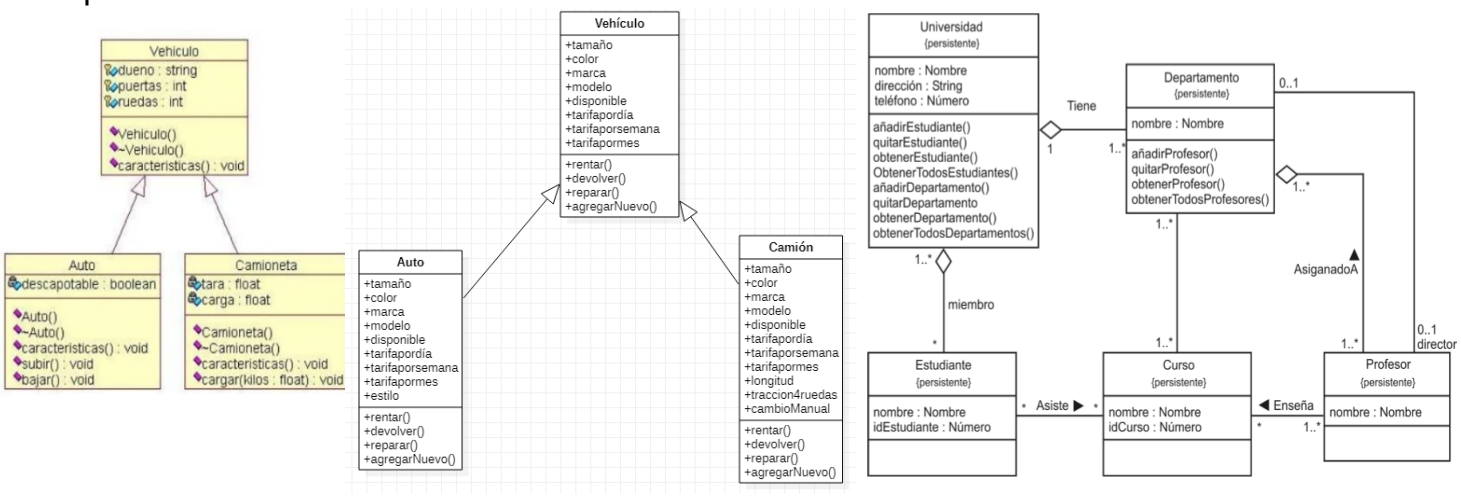
Ejemplo: La bicicleta, donde una bicicleta se modela como un agregado de ruedas, sillín, manillar, cuadro y pedales.



- La **composición** (diamante negro) implica que los componentes de un objeto sólo pueden pertenecer a un solo objeto agregado, de forma que cuando el objeto agregado es destruido todas sus partes son destruidas también.

Herencia (tríangulo vacio)
Las clases pueden tener hijos; es decir, se puede crear una clase a partir de otra. En UML, la clase original (o padre) se conoce como clase base; a la clase hija se le denomina clase derivada. Podemos crear una clase derivada de tal forma que herede todos los atributos y comportamientos de la clase base. Sin embargo una clase derivada puede tener atributos y comportamientos adicionales.

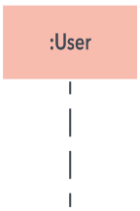
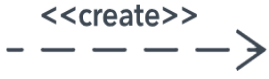




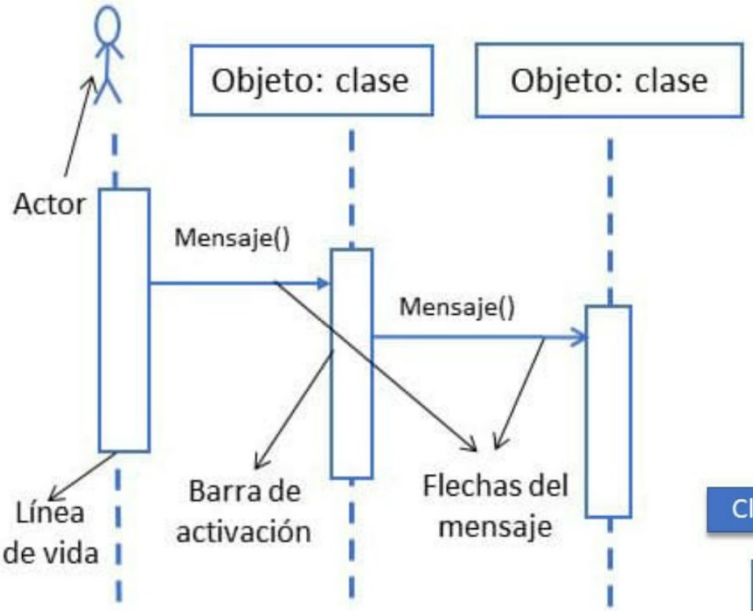
DIAGRAMAS DE SECUENCIA

Un diagrama de secuencia es un tipo de diagrama de interacción porque describe cómo —y en qué orden— un grupo de objetos funcionan en conjunto.

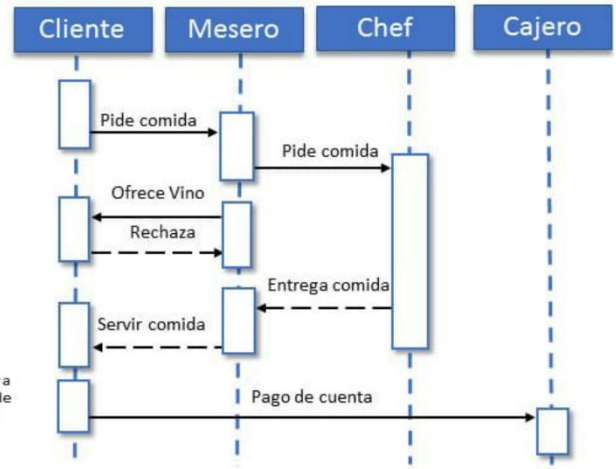
Componentes y símbolos básicos

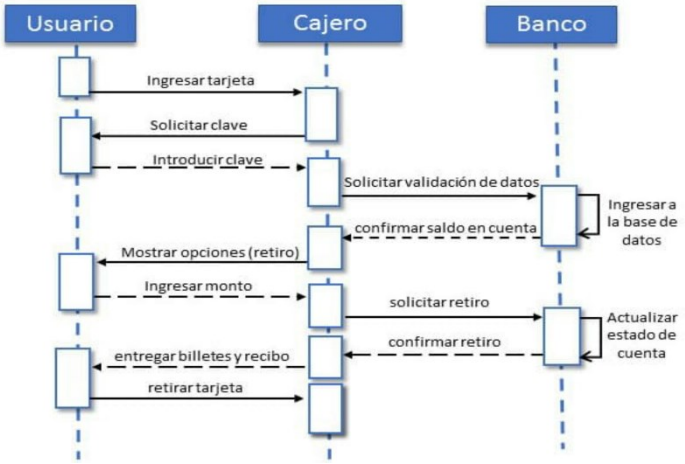
Símbolo	Nombre	Descripción
	Símbolo de objeto	Representa una clase u objeto en UML.
	Casilla de activación	Representa el tiempo necesario para que un objeto finalice una tarea. Cuanto más tiempo lleve la tarea, más larga será la casilla de activación.
	Símbolo de actor	Muestra entidades que interactúan con el sistema o que son externas al sistema.
	Símbolo de línea de vida	Representa el paso del tiempo a medida que se extiende hacia abajo. Esta línea vertical discontinua representa eventos secuenciales que le ocurren a un objeto durante el proceso graficado. Las líneas de vida pueden comenzar con una figura rectangular etiquetada o un símbolo de actor.
	Símbolo de mensaje sincrónico	Representados por una línea continua y una punta de flecha sólida. Este símbolo se utiliza cuando un remitente debe esperar una respuesta a un mensaje antes de proseguir. El diagrama debe mostrar el mensaje y la respuesta.
	Símbolo de mensaje asincrónico	Representados por una línea continua y una punta de flecha simple. Los mensajes asincrónicos no necesitan una respuesta para que el remitente siga adelante. Solo la llamada se debe incluir en el diagrama.
	Símbolo de mensaje de respuesta asincrónico	Representados por una línea discontinua y una punta de flecha simple.
	Símbolo de crear mensaje asincrónico	Representados por una línea discontinua y una punta de flecha simple. Este mensaje crea un nuevo objeto.
	Símbolo de mensaje de respuesta	Están representados con una línea discontinua y una punta de flecha simple. Estos mensajes son las respuestas a las llamadas.
	Símbolo de eliminar mensaje	Están representados por una línea continua y una punta de flecha sólida, seguida de una X. Este mensaje destruye un objeto.

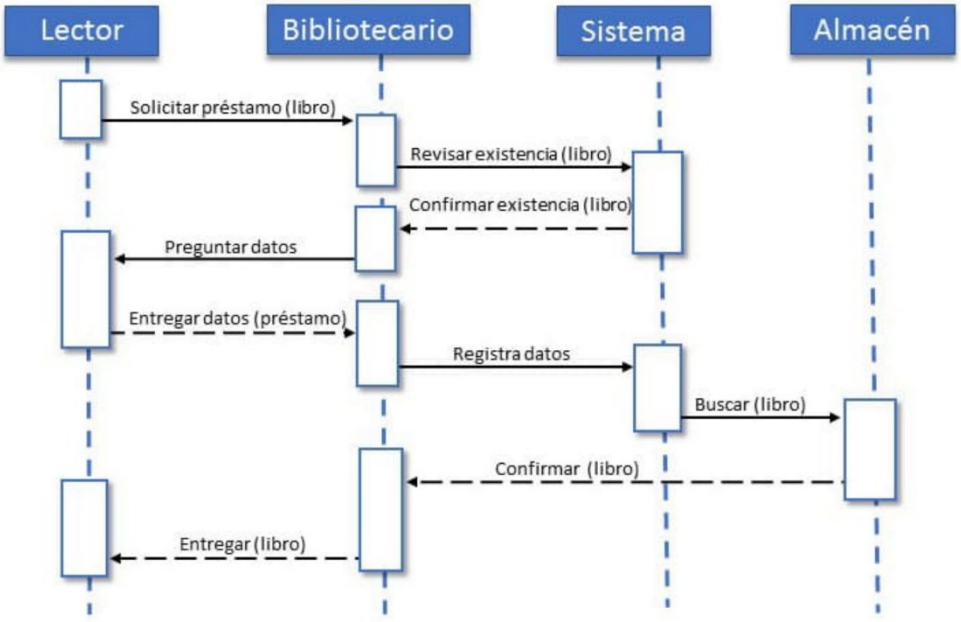
Elementos y simbologías en una estructura de diagrama de secuencia



Mensaje	Flecha
Sincrónico	→
Asincrónico	→
Respuesta	- - - →







Actividad 2. Resolver el siguiente cuestionario.

1. ¿Por qué UML es importante en el desarrollo de software?

_____
_____
_____

2. ¿Cuáles son los tipos de diagramas más comunes en UML y qué representan?

_____
_____
_____
_____
_____

3. Realizar una infografía por cada diagrama UML:

a. Diagrama de casos de uso
b. Diagrama de clases
c. Diagrama de secuencia

4. Realiza los siguientes diagramas UML, relacionado con el uso de Instagram, desde la perspectiva de un usuario:

- Diagrama de Casos de Uso (interacción usuario-sistema).
  - Un **usuario**, puede publicar una foto/video, dar «me gusta» a una publicación, comentar una publicación, seguir a otro usuario, explorar el feed de noticias, buscar perfiles/hashtags y recibir notificaciones.

- Diagrama de Clases (estructura de datos).
  - **Clases clave:**
    - **Usuario** (atributos: username, email, seguidores[]).
    - **Publicación** (atributos: foto, fecha, likes).
    - **Comentario** (atributos: texto, fecha).
    - **Notificación** (atributos: tipo, mensaje).
  - **Relaciones:**
    - Un **Usuario** puede tener muchas **Publicaciones** (1 a *n*).
    - Una **Publicación** puede tener muchos **Comentarios** (1 a *n*).
    - Un **Usuario** puede recibir muchas **Notificaciones** (1 a *n*).

- Diagrama de Secuencia (flujo de mensajes en un escenario): Dar like a una publicación.
  - **Actores/Objetos:**
    - **Usuario**
    - **App Instagram**
    - **Base de Datos**
  - **Pasos:**
    - Usuario hace clic en "like".
    - La app envía la solicitud a la base de datos.
    - La base de datos actualiza el contador de likes.
    - La app notifica al usuario dueño de la publicación.