

# Diagramas De Interacción

# Diagramas UML

El Lenguaje Unificado de Modelos (UML) nos ofrece un conjunto estandarizado de herramientas para documentar el análisis y diseño de un sistema de software.

En el proceso de diseño de software se realizan una serie de diagramas que detallan el sistema desde diferentes perspectivas (vistas parciales).

Podemos encontrar:

- Diagramas estructurales
- Diagramas de comportamiento
- Diagramas de interacción

## Diagramas estructurales

Diagrama de Clases

Diagrama de Componentes

Diagrama de Despliegue

## Diagramas de comportamiento

Diagramas de Casos de Uso

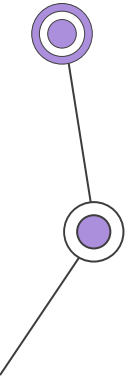
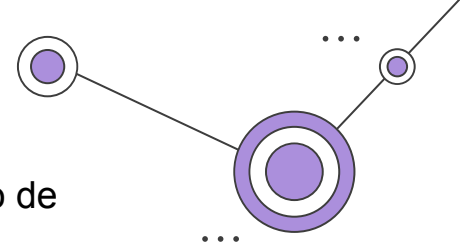
Diagramas de estado

Diagramas de actividad

## Diagramas de Interacción


Diagramas de Secuencia

Diagramas de comunicación



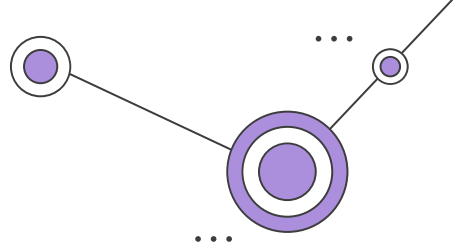


# Diagramas de Interacción

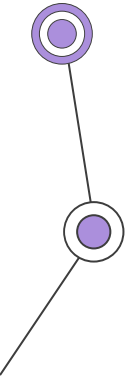
- Muestran cómo los **objetos** de un sistema **interactúan entre sí**.
  - Se enfocan en:
    - **Qué mensajes se envían.**
    - **Entre qué objetos.**
    - **En qué orden.**
  - Son útiles para entender el comportamiento dinámico del sistema.
  - Tipos principales:
    - **Diagrama de Secuencia** (énfasis en el tiempo).
    - **Diagrama de Comunicación/Colaboración** (énfasis en relaciones).
- 

# Diagrama de secuencia

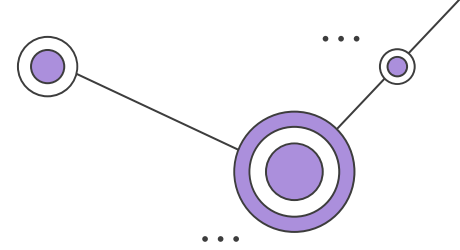
# Diagramas de secuencias



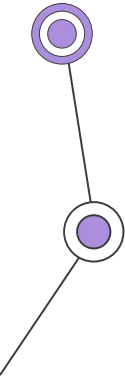
- Un diagrama de secuencias nos muestra cómo interactúan un conjunto de objetos en una aplicación a través del **tiempo** y se modela para cada caso de uso.
- Se lee de arriba hacia abajo (flujo temporal).
- Muestra:
  - Objetos participantes.
  - Mensajes enviados.
  - Orden de ejecución.
- Ideal para describir casos de uso paso a paso.



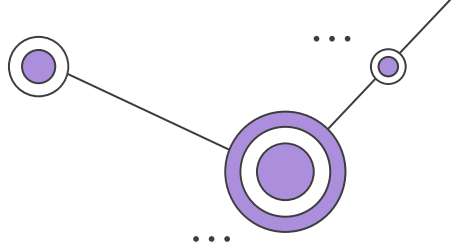
# ¿Debe ir en la documentación entregada al cliente?



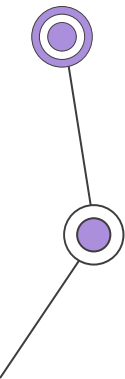
- Depende del tipo de proyecto y contrato.
- En proyectos formales (banca, gobierno, salud) donde se exige documentación UML completa, sí suele incluirse:
  - Casos de uso.
  - Diagramas de clases.
  - Diagramas de secuencia de los flujos principales.
- En proyectos ágiles con Scrum o startups, generalmente NO se entrega al cliente:
  - El cliente quiere funcionalidades, no diagramas.
  - Si se usa, es para el equipo interno, no como entregable.



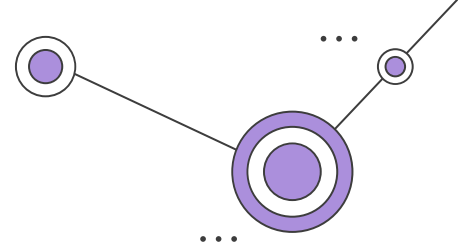
# Entonces, ¿Cuándo lo aplico?



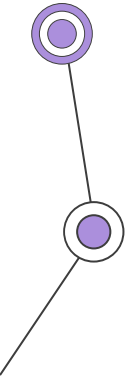
- Úsalo si:
  - El proceso involucra varias capas (UI → lógica → base de datos → servicio externo).
  - Hay confusión sobre el orden de mensajes.
  - Es una funcionalidad crítica que necesita documentación clara.
  - El equipo es grande y necesitas alinear a todos antes de programar.
- No lo uses si:
  - El flujo es directo (ej: Usuario hace login y sistema valida → fin).
  - Solo hay uno o dos mensajes simples.
  - Nadie va a leer el diagrama ni lo necesita para implementar.



# ¿En qué momento exacto del proyecto entra?



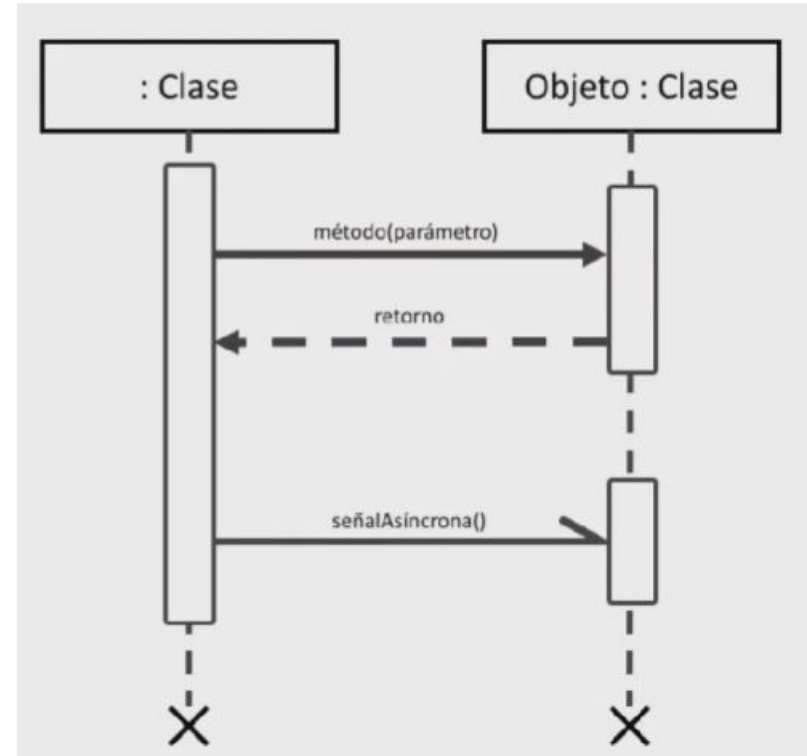
- **Durante el análisis:**
  - Para explorar cómo se resolverá un caso de uso complejo.
- **En el diseño técnico:**
  - Para definir la interacción entre módulos y clases antes de escribir código.
- **En documentación:**
  - Para dejar registro de procesos críticos que otro equipo mantendrá en el futuro.
- **En debugging:**
  - Para entender un flujo que no funciona y detectar en qué mensaje falla.



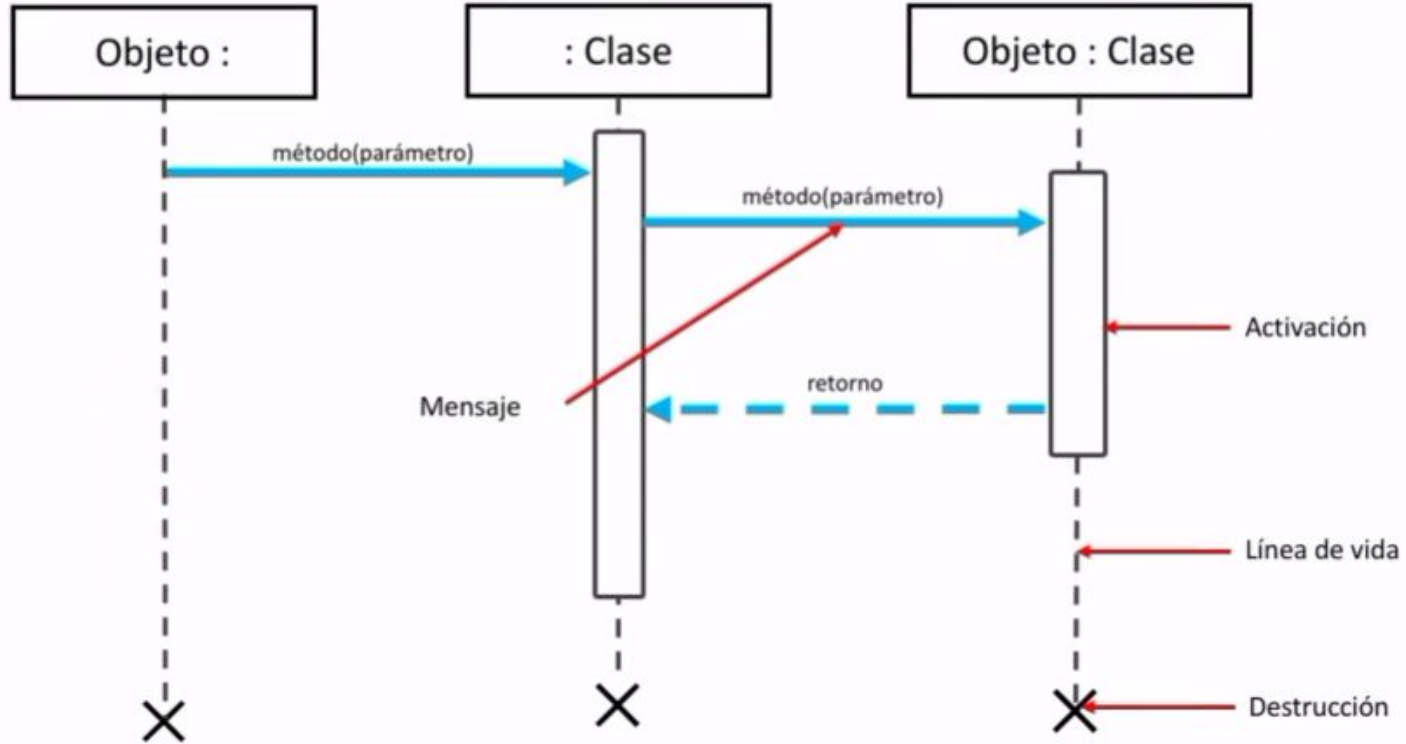


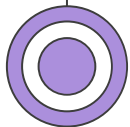
# Diagramas de secuencias

- Un diagrama de secuencia contiene detalles adicionales sobre la implementación, incluyendo los objetos y clases que se utilizan para implementar el escenario y los mensajes intercambiados entre los objetos.
- Por lo tanto, los escenarios de **caso de uso** son útiles para dibujar **diagramas de secuencia**.
- Esto resulta una oportunidad para revisar los casos de uso, replantearlos y modificarlos si es necesario.



# Ejemplo





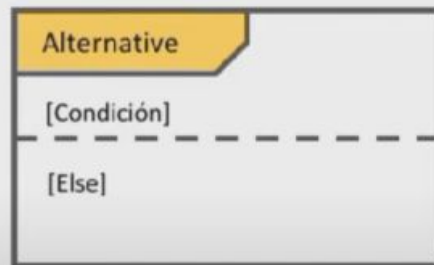
# Elementos



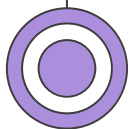
Actor  
(Entidad externa)







Bucle de opción



Alternativas (if - else)



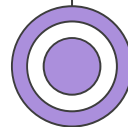
# Mensajes

Mensajes	
Simple	
Síncrono	
Asíncrono	
Devolución o respuesta	

- Las llamadas síncronas son las más comunes. Se utilizan cuando la clase emisora **espera** una respuesta de la clase receptora.
- Las llamadas asíncronas son las que envían el mensaje sin esperar que la clase emisora devuelva una respuesta.
- El grado de detalle del mensaje puede variar:
  - nombreMensaje()
  - nombreMensaje(parametro1, parametro2, ...)
  - nombreMensaje(tipo: nombreParametro)
  - etc

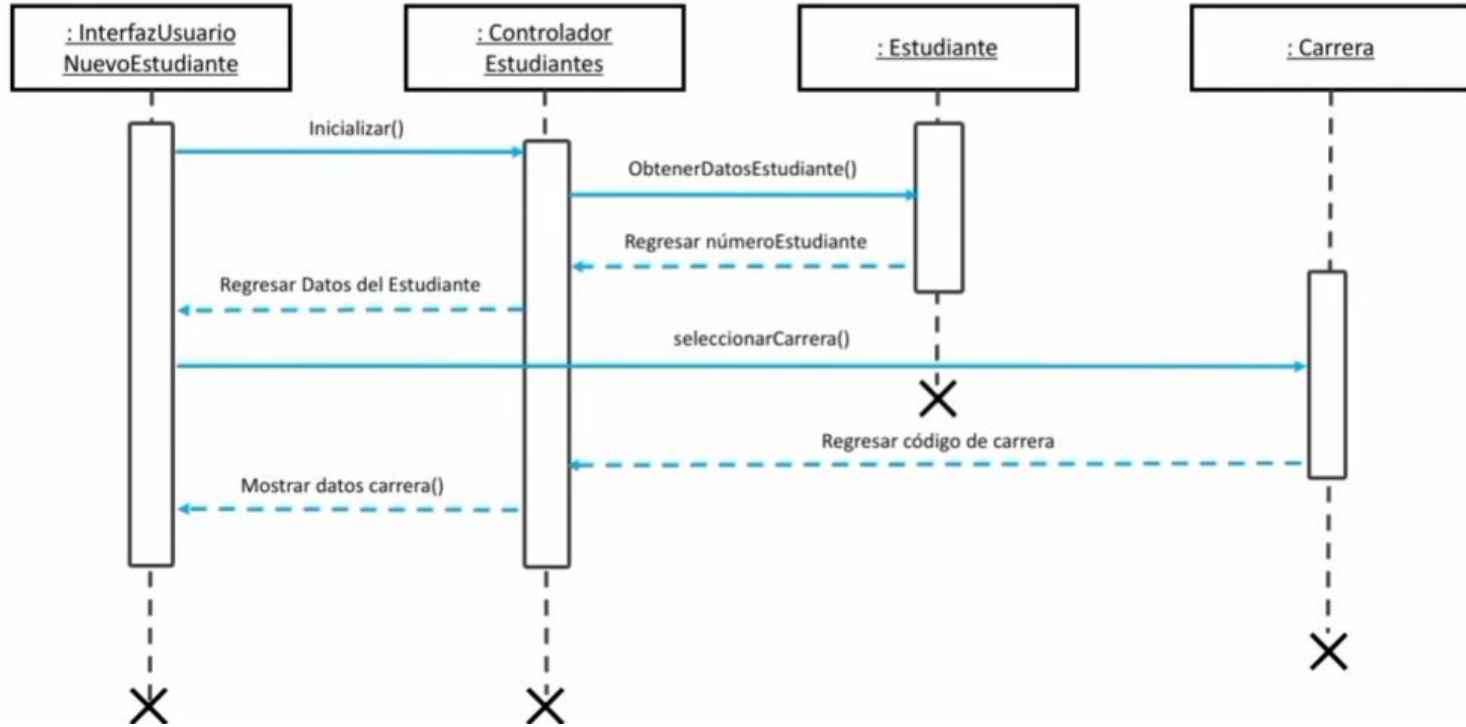


...

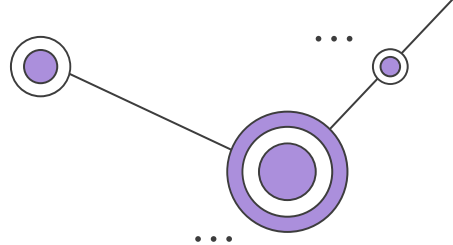


...

# Ejemplo: inscripción de estudiante



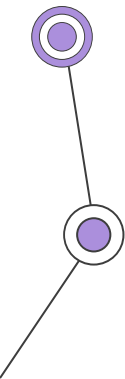
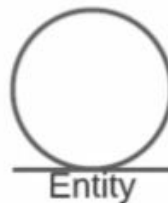
# Reglas de la última versión

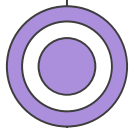


Las reglas para crear diagramas de secuencia son que todas las clases de interfaz (Vista) deben estar conectadas a una clase de control. (Controlador).

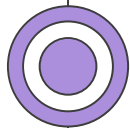
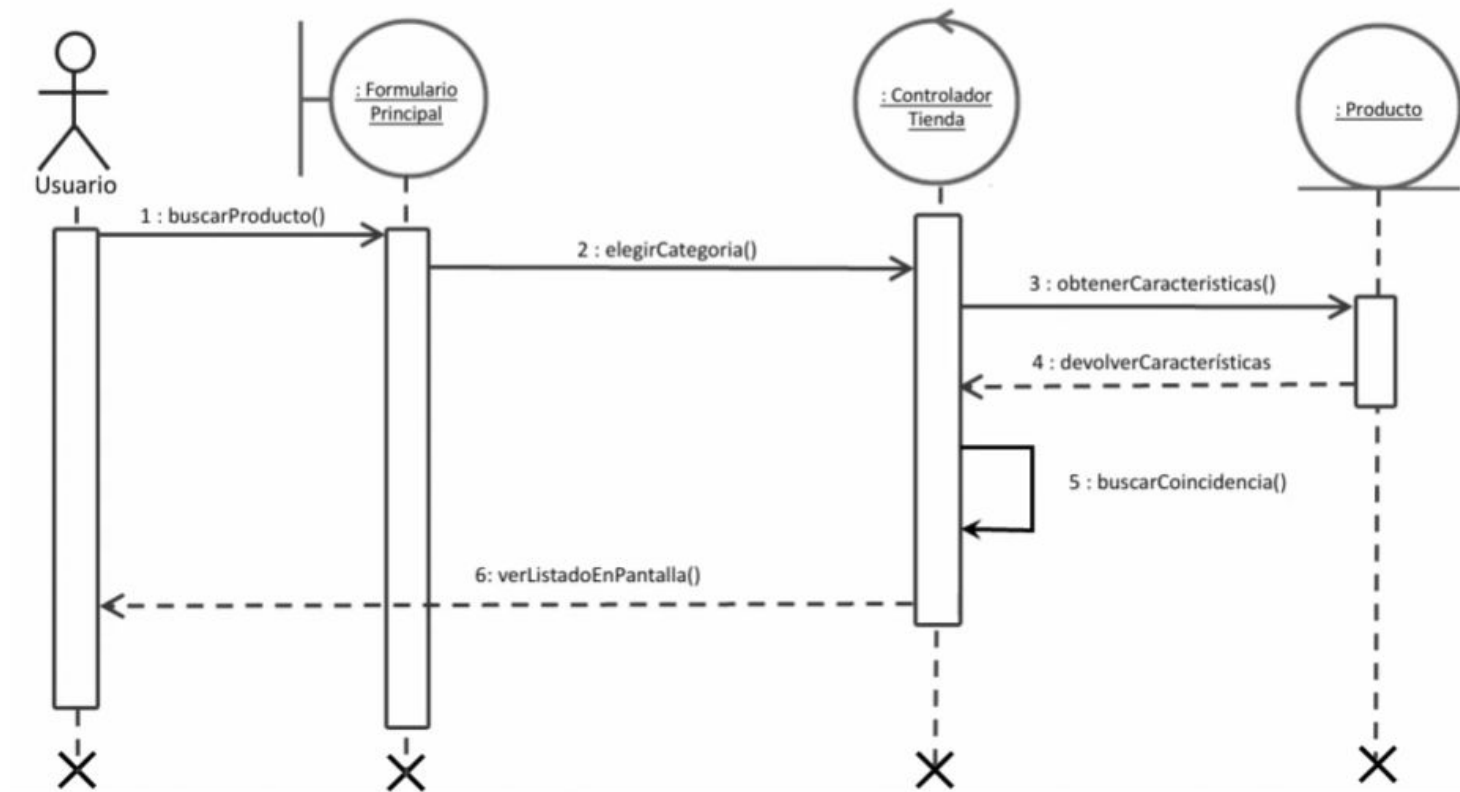
De manera similar, todas las clases de entidad (modelo) deben estar conectadas a una clase de control (Controlador).

Las clases de interfaz (Vista), nunca se conectan de manera directa a las clases de entidad (modelo).

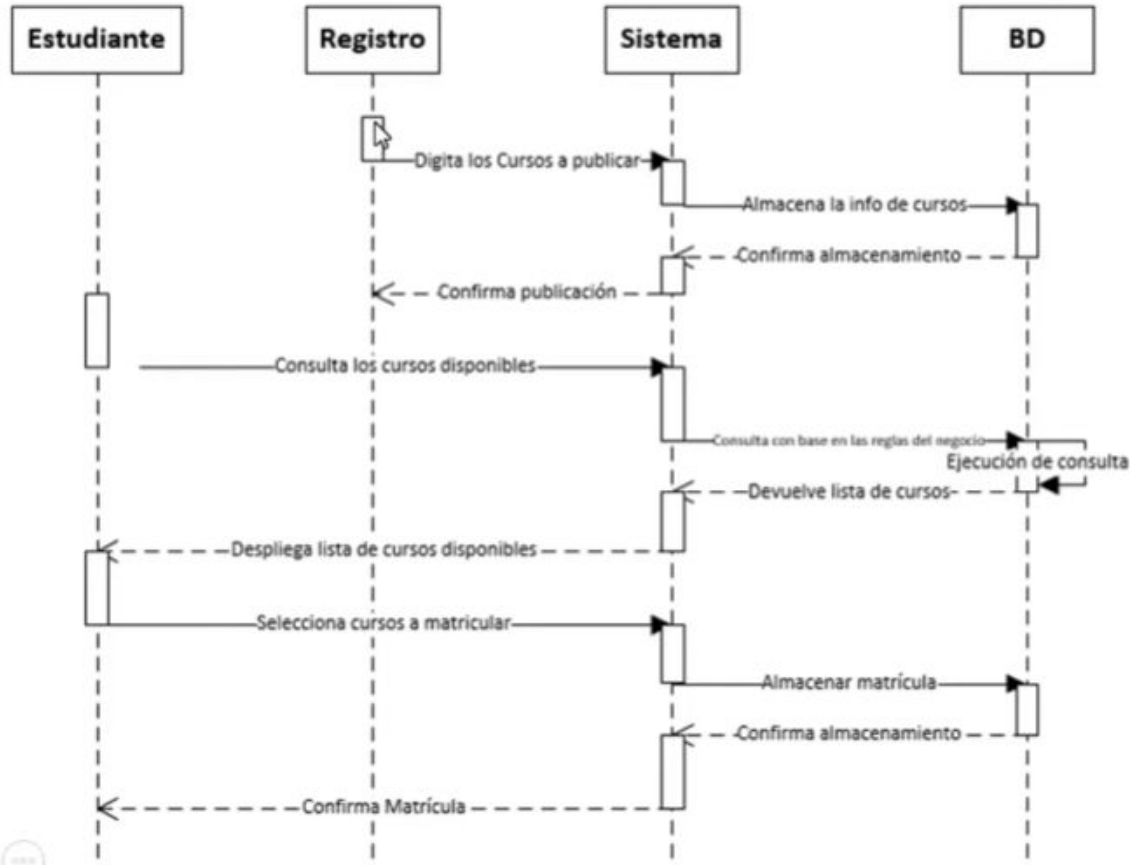




# Ejemplo: filtrar producto

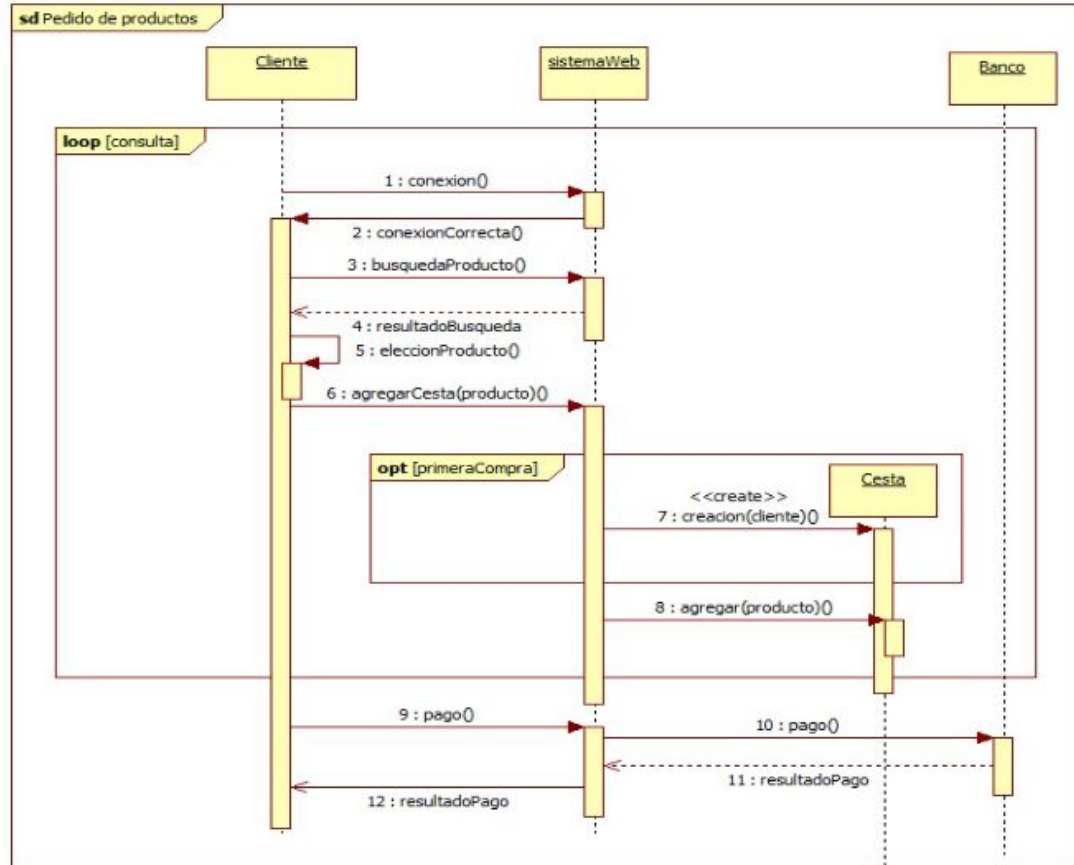


# Ejemplo: matricularse a un curso

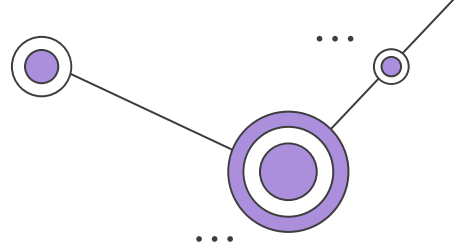




# Ejemplo loop + cond

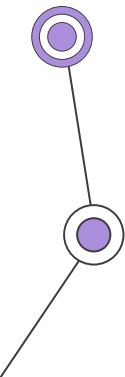


# Pasos para crear un diagrama de secuencia



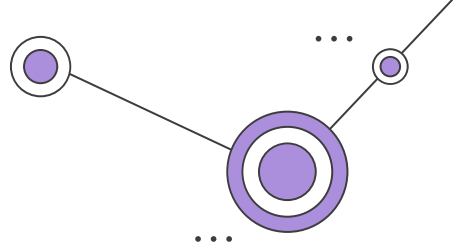
1. Identificar los **actores** que participan en el caso de uso.
2. Examinar qué **clases** de entidad hay en el caso de uso, para incluirlas en el diagrama.
3. Asegurarse de que haya al menos 1 clase de **control**, aunque se pueden crear más durante el diseño.
4. Crear prototipos de páginas web para todas las interfaces humanas.
5. Un error muy común es colocar demasiados detalles técnicos. Esto desordena el diagrama y dificulta su lectura.

Los diagramas de comunicación son una alternativa al diagrama de secuencia, el cuál contiene la misma información pero enfatiza la comunicación y no la sincronización.

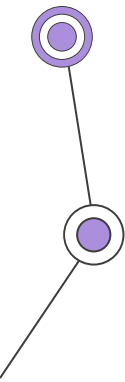


# Diagrama de colaboración

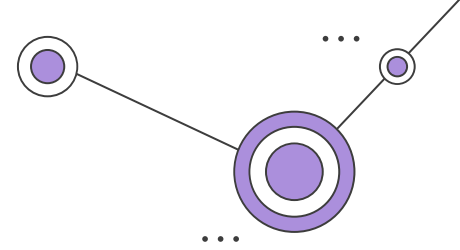
# Diagrama de Colaboración



- Es una forma de representar **interacción** entre objetos.
- También se lo conoce como diagrama de comunicación.
- Consiste en mostrar cómo las instancias específicas de las clases trabajan juntas para lograr un objetivo en común.
- Especifican un contrato entre objetos.
- Representan la parte esencial para la descripción de un patrón de diseño.



# Elementos de un Diagrama de Colaboración



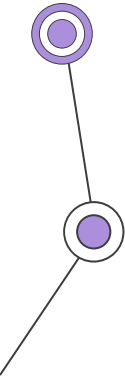
- **Instancias:**

- Las instancias representan un objeto o instancia cualquiera de una clase determinada.
- Una instancia u objeto se representa mediante un rectángulo que contiene el nombre y la clase del objeto en un formato *nombreObjeto:nombreClase*.

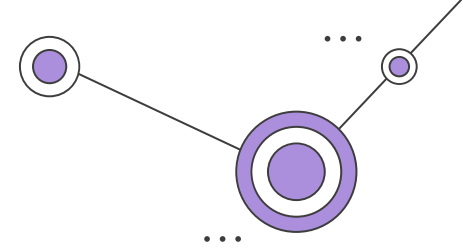


- **Enlace:**

- Los enlaces representan una conexión entre instancias que indican navegabilidad y visibilidad entre ellas.
- Se puede establecer una relación cliente-servidor entre las instancias.

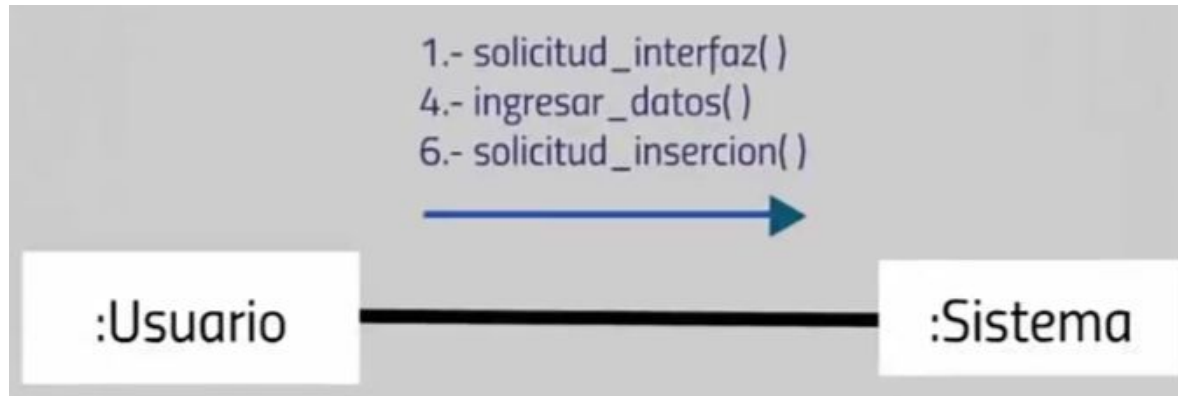


# Elementos de un Diagrama de Colaboración

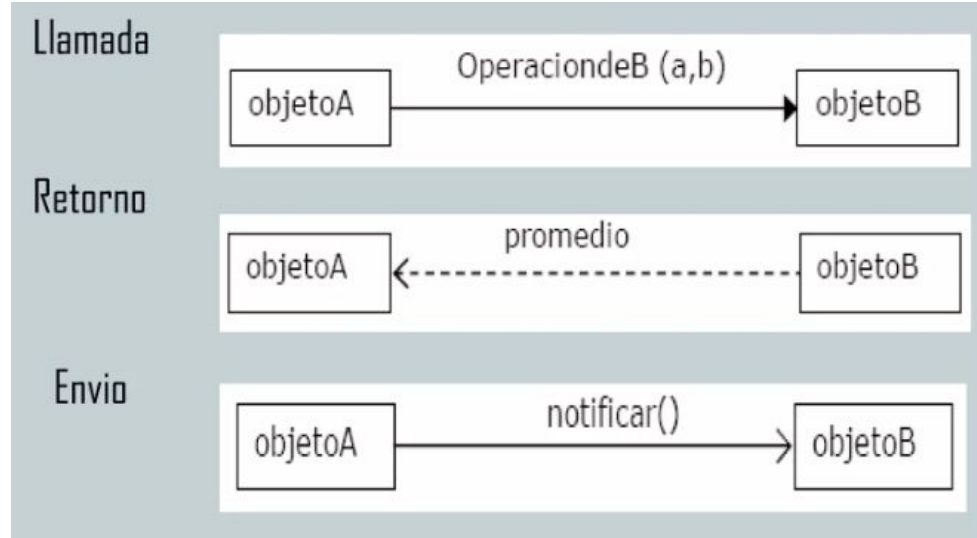


- **Mensajes:**

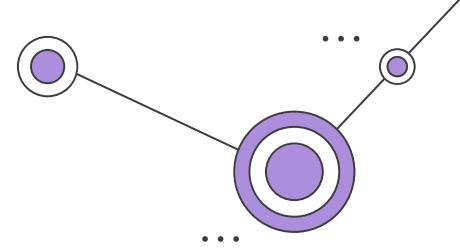
- Se representan mediante una flecha etiquetada.
- Un mensaje está asociado a un enlace y tienen asociados un número que indican el orden de ocurrencia, para así conocer de qué manera se realizarán.



# Interacciones de un Diagrama de Colaboración



# Condicionales

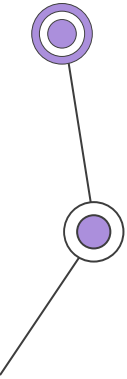


- **Bifurcación:**

- Los caminos alternativos tendrán el mismo número de secuencia, seguido del número de subsecuencia, y se debe distinguir por una condición.

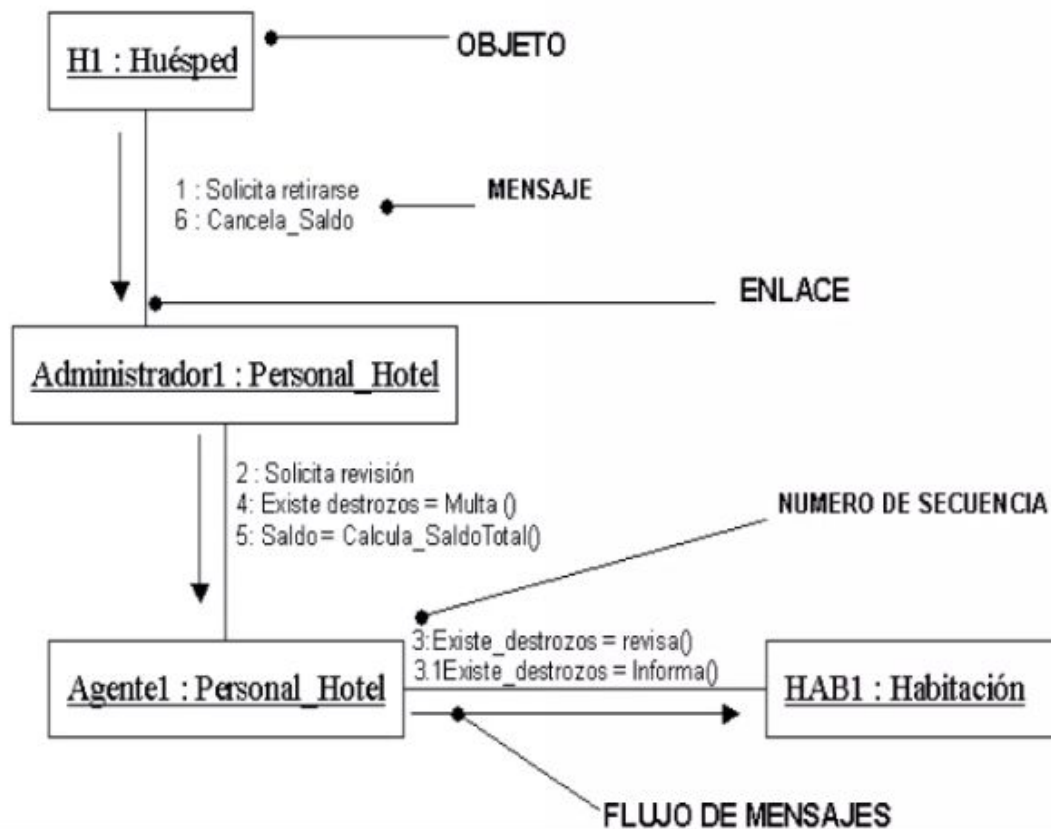
- **Condicionales:**

- Las cláusulas de condición se representan mediante [ ] (corchetes). Esto deja claro que hay dos caminos posibles.
- Se acostumbra a realizar un diagrama para el escenario de éxito y un diagrama para el escenario alternativo, ya que el uso de muchos condicionales hace que se ensucie el diagrama y sea complejo de entender.

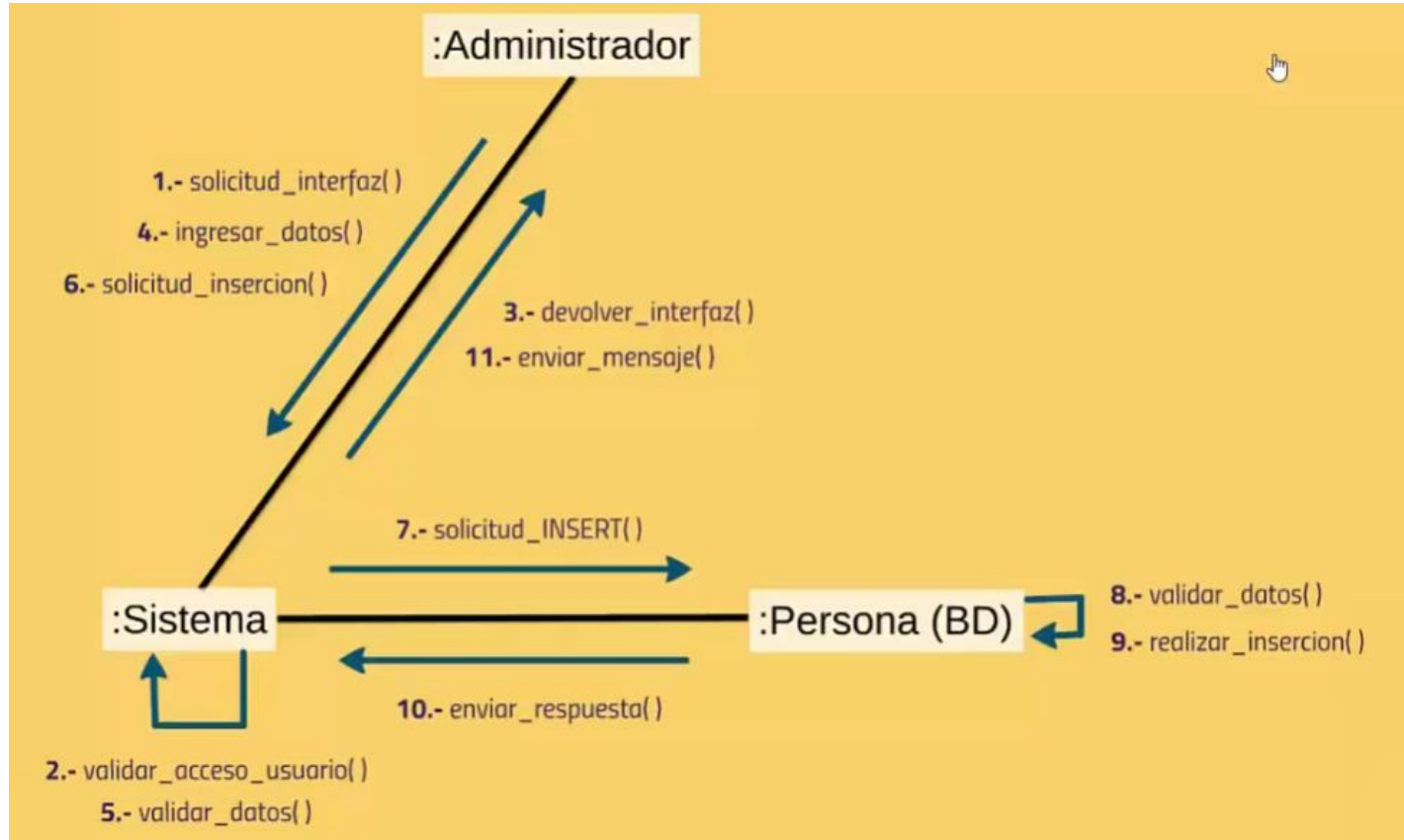




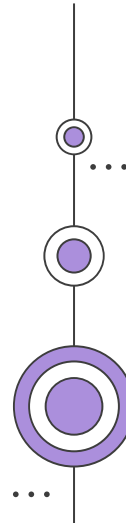
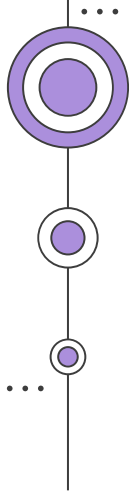
# Ejemplo completo



# Ejemplo: CU agregar registro de persona



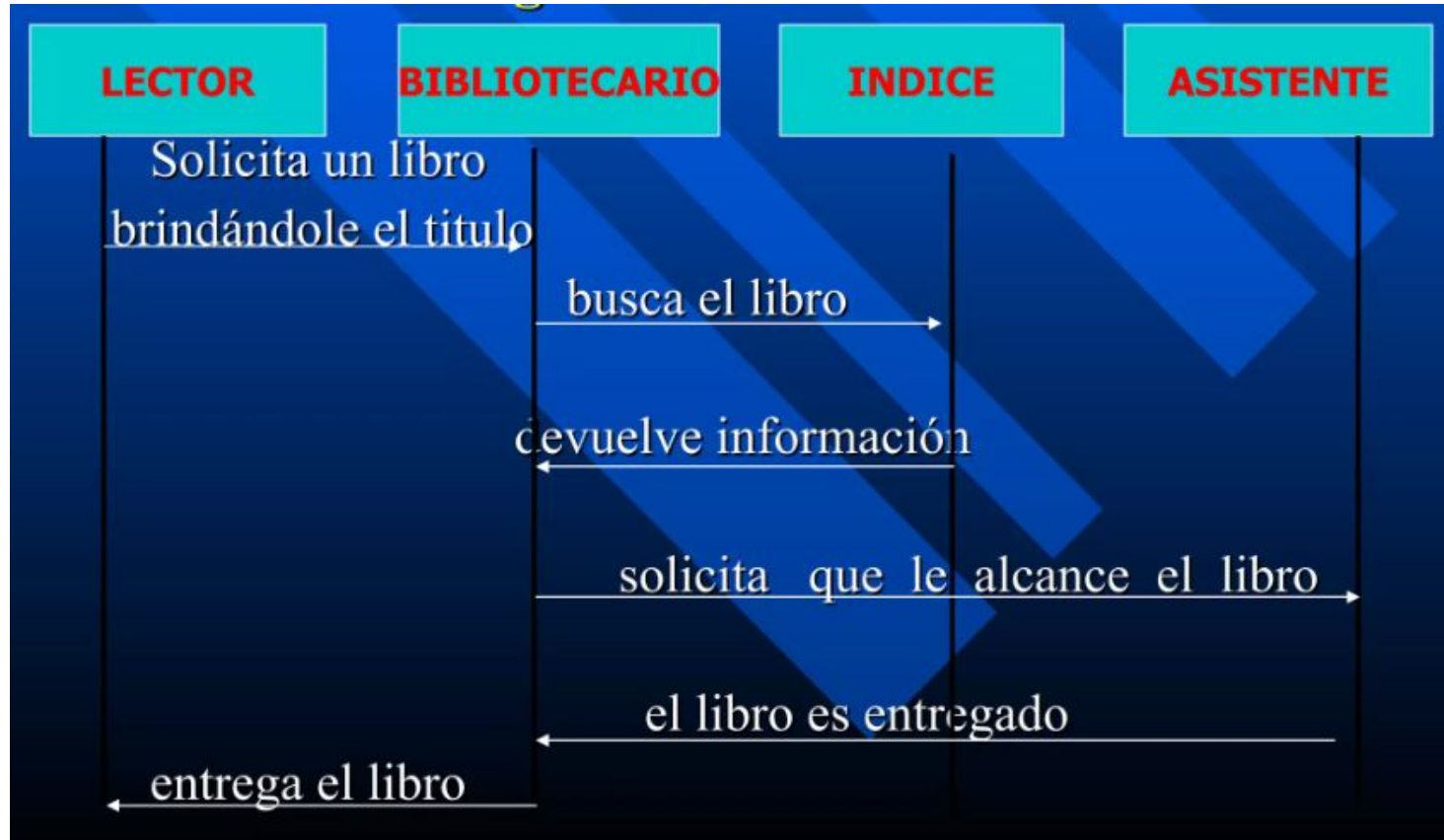
# Diferencias entre diagramas



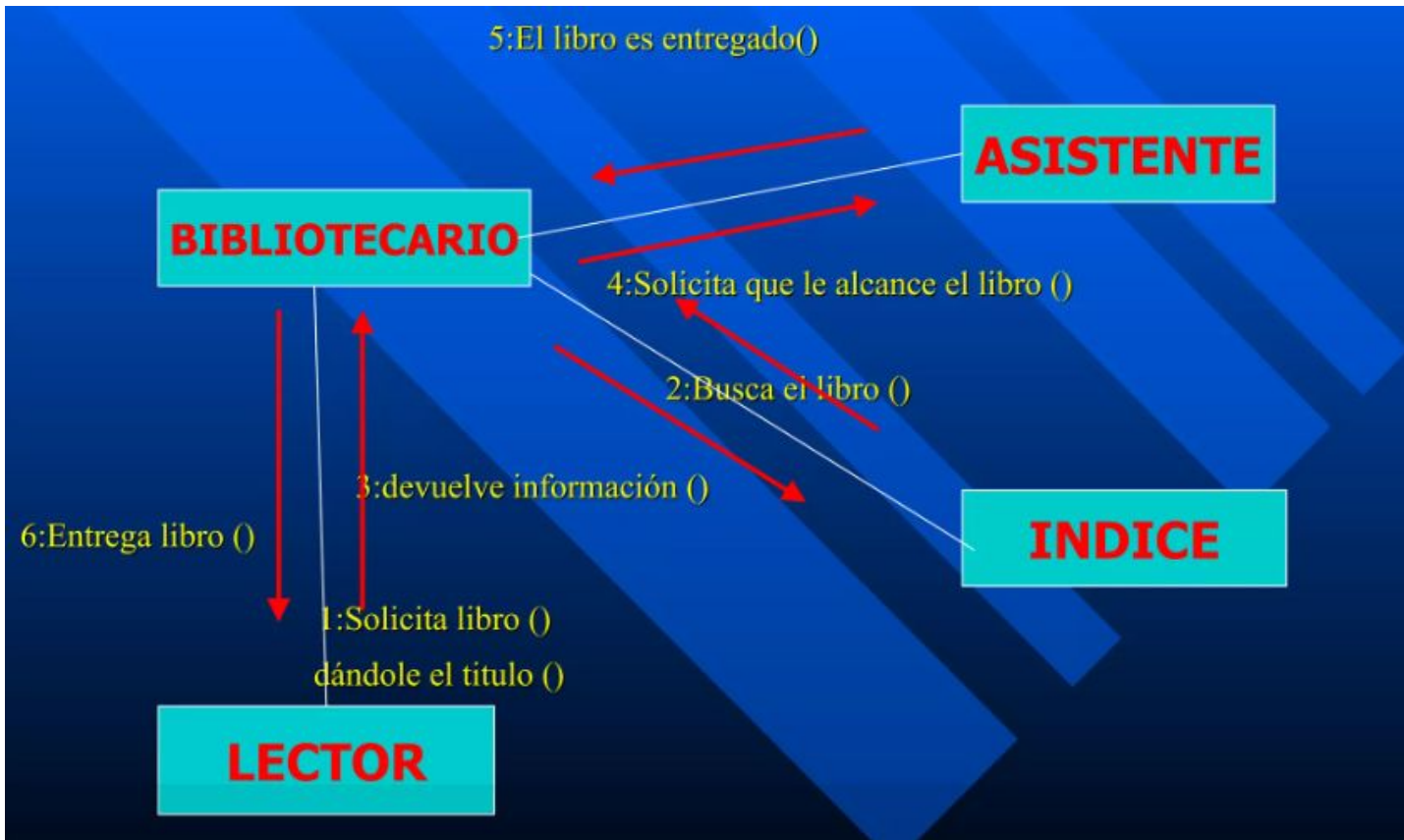
# Diferencias entre diagramas

Aspecto	Diagrama de Secuencia	Diagrama de Colaboración
Enfoque principal	Muestra el orden temporal de los mensajes entre objetos.	Muestra la relación estructural y colaboración entre objetos.
Representación del tiempo	Eje vertical representa el tiempo; mensajes bajan a medida que avanzan.	El tiempo no es gráfico; se indica con números de secuencia en los mensajes.
Visualización de objetos	Objetos alineados horizontalmente con líneas de vida	Objetos distribuidos y conectados con líneas (enlaces).
Orden de lectura	Se lee de arriba hacia abajo siguiendo el flujo temporal.	Se sigue la numeración de mensajes (1, 2, 3...).
Enfoque en diseño	Útil para detallar escenarios paso a paso y comportamiento dinámico.	Útil para mostrar qué objetos colaboran y cómo están conectados.
Complejidad visual	Puede volverse largo si hay muchos mensajes o pasos.	Puede volverse denso si hay muchos objetos o enlaces cruzados.
Uso típico	Análisis de requisitos, modelar casos de uso en detalle.	Diseño orientado a objetos, identificar relaciones y responsabilidades.

# Ejemplo: Diagrama de secuencia



# Ejemplo: Diagrama de colaboración



A decorative network diagram with purple nodes and lines. The nodes are represented by concentric circles, with some having a solid purple center and others being hollow. They are connected by thin black lines. The diagram is positioned around a large, light purple, irregular blob in the center of the slide. There are three main paths: one in the top right, one in the bottom left, and one in the bottom right. Each path consists of several nodes connected in a sequence, with some paths ending in ellipses to indicate they continue.

# ¡Gracias!