Programación Orientada a Objetos

En Java

Conceptos Clave en POO

1. Abstracción
   - Aislar un elemento de su contexto o el resto de los elementos que los acompañan. Se piensa en el ‘que’ y no en el ‘como’

2. Clasificación
   - Identificar el origen de un elemento

Conceptos Clave en POO

3. Interfaces utilizables
   - Dado que nosotros aislaremos los distintos elementos, deberemos proveer una forma para que los objetos interactúen entre ellos

4. Control de Acceso
   - Limitar el acceso que otros objetos tienen a uno para garantizar su buen funcionamiento

Java y POO

- Java nos provee de distintas herramientas para aplicar los conceptos claves anteriores
  - Control de acceso
  - Encapsulamiento
  - Herencia
  - Polimorfismo
  - Interfaces

¿Qué es un objeto?

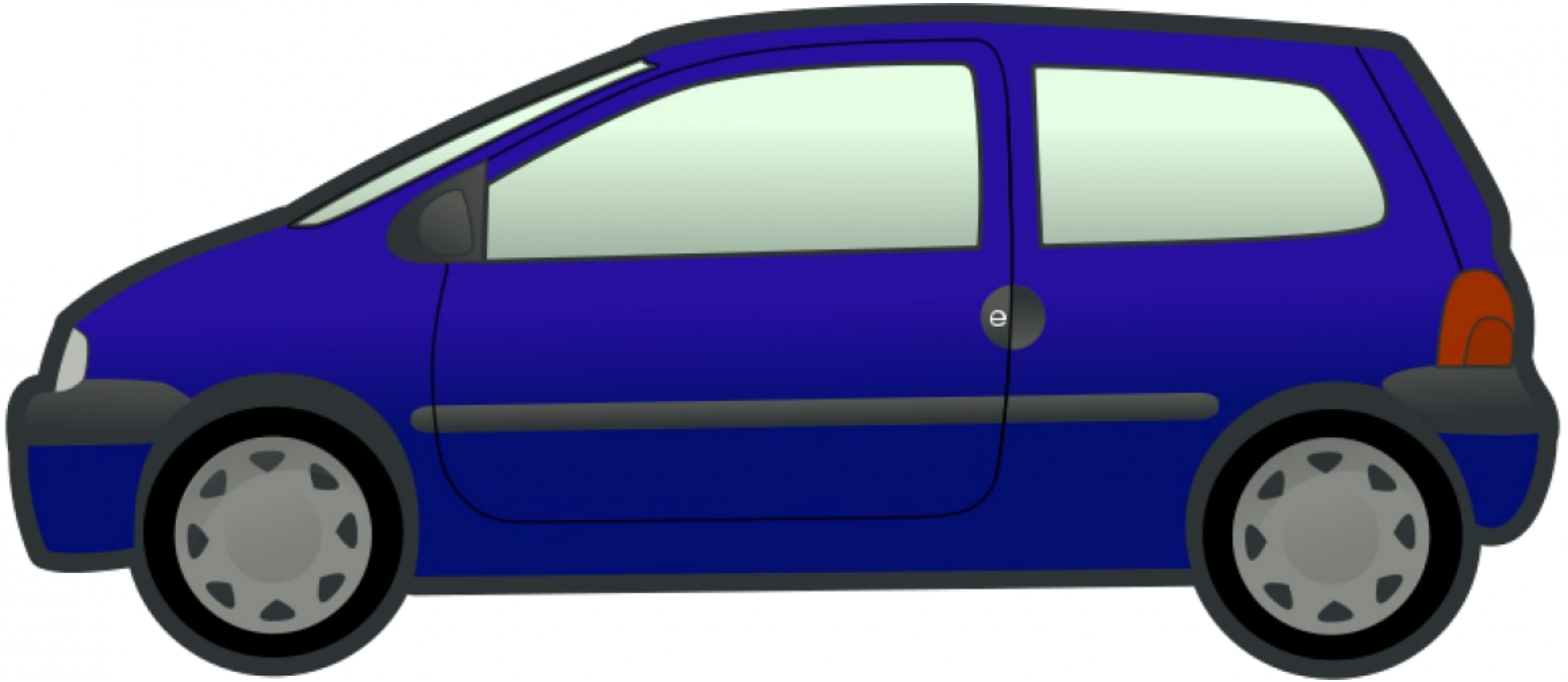
- Los objetos pueden ser físicos o conceptuales.
- Identificamos un objeto a partir de dos conceptos básicos
  - Características – Cosas que es o tiene
    - Se traducen en los programas como propiedades o atributos
  - Acciones – Cosas que hace
    - Se traducen en los programas como métodos

¿Qué es un objeto? – Declaración formal

- Los objetos son entidades que combinan estado, comportamiento e identidad.
  - El estado esta compuesto de uno o varios atributos a los que se habrán asignado valores concretos
  - El comportamiento está definido por los métodos con que se puede operar dicho objeto
  - La identidad es lo que diferencia un objeto del resto

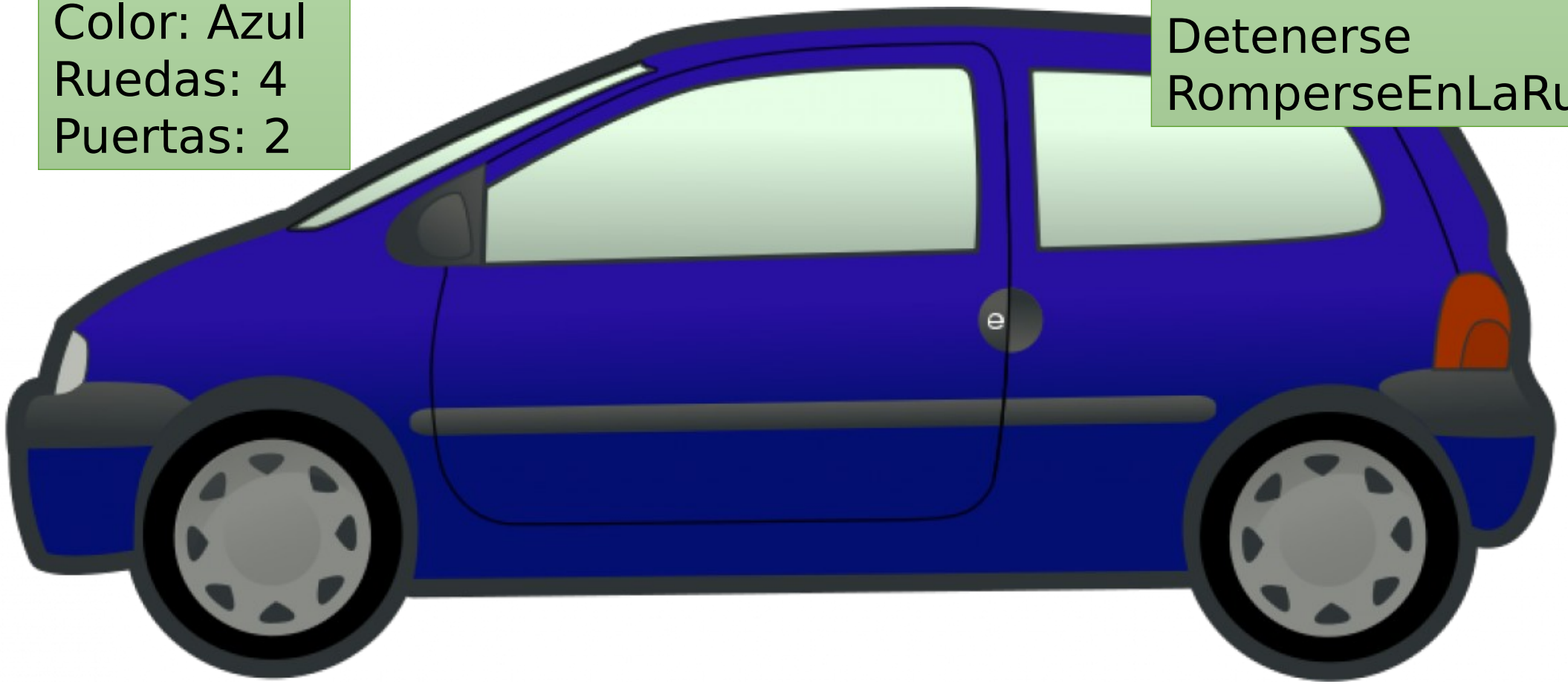
CAPTTON
IT CONSULTING

Identificando Objetos



Identificando Objetos

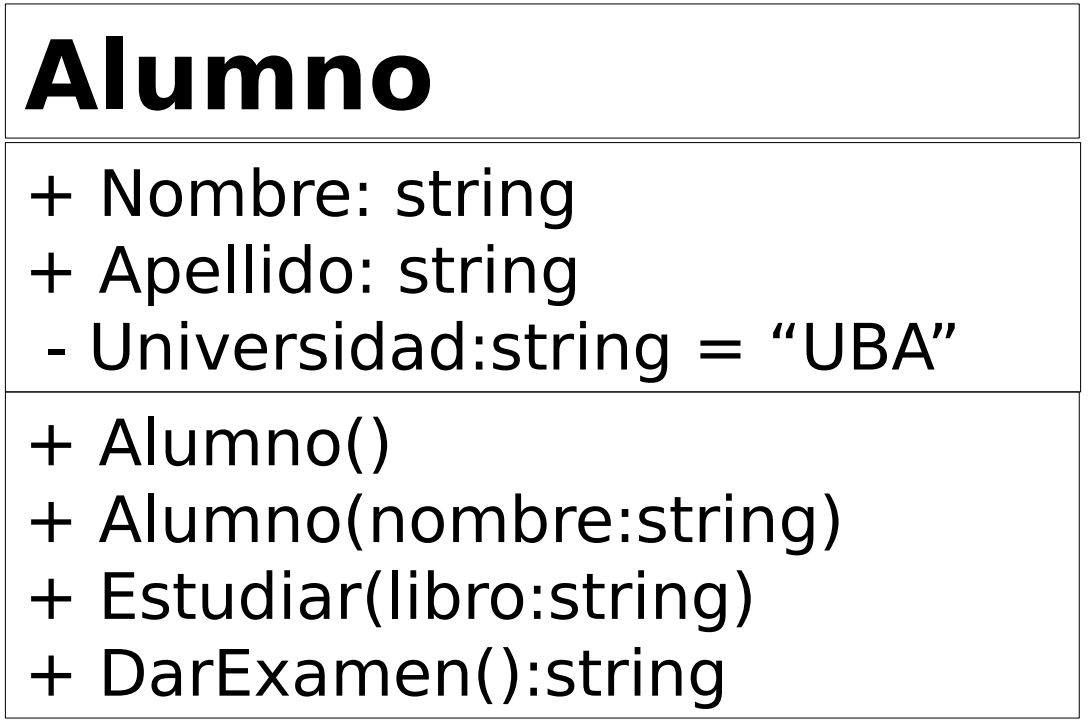
Color: Azul
Ruedas: 4
Puertas: 2



Arrancar
Detenerse
RomperseEnLaRuta

Modelando Objetos - UML

- Unified Modeling Languaje es una forma de describir objetos y sus relaciones.
- Hay distintos tipos, nosotros utilizaremos los de clases y relaciones



Encapsulamiento

Acoplamiento y Cohesión

Encapsulamiento

- El encapsulamiento se utiliza para proteger la información.
- Crear constructores para inicializar los valores.
- Crear métodos para modificar los atributos del objeto.
- Permite que las clases sean auto contenidas y reutilizables. Solo se puede acceder a ellas a través de sus métodos.
- Principio de Interface e Implementación.
- Reduce el Acoplamiento, simplifica los problemas.

Acoplamiento

- Grado en el cual una clase conoce de la otra
- Se busca que las clases tengan un '**bajo**' acoplamiento, esto significa, que una clase A conozca de una clase B sólo lo necesario para utilizar sus métodos y propiedades sin saber como B implementa los mismos

Ventajas de un bajo Acoplamiento

- Entender una clase sin leer otras
- Cambiar una clase sin afectar a otras
- Mejorar el mantenimiento del código

Cohesión

- Es la medida que indica si una clase tiene una función bien definida dentro del sistema
- Una prueba fácil de cohesión consiste en examinar una clase y decidir si todo su contenido está directamente relacionado con el nombre de la clase y descrito por el mismo

Ventajas de una alta Cohesión

- Entender qué hace una clase o método fácilmente
- Que los nombres de nuestras clases sean descriptivos
- Mejorar la reusabilidad de clases o métodos

Relacionando Clases

Asociación, Agregación y Composición

Asociación

- Los objetos asociados se ‘usan’ uno a otro pero sus ciclos de vida son separados. En otras palabras, uno puede existir sin el otro
- No hay un dueño definido
- Se puede pensar en términos de ‘USA UN’

Asociación – Ejemplo

- Si nosotros tenemos un objeto Supervisor y un objeto Tarjeta de Identificación, podemos establecer un relación de asociación diciendo
  - Un Supervisor **utiliza** su Tarjeta de Identidad para ingresar en la empresa

Agregación

- Como en la asociación, un objeto de la relación puede existir sin el otro pero a diferencia de la anterior hay un dueño definido en la relación
- Se puede pensar en términos de ‘TIENE UN’ objeto que puede seguir viviendo sin el dueño

Agregación - Ejemplo

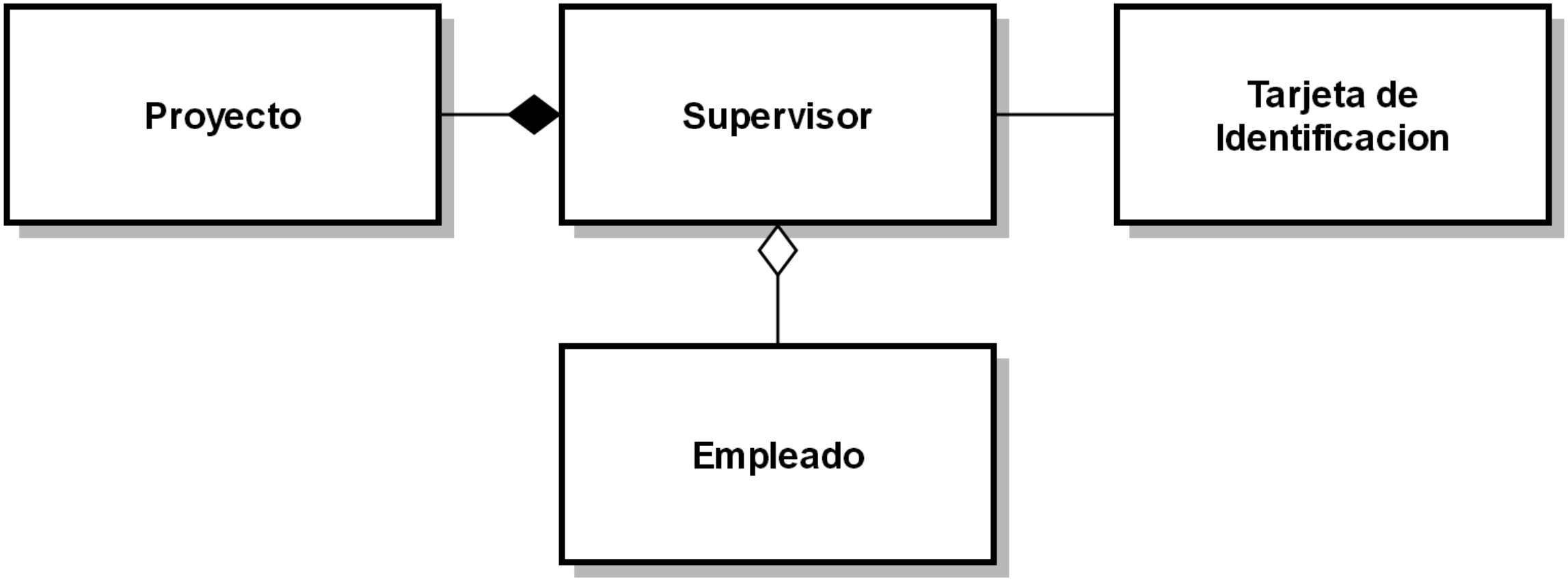
- Si nosotros tenemos un objeto Supervisor y un objeto Empleado, podemos establecer agregación diciendo
  - El supervisor **tiene** Empleados a cargo

Composición

- Tipo de relación dependiente. El objeto compuesto no existe fuera del ciclo de vida de su dueño
- Tiene un dueño definido
- Se puede pensar como un ‘TIENE UN’ objeto que no sobrevive sin el objeto dueño

Composición - Ejemplo

- Si nosotros tenemos un objeto Supervisor y un objeto Proyecto, podemos establecer composición diciendo
  - El supervisor **tiene a cargo** un proyecto
  - El salario del supervisor dependerá del éxito del proyecto


Proyecto
Supervisor
Tarjeta de Identificacion
Empleado

Resumiendo

	Asociación	Agregación	Composición
Dueño	Sin dueño	Un solo dueño	Un solo dueño
Ciclo de vida	Tienen ciclo de vida propio	Tienen ciclo de vida propio	El ciclo de vida es el del dueño
Objeto «Hijo»	Los objetos “Hijos” son completamente independientes	Los objetos “Hijos” pertenecen a un solo padre	Los objetos “Hijos” pertenecen a un solo padre

CAPTTON
IT CONSULTING