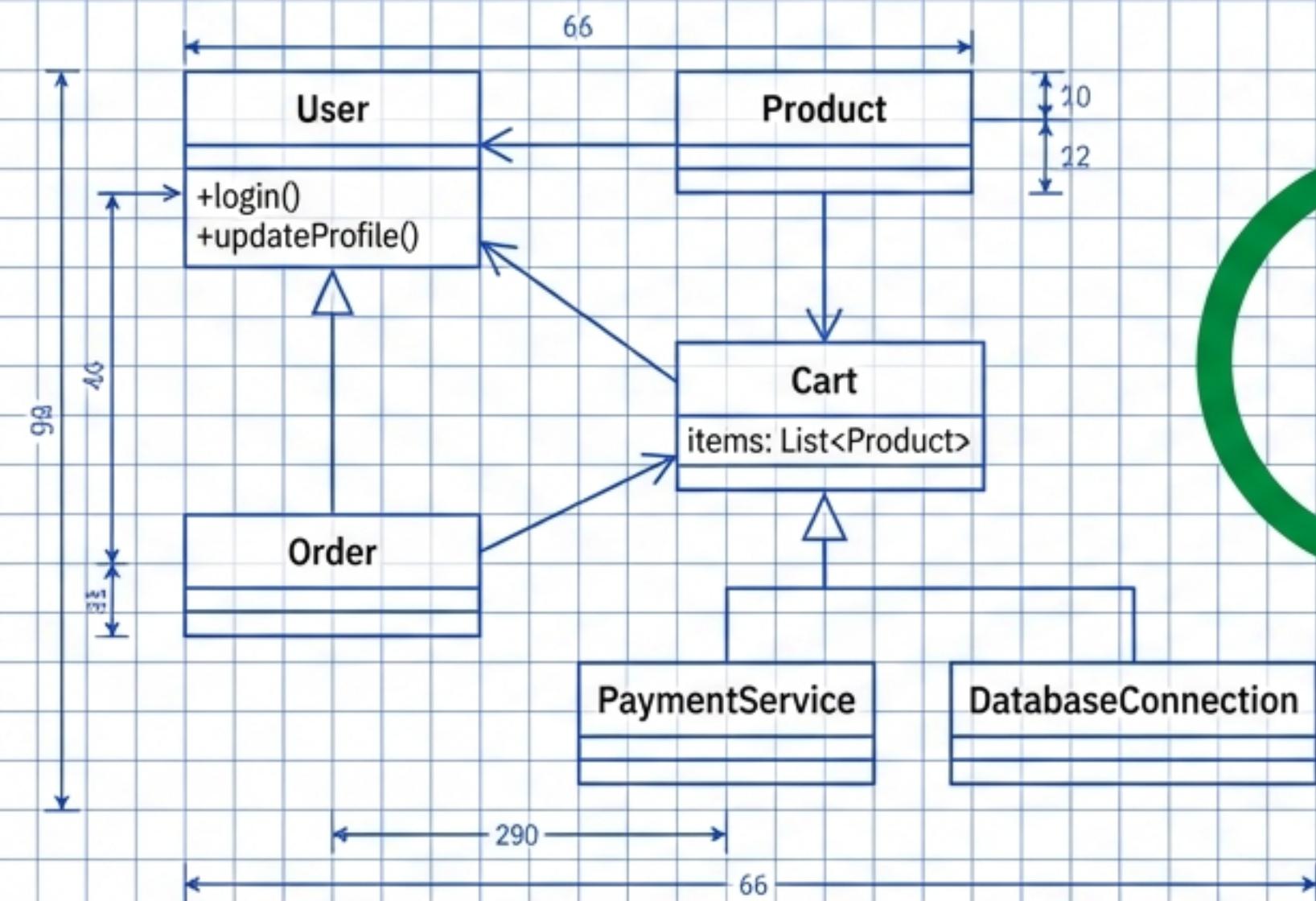


Entornos de Desarrollo: Diagramas de Comportamiento

Del modelado estructural a la ejecución dinámica



Más allá de la estructura de los componentes:
¿Cómo se relacionan cuando pulsamos 'Ejecutar'?

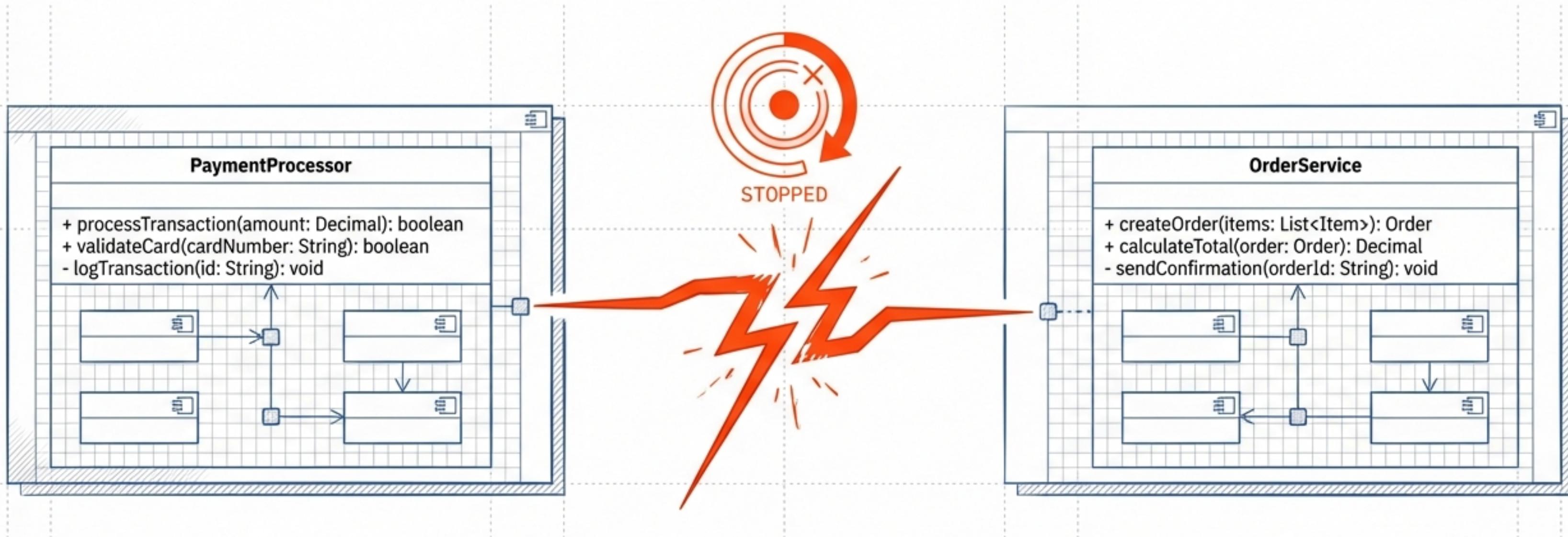
Tema 8: Introducción
y Contextualización

La pieza que falta en el modelado

El desafío: A menudo nos centramos únicamente en describir la estructura de los componentes (Diagramas de Clases).

El vacío: Esto ignora cómo se comportarán y relacionarán dichos componentes cuando la aplicación pase a un estado de ejecución.

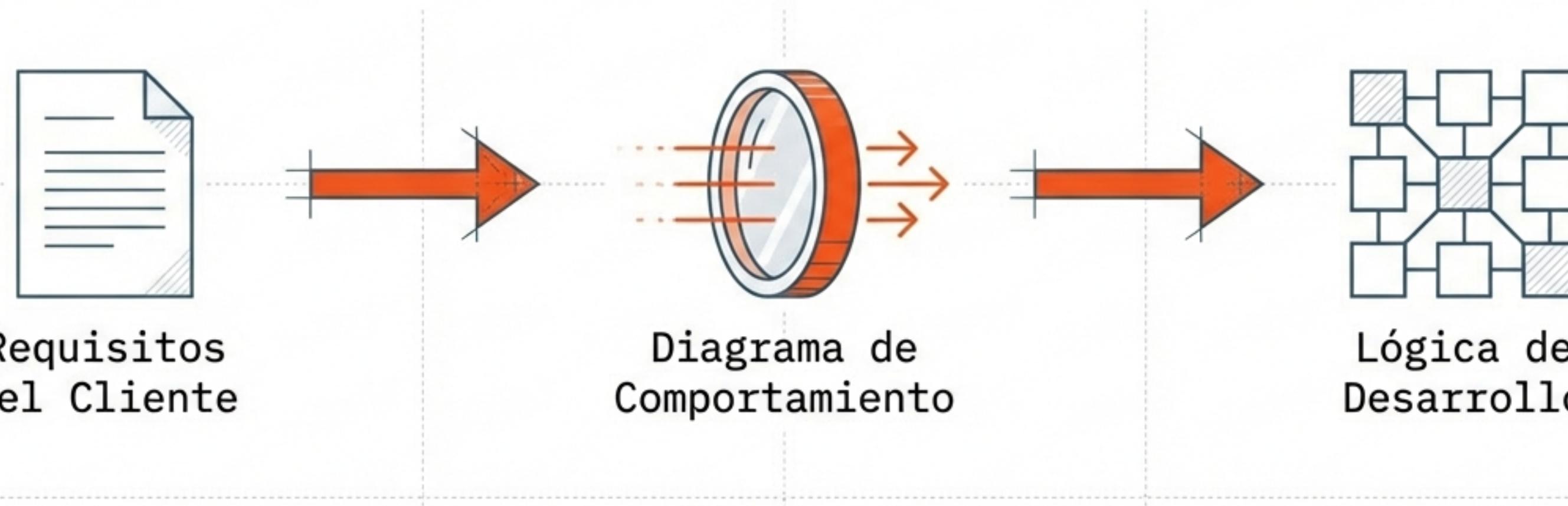
La necesidad: Utilizar UML para establecer el flujo de comunicación en tiempo de ejecución.



“A medida que el sistema crece, el modelado del flujo de ejecución se vuelve crítico para evitar errores.”

¿Qué es un Diagrama de Comportamiento?

Una representación visual que expresa la secuencia de estados por la que pasa un objeto a lo largo de su vida para responder a eventos del sistema.



- **Simplificación**

Traduce los requisitos del cliente a lógica técnica.

- **Documentación**

Visualiza cambios constantes en el sistema.

- **Visión**

Muestra cómo los objetos utilizan sus funcionalidades para relacionarse.

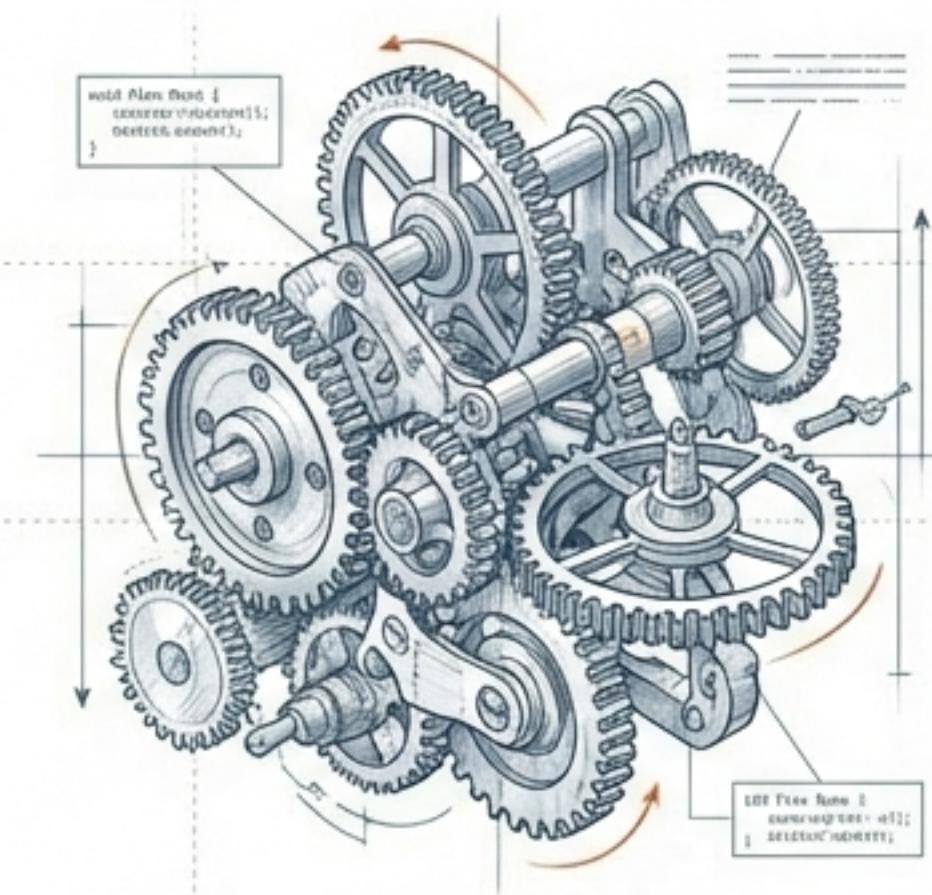
Las dos dimensiones del comportamiento (UML 2.4.1)

1. Comportamiento de Ejecución



- Ocurre cuando el objeto está en su flujo de ejecución.
- Relacionado directamente con la implicación de métodos a través de su interfaz.
- Acceso a características estructurales.

2. Comportamiento Emergente



- Ocurre por la interacción de uno o más objetos.
- Relación indirecta (objetos compuestos por otros).
- Suele complicar el diagrama debido a las múltiples partes participantes.

El espectro de los diagramas de comportamiento



Diagrama de Casos de Uso

La perspectiva del usuario. Muestra las operaciones del sistema y su relación con el entorno.

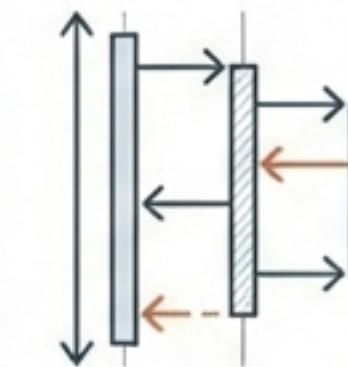


Diagrama de Secuencia

La perspectiva del tiempo. Describe interacciones entre objetos a lo largo de una línea temporal.

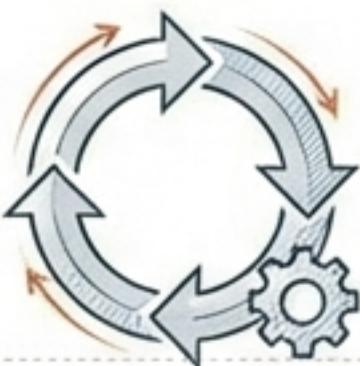


Diagrama de Estados

La perspectiva del ciclo de vida. Muestra condiciones de cambio y estados de un objeto.

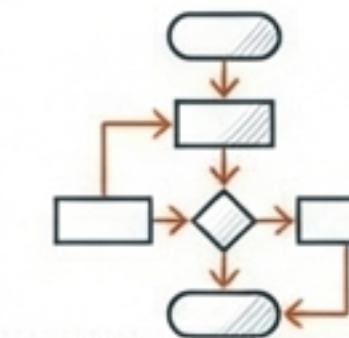


Diagrama de Actividad

La perspectiva del flujo de trabajo. Especialización en estados de acción y transiciones de tareas.

Diagrama de Casos de Uso: La visión funcional

Nivel de Abstracción: Alto. No indica el flujo de ejecución detallado, sino qué tareas se pueden realizar.

Enfoque: Punto de vista del usuario (ej. Cliente WEB).

Elementos: Actores (Cliente, PayPal) y Casos de Uso (Ver producto, Hacer pedido).

Takeaway: Ideal para especificar funcionalidad.

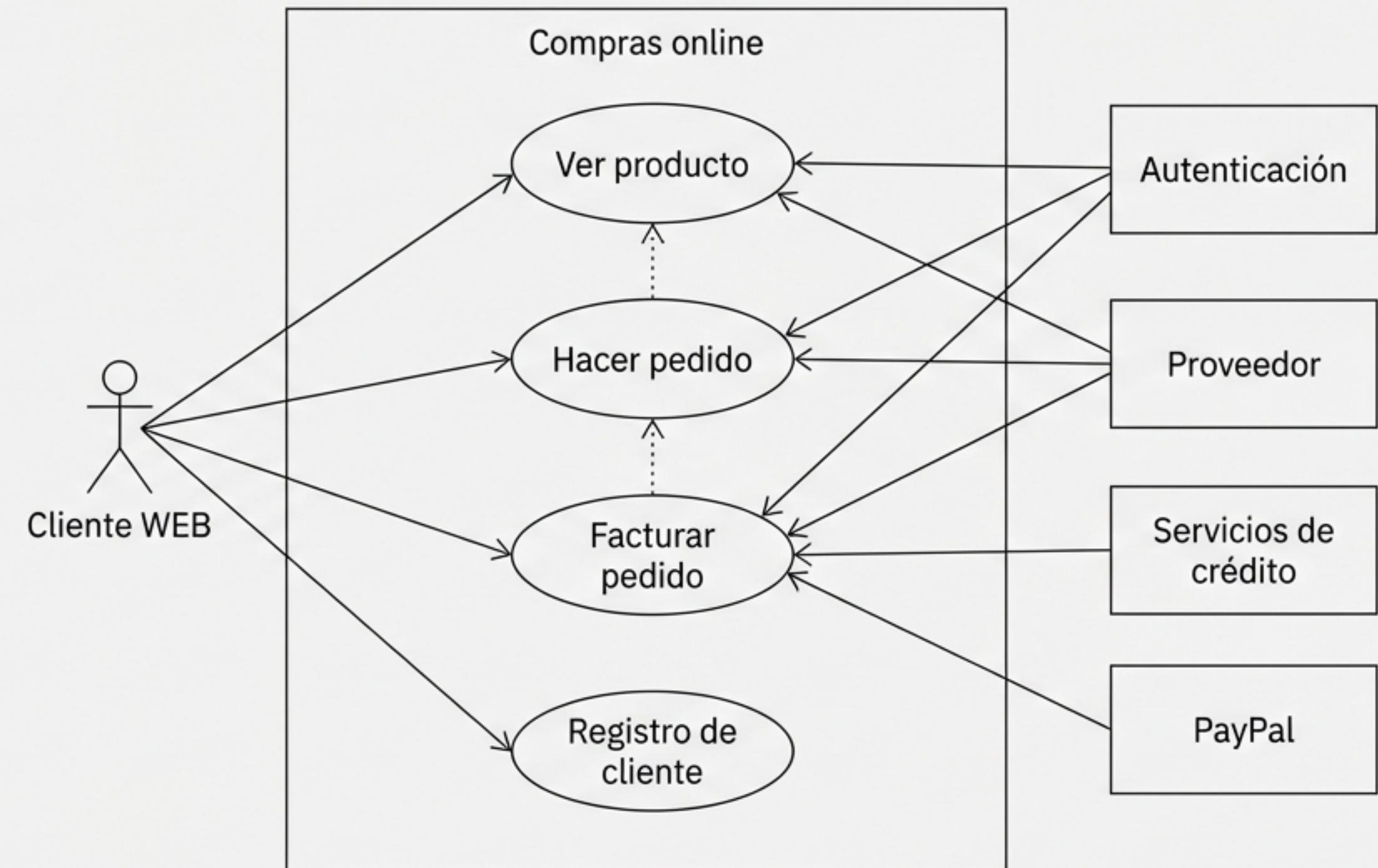
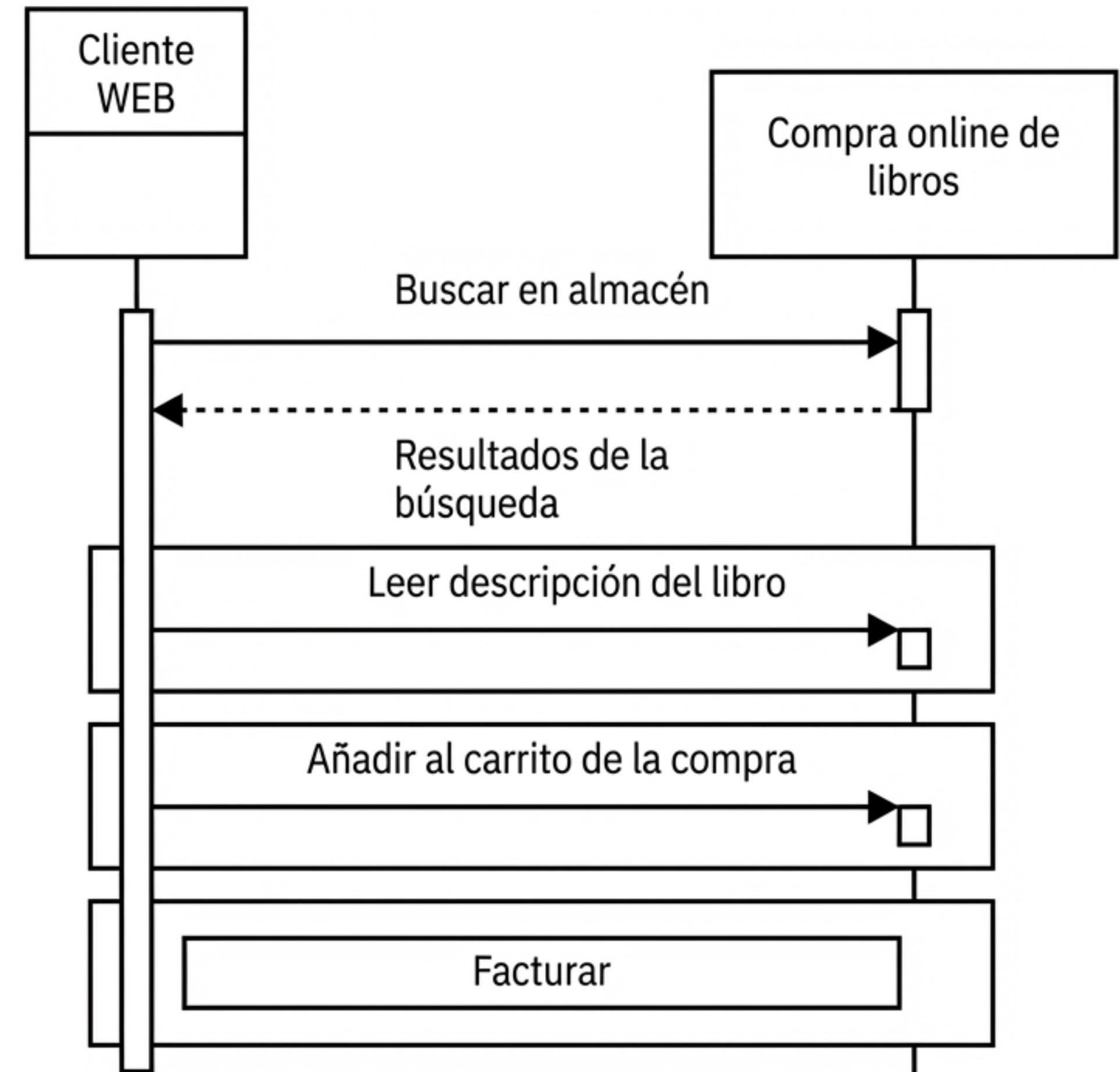


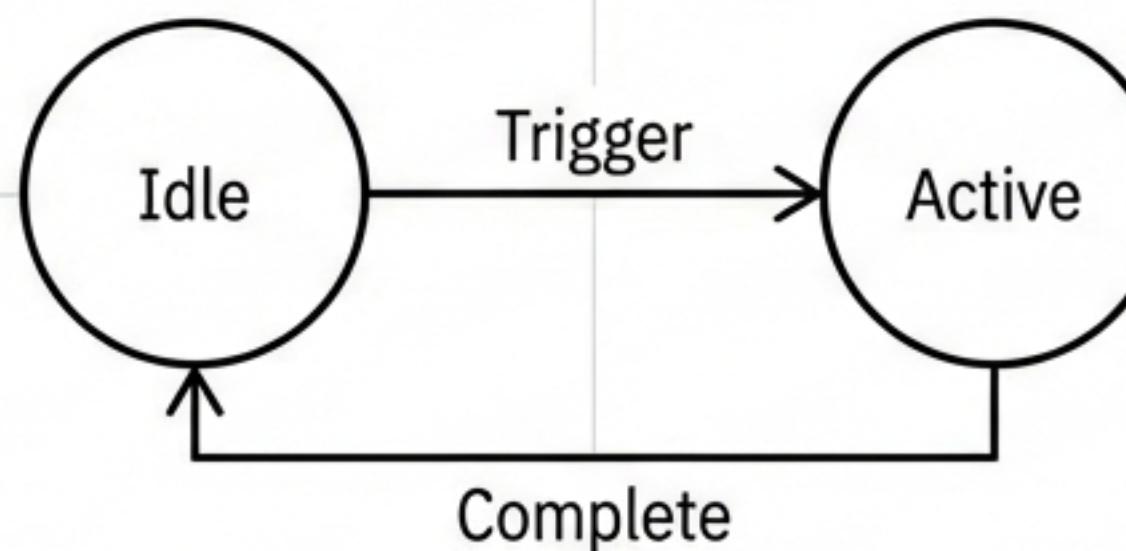
Diagrama de Secuencia: Secuencia: La visión interactiva

- **Factor Clave:** El tiempo.
Muestra las interacciones cronológicas entre objetos.
- **Función:** Amplía el grado de detalle de los casos de uso.
- **Flujo:** Muestra paso a paso la comunicación.
- **Takeaway:** Esencial para entender el orden de las llamadas.



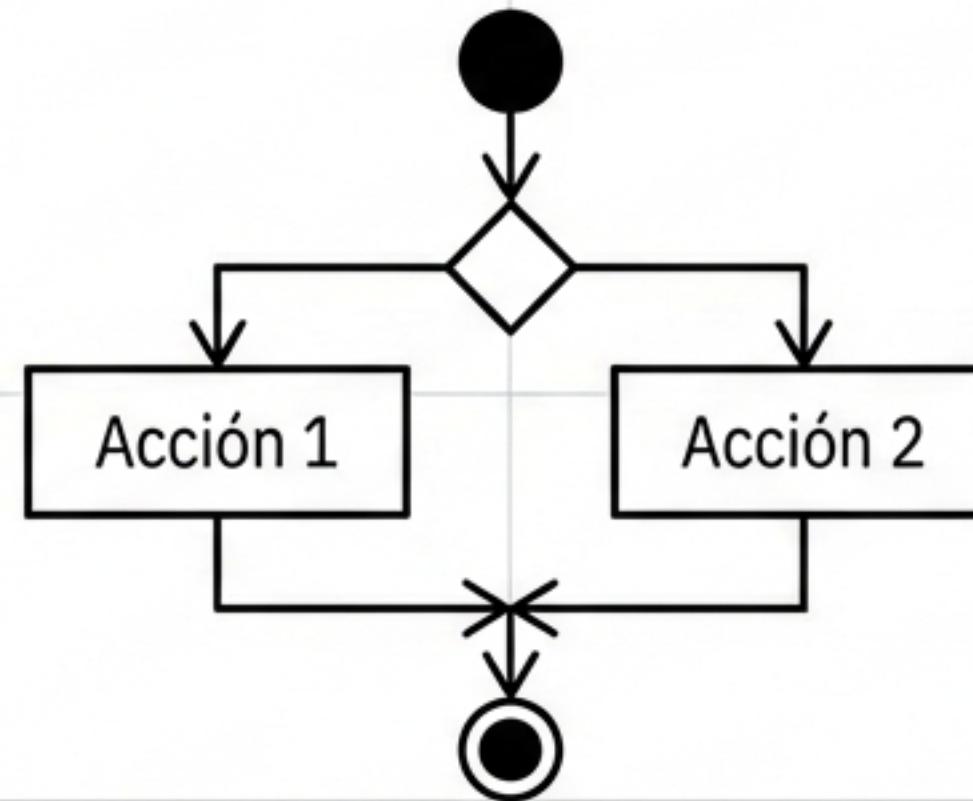
Diagramas de Estado y Actividad

Diagramas de Estados



Indican los estados por los que pasa un objeto. Definen las **condiciones** necesarias para cambiar de estado.

Diagramas de Actividad

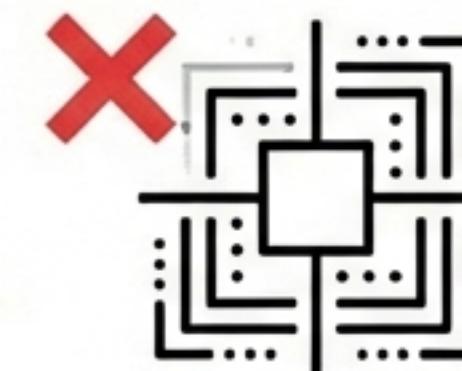


Una **especialización** de los diagramas de estado. Centrados en estados de **acción** y transiciones al finalizar una tarea. Útiles para procesos complejos.

Caso Práctico 1: El dilema de Calidad

El Nudo

El departamento de calidad sugiere usar solo un **Diagrama de Clases** para informar a los desarrolladores sobre las tareas en tiempo de ejecución.



El Desenlace

Veredicto: Decisión incorrecta.

Solución: Se requiere un Diagrama de Comportamiento.

Por qué: Permite plantear pruebas de calidad sólidas y ayuda a localizar errores (bugs) en el flujo lógico.

Clases + Comportamiento = Cobertura Completa

Caso Práctico 2: Eligiendo la herramienta correcta



El Desafío

Visualizar el flujo exacto desde que el usuario pulsa "RESERVAR" hasta el fin del proceso.

Propuesta incorrecta: Usar un Diagrama de Casos de Uso.

El Análisis

El Caso de Uso solo muestra *que* existe la funcionalidad. No muestra *el cómo*.

Corrección: Se necesita un **Diagrama de Secuencia**.

Evidencia: Necesitamos ver el flujo de ejecución y los pasos temporales.

Campo de aplicación: ¿Cuándo usarlos?



Evento

Cambio de estado repentina en el sistema.



Señal

Notificación de una condición externa.



Trigger

Mecanismo que inicia un proceso.



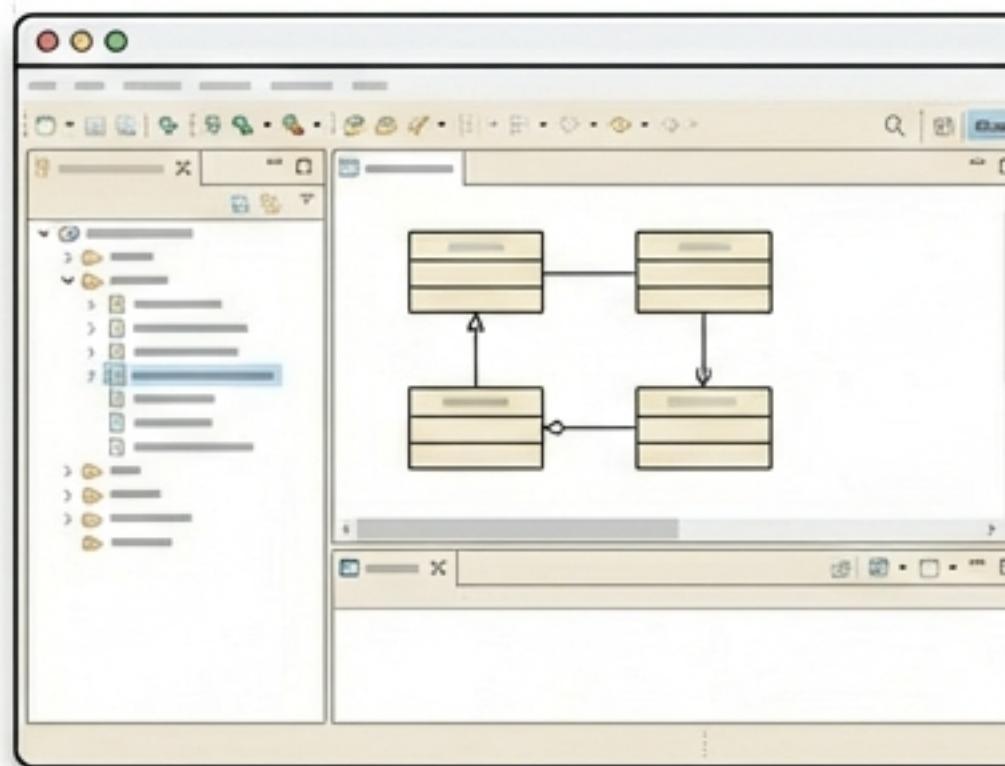
Petición en Búfer

Solicitud almacenada para ejecución posterior.

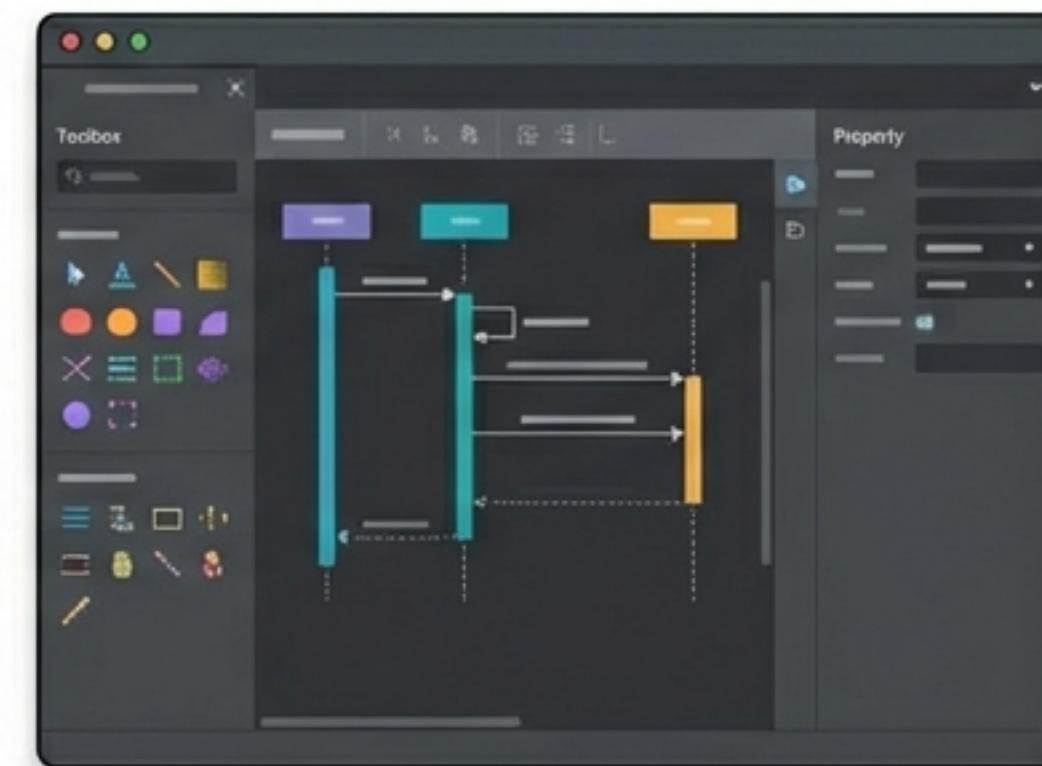
Objetivo: Entender qué tareas pertenecen a qué procesos para evitar cuellos de botella.

Herramientas y Entornos (CASE)

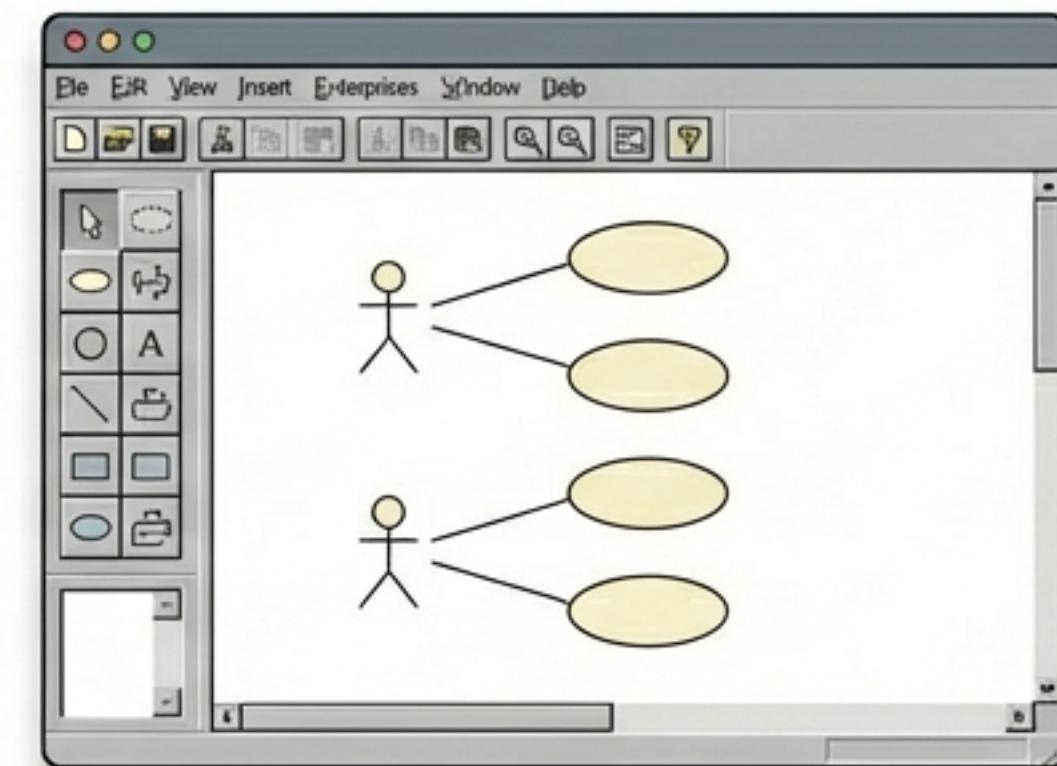
Las herramientas **CASE** (Computer Aided Software Engineering) permiten generar código a partir de diagramas (ingeniería inversa/directa).



Papyrus UML



StarUML



Rational Rose

Visualización de interacciones complejas e integración en el IDE.

Estrategia: El equilibrio de la documentación



Enfoque Multidisciplinar

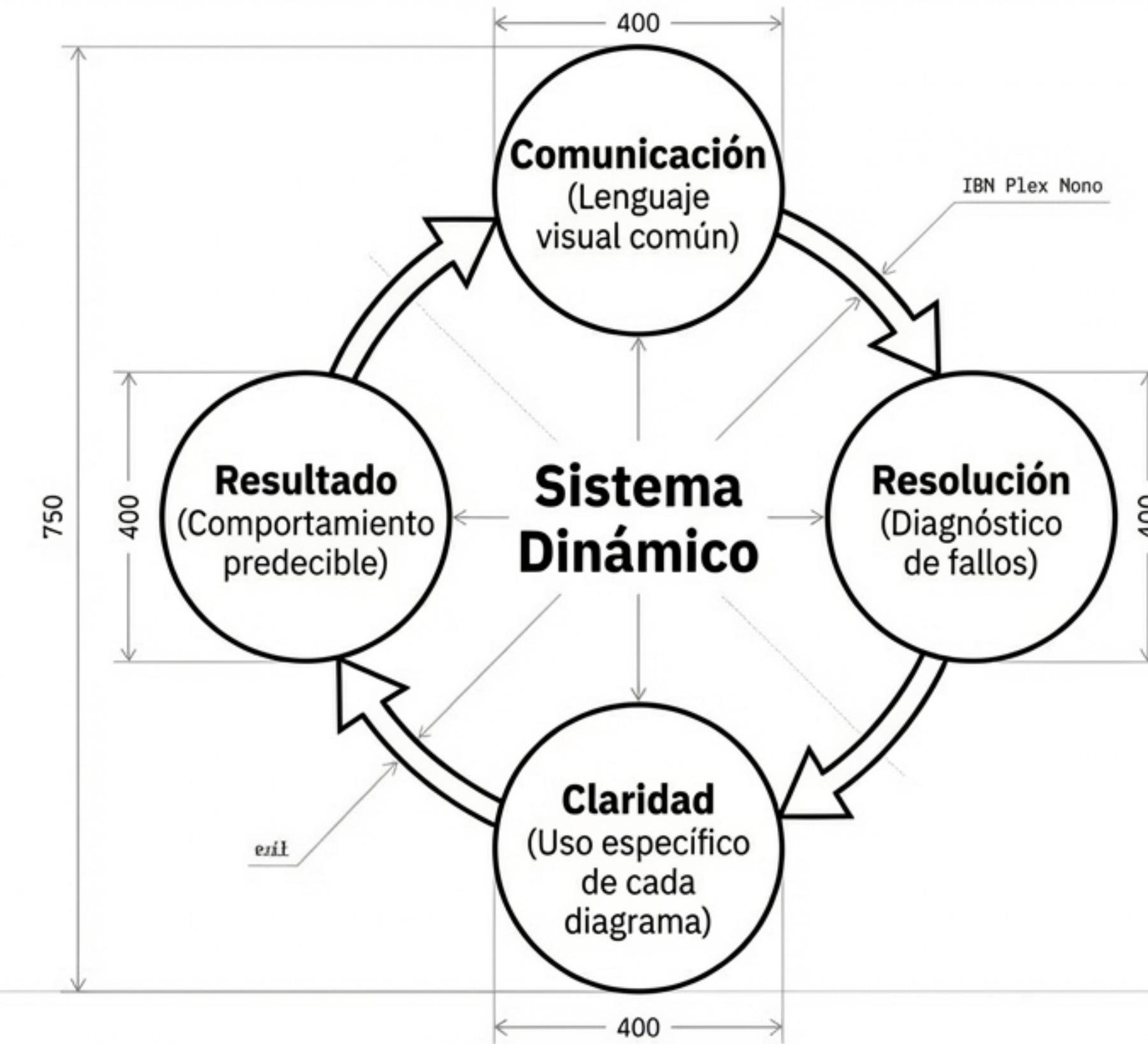
Una documentación enriquecida reduce errores y facilita el mantenimiento.

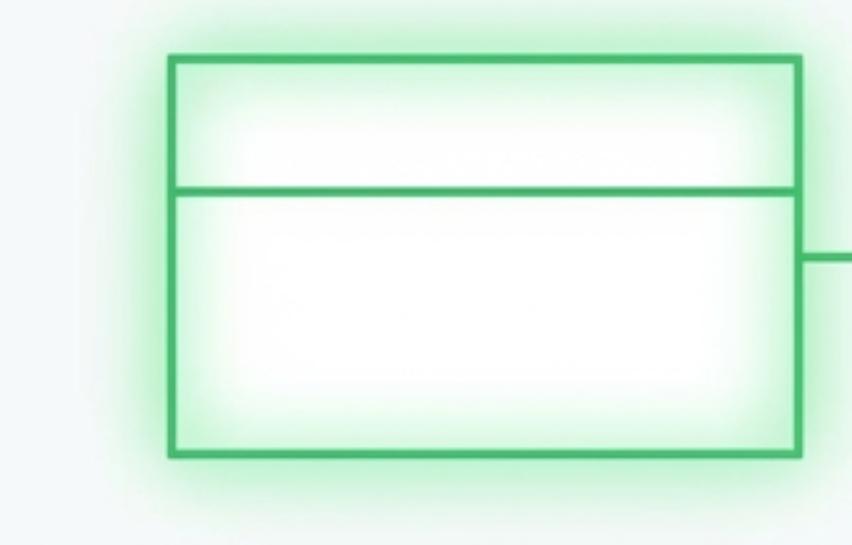
La Advertencia Ágil

Coste de Mantenimiento: Múltiples diagramas obligan a mantener la información por duplicado.

Best Practice: Es necesario llegar a un consenso entre el número de diagramas y las funcionalidades críticas a representar.

Resumen: La visión completa del sistema





Diseñar la estructura es solo el principio.
Modelar el comportamiento es lo que da
vida al sistema.

Conocer la forma de comunicación en tiempo de ejecución nos
permite realizar un diseño completo de todo el sistema.