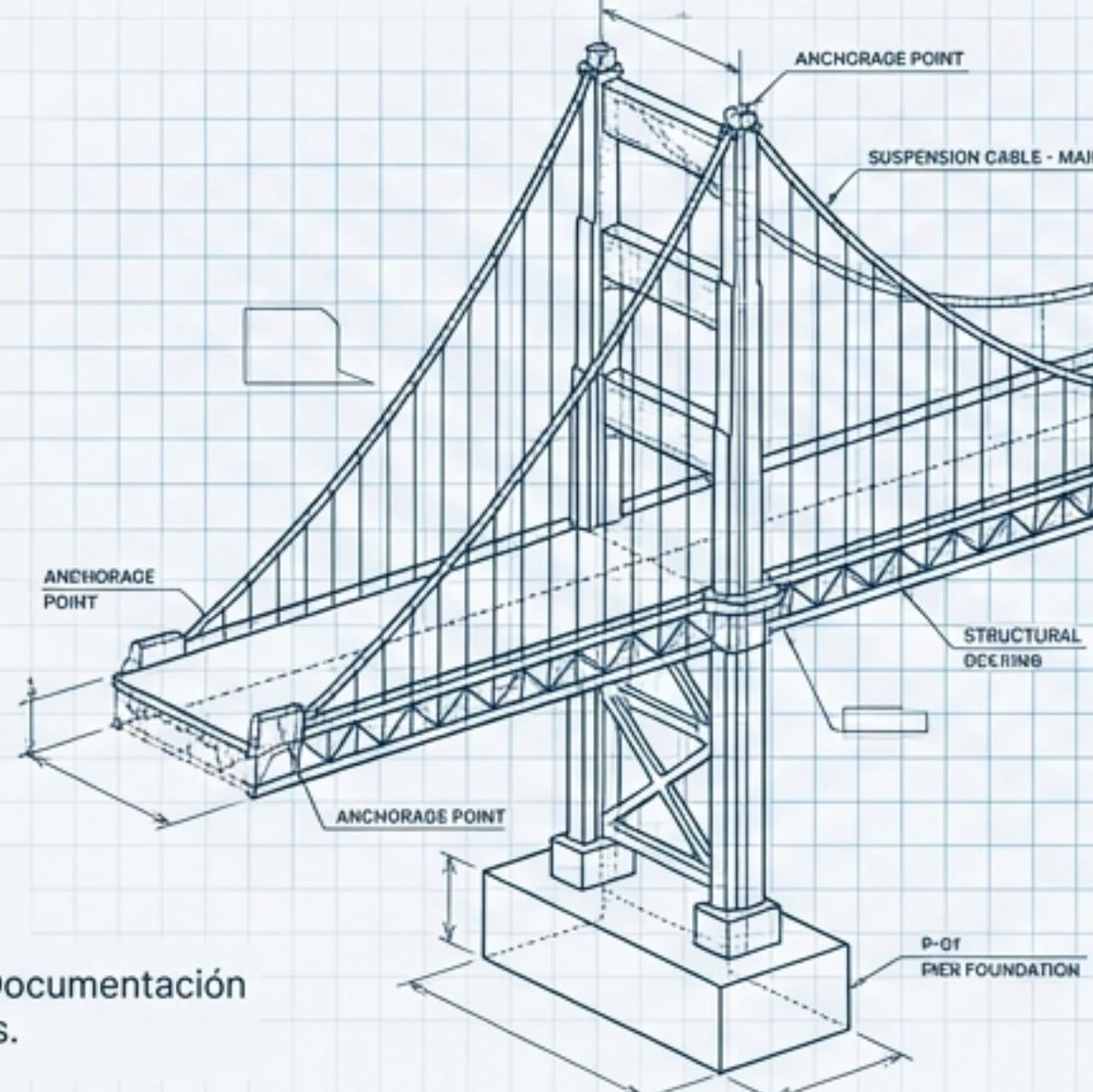


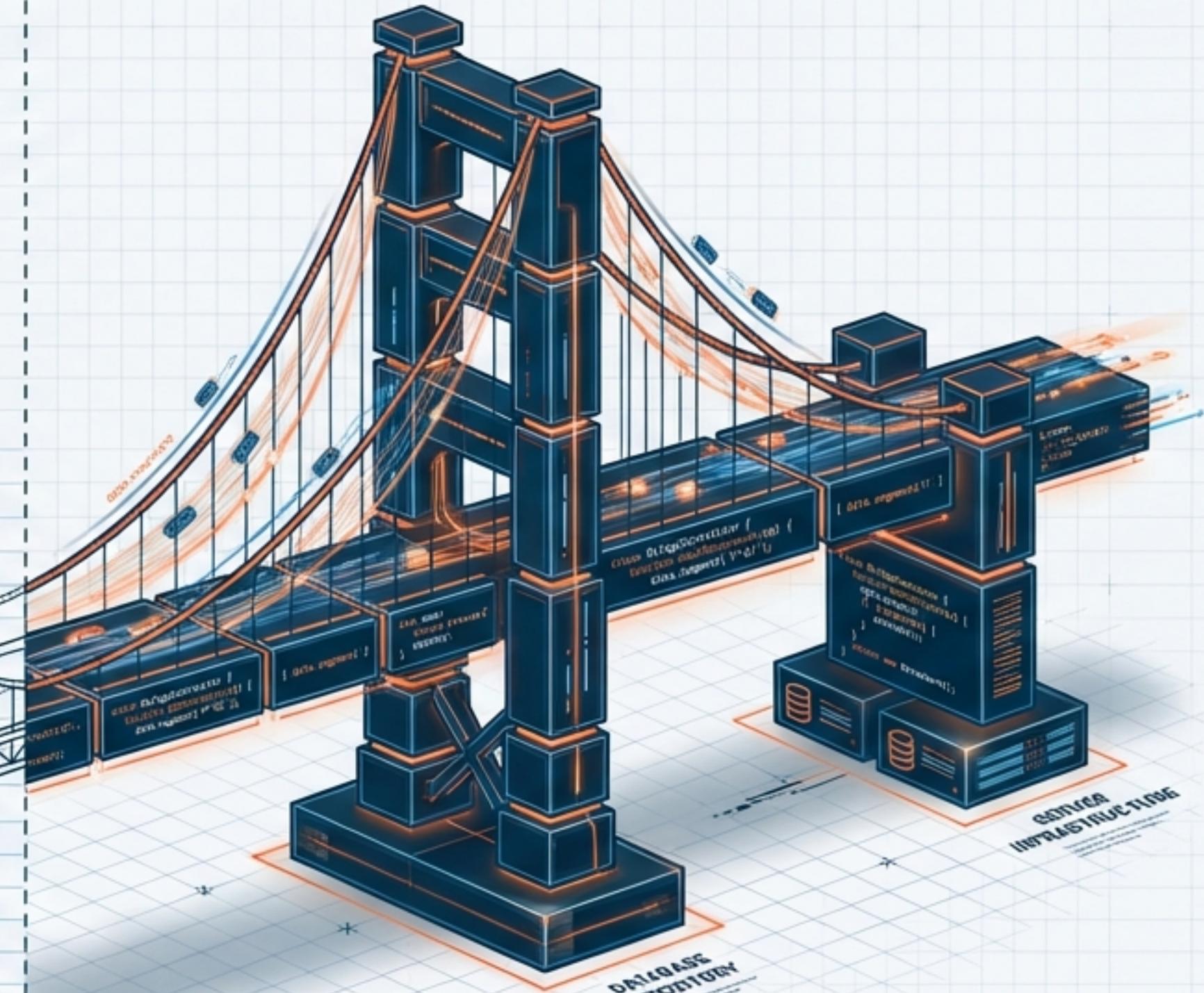
Diagramas de Casos de Uso: El puente entre requisitos y código

Unidad 9 - Entornos de Desarrollo



Inter Regular

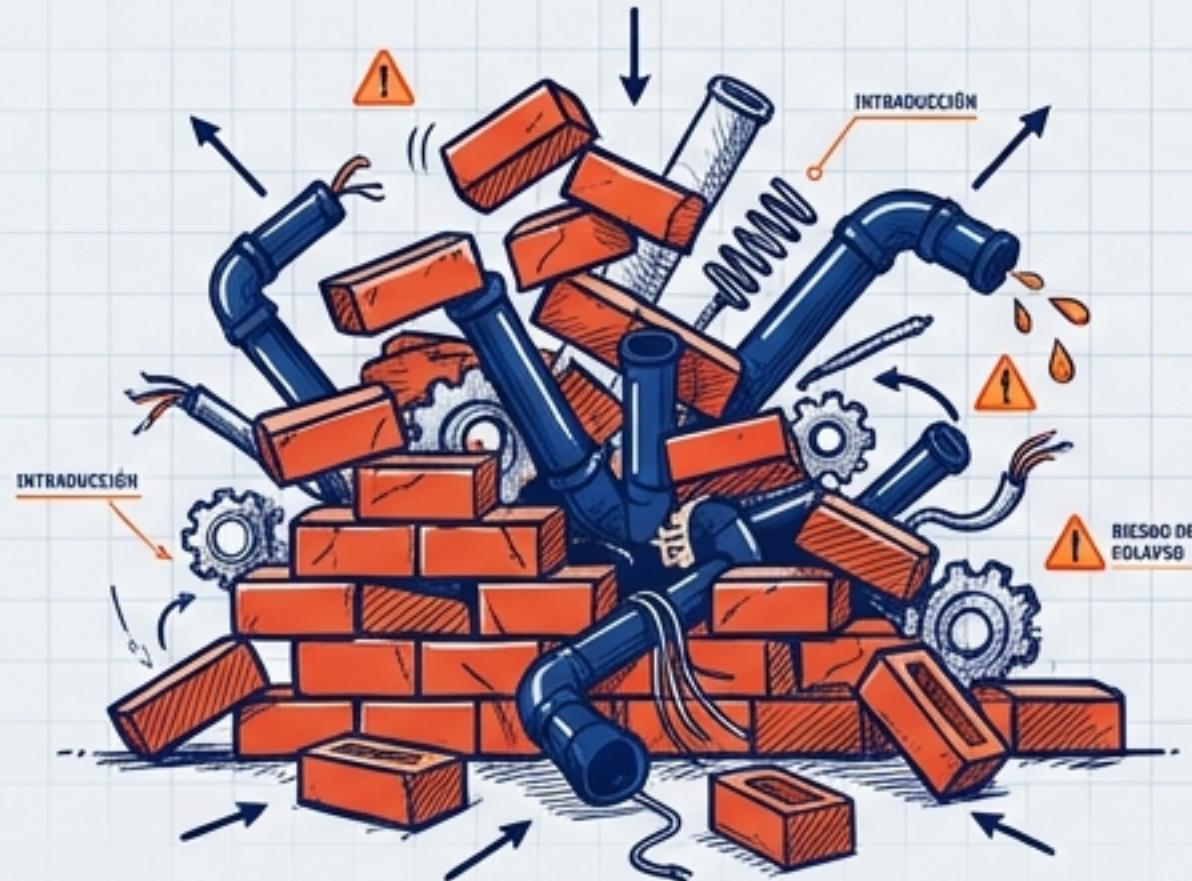
Diseño, Simbología y Documentación para sistemas robustos.



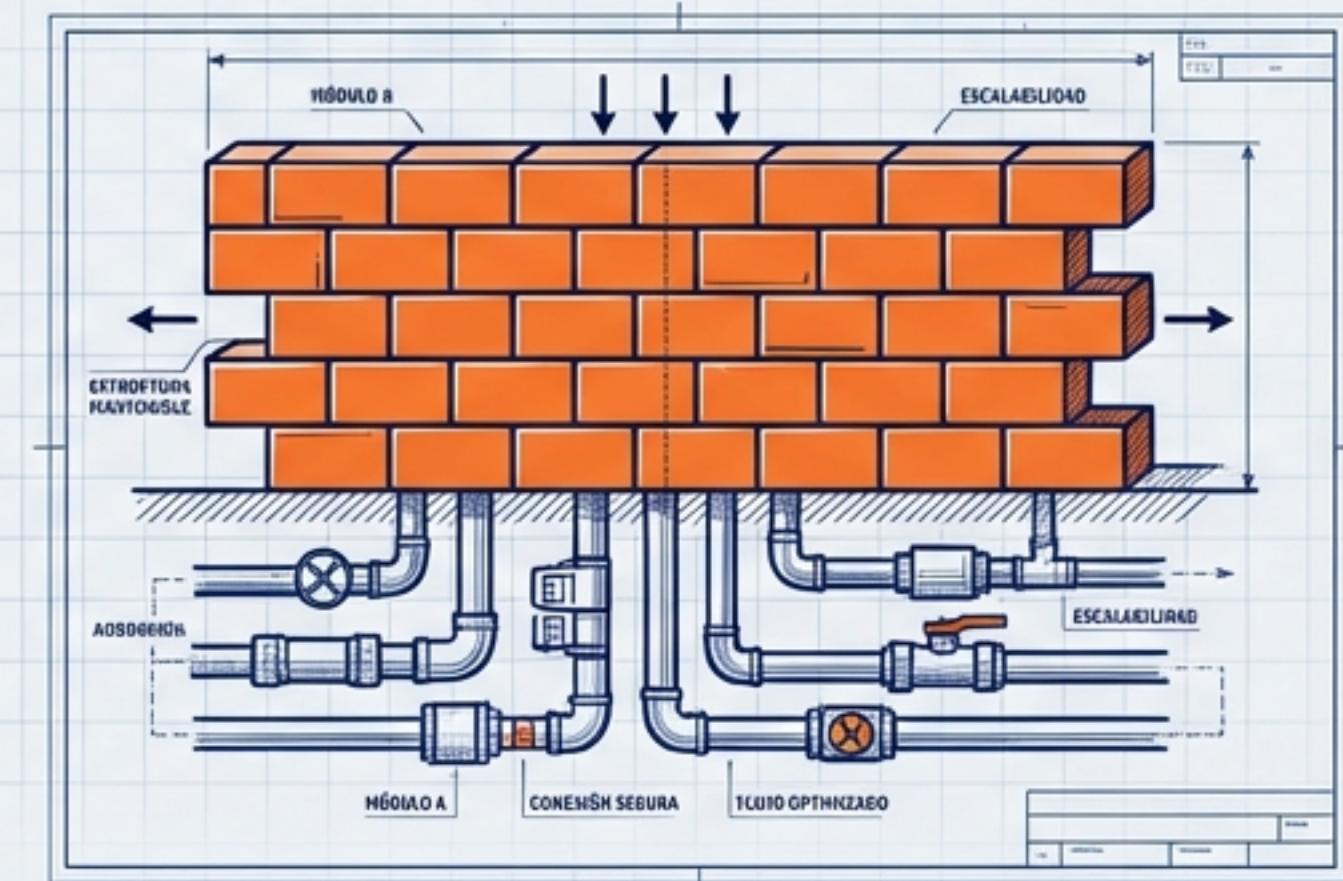
El peligro de la implementación inmediata



Sin Diseño ✗



Con Diseño ✓



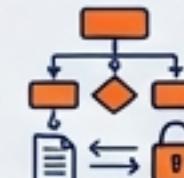
El Problema: Requisitos en crudo.

Los requisitos iniciales a menudo son imprecisos y carecen de estructura, presentándose como un conjunto desordenado de ideas y necesidades, análogo a la pila caótica de materiales sin clasificar.



El Riesgo: Saltar directamente al código parece un ahorro de tiempo, pero resulta en sistemas difíciles de mantener y sin reutilización.

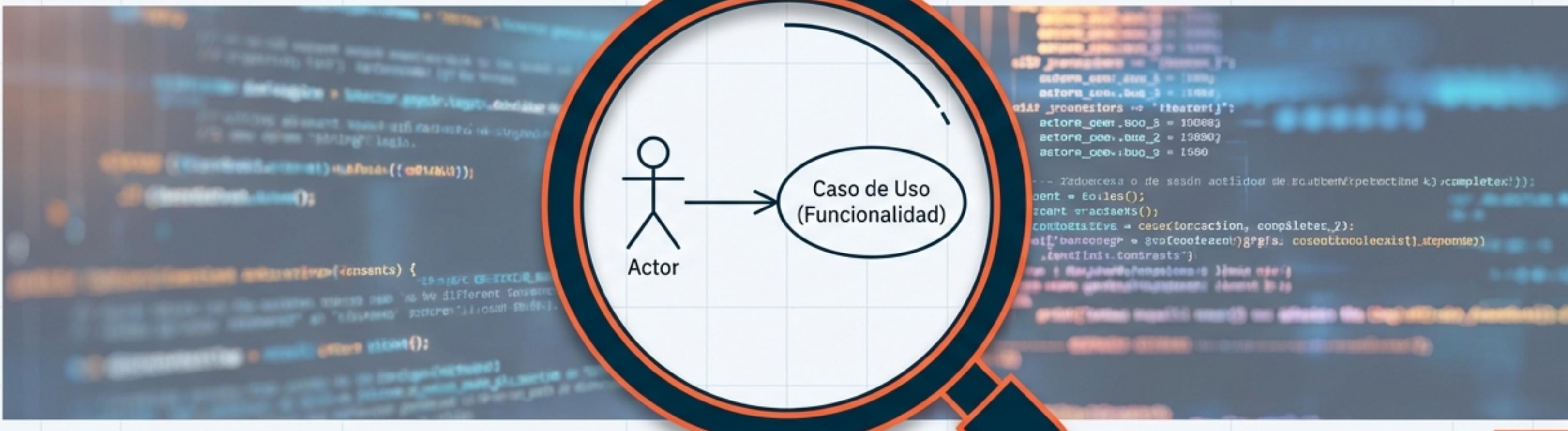
Este enfoque precipitado genera deuda técnica, aumenta la complejidad, introduce errores críticos y crea un código monolítico y frágil que se vuelve costoso de modificar a largo plazo.



La Solución: Necesitamos una capa de traducción formal.

Implementar una fase de diseño estructural y arquitectónico permite transformar requisitos crudos en un modelo claro, modular y documentado, asegurando la estabilidad, la escalabilidad y la eficiencia del desarrollo.

¿Qué es un Diagrama de Casos de Uso?



Abstracción

Determina el comportamiento sin entrar en la complejidad del código.



Interacción

Muestra cómo entidades externas interactúan con el sistema.

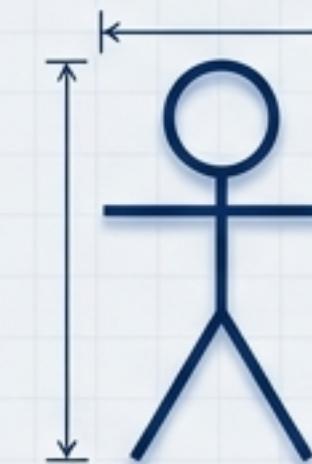


Estructura

Establece relaciones claras entre funcionalidades.

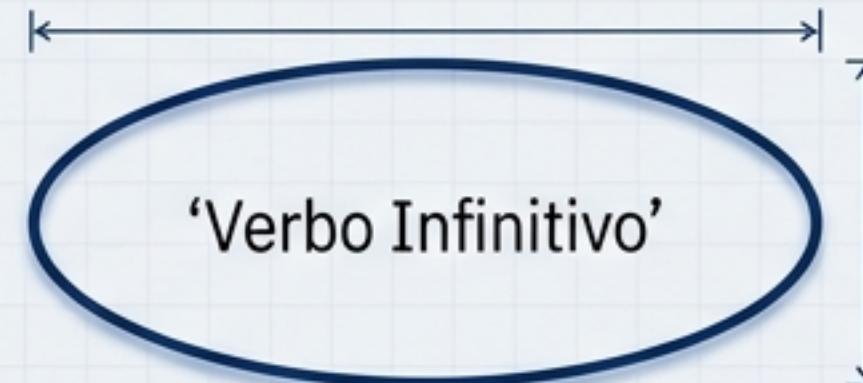
“Permiten indicar cómo los diferentes actores pueden trabajar con el sistema.”

Anatomía del Diagrama: Simbología Esencial



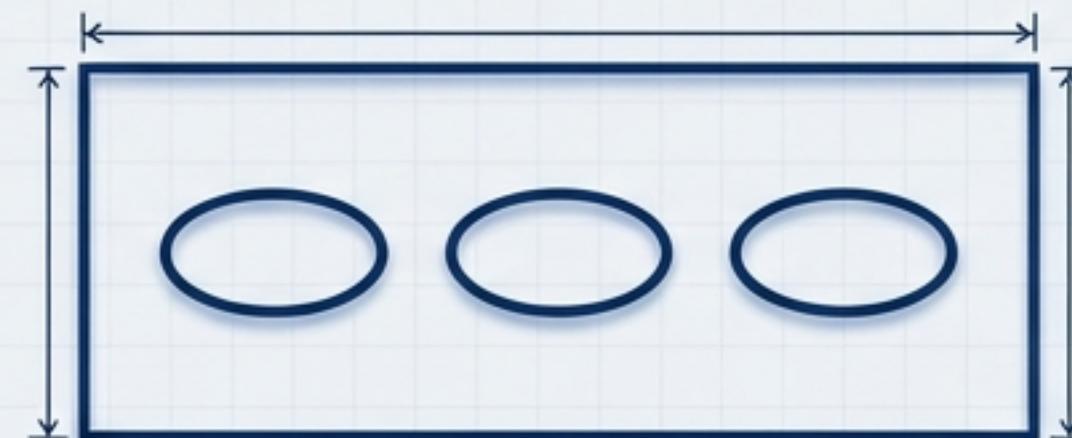
Actor (El Rol)

Entidades (usuarios o sistemas) que interactúan con los casos de uso.



Caso de Uso (La Funcionalidad)

La acción específica. Se etiqueta con un verbo en infinitivo (ej. 'Registrar usuario').



Sistema (El Límite)

Agrupa los casos en un módulo único. Define qué está dentro y qué fuera.



Relaciones (El Vínculo)

Conectan actores con casos, definiendo dependencias y flujos.

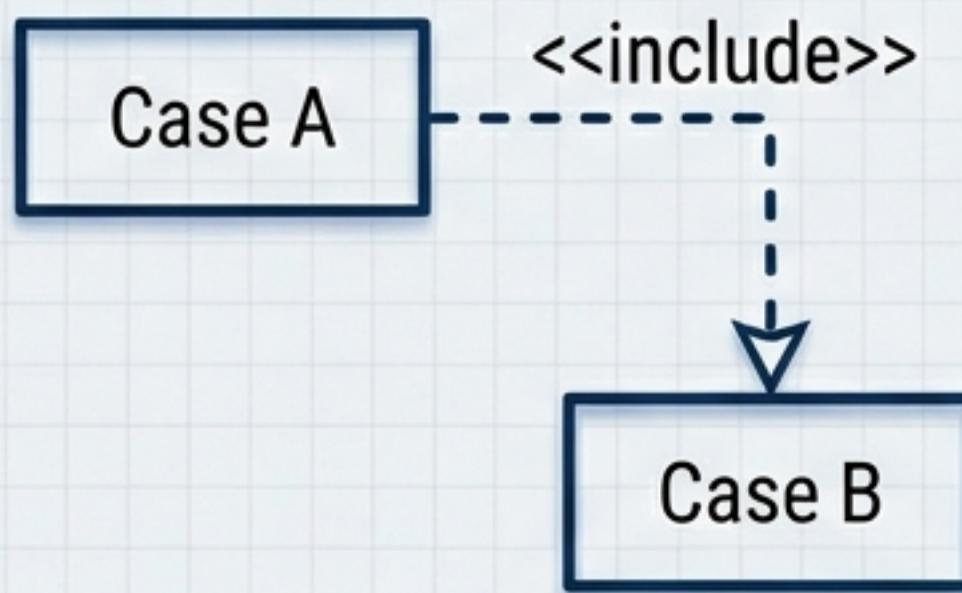
Dinámicas de Interacción: Roles y Actores



- El diagrama define permisos específicos.
- Una misma persona puede interpretar varios roles.

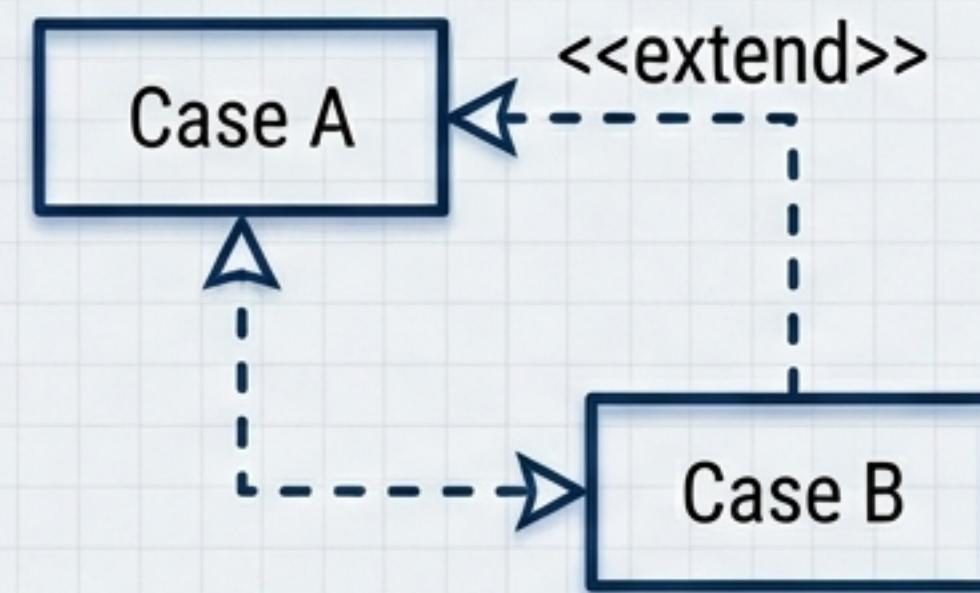
El Tejido Conectivo: Tipos de Relaciones

Inclusión (Include)



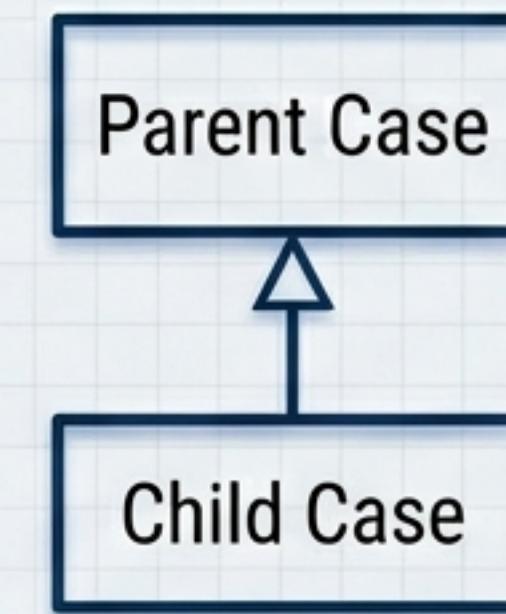
Obligatorio. Para ejecutar A, es necesario ejecutar B.

Extensión (Extend)



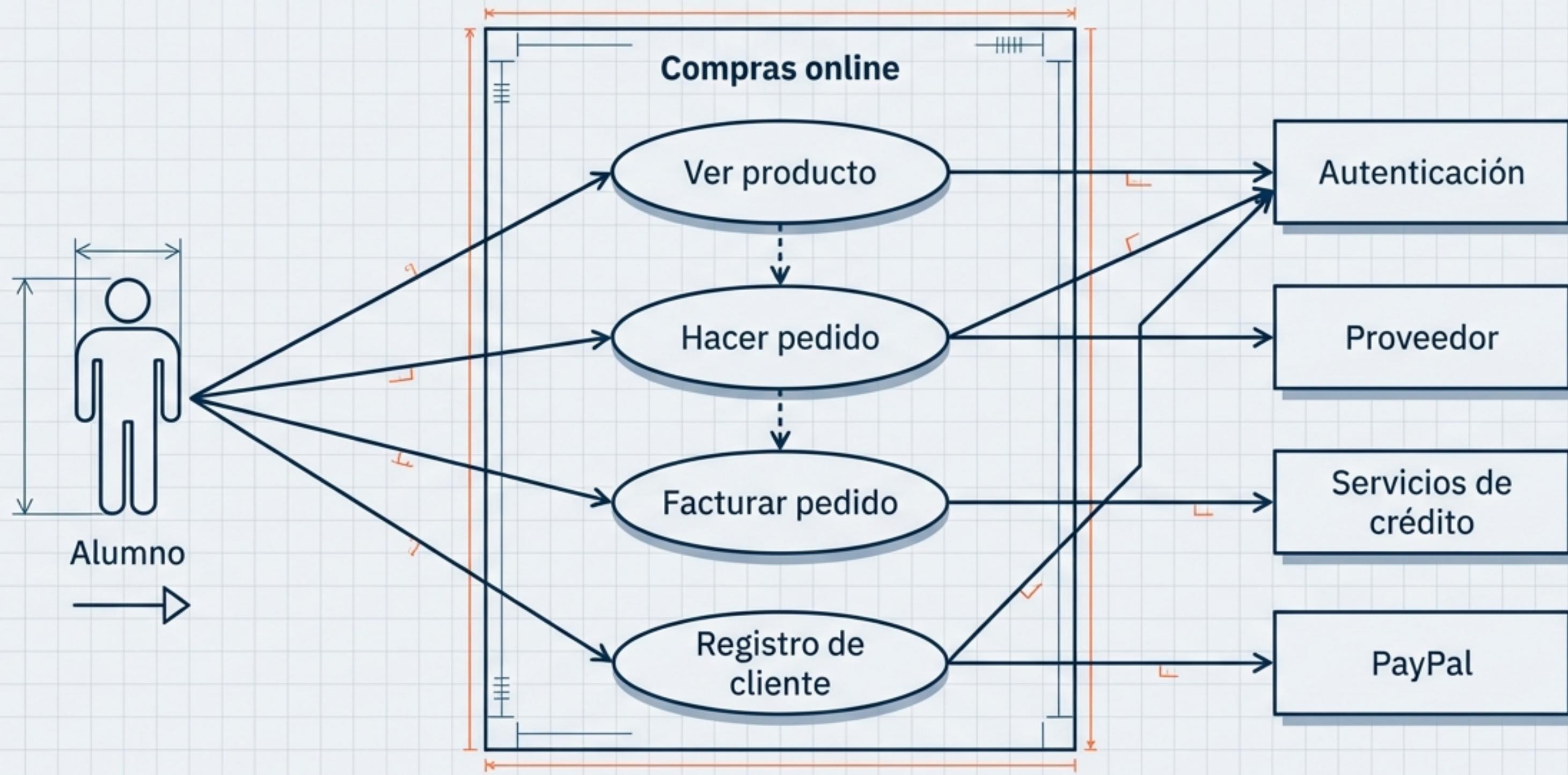
Opcional. Modifica el comportamiento bajo ciertas condiciones.

Generalización (Herencia)

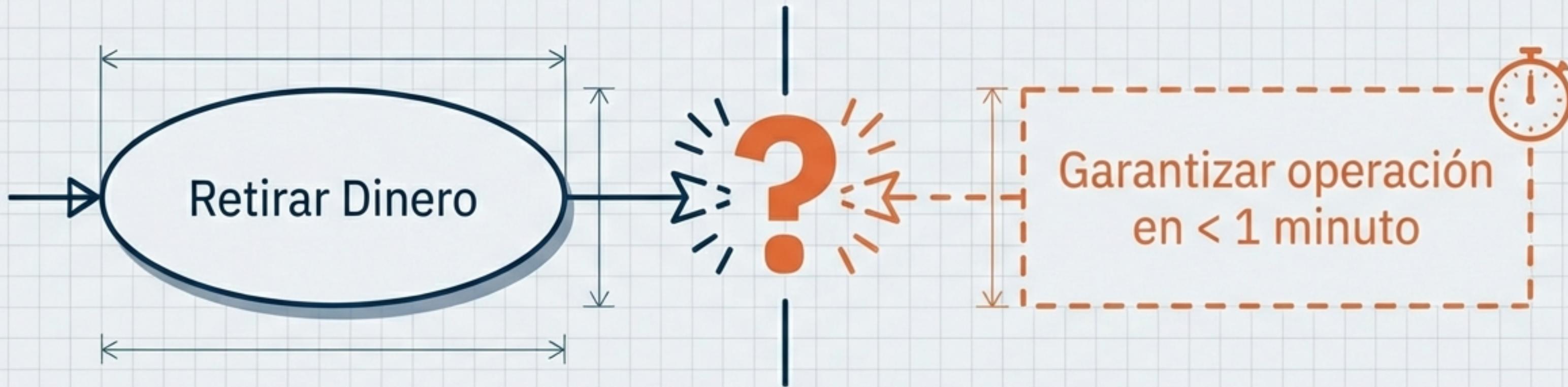


Jerarquía. Casos hijos se especializan a partir de un caso padre.

En la Práctica: Sistema de Compras Online



Las limitaciones de lo visual



- **El Conflicto:** Un diagrama puede mostrar la acción, pero no el rendimiento.
- **Caso Práctico:** El cliente bancario exige velocidad (500ms). Esto es invisible en el dibujo.
- **La Lección:** Los diagramas capturan funcionalidad, no calidad.

Completando la Historia: La Documentación

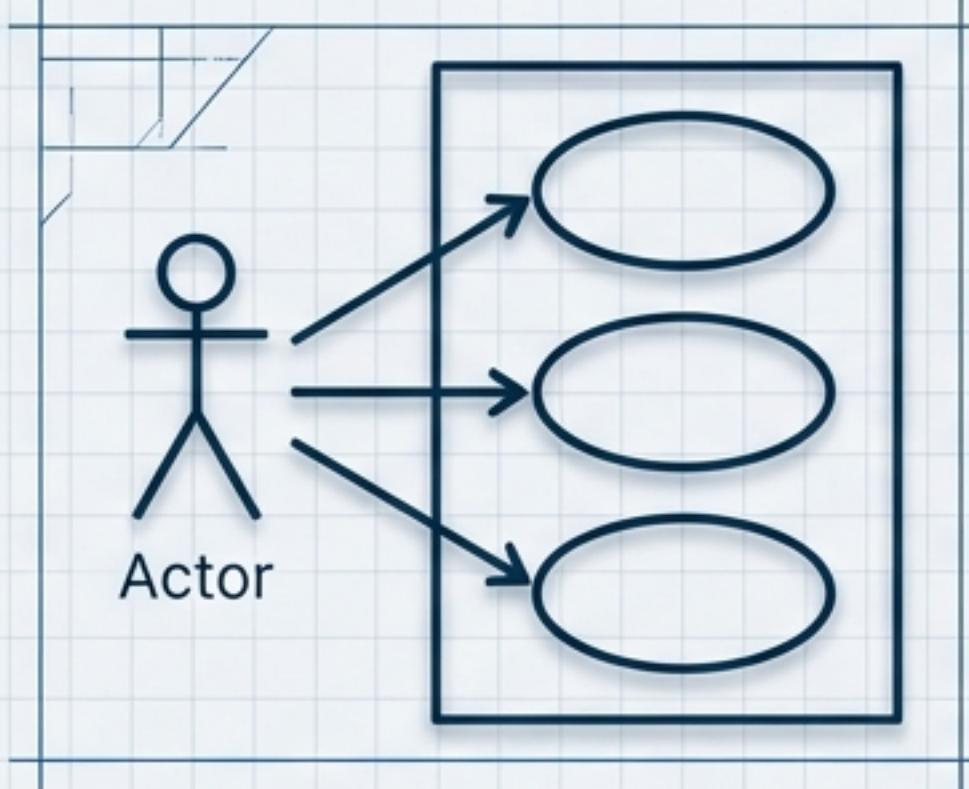
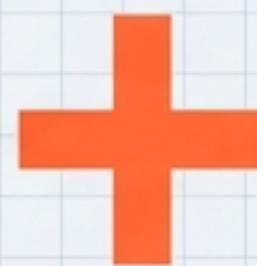
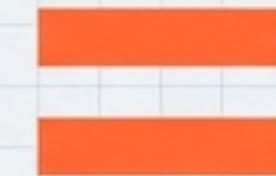


Diagrama UML
(Relaciones)



Documento Detallado
(Lógica)



Especificación
Robusta

- No existe solapamiento. Ambos ofrecen información diferente e indispensable para el desarrollador.
- Sin documentación escrita, la ambigüedad genera errores costosos.

Ficha Técnica de Caso de Uso

Campo	Detalle
ID	001
Caso	Registrar nuevos usuarios
Actores	Usuarios no registrados
Includes	Autenticación
Precondiciones	Usuario no registrado previamente
Postcondiciones	Usuario queda registrado en el sistema
Flujo Normal	Registro de formulario (correo y contraseña)

Requisitos Funcionales vs. No Funcionales



Requisitos Funcionales (El Qué)

- Funcionalidades concretas.
Lo que el sistema HACE.
- **Location:**
Se especifican en el Diagrama.
- **Example:**
Retirar efectivo.



Requisitos No Funcionales (El Cómo)

- Rendimiento y calidad.
Cómo se COMPORTA el sistema.
- **Location:**
Se especifican en la Documentación.
- **Example:**
Ejecución en < 500ms.

El Ciclo de Vida de una Acción: Pre y Post



Estas definiciones son críticas para que el departamento de QA (Calidad) diseñe las pruebas.

Mejores Prácticas y Resolución

- 1. Modularización:** Distinguir funcionalidades comunes para reutilizar código.

- 2. Acuerdo:** La documentación es un contrato que protege contra exigencias no pactadas.

- 3. Resolución:** Sin documento escrito, nos exponemos a riesgos comerciales.




Requisitos claros + Diagramas precisos + Documentación detallada = Desarrollo rentable.