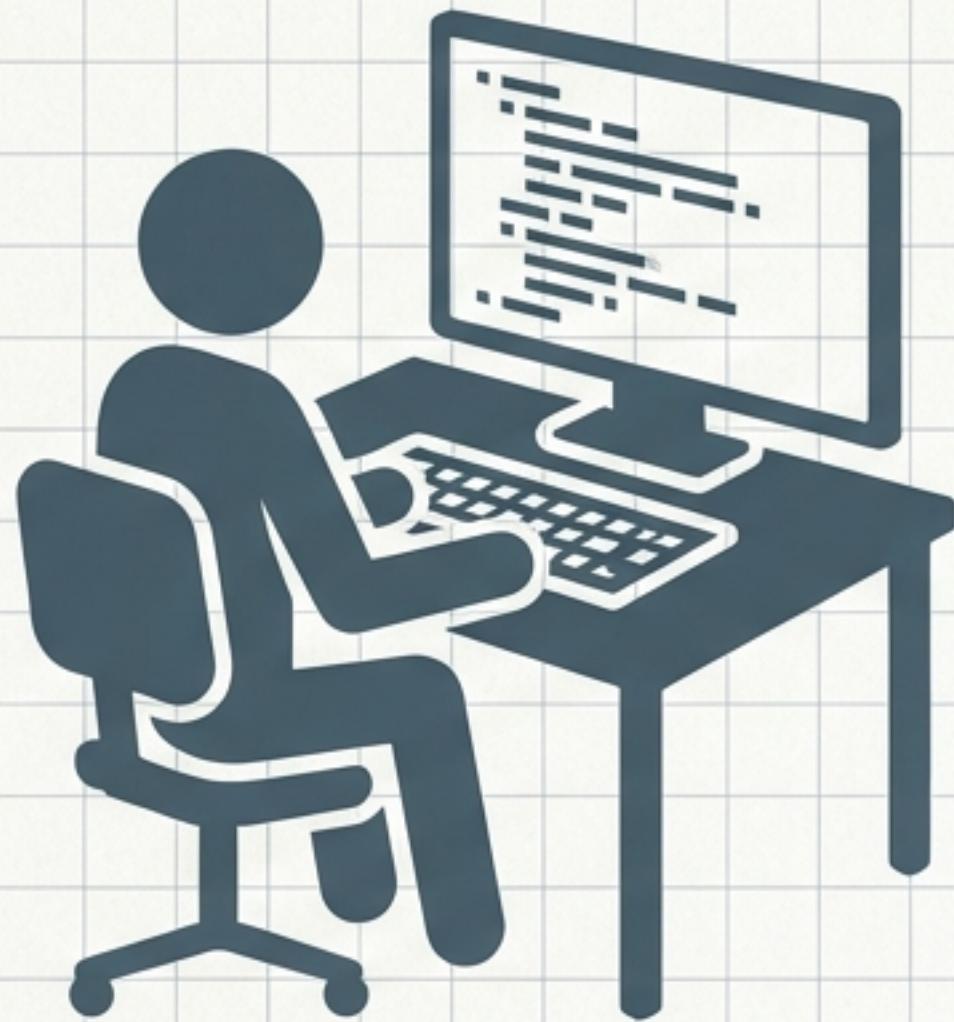


INGENIERÍA DEL SOFTWARE: LA ARQUITECTURA DETRÁS DEL CÓDIGO

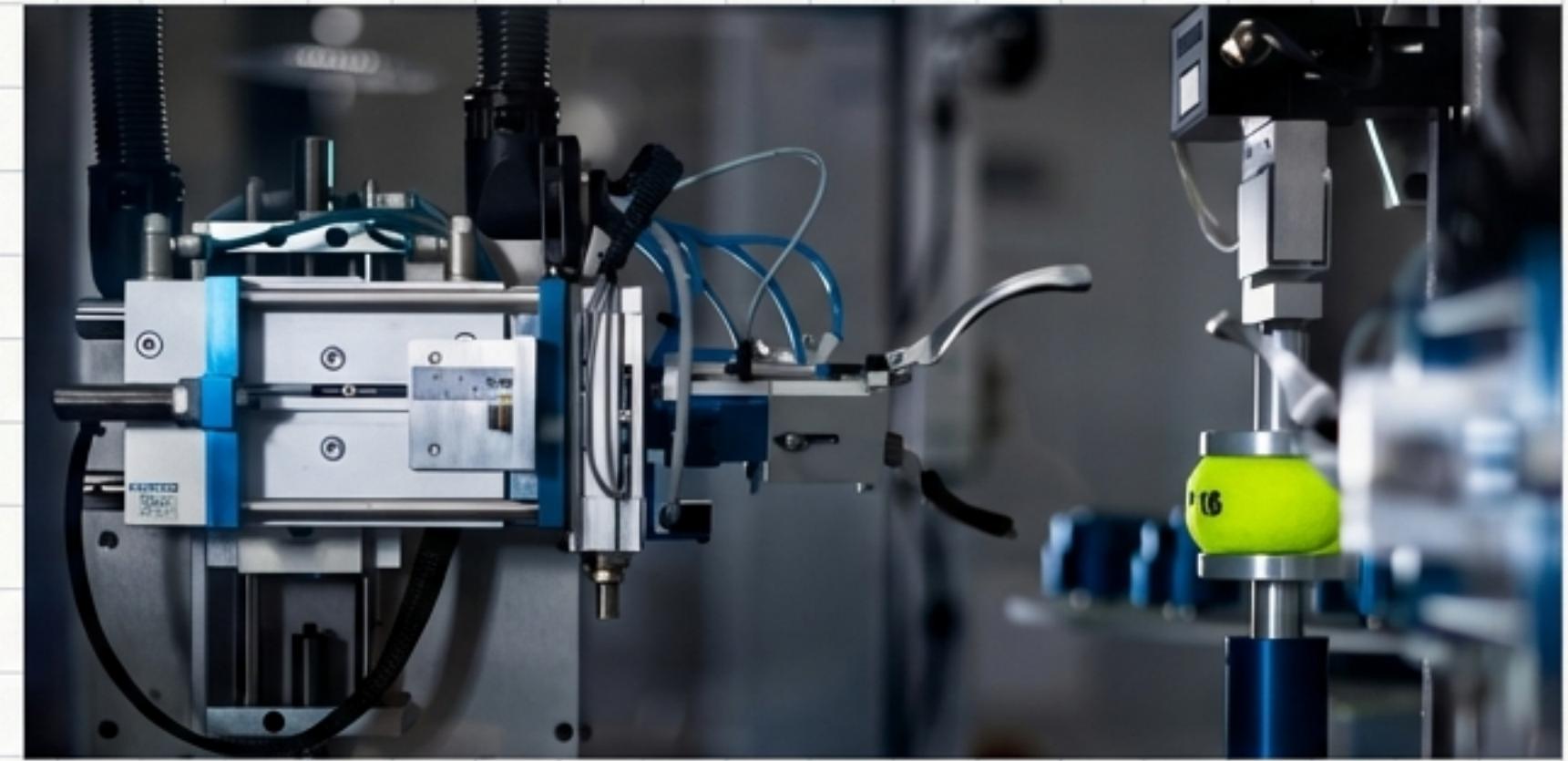
De la improvisación al proceso ingenieril

LA VISIÓN AMATEUR



El desarrollo se confunde con la simple escritura.
Sin un ciclo de vida, el proyecto fracasa.

LA VISIÓN INGENIERIL



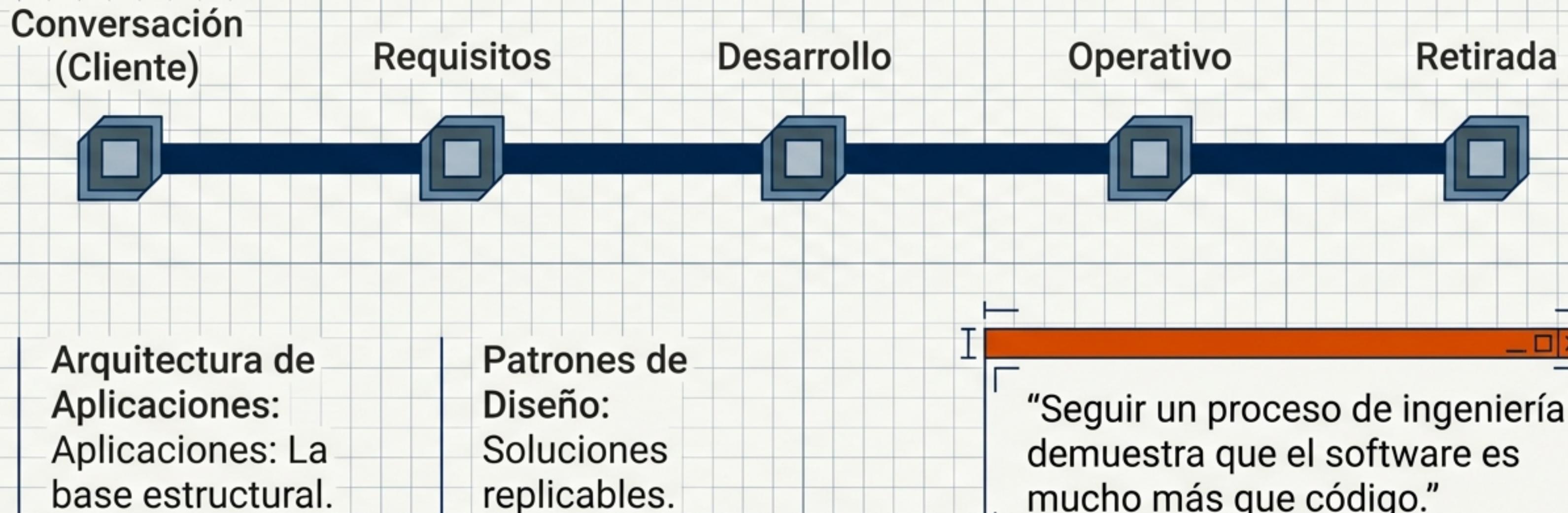
La Ilusión del "Solo Código"

Comparativa del Coste del Cambio:

1. Arquitectura Tradicional: Cambiar una viga de carga en un edificio terminado es casi imposible.
2. Software: Cambiar una línea es barato, pero encontrar el error en 10.000 líneas es costoso.

La diferencia fundamental es el coste de la búsqueda del error frente a su solución.

EL CICLO DE VIDA: UNA VISIÓN GLOBAL



CIMIENTOS: REQUISITOS Y LA TRAMPA DE LA PRISA



CASO PRÁCTICO 1: EL RIESGO REAL

El Problema: Saltar al diseño por falta de tiempo.

El Riesgo: Construir lo incorrecto. Un error aquí supone una gran pérdida económica.

Fase de Requisitos: Definición de funcionalidades mediante entrevistas.

Objetivo: Generar un Diagrama de Casos de Uso.

EL PLANO MAESTRO: ANÁLISIS Y DISEÑO



ANÁLISIS (El Qué)

- Creación de diagramas que especifican cada requisito.
- Estudio de dependencias y relaciones (Bases de Datos).

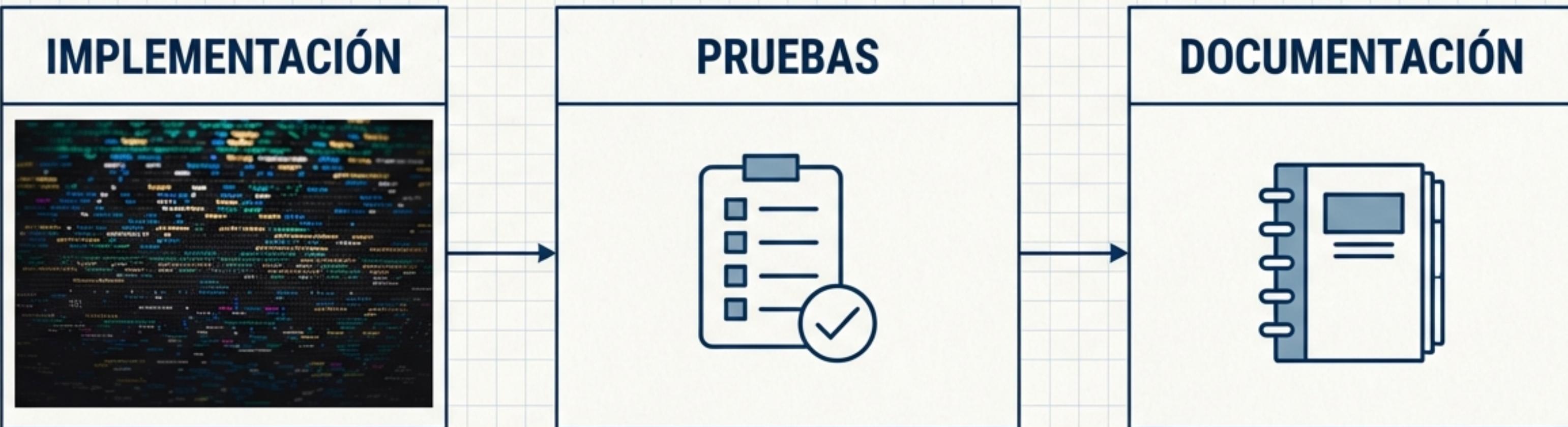


DISEÑO (El Cómo)

- Definición general de funcionalidades.
- Identificación de Recursos:
 - Físicos (Hardware)
 - Lógicos (Software/Licencias)

Disponer de un buen análisis permite un diseño seguro y la aprobación del cliente.

CONSTRUCCIÓN Y CONTROL DE CALIDAD



Codificación de lo diseñado.
La primera versión tangible.

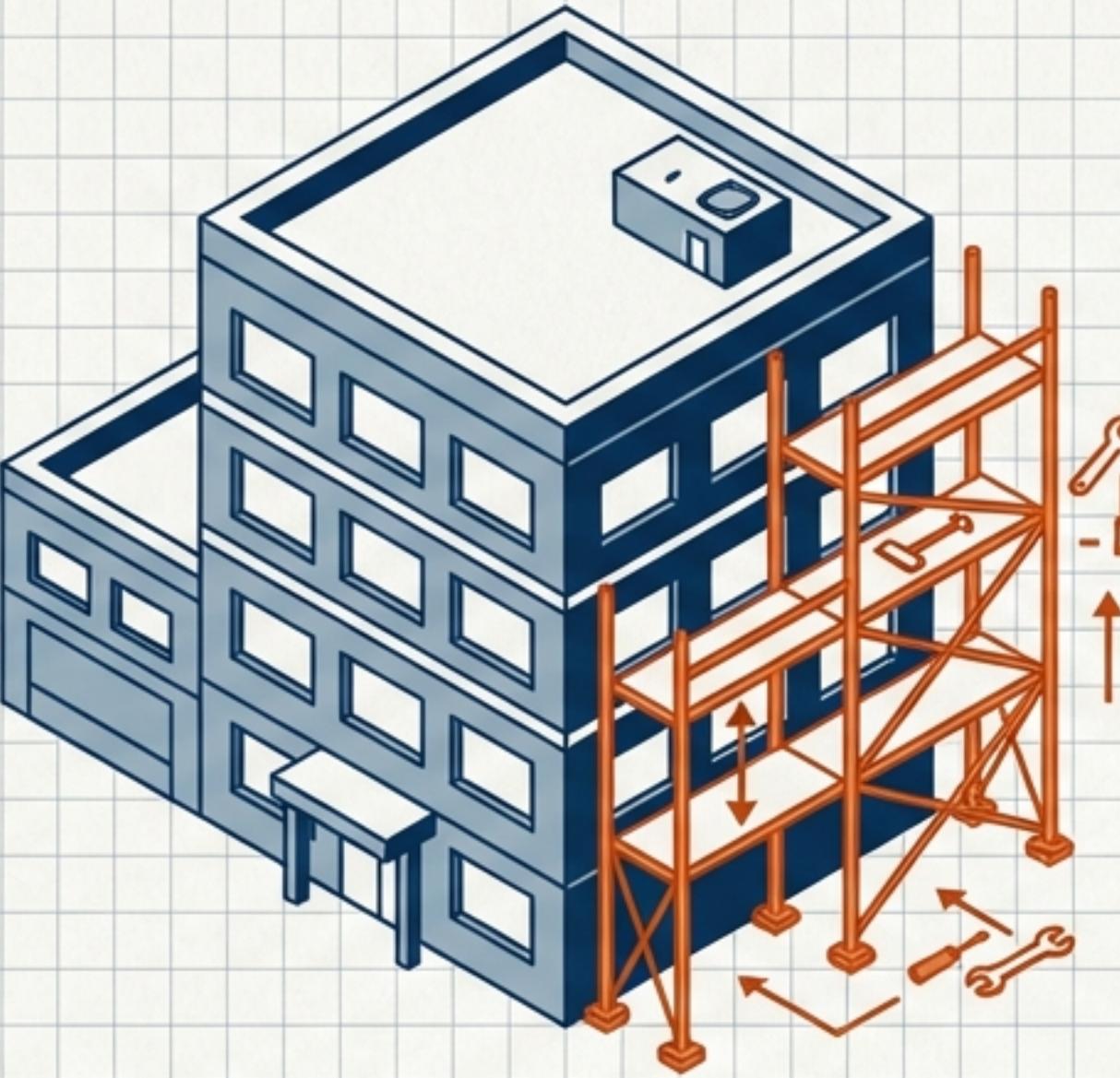
Identificación de errores
antes de la entrega.

El paso olvidado. Vital para
el futuro mantenimiento.

SOFTWARE VIVO: EXPLOTACIÓN Y MANTENIMIENTO

EXPLOTACIÓN

- Lanzamiento al entorno real con usuarios reales.



MANTENIMIENTO

- Corrección de errores lógicos y externos que surgen con el tiempo.

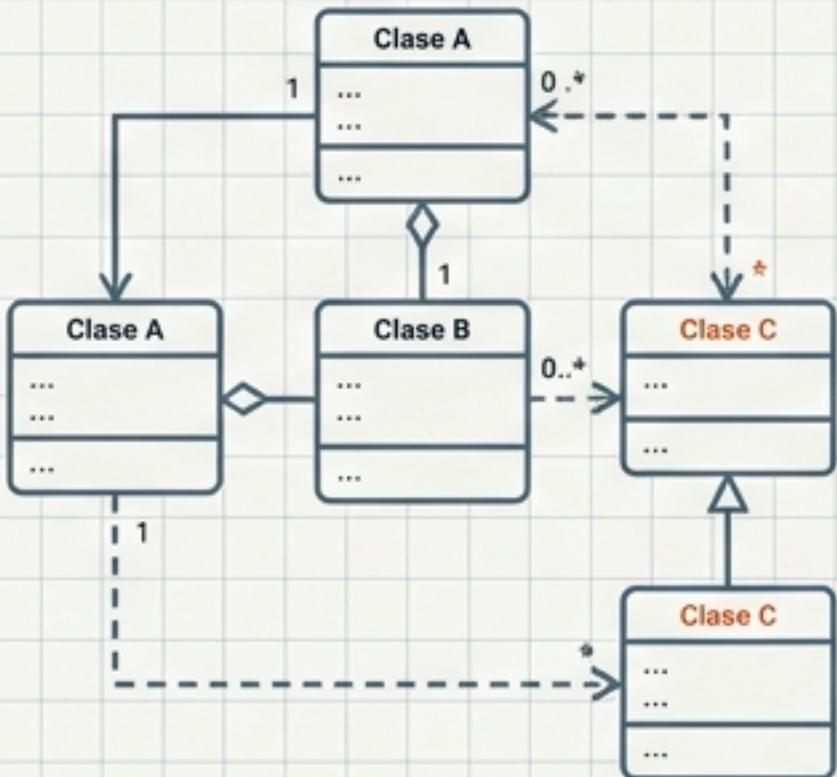
EL ESCENARIO DE LOS 10 AÑOS

Pregunta: ¿Cómo solucionar un error en producción tras 10 años sin tocar el código?

Respuesta: Sin proceso de ingeniería ni documentación previa, es prácticamente imposible.

EL LENGUAJE DE LA ARQUITECTURA

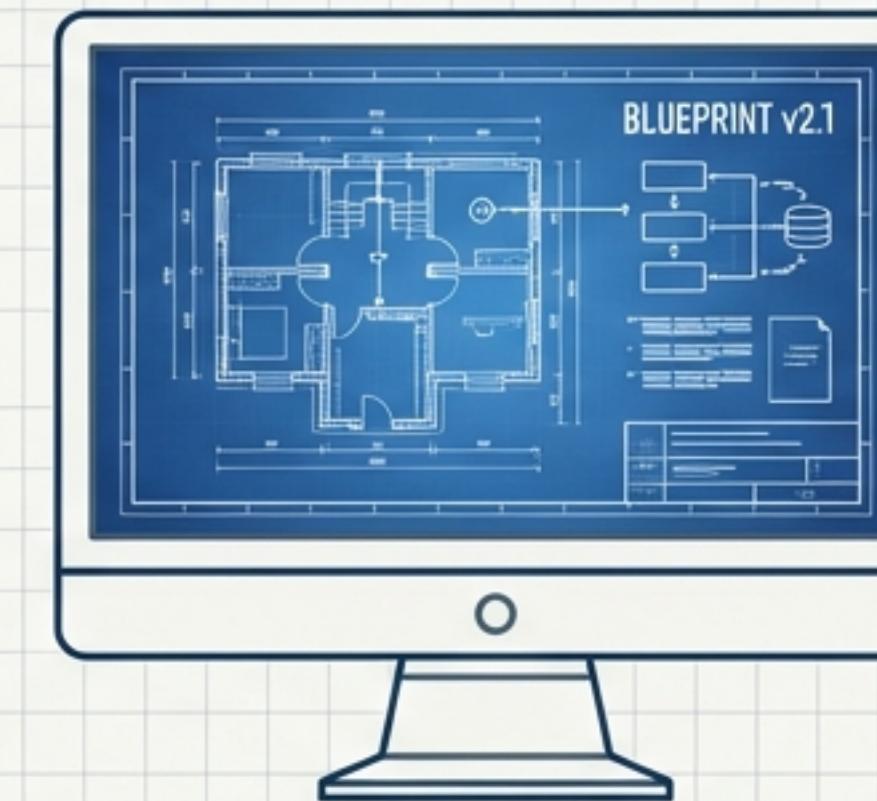
UML (Lenguaje de Modelado Unificado)



El estándar para la representación gráfica.

- Diagramas de casos de uso, secuencia, clases.

HERRAMIENTAS CASE



Computer Aided Software Engineering.

Software para digitalizar y gestionar los diagramas técnicos.

ELIGIENDO EL ENFOQUE: METODOLOGÍAS



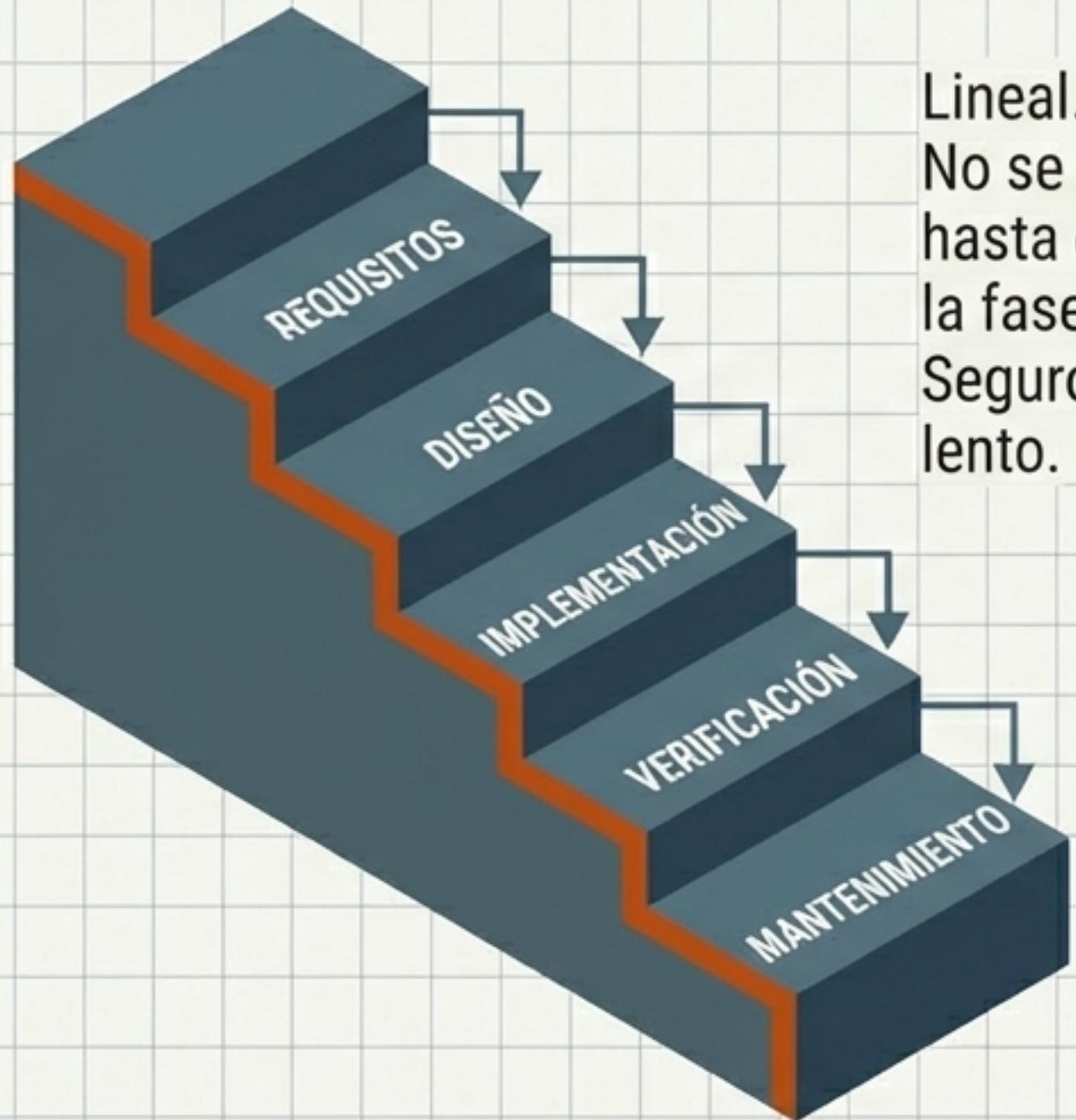
No existe una única forma de organizar las fases.
Una mala elección puede provocar grandes retrasos.

El Conflicto Central:

Estructura Rígida (Control) vs Velocidad de Adaptación (Agile)

MODELOS TRADICIONALES: ESTRUCTURA Y CONTROL

EN CASCADA (WATERFALL)



Lineal.
No se avanza hasta completar la fase anterior.
Seguro pero lento.

EN ESPIRAL



Ciclo continuo.
Producto funcional en menor tiempo, pero temporalización compleja.

LA VENTAJA DE LA VELOCIDAD: METODOLOGÍAS ÁGILES

Scrum & Extreme Programming

El Reto (Caso Práctico 2):

- Competidor usando Cascada = Lento.



Lleva el modelo en espiral al extremo.

La Estrategia:

- Utilizar Sprints cortos.
- Resultado: Versión funcional cada dos semanas. Captura del mercado.

EL EQUIPO DE CONSTRUCCIÓN: VISIONARIOS



ANALISTA DE SISTEMA

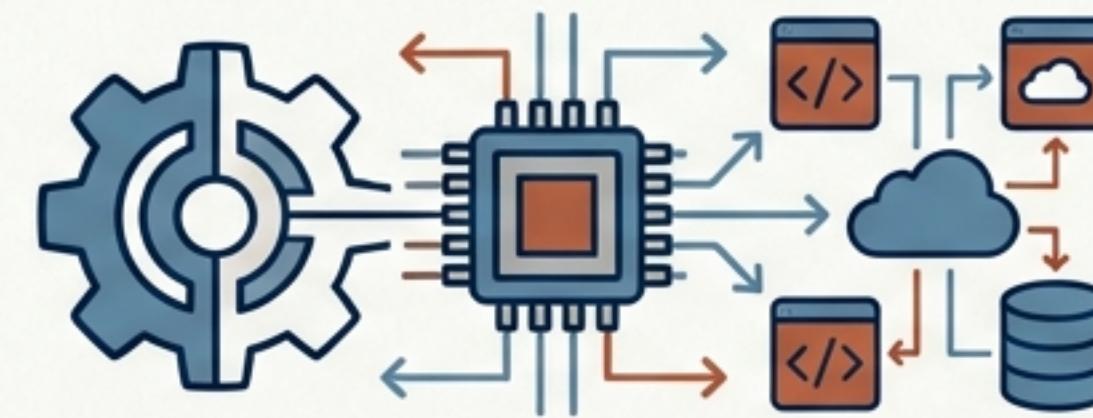


El puente con el cliente. Realiza el estudio del sistema y garantiza las expectativas.

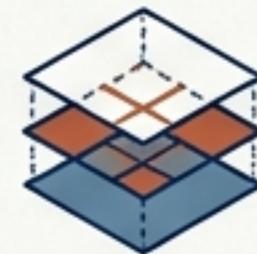


FASE: ANÁLISIS

ARQUITECTO DE SOFTWARE

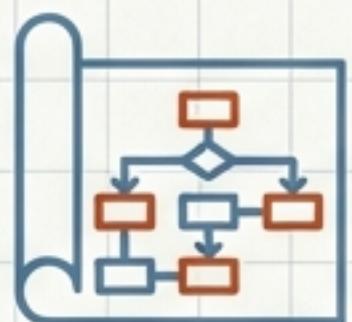


El perfil híbrido. Une el desarrollo con el conocimiento de frameworks y tecnologías.



FASE: TRANSVERSAL

EL EQUIPO DE CONSTRUCCIÓN: EJECUTORES



DISEÑADOR DE SOFTWARE

Diseña el sistema a implementar.



ANALISTA PROGRAMADOR

Visión de detalle. Trabaja en diseño, implementación y pruebas.



PROGRAMADOR

Codifica el estudio realizado por analistas y diseñadores.

EL COSTE REAL DE 'AHORRAR'

Resolución del Caso:

- Intentar ahorrar reduciendo roles o fases dispara el coste futuro.
- Ejemplo: Si no se define qué hace la consola cuando el usuario no decide, el código fallará.

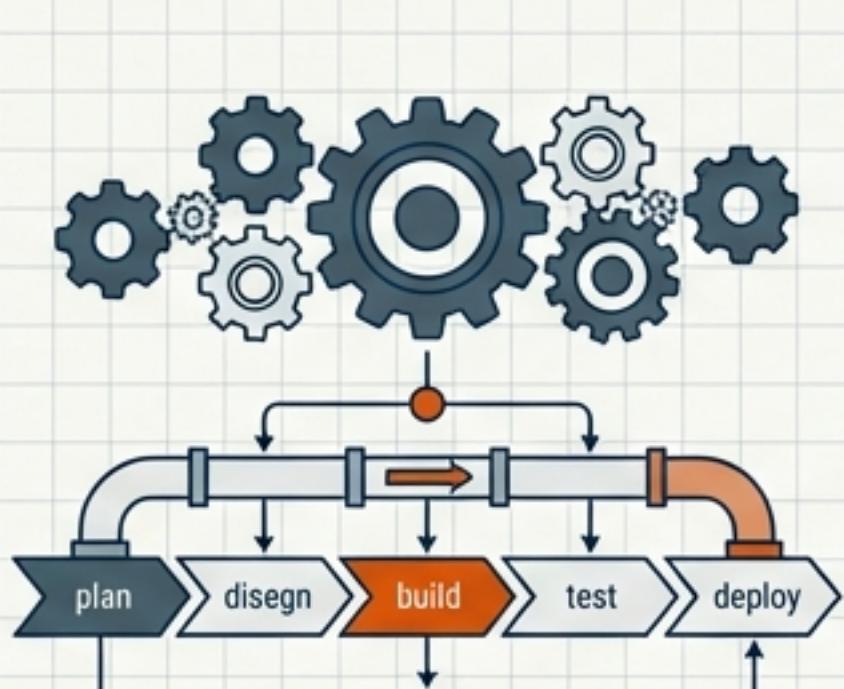


El ahorro en análisis se paga doble en mantenimiento.

CONCLUSIÓN: EL ENFOQUE GLOBAL

1. INGENIERÍA

Proceso industrial, no artesanal.



2. METODOLOGÍA

Cascada (Seguridad) vs Agile (Velocidad).



3. EQUIPO

Roles definidos para que las piezas encajen.



La búsqueda de soluciones requiere un enfoque global. Solo así el software sobrevive al paso del tiempo.