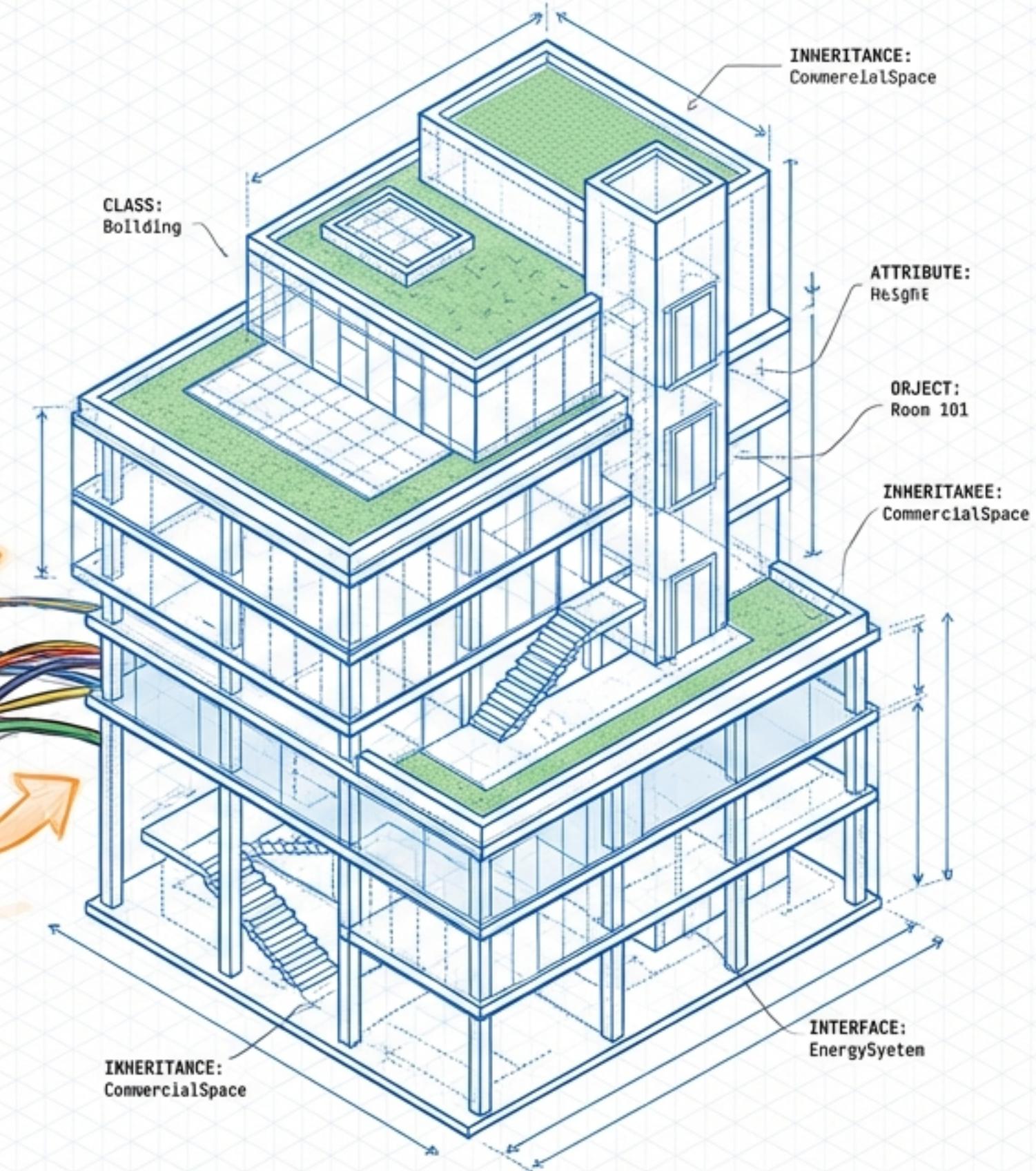
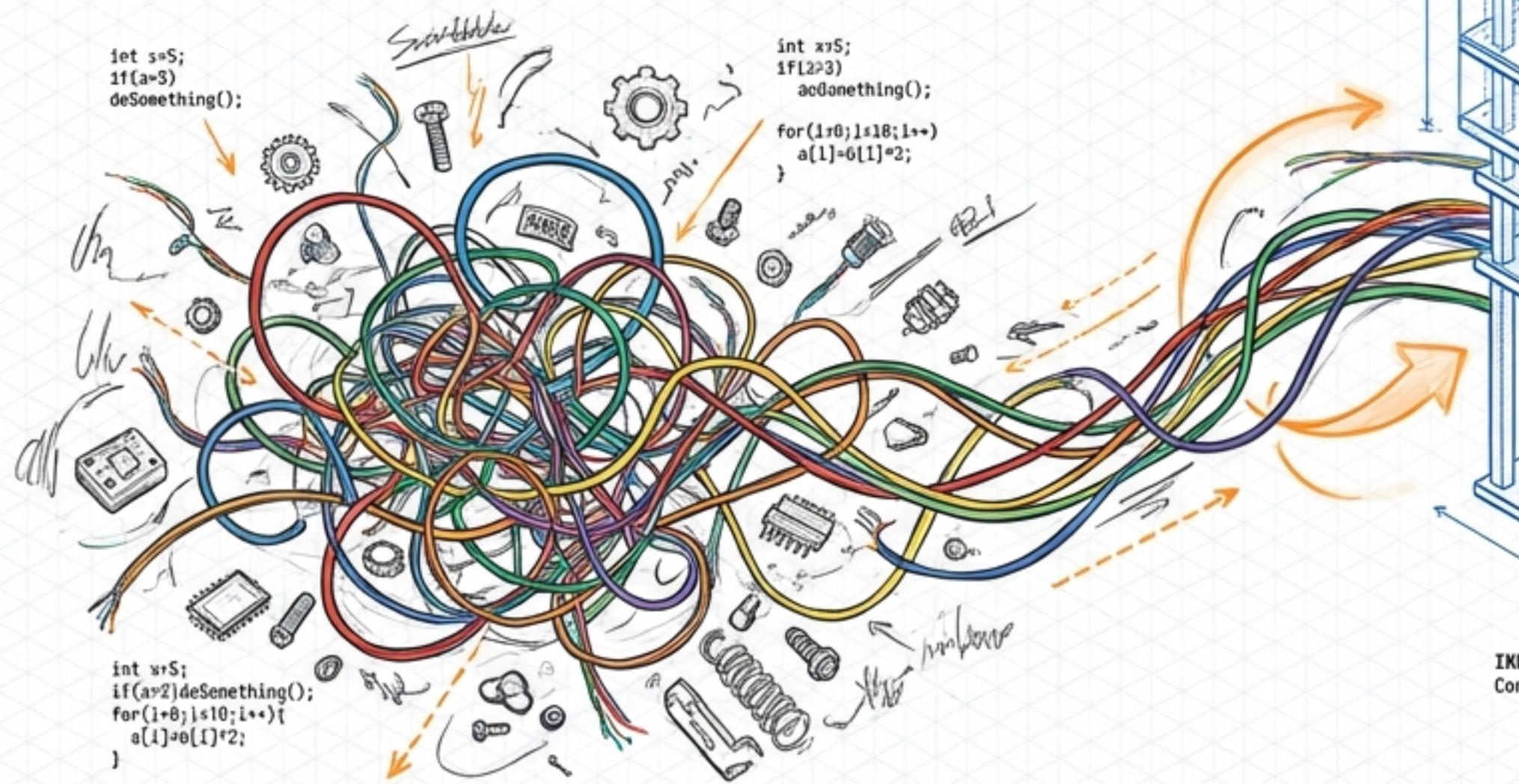


INTRODUCCIÓN A LA ORIENTACIÓN A OBJETOS

DEL CÓDIGO ESTRUCTURADO
AL MODELADO DEL MUNDO REAL



PROGRAMACIÓN: TEMA 7

Cuando el Código Estructurado "Se Queda Pequeño"

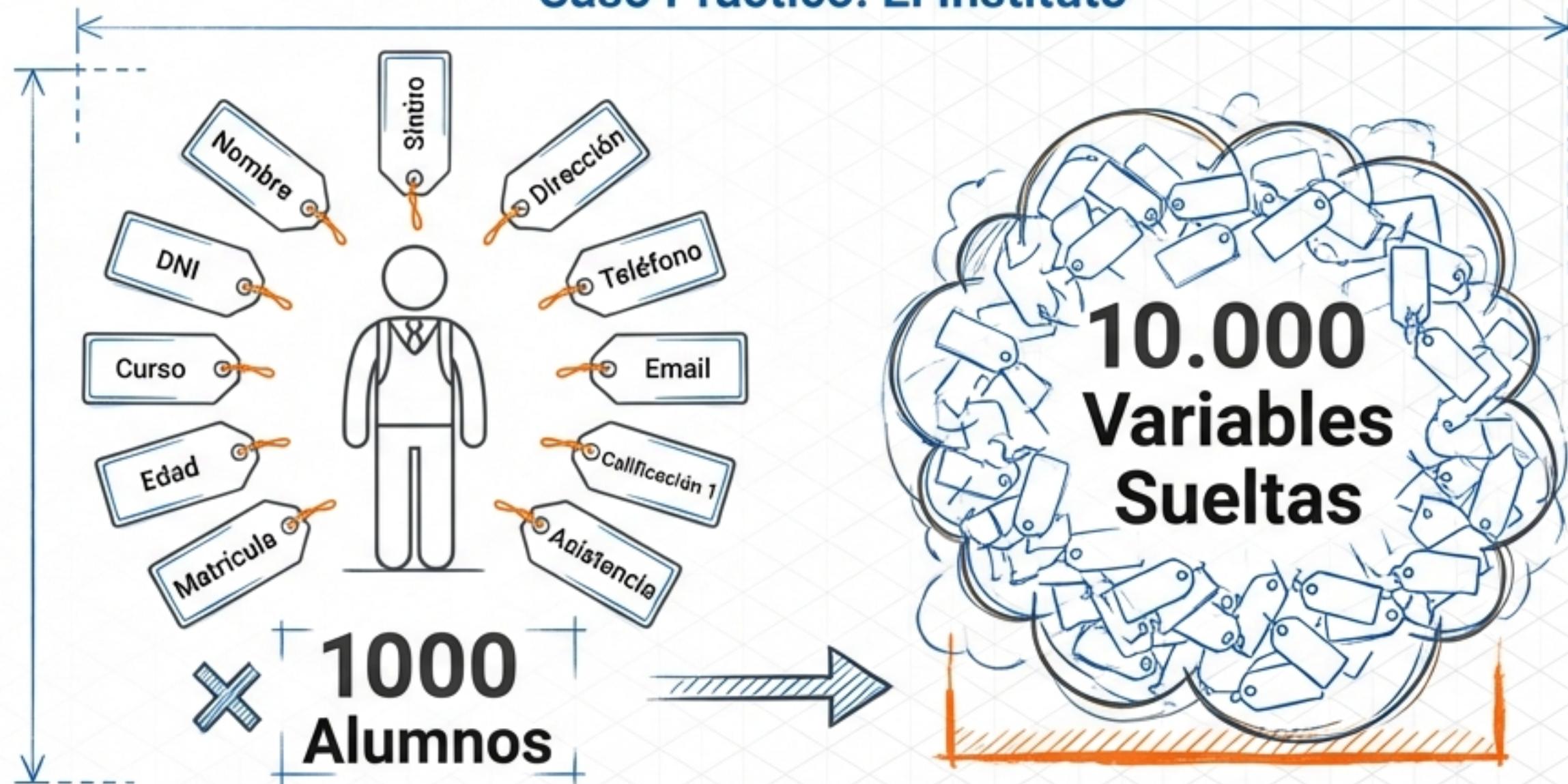
El Viejo Paradigma

Secuencias, condicionales y bucles funcionan para tareas simples.

El Problema

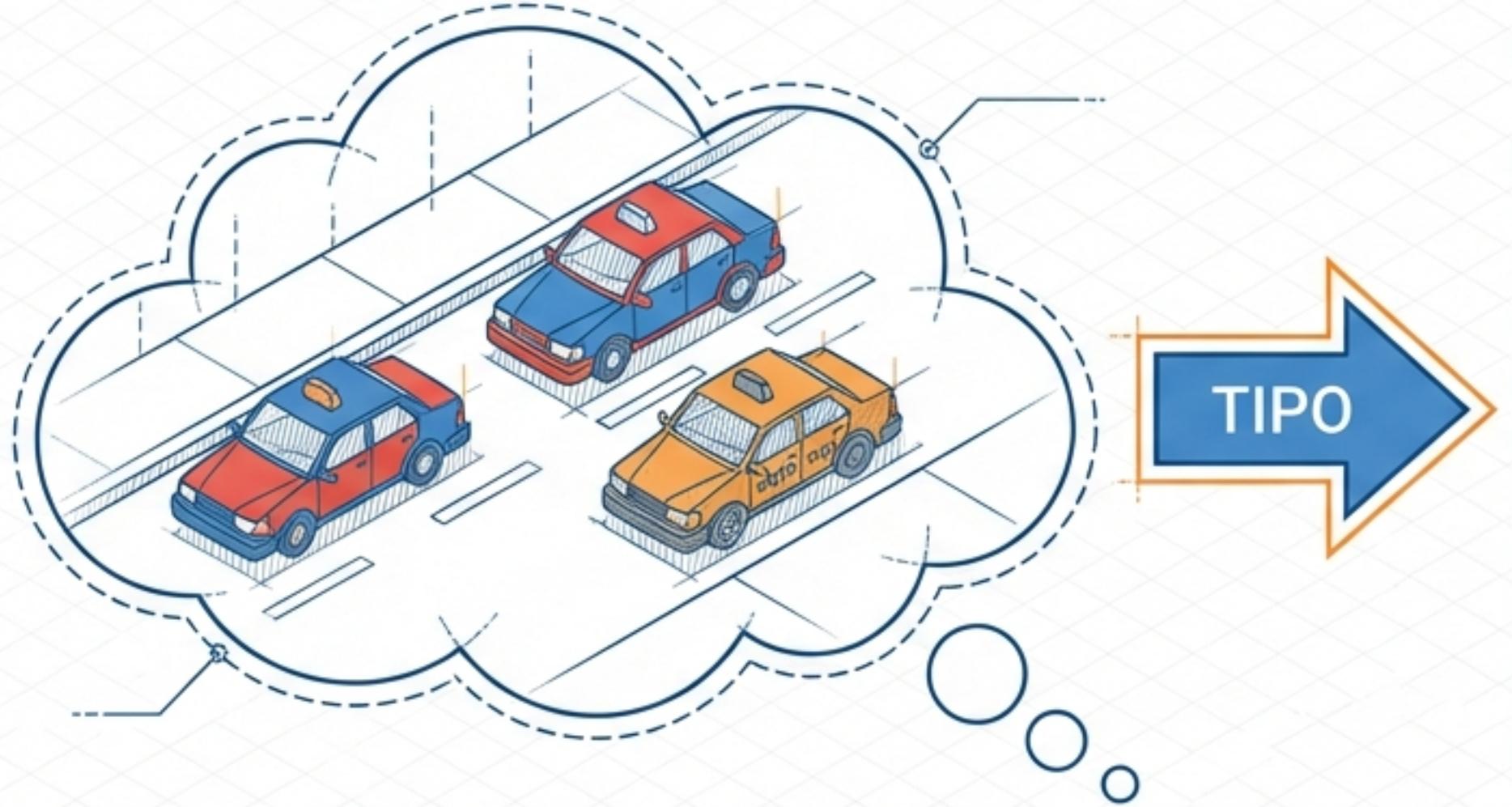
A medida que crece el código, los datos y las operaciones se separan. **El mantenimiento se vuelve imposible.**

Caso Práctico: El Instituto



La complejidad requiere una nueva forma de pensar: **no procesar datos, sino modelar objetos.**

POO: Un Nuevo Paradigma



```
Clase Taxi  
public class Taxi {  
  
    String ciudad; //Ciudad de cada objeto taxi  
    String matricula; //Matrícula de cada objeto taxi  
    String distrito; //Distrito asignado a cada objeto taxi  
    int tipoMotor; //tipo de motor  
  
    //Constructor: cuando se cree un objeto taxi se ejecutará  
    //el código que incluyamos en el constructor  
    public Taxi () {  
        ciudad = "Méjico D.F.";  
        matricula = "";  
        distrito = "Desconocido";  
        tipoMotor = 0;  
    } //Cierre del constructor  
  
    int valorTipoMotor() {  
        ...  
    }  
}
```

Antes:

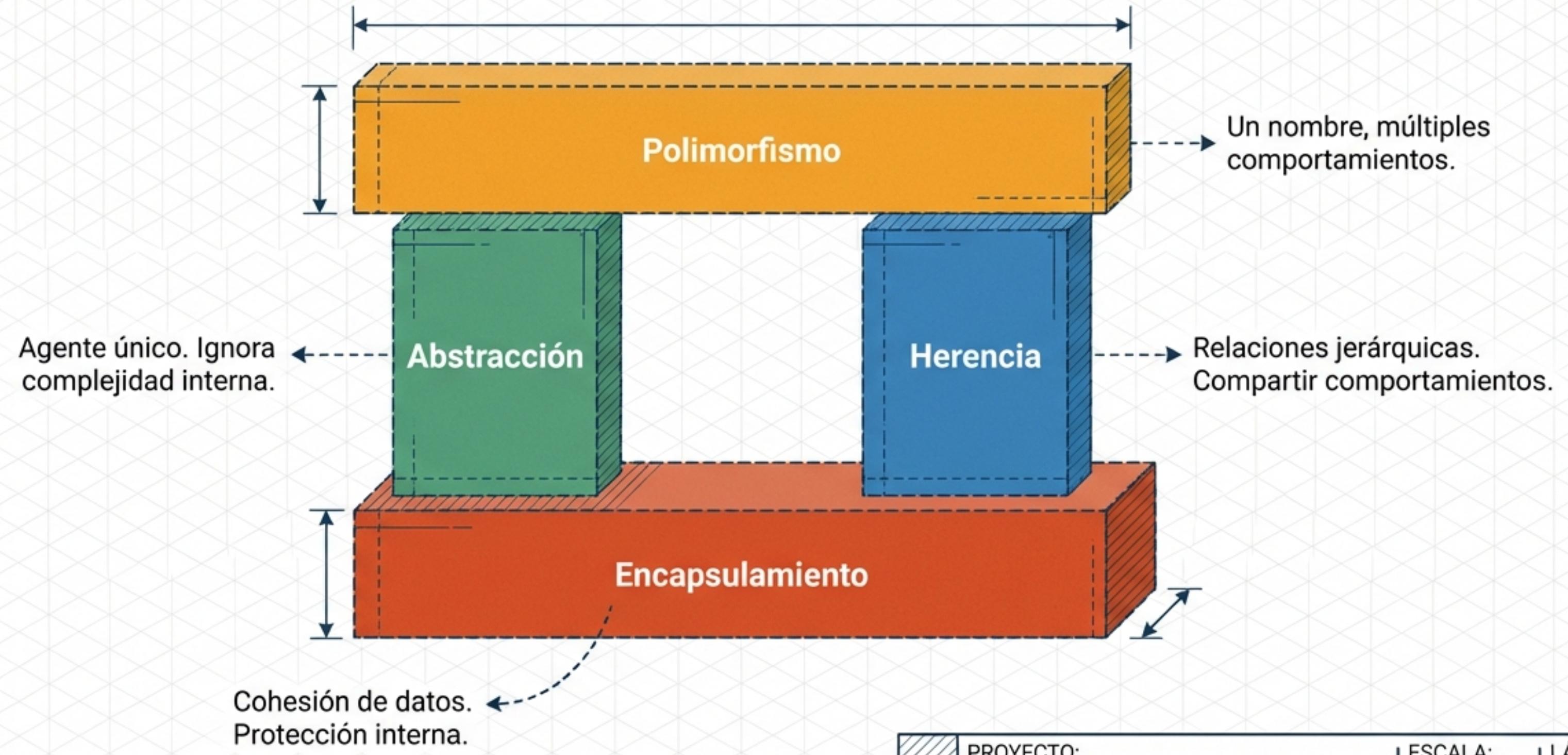
Variables (Datos) + Funciones (Instrucciones)
por separado.

Ahora: Objetos.

Unidades que combinan datos y las operaciones
que pueden realizar sobre ellos.

"La ejecución del sistema es un proceso de comunicación mediante objetos."

Los 4 Pilares Fundamentales



PROYECTO:

FUNDAMENTOS DE POO

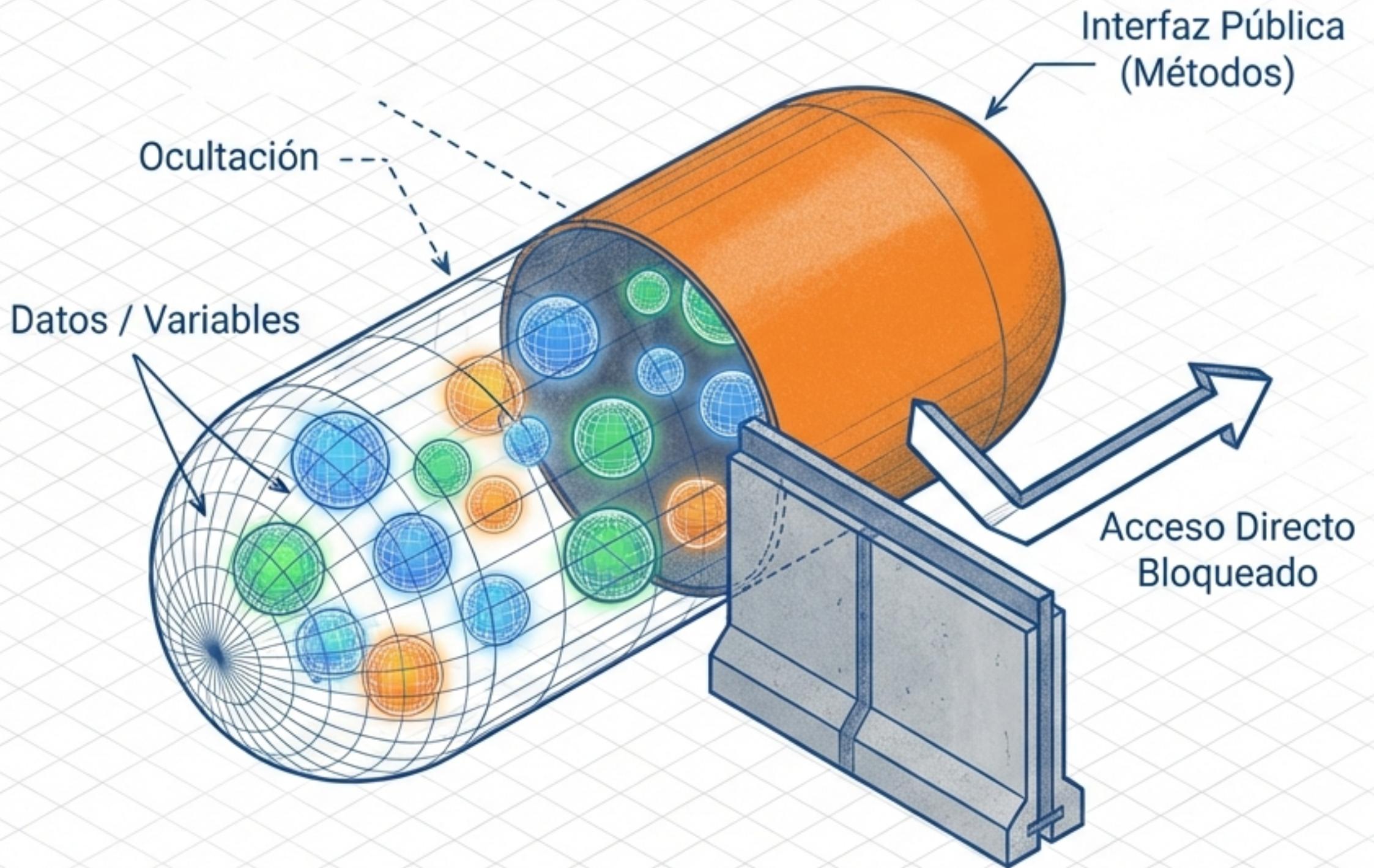
ESCALA:

N/A

LÁMINA:

03

Protección de Datos: El Concepto de "Caja Negra"



Ocultación: Los datos internos no son visibles desde fuera. Se accede a ellos solo mediante métodos controlados.

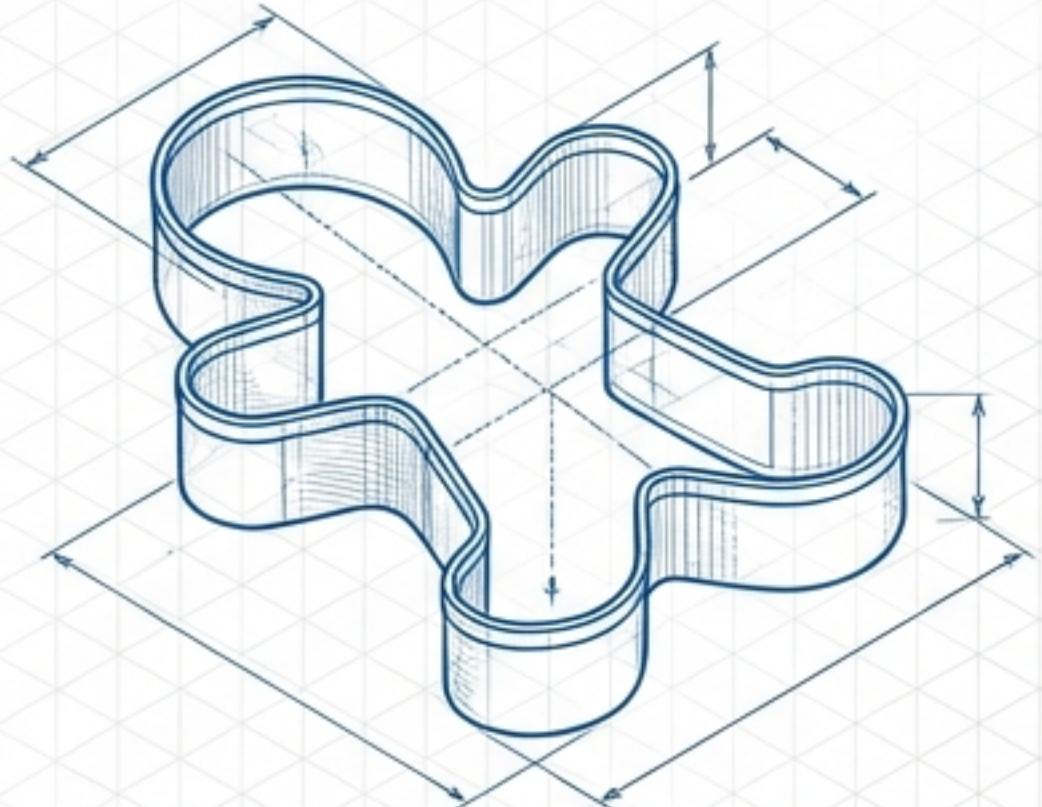
Interfaz Pública: Lo único que ve el exterior. Sabemos qué hace el objeto, no cómo lo hace.

Beneficio: Evita la corrupción de datos y facilita el mantenimiento.

PROYECTO:	FUNDAMENTOS DE POO	
ESCALA:	N/A	LÁMINA: 04

El Molde y La Galleta

Distinguiendo Clase vs. Objeto



La Clase (El Molde)

Concepto abstracto.

La plantilla (Ej: 'Persona').

El Objeto (La Galleta)

Instancia concreta con valores específicos
(Ej: 'José', 'Francisco').

Test de Identificación de Clases

- ¿Existe en el mundo real?
- ¿Tiene propiedades?
- ¿Realiza acciones?

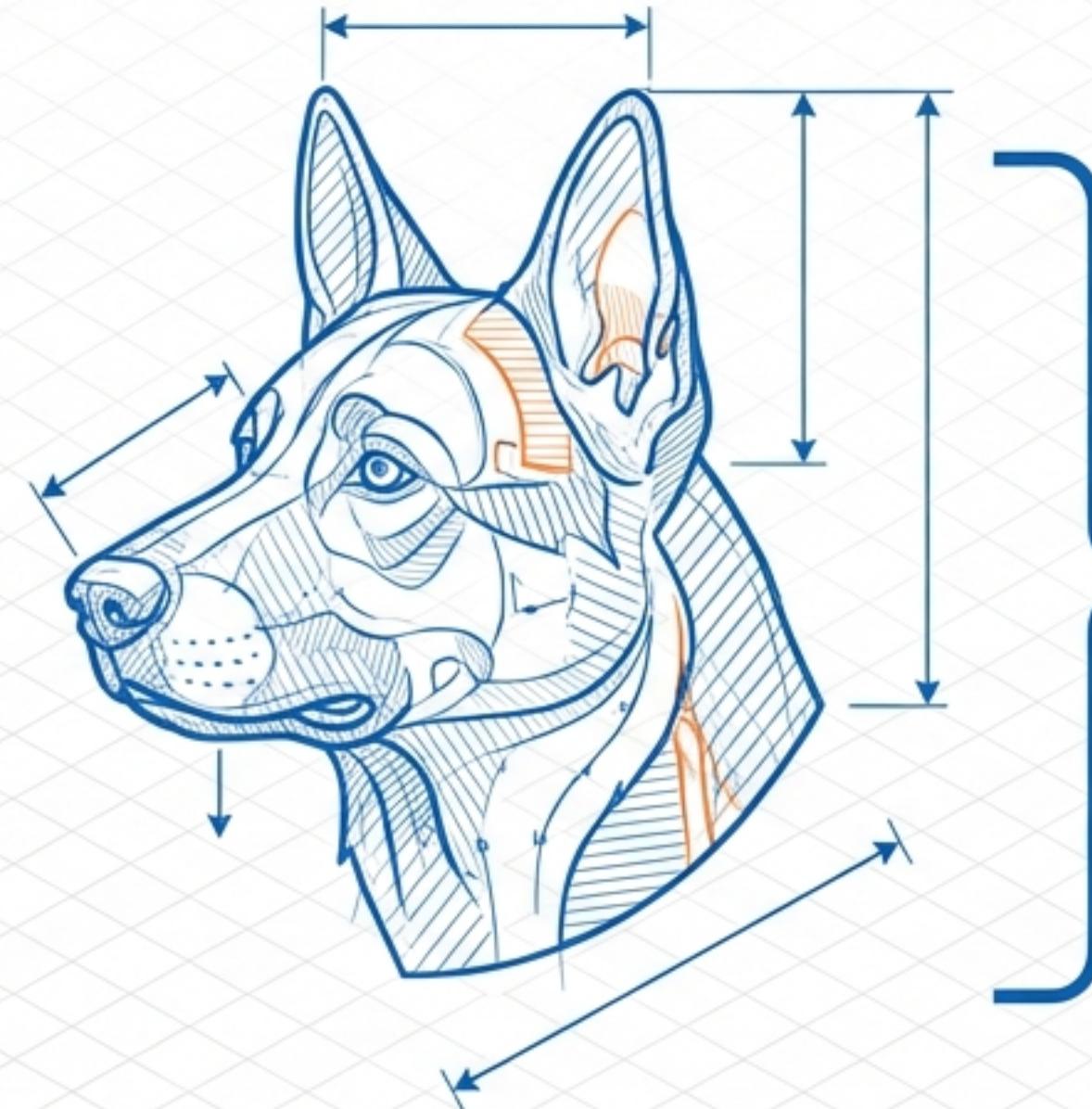
→ Si es Sí, es candidato a ser una Clase.

Anatomía de una Clase

Atributos (Datos)

- + nombre
- + color
- + raza
- + altura

Lo que el objeto ES o TIENE.



Métodos (Comportamiento)

- + ladrar
- + comer
- + dormir
- + correr

Lo que el objeto HACE.

Tipos de Métodos

Observadores (get), Modificadores (set), Constructores (inicio), Destructores (fin).

PROYECTO:	FUNDAMENTOS DE POO	
ESCALA:	N/A	
LÁMINA:	06	

Del Concepto al Código

```
1 public class Perro {  
2     final String nombre = "Nerón";  
3     String color;  
4     String raza;  
5     double altura;  
6 }
```

Constante
(Valor inmutable)

Declaración de Clase
(Mayúscula inicial)

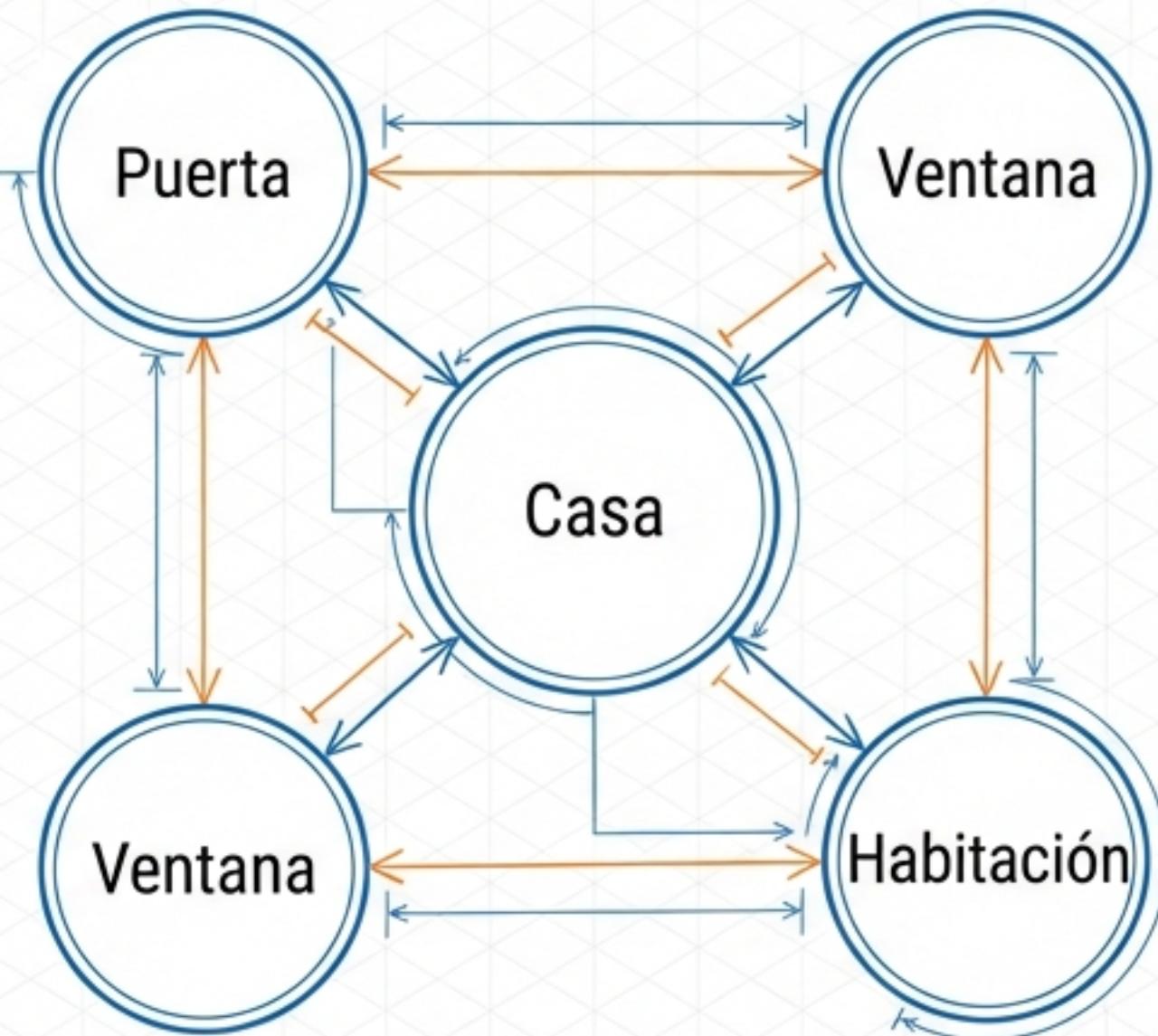
Atributo y Tipo de Dato

Nota: Java gestiona la memoria automáticamente (Garbage Collection).

Ningún Objeto es una Isla

El Ecosistema de Clases

Las clases viven interconectadas. Un sistema robusto es una red de cooperaciones.



Razones para Relacionar

- Modelado de la Realidad:** Una casa tiene puertas; un coche tiene ruedas.
- Reutilización:** No reinventar la rueda, usar la clase Rueda.
- Mantenimiento:** Separación de responsabilidades.

Tipos: Asociación | Dependencia | Generalización

PROYECTO:	FUNDAMENTOS DE POO
ESCALA:	N/A

Asociación y Dependencia

Asociación (“Tiene-un”)



Relación Estructural. El objeto “posee” al otro.

Ejemplo: Coche tiene Ruedas.

Multiplicidad: 1..1 (Uno a uno), 1..* (Uno a muchos).

Dependencia (“Usa-a”)



Relación Temporal. Un objeto necesita a otro momentáneamente.

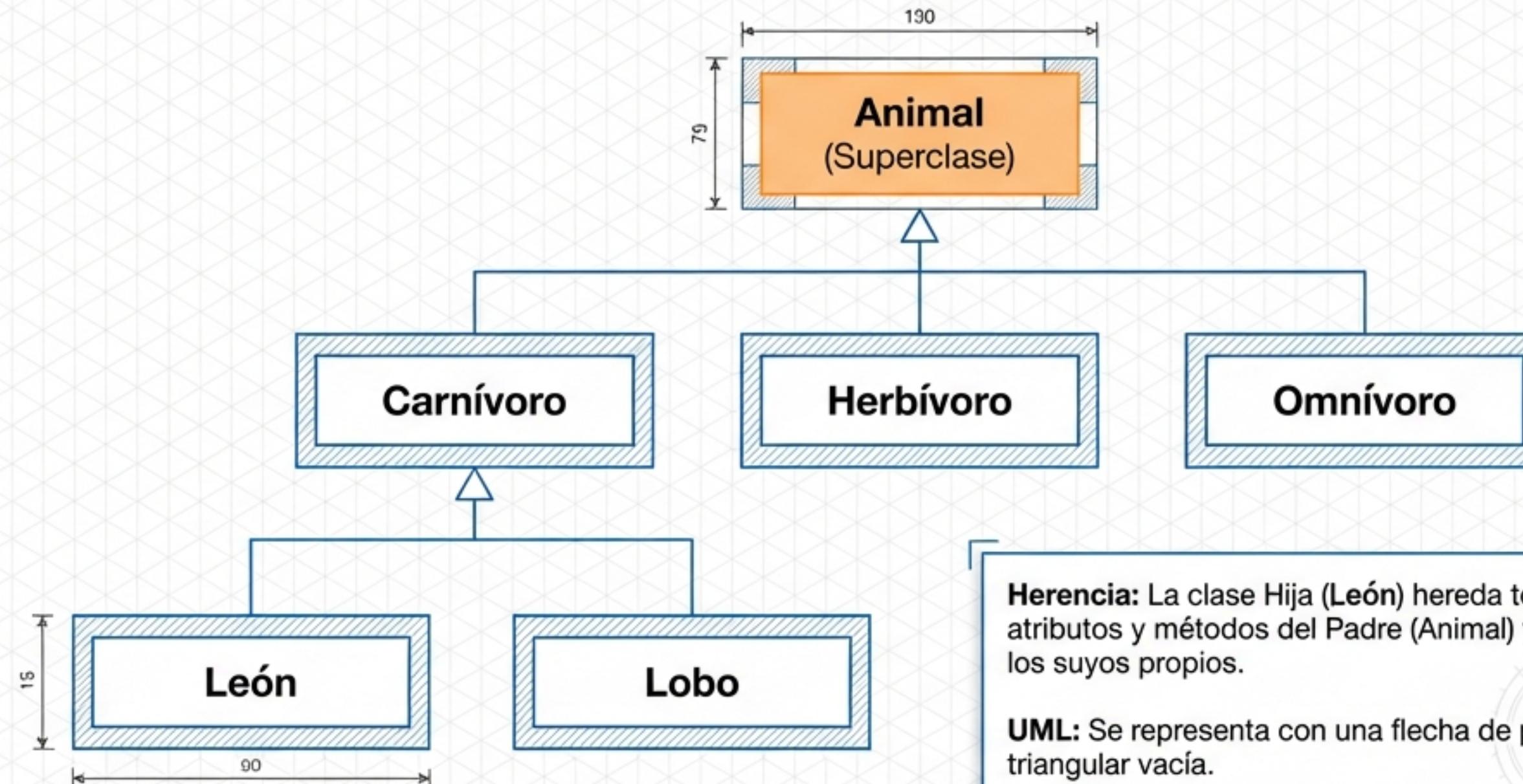
Ejemplo: Profesor usa Tiza.

La relación termina cuando finaliza la tarea.

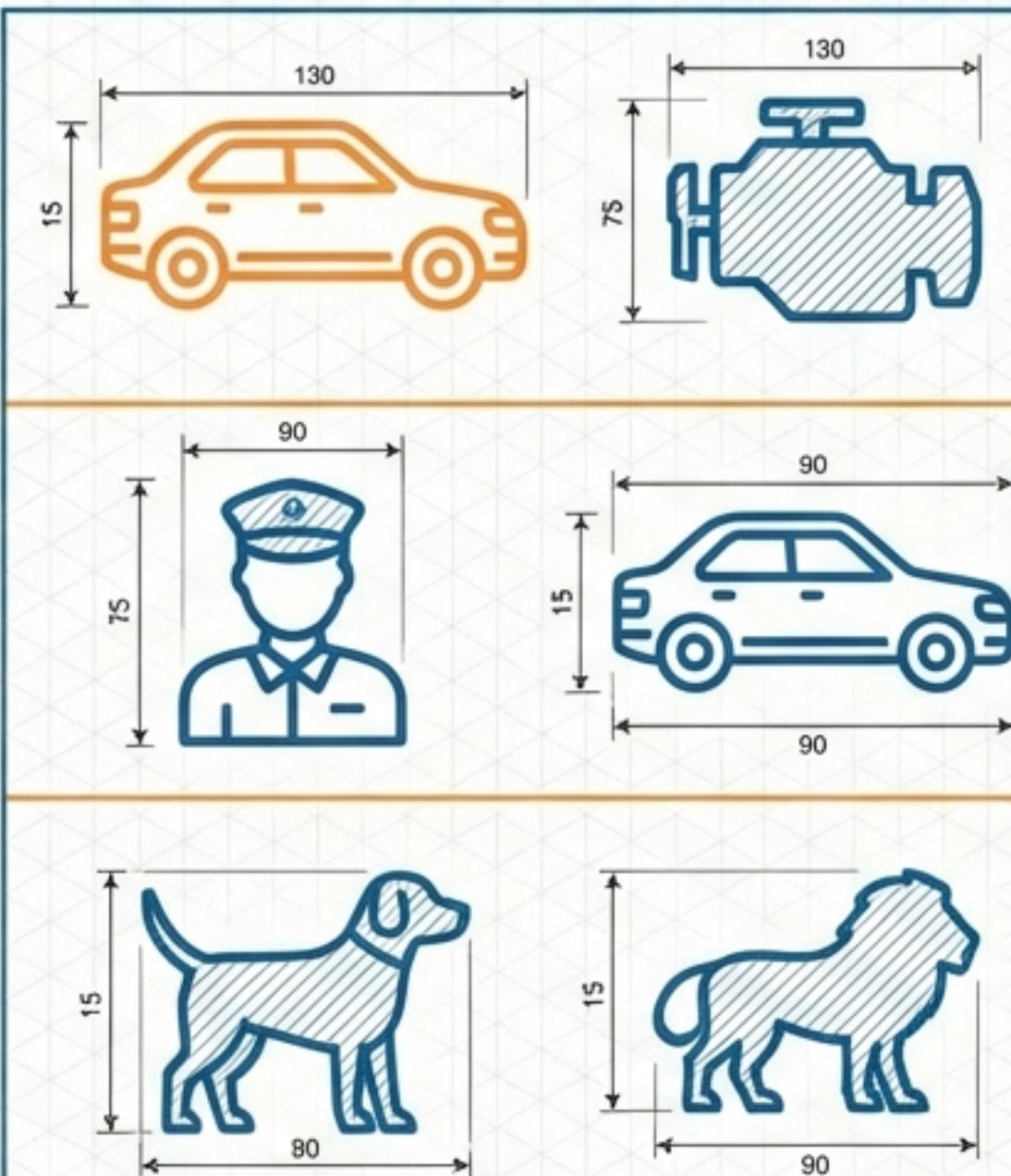
PROYECTO:	FUNDAMENTOS DE POO	
ESCALA:	N/A	LÁMINA: 09

Generalización (Herencia)

La Relación “Es-un”



Caso Práctico: Identificando Vínculos



Dependencia

El coche *necesita* usar el motor para funcionar.

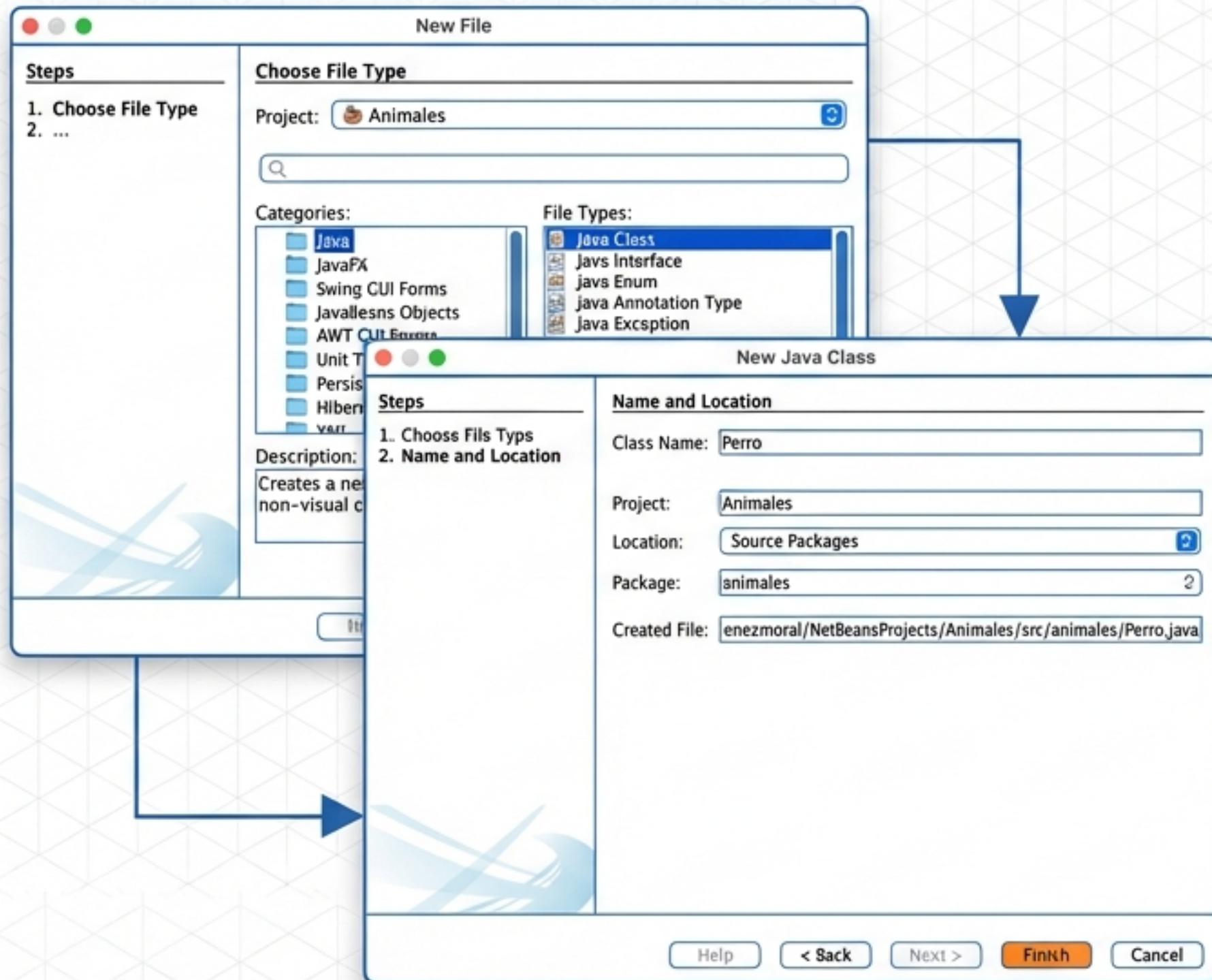
Asociación

El vehículo *tiene* un conductor asignado.

Generalización

El perro *es un* tipo de animal (Herencia).

Creación en el Entorno de Desarrollo



- 1. File -> New File:** Seleccionar proyecto y tipo 'Java Class'.
- 2. Nombrado:** El nombre debe coincidir exactamente (Ej: Perro.java).
- 3. Resultado:** Un fichero independiente listo para definir atributos.

Nota: El IDE gestiona automáticamente la estructura de carpetas y el nombre del archivo.

PROYECTO: FUNDAMENTOS DE POO

	ESCALA: N/A	LÁMINA: 12
--	----------------	---------------

Documentación Profesional con Doxygen

```
/**  
 * Esta clase representa un coche  
 * @version 1.0  
 * @author Francisco Jiménez Moral  
 */  
public class Coche {
```

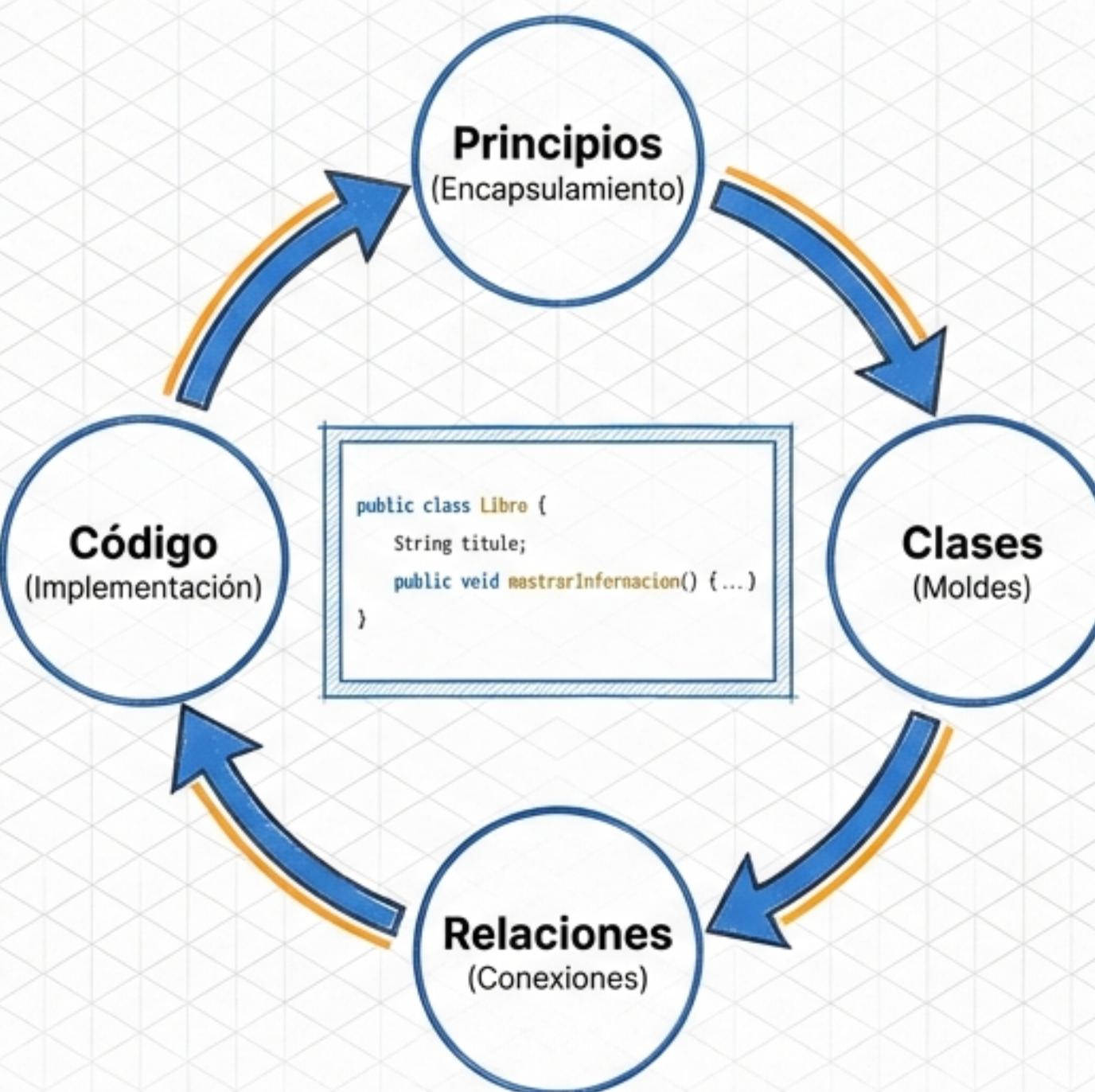
Control de versiones

Autoría del código

¿Por qué documentar?

El código dice qué hace la máquina; la documentación explica qué significa para el humano. Facilita colaboración y genera manuales automáticos.

Resumen: El Poder de los Objetos



Hemos pasado de variables sueltas a **Sistemas Robustos**.
La POO no es solo sintaxis; es la arquitectura de soluciones complejas.