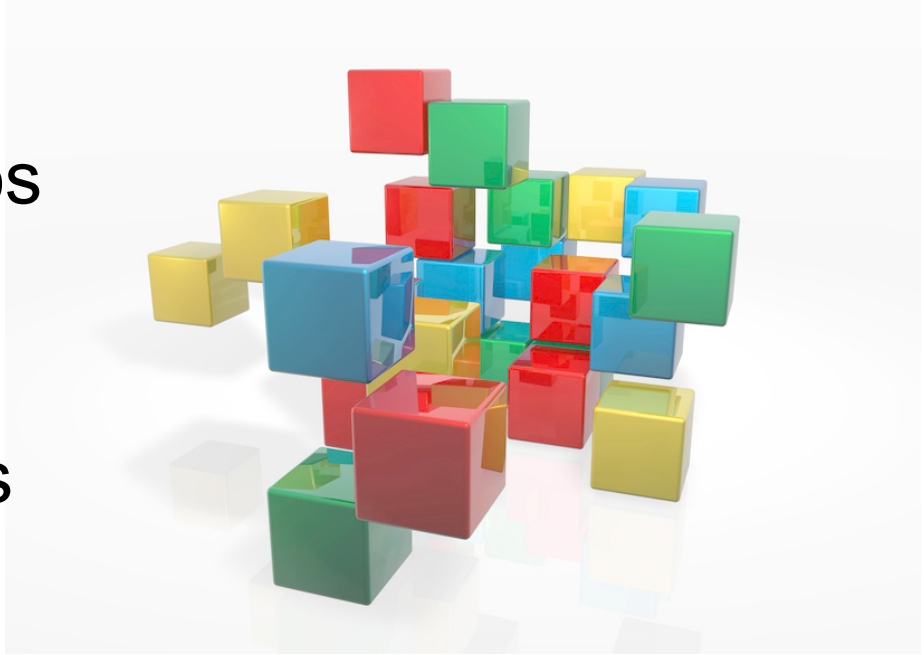
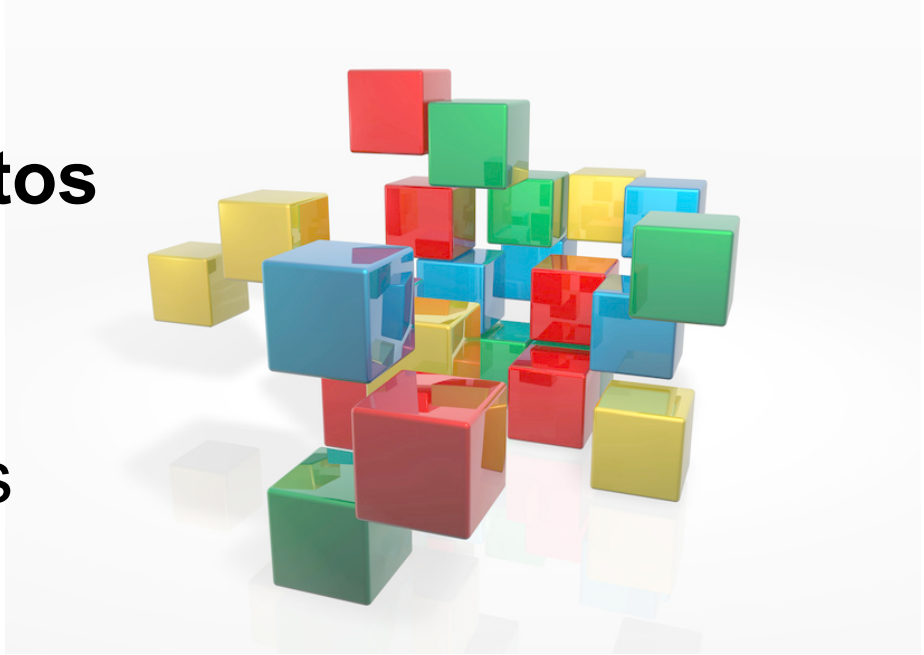
El Paradigma Objetos

- Contenido
  - La Naturaleza de los Objetos
  - Relaciones entre Objetos
  - La Naturaleza de las Clases
  - Relaciones entre Clases
  - La Interacción entre Clases y Objetos



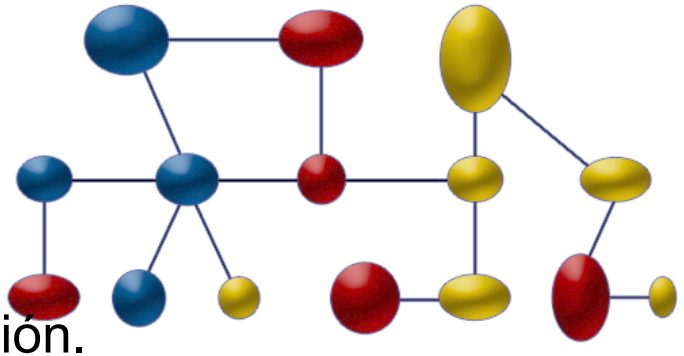
El Paradigma Objetos

- Contenido
  - **La Naturaleza de los Objetos**
  - Relaciones entre Objetos
  - La Naturaleza de las Clases
  - Relaciones entre Clases
  - La Interacción entre Clases y Objetos



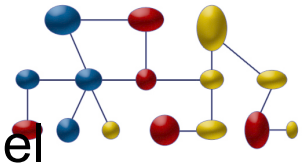
La Naturaleza de los Objetos (1)

- Se definió anteriormente a un objeto como ***"una entidad tangible que exhibe algún comportamiento bien definido".***
- Desde la perspectiva del conocimiento humano, y sin entrar en cuestiones filosóficas, un objeto es cualquiera de las siguientes cosas:
  - Una cosa tangible y/o visible.
  - Algo que puede comprenderse intelectualmente.
  - Algo hacia lo que se dirige un pensamiento o acción.



La Naturaleza de los Objetos (2)

- Un objeto modela alguna parte de la realidad y es, por lo tanto, algo que existe en el tiempo y en el espacio.



- Los objetos del mundo real no son el único tipo de objeto de interés en el desarrollo del software. Entonces, en términos aún más generales, podemos definir un objeto como:

"Cualquier cosa (real o abstracta) que tenga una frontera definida con nitidez."

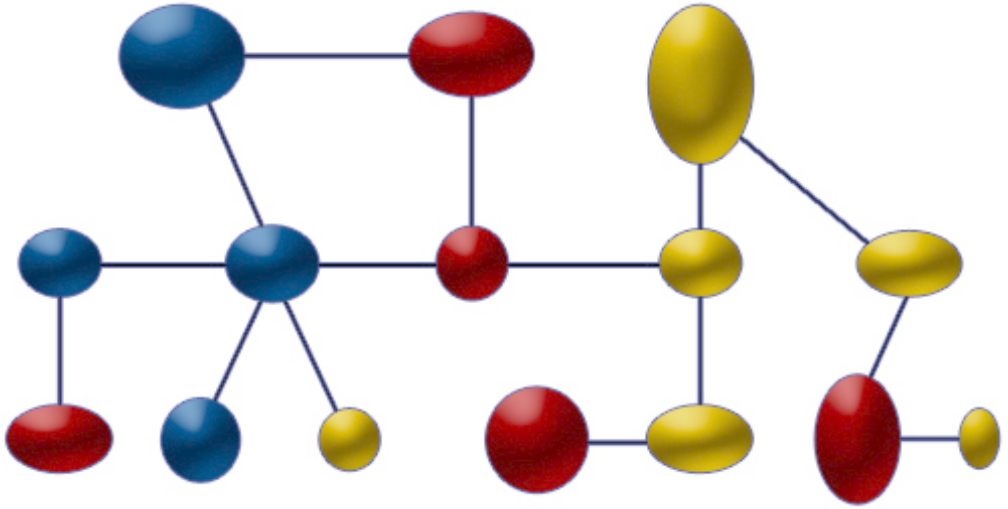
- Algunos objetos pueden tener límites conceptuales precisos, pero aún así pueden representar eventos o procesos intangibles.
  - Por ejemplo, la inscripción de un alumno para cursar una asignatura en la facultad, puede tratarse como un objeto, porque tiene una frontera conceptual clara, interactúa con otros objetos y presenta un comportamiento bien definido.

La Naturaleza de los Objetos (3)

- Así, podemos definir formalmente un objeto como sigue:

"Un objeto tiene un estado, comportamiento e identidad; la estructura y comportamiento de objetos similares están definidos en su clase común; los términos instancia y objeto son intercambiables."

- Estado
- Comportamiento
- Identidad

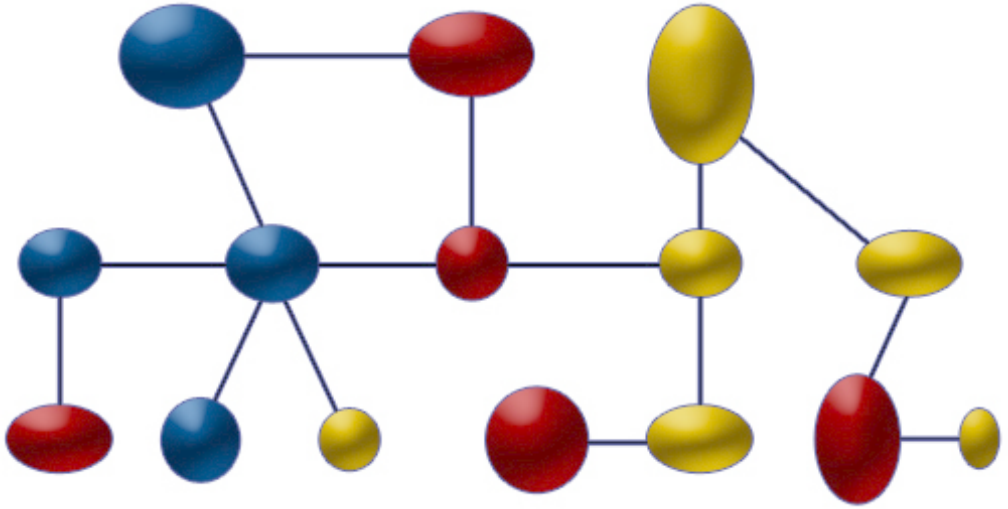


La Naturaleza de los Objetos (3)

- Así, podemos definir formalmente un objeto como sigue:

"Un objeto tiene un estado, comportamiento e identidad; la estructura y comportamiento de objetos similares están definidos en su clase común; los términos instancia y objeto son intercambiables."

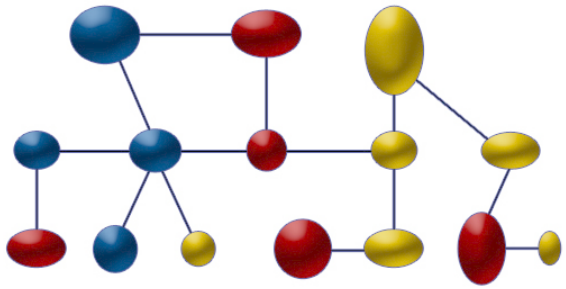
- **Estado**
- Comportamiento
- Identidad



Estado (1)

- El comportamiento de un objeto está influenciado por su historia: el orden en que se opera sobre el objeto es importante.
- La razón para este comportamiento dependiente del tiempo y los eventos, es la existencia de un **estado** en el interior del objeto.
- Definición

"El estado de un objeto abarca todas las propiedades (normalmente estáticas) del mismo más los valores actuales (normalmente dinámicos) de cada una de esas propiedades."

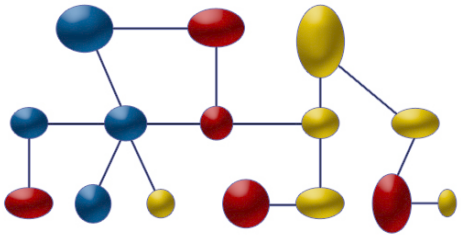


Estado (2)

Estado (3)

Supongamos una máquina expendedora de latas de gaseosas:

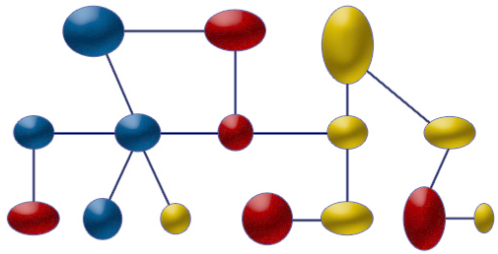
- El comportamiento usual de tales objetos es que cuando se introducen monedas en una ranura y se pulsa un botón con nuestra gaseosa preferida, surge la bebida de la máquina.
- Si un usuario realiza primero la selección y luego introduce las monedas, la máquina no hace nada porque se han violado las suposiciones básicas de funcionamiento.
- También, si el usuario ignora la luz de advertencia que dice “sólo dinero exacto”, seguro no recibirá cambio.
- Un estado esencial asociado a esta máquina será la cantidad de dinero que acaba de introducir un usuario sin haber realizado la selección.
- Otro estado esencial sería la cantidad de bebidas disponibles.



© Maximiliano Odstrcil - 9

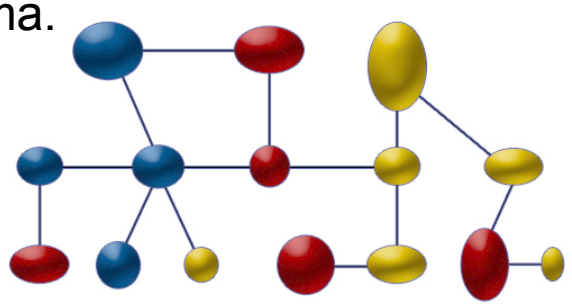
Estado (4)

- Otra propiedad de la máquina es que puede aceptar monedas. Esta es una **propiedad estática**, lo que significa que es una característica esencial de una máquina expendedora.
- En contraste, la cantidad de monedas que se ha aceptado en un momento dado, representa el **valor dinámico** de esa propiedad.
- Análogamente, una propiedad de un objeto persona es que tiene una edad determinada (propiedad estática); y la cantidad de años en un momento dado representa el valor dinámico de esa propiedad.
- Se dice que los valores son **"normalmente dinámicos"** porque en algunos casos los valores son estáticos: el número de serie de la máquina expendedora es una propiedad estática y su valor es estático.



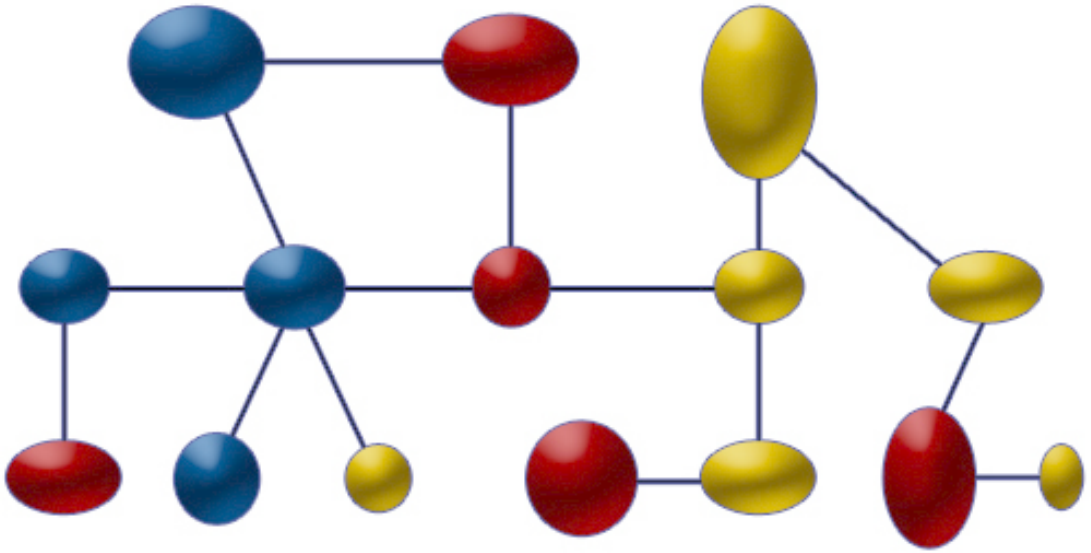
Estado (5)

- Una **propiedad** es una característica inherente o distintiva, un rasgo o cualidad que contribuye a hacer que un objeto sea un objeto y no otro.
- Las propiedades suelen ser estáticas, porque los atributos suelen ser inmutables y fundamentales para la naturaleza del objeto.
  - Por ejemplo, no tiene sentido de hablar de un DNI para una jirafa ni el número de pétalos de una silla.
- Todas las propiedades tienen algún valor, que puede ser una mera cantidad o puede denotar a otro objeto.
  - Por ejemplo, parte del estado de una persona puede tener el valor 25, que denota la edad en años de la misma.



Estado (6)

- El hecho de que todo objeto tiene un estado implica que todo objeto toma cierta cantidad de espacio, ya sea en el mundo físico o en la memoria de la computadora.
- Puede decirse que todos los objetos de un sistema encapsulan algún estado, y que todo estado de un sistema está encapsulado en objetos.

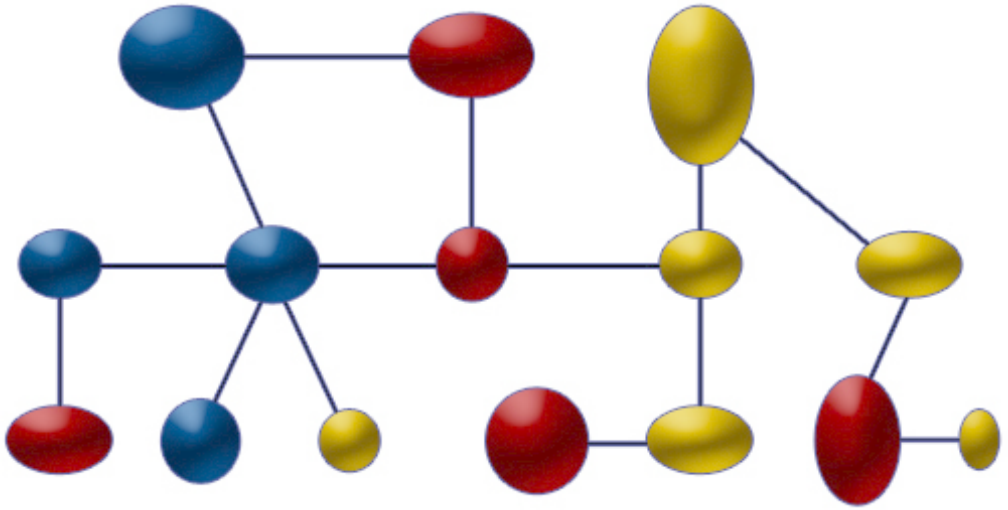


La Naturaleza de los Objetos (3)

- Así, podemos definir formalmente un objeto como sigue:

"Un objeto tiene un estado, comportamiento e identidad; la estructura y comportamiento de objetos similares están definidos en su clase común; los términos instancia y objeto son intercambiables."

- Estado
- **Comportamiento**
- Identidad

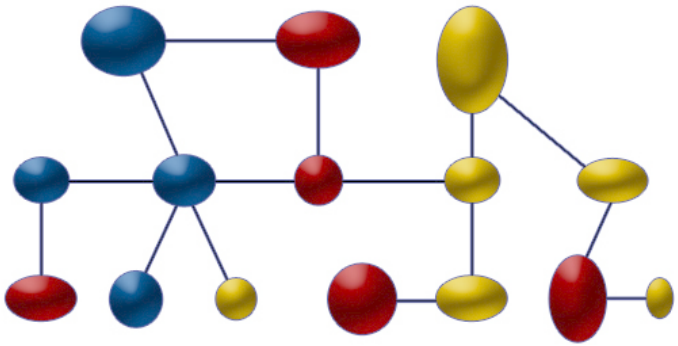


Comportamiento (1)

- Ningún objeto existe de forma aislada. En vez de eso, los objetos reciben acciones, y ellos mismos actúan sobre otros objetos.
- Definición:

"El comportamiento es cómo actúa y reacciona un objeto, en términos de sus cambios de estado y paso de mensajes."

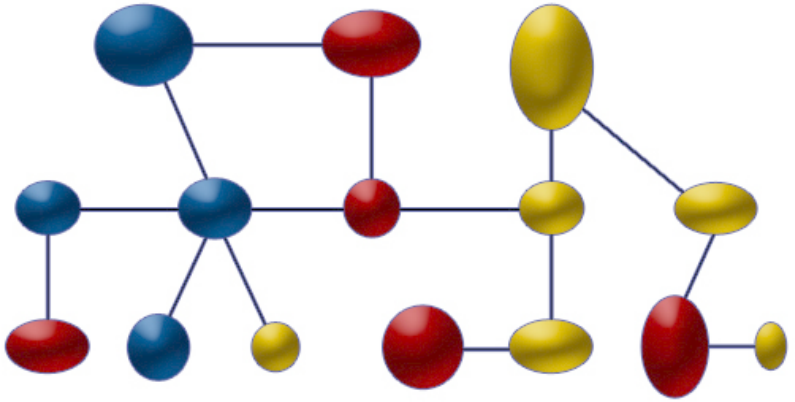
- En otras palabras, el comportamiento de un objeto representa su actividad visible y comprobable exteriormente.



Comportamiento (2)

Comportamiento (3)

- Una **operación** es una acción que un objeto efectúa sobre otro con el fin de provocar una reacción.
- También se habla de que un objeto le pasa un **mensaje** a otro objeto, o que un objeto invoca una **función miembro** de otro objeto.
- En la mayoría de los lenguajes de programación orientados a objetos, las operaciones que los clientes pueden realizar sobre un objeto suelen declararse como **métodos**.

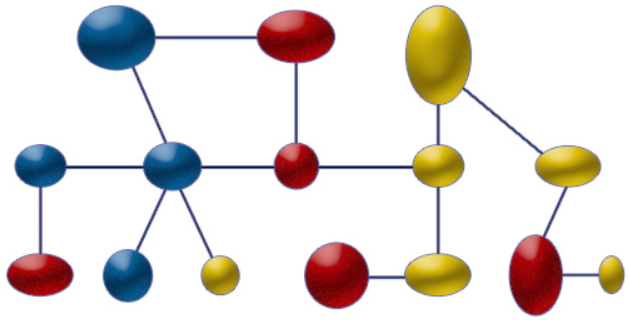


© Maximiliano Odstrcil - 16

Comportamiento (4)

- El paso de mensajes es una parte de la ecuación que define el comportamiento de un objeto.
- Se puede decir que el comportamiento de un objeto es función de su estado así como de la operación que se realiza sobre él. Algunas operaciones pueden modificar, además, el estado de un objeto.
- Podemos refinar la definición de estado:

"El estado de un objeto representa los resultados acumulados de su comportamiento."



Comportamiento (5)

- Las operaciones, que denotan un servicio que una clase ofrece a sus clientes, se agrupan en cinco tipos diferentes.

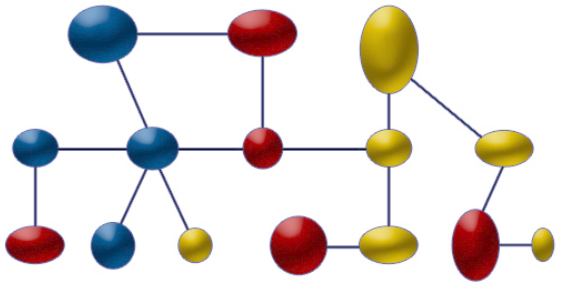
  a) **Modificador**: una operación que altera el estado de un objeto.

  b) **Selector**: una operación que accede al estado de un objeto, pero no altera ese estado.

  c) **Iterador**: una operación que permite acceder a todas las partes de un objeto en algún orden perfectamente establecido.

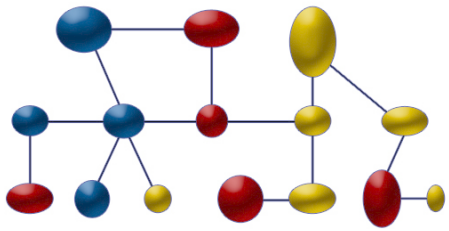
  d) **Constructor**: una operación que crea un objeto y/o inicializa su estado.

  e) **Destructor**: una operación que libera el estado de un objeto y/o destruye el propio objeto.



Comportamiento (6)

- Como dijimos antes, el conjunto de todos los métodos asociados con un objeto concreto constituyen su **protocolo**.
- El protocolo define así la envoltura del comportamiento admisible en un objeto, y por lo tanto engloba la visión estática y dinámica del mismo.
- Algunos otros conceptos asociados al comportamiento de un objeto, se detallan a continuación:
  - **Papel (rol):** las particiones del protocolo de un objeto en grupos lógicos de comportamiento denotan los papeles que un objeto puede desarrollar.
  - **Objeto activo**: es aquel que comprende su propio hilo de control. Pueden exhibir su comportamiento sin que ningún otro objeto opere sobre ellos (autonomía).
  - **Objeto pasivo**: es aquel que no comprende su hilo de control. Sufren un cambio de estado cuando se actúa explícitamente sobre ellos.

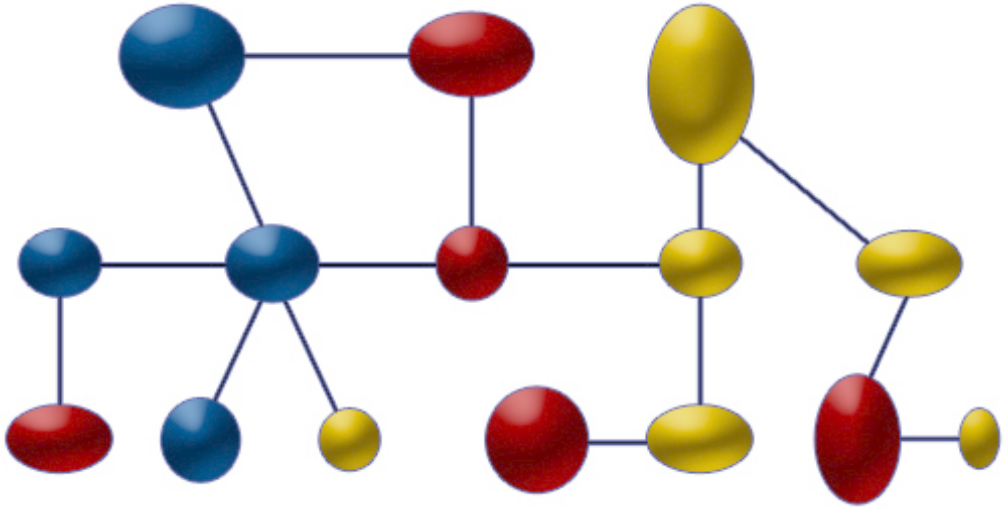


La Naturaleza de los Objetos (3)

- Así, podemos definir formalmente un objeto como sigue:

"Un objeto tiene un estado, comportamiento e identidad; la estructura y comportamiento de objetos similares están definidos en su clase común; los términos instancia y objeto son intercambiables."

- Estado
- Comportamiento
- **Identidad**

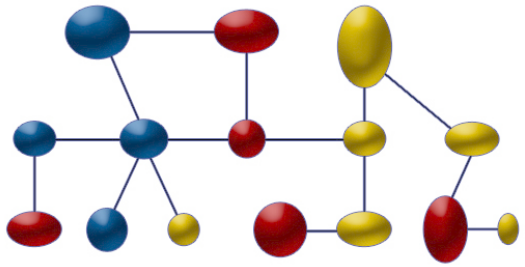


Identidad (1)

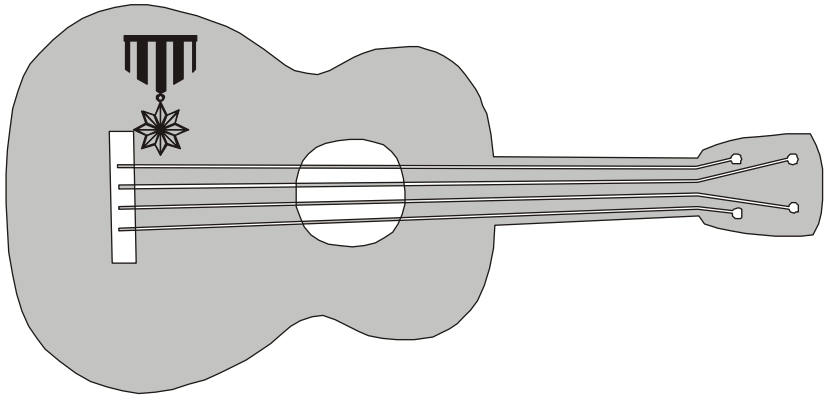
- Definición:

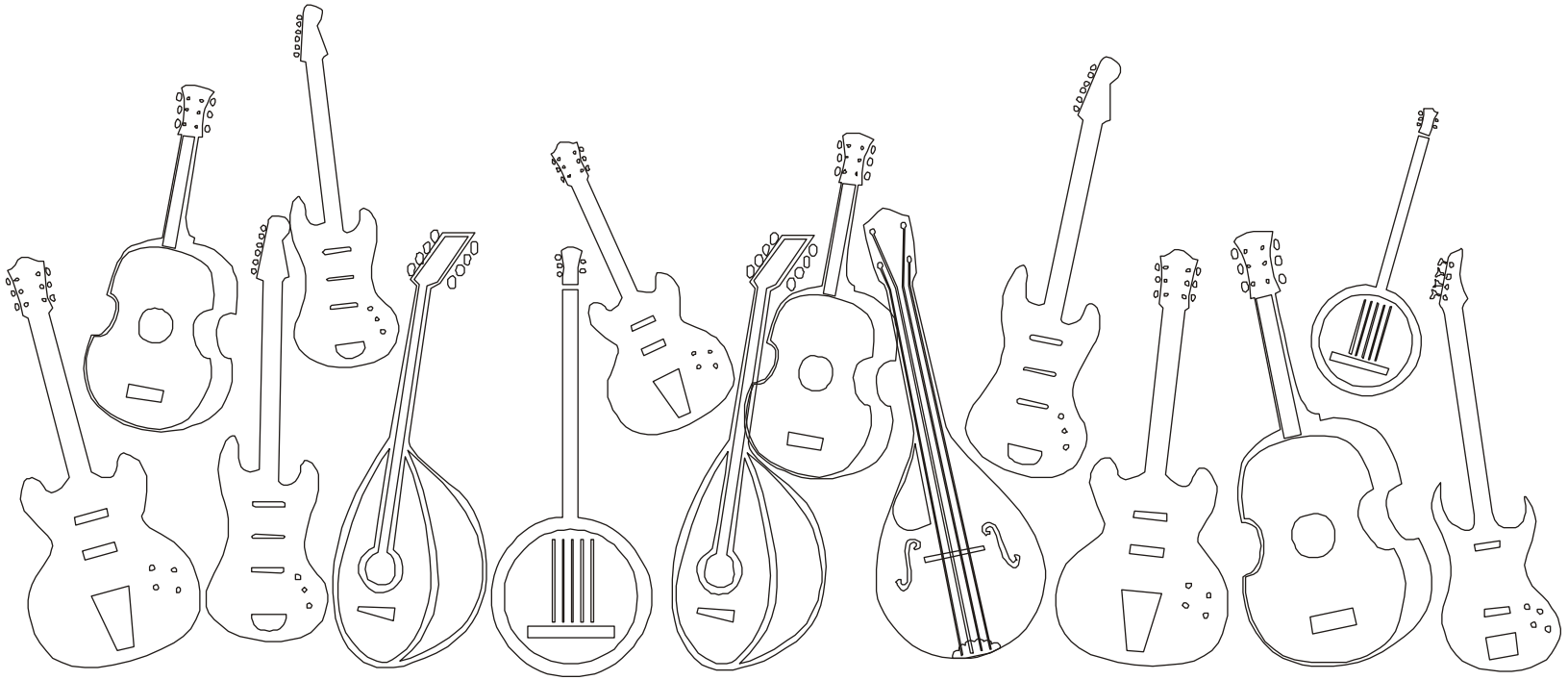
"La identidad es aquella propiedad de un objeto que lo distingue de todos los demás objetos."

- Si la abstracción representa un simple registro de otros objetos y no tiene un comportamiento verdaderamente interesante, es recomendable construir una estructura de tipos.
- Si la abstracción exige un comportamiento más intenso que la simple introducción y recuperación de elementos, se debe construir una clase.



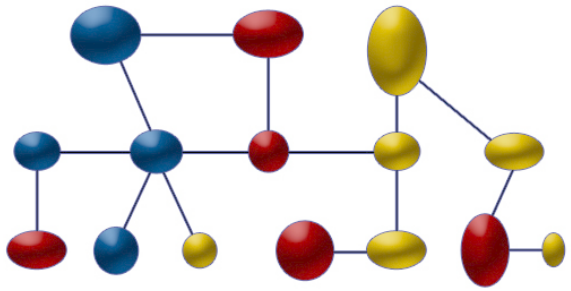
Identidad (2)





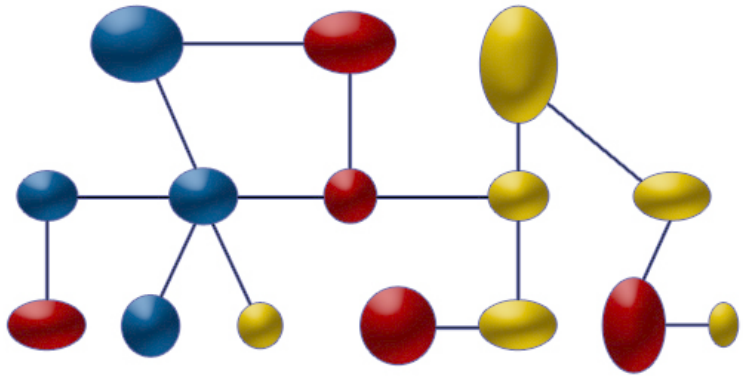
Identidad (3)

- La identidad única (pero no necesariamente el nombre) de cada objeto se preserva durante el tiempo de vida del mismo, incluso cuando su estado cambia.
- El tiempo de vida de un objeto se extiende desde el momento en que se crea por primera vez (y consume así espacio por primera vez) hasta que ese espacio se recupera cuando el objeto es destruido.
- Cada vez que se crea un objeto, se invoca automáticamente a su constructor que asigna espacio en memoria al objeto y establece un estado inicial.



Identidad (4)

- Cada vez que se destruye un objeto, se invoca automáticamente a su destructor que devuelve el espacio asignado al objeto y libera recursos ocupados.
- Los objetos persistentes tienen una semántica ligeramente diferente respecto a la destrucción (y la creación): dichas operaciones son función de la política de la base de datos de referencia

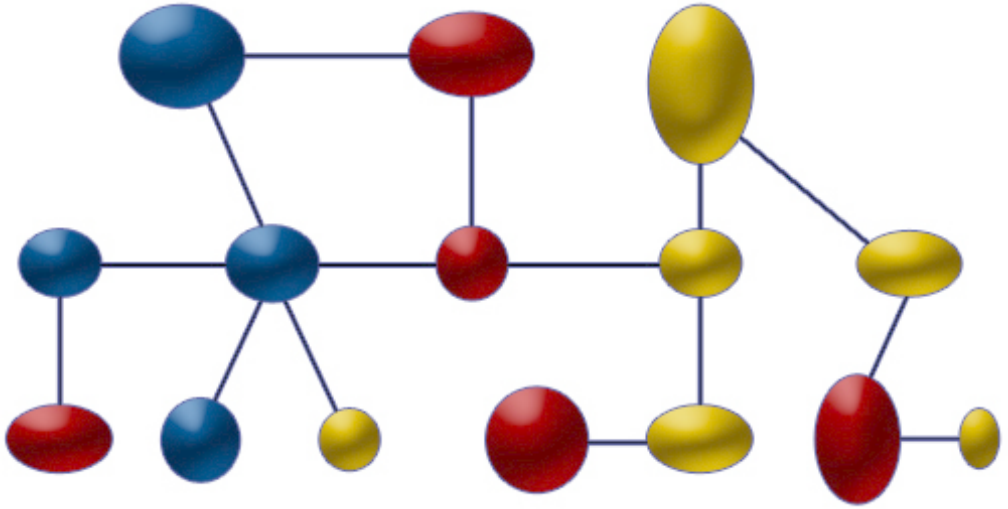


La Naturaleza de los Objetos (3)

- Así, podemos definir formalmente un objeto como sigue:

"Un objeto tiene un estado, comportamiento e identidad; la estructura y comportamiento de objetos similares están definidos en su clase común; los términos instancia y objeto son intercambiables."

- Estado
- Comportamiento
- Identidad

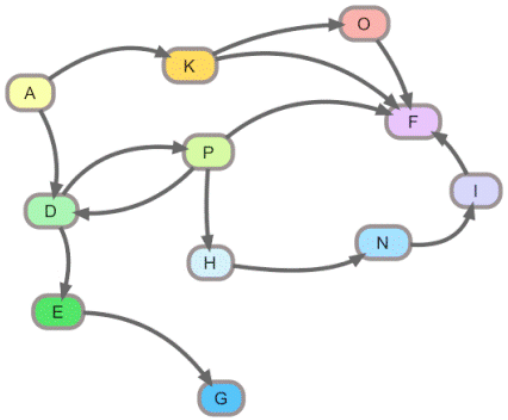


El Paradigma Objetos

- Contenido
  - La Naturaleza de los Objetos
  - **Relaciones entre Objetos**
  - La Naturaleza de las Clases
  - Relaciones entre Clases
  - La Interacción entre Clases y Objetos

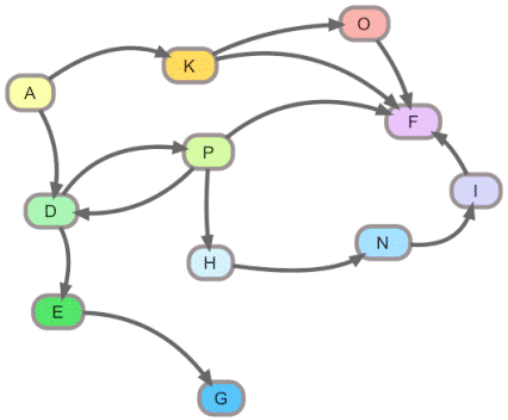
Relaciones entre Objetos

- Un objeto por sí mismo es bastante poco interesante. Los objetos contribuyen al comportamiento de un sistema colaborando con otros.
- La relación entre dos objetos cualesquiera abarca las suposiciones que cada uno realiza acerca del otro, incluyendo qué operaciones pueden realizarse y qué comportamiento se obtiene.
- Se ha encontrado que hay dos tipos de jerarquías de objetos:
  - Enlaces
  - Agregación



Relaciones entre Objetos

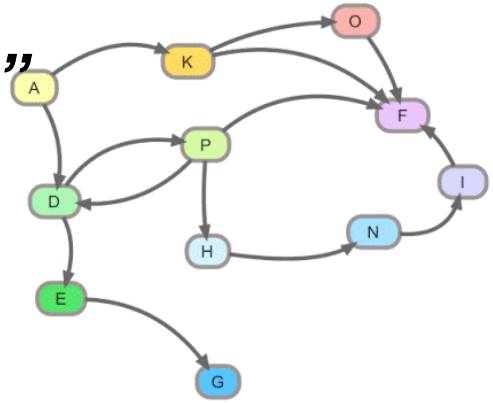
- Un objeto por sí mismo es bastante poco interesante. Los objetos contribuyen al comportamiento de un sistema colaborando con otros.
- La relación entre dos objetos cualesquiera abarca las suposiciones que cada uno realiza acerca del otro, incluyendo qué operaciones pueden realizarse y qué comportamiento se obtiene.
- Se ha encontrado que hay dos tipos de jerarquías de objetos:
  - **Enlaces**
  - Agregación



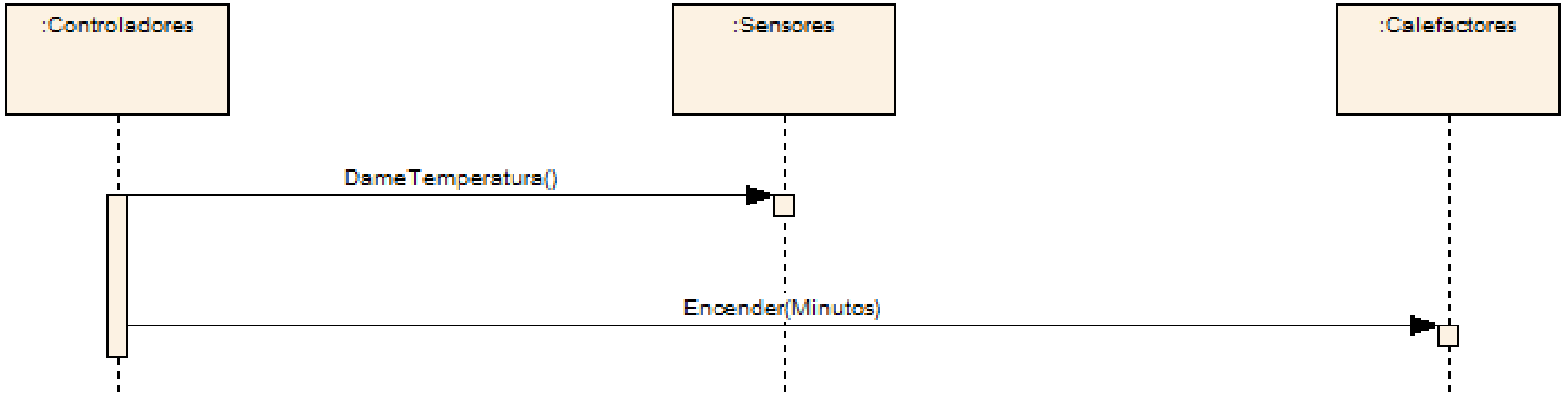
Enlaces (1)

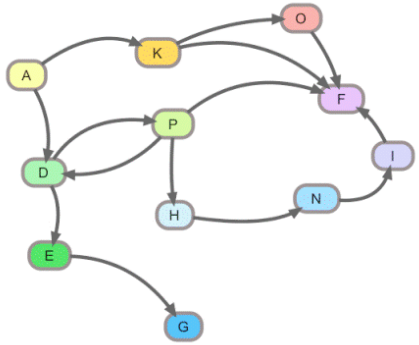
- El enlace es una conexión física o conceptual entre objetos.
- Un objeto colabora con otros objetos a través de sus enlaces con éstos.
- Definición:

"Un enlace denota la asociación específica por la cual un objeto (cliente) utiliza los servicios de otro objeto (servidor), o a través de la cual un objeto puede comunicarse con otro."



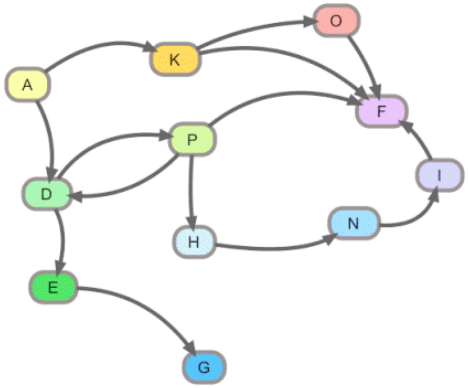
Enlaces (2)


:Controladores
:Sensores
:Calefactores
DameTemperatura()
Encender(Minutos)


O
K
A
F
P
I
D
N
H
E
G

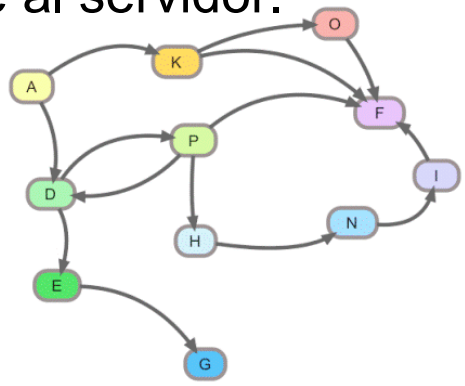
Enlaces (3)

- En la figura observamos parte de un modelo que ayuda a controlar la temperatura en un vivero.
  - La línea entre los objetos representa la existencia de un enlace entre ambos.
  - La etiqueta denota el nombre del mensaje y la flecha la dirección del mismo.
  - En este caso el objeto Controlador (cliente) envía un mensaje (dametemperatura) al objeto Sensor A (servidor)



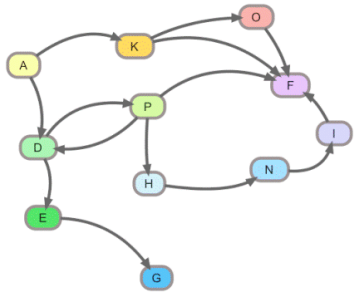
Enlaces (4)

- El paso de mensajes entre objetos es típicamente unidireccional, aunque ocasionalmente puede ser bidireccional.
- Aunque el mensaje fluye desde el cliente hacia el servidor, los datos pueden fluir en ambas direcciones:
  - En el caso de la operación dametemperatura(), el dato fluye desde el servidor al cliente.
  - En la operación encender (t), el dato fluye desde el cliente al servidor.



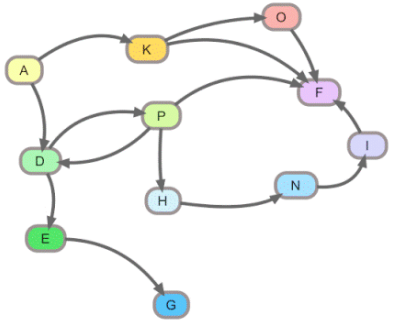
Enlaces (5)

- Como participante de un enlace, un objeto puede desempeñar uno de los siguientes roles o papeles:
  - **Actor**: un objeto que puede operar sobre otros objetos pero nunca se opera sobre él por parte de otros objetos.
  - **Servidor**: un objeto que nunca opera sobre otros objetos; sólo otros objetos operan sobre él.
  - **Agente**: un objeto que puede operar sobre otros objetos y además otros objetos pueden operar sobre él.
- Un agente se crea normalmente para realizar algún trabajo en nombre de un actor u otro agente.



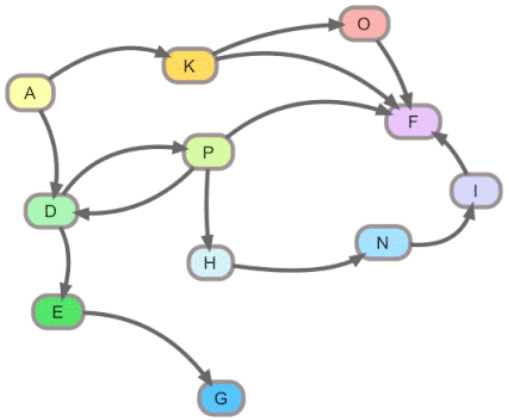
Enlaces (6)

- Para que un objeto A envíe un mensaje a otro B, debe suceder que B sea visible para A.
- Las 4 formas en que un objeto puede tener visibilidad para otro son:
  - El objeto servidor es global para el cliente.
  - El objeto servidor es un parámetro de alguna operación del cliente.
  - El objeto servidor es parte del objeto cliente.
  - El objeto servidor es un objeto declarado localmente en alguna operación del cliente.



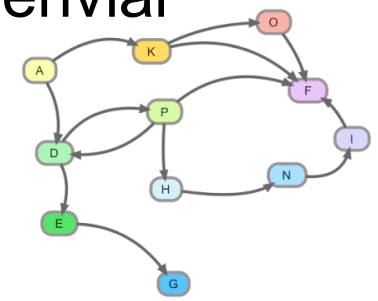
Relaciones entre Objetos

- Un objeto por sí mismo es bastante poco interesante. Los objetos contribuyen al comportamiento de un sistema colaborando con otros.
- La relación entre dos objetos cualesquiera abarca las suposiciones que cada uno realiza acerca del otro, incluyendo qué operaciones pueden realizarse y qué comportamiento se obtiene.
- Se ha encontrado que hay dos tipos de jerarquías de objetos:
  - Enlaces
  - **Agregación**

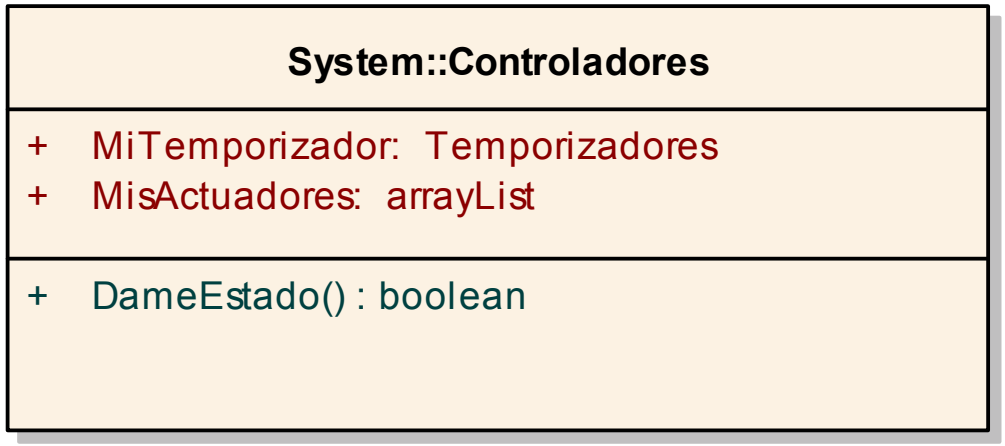


Agregación (1)

- Se observa que los enlaces denotan relaciones de igual a igual o cliente/servidor entre objetos.
- La agregación denota una **jerarquía todo/parte**, con la capacidad de ir desde el todo (agregado) hasta sus partes (atributos).
- La agregación puede o no denotar contención física y tienen la ventaja con respecto al enlace de que encapsula partes y secretos del todo.
- Por implicación, un objeto que es atributo de otro tiene un enlace a su agregado. A través de este enlace, el agregado puede enviar mensajes a sus partes.

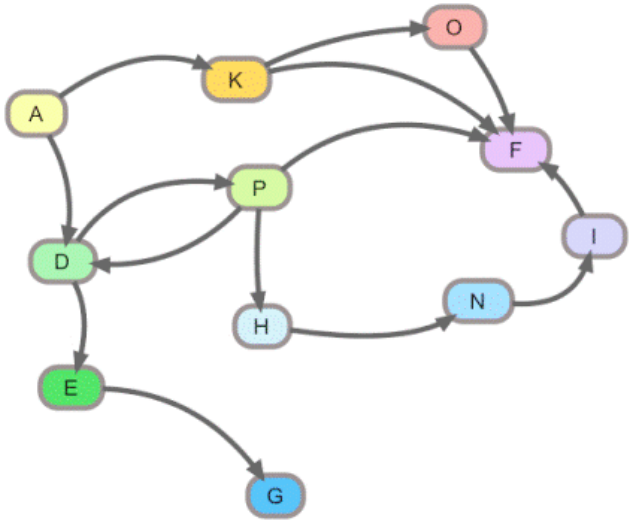


Agregación (2)



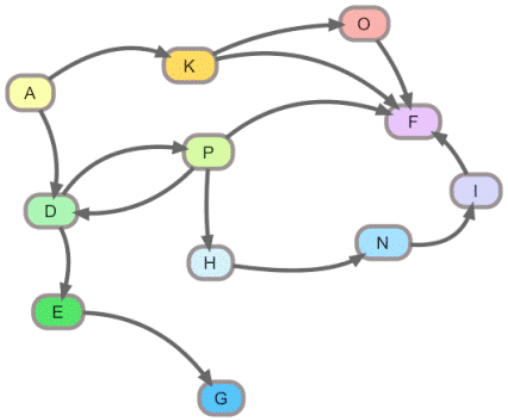
Agregación (3)

- En la figura se observa que el objeto Controlador tiene un atributo t que es un objeto temporizador.
- En otras palabras, t es una parte del estado del Controlador (todo).
- Existe un enlace entre el temporizador y el controlador, dado por la agregación, a través del cual se puede enviar mensajes.



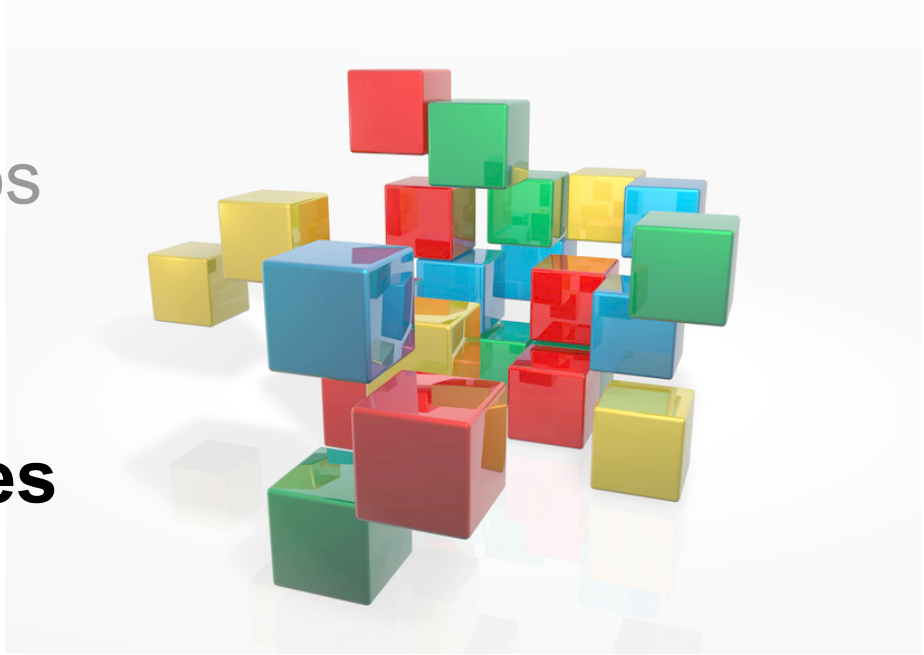
Relaciones entre Objetos

- Un objeto por sí mismo es bastante poco interesante. Los objetos contribuyen al comportamiento de un sistema colaborando con otros.
- La relación entre dos objetos cualesquiera abarca las suposiciones que cada uno realiza acerca del otro, incluyendo qué operaciones pueden realizarse y qué comportamiento se obtiene.
- Se ha encontrado que hay dos tipos de jerarquías de objetos:
  - Enlaces
  - Agregación



El Paradigma Objetos

- Contenido
  - La Naturaleza de los Objetos
  - Relaciones entre Objetos
  - **La Naturaleza de las Clases**
  - Relaciones entre Clases
  - La Interacción entre Clases y Objetos



La Naturaleza de las Clases (1)

- Los conceptos de clase y objeto están estrechamente ligados, porque no puede hablarse de un objeto sin atención a su clase.
- Existen diferencias importantes entre ambos términos:
  - Un objeto es una entidad concreta que existe en el tiempo y en el espacio.
  - Una clase representa sólo una abstracción, la esencia de un objeto.
- Así, se puede hablar de la clase Sillas, que representa las características comunes a todas las sillas.
- Para identificar a una silla en particular en esa clase, hay que hablar de “esta silla” o “aquella silla”.

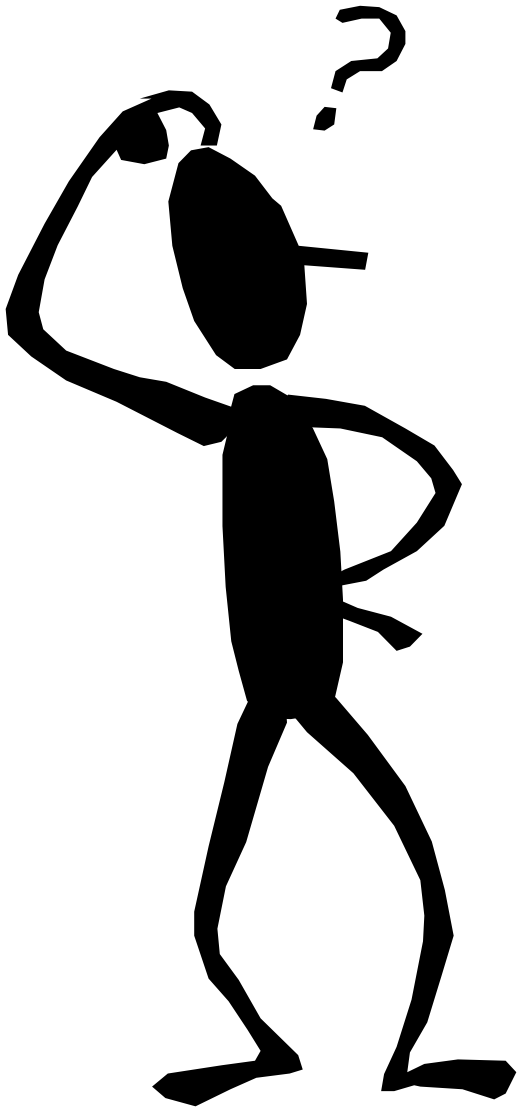
La Naturaleza de las Clases (2)

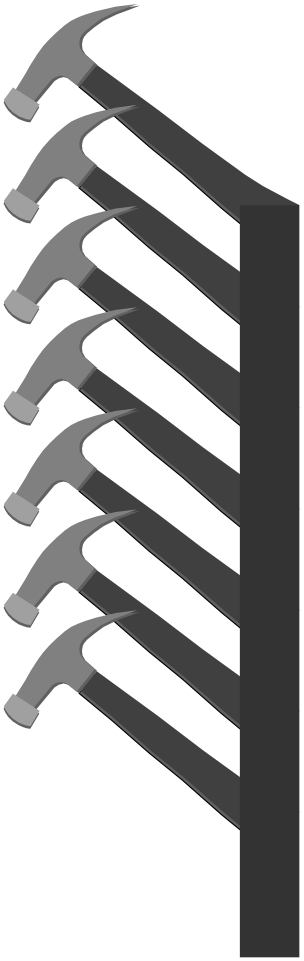
- Definición:

"Una clase es un conjunto de objetos que comparten una estructura y un comportamiento común."

- Un solo objeto no es más que la instancia de una clase.
- Los objetos que no comparten estructura y comportamiento similares no pueden agruparse en una clase porque no están relacionados entre sí a no ser por su naturaleza general de objetos.

La Naturaleza de las Clases (3)





Clase Martillos

La Naturaleza de las Clases (4)

- Un objeto individual es una entidad concreta que desempeña algún papel en el sistema global.
- En cambio, la clase captura la estructura y el comportamiento comunes a todos los objetos relacionados.
- Una clase sirve como una especie de contrato que vincula a una abstracción y a todos sus clientes.
- Esta visión de la programación como un contrato, lleva a distinguir entre la **visión externa** y la **visión interna** de una clase.

La Naturaleza de las Clases (5)

Interfaz e implementación:

"La interfaz proporciona una visión externa, ocultando la estructura y el comportamiento y, por lo tanto, enfatizando la abstracción."

- La interfaz se compone principalmente de las declaraciones de todas las operaciones aplicables a las instancias de dicha clase, pero también puede incluir la declaración de otras clases, constantes, variables y excepciones.

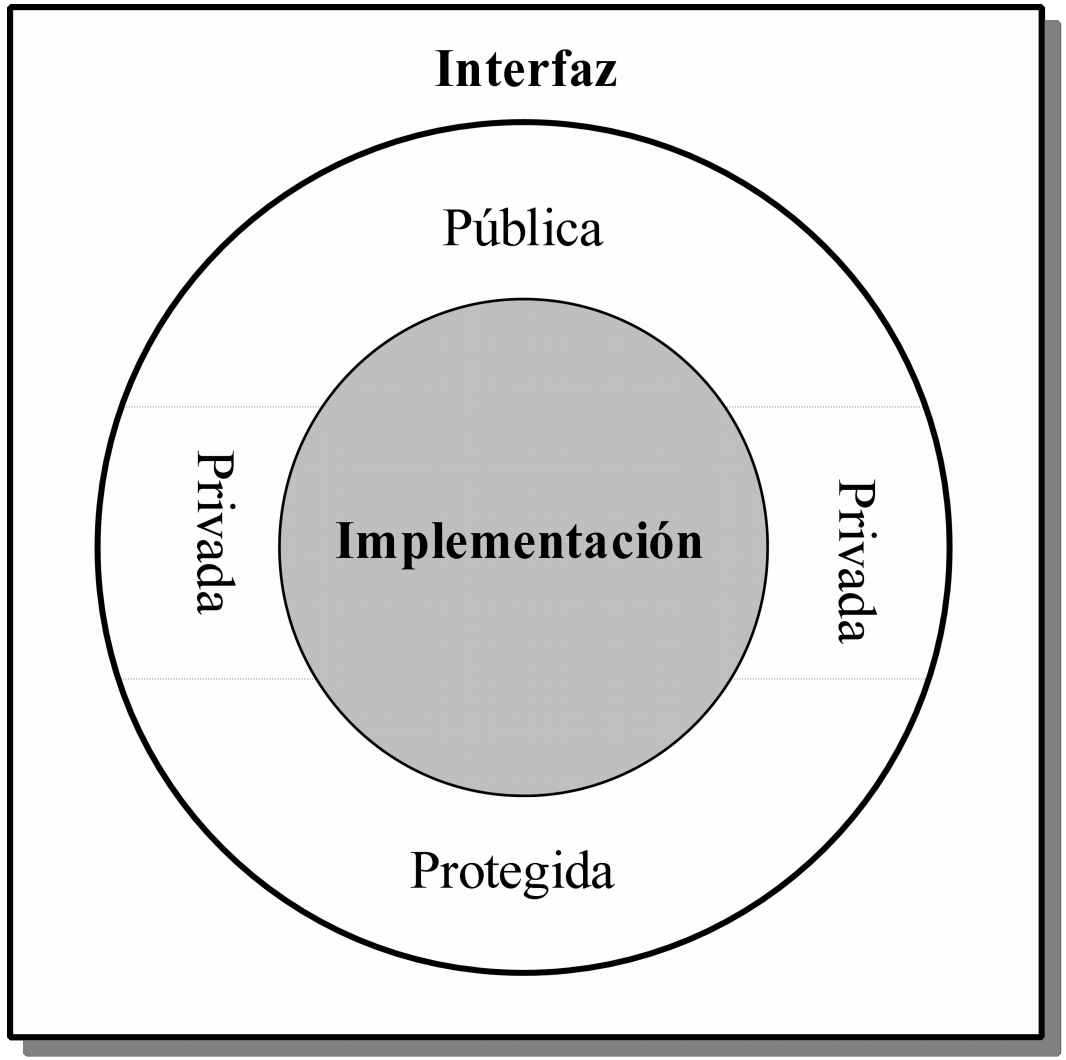
La Naturaleza de las Clases (6)

- La Interfaz de una clase se puede dividir en tres partes:
  - **Pública**: una declaración accesible a todos los clientes.
  - **Protegida**: una declaración accesible sólo a la propia clase, sus subclases y sus clases amigas.
  - **Privada**: una declaración accesible sólo a la propia clase y sus clases amigas.

La implementación de una clase proporciona la visión interna, es decir, los secretos de su comportamiento.

- La implementación se compone principalmente de la implementación de todas las operaciones definidas en la interfaz de la misma.

La Naturaleza de las Clases (7)



El Paradigma Objetos

- Contenido
  - La Naturaleza de los Objetos
  - Relaciones entre Objetos
  - La Naturaleza de las Clases
  - **Relaciones entre Clases**
  - La Interacción entre Clases y Objetos

Relaciones entre Clases (1)

- Consideremos las analogías y diferencias entre las siguientes clases de objetos: mamíferos, vacas, perros y orejas.
- Pueden hacerse las observaciones siguientes:
  - Una vaca es un mamífero.
  - Un perro es un tipo de mamífero, pero distinto de la vaca.
  - Los Fox Terrier y los Doberman son razas (tipos) diferentes de perros.
  - Las vacas y los perros tienen 2 orejas.
  - Algunos perros se utilizan en el pastoreo de vacas.

Relaciones entre Clases (2)

- Las clases, al igual que los objetos, no existen aisladamente, sino que suelen estar relacionadas por vías muy diversas e interesantes, formando la estructura de clases del diseño.

- Se establecen relaciones entre clases porque:
  - Una relación entre clases podría indicar que comparten algo.
    - Por ejemplo las vacas y los perros son mamíferos, es decir que ambos tienen glándulas mamarias y cuerpo cubierto de pelos, entre otras cosas.
  - Una relación entre clases podría indicar algún tipo de conexión semántica.
    - Por ejemplo los Fox Terrier y los Doberman se parecen más que los perros y las vacas, y que existe una conexión entre los perros y las vacas: los perros ayudan a arrear el ganado vacuno.

Relaciones entre Clases (3)

- Existen tres tipos básicos de relaciones:
  - **Generalización/especialización**: es la relación “es-un”.
    - Por ejemplo una vaca es un tipo de mamífero.
  - **Todo/parte**: es la relación “parte-de”.
    - Por ejemplo las orejas son parte del perro.

  - **Asociación**: denota alguna dependencia semántica entre clases que serían de otro modo independientes.
    - Por ejemplo, las vacas y las papas son claramente independientes, pero ambas representan cosas que pueden ser servidas en un plato en un restaurante.

Relaciones entre Clases (4)

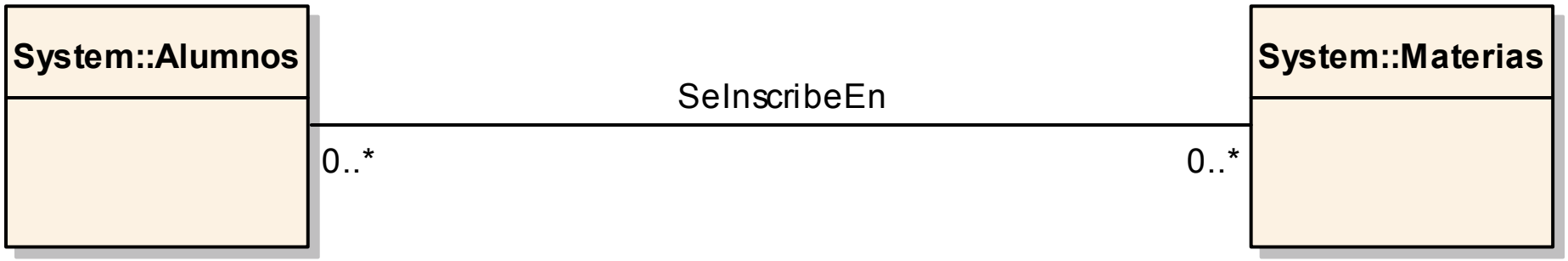
- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - Asociación
  - Herencia
  - Agregación
  - Uso
  - Instanciación
  - Metaclase

Relaciones entre Clases (4)

- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - **Asociación**
  - Herencia
  - Agregación
  - Uso
  - Instanciación
  - Metaclase

Asociación (1)

- Vamos a suponer un sistema de gestión de alumnos.
  - Existen abstracciones clave como son los alumnos y las materias.



Asociación (2)

- Como se muestra en la figura , existe una relación simple entre las clases:
  - Dado un alumno, se puede saber las materias en las que se ha inscripto, y dada una instancia de materia, deberíamos ser capaces de encontrar los alumnos inscriptos.
- Como sugiere este ejemplo, una asociación denota sólo una **dependencia semántica** y no establece:
  - La dirección de esta dependencia.
  - La forma exacta en que una clase se relaciona con otra.
- Esta semántica es suficiente durante el análisis del problema para identificar estas dependencias y para plasmar quiénes son los participantes de la relación y cuál es su **cardinalidad**.

Asociación (3)

- La cardinalidad denota la **multiplicidad** en la asociación.
- En la práctica, existen tres tipos habituales de cardinalidad en una asociación:
  - **Uno a uno**: una instancia de una clase se relaciona con solamente una instancia de la otra clase.
  - **Uno a muchos**: una instancia de una clase A se relaciona con varias instancias de la clase B, pero una instancia de la clase B sólo se relaciona con una única instancia de la clase A.
  - **Muchos a muchos**: una instancia de una clase se relaciona con varias instancias de la otra clase, y viceversa. Esta es la cardinalidad mostrada en la figura.

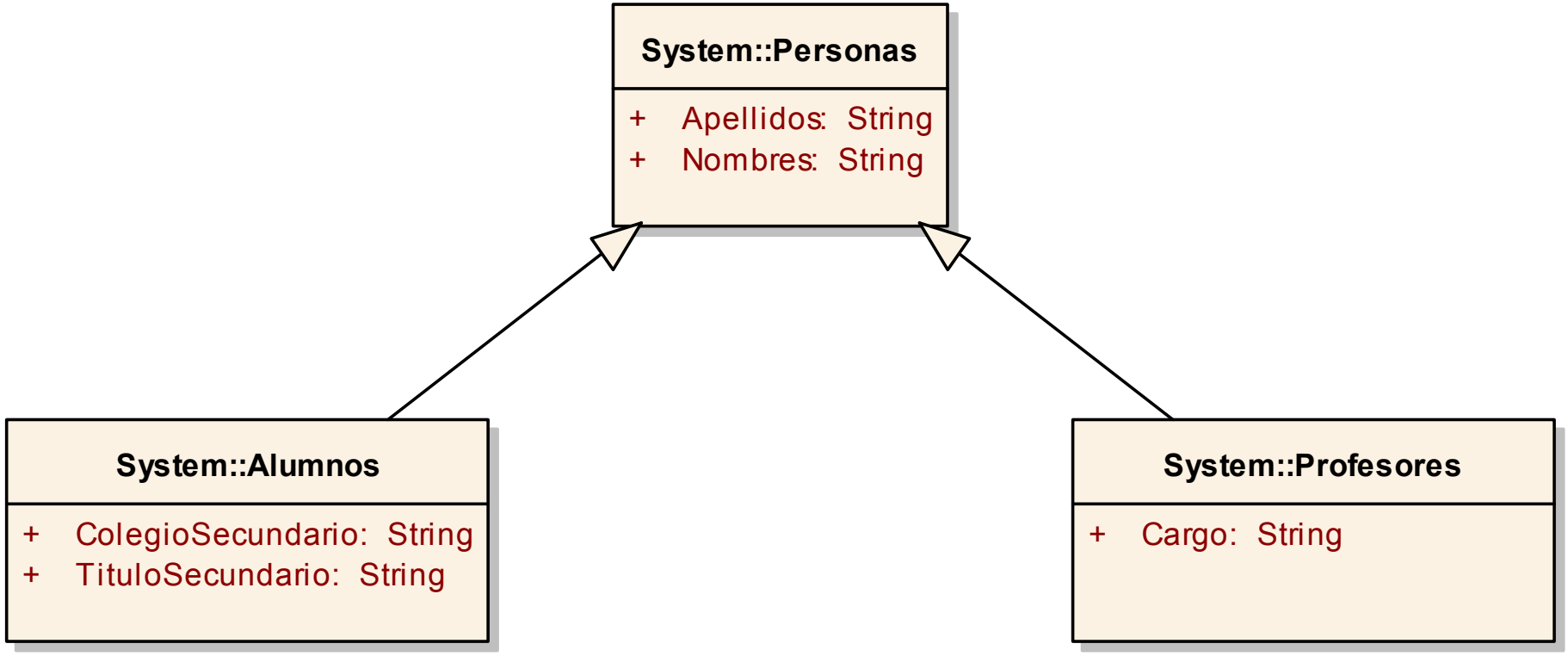
Relaciones entre Clases (4)

- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - Asociación
  - **Herencia**
  - Agregación
  - Uso
  - Instanciación
  - Metaclase

Herencia (1)

- Sistema de gestión de alumnos:
  - Existen abstracciones clave como son alumnos y profesores de las materias.
  - Estas personas comparten atributos y operaciones comunes: tienen nombre, apellido, domicilio, teléfono, etc.
  - Pero, particularmente, tienen propiedades y comportamientos diferentes.
  - Una forma conveniente de declarar estas clases es mediante la construcción de una jerarquía, donde las clases especializadas heredan la estructura y el comportamiento definido por la clase general.

Herencia (2)


System::Personas
+ Apellidos: String
+ Nombres: String
System::Alumnos
+ ColegioSecundario: String
+ TituloSecundario: String
System::Profesores
+ Cargo: String


Aves
Loro
Seres vivos que vuelan
Avestruz
Mariposa
Pato
Pingüino
Pez volador
Ballena
Seres vivos que nadan

Herencia (3)

- Como vemos en la figura:
  - Las clases especializadas Alumnos y Profesores heredan la estructura y el comportamiento de la clase generalizada Personas.
  - Aparte, le añaden nuevos atributos y pueden redefinir el comportamiento.

"La herencia es una relación entre clases en la que una clase comparte la estructura y/o comportamiento definidos en una (herencia simple) o más clases (herencia múltiple)."

- La clase de la que las otras heredan se denomina superclase.
- La clase que hereda de otra o más clases se denomina subclase.
  - Podemos hablar, entonces, de una jerarquía de tipos. Así, los Alumnos y Profesores son tipos de Personas.

Herencia (4)

- Una subclase habitualmente aumenta o restringe la estructura y comportamiento existentes en sus superclases.
  - Una subclase que aumenta sus superclases se dice que utiliza herencia por **extensión.**
  - Una subclase que restringe el comportamiento de sus superclases se dice que utiliza herencia por **restricción**.
- En la práctica, se espera que algunas clases tengan instancias y que otras no las tengan.
- Las clases que no tienen instancias, generalmente clases intermedias, se denominan **clases abstractas**.

- Una clase abstracta se modela para que sus subclases hereden su estructura y comportamiento, facilitando así la clasificación inteligente de las abstracciones.

Herencia (5)

- La clase más generalizada en una estructura de clases se llama **clase base**.
- Así, la estructura jerárquica de sistemas complejos suele ser "bosques de muchos árboles" de herencias.
- Existe una auténtica tensión entre la herencia y el encapsulamiento, ya que el uso de la herencia expone algunos secretos de una clase heredada.
- Como las subclases heredan el comportamiento de sus superclases, pueden redefinir los métodos heredados dando lugar al **polimorfismo**.

Herencia (6)

- Ejemplo anterior:
  - Supongamos que definimos la operación Ingresar en la clase personas. Dicha operación permite ingresar a la persona en la facultad.
  - Si es un alumno, se le asigna carrera e inscripciones en las materias del primer año.
  - Si es un profesor, se le asignan las materias que enseña y el instituto al que pertenece.

- El **polimorfismo** es un concepto en el que dado un parámetro que puede denotar instancias de varias clases diferentes, pero que están relacionadas por una clase en común, es capaz de responder a algún conjunto común de operaciones de maneras diferentes.

Herencia (7)

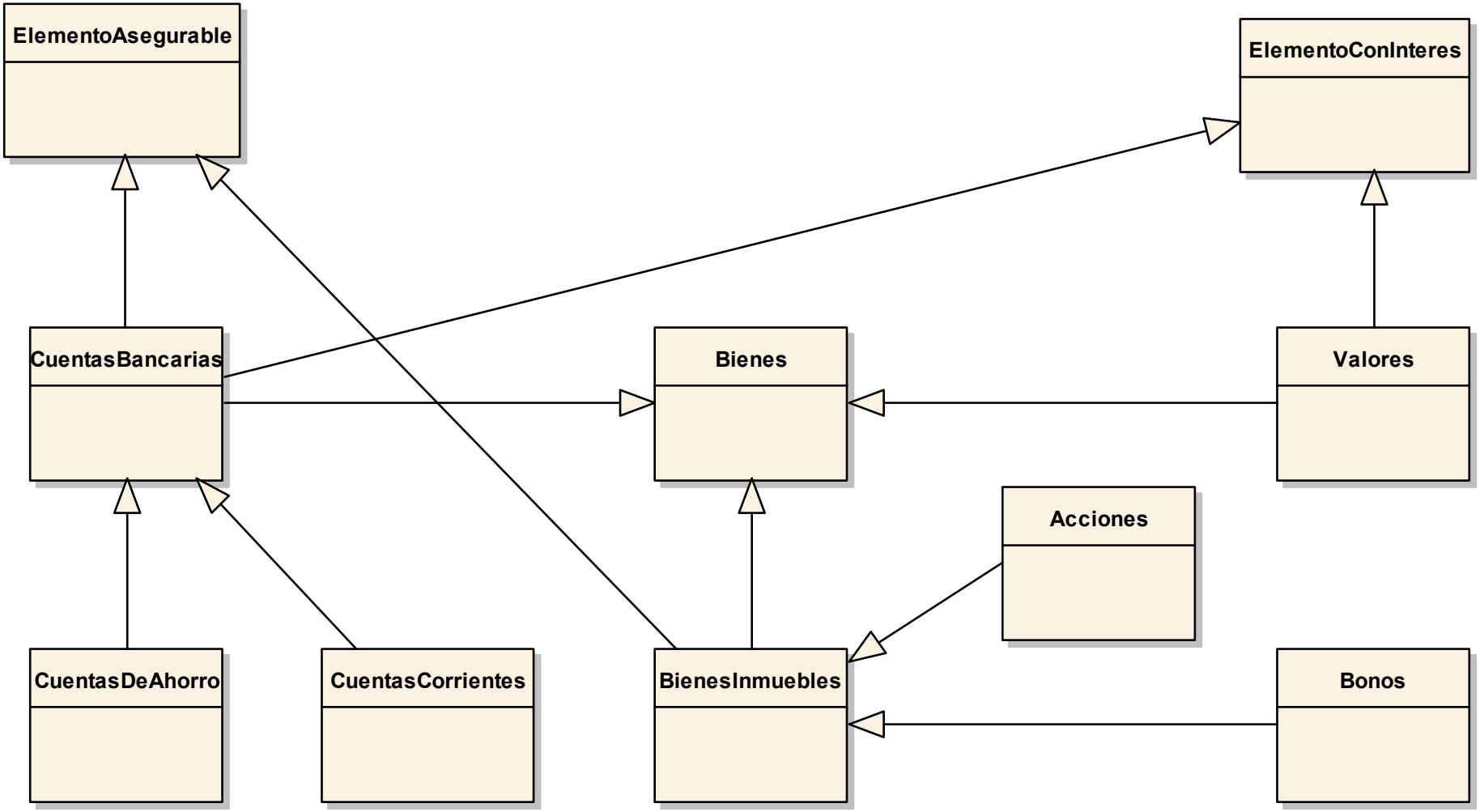
- Así, al definir una instancia p de Profesores y una instancia a de Alumnos, la operación ingresar se comporta de manera diferente a pesar de ser un comportamiento heredado de la superclase.
- La implementación de la operación ingresar se redefine en cada subclase.
- Sin polimorfismo, el desarrollador acaba por escribir código que consiste en sentencias Case .
- Con polimorfismo no son necesarias ya que cada objeto conoce implícitamente su tipo.

Herencia (8)

Herencia múltiple

- En la herencia múltiple, una subclase tiene varias superclases de las cuales hereda.
  - Consideremos cómo podrían organizarse varios bienes como cuentas de ahorro, inmuebles, acciones y bonos.
  - Las cuentas de ahorros y las cuentas corrientes son los bienes que maneja un banco, de ese modo podríamos clasificar a ambos como cuentas bancarias.
  - Las cuentas bancarias y los bienes inmuebles son elementos asegurables mientras que las acciones, los bonos y las cuentas bancarias pueden arrojar un dividendo o interés.

Herencia (9)


ElementoAsegurable
ElementoConInteres
CuentasBancarias
Bienes
Valores
Acciones
CuentasDeAhorro
CuentasCorrientes
BienesInmuebles
Bonos

Herencia (10)

- Cuando se tiene herencia múltiple aparecen dos problemas:
  - **Colisiones de nombres:**
    - Aparecen cuando dos o más superclases diferentes utilizan el mismo nombre para algún elemento de sus interfaces, como las variables de instancia o los métodos.
    - Introducen ambigüedad en el comportamiento de la subclase que hereda de forma múltiple.
    - Para resolver este tipo de desacuerdo, la semántica del lenguaje o la herramienta de diseño puede interpretarlo como algo incorrecto.

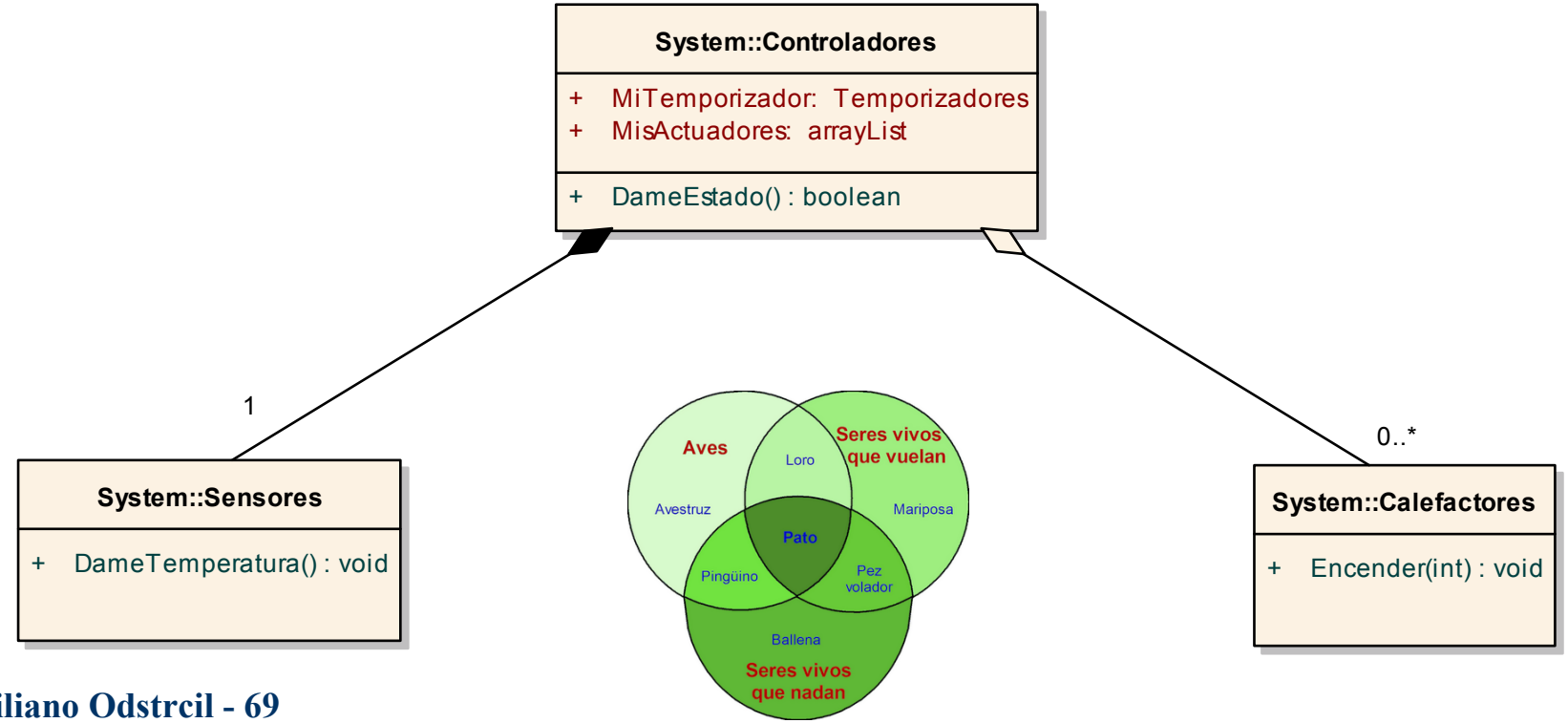
  - **Herencia repetida:**
    - Aparece cuando una clase es un antecesor de otra por más de una vía.
    - Por ejemplo, una clase D que hereda de las clases B y C, cada una de las cuales hereda de una clase A.
    - Para tratar el problema se puede tratar la herencia repetida como algo incorrecto, o bien se puede permitir la duplicación de superclases pero tratando las referencias como si vinieran de la misma clase (mediante operaciones sobre el grafo de clases).

Relaciones entre Clases (4)

- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - Asociación
  - Herencia
  - **Agregación**
  - Uso
  - Instanciación
  - Metaclase

Agregación (1)

- Las relaciones de agregación entre clases tienen un paralelismo directo con las relaciones de agregación entre los objetos correspondientes a esas clases.
  - Supongamos el caso de un controlador de temperatura de un vivero.
  - Existen las abstracciones Controlador, Calentador y Sensor.



Agregación (2)

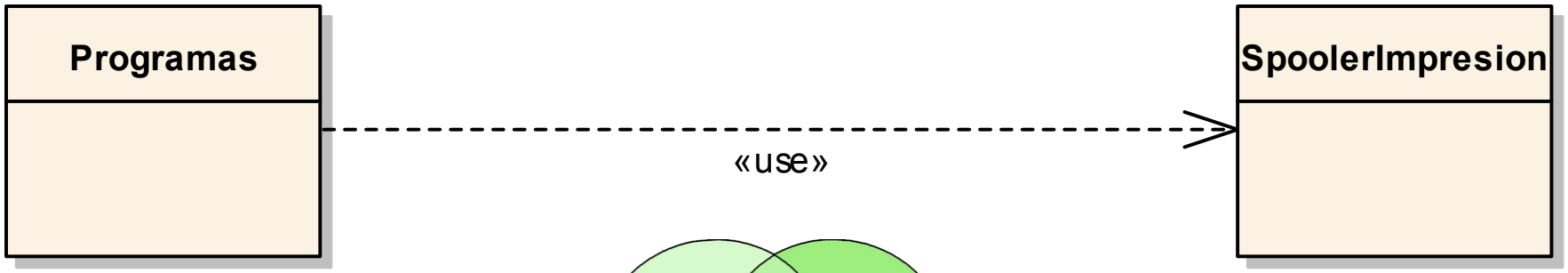
- Como se muestra en la figura:
  - La clase Controladores es el todo mientras que las clases Calentadores y Sensores son las partes.
  - Hay una **contención por valor** (o composición), es decir un tipo de contención física que indica que el objeto Sensor no existe independientemente de la instancia Controlador que lo contiene.
  - La agregación con Calentadores es una **contención por referencia** o agregación propiamente dicha.
  - En este caso, hay cierta dependencia que obliga a crear y destruir instancias de cada clase independientemente.
- Las agregaciones presentan también una cardinalidad que puede tener los valores uno y muchos.

Relaciones entre Clases (4)

- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - Asociación
  - Herencia
  - Agregación
  - **Uso**
  - Instanciación
  - Metaclase

Uso (1)

- Las relaciones de uso entre clases corren paralelas a los enlaces hermano a hermano entre las instancias correspondientes a esas clases.
  - Por ejemplo, en un sistema operativo, los programas necesitan el spooler de impresión para poder realzar sus impresiones.



Uso (2)

- Mientras que una asociación denota una conexión semántica bidireccional entre las clases, una relación de uso es un posible **refinamiento** de una asociación, por lo que se establece qué abstracción es el **cliente** y qué abstracción es el **servidor**.
- En nuestro ejemplo, la abstracción Programas es el cliente, mientras que la abstracción Spooler de impresión es el servidor que proporciona los servicios.

Relaciones entre Clases (4)

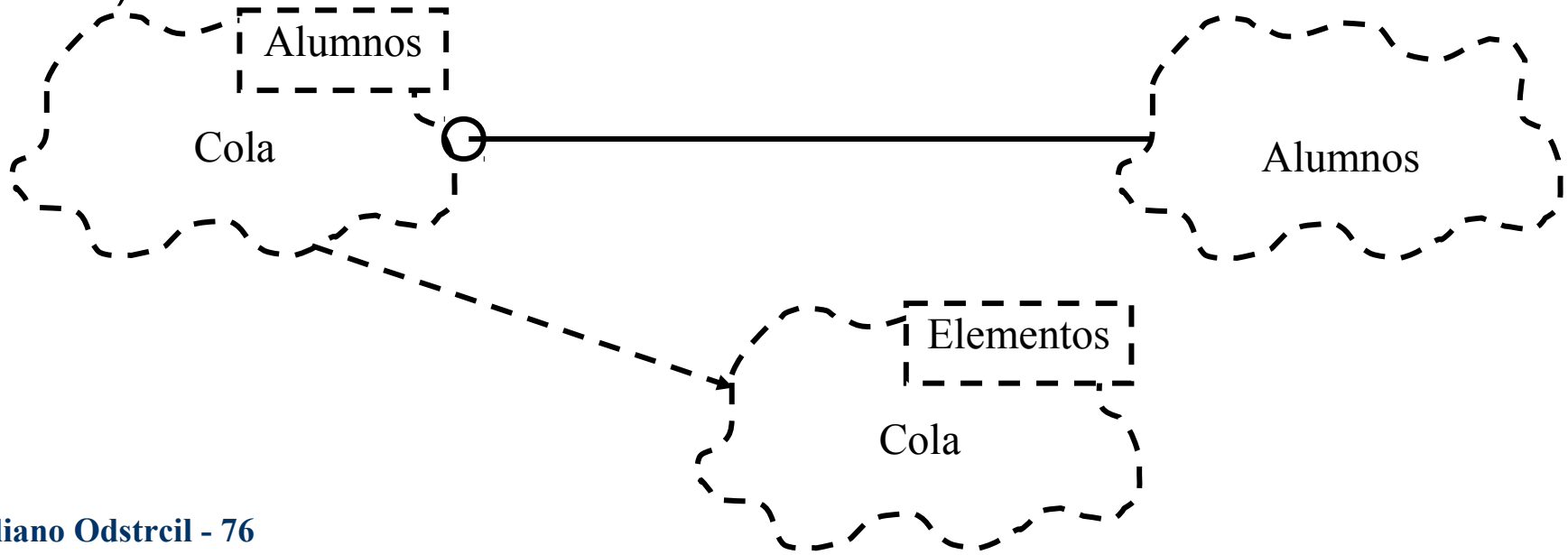
- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - Asociación
  - Herencia
  - Agregación
  - Uso
  - **Instanciación**
  - Metaclase

Instanciación (1)

- También llamada creación de instancias, esta es una relación que se utiliza principalmente para la creación de clases parametrizadas.
- Una **clase parametrizada** denota una familia de clases cuya estructura y comportamiento están definidos independientemente de sus parámetros de clase formales.
- Hay que hacer corresponder estos parámetros formales con parámetros actuales, proceso que se conoce como **instanciación**, para formar una clase concreta de esa familia.

Instanciación (2)

- Ejemplo:
  - Supongamos que se quiere definir la clase parametrizada Cola.
  - Esta clase denota la familia de clases de estructura y comportamiento de una cola (FIFO) de elementos.
  - Para formar una clase concreta de la familia de las colas, se procede a instanciarla.
  - En este caso se va a declarar una cola de alumnos para el sistema de alumnos (Booch).

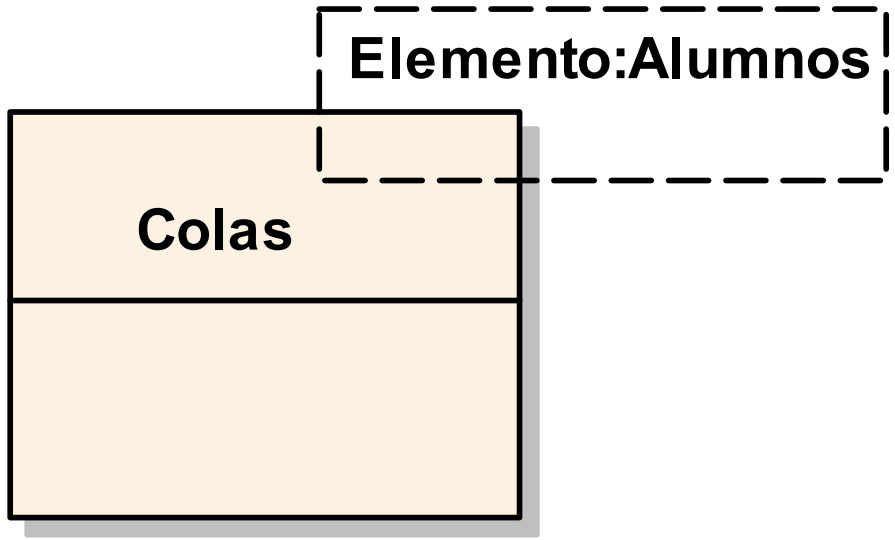


Instanciación (3)

- En la figura observamos:
  - La clase parametrizada Cola de Elementos.
  - La instancia es la clase Cola de Alumnos, que usa a la clase alumnos.
  - Una clase parametrizada no puede tener instancias a menos que antes se la instancie (valga la redundancia).
- Las clases parametrizadas tienen las siguientes características:
  - Las instanciaciones son seguras respecto a tipos.
  - Una clase parametrizada sirve como modelo para otras clases.
  - Las relaciones de instanciación requieren casi siempre alguna relación de uso.

Instanciación (4)

- Esta relación que propone Booch es reemplazada en UML 2 por la clase parametrizada con Object como clase parámetro, como se muestra en el ejemplo.

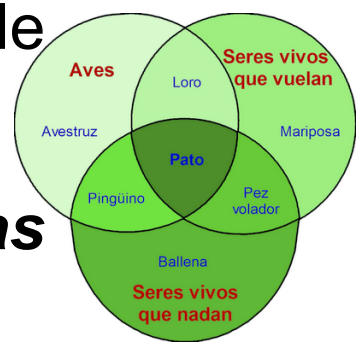


Relaciones entre Clases (4)

- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - Asociación
  - Herencia
  - Agregación
  - Uso
  - Instanciación
  - **Metaclase**

Metaclase (1)

- Se ha dicho que todo objeto es una instancia de una clase. ¿Qué ocurre si se trata a la propia clase como un objeto que puede manipularse?.

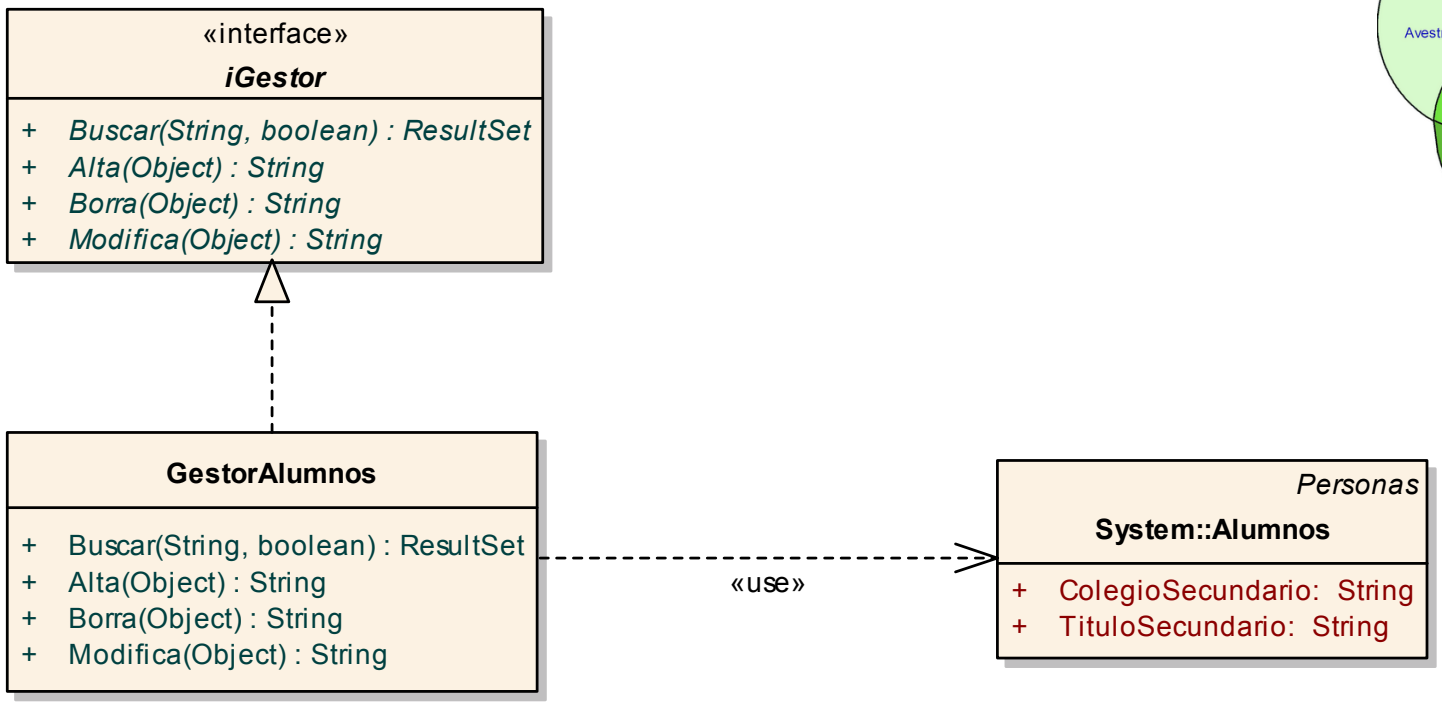


" Una metaclase es una clase cuyas instancias son, ellas mismas, clases."

- En un sistema bajo desarrollo, una clase proporciona al desarrollador una interfaz para comunicarse con la definición de objetos.
- De la misma manera, es extremadamente útil para las clases ser objetos que pueden ser manipulados de la misma forma que todas las demás descripciones.

Metaclase (2)

- El propósito general de una metaclase es proporcionar variables de clase (compartidas por todas las instancias de la clase) y operaciones para inicializar variables de clase y crear instancias simples de la metaclase. En nuestro sistema de gestión de alumnos:

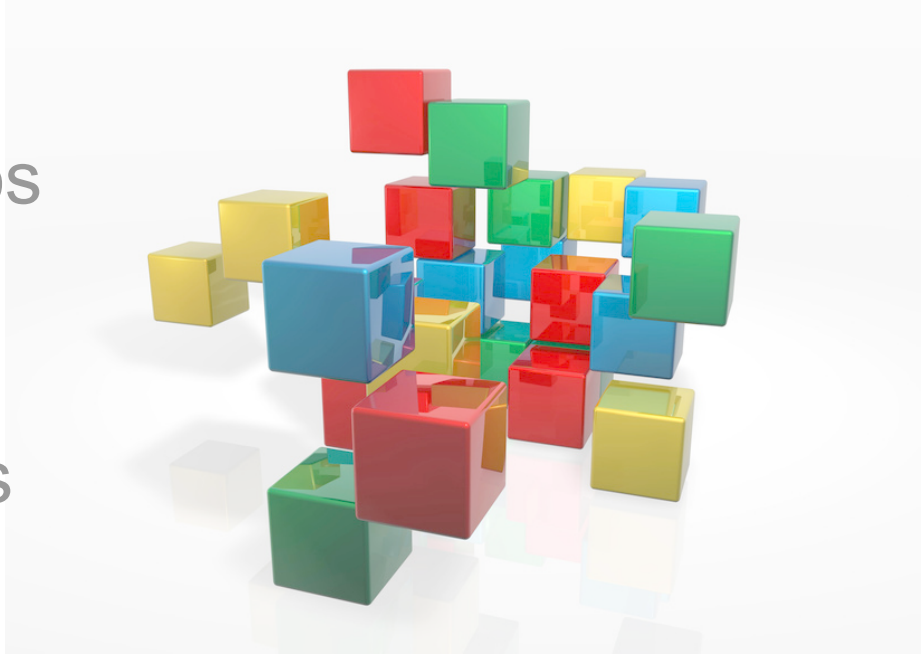


Relaciones entre Clases (4)

- Han evolucionado varios enfoques comunes para plasmar los tres tipos de relaciones anteriores. La mayoría de los lenguajes orientados a objetos ofrecen soporte para las siguientes relaciones:
  - Asociación
  - Herencia
  - Agregación
  - Uso
  - Instanciación
  - Metaclase

El Paradigma Objetos

- Contenido
  - La Naturaleza de los Objetos
  - Relaciones entre Objetos
  - La Naturaleza de las Clases
  - Relaciones entre Clases
  - **La Interacción entre Clases y Objetos**



La Interacción entre clases y objetos (1)

- Las clases y objetos son conceptos separados pero en íntima relación:
  - Todo objeto es instancia de alguna clase
  - Toda clase tiene cero o más instancias.

- Para la gran mayoría de las aplicaciones, las clases son estáticas:
  - Su existencia, semántica y significado están fijados antes de la ejecución de un programa.
- Análogamente, la clase de la mayoría de los objetos es estática, lo que significa que cuando se instancia su objeto, su clase está fijada.
- Sin embargo, los objetos se crean y destruyen típicamente a un ritmo trepidante durante el tiempo de vida de una aplicación.

La Interacción entre clases y objetos (2)

- Por ejemplo, el caso de la aplicación de gestión de alumnos:
  - Existen abstracciones importantes como son las carreras, las currículas de cada carrera, las materias, los alumnos y profesores involucrados.
  - Estas clases son relativamente estáticas, ya que si no lo fueran no podría construirse una aplicación que contuviese el conocimiento de hechos tales como:
    - Los profesores dictan materias.
    - Las materias forman parte de currículas de una carrera determinada.
    - Los alumnos se inscriben en el cursado y rinden exámenes de dichas materias.
  - Por el contrario, las instancias de estas clases son dinámicas:
    - Las currículas agregan unas materias y dan de baja otras.
    - Los profesores que la dictan cambian.
    - Ingresan alumnos nuevos y se gradúan o dan de baja otros, se agregan carreras nuevas y desaparecen otras, etc.

La Interacción entre clases y objetos (3)

- Durante el análisis , el desarrollador tiene dos tareas principales:
  - Identificar las clases y objetos que forman el vocabulario del dominio del problema. En nuestro ejemplo: materias, carreras, alumnos, correlativas, etc.
  - Idear las estructuras por las que conjuntos de objetos trabajan juntos para lograr los comportamientos que satisfacen los requerimientos del problema.
- Se llaman a estas clases y objetos las **abstracciones clave** del problema.
- Se denomina a esas estructuras cooperativas los mecanismos de la implantación.
- Durante estas fases del desarrollo, el interés principal debe estar en la vista externa de estas abstracciones clave y mecanismos.

La Interacción entre clases y objetos (4)

- Esta vista representa el marco de referencia lógico del sistema:
  - Estructura de clases y estructura de objetos del mismo.
- En las etapas finales del diseño, el centro de atención debe estar en la vista interna de estas abstracciones clave y mecanismos, involucrando a su representación física.
- Este enfoque es fundamental a la hora de dominar la complejidad.

El Paradigma Objetos

- Contenido
  - La Naturaleza de los Objetos
  - Relaciones entre Objetos
  - La Naturaleza de las Clases
  - Relaciones entre Clases
  - La Interacción entre Clases y Objetos