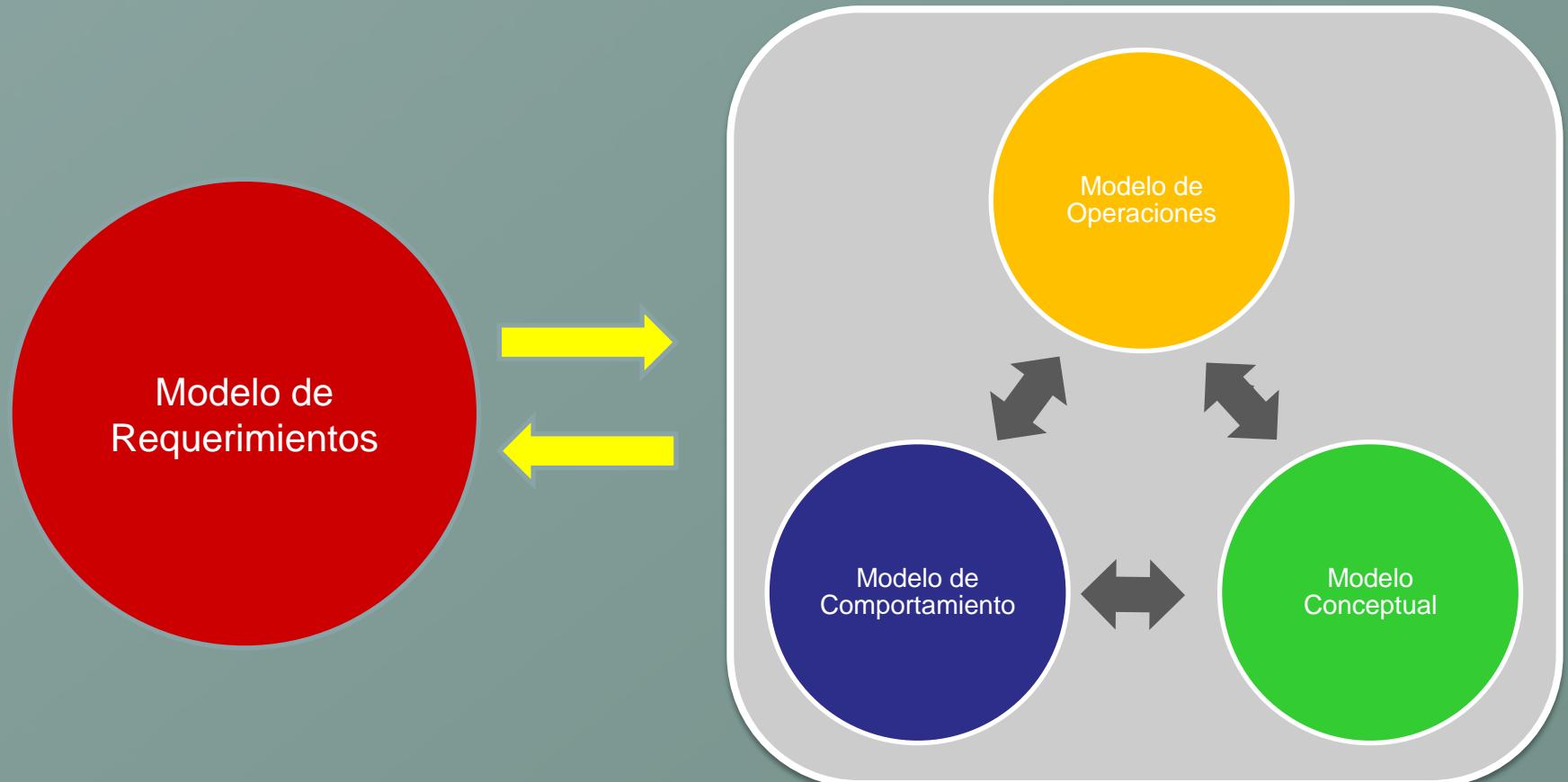


Modelo Conceptual

Diagrama de Clases

Los modelos se complementan



Especificación

Modelo Conceptual

- Explica cuales son y como se relacionan los conceptos relevantes en la descripción del problema
- Existen muchas variantes, con distintos grados de sofisticación, para describir el modelo conceptual.
 - Diccionario/Glosario
 - Diagrama de Entidad Relación
 - Diagrama de Clases
- También conocido como modelo de dominio.

¿Conceptos Relevantes?

- ¿Cuales son?
- ¿Cómo se relacionan?

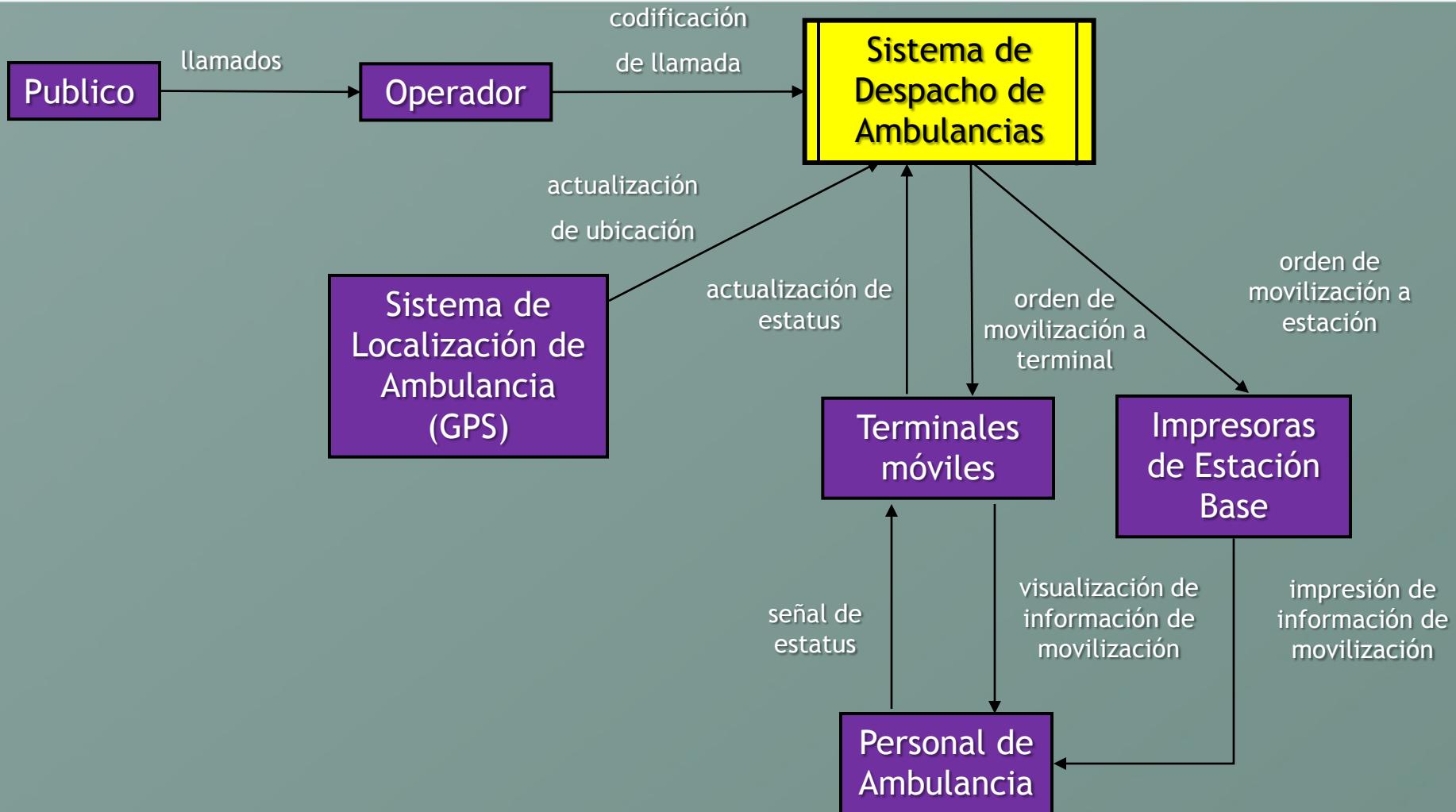
Objetivo Lograr [Primera Intervención de Ambulancia]

Categoría Seguridad

Definición Para cada llamada urgente reportando un accidente, una primer ambulancia deberá arribar al lugar del incidente dentro de los 8 minutos para incidentes de categoría A (peligro de vida inminente) y dentro de los 14 minutos para incidentes de categoría B.

DefFormal $(\forall l: \text{LlamadaUrg}, inc:\text{Incidente}) (\text{Reporte}(l, inc) \Rightarrow$
 $inc.\text{Categoría} = 'A' \rightarrow \langle\rangle \leq 8\text{min } (\exists a:\text{Ambulancia}) \text{Intervención}(a,$
 $inc))$ $inc.\text{Categoría} = 'B' \rightarrow \langle\rangle \leq 14\text{min } (\exists a:\text{Ambulancia}) \text{Intervención}(a,$
 $inc))$

¿Conceptos Relevantes?



¿Conceptos Relevantes?

- ¿Cuales son?
- ¿Cómo se relacionan?

Operación: PlanificarReunión

Usuarios: Iniciador de reunión

Responsable: Software

Def: Fija fecha de reunión a partir de las restricciones informadas por invitados

Entrada: r: reunión

Salida: r:reunión

Pre: Las restricciones de cada invitado han sido informadas (G223)

Post: La fecha de reunión no está dentro de las fechas excluidas por cada invitado (G223)

¿Conceptos Relevantes?

Caso de Uso: Ingresando Orden de Compra

Actor: Vendedor

PRE: Vendedor autenticado

POST: Orden de compra registrada

1. El vendedor ingresa el número de cliente en el sistema.
2. El sistema muestra información básica sobre el cliente.
3. El vendedor ingresa el código del producto que el cliente quiere comprar, informando su cantidad.
4. El sistema muestra información del producto solicitado, y confirma su disponibilidad.
5. Se repite el paso 3 hasta que el cliente no solicita más productos.
6. El sistema notifica que la orden de compra ha sido registrada.
7. Fin del caso de uso.

Objetos y Clases

- *Objeto Conceptual*: Denota una entidad o concepto del dominio del problema
 - *Obj. pasivos*: El alfajor en mi mochila, remito X33442 de Officenet
 - *Obj. activos*: Camión “BEG 232”, Caja registradora #1 del super de la equina de casa
 - *Personas*: Perico Perez, Susana Gimenez, Javier Altauz
 - *Estructuras*: dc.uba.ar
 - etc..
- *Clase Conceptual*: Denota un conjunto de objetos conceptuales que comparten características comunes. Estas características pueden ser atributos o relaciones.
 - Producto, Remito, Camión, Empleado, Caja Registradora, Departamento
- Para simplificar hablaremos de clases y objetos en vez de clases conceptuales y objetos conceptuales. Pero ojo, no confundir con su significado en programación orientada a objetos. NO ES LO MISMO!

Un poquito de POO

- **¿Qué es un objeto?**

Un objeto es un componente de software que contiene variables y métodos y que es usado para modelar algún aspecto de la “vida real”. Es una abstracción de la realidad.

- **¿Qué es una clase?**

Una clase es un plano o prototipo que define las variables y los métodos comunes a todos los objetos de un cierto tipo.

Un poquito de POO

Los **objetos** son representaciones
(simples/complejas)
(reales/imaginarias) de cosas: reloj, avión
empleado, etc.

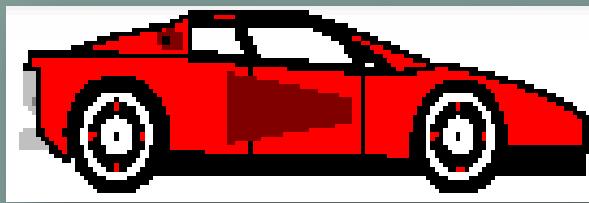


Un poquito de POO

Abstracción funcional

Hay cosas que sabemos
que los coches hacen
pero no como lo hacen:

- avanzar
- parar
- girar a la dcha
- girar a la izda



Abstracción de datos

Un coche tiene además
ciertos atributos:

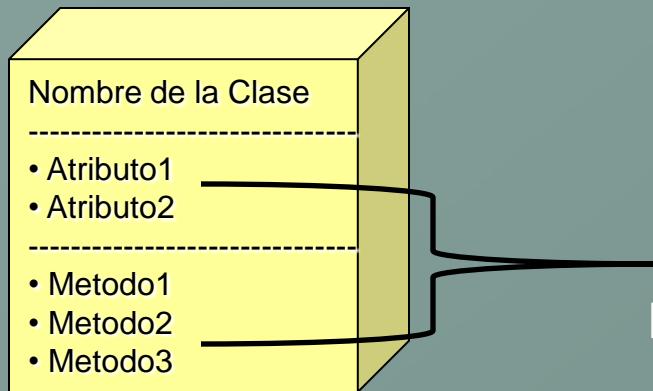
- color
- velocidad
- tamaño
- etc...

Un poquito de POO

Los objetos encapsulan **variables** permitiendo acceso a ellas únicamente a través de los **métodos**

Variables: *Contenedores de valores*

Métodos: *Contenedores de funciones*



Se puede permitir o restringir su acceso desde "afuera"

Pueden ser **Públicos** o **Privados**

Estado: representado por el contenido de sus variables

Comportamiento: definido por sus métodos

Objeto = Identidad + Estado + Comportamiento

Un poquito de POO: Identidad

- Oid (Object Identifier)

Cada objeto posee un oid. El oid establece la identidad del objeto y tiene las siguientes características:

- Constituye un identificador único y global para cada objeto dentro del sistema.
- Es determinado en el momento de la creación del objeto.
- Es independiente de la localización física del objeto, es decir, provee completa independencia de localización.
- Es independiente de las propiedades del objeto, lo cual implica independencia de valor y de estructura.
- No cambia durante toda la vida del objeto. Además, un oid no se reutiliza aunque el objeto deje de existir.
- No se tiene ningún control sobre los oids y su manipulación resulta transparente.

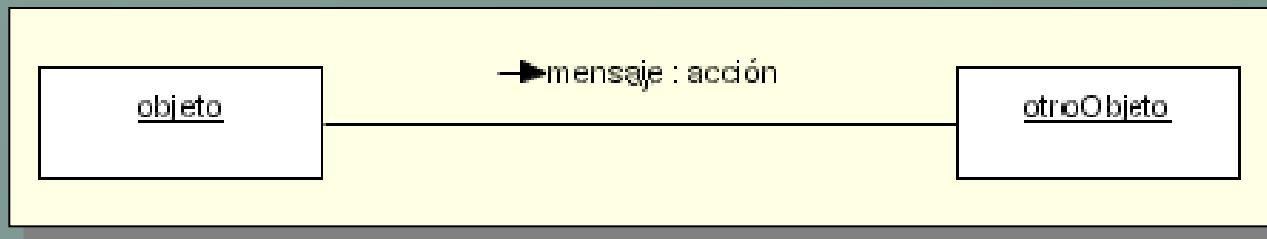
- Sin embargo, es preciso contar con algún medio para hacer referencia a un objeto utilizando referencias del dominio (valores de atributos).

Un poquito de POO: Estado

- El estado evoluciona con el tiempo.
- Algunos atributos pueden ser constantes.
- El comportamiento agrupa las competencias de un objeto y describe las acciones y reacciones de ese objeto.
- Las operaciones de un objeto son consecuencia de un estímulo externo representado como mensaje enviado desde otro objeto.

Un poquito de POO: Comportamiento

- Los mensajes navegan por los enlaces, a priori en ambas direcciones.
- La unidad de comunicación entre objetos se llama mensaje.
- Estado y comportamiento están relacionados.
- Ejemplo: no es posible aterrizar un avión si no está volando. Está volando como consecuencia de haber despegado del suelo.
- Un estímulo causará la invocación de una operación, la creación o destrucción de un objeto o la aparición de una señal.
- Un mensaje es la especificación de un estímulo.



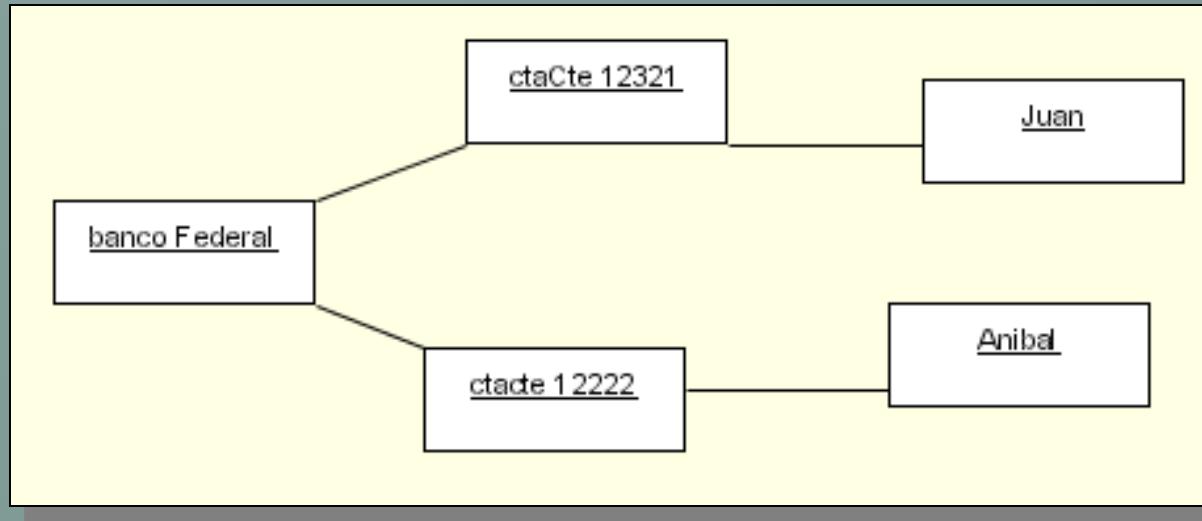
Un poquito de POO: Objetos

- En UML, un objeto se representa por un rectángulo con un nombre subrayado.



Un poquito de POO: Objetos

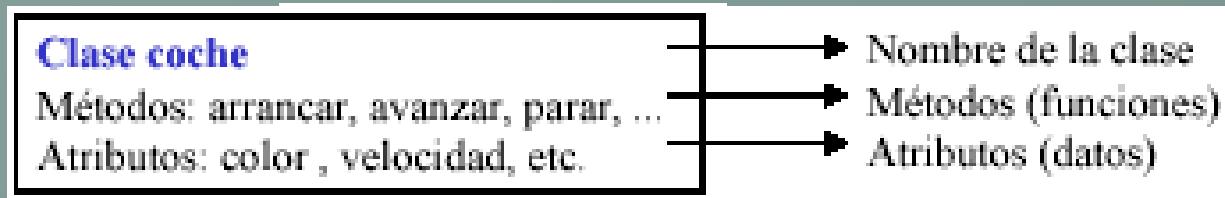
- Ejemplo de varios objetos relacionados:



Un poquito de POO: Objetos vs. Clases

Una **clase** es una entidad abstracta

- Es un tipo de clasificación de cosas
- Define el comportamiento y atributos de un grupo de estructura y comportamientos similar



Un **objeto** es una instancia o variable de una clase

- Un objeto se distingue de otros miembros de la clase por sus atributos.

Objeto Ferrari
Perteneciente a
la clase coche



Nombre: Ferrari
Métodos: arrancar, avanzar, parar, ...
Atributos: color = "rojo";
velocidad 300Km/h

Atributos y Enlaces

- *Atributo*: Es una característica intrínseca de un objeto, es independiente de otros objetos. Tiene un nombre y un rango posible de valores.
 - Ej. # de Empleado, Nombre, Fecha de Nacimiento
 - En cada instante de tiempo, cada atributo de un objeto tiene un valor único.
 - Atributos pueden ser mutables o inmutables y describen el “estado” del objeto.
- Un enlace o “link” es una característica que vincula conceptualmente a varias objetos. Cada objeto juega un rol conceptual en ese vínculo.
 - Ej. Pedro es el conductor del camión “BEG 232”

Técnicas de Modelado

- **Diccionario/Glosario**
 - Lista de clases con sus atributos y relaciones
 - Poca estructura / Difícil de analizar
 - Lenguaje natural permite explicar denotación en detalle
- **Diagrama entidad relación (DER)**
 - Lenguaje gráfico que introduce estructura
 - Utilizado para el diseño de bases de datos
 - Cubierto por materia BD.
- **Diagrama de clases**
 - Extiende DER con varias características
 - Herencia
 - Modificadores
 - ...

Clases Conceptuales: De dónde vienen?

- Estrategias para identificar clases
 - Identificar frases nominales en descripciones del dominio.
 - Frase nominal: Un sustantivo o un conjunto de palabras que actúan como tal
 - Utilizar lista de categorías de clases conceptuales.

Categorías de Clases Conceptuales

Categoría	Ejemplo
Objetos tangibles o físicos	Casa, Avión
Especificaciones, diseños o descripciones de las cosas	PlanoDeLaCasa, EspecificaciónDelProducto, DescripciónDelVuelo
Lugares	Tienda, Aula
Transacciones	Venta, Pago, Reserva, Transferencia
Líneas de la transacción	LíneaDeVenta
Roles de la gente	Cajero, Piloto, Jefe
Contenedores de otras cosas	Aula, Círculo, Lata, Mochila
Contenidos	Pasajero, Artículo, ÚtilEscolar
Otros sistemas informáticos o electromecánicos externos al sistema	SistemaDeAutorización, ControlDeTraficoAéreo
Conceptos abstractos	Amor, Celos, Ansiedad, Acrofobia
Organizaciones	DepartamentoDeVentas, CompañíaAérea
Hechos	Reunión, Vuelo, Aterrizaje, Venta, Pago
Procesos (normalmente no se representan como conceptos, pero podría ocurrir)	VentaDeUnProducto, ReservaDeUnAsiento (no confundir con trasacciones)
Reglas y políticas	PolíticaDeReintegro, PolíticaDeCancelación
Catálogos	CatálogoDeProductos
Registros de finanzas, trabajo, contratos, cuestiones legales	Recibo, Remito, Factura, ContratoDeEmpleo, Expediente
Instrumentos y servicios financieros	LíneaDeCrédito, Stock
Manuales, documentos, artículos de referencia, libros	ManualDeReparaciones, ListaDeCambios
Relaciones	Amistad, Parentesco (podría estar en conceptos abstractos pero es bueno destacarlo ya que las relaciones no suelen ser consideradas al modelar)

Diagrama de Clases

- El propósito de este diagrama es el de representar los objetos fundamentales del sistema, es decir los que percibe el usuario y con los que espera tratar para completar su tarea en vez de objetos del sistema o de un modelo de programación.
- La clase define el ámbito de definición de un conjunto de objetos.
- Cada objeto pertenece a una clase.
- Los objetos se crean por instanciación de las clases.

Diagrama de Clases

- Cada clase se representa en un rectángulo con tres compartimientos:

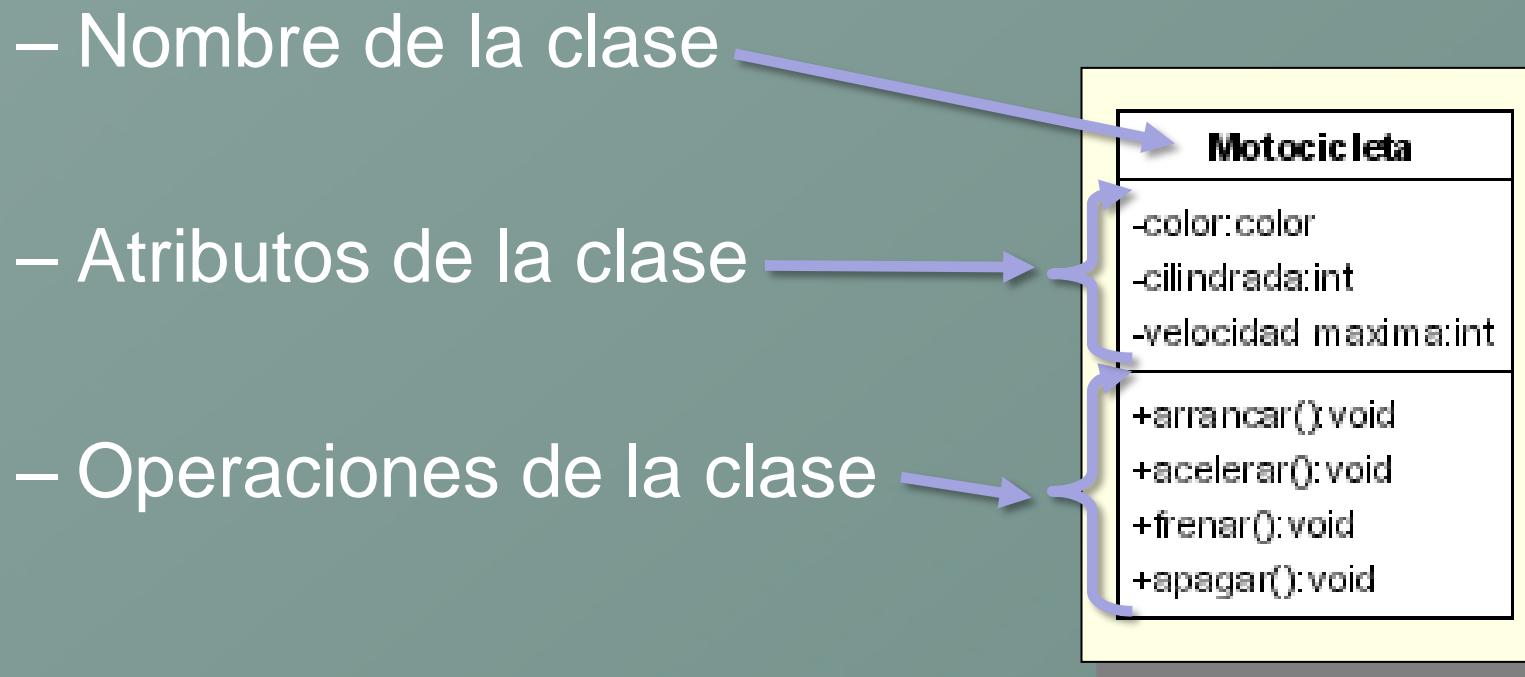


Diagrama de Clases (Conceptuales)

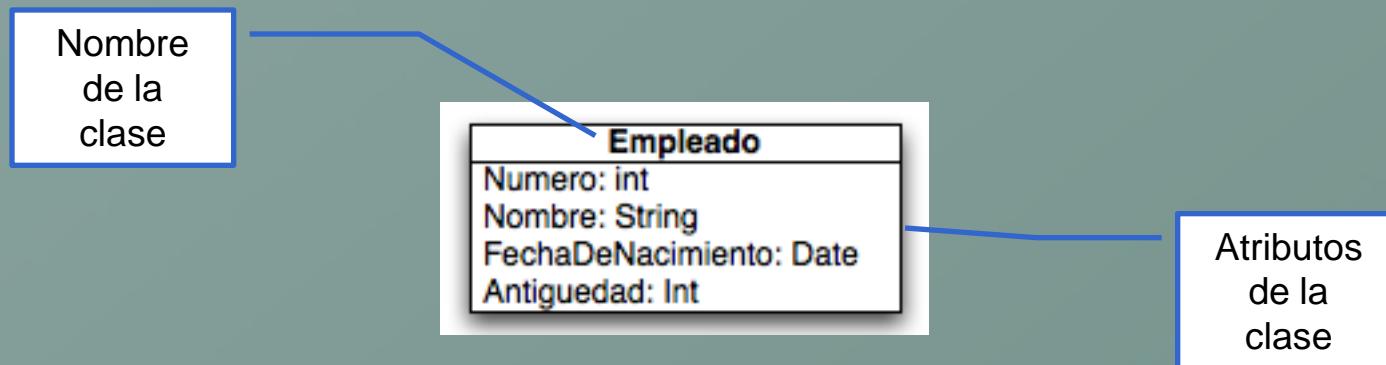


Diagrama de Clases: Atributos

- **Tipo:** puede llegar a depender del lenguaje de programación a utilizar.
- **Valor inicial:** valor que poseerá el atributo al crear un objeto.
- **Visibilidad:** está relacionado con el encapsulamiento.
- **Multiplicidad:** determinar si un atributo debe estar o no, y si posee un único valor o una lista de valores.
- **Ordenamiento:** especifica si el atributo determina alguna relación de orden dentro de la clase.
- **Capacidad de cambio:** permite definir atributos con valores constantes.
- **Modificadores:** un atributo puede ser de clase, derivado, volátil, transitorio.

El atributo fecha de nacimiento es público.

El atributo edad es derivado (puede calcularse a partir de la fecha de nacimiento), y determina una relación de orden entre las instancias de las personas.

El atributo DNI es un atributo protegido.

El atributo coloresPreferidos representa una colección o conjunto de valores del tipo Color

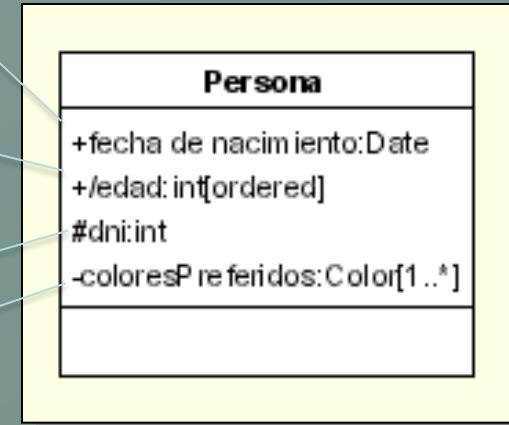


Diagrama de Clases (Conceptuales)

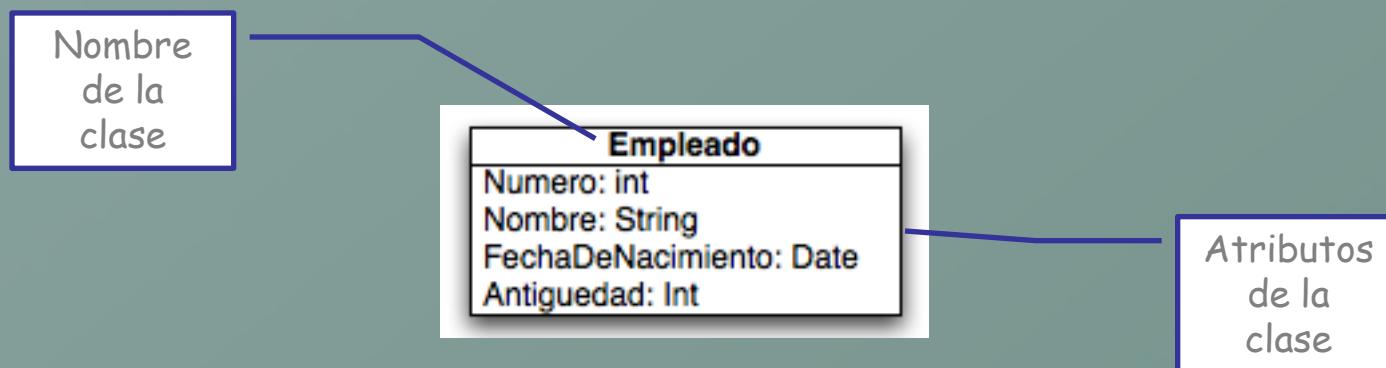


Diagrama de Clases: Atributos

Visibilidad

La encapsulamiento presenta tres ventajas básicas:

- Se protegen los datos de accesos indebidos
- El acoplamiento entre las clases se disminuye
- Favorece la modularidad y el mantenimiento

Los atributos de una clase no deberían ser manipulables directamente por el resto de objetos.

Niveles de encapsulamiento:

- (-) **Privado** : es el más fuerte. Esta parte es totalmente invisible desde fuera de la clase (excepto para clases friends en terminología C++).
- (~) **Package** : Sólo es visible dentro del mismo package.
- (#) Los atributos/operaciones **protegidos** están visibles para las clases friends y para las clases derivadas de la original.
- (+) Los atributos/operaciones **públicos** son visibles a otras clases (cuando se trata de atributos se está transgrediendo el principio de encapsulamiento).

Diagrama de Clases: Atributos

Visibilidad

La encapsulamiento presenta tres ventajas básicas:

- Se protegen los datos de accesos indebidos
- El acoplamiento entre las clases se disminuye
- Favorece la modularidad y el mantenimiento

Los atribu

Niveles

(-) Priv

fuer

(~) Pac

(#) Los a

riend

(+) Los a

EN CONTEXTO DEL MODELO CONCEPTUAL ESTO NO

TIENE SENTIDO/USO/ETC

se trata de atributos se está transgrediendo el principio de encapsulamiento).

Diagrama de Clases

Relaciones entre Clases

- Una asociación es una conexión estructural simple entre clases. Las instancias de las clases implicadas en una asociación estarán probablemente comunicándose en el momento de ejecución.
- Los enlaces entre de objetos pueden representarse entre las respectivas clases
- Formas de relación entre clases:
 - Asociación y Agregación (vista como un caso particular de asociación)
 - Generalización/Especialización

Diagrama de Clases: Asociación

- La asociación expresa una conexión bidireccional entre objetos.
- Una asociación es una abstracción de la relación existente en los enlaces entre los objetos.

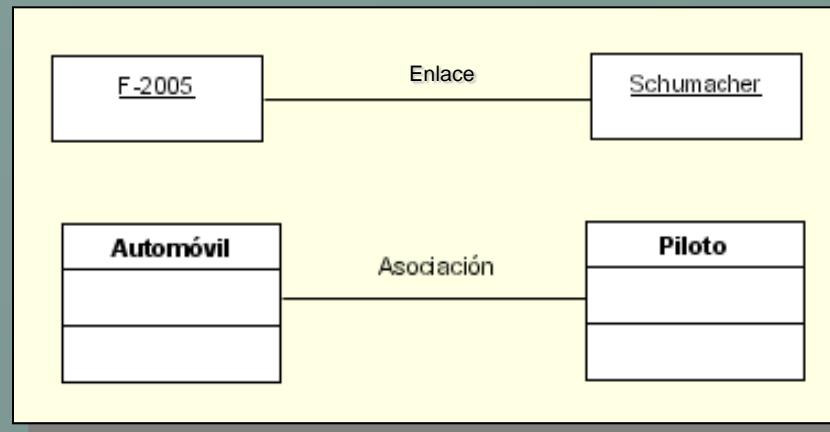
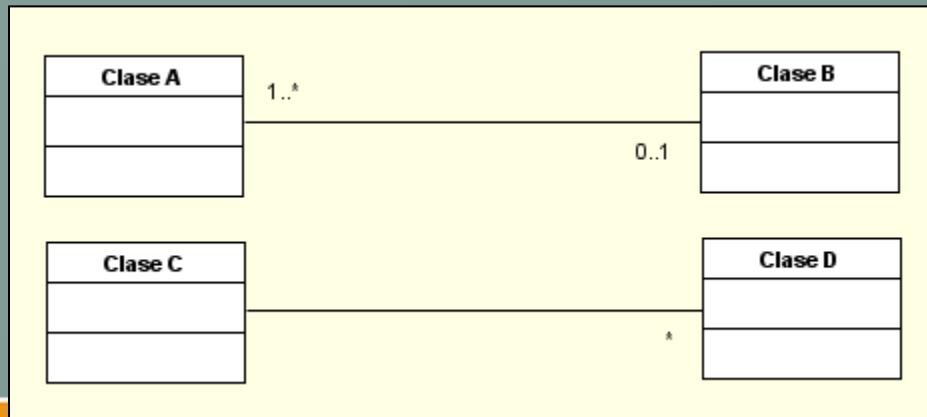


Diagrama de Clases: Atributos

Multiplicidad

- 1 El atributo debe tener un único valor.
- 0..1 El atributo puede o no tener un valor.
- 0..* El atributo puede tener varios valores o ninguno.
- 1..* El atributo puede tener varios valores, pero debe tener al menos uno
- * El atributo puede tener varios valores.
- M..N El atributo puede tener entre M y N valores.

No especificar multiplicidad significa dejarlo sub-especificado (0..*)

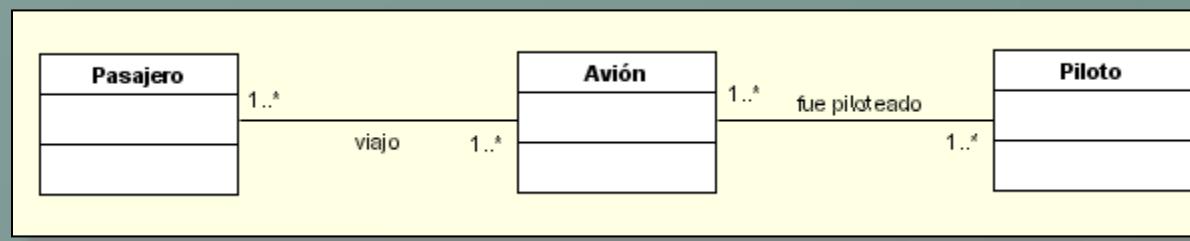


Cada objeto de la clase A esta asociado con un objeto de la clase B como máximo

Cada objeto de la clase B esta asociado con uno o mas objetos de la clase A

Visión Semántica

- Un diagrama de clases define
 - Conjuntos de objetos
 - Relaciones entre elementos de conjuntos
 - Restricciones sobre conjuntos y relaciones

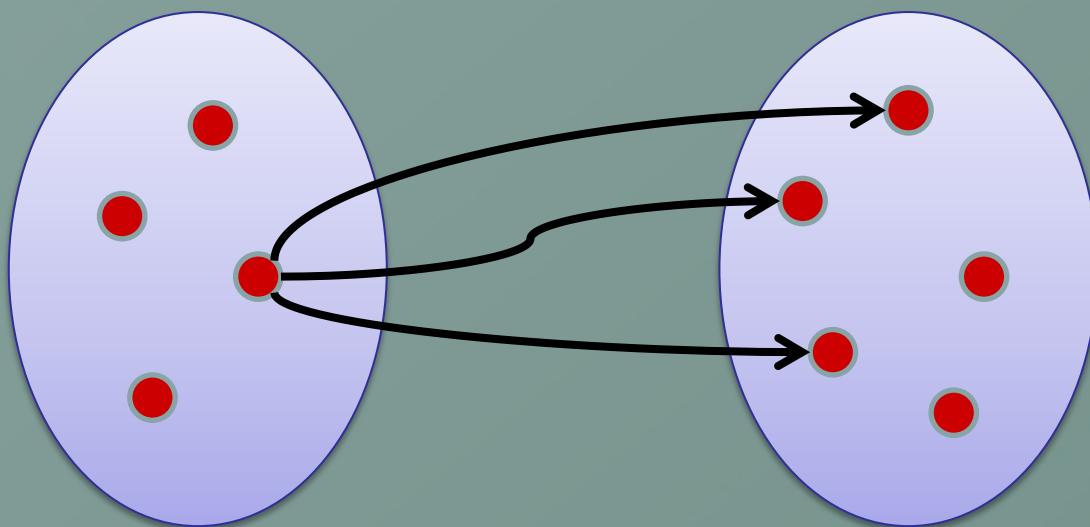


- Sean tres conjuntos: Pasajeros, Aviones, Pilotos
- Sean dos relaciones:
 - Viajo incluido en Pasajeros × Aviones
 - FuePiloteado incluido en Aviones × Pilotos
- Tal que
 - Para todo p en Pasajeros: Existe a en Aviones tq: Viajo(p, a)
 - Para todo p en Aviones: Existe a en Pasajeros tq: Viajo(p, a)
 - Para todo a en Aviones: Existe p en Pilotos tq: FuePiloteado(a, p)
 - Para todo p en Pilotos: Existe a en Aviones tq: FuePiloteado(a, p)

*¿un ejemplo
concreto?*

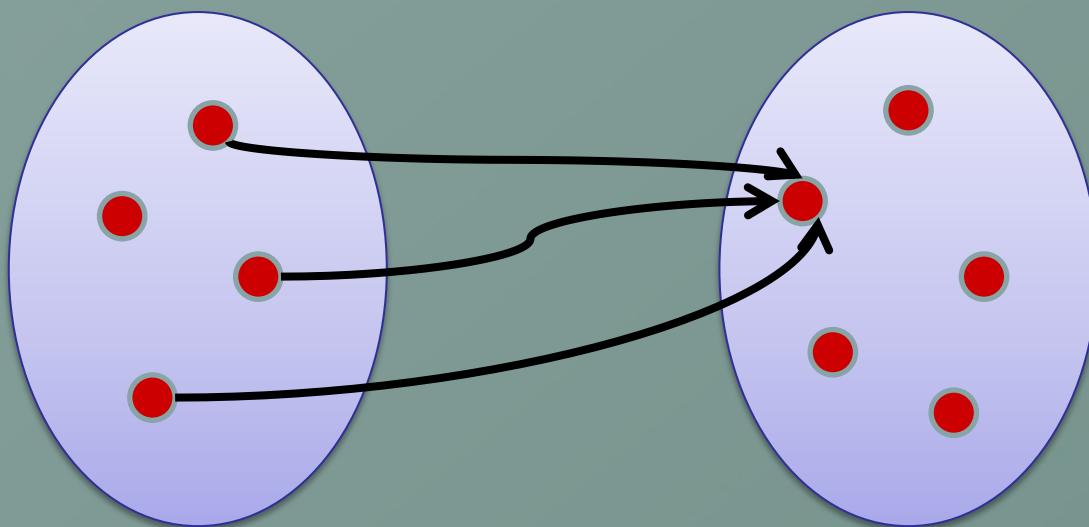
Asociaciones Típicas

- Uno a Muchos (One to Many)



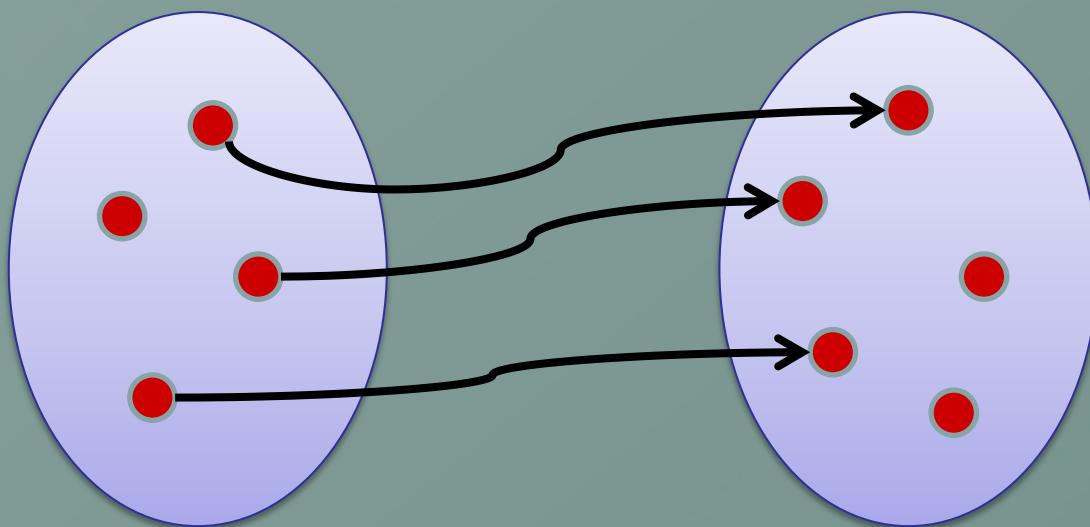
Asociaciones Típicas

- Muchos a Uno (Many to One)



Asociaciones Típicas

- Uno a Uno (One to One)



Asociaciones Típicas

- Muchos a Muchos (Many to Many)

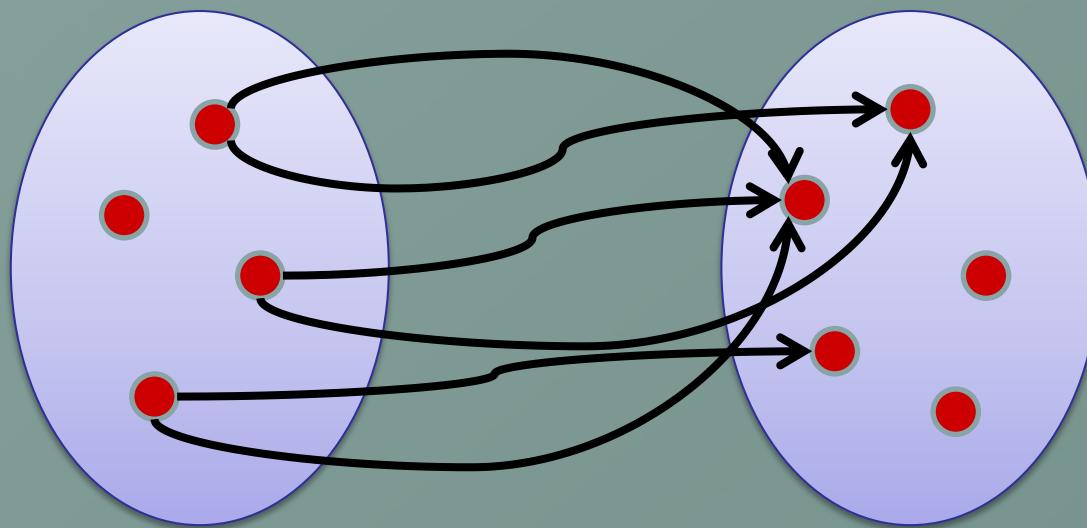


Diagrama de Clases: Asociación

Rol

- Identificado como un nombre a los finales de la asociación, describe la semántica de la relación en el sentido indicado.
- Cada asociación tiene dos roles; cada rol es una dirección en la asociación.

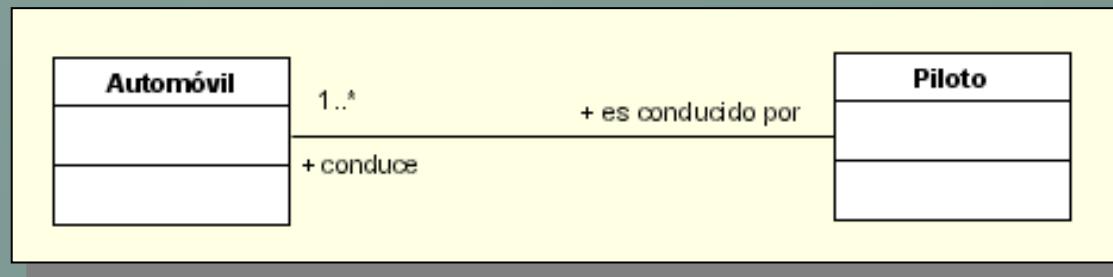


Diagrama de Clases: Asociación

- Se asume que una asociación es bidireccional, es decir que se puede navegar desde cualquiera de las clases implicadas a la otra, pero es posible indicar que la navegación ocurrirá en una sola dirección.

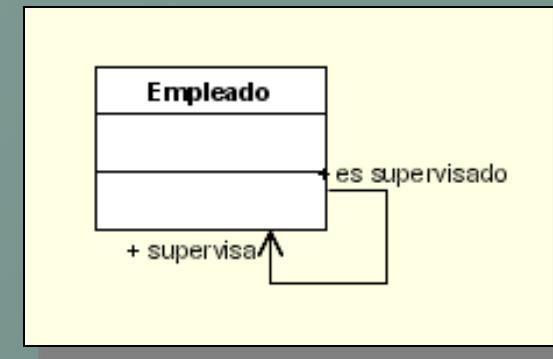


Diagrama de Clases: Agregación

- Es una asociación especial, una relación del tipo “todo/parte” dentro de la cual una o más clases son partes de un conjunto.

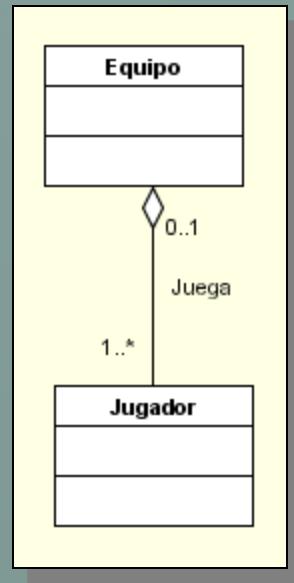


Diagrama de Clases: Composición

- La composición es una forma ‘fuerte’ de agregación. Se diferencian en:
 - En la composición tanto el todo como las partes tienen el mismo ciclo de vida.
 - Un objeto puede pertenecer solamente a una composición.

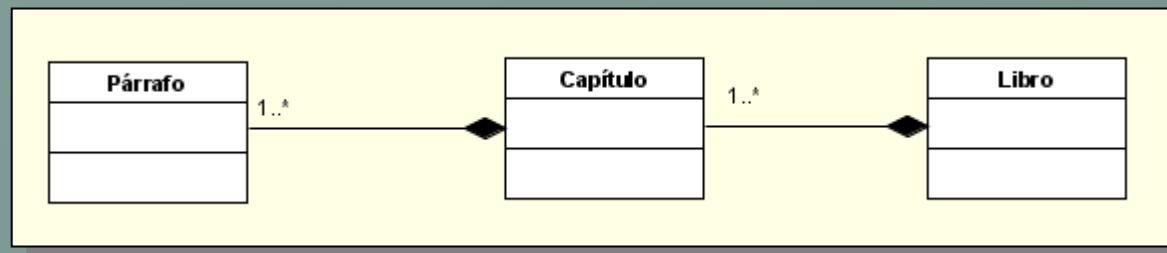


Diagrama de Clases: Asociación Calificada

- Un calificador es un atributo (o tupla de atributos) de la asociación cuyos valores sirven para particionar el conjunto de objetos enlazados a otro.
- Un calificador se representa como un pequeño rectángulo conectado al final de una asociación y a la clase.
- El rectángulo del calificador es parte de la asociación, y no parte de la clase.

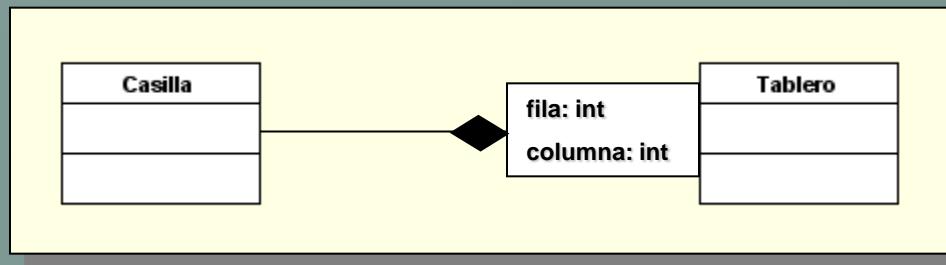
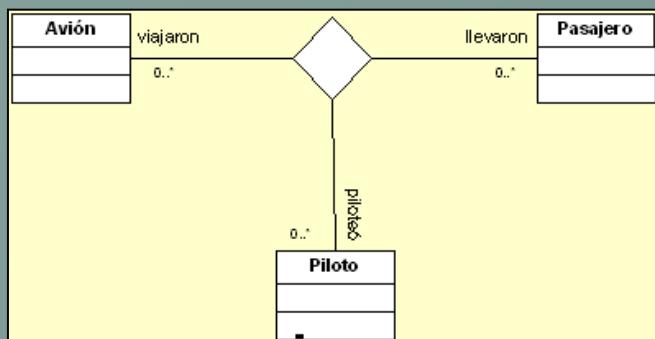
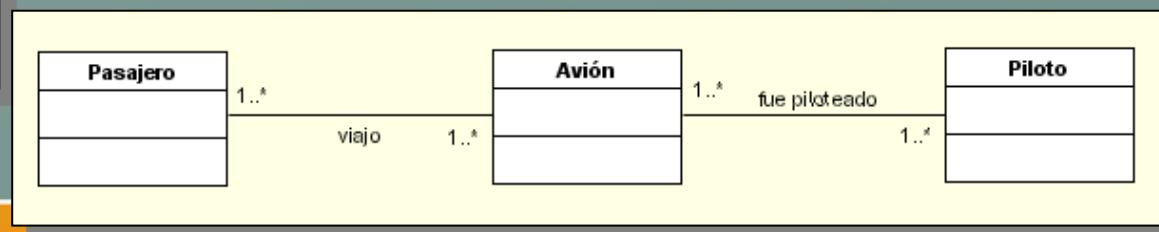


Diagrama de Clases: Asociación n-arias

- Son asociaciones que se establecen entre más de dos clases
- Una clase puede aparecer varias veces desempeñando distintos roles.
- Las asociaciones n-arias se representan a través de rombo que se une con cada una de las clases.



**La relaciones n-arias
pueden ser usadas para
impedir inconsistencias
en el modelo.**



Multiplicidad en Asociaciones n-arias

- No es fácil definir las multiplicidades en clases n-arias
 - Suelen interpretarse 1 clase contra las demás
 - es la interpretación de la cátedra
 - introduce limitaciones expresivas
- Solución:
 - Subespecificar..., o
 - resolverlas introduciendo nuevas clases...
 - ¿Cómo sería para los aviones/pasajeros/Pilotos?

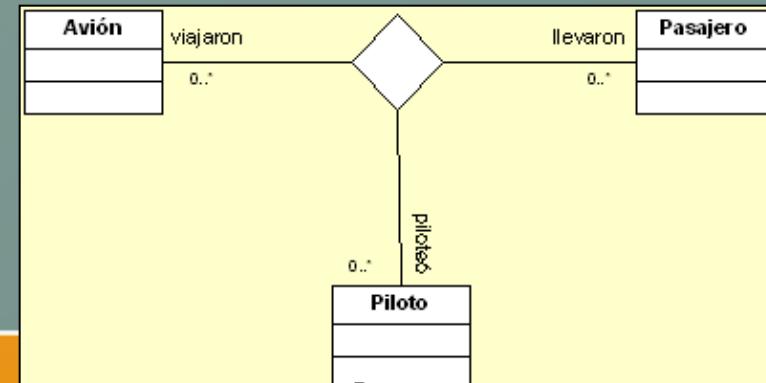
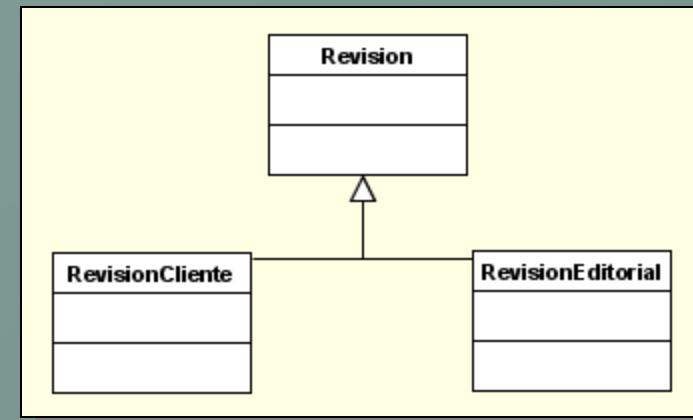
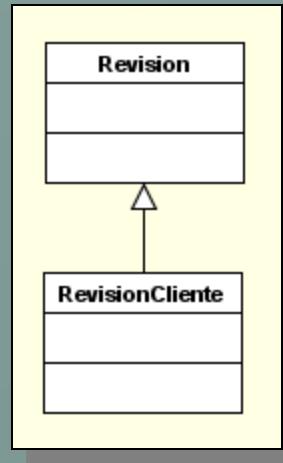
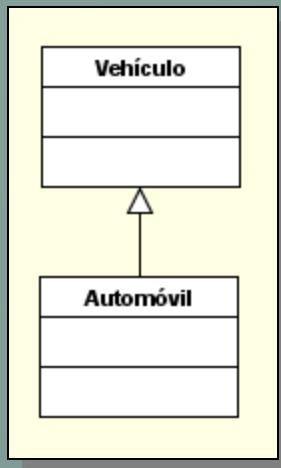


Diagrama de Clases: Generalización

- Una generalización se refiere a una relación entre una clase general (superclase o padre) y una versión más específica de dicha clase (subclase o hija).



Generalización

- Nombres usados: clase padre - clase hija. Otros nombres: superclase - subclase, clase base - clase derivada.
- Las subclases heredan las características de sus superclases es decir, atributos y relaciones.
- Las subclases pueden incorporar nuevos atributos o relaciones que las superclases no tienen

Algunos modificadores

- Overlapping/Disjoint
- Complete/Incomplete

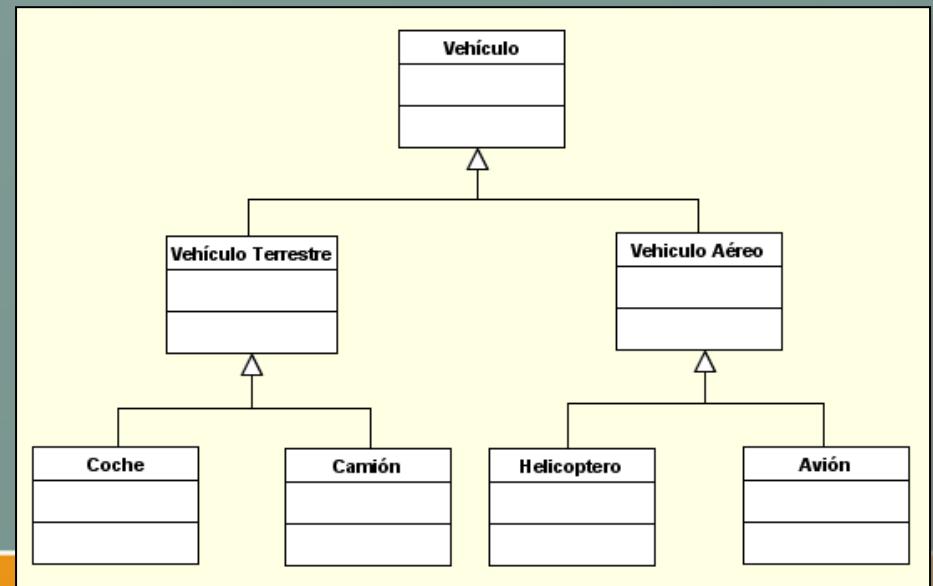


Diagrama de Clases: Generalización

- Particionamiento del espacio de objetos → Clasificación Estática
- Particionamiento del espacio de estados de los objetos → Clasificación Dinámica
- En ambos casos se recomienda considerar generalizaciones/especializaciones disjuntas
- Usando discriminadores se pueden tener varias especializaciones de una misma clase padre

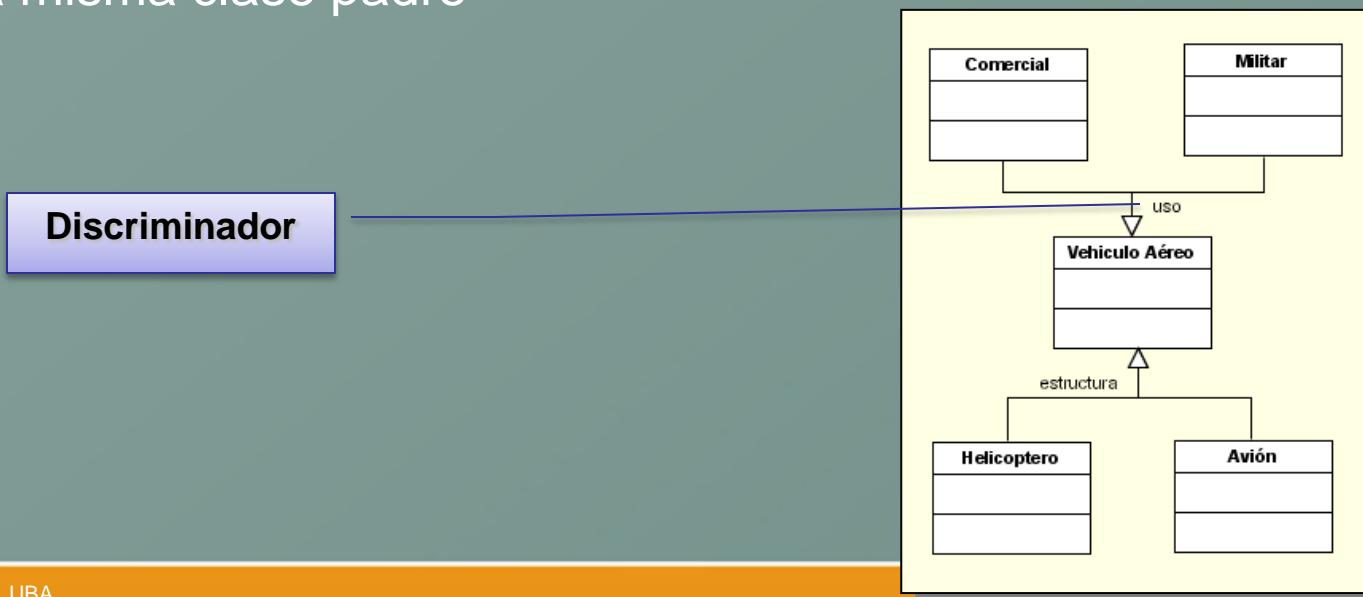


Diagrama de Clases: Generalización

- La herencia múltiple debe manejarse con precaución. Algunos problemas son el conflicto de nombre y el conflicto de precedencia.
- Se recomienda un uso restringido y disciplinado de la herencia.
- Permite modelar jerarquías alternativas.

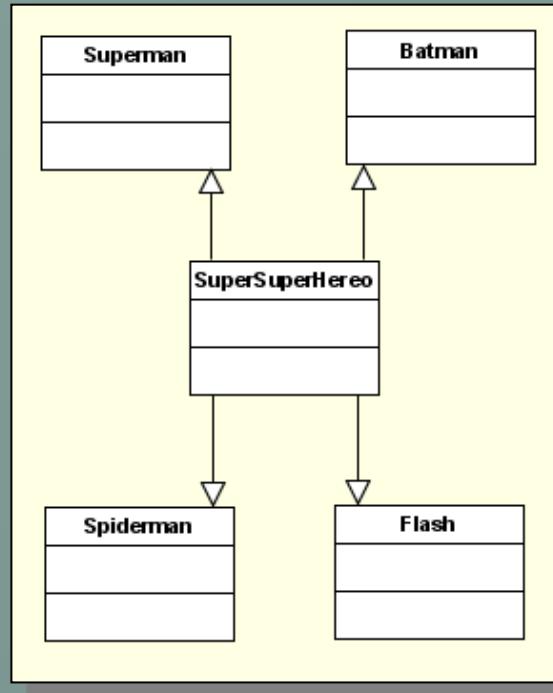
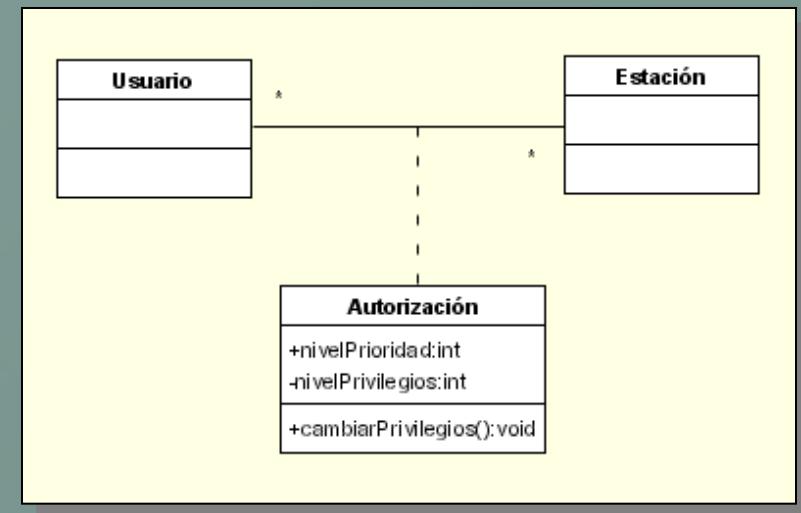
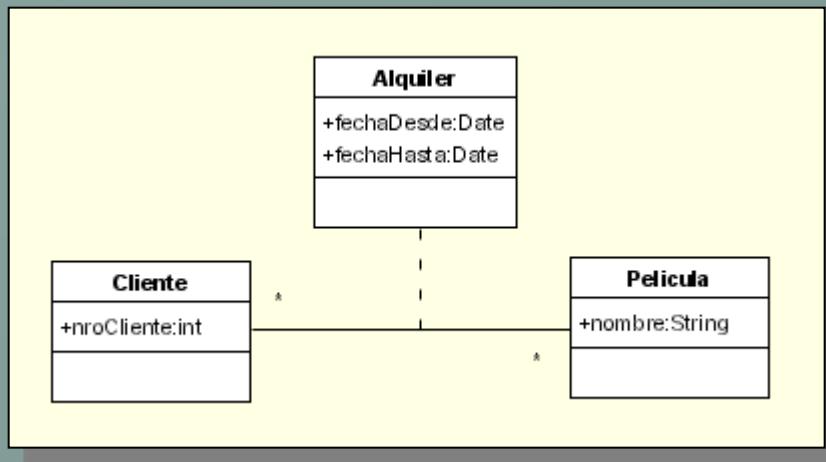


Diagrama de Clases: Clase de asociación

- Es una asociación y una clase simultáneamente.
- Hay que tener en cuenta dónde se colocan los atributos.



Clases de Asociación

- Modelan características de una asociación que son independientes de las clases que asocia.
- Si C es una clase de asociación para la asociación R en (A x B), entonces introduce función f: R → C
 - Garantiza que no hay dos c, para un mismo par (a, b)
- No tiene sentido la clase en por si sola
 - Para todo c en C, existe r en R tal que f(r) = c
 - Si c = f(r), c no puede cambiar sólo (si cambia es porque la naturaleza de la asociación de r cambió)

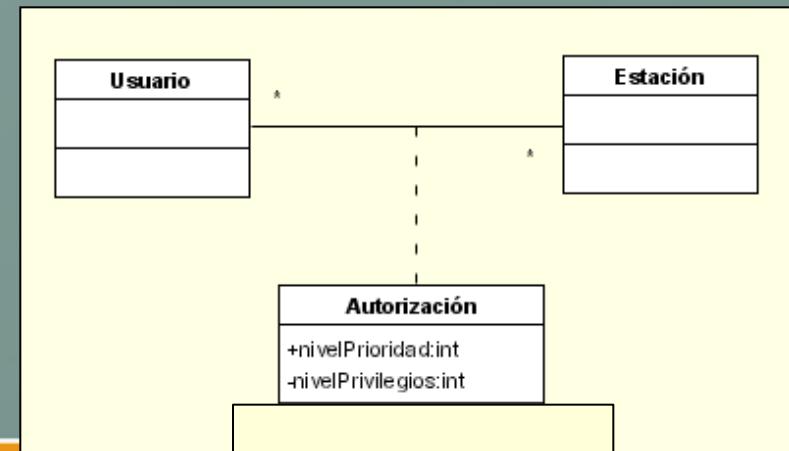
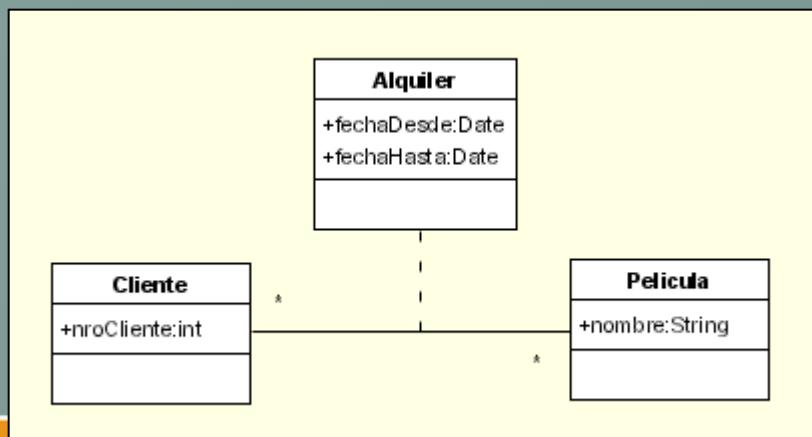


Diagrama de Objetos

- El diagrama de clases define la estructura del mundo
- El diagrama de objetos define el mundo en un instante dado
- La relación entre los objetos se corresponde con la de sus clases
 - Toda instancia de una clase X tiene enlaces con instancias de Y de acuerdo al tipo de relación, atributos y modificadores que X tiene con Y en el modelo de clases
- Instanciación:
 - Un objeto es una instancia de una clase
 - un diagrama de objetos es una instancia de un diagrama de clases

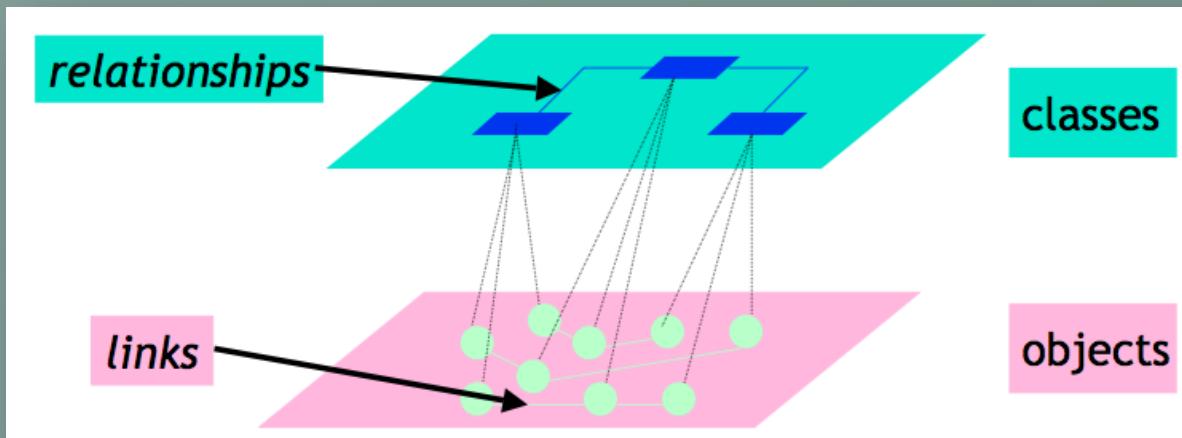
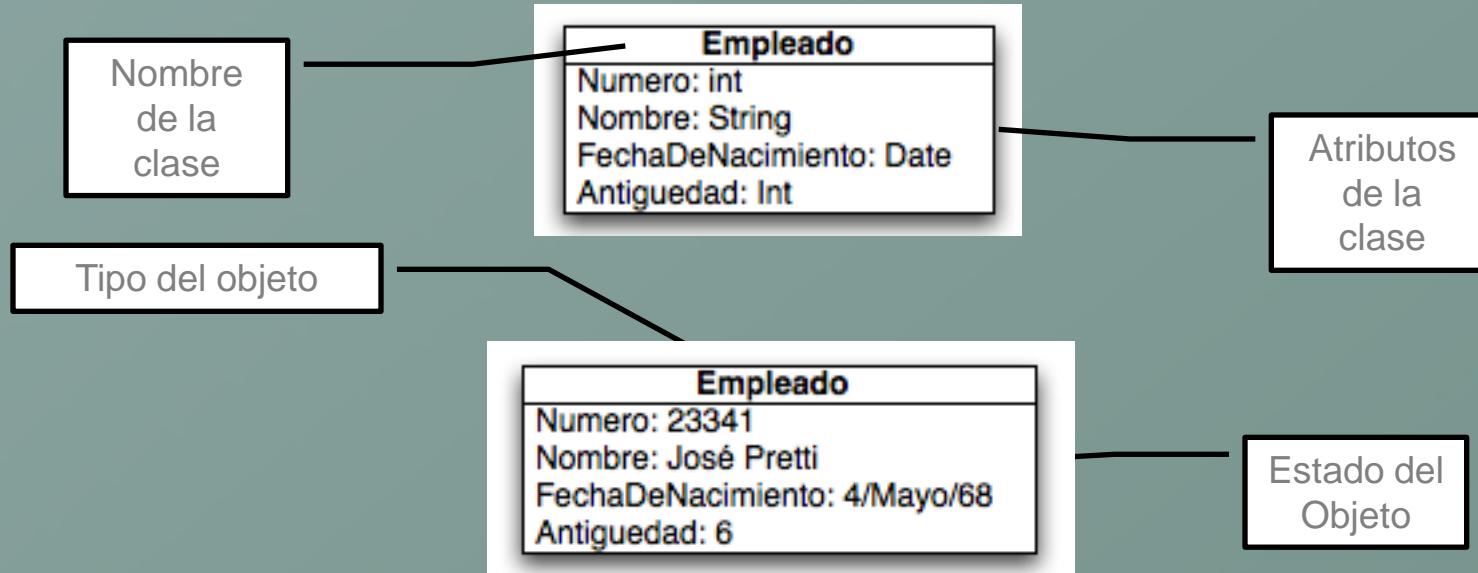


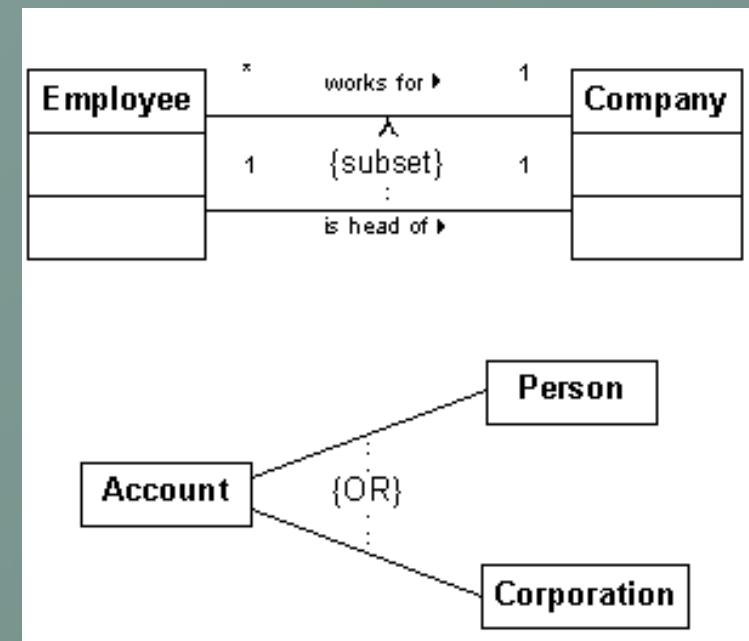
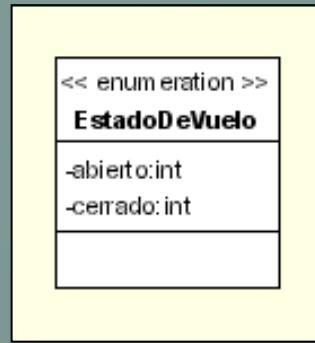
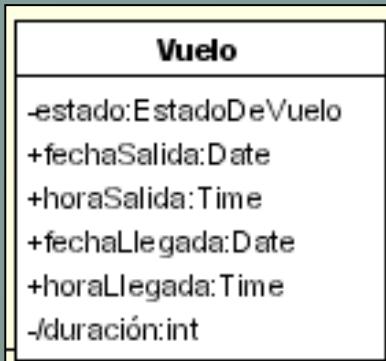
Diagrama de Objetos



- Un objeto denota una entidad conceptual del mundo real.
- Aunque los estados del objeto cambien, el objeto continua denotando la misma entidad (ahora con un estado distinto)
- Es decir, los objetos tienen una identidad mas allá de los valores de sus atributos

Algunos Elementos Sintácticos

- Existe una pléthora de elementos sintácticos que pueden utilizarse en un modelo conceptual
- Aquí hay algunos que usaremos en la práctica
 - Clases enumeradas: Conjunto de valores pre-fijados
 - OR/XOR de relaciones
 - Subset
 - Atributos Derivados
 - Explicitan invariantes



Modelo Conceptual vs Diseño

- El diagrama de clases y de objetos se usan para modelar el dominio del problema y el de la solución
 - Modelo Conceptual:
 - Clases denotan “tipos” de entidades conceptuales del mundo real.
 - Objetos denotan entidades conceptuales del mundo real
 - Modelo de Diseño:
 - Clases y objetos discriminan cómo se agrupará el código y los datos al momento de programar el software y al momento de ejecutarlo
- Operaciones
 - En el modelo de Diseño es común asignarle operaciones a las clases y objetos.
 - En el modelo conceptual esto sólo tiene sentido para entidades activas (Agentes).
 - ¿Tiene sentido pedirle un libro que se rompa?
 - Meilir dixit: “When it comes time to milk a cow, do we tell the milk to exit the cow or the cow to release the milk? ”

Modelo de Dominio vs. Modelo de Diseño

- El diagrama de clases puede utilizarse con distintos fines en distintas etapas del proceso de desarrollo.
- Durante la etapa de análisis, el ***modelo de dominio*** es encargado de mostrar el conjunto de clases conceptuales del problema y las relaciones presentes entre sí.
- Durante la etapa de diseño, el ***modelo de diseño*** determina las futuras componentes de software (clases) y sus relaciones entre sí.

Síntesis

- Es una representación de las cosas, entidades, idea, clases conceptuales u objetos del “mundo real” o dominio de interés, no de componentes de software.
- Muestra clases conceptuales significativas en un dominio del problema.
- Se usa como base para el diseño de los objetos de software.

Síntesis

- Es el artefacto más importante del análisis.
- Podría ser considerado como un diccionario visual de abstracciones de clases conceptuales, vocabulario e información del dominio.
- No es absolutamente correcto o incorrecto, su intención en ser útil sirviendo como una herramienta de comunicación.

Modelo de Dominio

- Otros *nombres*: modelo conceptual, modelo de objetos del dominio y modelo de los objetos de análisis.
- Según el punto de vista, tiene puntos en común con el *Diagrama de Entidad Relación*.
- Usando UML, el MD se representa con un conjunto de diagramas de clases. Se puede mostrar:
 - *objetos del dominio* o *clases conceptuales*
 - *asociaciones* entre las clases conceptuales
 - *atributos* de las clases conceptuales

NO SE DEFINE NINGUNA OPERACIÓN.

La asignación de responsabilidades de los objetos no forma parte de este modelo.

Modelo de Dominio: Clases Conceptuales

Es válido...

- Tener clases conceptuales sin atributos.
- Tener clases conceptuales para las cuales no haya requerimientos de información a registrar.
- Tener clases conceptuales con **rol de comportamiento**, en lugar de información.

Estrategias para identificar

- Utilizar lista de categorías de clases conceptuales.
- Identificar frases nominales (sustantivos o frases).

Modelo de Dominio: Clases Conceptuales

Categoría	Ejemplo
Objetos tangibles o físicos	Casa, Avión
Especificaciones, diseños o descripciones de las cosas	PlanoDeLaCasa, EspecificaciónDelProducto, DescripciónDelVuelo
Lugares	Tienda, Aula
Transacciones	Venta, Pago, Reserva, Transferencia
Líneas de la transacción	LíneaDeVenta
Roles de la gente	Cajero, Piloto, Jefe
Contenedores de otras cosas	Aula, Ciolectivo, Lata, Mochila
Contenidos	Pasajero, Artículo, ÚtilEscolar
Otros sistemas informáticos o electromecánicos externos al sistema	SistemaDeAutorización, ControlDeTraficoAéreo
Conceptos abstractos	Amor, Celos, Ansia, Acrofobia
Organizaciones	DepartamentoDeVentas, CompañíaAérea
Hechos	Reunión, Vuelo, Aterrizaje, Venta, Pago
Procesos (normalmente no se representan como conceptos, pero podría ocurrir)	VentaDeUnProducto, ReservaDeUnAsiento (no confundir con trasacciones)
Reglas y políticas	PolíticaDeReintegro, PolíticaDeCancelación
Catálogos	CatálogoDeProductos
Registros de finanzas, trabajo, contratos, cuestiones legales	Recibo, Remito, Factura, ContratoDeEmpleo, Expediente
Instrumentos y servicios financieros	LíneaDeCrédito, Stock
Manuales, documentos, artículos de referencia, libros	ManualDeReparaciones, ListaDeCambios
Relaciones	Amistad, Parentesco (podría estar en conceptos abstractos pero es bueno destacarlo ya que las relaciones no suelen ser consideradas al modelar)

Modelo de Dominio: Clases Conceptuales

Identificar frases nominales (sustantivos o frases)

Se intenta identificar sustantivos o frases nominales en el vocabulario y descripciones del dominio del problema. Esta técnica práctica no puede ser aplicada mecánicamente sino que hay que usar el “sentido común” y capturar las abstracciones adecuadas puesto que el lenguaje natural es ambiguo y los conceptos relevantes no siempre se encuentran de manera explícita.

Ejemplo

Un posible modelo de dominio para el caso del local de venta de electrodomésticos...

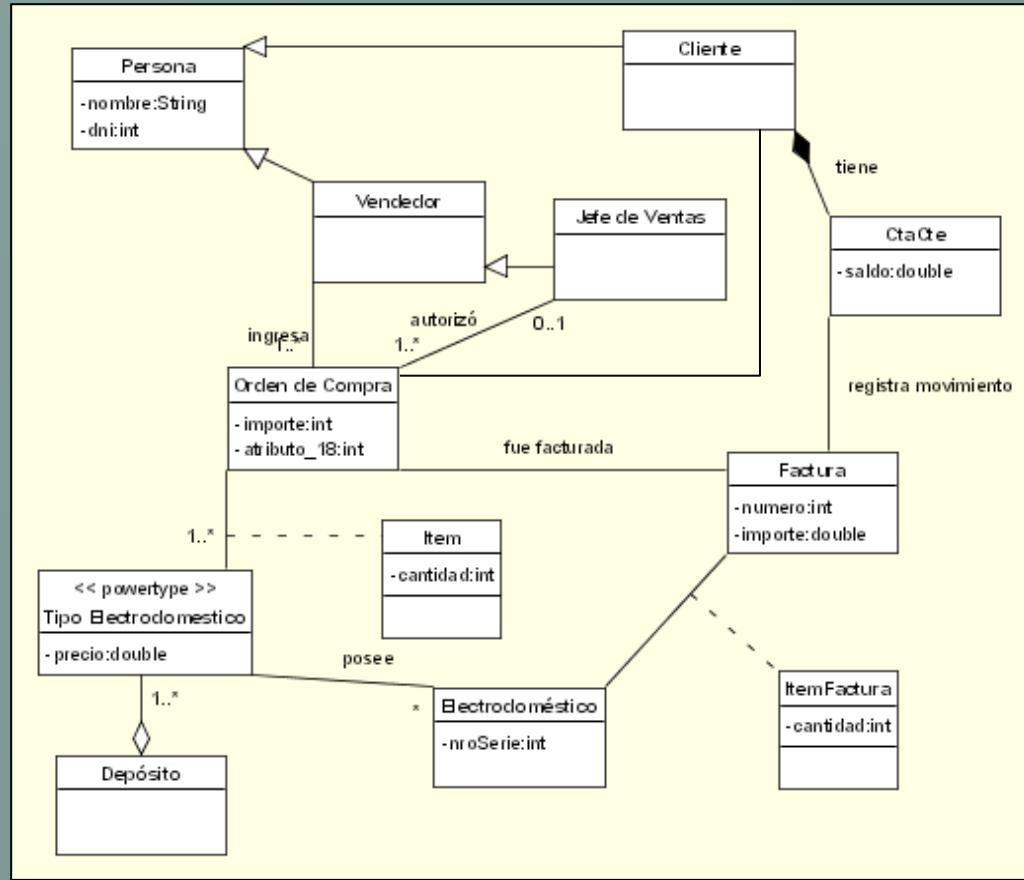
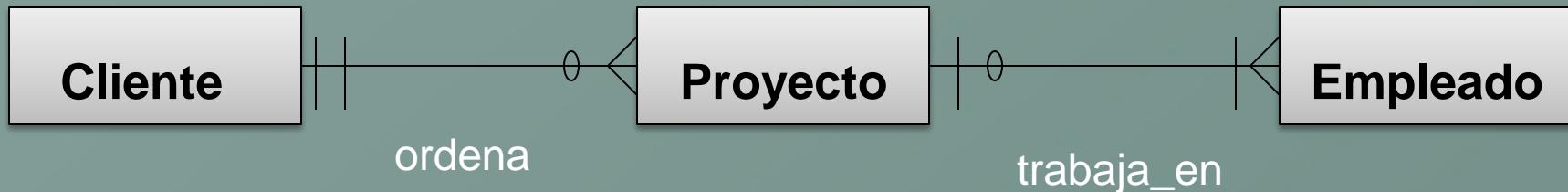


Diagrama Entidad Relación

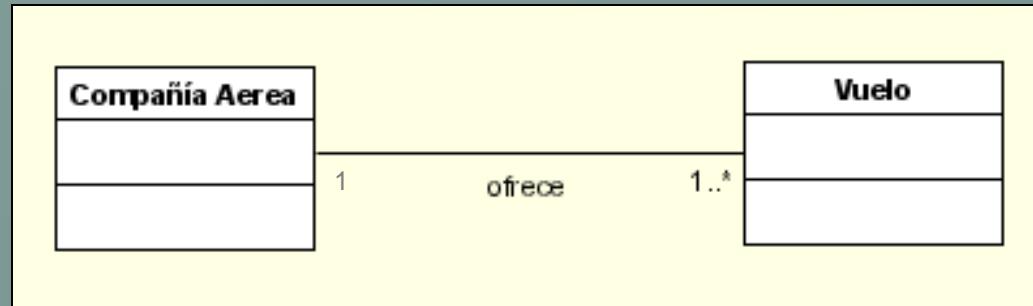
- No pertenece a UML
- Nacido para describir bases de datos relacionales (Chen).
- 2 conceptos: entidades y relaciones.
 - Entidades: conjuntos de individuos que poseen atributos.
 - Relaciones entre individuos especificando cardinalidad y opcionalidad.



Compañía Área

“Las compañías aéreas ofrecen varios vuelos”

- Compañía aérea y Vuelo son conceptos importantes del mundo real con atributos y comportamientos, por lo que son clases candidatas para nuestro modelado estático de dominio



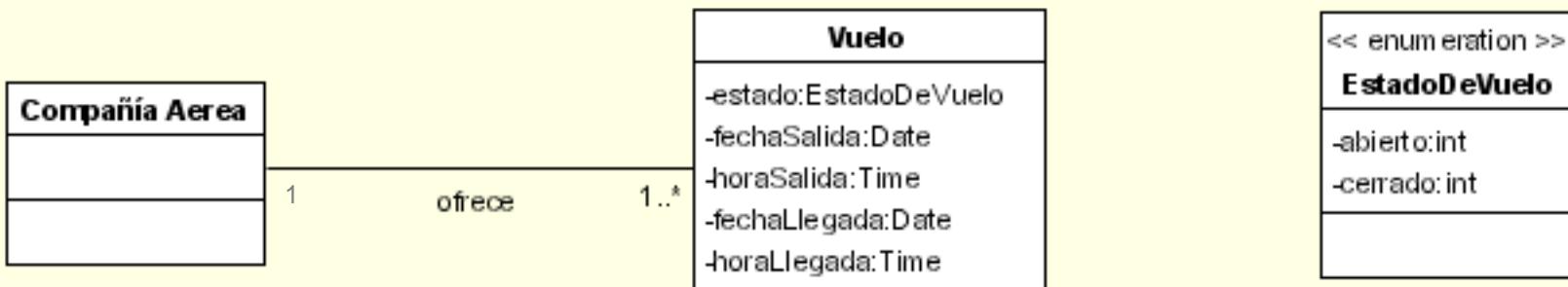
Compañía Área

“Una compañía abre y cierra las reservas para un determinado vuelo”



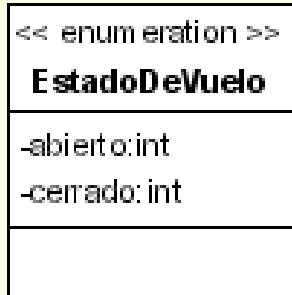
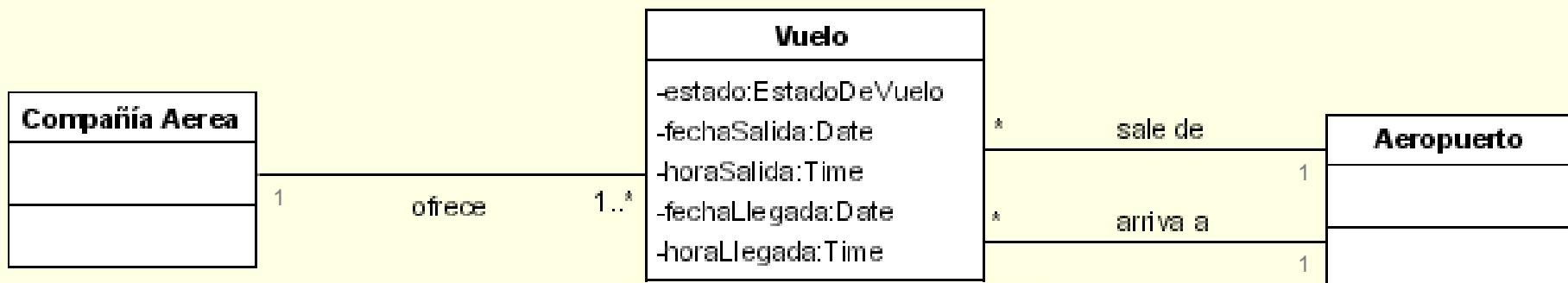
Compañía Área

“Un vuelo tiene un día y una hora de salida y un día y hora de llegada”



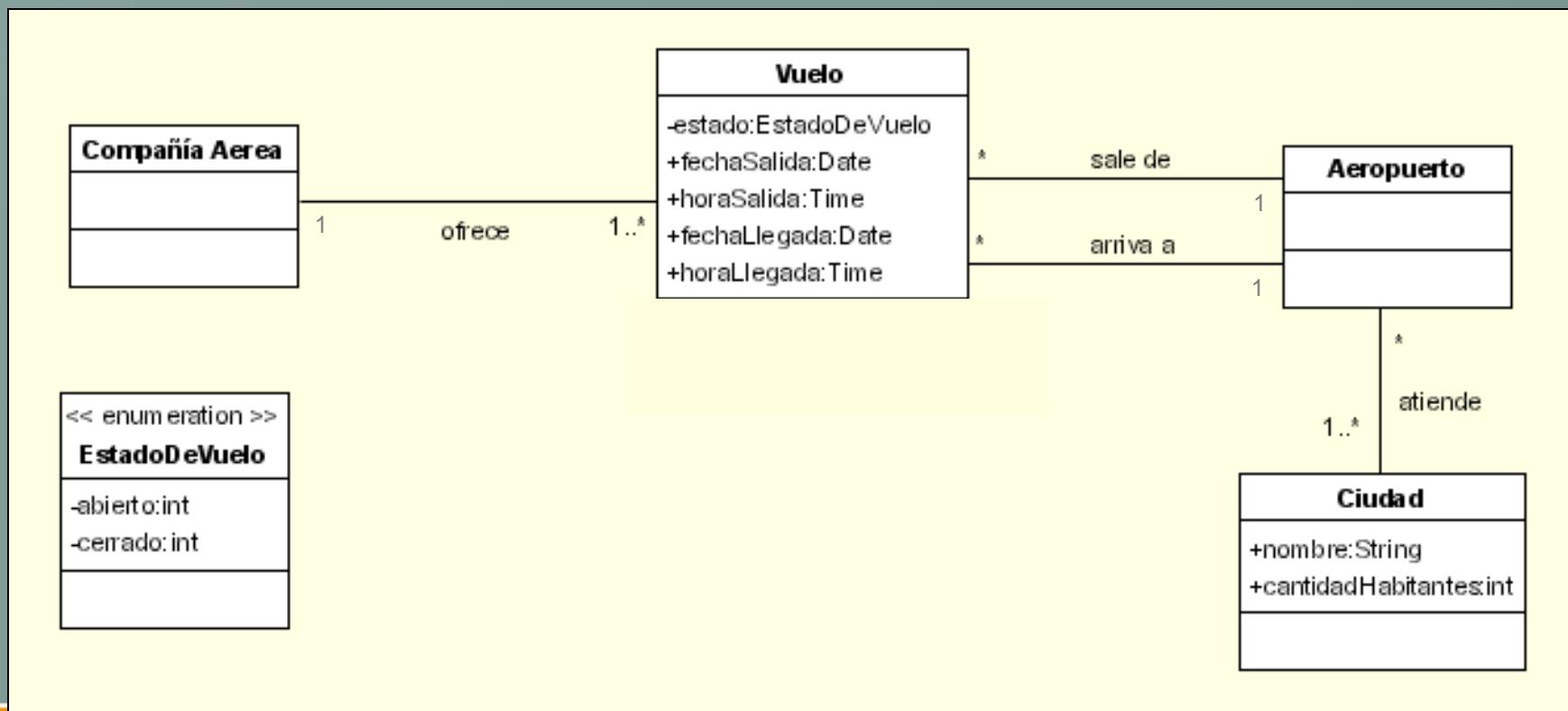
Compañía Área

“Un vuelo tiene un aeropuerto de salida y otro de llegada”



Compañía Área

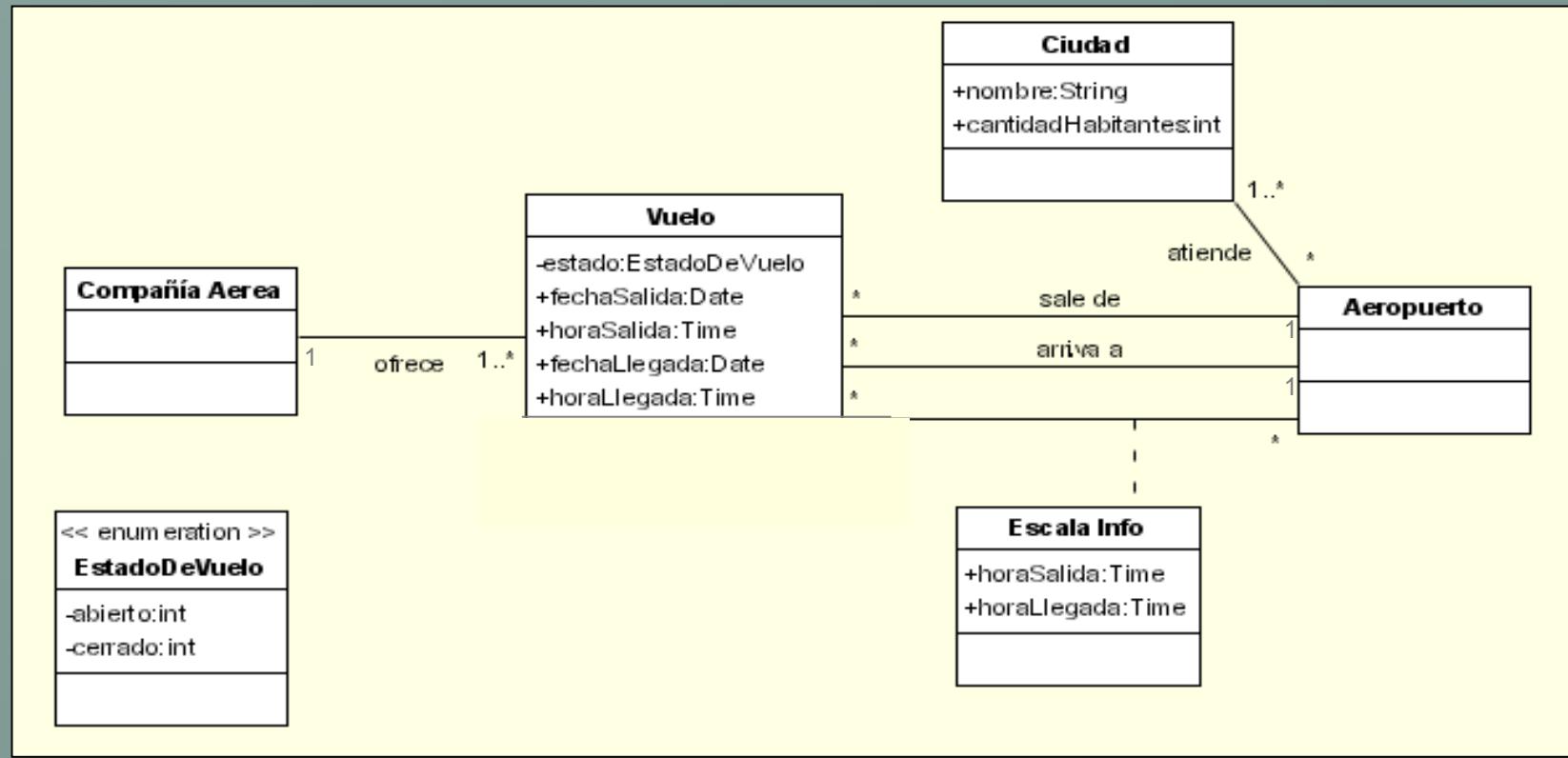
“Cada aeropuerto atiende a una o varias ciudades”



Compañía Aérea

“Un vuelo puede implicar escalas en aeropuertos”

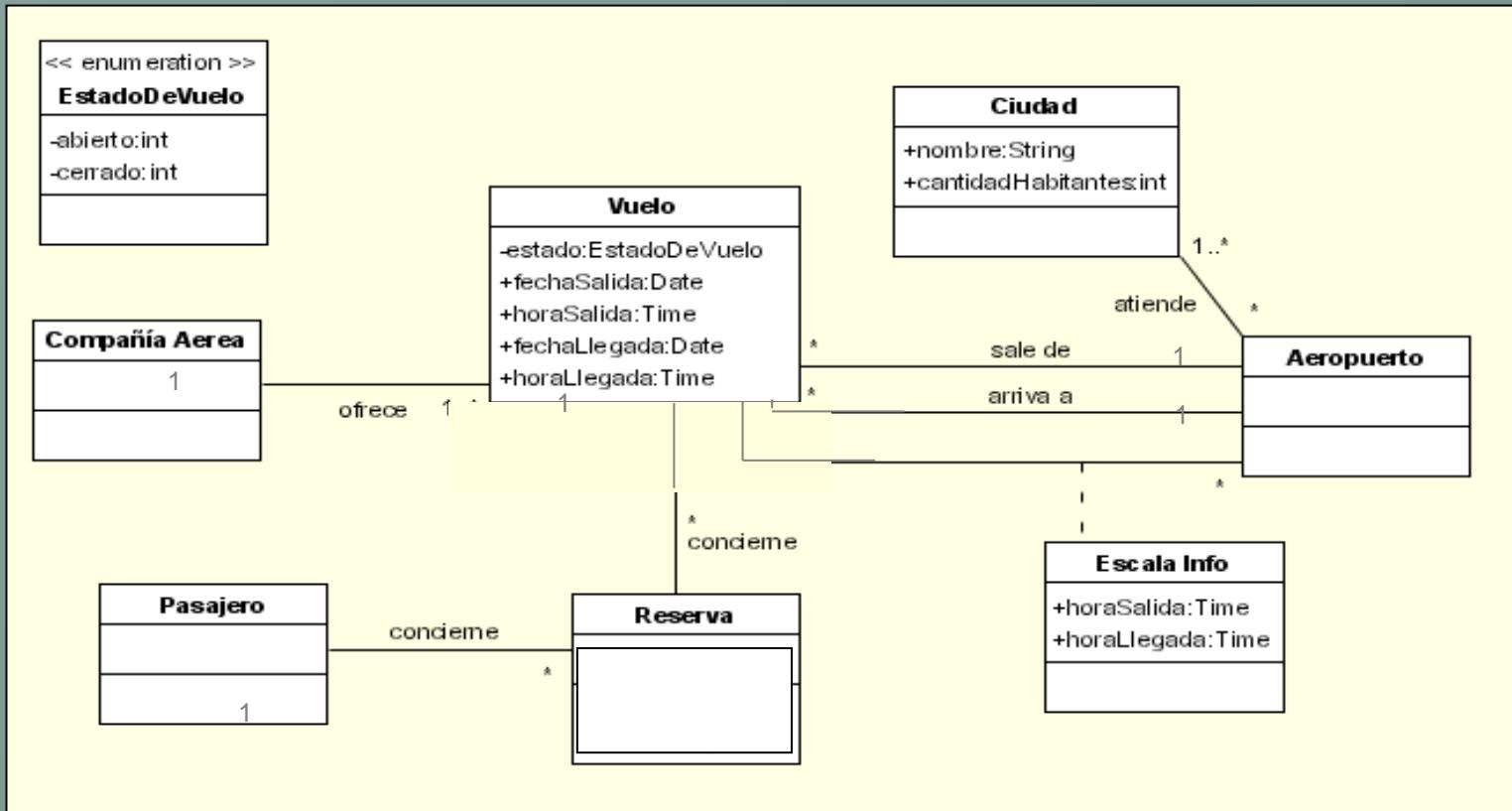
“Una escala tiene una hora de llegada y otra de salida”



Compañía Área

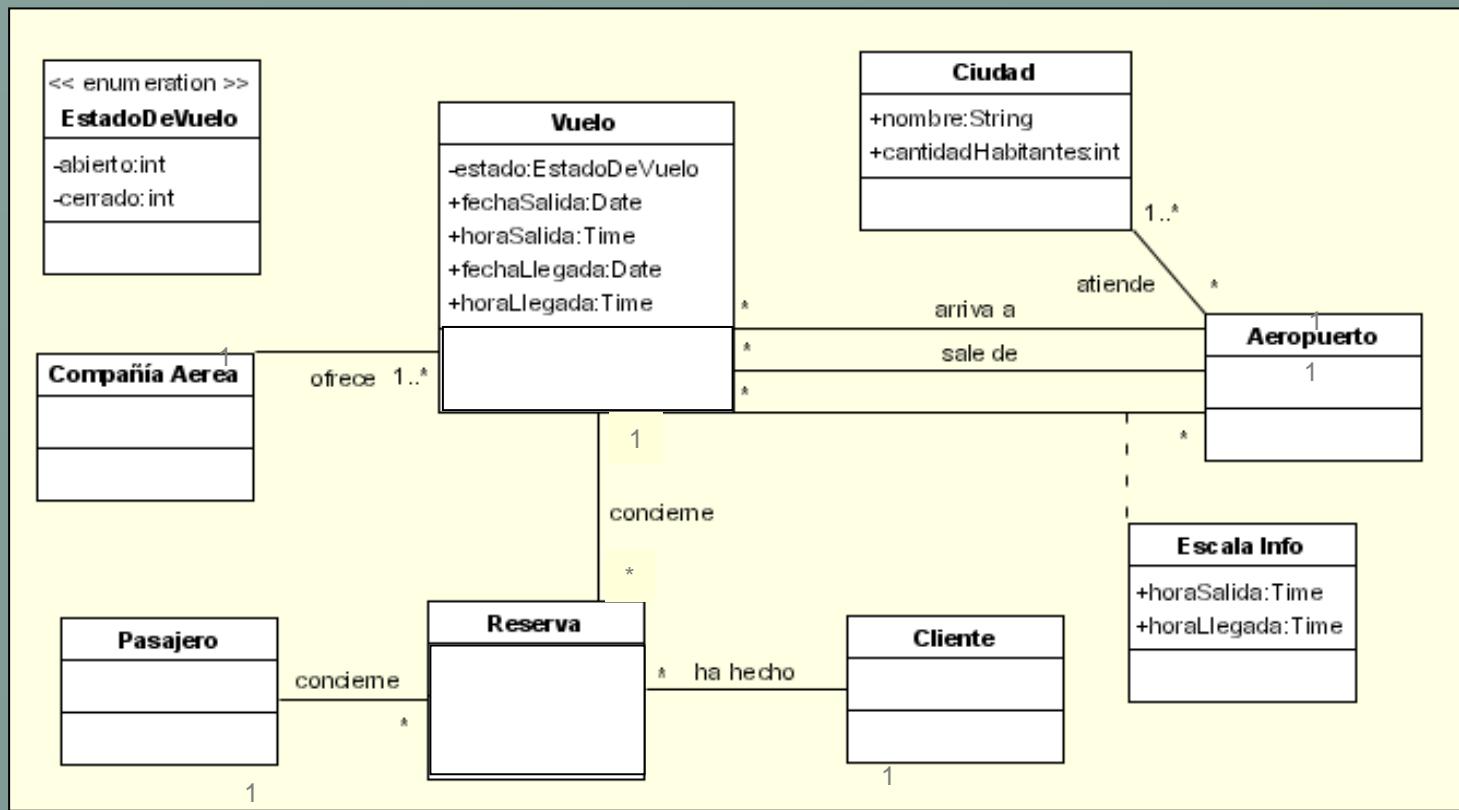
“Una reserva implica un único vuelo y un único pasajero”.

“Una reserva puede cancelarse o confirmarse”.

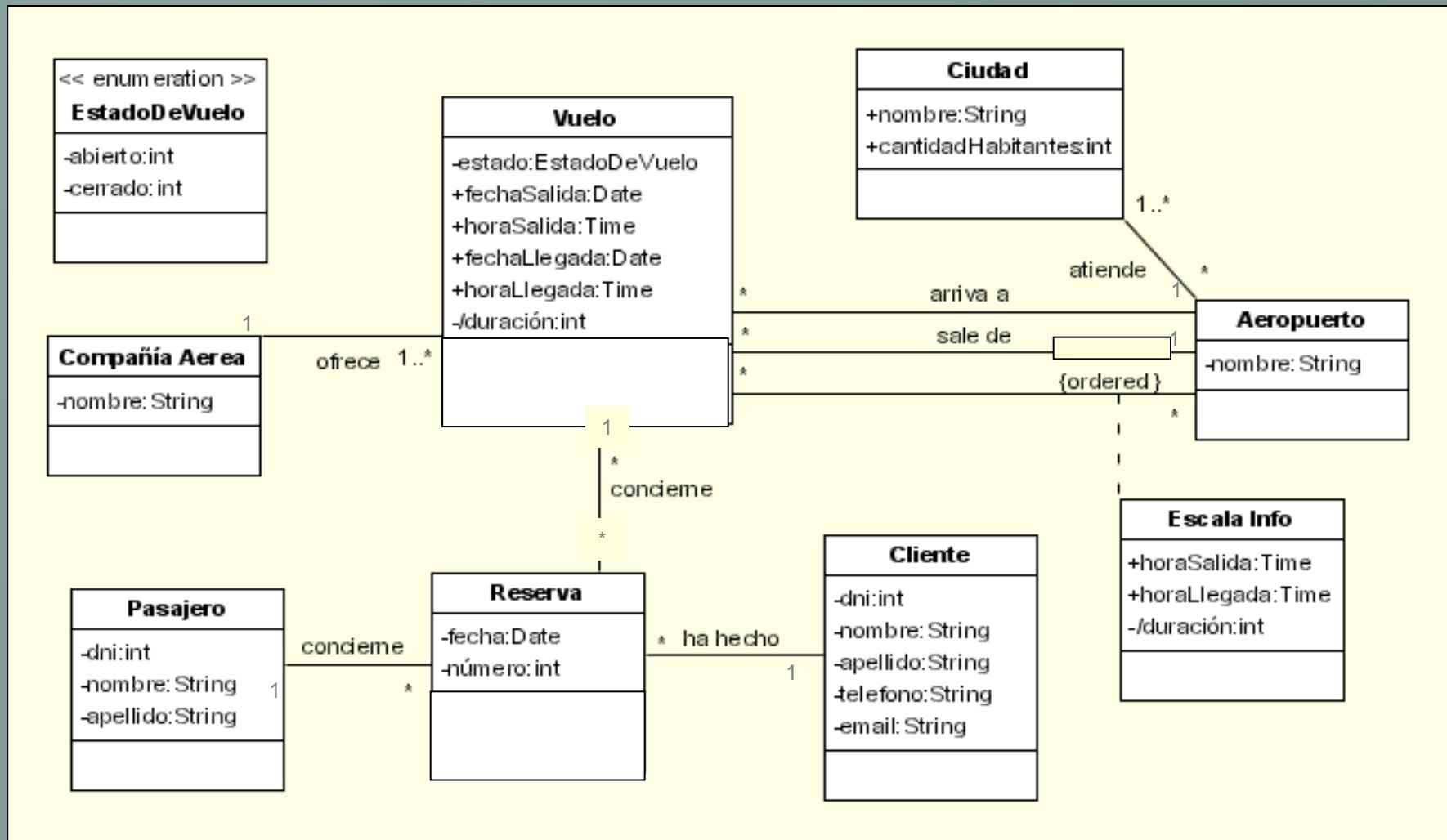


Compañía Área

“Un cliente puede reservar uno o más vuelos y para pasajeros diferentes”.



Compañía Área



Relación con Otros Modelos

- **Modelo de Objetivos**
 - Describe la estructura estática del mundo sobre la que los objetivos predicen
 - Es la base para formalizar los objetivos
 - Para todo tren, $t.puertasAbiertas$ si y solo si $t.velocidad=0$ y Existe plataforma p tal que $t.posición = p.posición$
- **Diagrama de Contexto**
 - Describe el estado interno de agentes
 - No toda clase corresponde con un agente
 - No necesariamente valga la pena tener todos los agentes en el modelo conceptual.
- **Modelo de Operaciones**