

INFORMÁTICA INDUSTRIAL I

Ingeniería Electrónica Industrial y Automática
3^{er} curso

1. Introducción a la Programación Orientada a Objetos

Profesores: M. Abderrahim, A. Castro González
Departamento de Ingeniería de Sistemas y Automática

Características Generales

- La orientación a objetos se presentó como la gran promesa frente a la *crisis del software*.
- Tiene como objetivos fundamentales la **calidad** y la **productividad**.
- Estimación de reducción de 40% con respecto a la programación convencional.
- Las razones de su popularidad son varias:
 - La POO es adecuada para programas de cualquier tamaño.
 - Reduce la distancia entre el problema y la solución.
 - Hay una gran disponibilidad de librerías.
 - Se supone que la transición entre los lenguajes que no son OO es sencilla.

¿Qué es la POO?

- Pensar en **objetos**

- Agrupación de datos y métodos para manipular esos datos
- Ej: ventana del S.O.



Datos: tamaño, posición, color, título, ...

Métodos: redimensionar, click en un botón, ...

- Comunicación entre objetos

- Ej: Click en un botón genera una ventana

¿Qué es la POO?

- Es un paradigma de programación que utiliza objetos que se comunican mediante mensajes para resolver un problema.
- Extensión de la programación estructurada
- Trata de potenciar la **modularidad** y la **reutilización** de código

Fundamentos de la POO: Objetos

● Objetos



- Los detalles internos están ocultos para el usuario. Los crea el desarrollador.
- Se relaciona con mensajes

Fundamentos de la POO

● Mensajes

- Los objetos reciben, procesan y responden a mensajes de otros objetos
- El objeto que recibe el mensaje debe saber qué hacer
- El objeto que envía un mensaje no necesita conocer cómo se desarrolla, sólo que se está desarrollando

● Métodos

- Definen cómo actuar cuando se recibe un mensaje
- Pueden enviar mensajes a otros objetos
- Su funcionamiento está oculto para el usuario

Fundamentos de la POO: Clases

- Descripción de objetos
- Abstracción de los métodos y datos comunes a un conjunto de objetos



CLASE PERSONA

Atributos: altura, peso, sexo, edad, nombre, ...

Métodos: andar, respirar, dormir, comer, beber, ...

Fundamentos de la POO: Clases

- Objeto: zona de memoria donde se almacena una copia de los elementos especificados por una clase
- Todo objeto es una instancia de una clase



Características de la POO: Abstracción

- Generalización de atributos y propiedades de un conjunto de objetos para definir su clase.
- No detenerse en los detalles
- Generalizar para tener una visión global

Características de la POO: Encapsulamiento

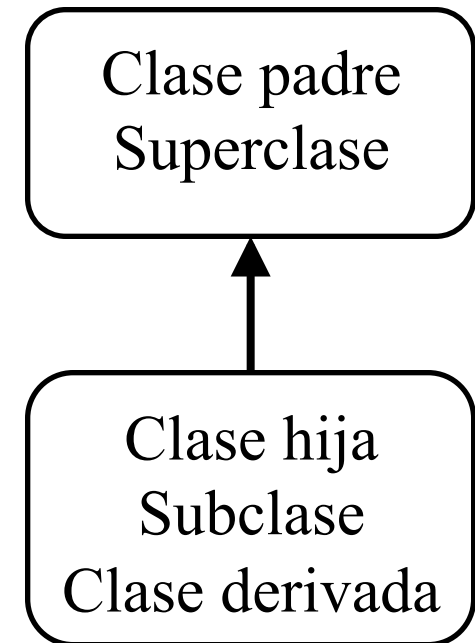
- Los objetos son “cajas negras” donde los detalles internos quedan ocultos
 - Ej. x , y y z son objetos de la clase *complejo* que representa números complejos y podemos hacer:

$$z = x / y$$

¿Cómo se realiza la división de complejos?

Características de la POO: Herencia

- Reutilización de código (métodos y datos miembro de una clase)
- Estructura jerarquizada siguiendo una estructura de herencia
- Una subclase heredará atributos (métodos y datos) de cualquier superclase que está por encima de ella siguiendo el árbol de herencia.
- Representa el concepto de “**ser un tipo especial de**” o “**ser un/a**”.



Características de la POO: Herencia



CLASE PERSONA

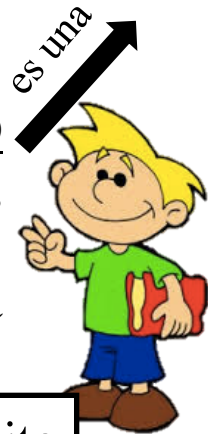
Atributos: altura, peso, sexo, edad, nombre, ...

Métodos: andar, respirar, dormir, comer, beber, ...

CLASE ALUMNO

Atributos: curso, asignaturas, nota media, ...

Métodos: estudiar, ir a tutorías, realizar prácticas, ...



Alumno Pepito

Atributos: 1,54m, 64kg, [...],
1º, programación, 8.4,...

Métodos: andar, respirar, [...],
estudiar, ir a tutorías, ...



CLASE PROFESOR

Atributos: despacho, departamento, categoría, ...

Métodos: dar clase, examinar, investigar,, ...

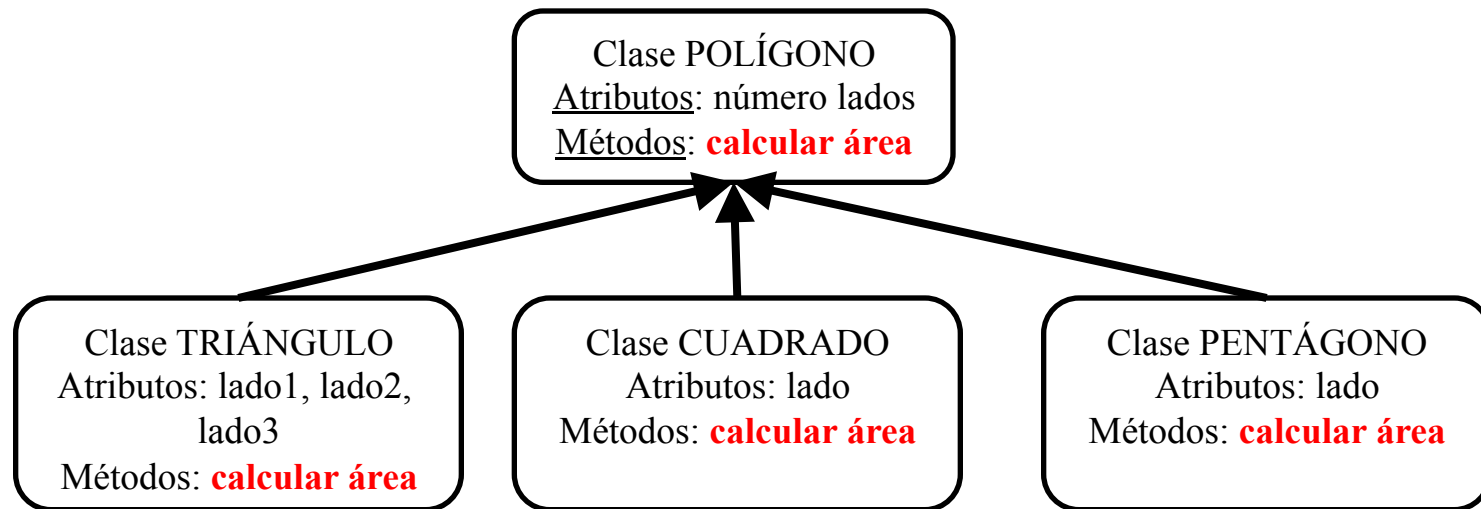
Profesor Luis

Atributos: 1,84m, 98kg, [...],
1.3.B01, catedrático,...

Métodos: andar, respirar, [...],
dar clase, examinar, ...

Características de la POO: Polimorfismo

- Diferentes implementaciones para un mismo método dependiendo de la clase con la que se realice
- Acceso a varios métodos distintos que tienen el mismo nombre
- Una variable p
- Una operación con la misma interfaz puede corresponder a operaciones distintas, según la clase del objeto.



¿Cómo modelar un problema según la POO?

- Descripción del problema
- Análisis
 - Nombres:
 - Objetos
 - Propiedades de objetos
 - Adjetivos:
 - Valores de las propiedades
 - Verbos:
 - Comportamiento de los objetos
- Ejemplo:
 - “El avión tiene color azul y se mueve”
 - “El documento tiene letra grande y se muestra”

Ventajas de la POO

- Mejor mantenimiento.
- Estructuras más reales de la información.
- Escalabilidad.
- Adaptabilidad.
- Más apropiada para aplicaciones dirigidas por eventos.

Desventajas de la POO

- Ejecuciones más lentas
- Necesidades de estandarización:
 - Notación de Modelado (OMG, Object Management Group).
 - Lenguajes de Programación.
- Coste de conversión de software heredado.

INFORMÁTICA INDUSTRIAL I

Ingeniería Electrónica Industrial y Automática
3^{er} curso

2. Modelado. El Lenguaje Unificado de Modelado (UML)

Profesor: M. Abderrahim / A. Castro
Departamento de Ingeniería de Sistemas y Automática

El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

¿Qué es UML?

- Consiste en un conjunto integrado de diagramas definidos para ayudar a los desarrolladores de software y de sistemas a realizar las tareas de:
 - Especificación
 - Visualización
 - Diseño Arquitectónico
 - Construcción
 - Simulación y pruebas
 - Documentación

¿Por qué es necesario?



Como el cliente lo explicó



Como el lider del proyecto lo entendió



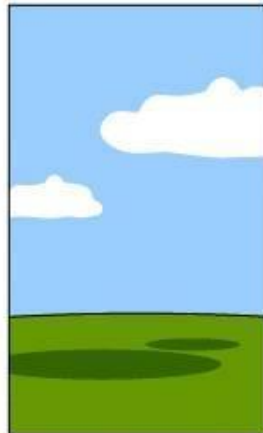
Como el analista lo diseñó



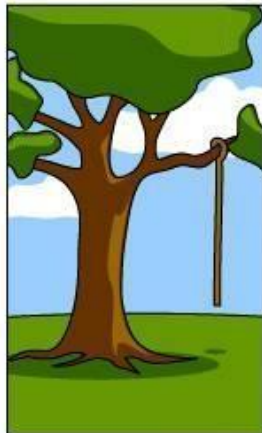
Como el programador lo escribió



Como el vendedor lo describió



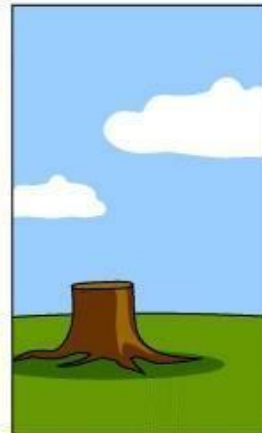
Como fue documentado el proyecto



Que aplicaciones se instalaron



Como le fue facturado al cliente



Como se le dio soporte



Lo que el cliente realmente necesitaba

Conceptos de Modelado Estructural

- **Objeto:** representación de algo que se describe mediante un *identificador*, una *estructura (propiedades)* y un *comportamiento*
- **Propiedades de los objetos:**
 - **Atributos:** propiedades relevantes de un objeto que representa su estructura (diferentes objetos tienen algunos valores diferentes)
 - Simples: un solo valor
 - Compuestos: múltiples valores
 - **Relaciones:** asociaciones entre los objetos

Conceptos fundamentales

- **Clasificación:** agrupación de objetos con propiedades y comportamiento similares dentro de una clase
- **Clase:** definición de la estructura y el comportamiento de un conjunto de objetos que tienen el mismo patrón estructural y de comportamiento 
- **Instancia:** Cada objeto que pertenece a una clase

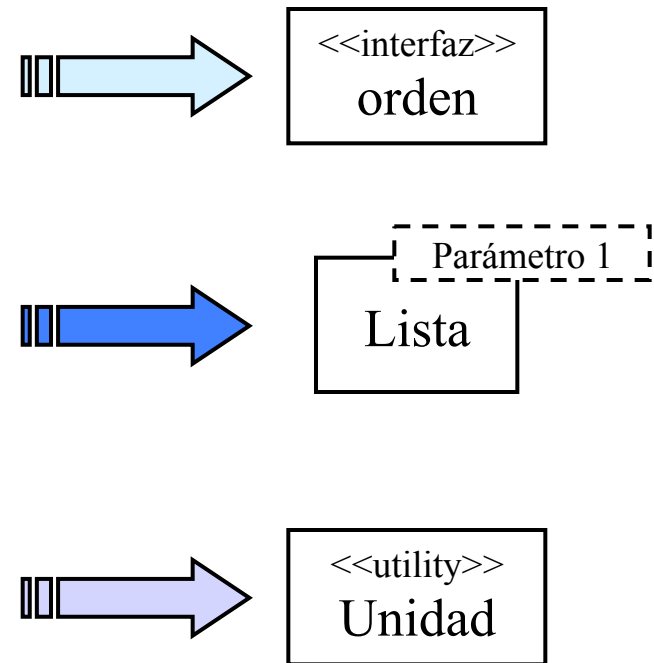
Conceptos fundamentales

- **Extensión de clase:** conjunto de todas las instancias de una clase
- **Generación de objetos**
 - **Instanciación:** proceso de generación o creación de las instancias de una clase
- **Objetos de una clase**
 - **nombre del objeto:** nombre de la clase a la que pertenece el objeto, ambos subrayados
 - Cada atributo definido en la clase tendrá su valor específico

<u>vueloIB6545 : Avion</u>
idAvion : Object enTierra : Boolean = TRUE

Conceptos fundamentales

- **Clases:** definen el vocabulario básico de un modelo y fundamentan el modelado de datos
 - **Interfaz:** clase asociada que describe su comportamiento visible.
 - **Clases abstractas o parametrizables:** modelos de clases que se corresponden con clases genéricas. No son instanciables.
 - **Clases utilitarias:** librerías de funciones

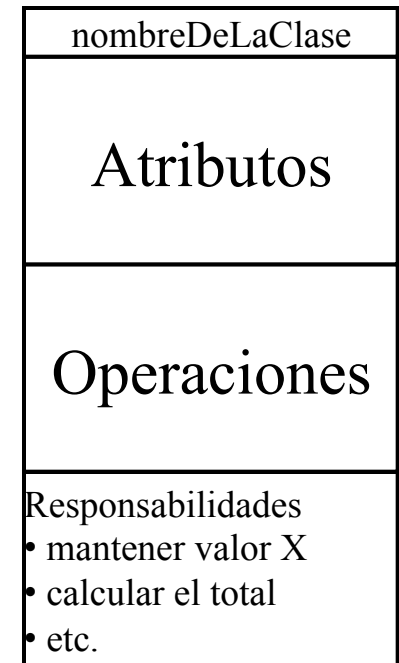


Conceptos fundamentales

● Responsabilidades de una clase

- Representa las obligaciones que una clase tiene en relación al resto de las clases
- Normalmente están soportadas por las operaciones de la clase

- Se pueden agregar compartimentos al final de las clases para representar cualquier información necesaria en el modelo



¿Qué es UML?

Abstracción

- La técnica de hacer un modelo de tus ideas del mundo es el uso de la **abstracción**
 - Por ejemplo, un mapa es un modelo del mundo, no el mundo en miniatura
- En los diagramas UML se muestra una abstracción del sistema, no todo el sistema, con el objetivo de que sea fácil de entender

¿Qué es UML?

Puntos de vista

- UML permite crear diagramas que reflejan diferentes **puntos de vista** del mismo sistema.
 - Por ejemplo, hay mapas físicos, mapas políticos, mapas históricos ... todos sobre el mismo mundo
- Esto permite mostrar ciertos aspectos y ocultar otros para que sean más fáciles de comprender

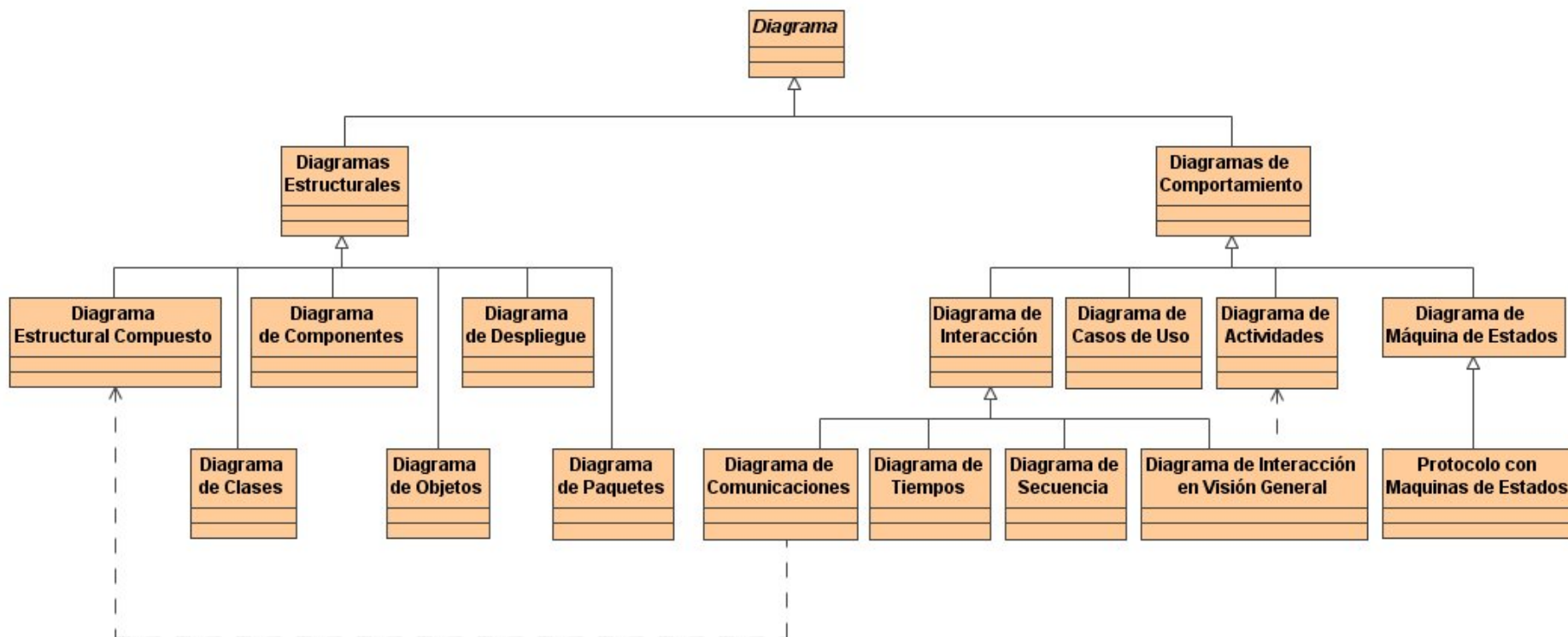
¿Qué es UML?

Tipos de diagramas

- **Diagramas Estructurales:** Muestran los elementos de construcción del sistema. Características que no cambian con el tiempo
- **Diagramas de Comportamiento:** Muestra como el sistema responde a las peticiones o evoluciona con el tiempo.
- **Diagramas de Interacción:** Engloba a ciertos diagramas de comportamiento que muestran el intercambio de mensajes dentro de un grupo de objetos que cooperan (colaboración) para obtener un objetivo

¿Qué es UML?

Tipos de diagramas



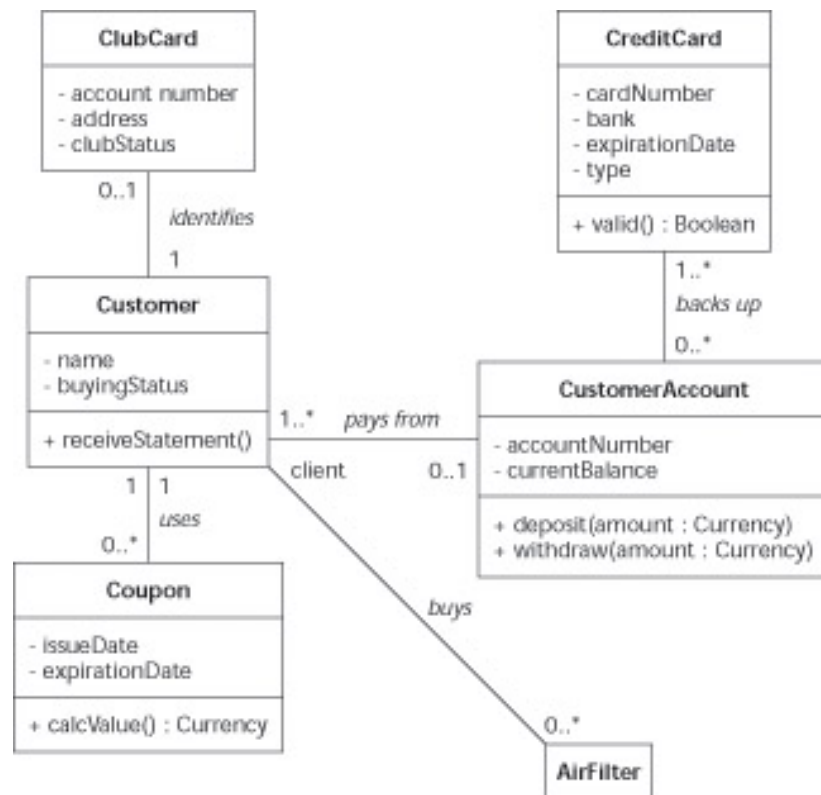
¿Qué es UML?

Tipos de diagramas

● Diagramas Estructurales

– Diagrama de Clases

- Muestra entidades del mundo real, elementos de análisis y diseño o clases de implementación y sus relaciones



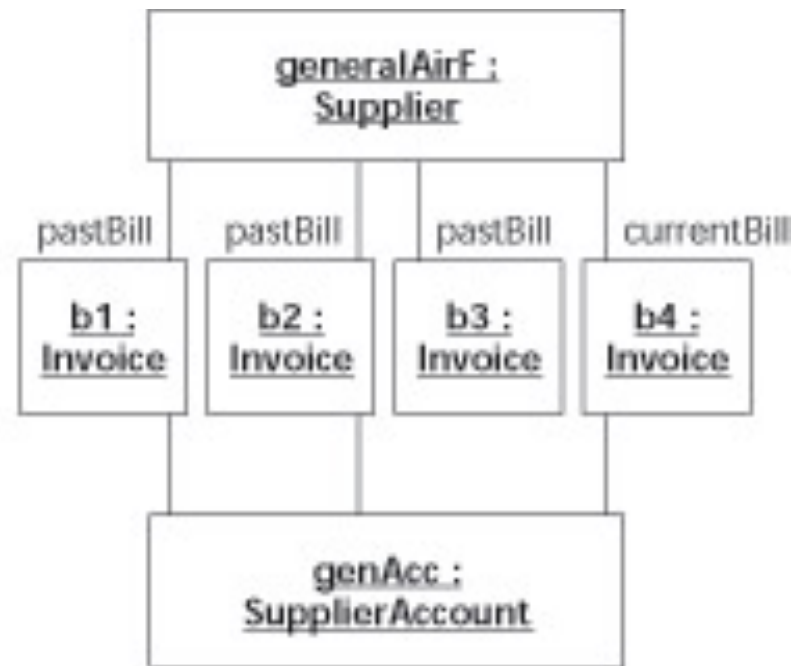
¿Qué es UML?

Tipos de diagramas

● Diagramas Estructurales

– Diagrama de Objetos

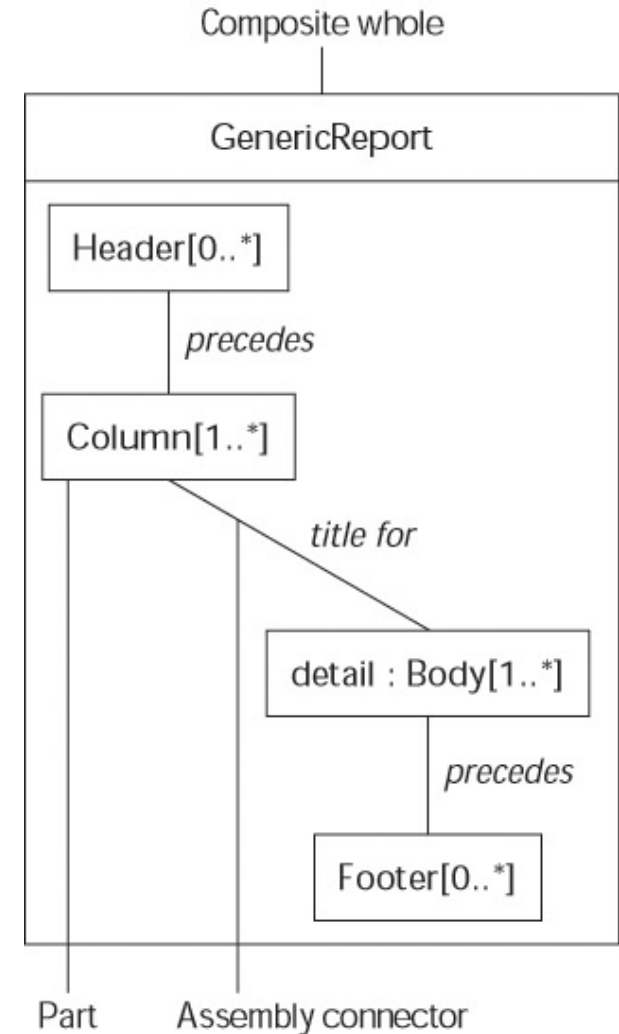
- Muestra un ejemplo ilustrativo de objetos y sus enlaces
- Pueden ser objetos en memoria, objetos en base de datos, objetos de modelado...



¿Qué es UML?

Tipos de diagramas

- Diagramas Estructurales
 - Diagrama Estructural Compuesto
 - Muestra de que está compuesto algún elemento
 - Especialmente útil en estructuras formadas por otras estructuras o diseño basado en componentes



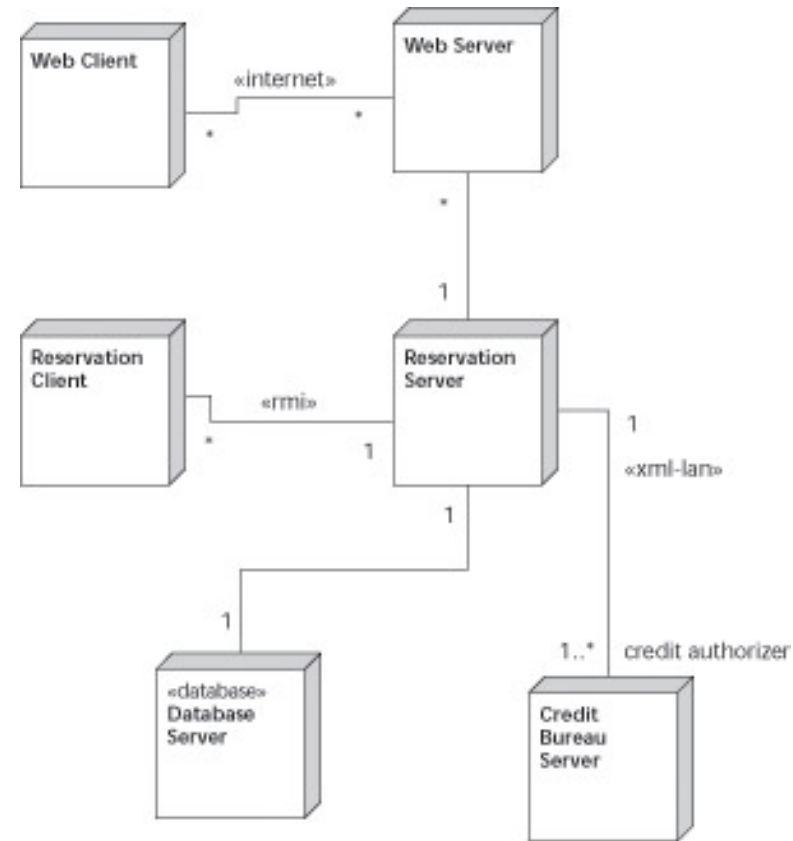
¿Qué es UML?

Tipos de diagramas

● Diagramas Estructurales

– Diagrama de Despliegue

- Muestra la arquitectura cuando el sistema está en ejecución.
- Las plataformas hardware, elementos de software (elementos ejecutables) y los entornos de ejecución (sistemas operativos, máquinas virtuales, ...)



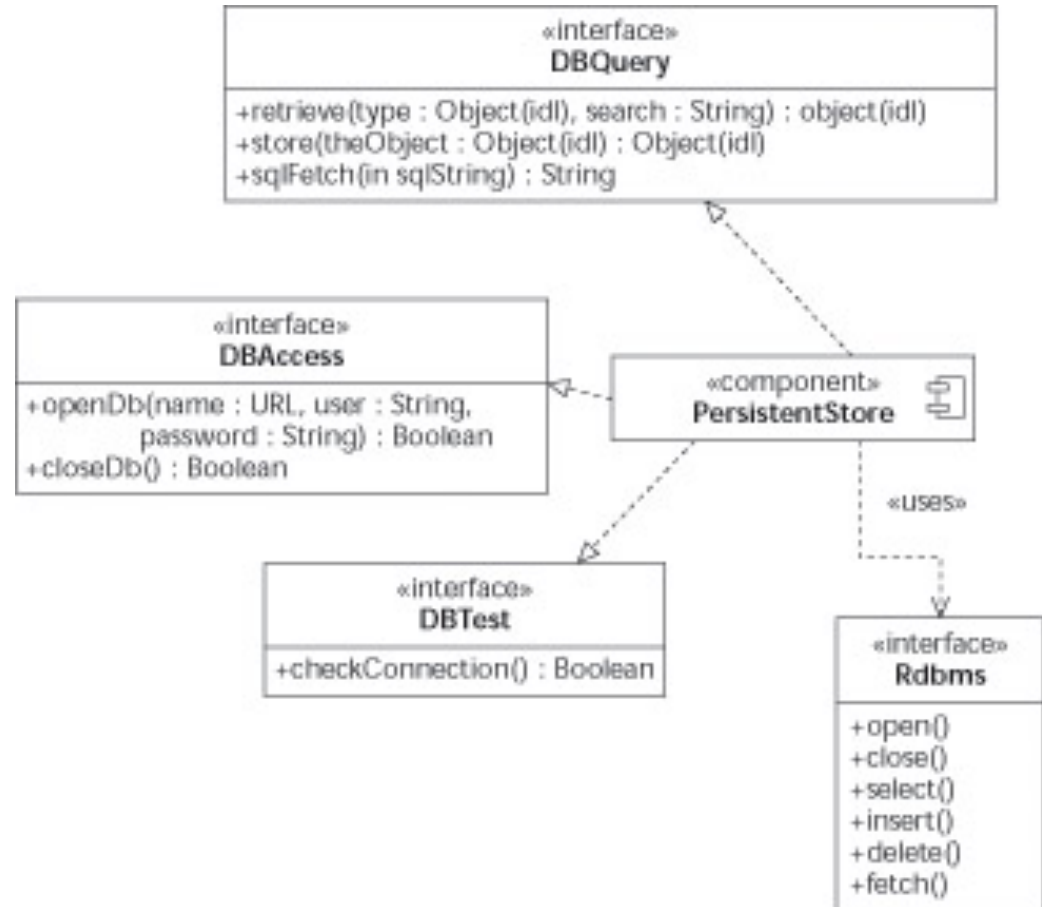
¿Qué es UML?

Tipos de diagramas

● Diagramas Estructurales

– Diagrama de Componentes

- Muestran la organización y las relaciones entre los componentes software del sistema



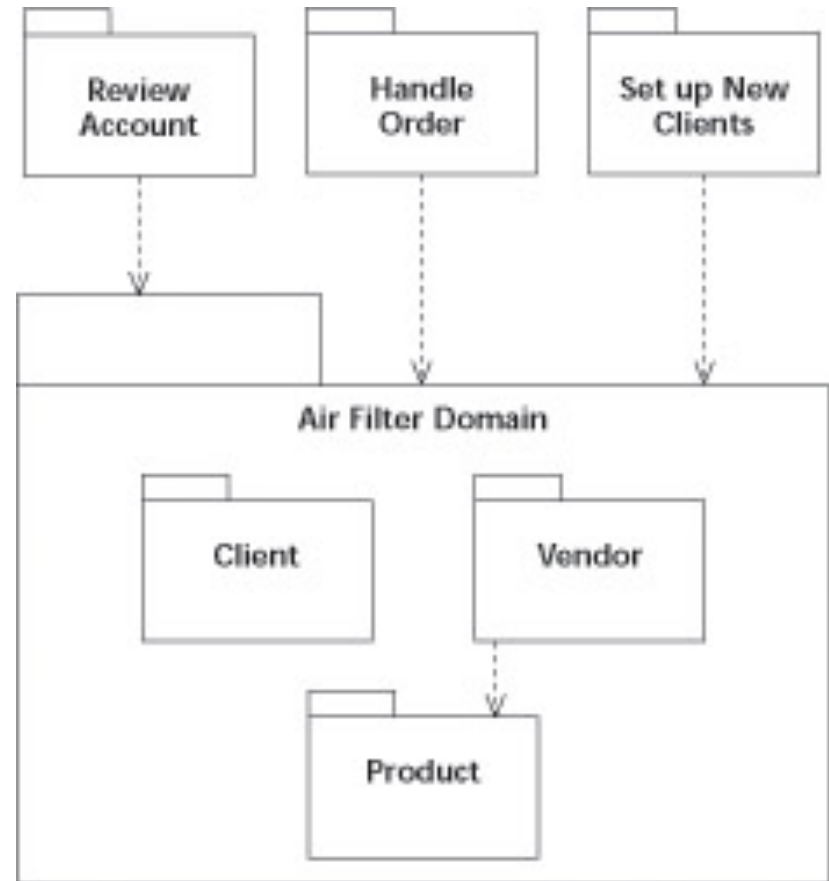
¿Qué es UML?

Tipos de diagramas

- Diagramas Estructurales

- Diagrama de Paquetes

- Agrupa los elementos del modelo y muestra las dependencias entre ellos
- Concepto de paquete similar al de Java



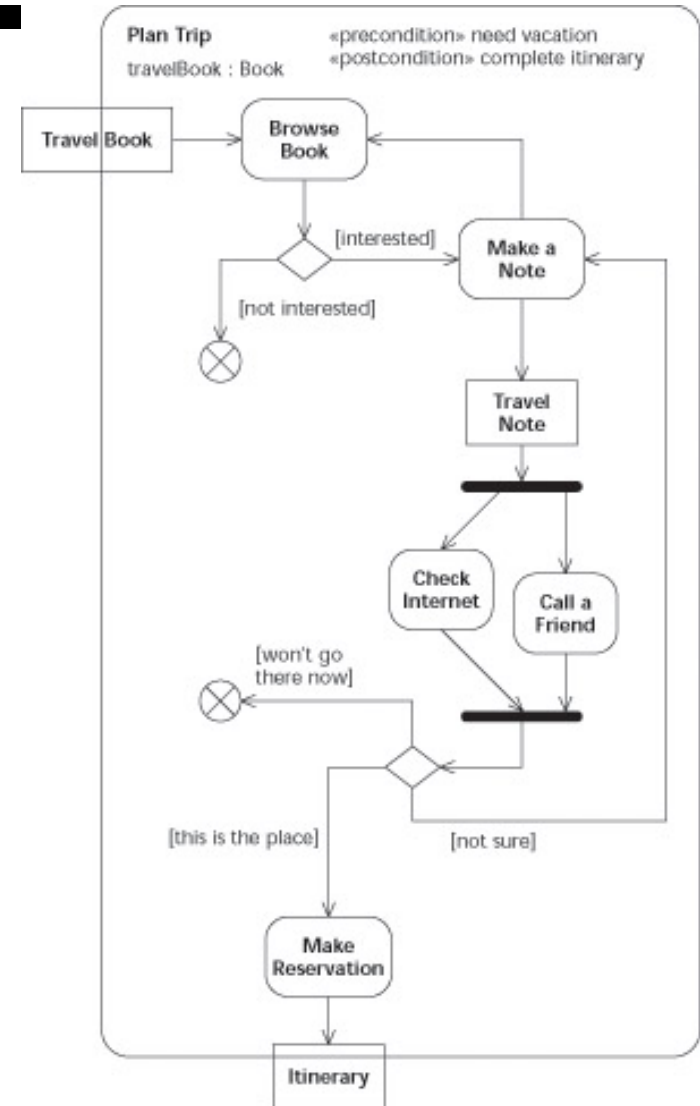
¿Qué es UML?

Tipos de diagramas

- Diagramas de Comportamiento

- Diagrama de Actividades

- Muestra el flujo de datos o el flujo de control
 - Captura el flujo de trabajo de objetos que cooperan



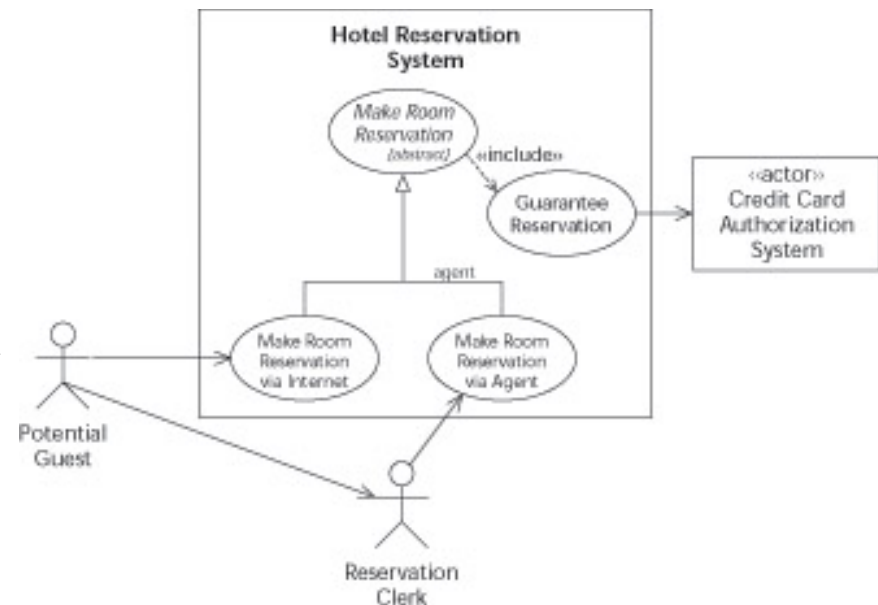
¿Qué es UML?

Tipos de diagramas

- Diagramas de Comportamiento

- Diagrama de Casos de Uso

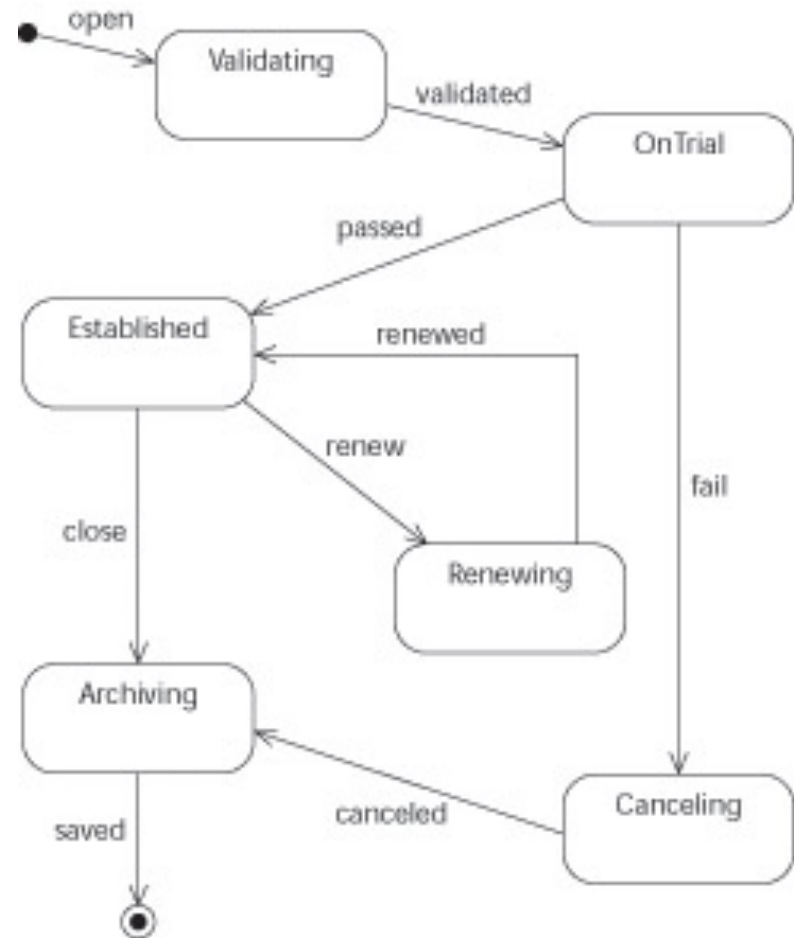
- Muestra los servicios que los actores (usuarios y otros sistemas) pueden pedir al sistema
 - No se muestran en este módulo por estar más relacionados con el Proceso Unificado de Desarrollo



¿Qué es UML?

Tipos de diagramas

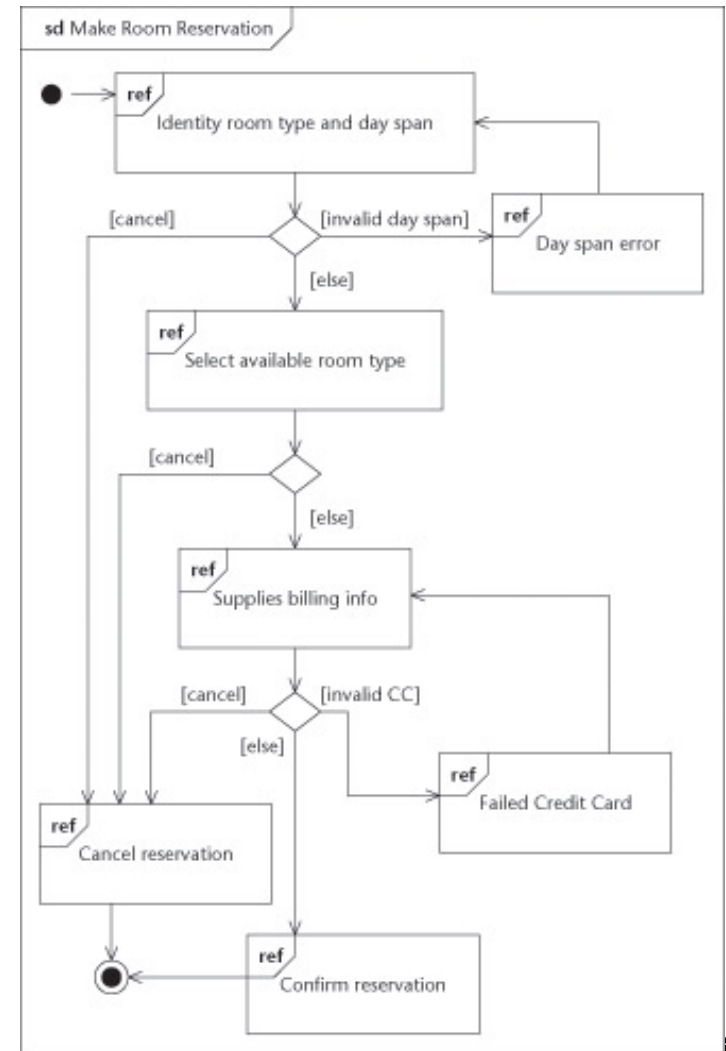
- **Diagramas de Comportamiento**
 - **Diagrama de Máquina de Estados / Protocolos con Máquinas de Estados**
 - Muestra el ciclo de vida de un objeto en particular
 - Los estados en los que se puede encontrar y como transita de uno a otro



¿Qué es UML?

Tipos de diagramas

- Diagramas de Interacción
 - Diagrama de Visión General
 - Muestra muchos escenarios de interacción diferentes para la misma **colaboración** (el conjunto de objetos que trabajan juntos para cumplir un objetivo)



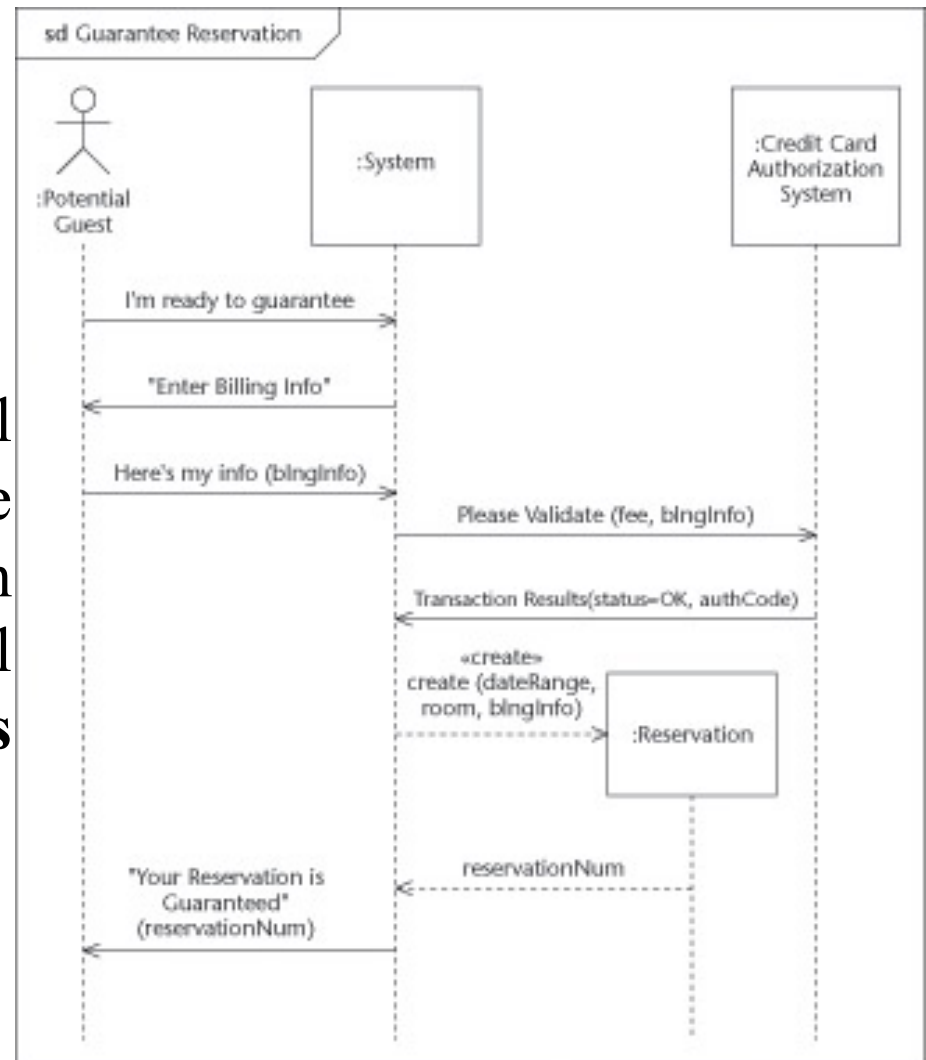
¿Qué es UML?

Tipos de diagramas

- Diagramas de Interacción

- Diagrama de Secuencia

- Se centra en el intercambio de mensajes entre un grupo de objetos y el orden de los mensajes



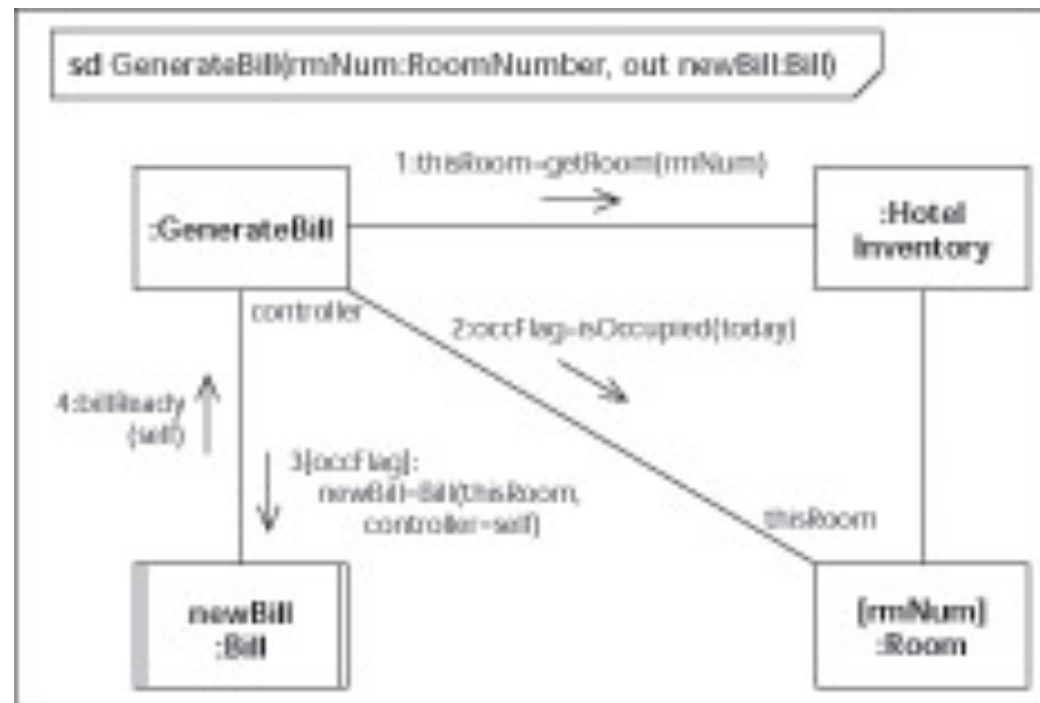
¿Qué es UML?

Tipos de diagramas

● Diagramas de Interacción

– Diagrama de Comunicación

- Se centra en el intercambio de mensajes entre un grupo de objetos y las **relaciones entre los objetos**



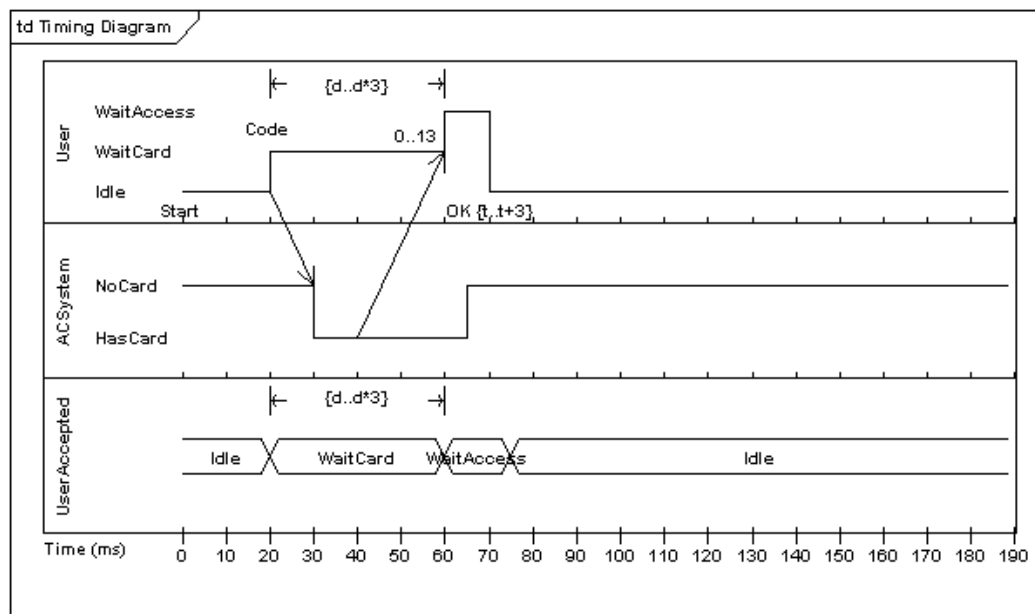
¿Qué es UML?

Tipos de diagramas

● Diagramas de Interacción

– Diagrama de Tiempos

- Muestra los cambios y su relación con el tiempo en sistema de tiempo real



¿Qué es UML?

Diagramas y Modelo

- UML define el formato de un conjunto de diagramas
- Un modelo representa los elementos del sistema
 - Clases con sus métodos y atributos
 - Objetos con sus relaciones
- Cuando existe un conjunto “consistente” de diagramas se forma un modelo
- Es posible que elementos del modelo no aparezcan en ningún diagrama

El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de clases

- Los diagramas de clases están formados por

- Clases

- Atributos (Propiedades)

- Métodos (Operaciones)

- Relaciones

- Asociación

- Herencia

- Composición

- Uso (Dependencia)

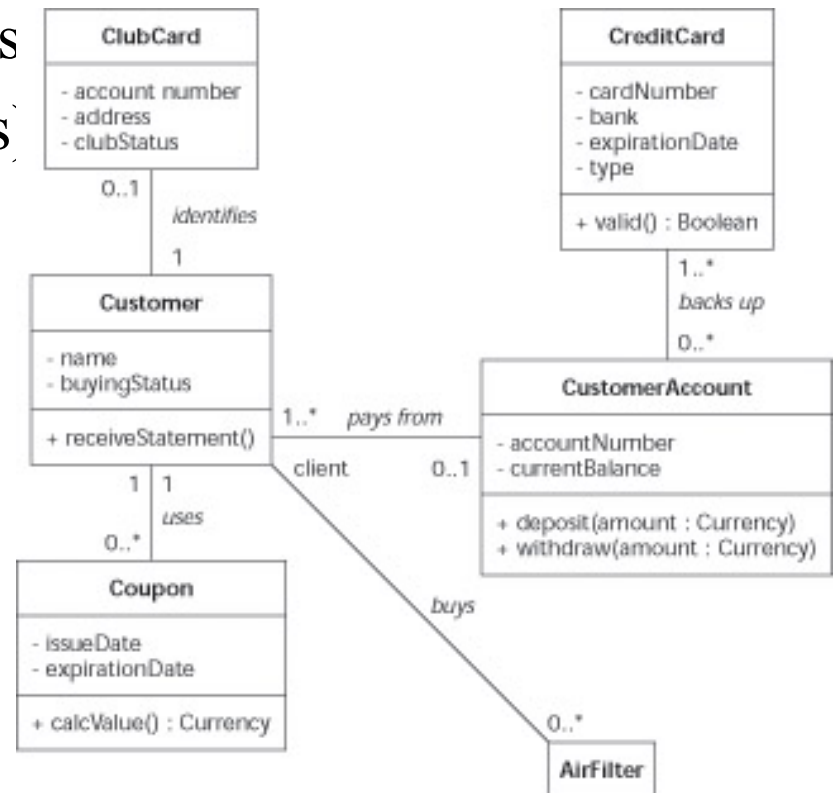


Diagrama de clases

- Formato gráfico de una clase

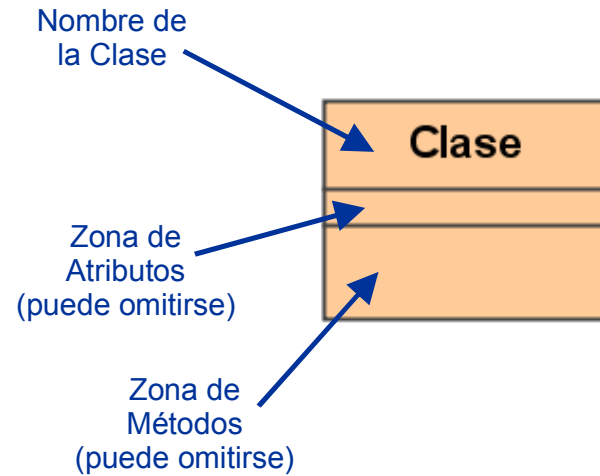


Diagrama de clases

Atributos (Propiedades)

- Sintáxis Atributos

```
nombreAtributo : tipoAtributo
```

- Tipos

- Tipos UML: Integer, Boolean, String
- Tipos de cualquier lenguaje de programación
- Clases del modelo
- Se pueden omitir en bocetos

- Ejemplos

```
numPies: Integer  
nombre: String
```

Diagrama de clases

Atributos (Propiedades)

- Valores por defecto

nombreAtributo : tipoAtributo = valorPorDefecto

numPies: Integer = 5
nombre: String = 'Pepe'

- Multiplicidad (Arrays)

nombreAtributo : tipoAtributo [Multiplicidad]

numPies: Integer [0..1]
nombres: String [1..*] = ('Pedro', 'Juan', 'Antonio')
fechas: Date [2,3,4]

Diagrama de clases

Métodos (Operaciones)

- Sintaxis Métodos

```
nombreOperacion(parametros) : tipoDevuelto
```

- Ejemplos

```
rentOutDummy (aDummy:CrashDummy,forClient:Person): SuccessKind  
acceptProposal ():Boolean  
rentOutDummy(aDummy, toPerson)
```

Diagrama de clases

Visibilidad

Símbolo	Visibilidad
+	Público
-	Privado
#	Protegido
~	Paquete

● Ejemplos

Clase
-numPies : Integer [0..1] +nombres : String [0..*]
+dameAlgo(parametro : boolean, parametro2 : Integer) : double #dameAlgo(parametro : boolean) : int

Diagrama de clases

Miembros Estáticos

- Los miembros (atributos o métodos) estáticos se subrayan

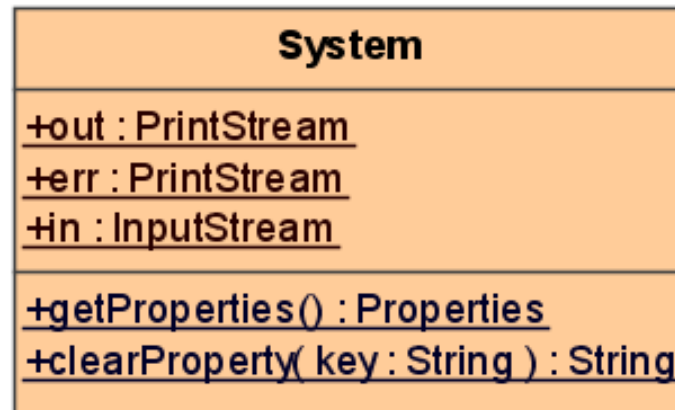


Diagrama de clases

Elementos Abstractos

- Los elementos abstractos aparecen en cursiva (Clases o métodos)

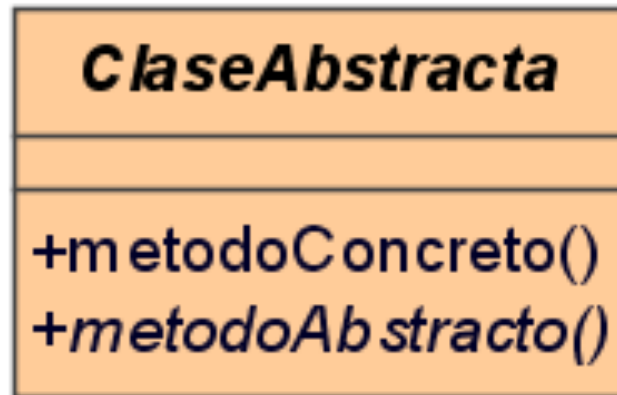


Diagrama de clases

Relación de Asociación

- Una asociación se forma al unir dos clases con una línea
- Puede tener nombre y se coloca sobre la línea (se lee de izq a derecha y suele ser un verbo)
- Puede tener los roles que juega cada clase en la asociación

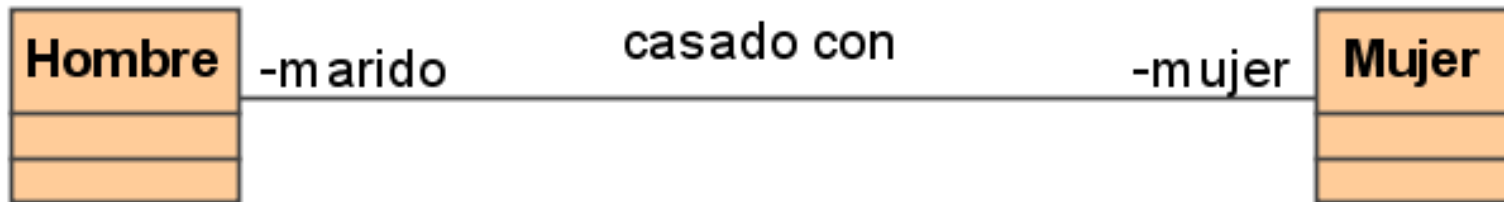


Diagrama de clases

Relación de Asociación

- Puede tener Multiplicidad (*, 0..1, 1..*)
- Puede tener restricciones entre { }
- Se puede indicar el sentido de la lectura

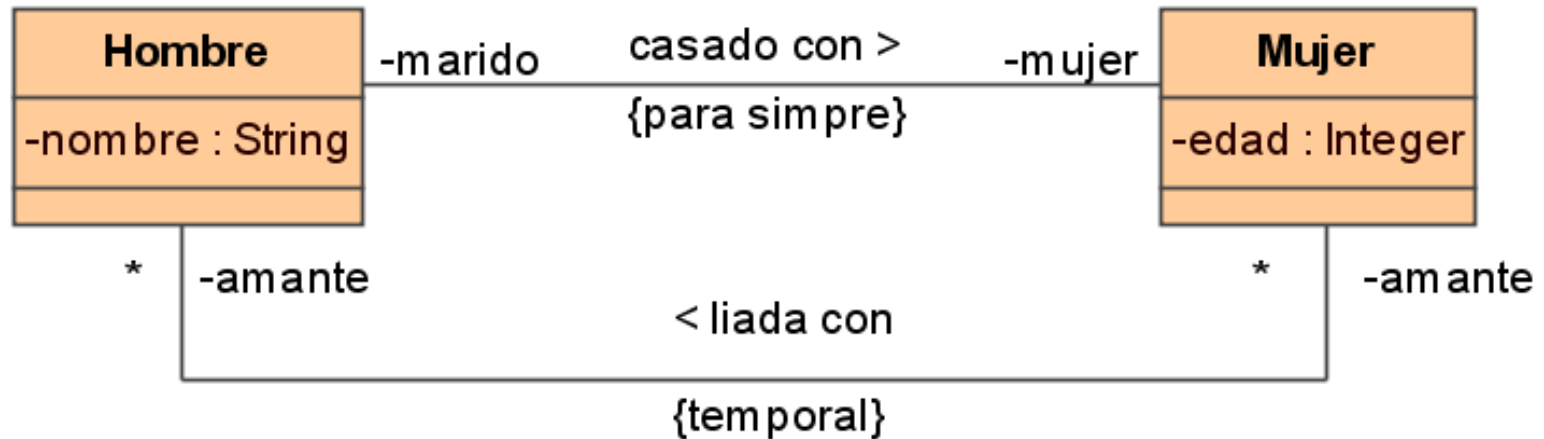


Diagrama de clases

Relación de Asociación

- Puede tener navegación direccional (sólo es posible la navegación en un sentido)
- Es posible que una clase se asocie consigo misma (asociación reflexiva)

```
public class Nodo {
    List<Nodo> hijos;
    Object valor
}
```

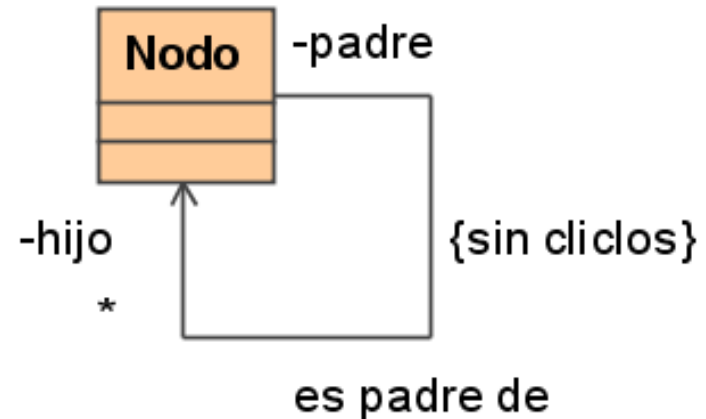


Diagrama de clases

Relación de Asociación

● Clases de Asociación

- Similares a las relaciones atribuidas de BBDD
- La clase de asociación debe llamarse como la asociación

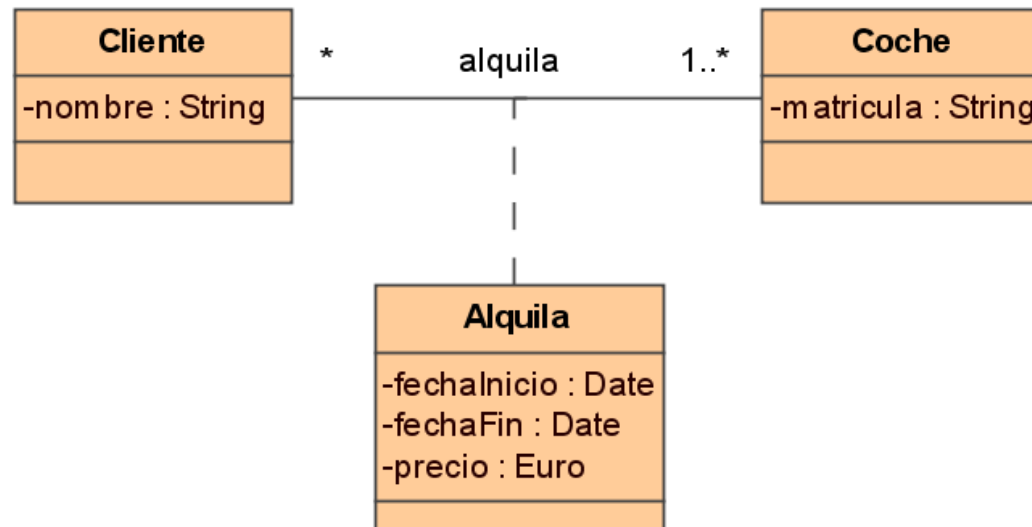


Diagrama de clases

Estereotipos

- Los estereotipos son marcas asociadas a cualquier elemento de un diagrama (cualquier tipo de diagrama)
- Indican que el elemento marcado es algo diferente que el elemento sin marcar
- Los estereotipos pueden usarse por las herramientas de generación de código

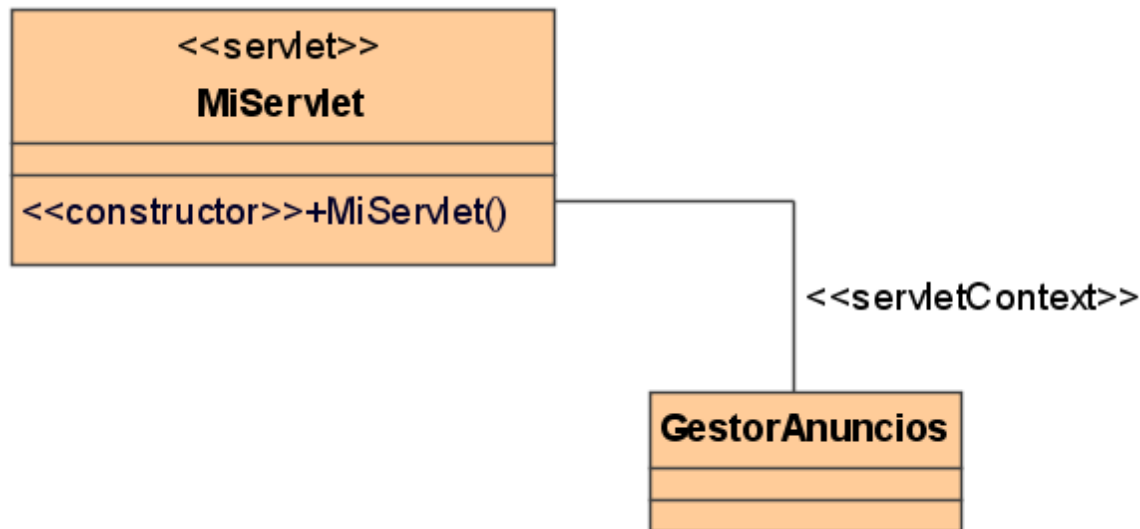


Diagrama de clases

Relación de Agregación

- Una **relación** de agregación es la que forma un **todo con sus partes**
- Son un tipo especial de relación de asociación
- Pueden tener nombre, roles, multiplicidad, ...
- En las relaciones de agregación, un objeto que representa una **parte** puede estar **compartido** por varios objetos que representan el **todo** (un alumno está en un curso y también puede estar en un grupo de amigos)

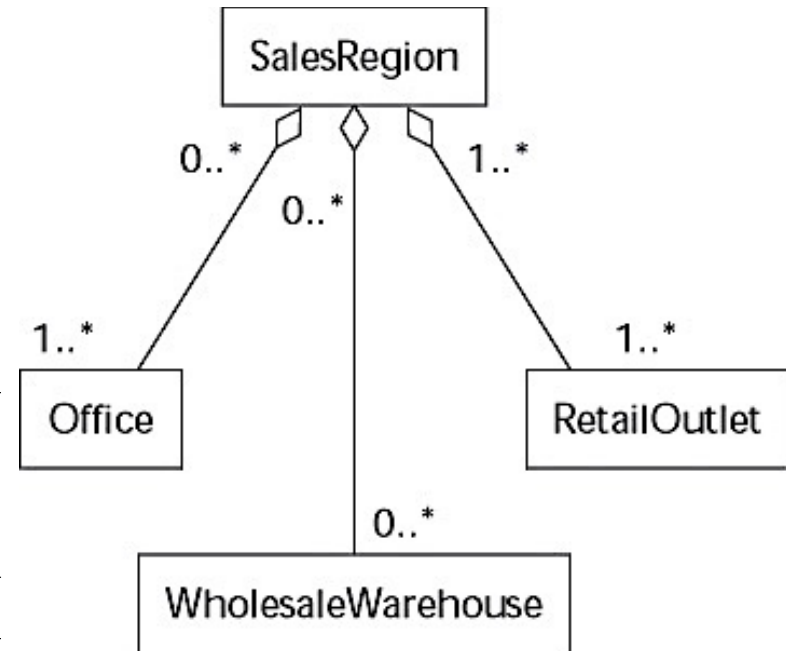


Diagrama de clases

Relación de Composición

- Las relaciones de **composición** son un tipo especial de relación de **agregación**
- Los objetos **parte** siempre están asociados a **un** objeto **todo** y sólo a uno, se crean y se destruyen con él (coche y ruedas).
- Los objetos **parte** no pueden **compartirse** entre varios objetos **todo**.

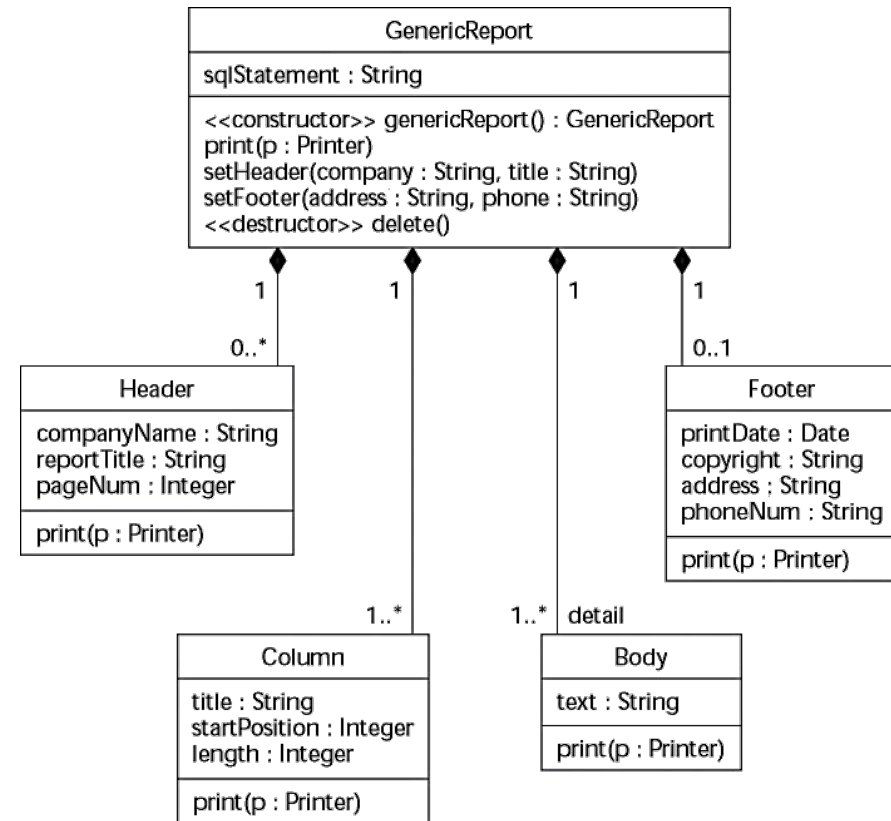


Diagrama de clases

Relación de Herencia

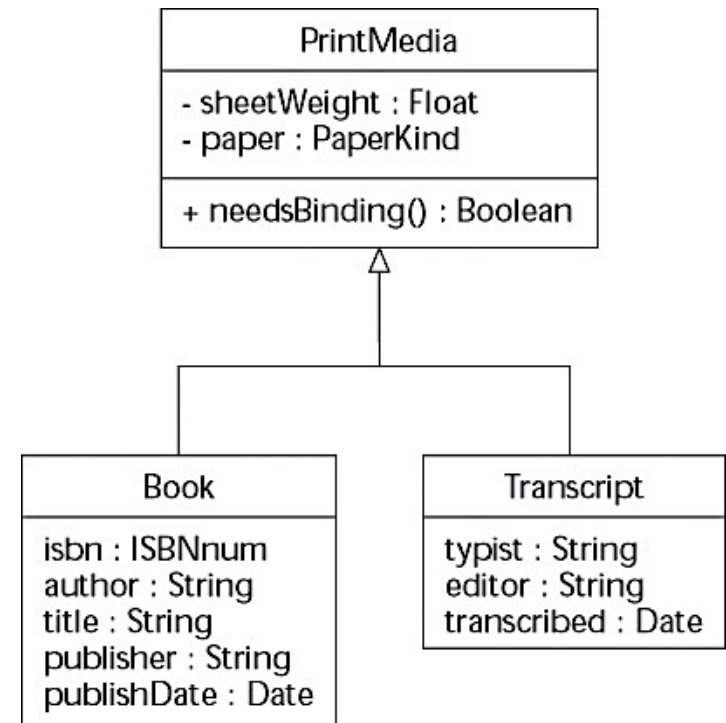
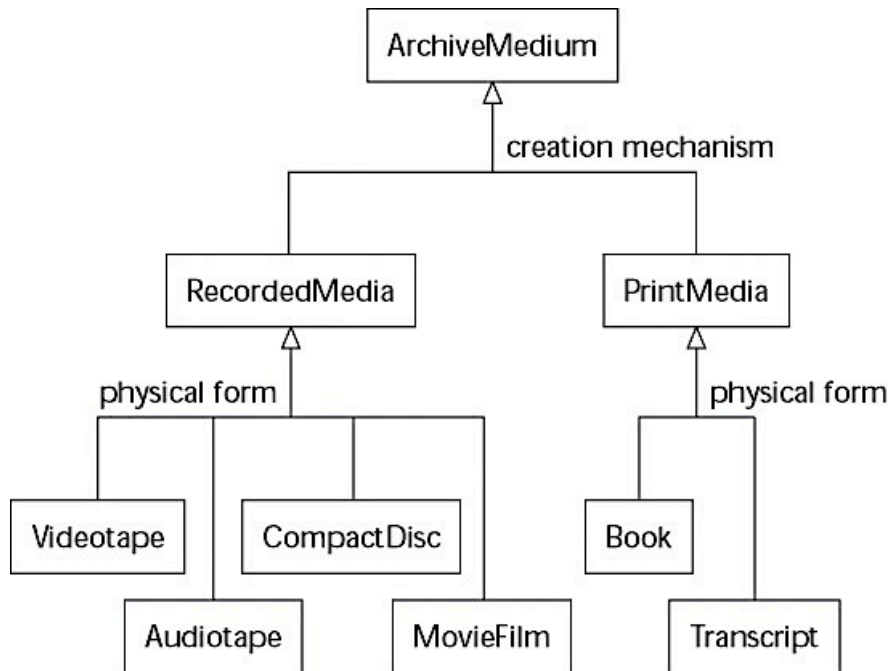


Diagrama de clases

Relación de Dependencia

- Muestra la dependencia entre una clase y otra
- Existe cuando los cambios en la clase independiente pueden afectar a la clase que depende
- Relación de uso

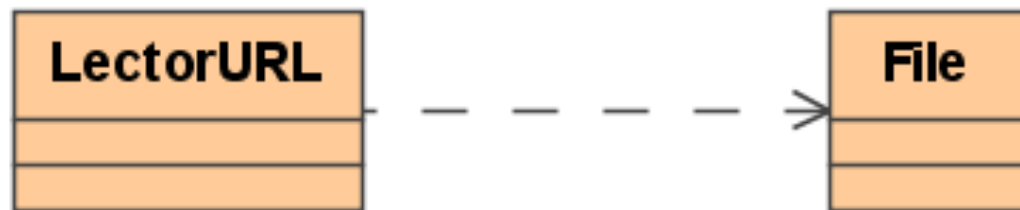


Diagrama de clases

Interfaces

- Tienen dos representaciones gráficas
- Se relacionan con las clases que los implementan con una relación de **realización**

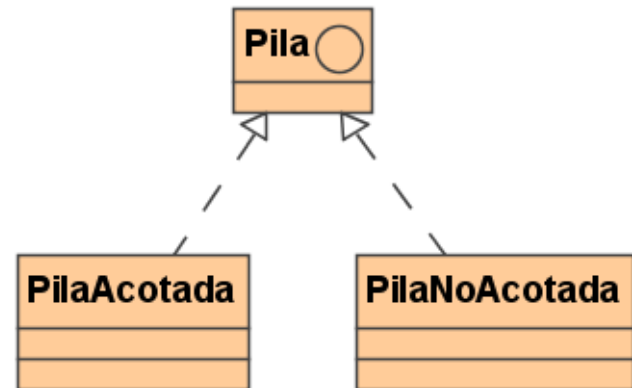
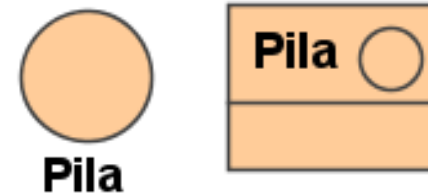
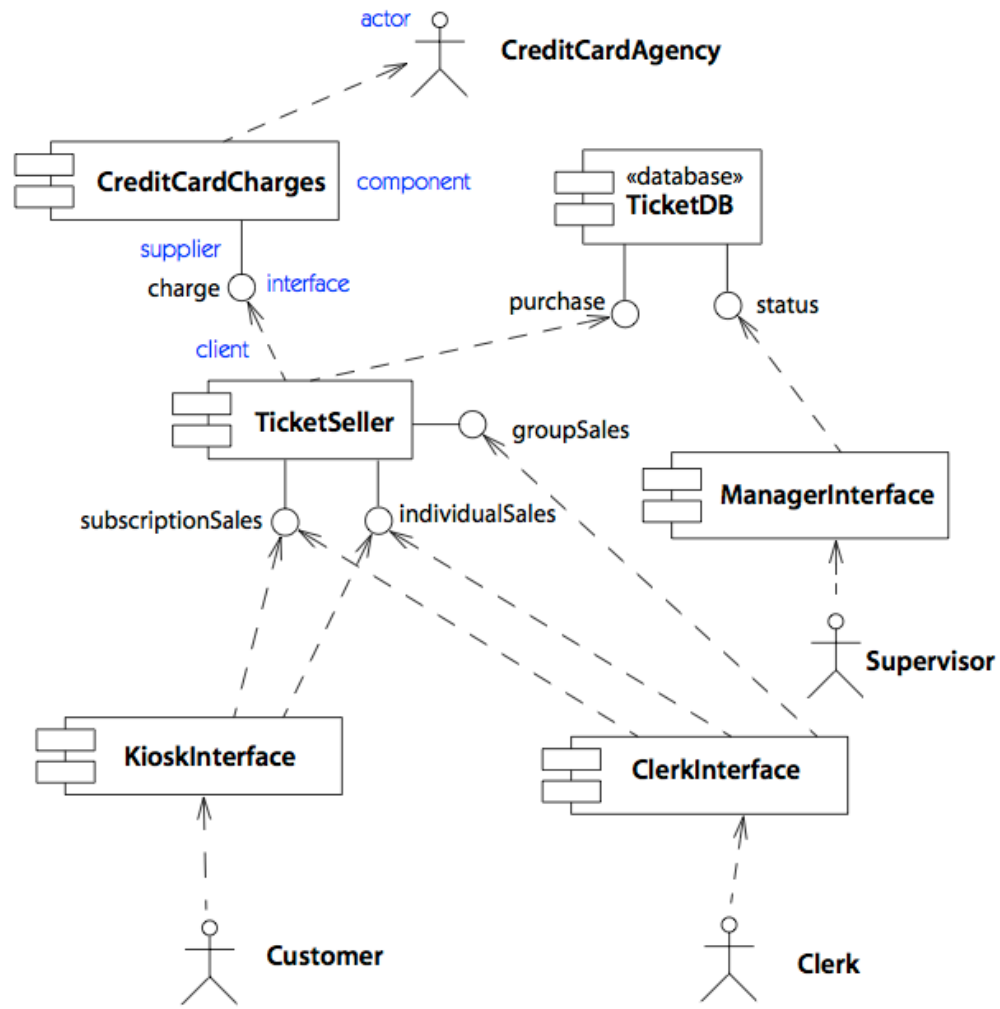


Diagrama de clases

Interfaces

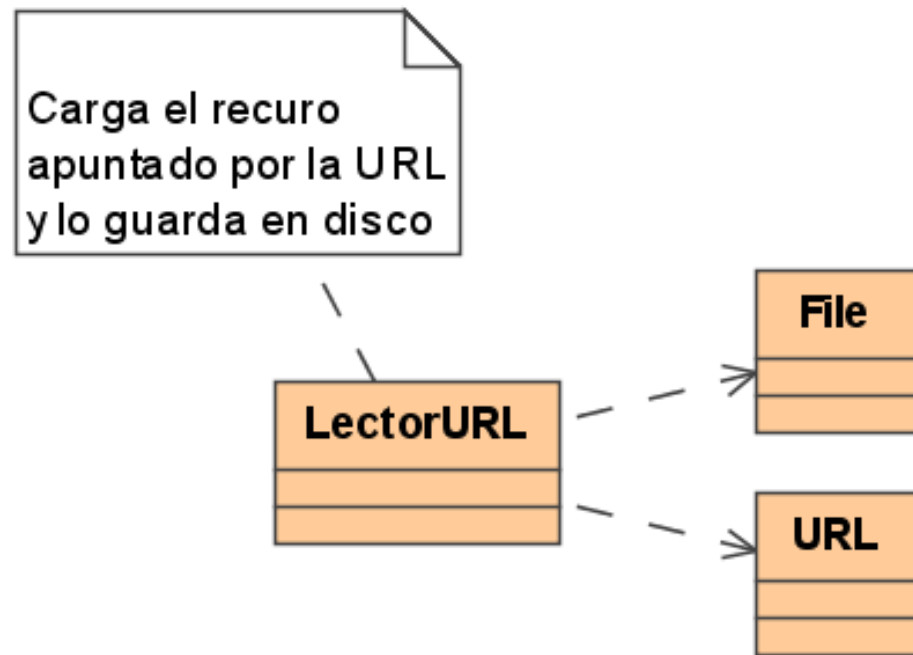


El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- **Notas**
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Notas

- Las notas pueden aparecer en cualquier diagrama



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- **Diagrama de Paquetes**
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue

Diagrama de Paquetes

- Muestran los paquetes de un sistema y la dependencia entre unos y otros

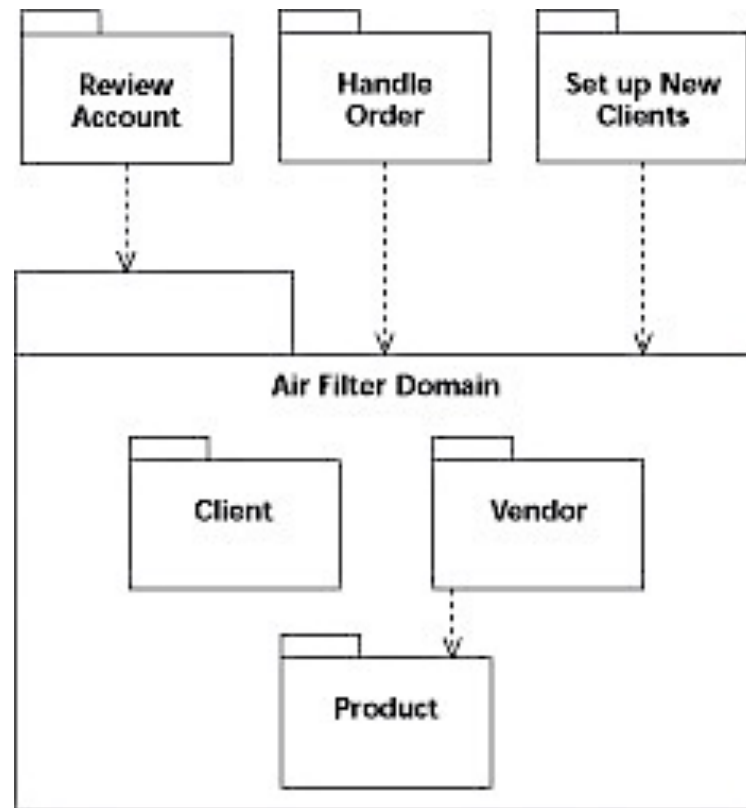


Diagrama de Paquetes

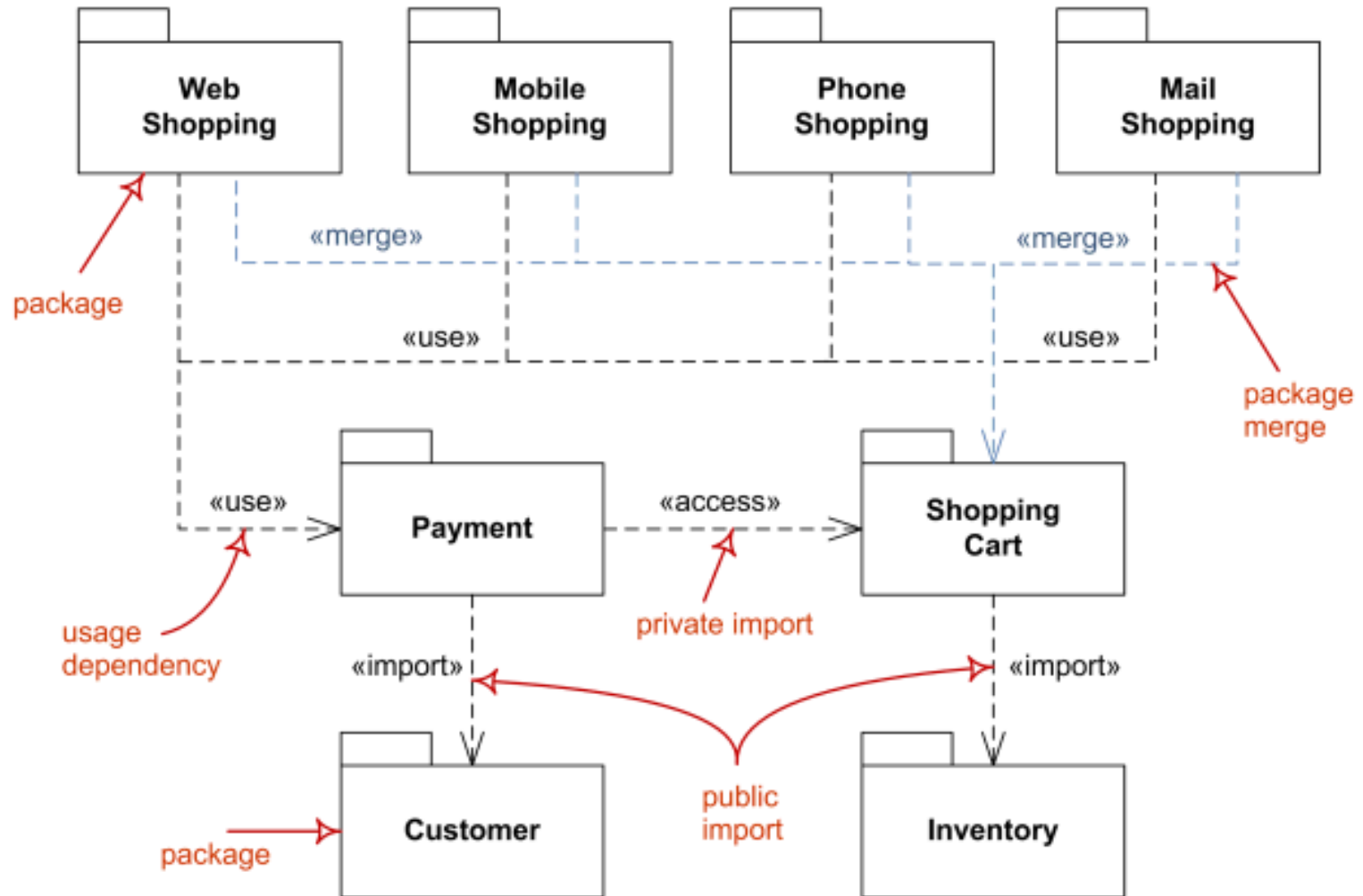
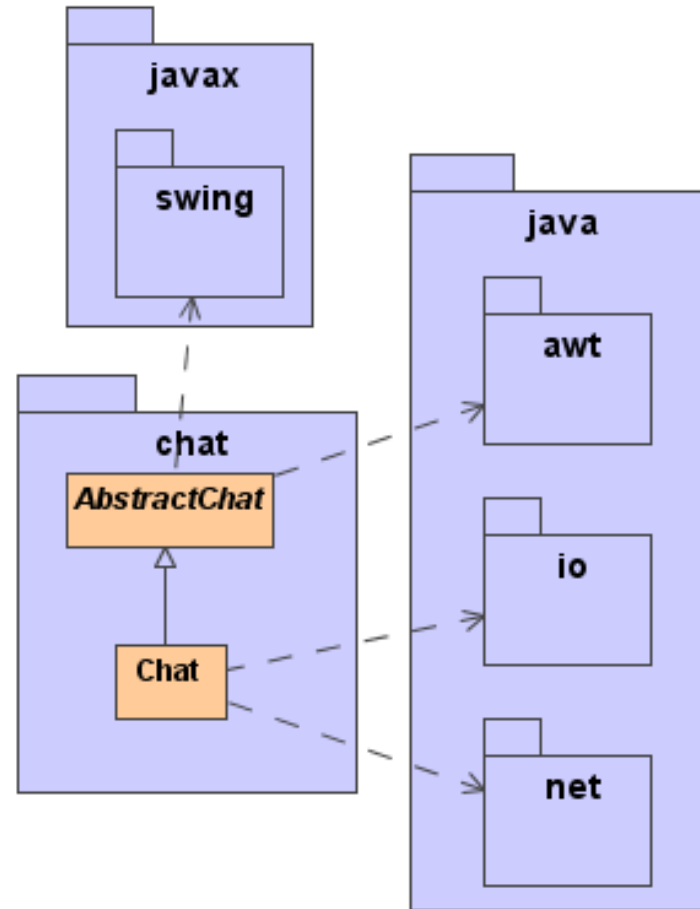


Diagrama de Paquetes

- Los diagramas de paquetes se combinan con los diagramas de clases



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- **Diagrama de Objetos**
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de Objetos

- Se usan para mostrar el estado del sistema en un momento concreto
- Si en el diagrama de clases aparece una asociación, en el diagrama de objetos puede aparecer un **enlace**

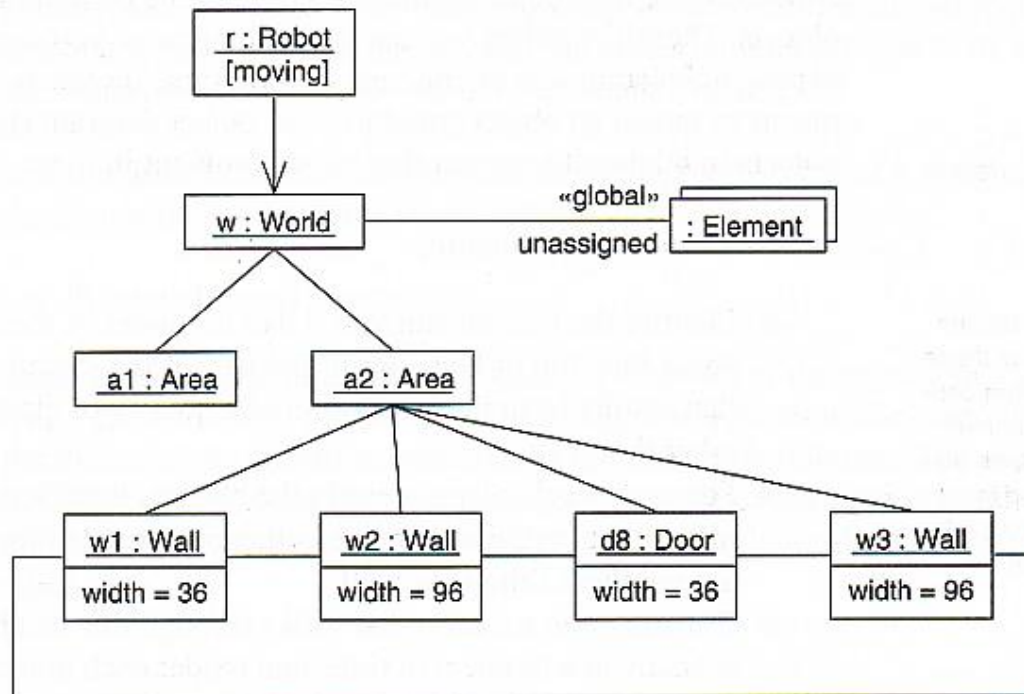


Diagrama de Objetos

- Formato gráfico de un objeto
 - Nombre del Objeto (se puede omitir)
 - Clase del objeto (se puede omitir)
 - Valores de los atributos (se pueden omitir)

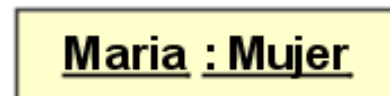
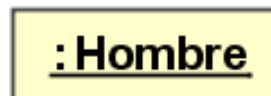
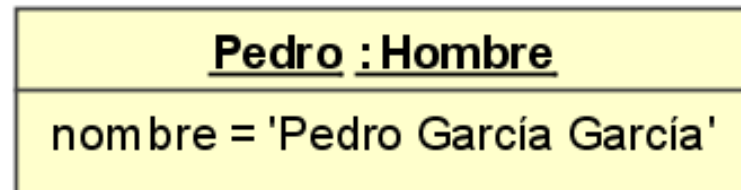
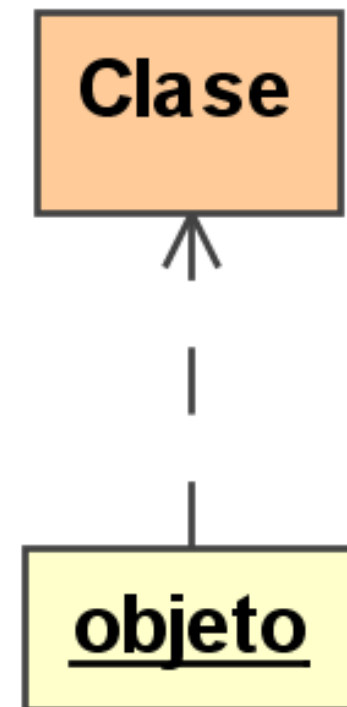


Diagrama de Objetos

- Se puede relacionar un objeto con su clase

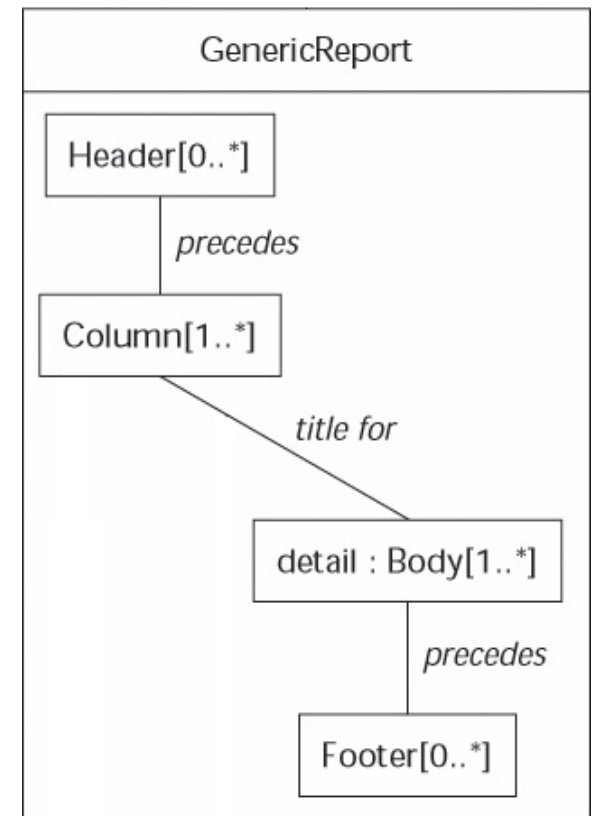
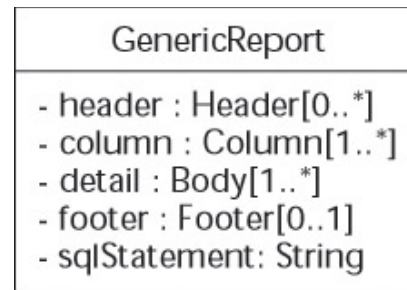
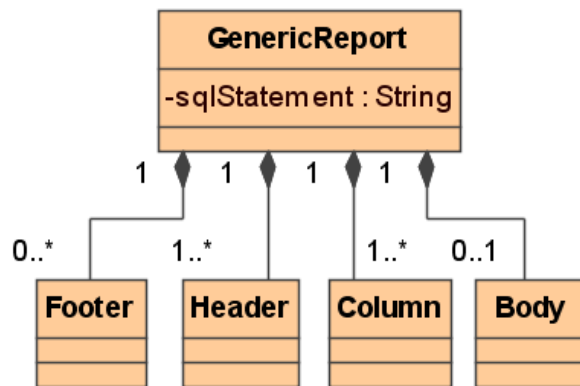


El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- **Diagrama Estructural Compuesto**
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama Estructural Compuesto

- Es una forma alternativa de mostrar una relación de composición incluyendo las clases parte dentro de la clase todo

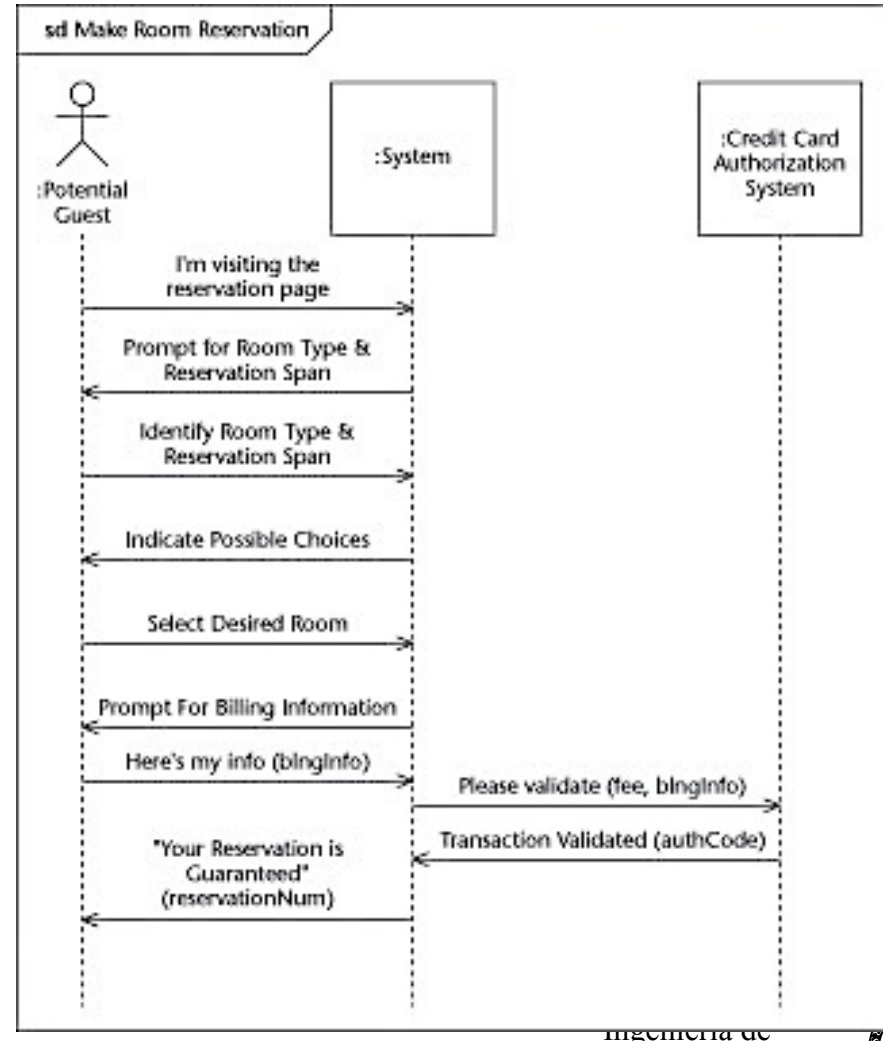


El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- **Diagrama de Secuencia**
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de Secuencia

- Los diagramas de secuencia muestran como los objetos se intercambian mensajes a lo largo del tiempo
- Los mensajes aparecen en orden cronológico desde la parte superior del diagrama



Diagramas de Secuencia

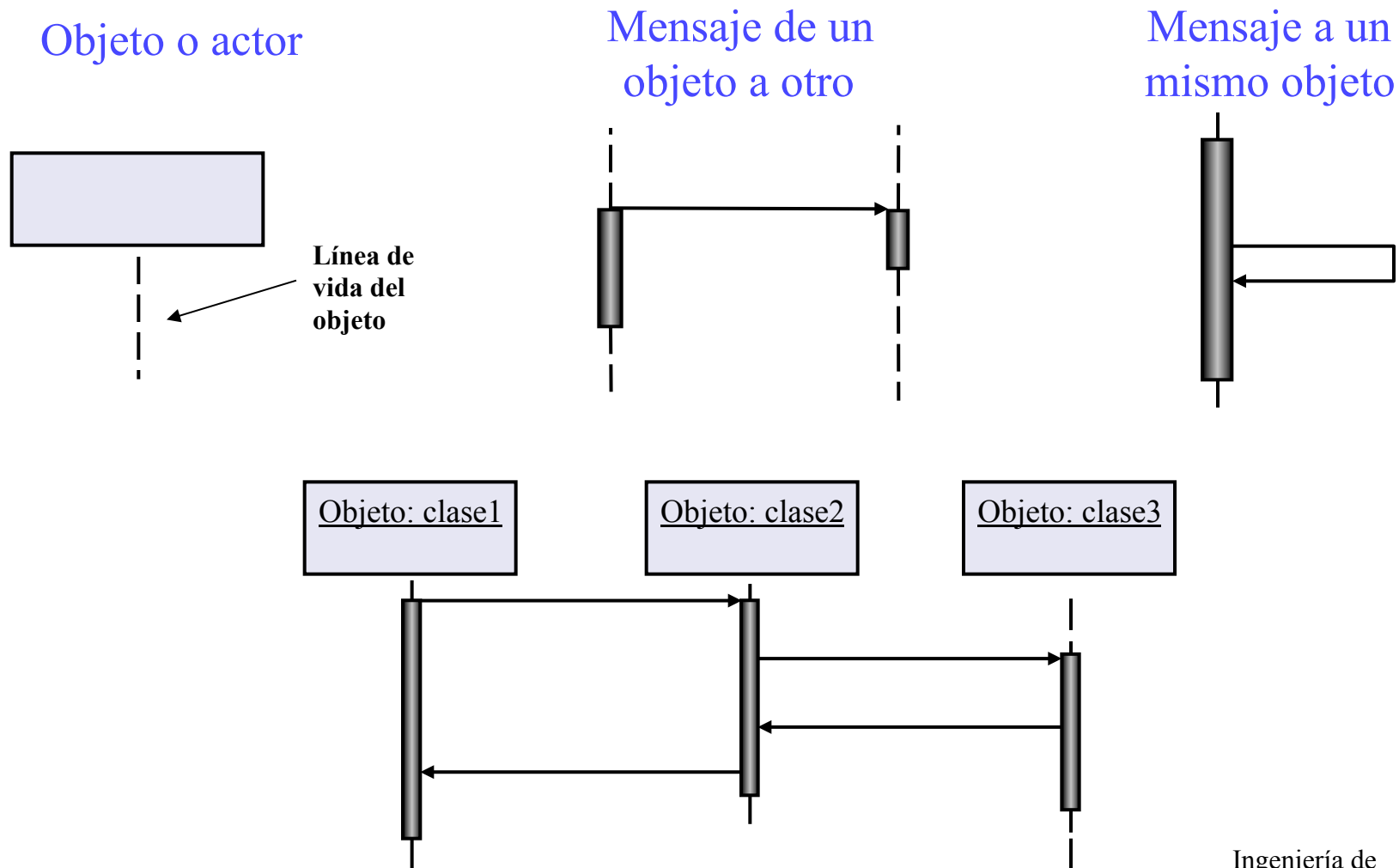


Diagrama de Secuencia

- Los mensajes pueden estar escritos en lenguaje natural o con una sintaxis más precisa y cercana a los lenguajes de programación dependiendo del nivel de abstracción

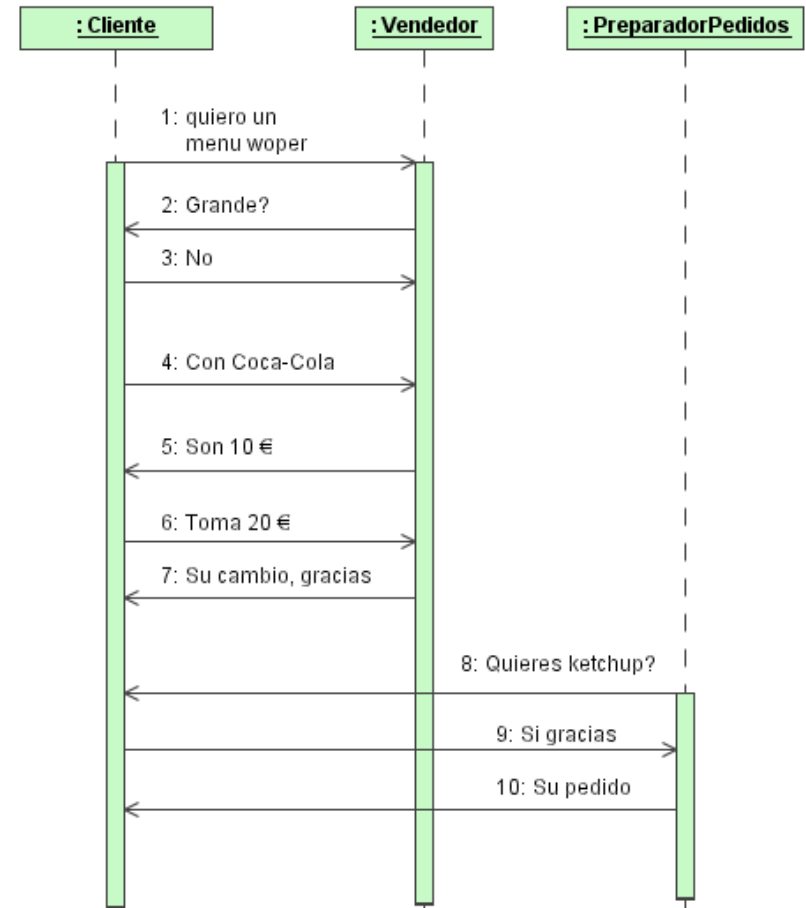


Diagrama de Secuencia

- Sintaxis mensaje

mensaje (params)

- Sintaxis parámetro

dirección nombreParametro: Tipo [Multiplicidad] = valorDefecto

- Dirección: in, out, inout (por defecto in)
- Para no especificar un parámetro se pone -

- Ejemplo

Transaction Results (status=OK, authCode)

Diagrama de Secuencia

- Tipos de mensajes
 - Mensajes en un diagrama alto nivel
- Mensajes en diagramas de bajo nivel
 - Síncronas (p.e llamadas a métodos)



Diagrama de Secuencia

- Cuando se usan mensajes síncronos (llamadas a métodos) se puede poner explícitamente el retorno (return) o se puede omitir

```
File fichero = ...
```

```
String ruta = fichero.getPath();
```

```
System.out.println("Ruta: "+ruta);
```

```
long tamaño = fichero.length();
```

```
System.out.println("Tamaño: "+tamaño);
```

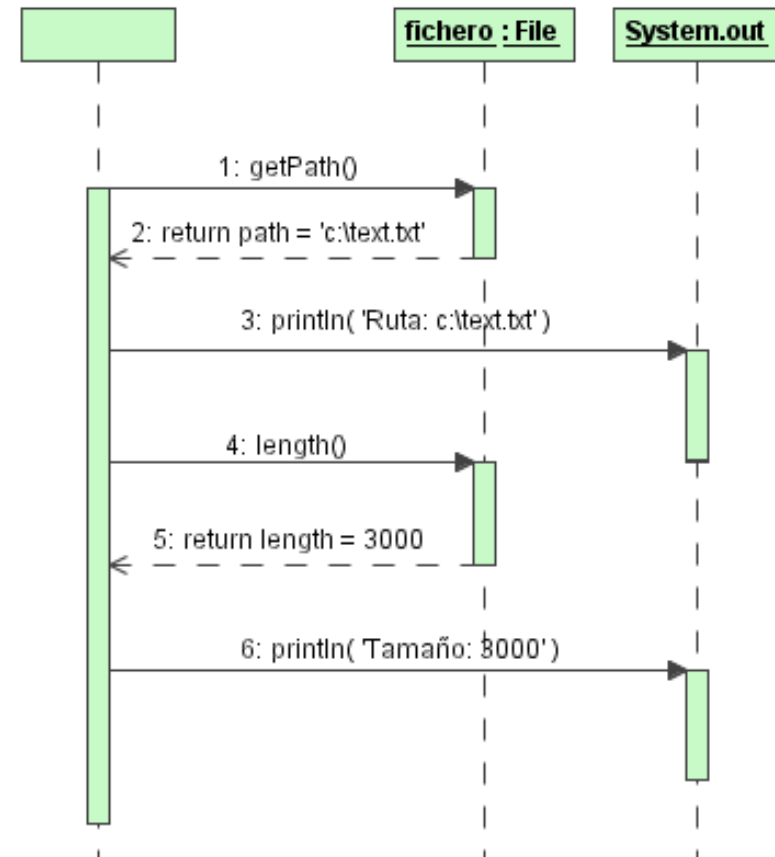


Diagrama de Secuencia

- Hay veces que los objetos no existen durante todo el tiempo
- Se usan mensajes de creación y de destrucción

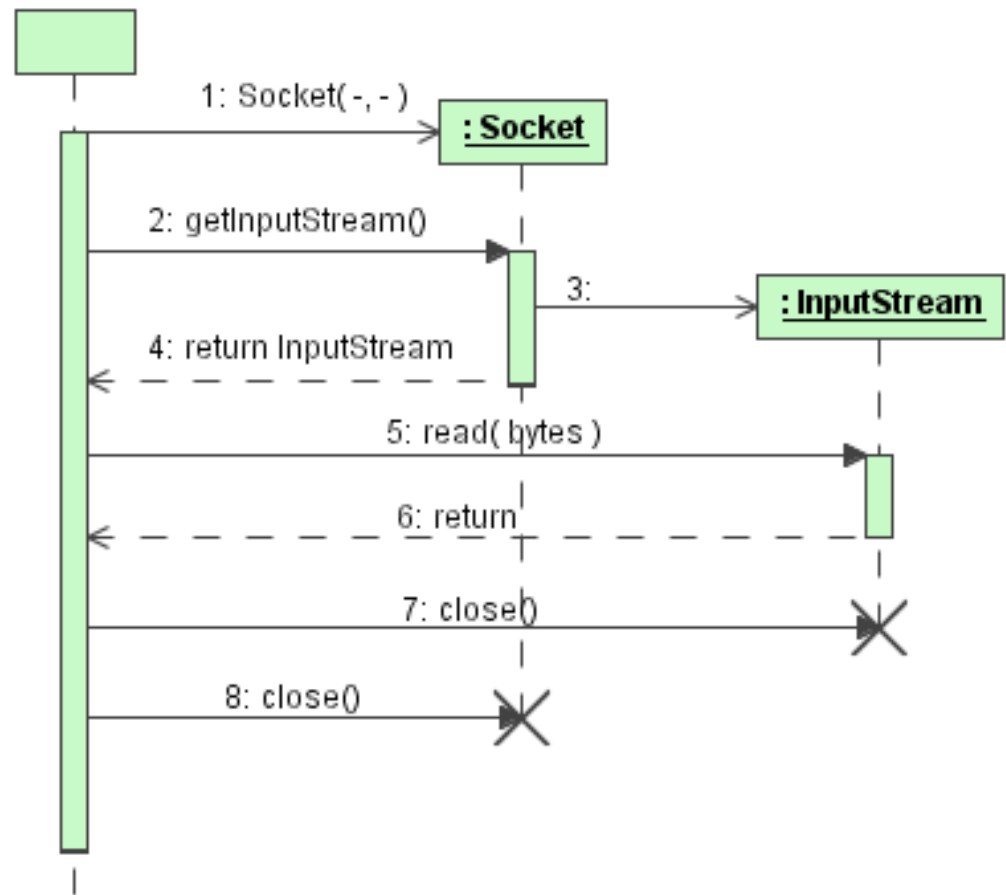


Diagrama de Secuencia

- Llamadas recursivas (autodelegación)
- Nuevo hilo de ejecución

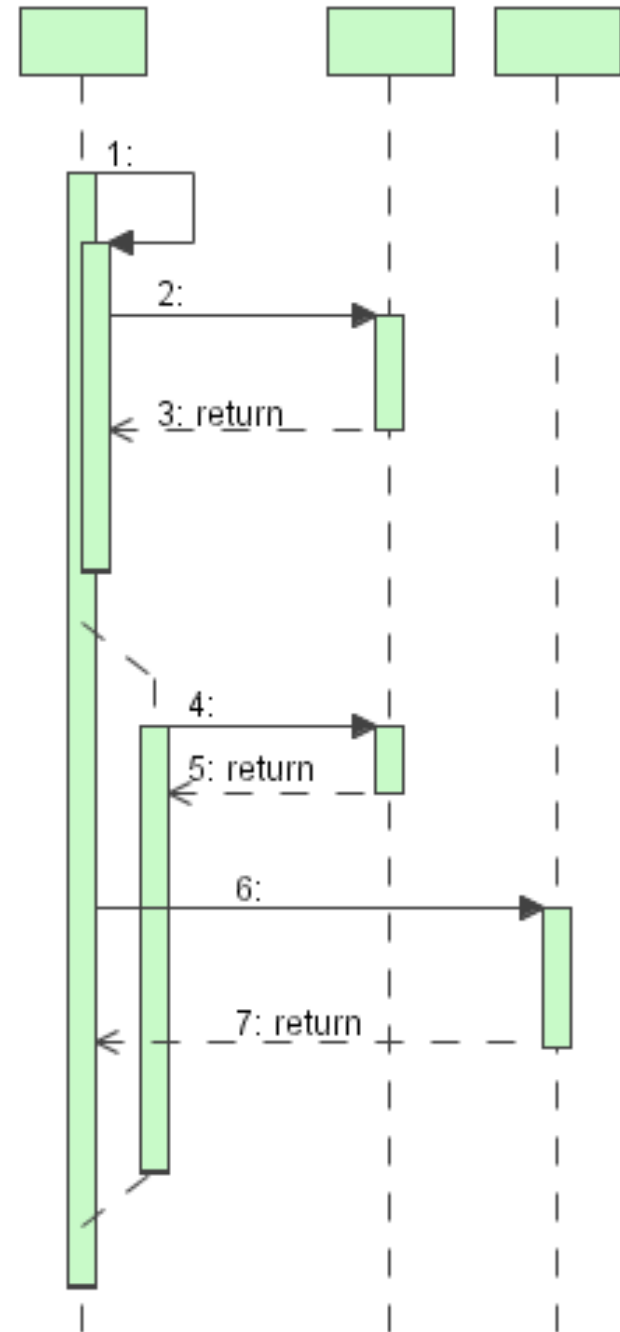


Diagrama de Secuencia

- Reutilización de diagramas de secuencia
- Funcionalidad similar a los métodos en programación
- Se pueden especificar parámetros
- Se puede especificar el valor devuelto o el nombre de una variable

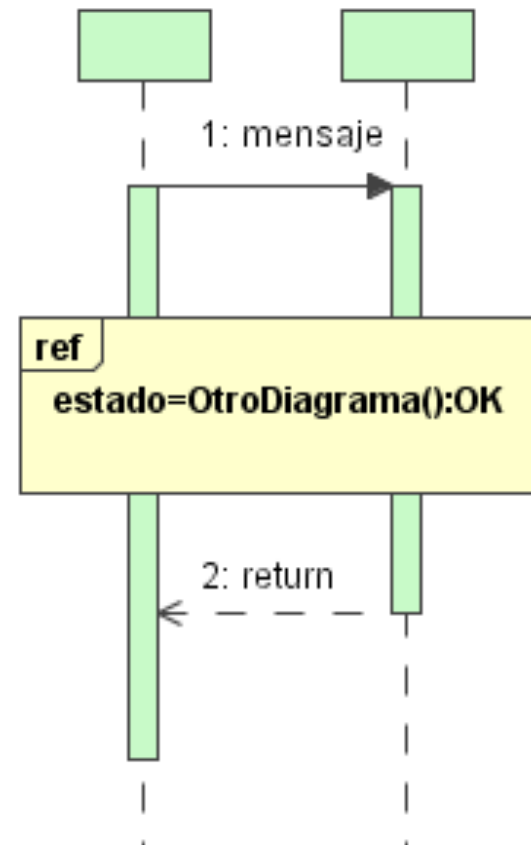
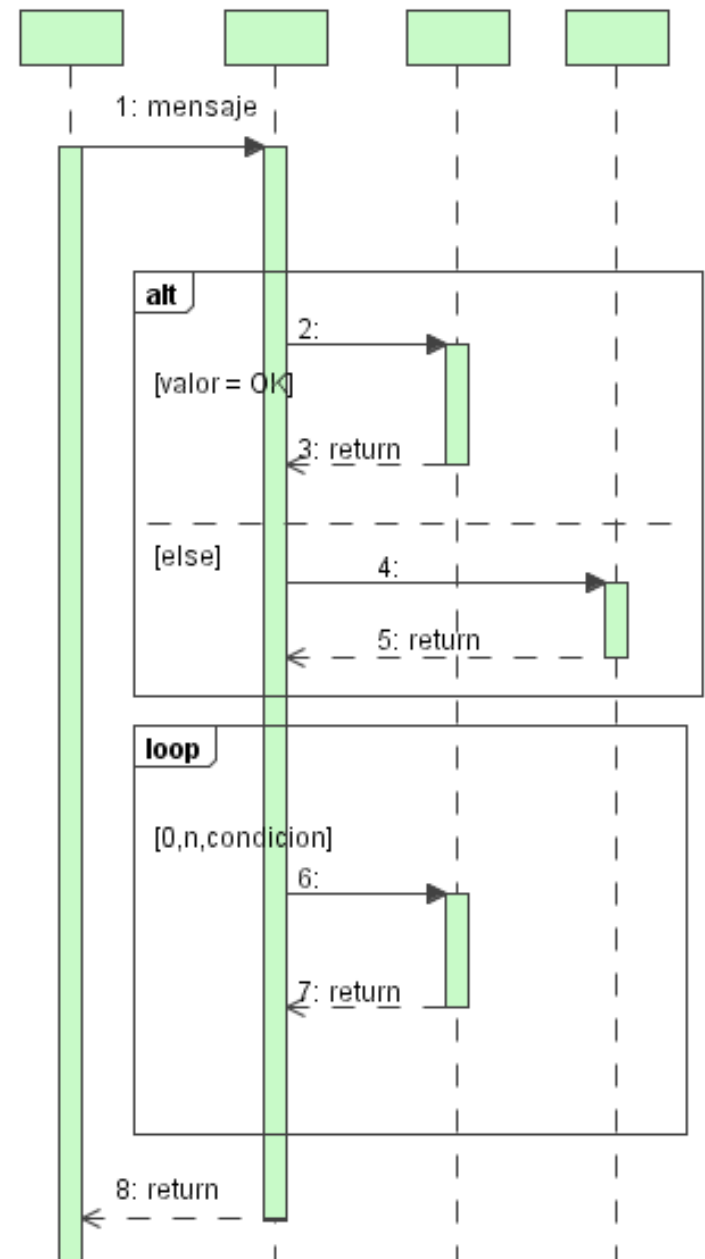


Diagrama de Secuencia

- Sentencias de control de flujo de ejecución en los diagramas de secuencia
 - alt: Equivale a un if/else
 - loop: Equivale a un for
 - Existen más sentencias



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- **Diagrama de Actividad**
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de Actividad

- Similares a los flujogramas
- Se usan para mostrar el flujo de datos o el flujo de control
- Puede representar el flujo de control en diferentes clases

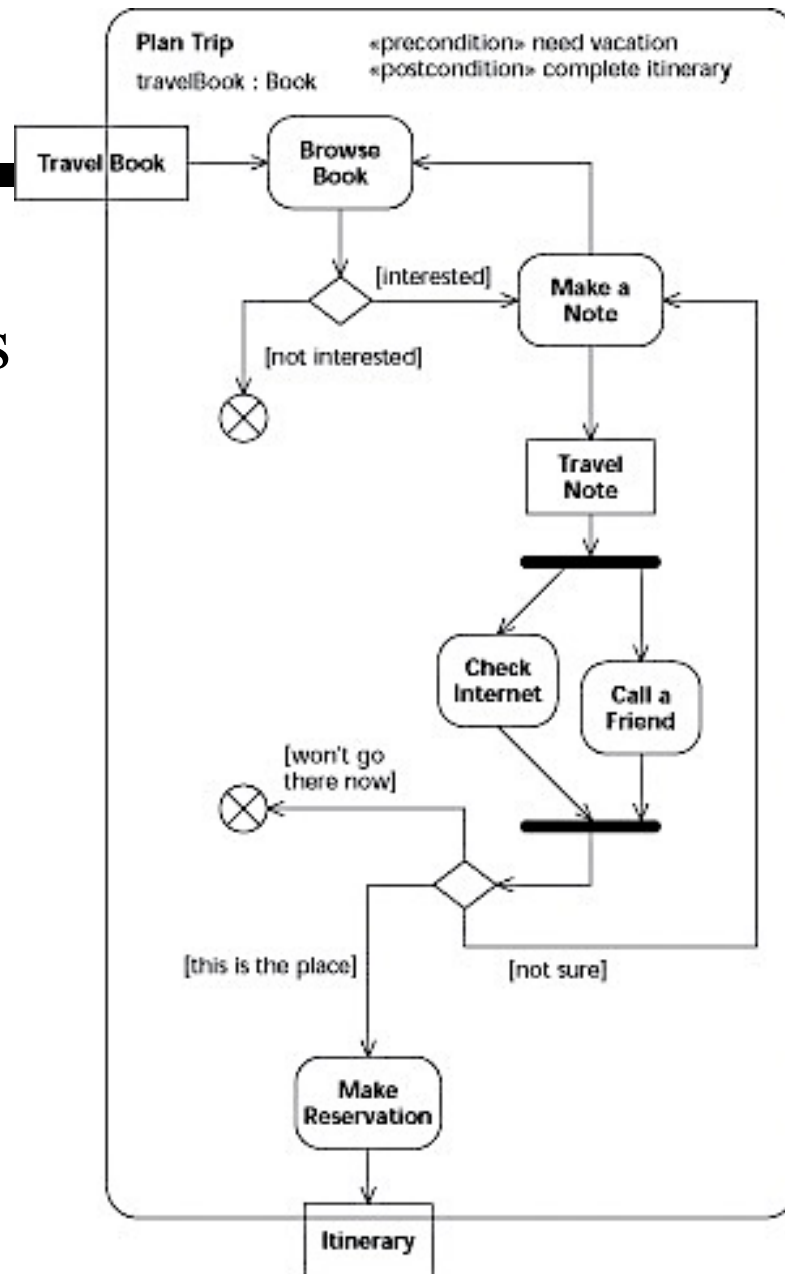


Diagrama de Actividad

- Acción / Estado

- Obtener o establecer un valor de un atributo
- Invocar la operación de una clase
- Llamar a una función
- Invocar una actividad que contiene acciones
- Enviar una señal o notificación de un evento a un grupo de objetos

- Actividad

- Agrupación de acciones, flujos de objetos y de control

Diagrama de Actividad

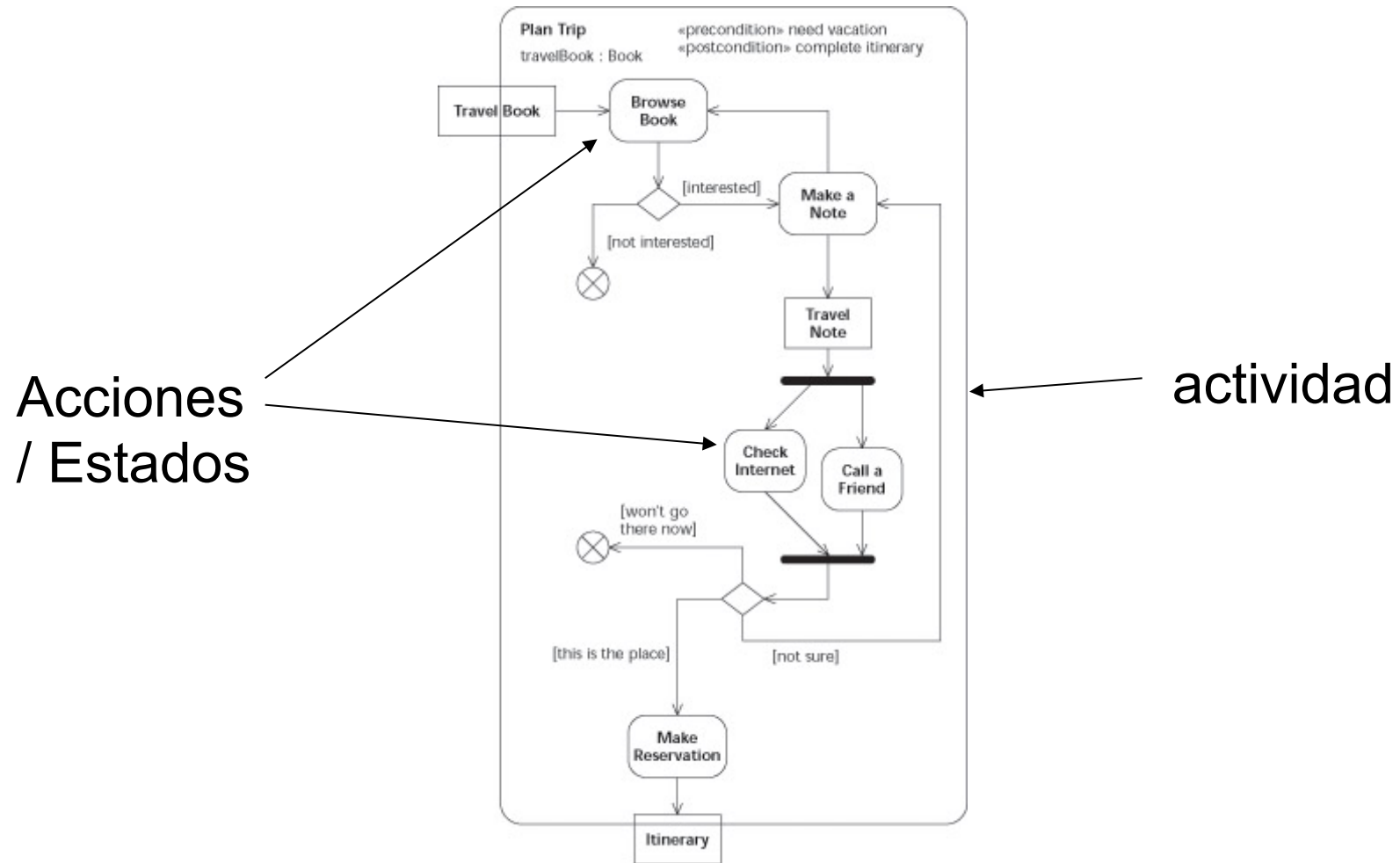


Diagrama de Actividad

- Flujo de Control
 - Conecta las acciones por las que se ejecuta el flujo de control
- Nodo Objeto
 - Representa un objeto que sale de una acción y entra en otra acción
- Flujo de objeto
 - Los flujos que unen los objetos con las actividades

Diagrama de Actividad

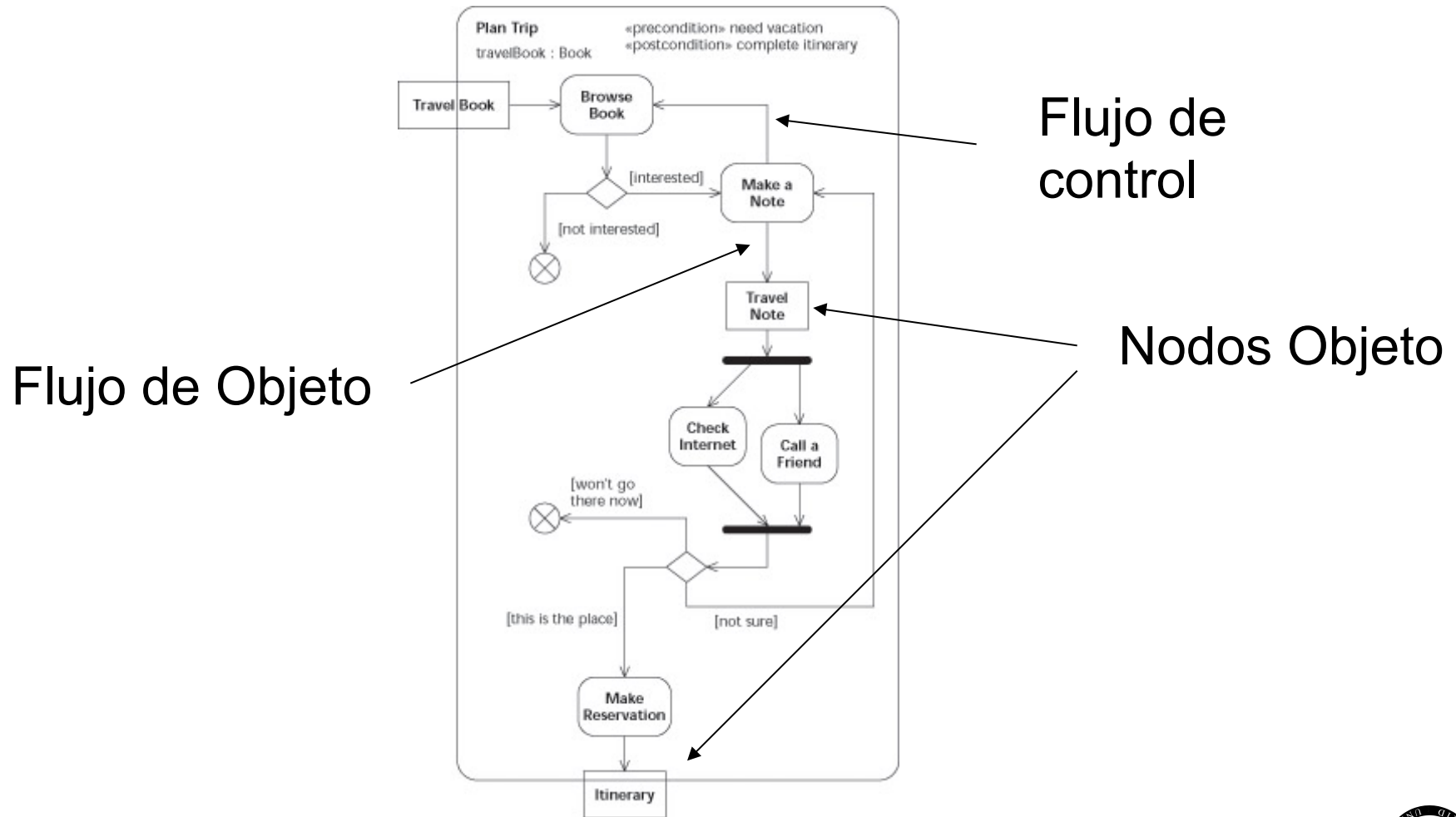


Diagrama de Actividad

- Nodo de control
 - Nodos que guían el flujo de control y objetos
 - Inicial, Actividad final, Flujo final, Decisión, Fusión, Bifurcación, Unión, Conector
- Existen tipos de nodos más avanzados

Diagrama de Actividad

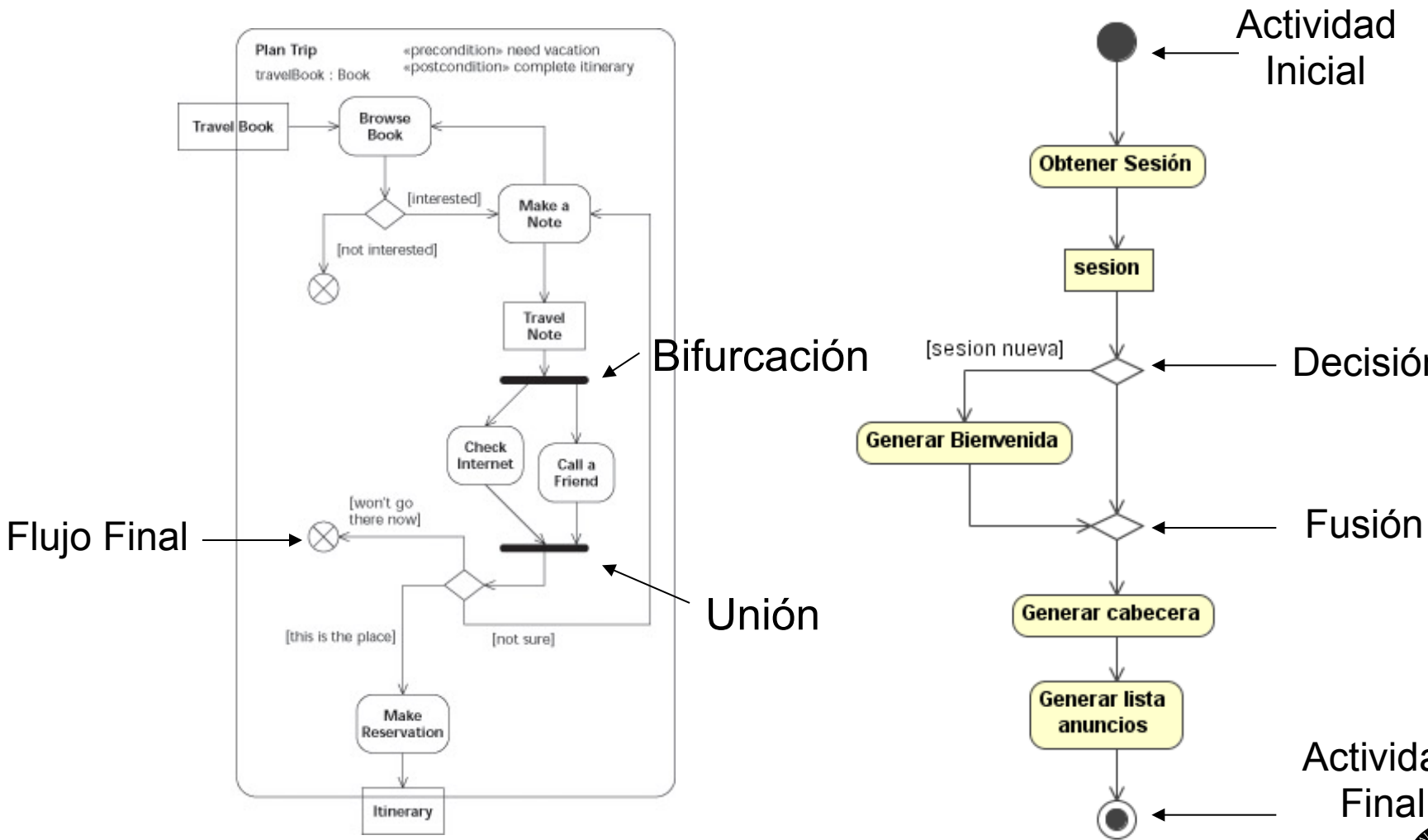


Diagrama de Actividad

- Para asignar la responsabilidad
 - Se pueden poner calles de piscinas (swim lanes)
 - Se puede poner el nombre entre paréntesis

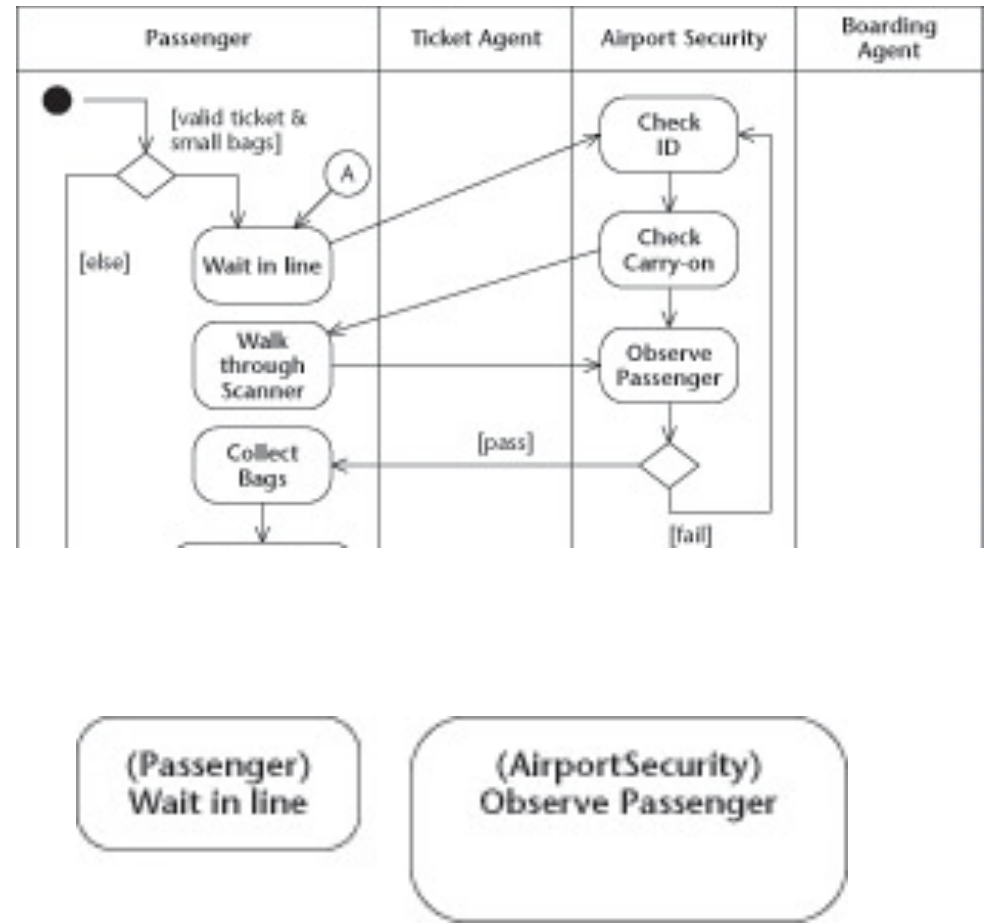
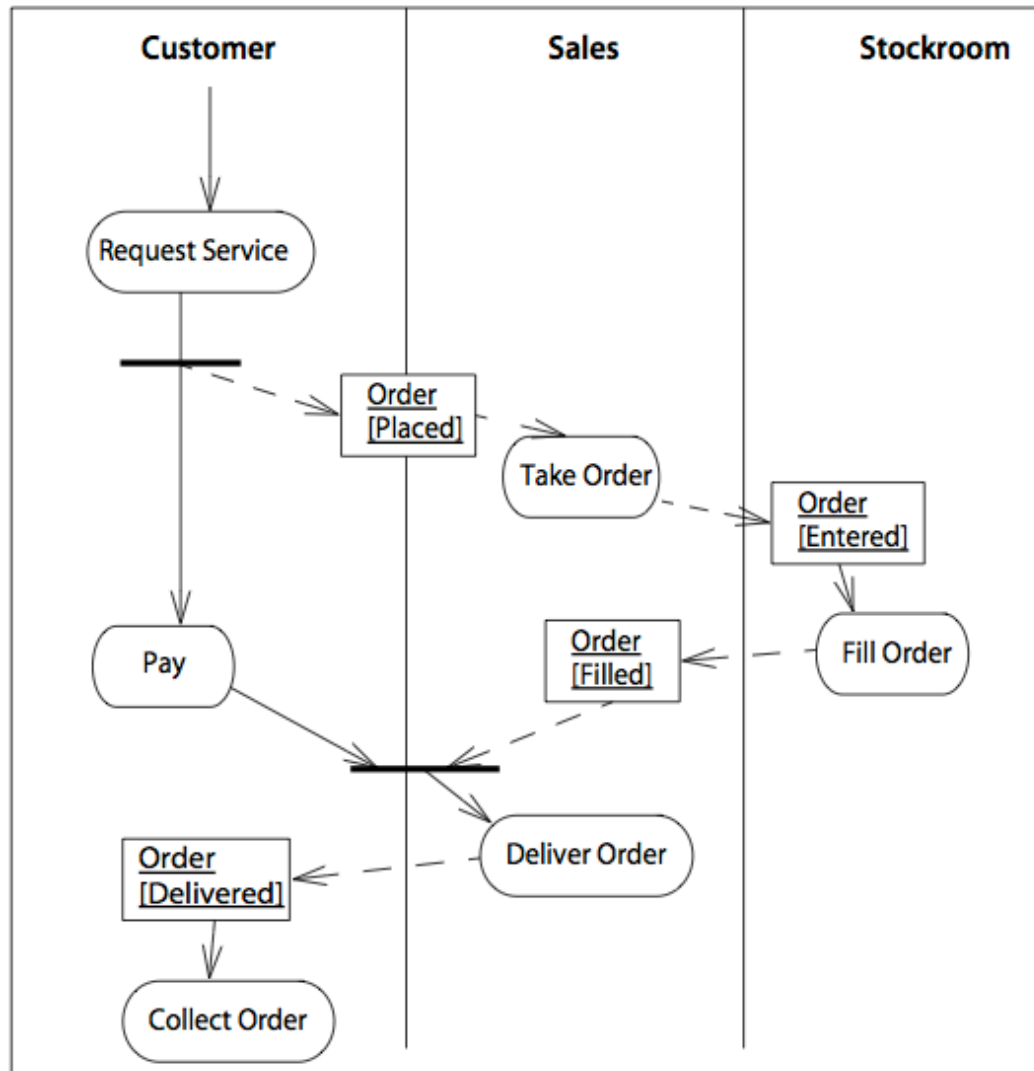


Diagrama de Actividad

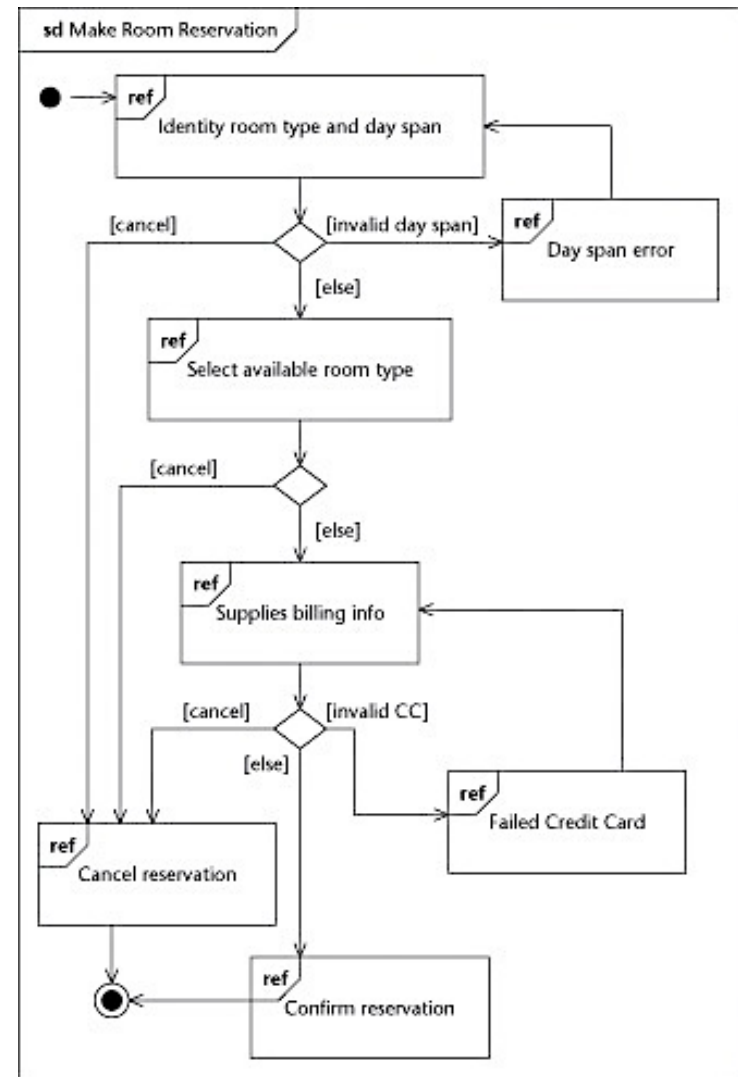


El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- **Diagrama de Interacción en Visión General**
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de Interacción en Visión General

- Similar al diagrama de actividad pero en vez de actividades se enlazan con los flujos de control **nodos de ocurrencia de interacción**
- Son referencias a diagramas de secuencia



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- **Diagrama de Comunicación**
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de Comunicación

- Antes llamados “Diagramas de Colaboración”
- Similares a los diagramas de secuencia pero muestran los enlaces entre los objetos
- Para mostrar el orden y anidamiento de los mensajes, se numeran

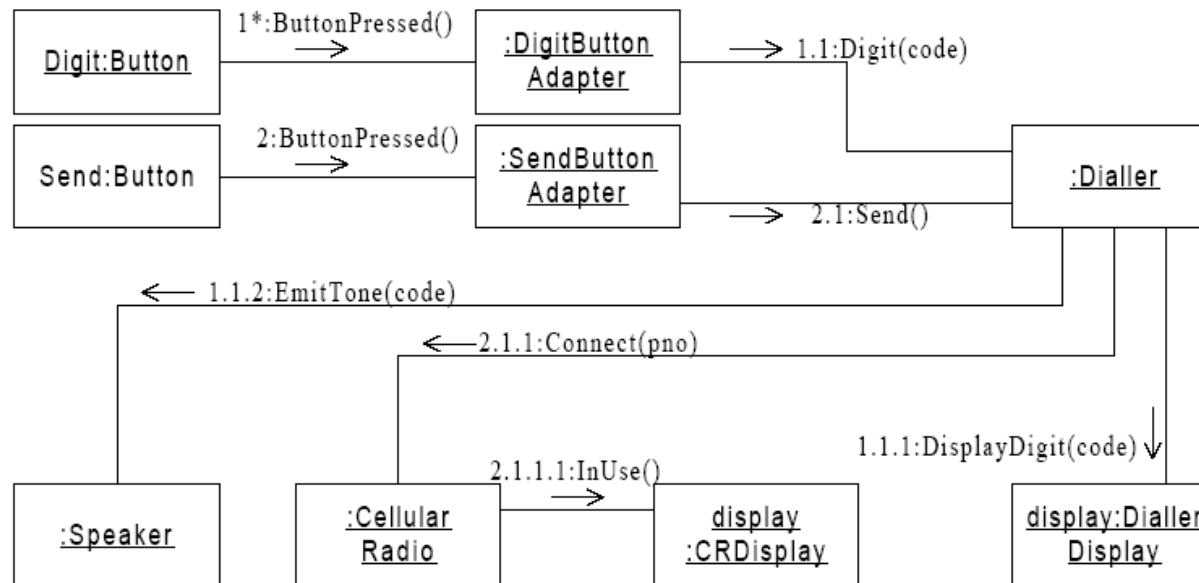


Diagrama de Comunicación

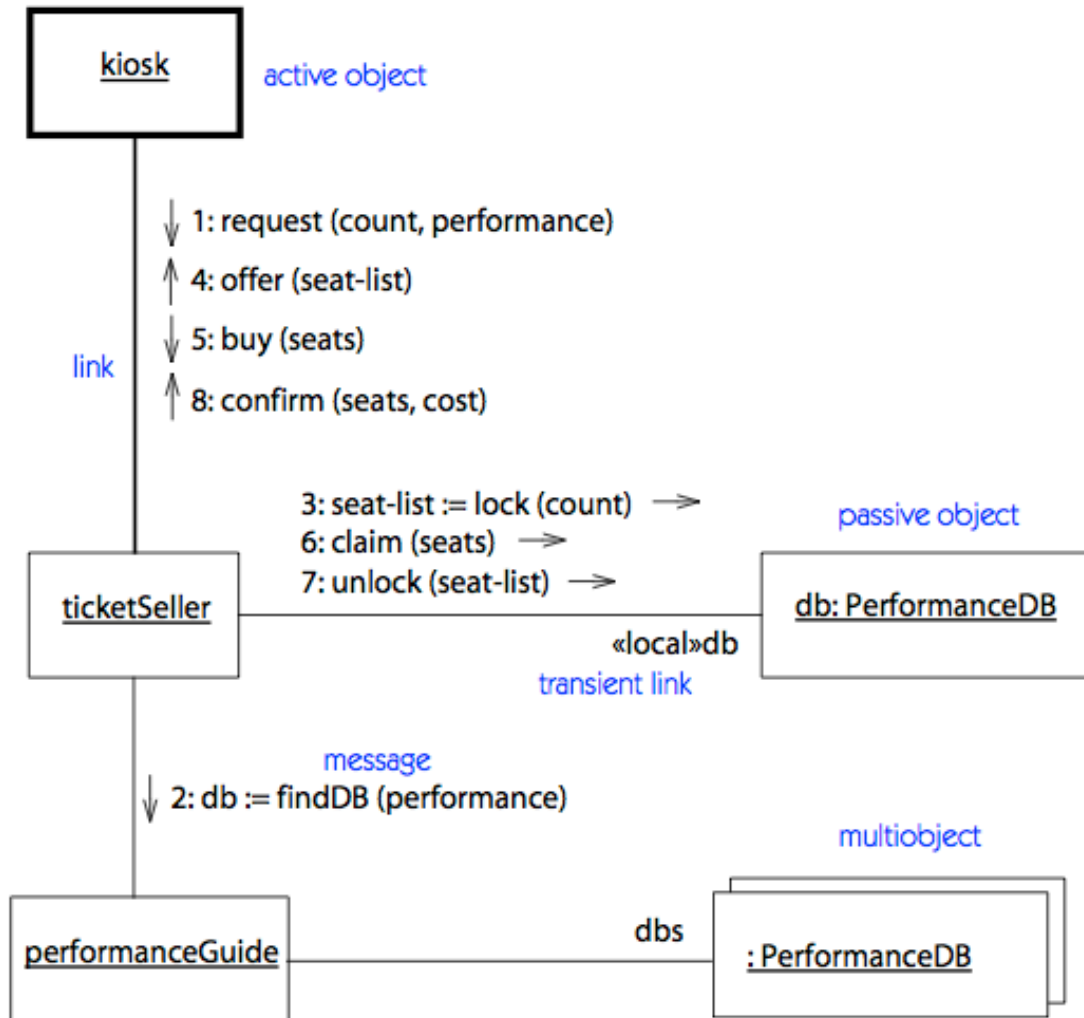
- En el mensaje se puede incluir una condición, que debe cumplirse para que se pueda enviar el mensaje

[condición] mensaje(parametros)



- Cuando dos objetos están enlazados se pueden enviar mensajes. Pueden ser atributos, variables o parámetros

Diagrama de Comunicación



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- **Diagrama de Estados**
- Diagrama de Componentes
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de Estados

- Representan una máquina de estados
- Muestra los estados y las transiciones entre los estados dependiendo de los eventos que se producen

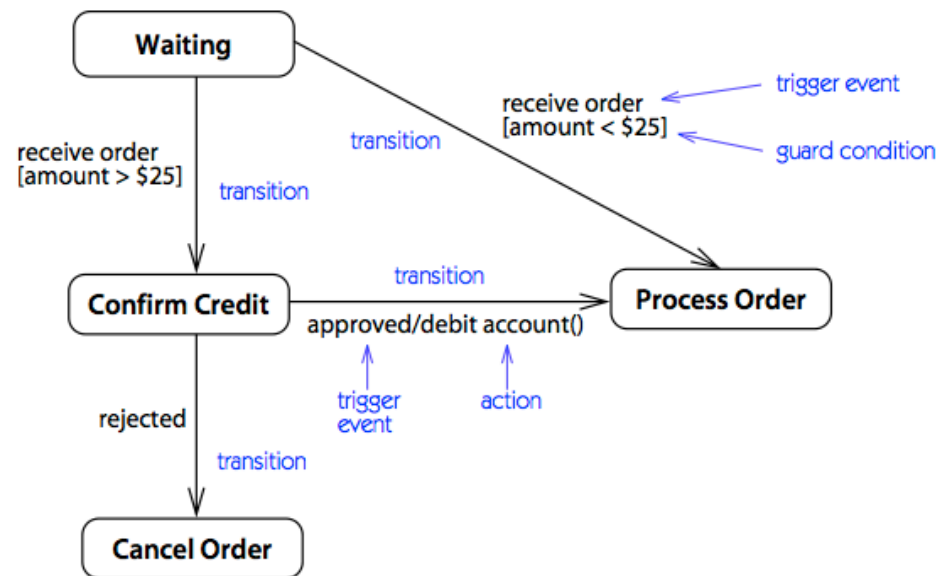
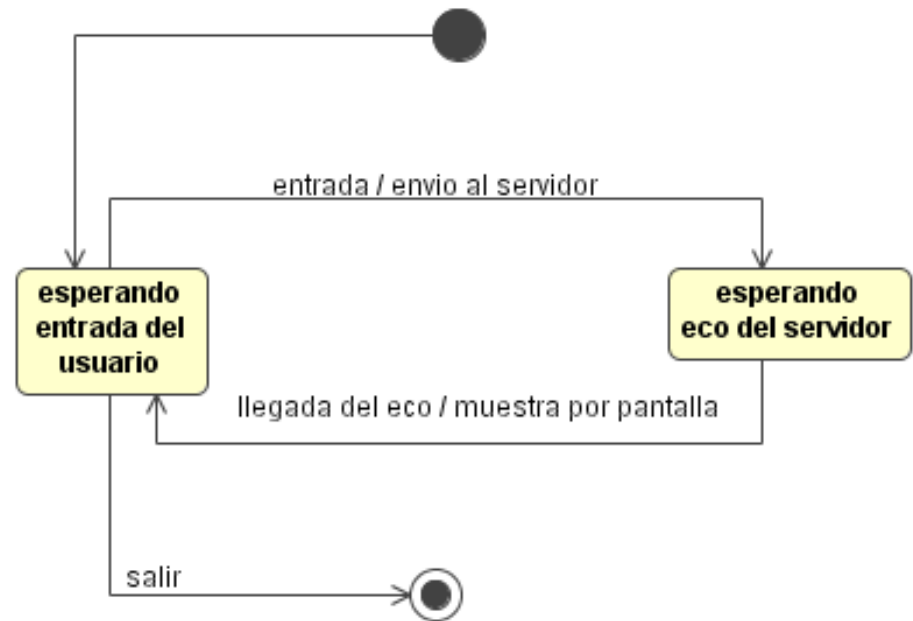


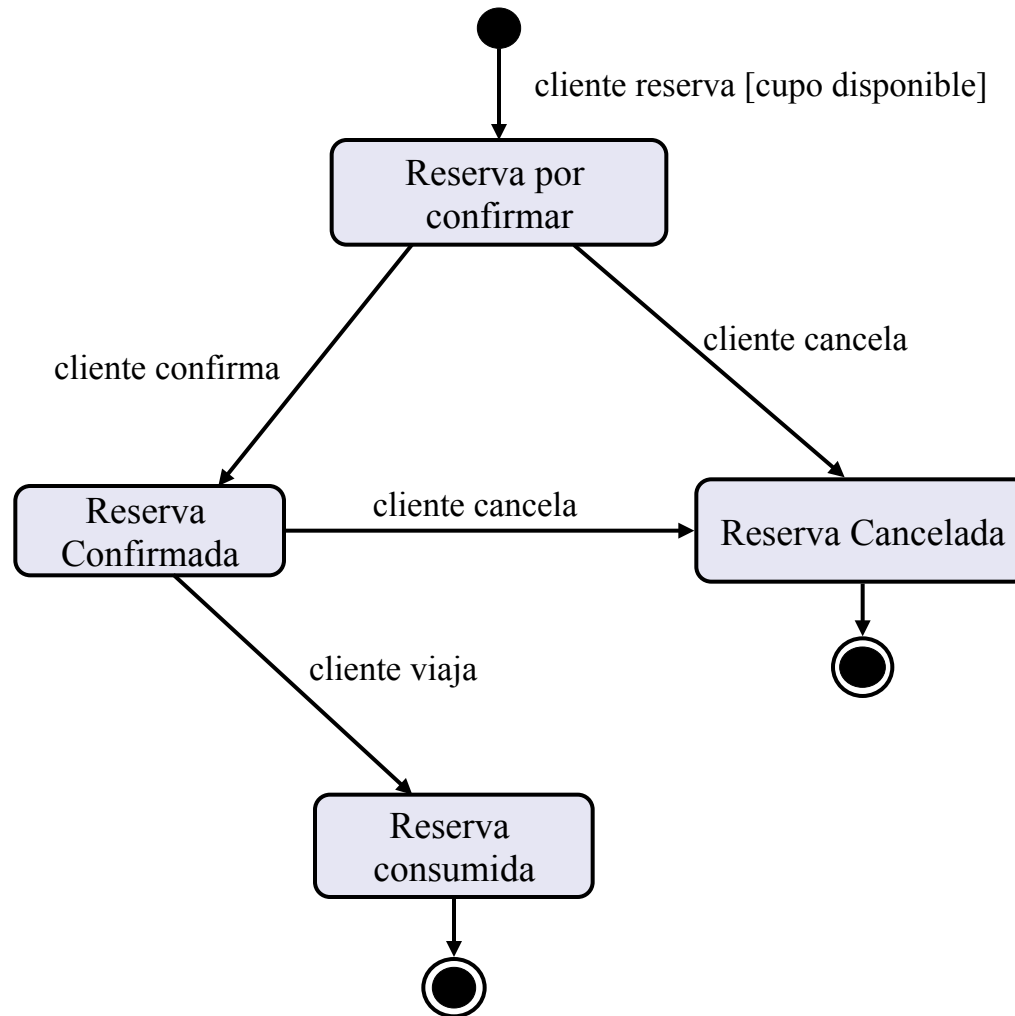
Diagrama de Estados

- Cuando llega un evento, se puede realizar una acción durante la transición a otro estado

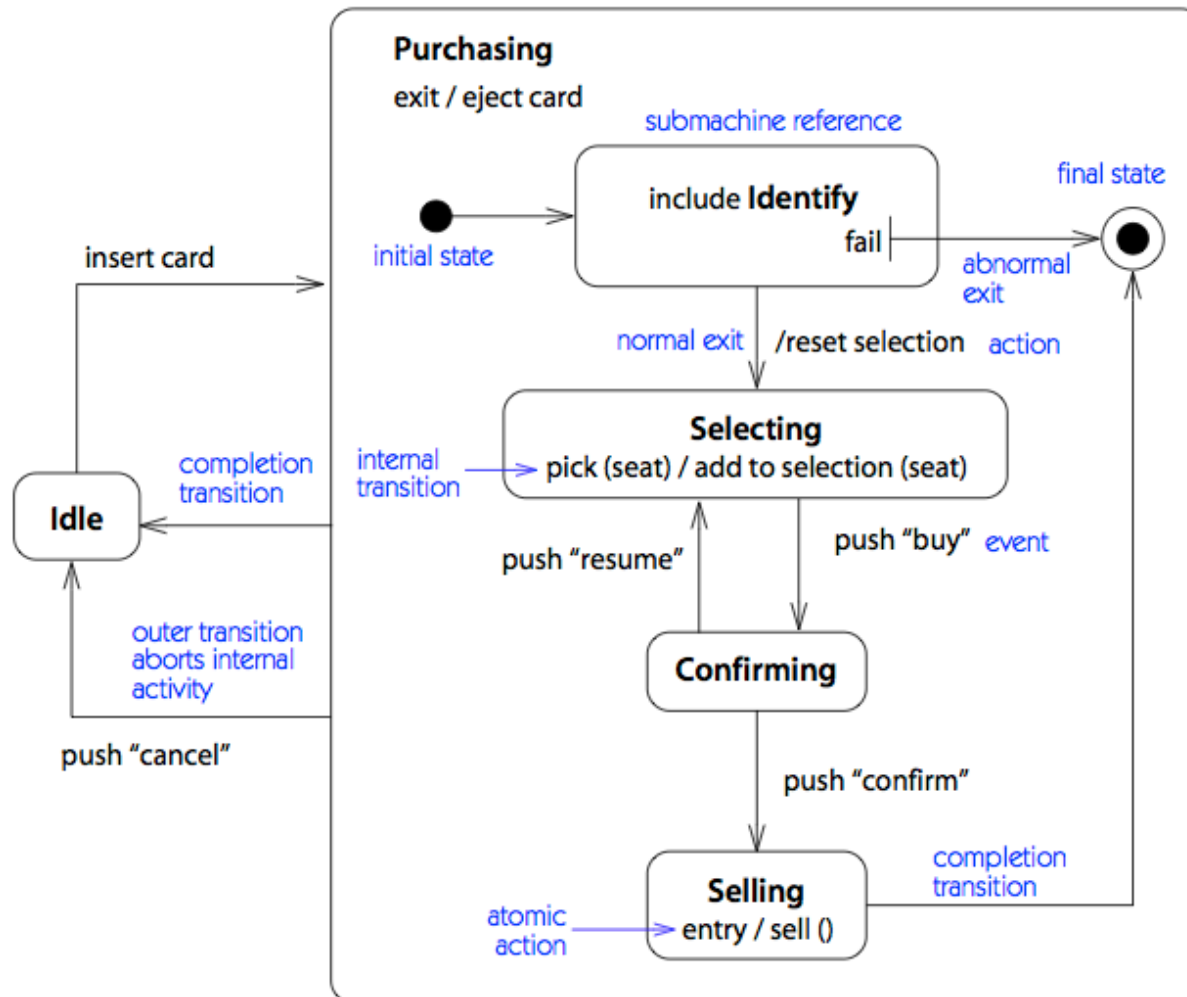


Diagramas de Estado

Ejemplo



Diagramas de Estado



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- **Diagrama de Componentes**
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Diagrama de Componentes

- Un componente es una unidad autónoma reemplazable de un sistema
- Los componentes indican los interfaces públicos para que otros componentes los usen (relación de realización)
- Los componentes pueden indicar los interfaces requeridos en otros componentes (relación de uso)

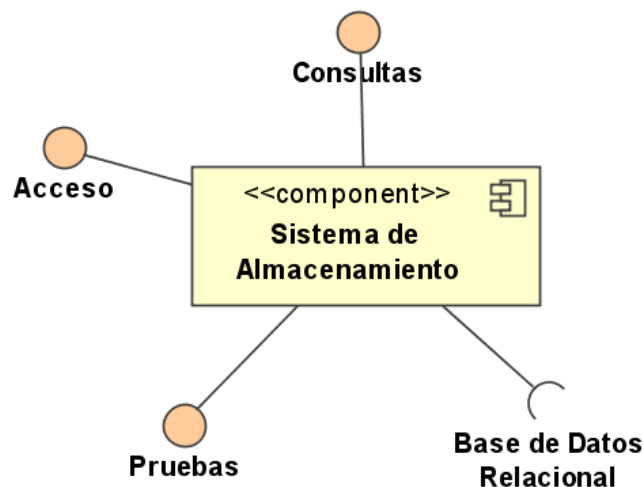


Diagrama de Componentes

- En un mismo diagrama pueden aparecer varios componentes conectados mediante interfaces

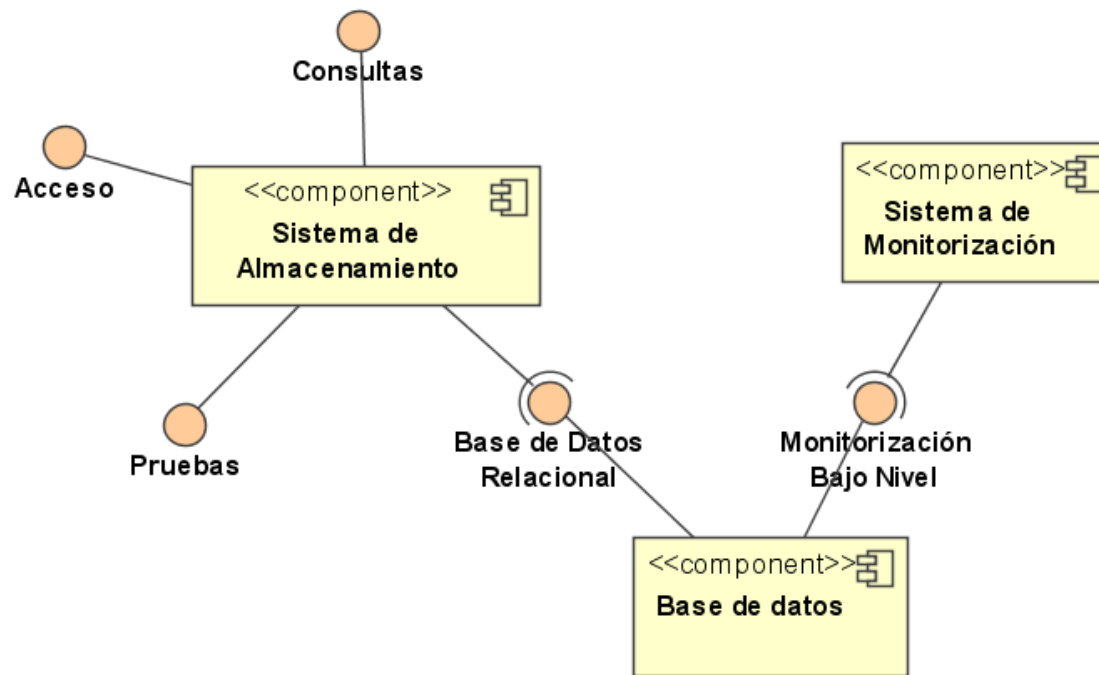
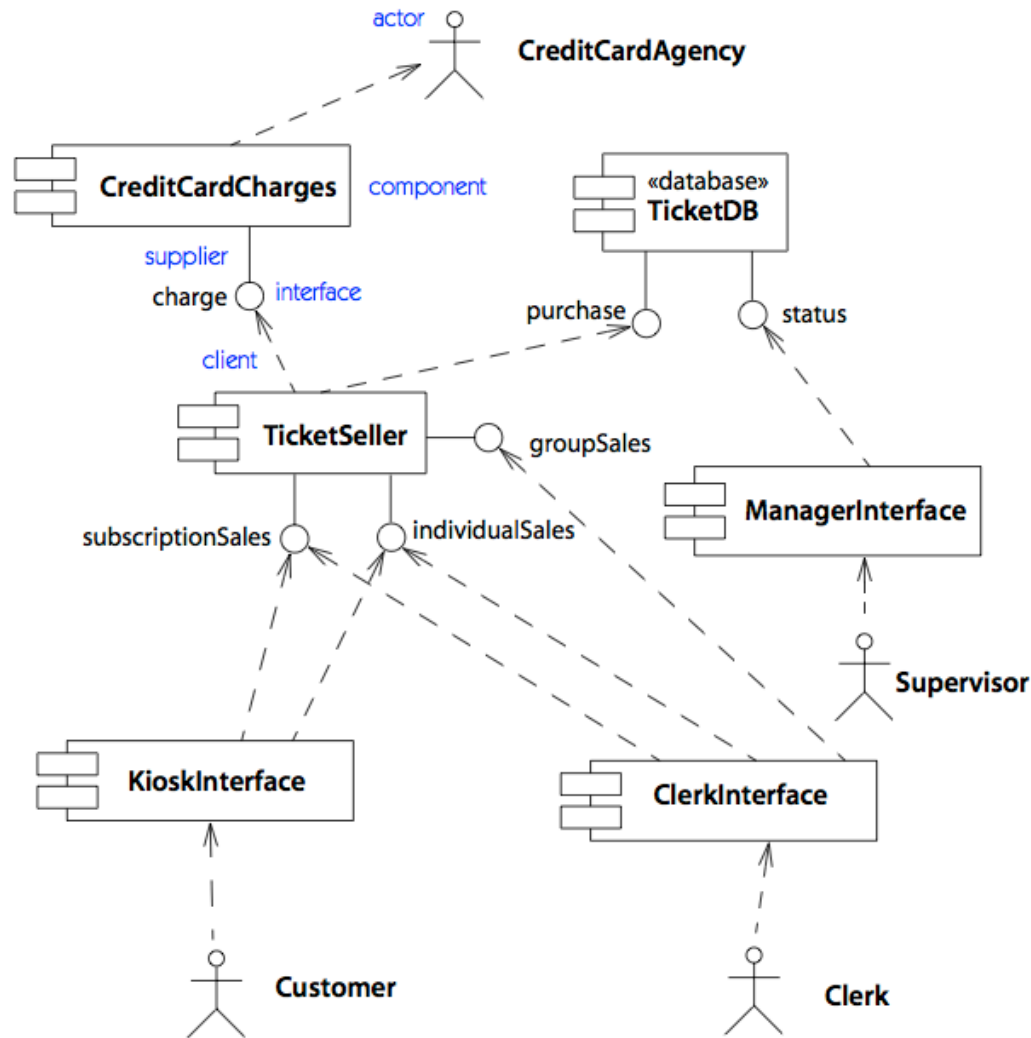


Diagrama de Componentes



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- **Diagrama de Despliegue**
- Diagrama de Casos de Uso

Diagrama de Despliegue

- Muestra las partes físicas del sistema
 - PCs, Servidores
 - Impresoras, scanners
 - PDAs, móviles
- Conectadas por Líneas de comunicaciones
 - Internet, LAN, USB, Bluetooth

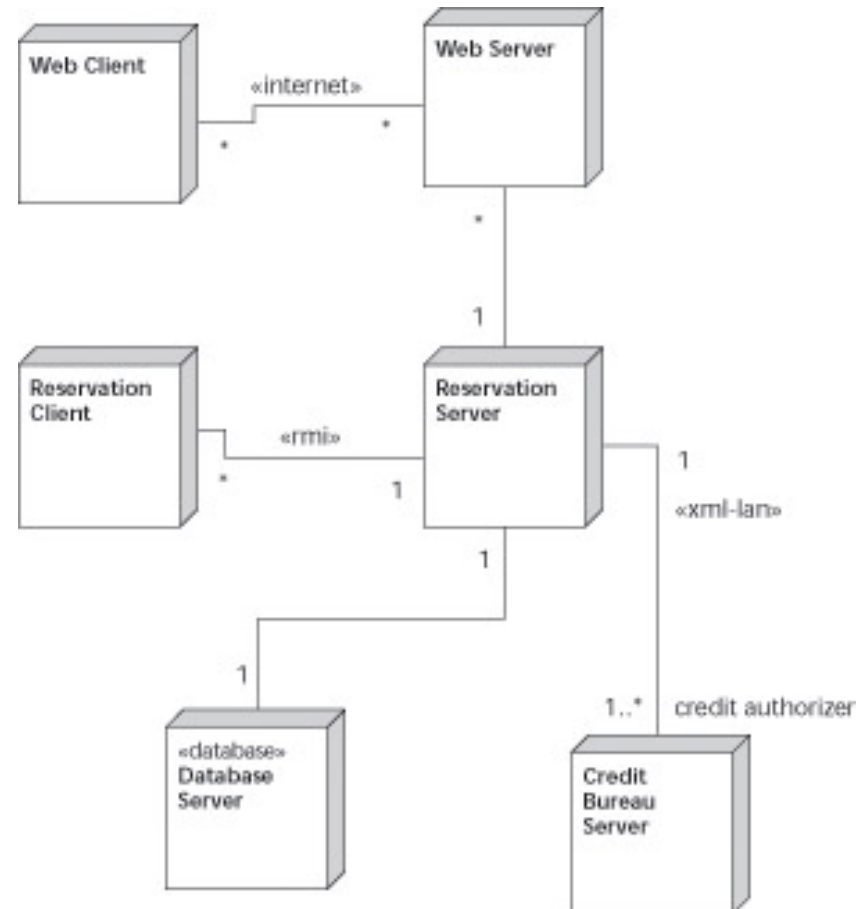


Diagrama de Despliegue

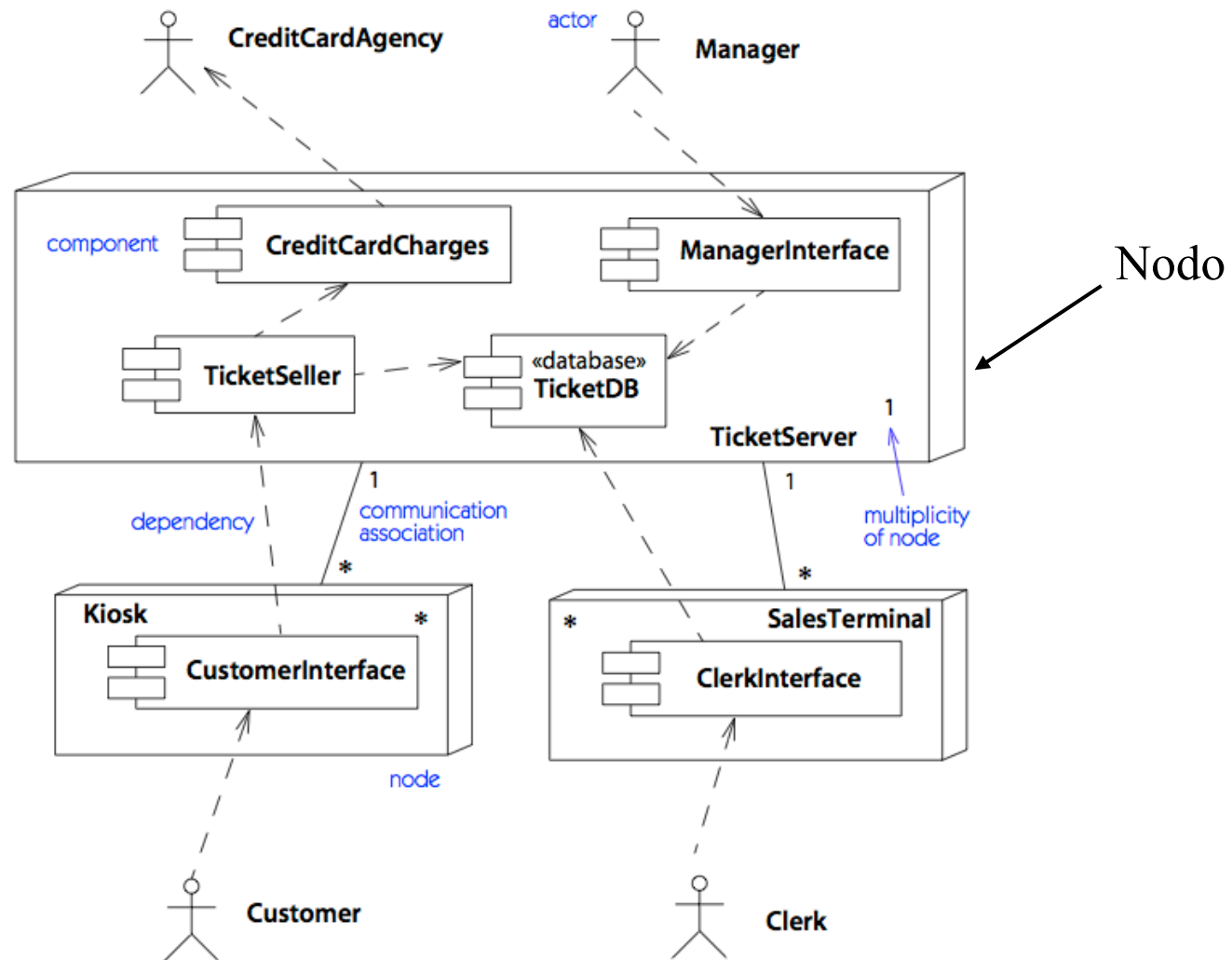


Diagrama de Despliegue

- Los nodos del diagrama (hardware) son clases con una representación especial
 - Pueden tener propiedades (con valores)
 - Pueden tener operaciones
 - ☹ Es posible que la herramienta no lo soporte
- Las líneas de comunicación son asociaciones
 - Pueden tener nombre, roles, multiplicidad, estereotipos, restricciones...

Diagrama de Despliegue

Artefactos

- Un artefacto representa un fichero físico en el sistema
- En versiones anteriores de UML se les llamaba componentes
- Se pueden poner las dependencias entre artefactos

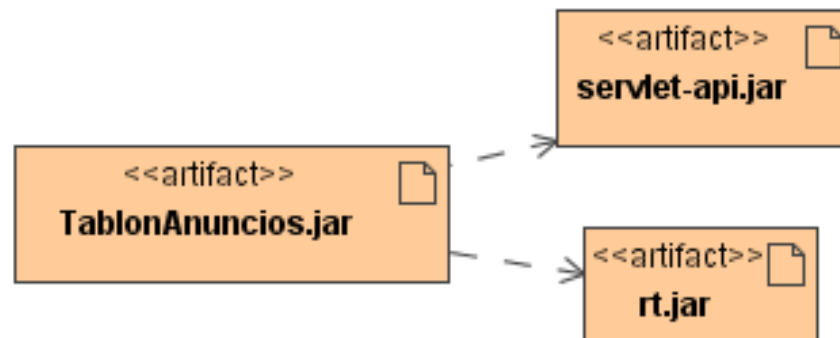
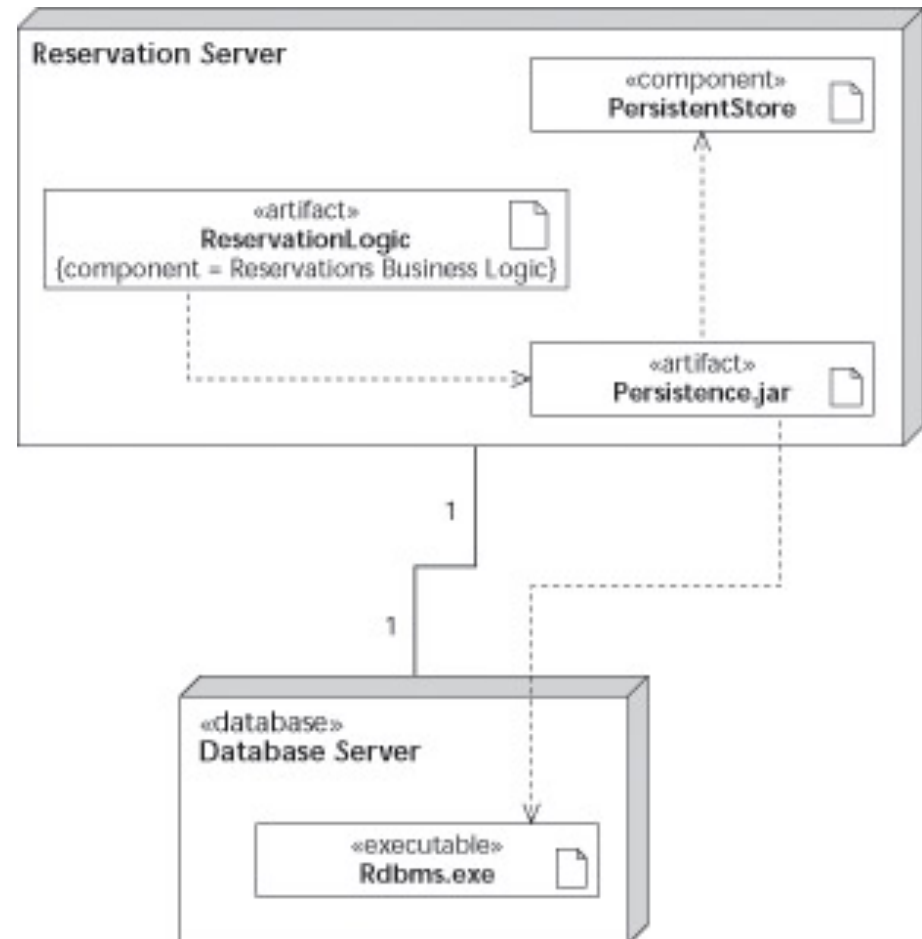


Diagrama de Despliegue Artefactos

- Se pueden mostrar los artefactos, los nodos y los componentes y las relaciones entre ellos
- Los componentes y artefactos deben estar en nodos



El Lenguaje Unificado de Modelado

- ¿Qué es UML?
- Diagrama de Clases
- Notas
- Diagrama de Paquetes
- Diagrama de Objetos
- Diagrama Estructural Compuesto
- Diagrama de Secuencia
- Diagrama de Actividad
- Diagrama de Interacción en Visión General
- Diagrama de Comunicación
- Diagrama de Estados
- Diagrama de Componentes
- Diagrama de Despliegue
- **Diagrama de Casos de Uso**

Diagrama de Casos de Uso

- Documentan el comportamiento de un sistema desde el punto de vista del usuario
- Representan las funciones que un sistema puede ejecutar
- Fácil interpretación

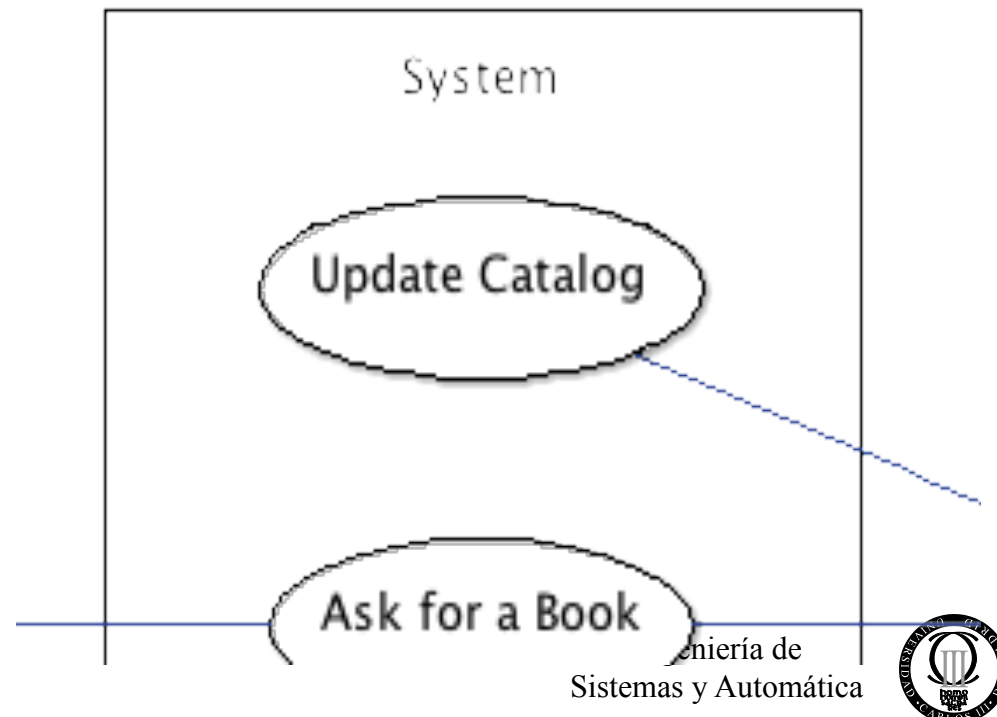


Diagrama de Casos de Uso

Elementos Básicos: Actores

- Usuarios del sistema: entidad externa que interactúa con el sistema
 - Humano, sistema informático, empresa...
- Independizar los actores de la forma en que se interactúa con el sistema
 - Un teclado no es un actor
- Un actor representa un rol que alguien puede estar jugando, no un individuo particular



Diagrama de Casos de Uso

Elementos Básicos: Casos de Uso

- Es una tarea que debe poder llevarse a cabo con el apoyo del sistema que se está desarrollando.
- Se representan mediante un óvalo
- Cada caso de uso debe detallarse, habitualmente mediante una descripción textual

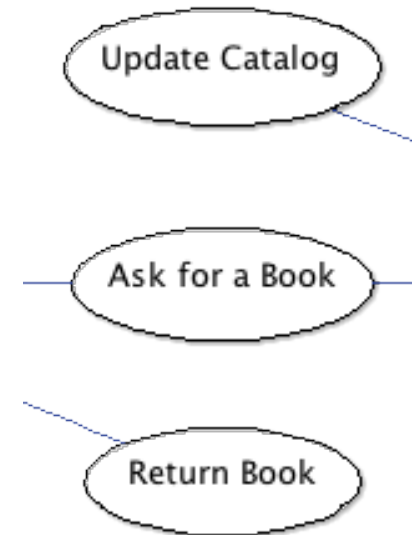


Diagrama de Casos de Uso

Elementos Básicos: Asociaciones

- Hay una asociación entre un actor y un caso de uso si el actor interactúa con el sistema para llevar a cabo el caso de uso
- Tipos: Include, Extend, Generalización

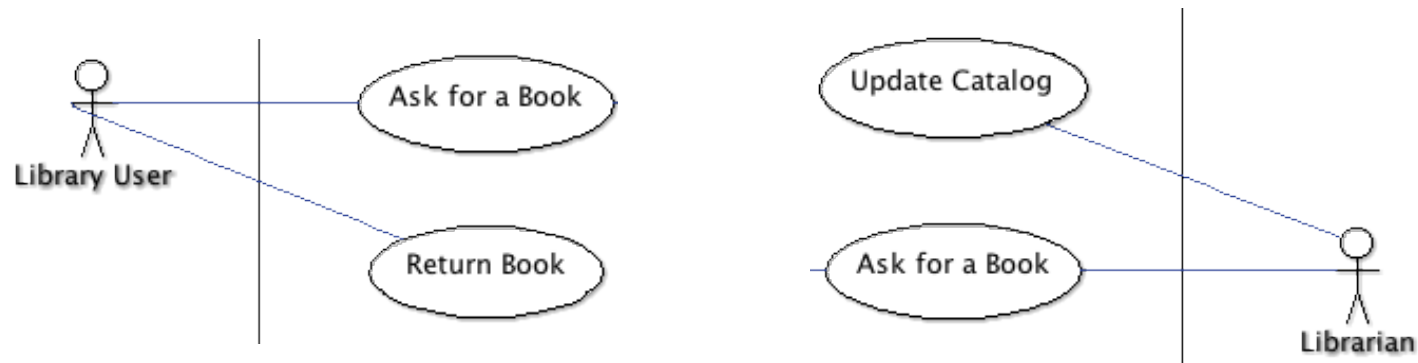
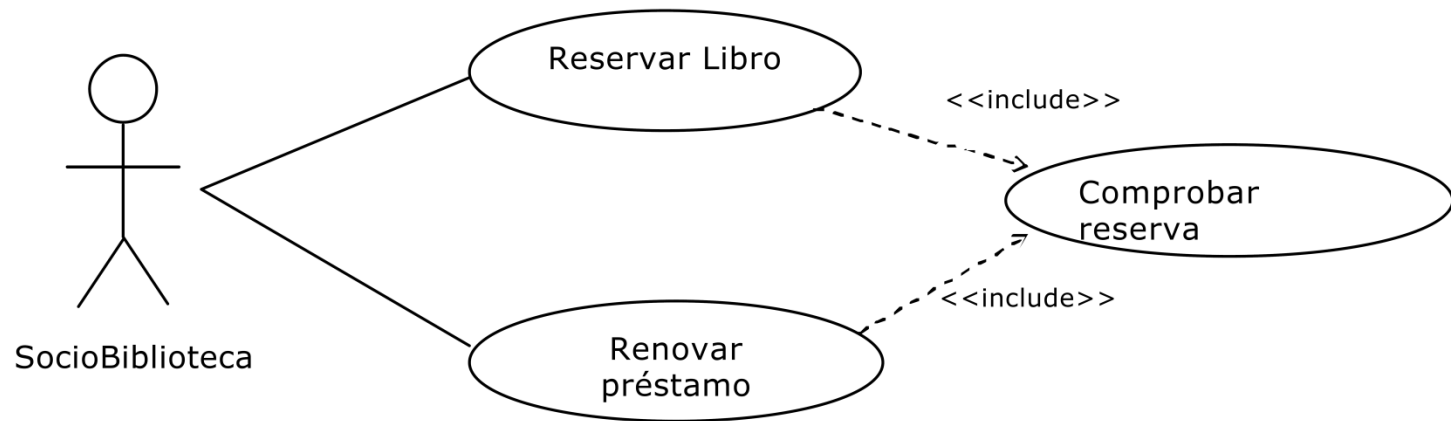


Diagrama de Casos de Uso

Elementos Básicos: Asociaciones

- Include: Se puede incluir una relación entre dos casos de uso de tipo “include” si se desea especificar comportamiento común en dos o más casos de uso



tanto “Reservar Libro” como “Renovar préstamo” hacen algo en común “Comprobar reserva”.

Diagrama de Casos de Uso

Elementos Básicos: Asociaciones

- Extend: Se puede incluir una relación entre dos casos de uso de tipo “extend” si se desea especificar diferentes variantes del mismo caso de uso
- Esta relación implica que el comportamiento de un caso de uso es diferente dependiendo de ciertas circunstancias
- La flecha en el caso de las relaciones “extend” va hacia el caso de uso “original”

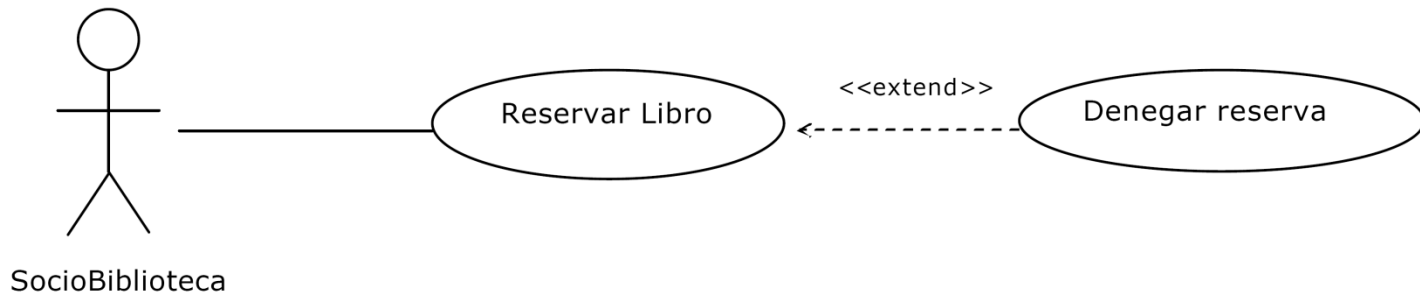
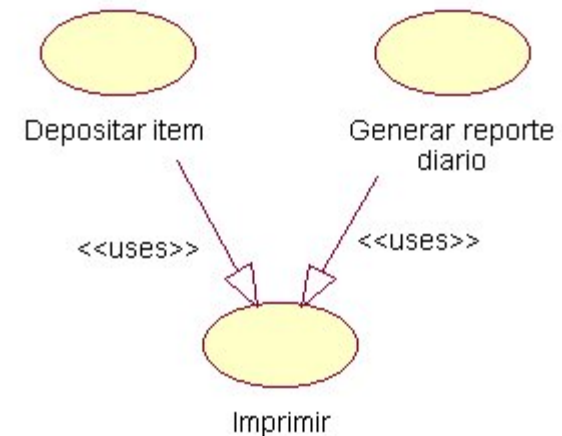
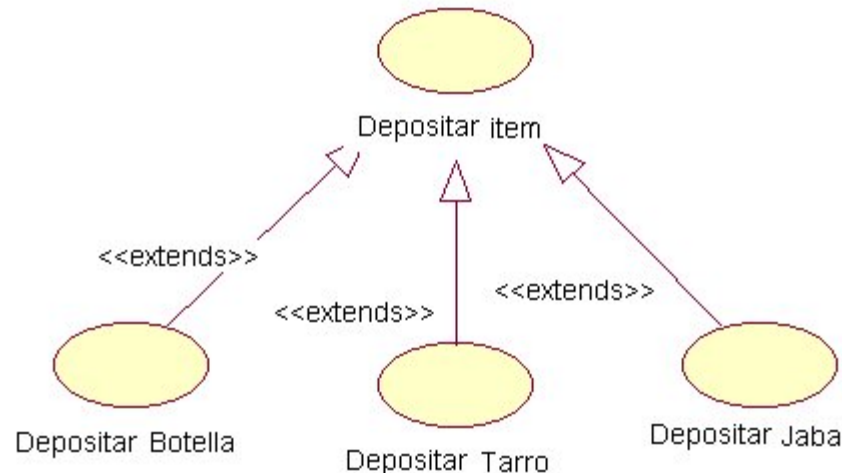


Diagrama de Casos de Uso

Elementos Básicos: Asociaciones

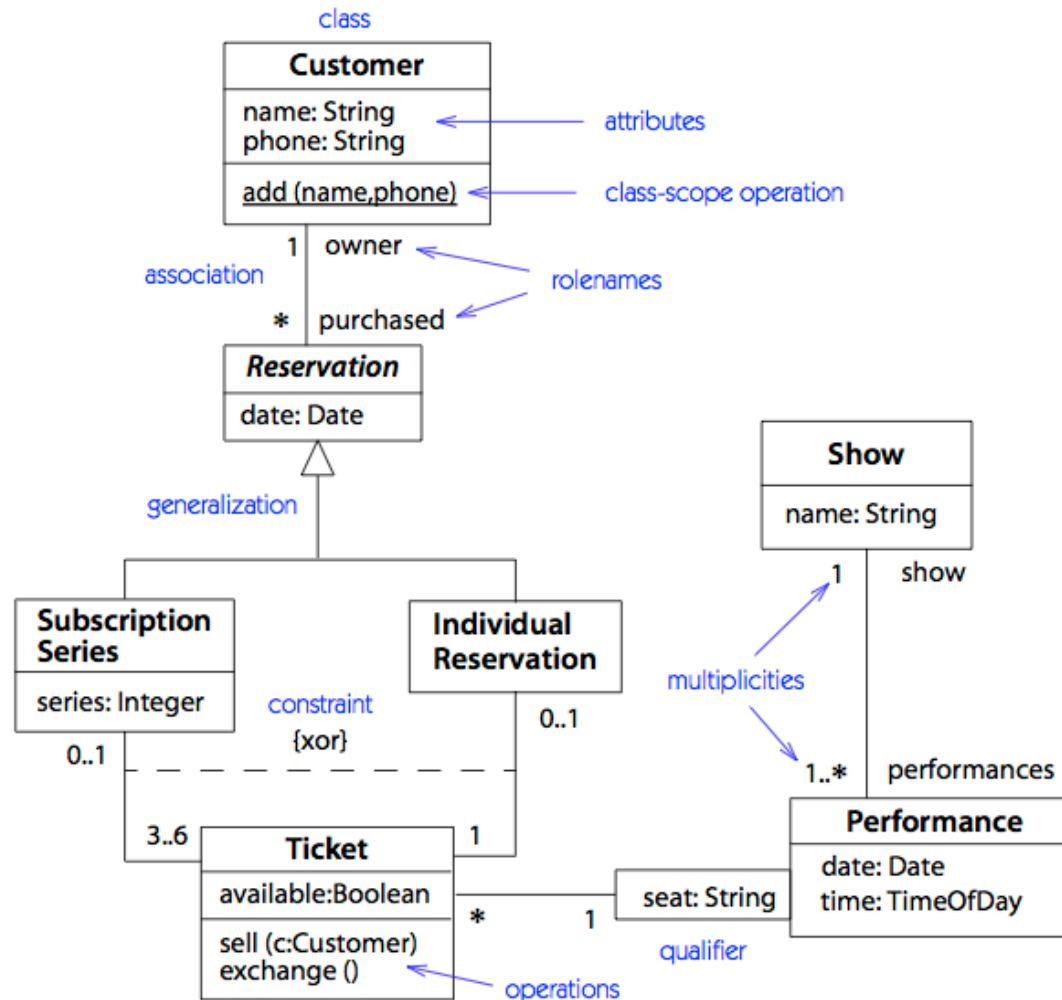
- Generalización/Especialización: En un diagrama de casos de uso también pueden mostrarse generalizaciones (relaciones de herencia) para mostrar que diferentes elementos están relacionados como tipos de otros
- Son aplicables a actores o casos de uso, pero para estos últimos la semántica es muy similar a las relaciones “extend”



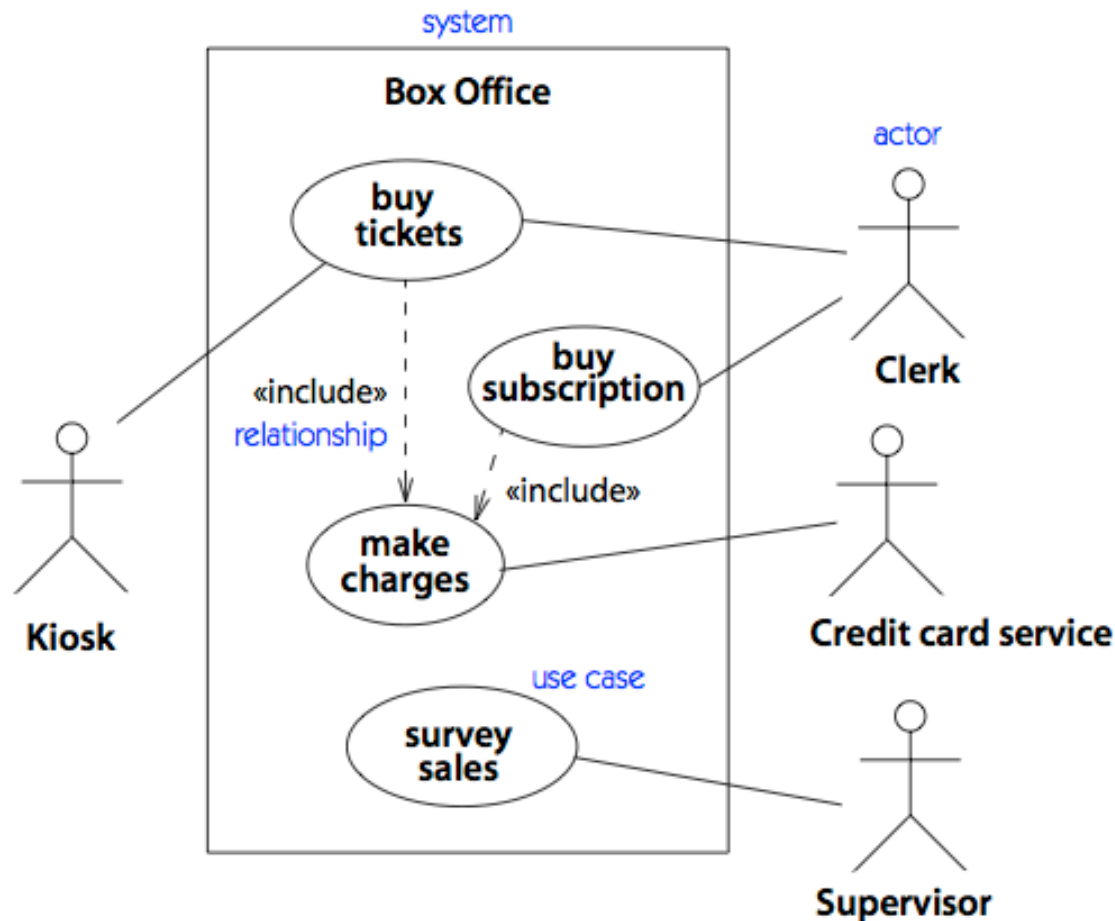
Resumen

- A continuación se muestran algunos ejemplos de diagramas referidos a un sistema de venta de entradas

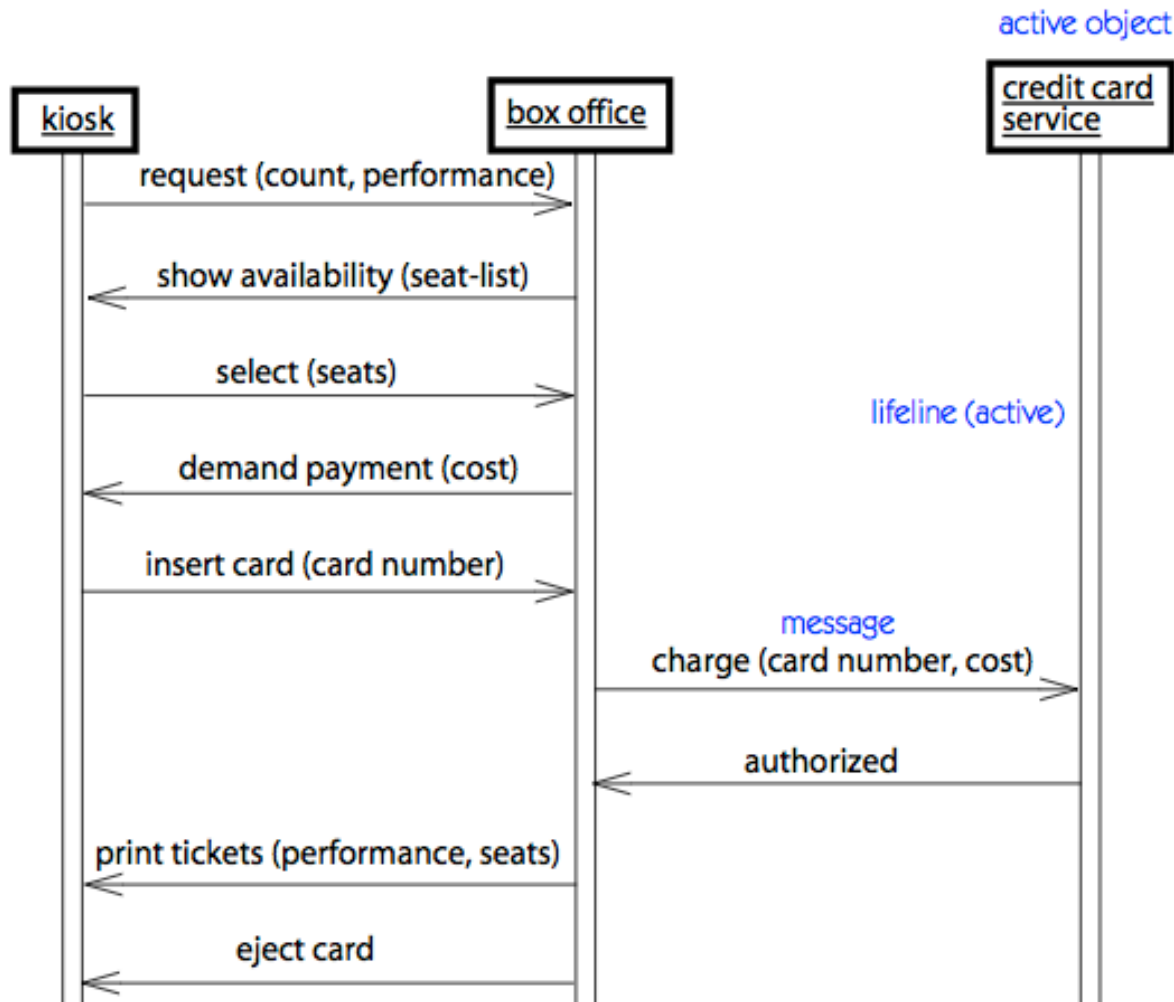
Ejemplo: Diagrama de clases



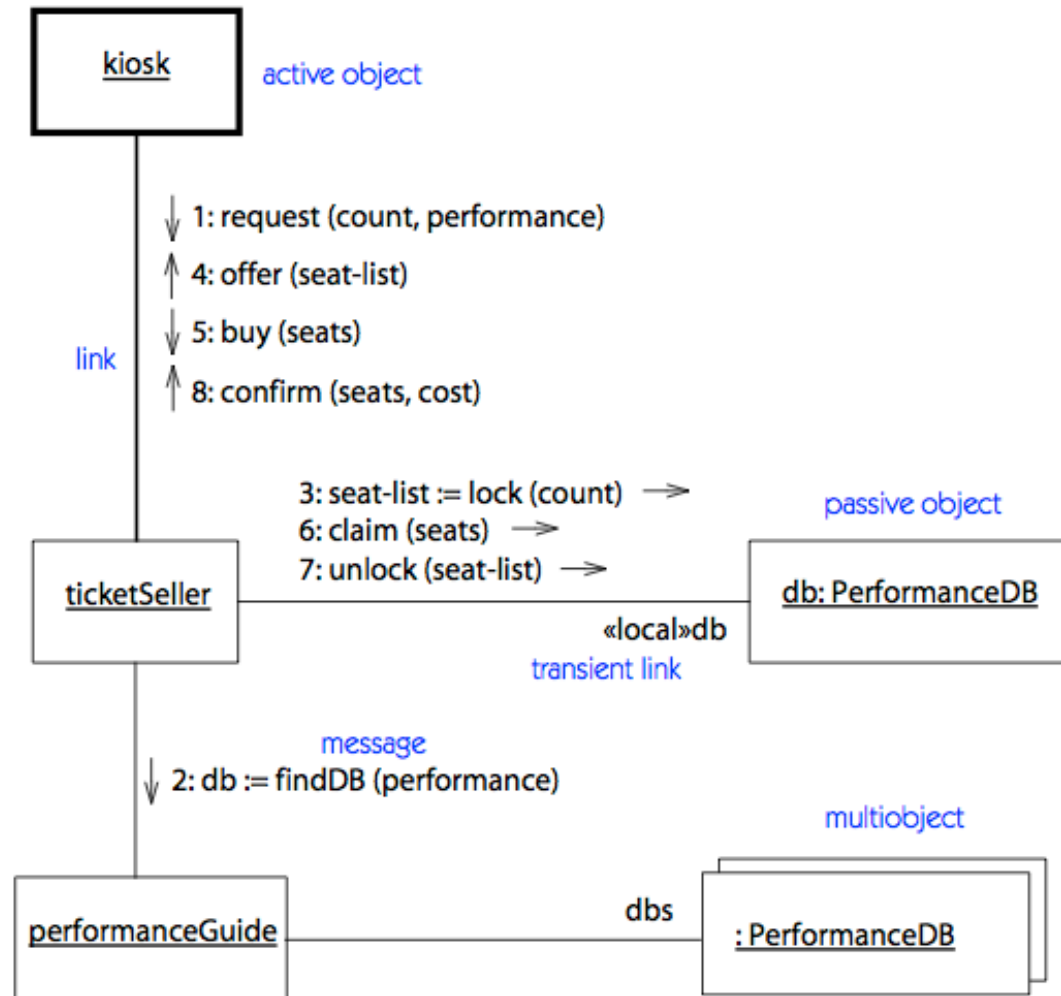
Ejemplo: Diagrama de casos de uso



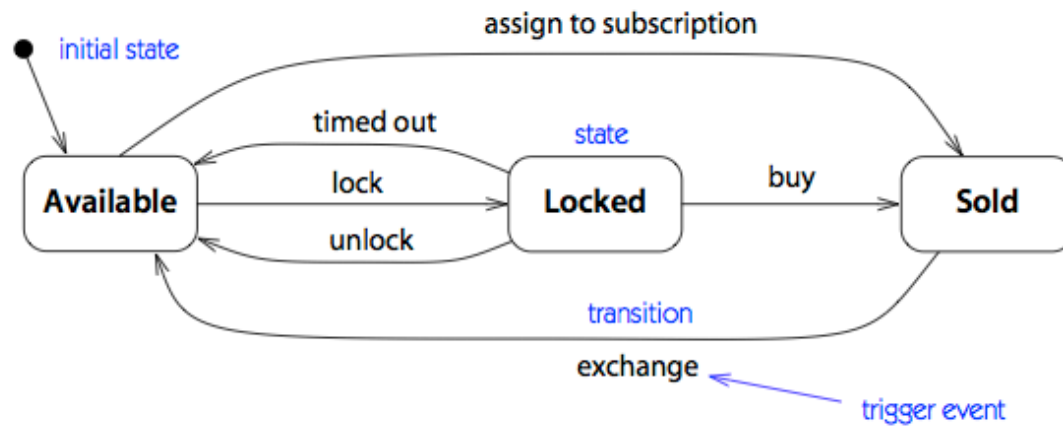
Ejemplo: Diagrama de secuencia



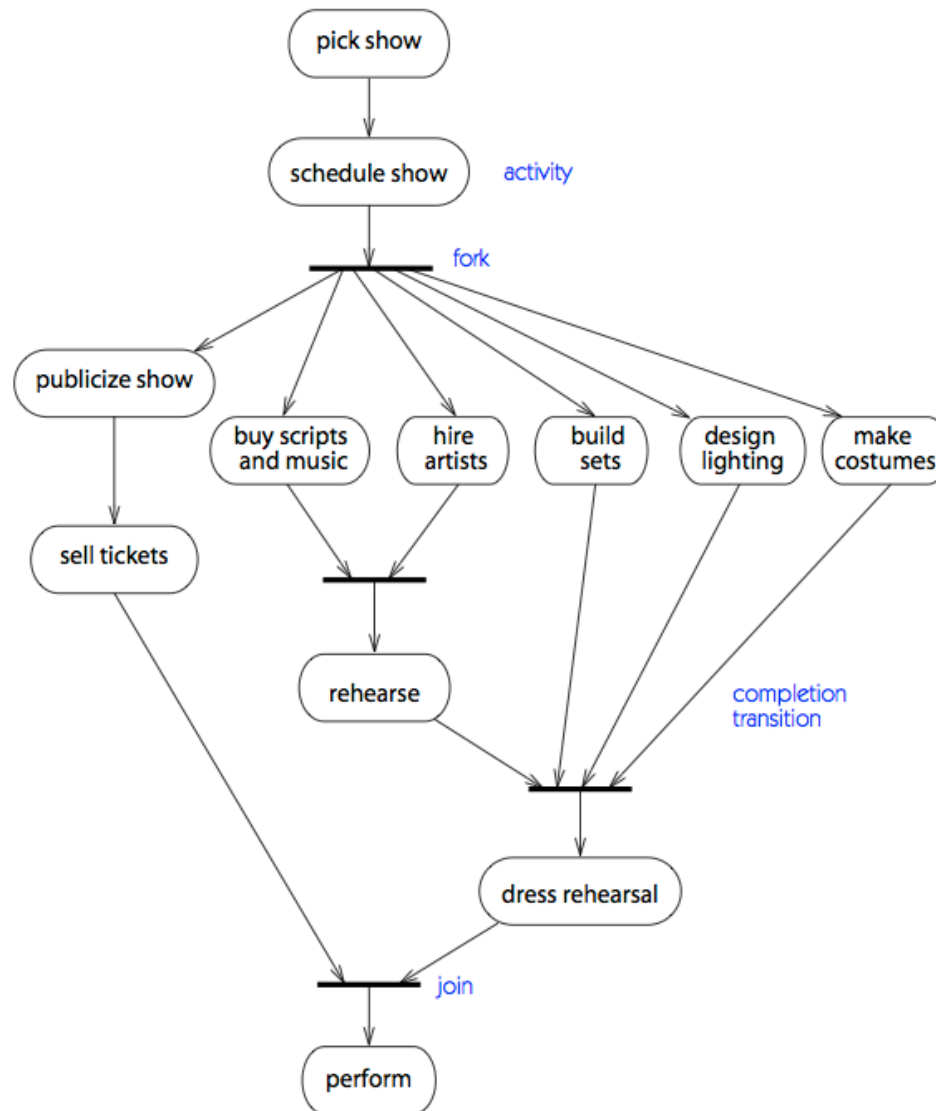
Ejemplo: Diagrama de colaboración



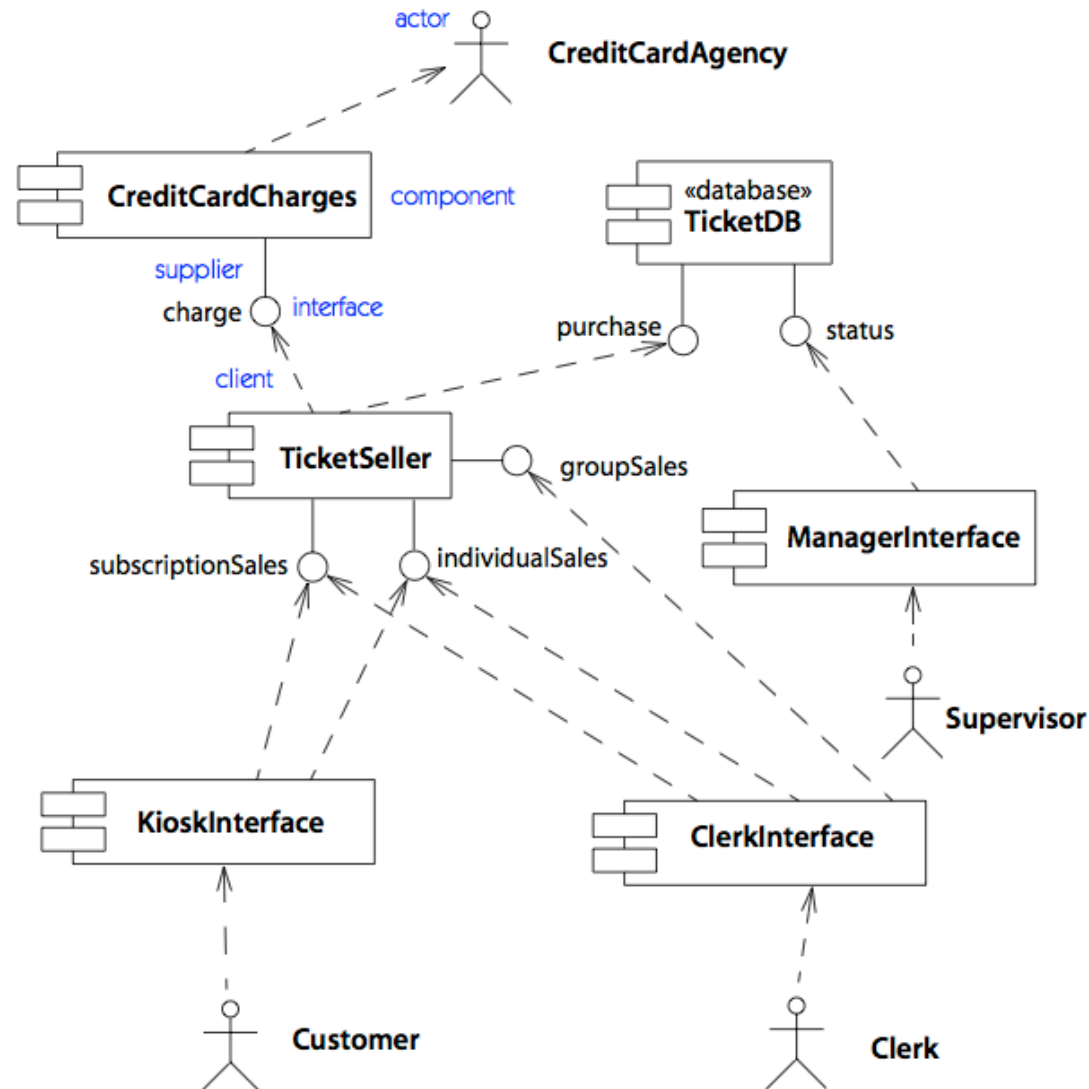
Ejemplo: Diagrama de estados



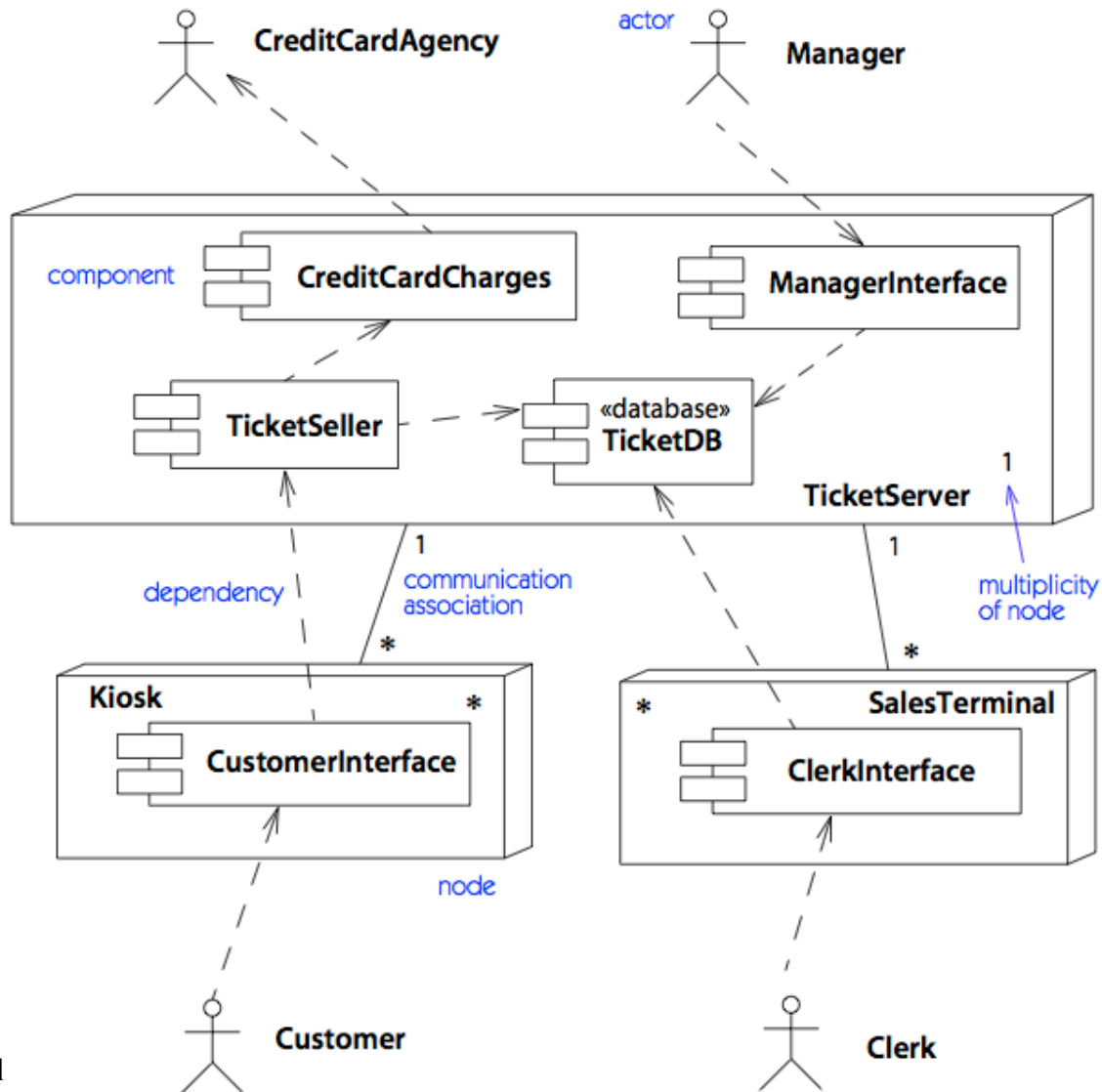
Ejemplo: Diagrama de actividad



Ejemplo: Diagrama de componentes



Ejemplo: Diagrama de despliegue



Ejemplo: Diagrama de paquetes

