Florida
CICLES FORMATIUS

Lenguaje Unificado de Modelado

UML

- **UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad.**
- **Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.**
- **UML ofrece un estándar para describir un "plano" del sistema (modelo)**

UML

- Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

- Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software

- UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

Tipos de Diagramas UML

Diagramas de comportamiento: Permiten exhibir comportamientos de un sistema o de los procesos de las organizaciones.
Incluyen :

- Diagrama de actividad
- Diagrama de estado
- Diagrama de caso típico
- Diagrama de interacción

Tipos de Diagramas UML

- **Diagramas de Interacción:** Es un subconjunto de los diagramas de comportamiento que permiten enfatizar las interacciones entre los objetos.
Incluyen:

- Diagrama de comunicaciones
- Diagrama de secuencia
- Diagrama de tiempo.

Tipos de Diagramas UML

- Diagramas de estructura: Muestran los elementos de una especificación que sean independientes del tiempo.

Incluyen:
- Diagrama de clases
- Diagrama de estructura
- Diagrama de componentes
- Diagrama despliegue
- Diagrama de objeto
- Diagrama de paquetes.

Diagrama de Clases

- Los diagramas de clases muestran las diferentes clases que componen un sistema y cómo se relacionan unas con otras.

- Son diagramas “estáticos” porque muestran las clases, junto con sus métodos y atributos, así como las relaciones estáticas entre ellas: qué clases “conocen” a qué otras clases o qué clases “son parte” de otras clases, pero no muestran los métodos mediante los que se invocan entre ellas.

Clases, Propiedades y Métodos

Las clases están representadas por rectángulos, con el nombre de la clase, y también pueden mostrar atributos y métodos de la clase en otros dos “compartimentos” dentro del rectángulo.

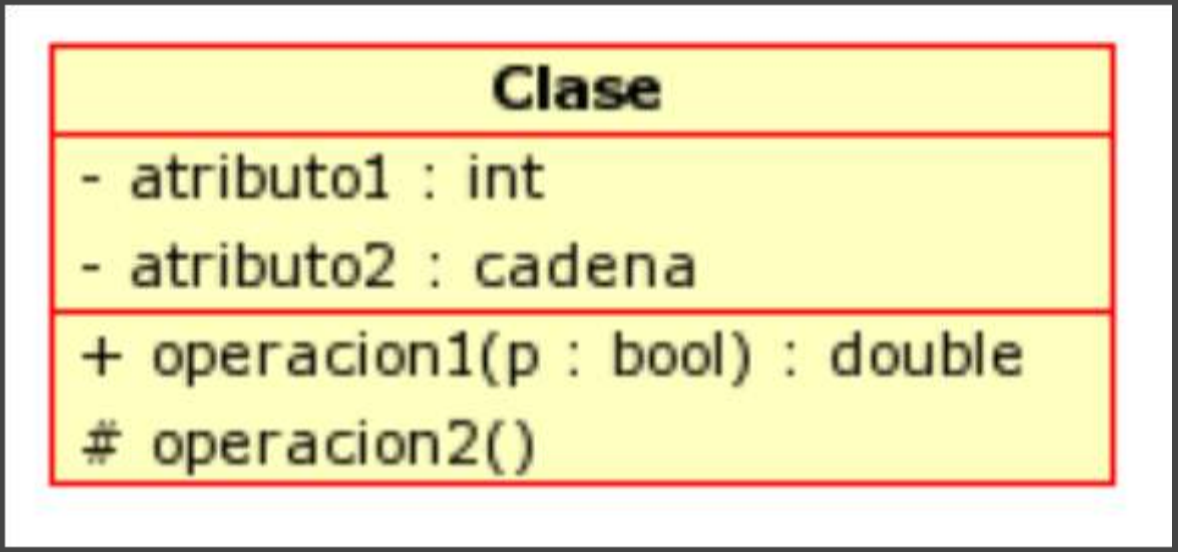
Los propiedades se muestran al menos con su nombre, y también pueden mostrar su tipo, valor inicial y otras propiedades.

Los métodos también se muestran al menos con su nombre, y pueden mostrar sus parámetros y valores de retorno.

Clases, Propiedades y Métodos

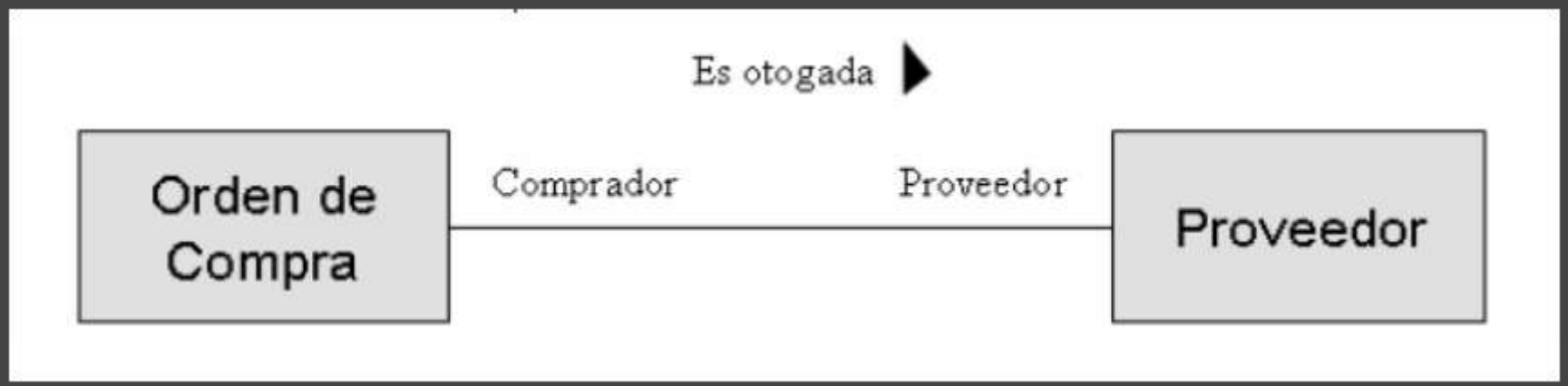
\+ Indica propiedades / métodos *públicos*
\# Indica propiedades / métodos *protegidos*
\- Indica propiedades / métodos *privados*

Representación de una clase:



Relaciones Entre Clases

En un diagrama de clases, los vínculos entre clases se representan por líneas. A las que se les de diferentes características dependiendo del tipo de relación. Adicionalmente, en los extremos de estas líneas, puede colocarse la descripción del **Rol** que asume cada clase en esa relación



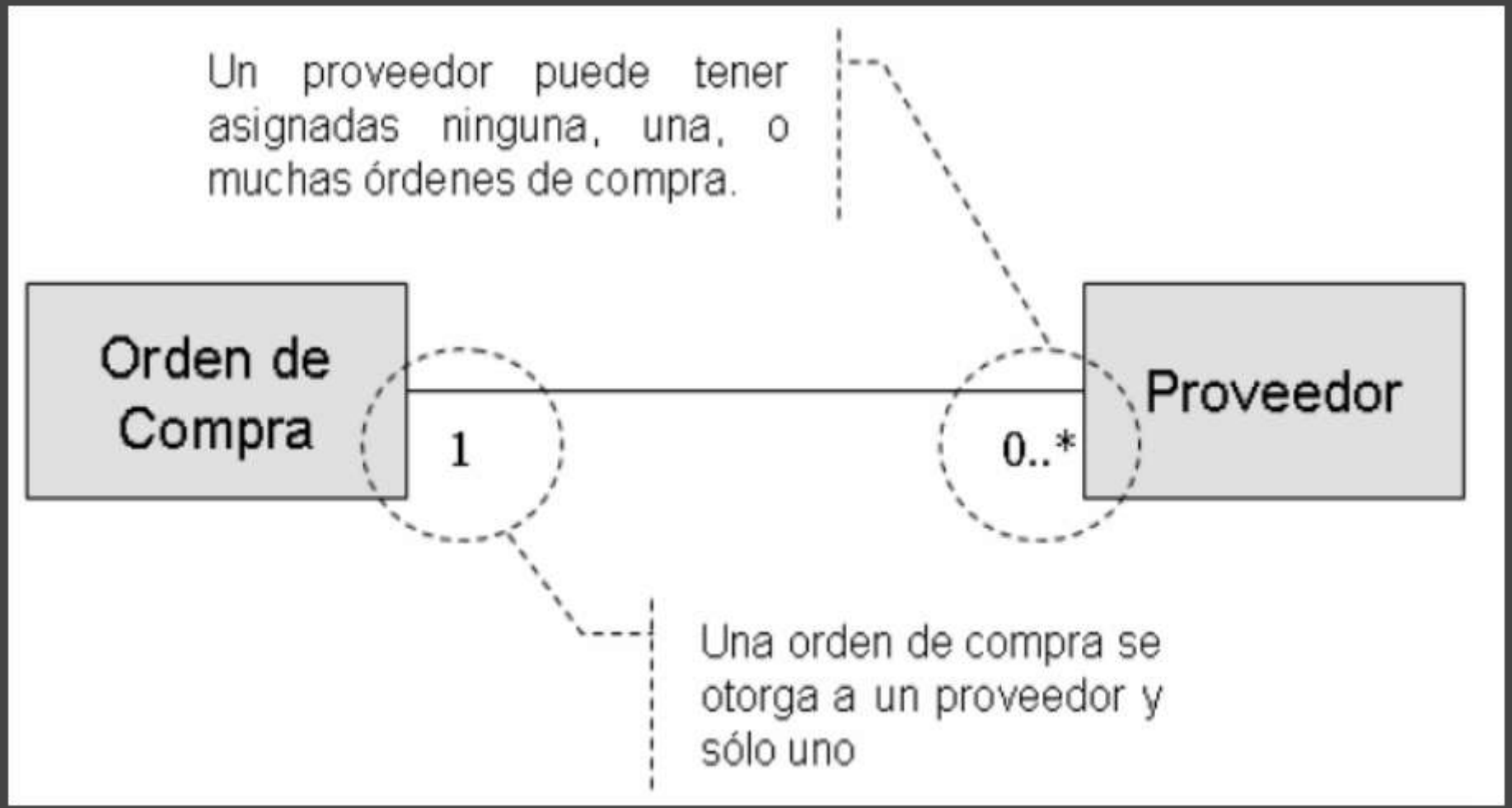
Cardinalidad

También en los extremos de la línea, se coloca la **Cardinalidad**, que describe cuántos objetos de cada clase pueden participar en la relación.(mínimo..máximo)
La **Cardinalidad** de una relación puede ser:

- Ninguno o Muchos 0..* o * o (0..n)
- Uno o muchos 1..* o (1..n)
- Exactamente uno 1 o (1)
- Un número fijo m o (m)
- Un numero variable 2..6 o (2..6)

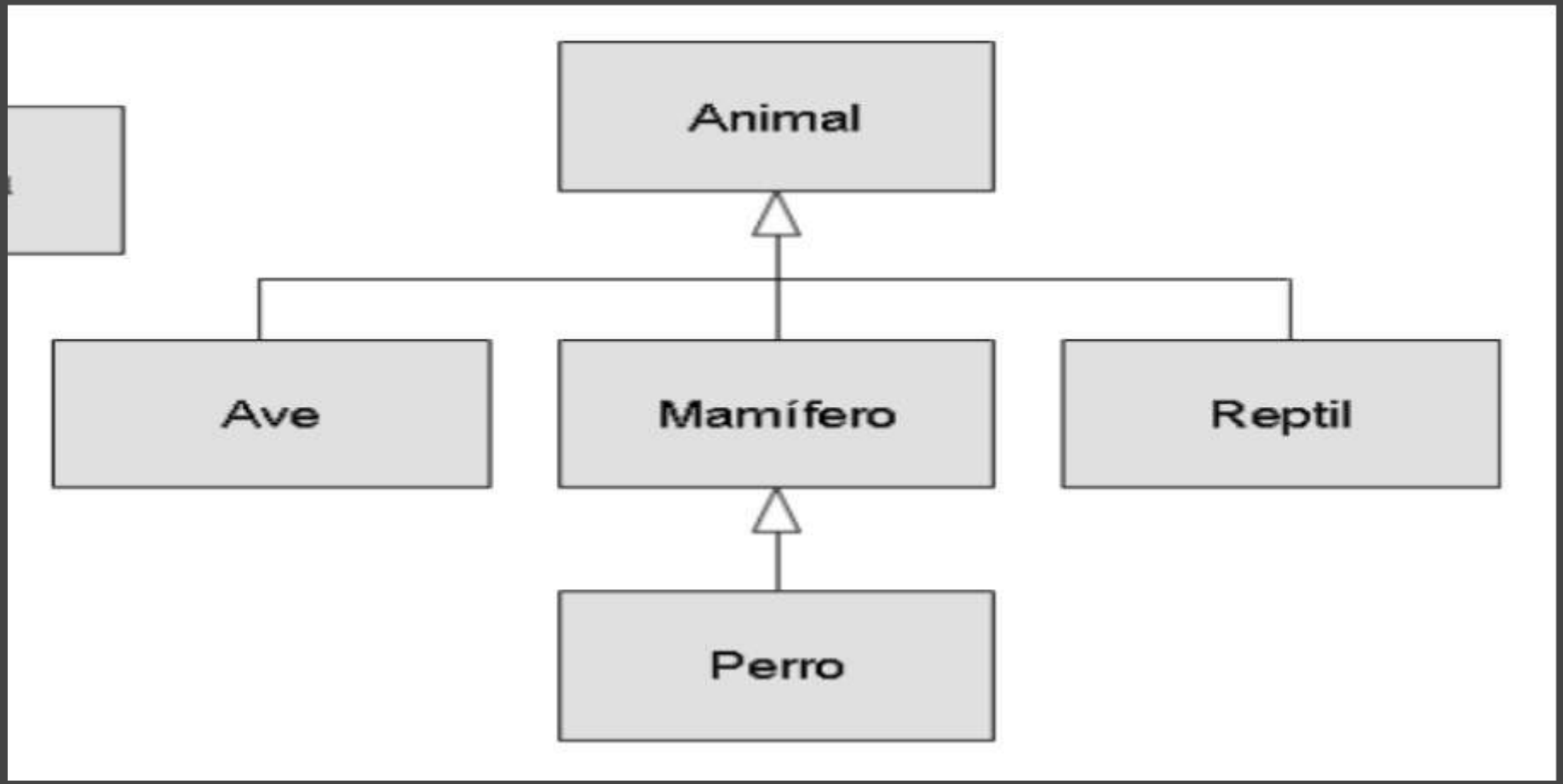
Cardinalidad

Ejemplo Cardinalidad :



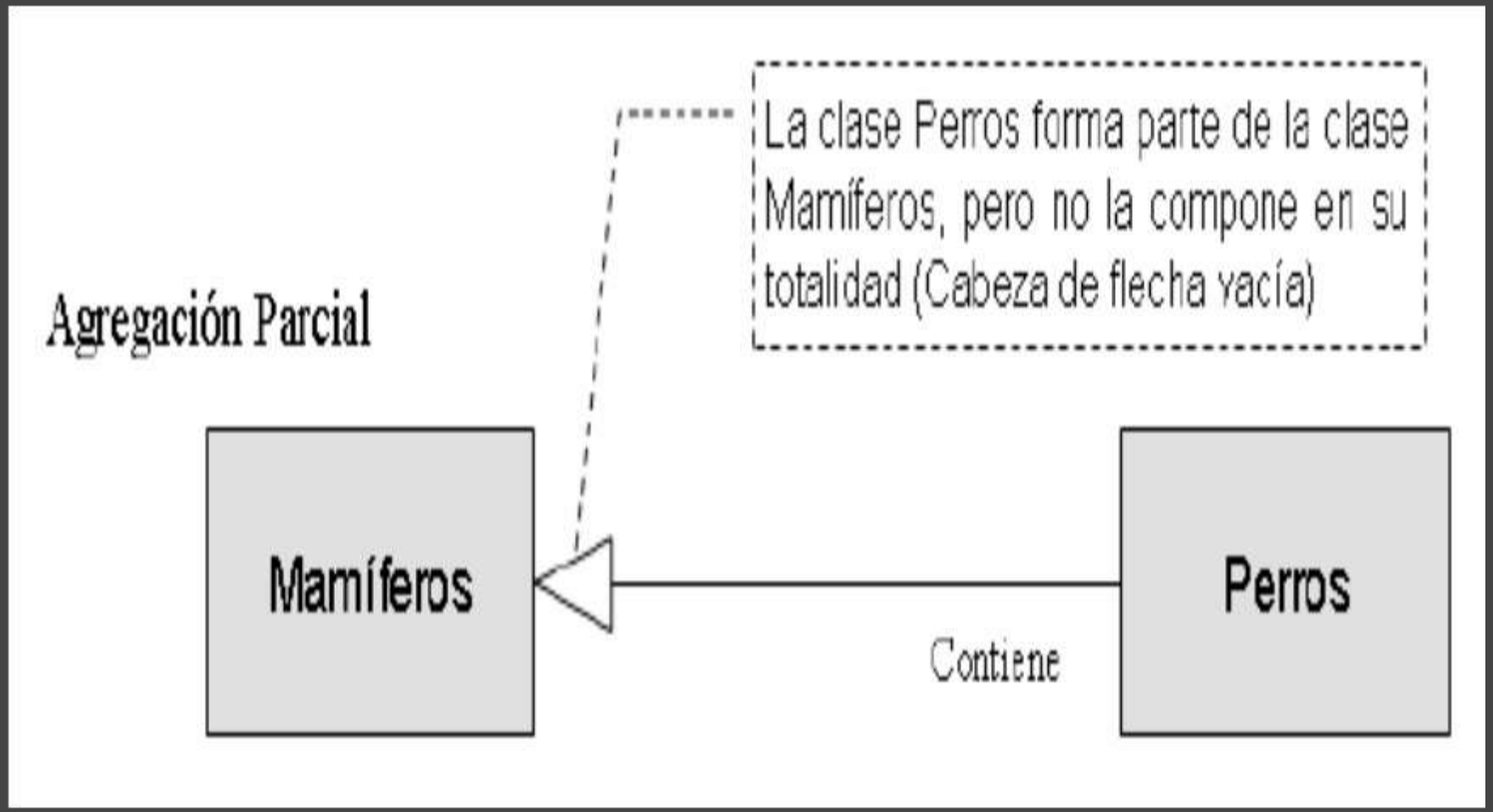
Generalización

Como hemos visto anteriormente entre dos clases puede existir una relación de **Herencia** o, en la terminología de UML , de **Generalización**.

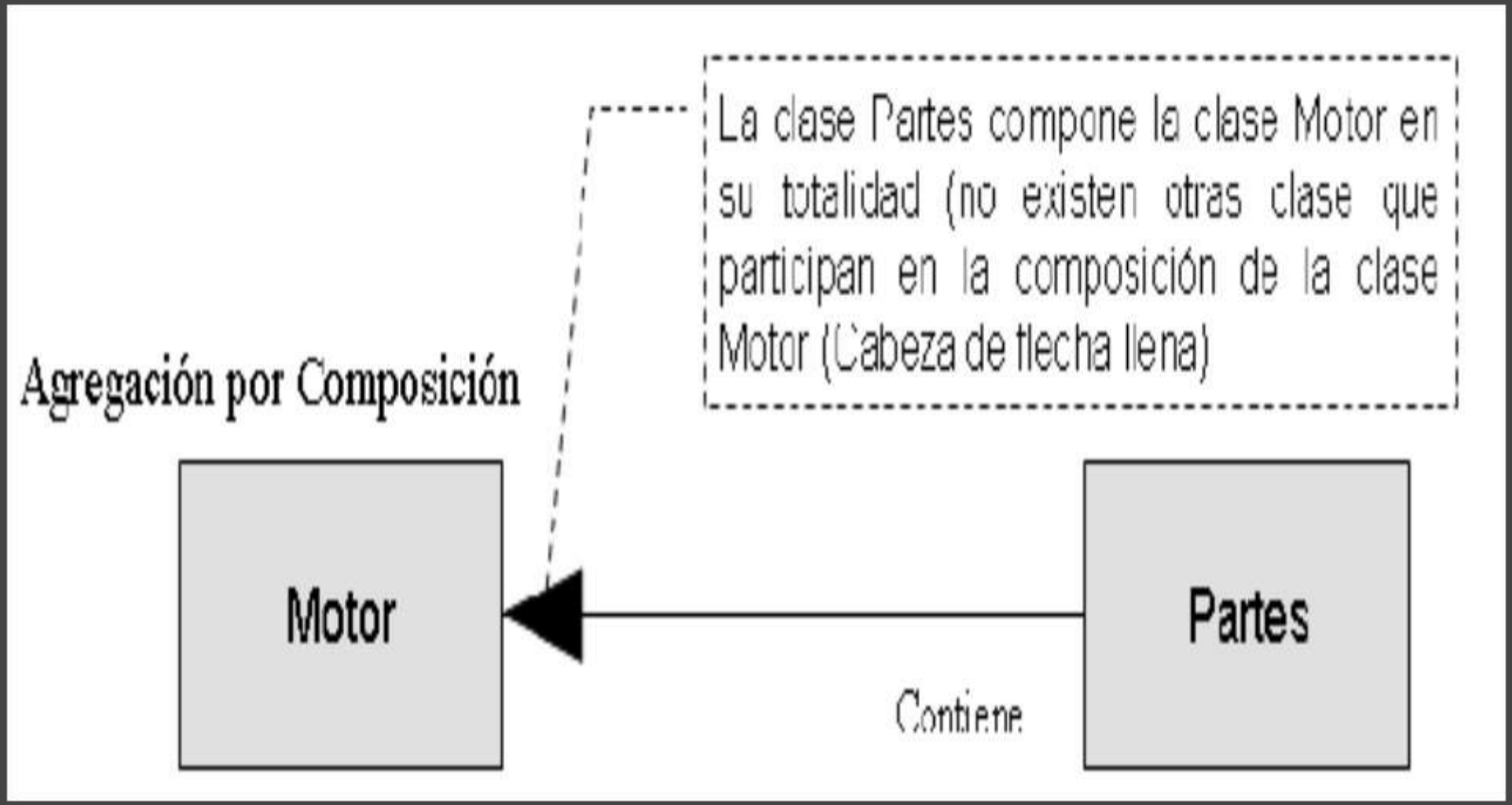


Agregación

- Cuando una clase es parte o componente de otra clase se le denomina **Agregación**.

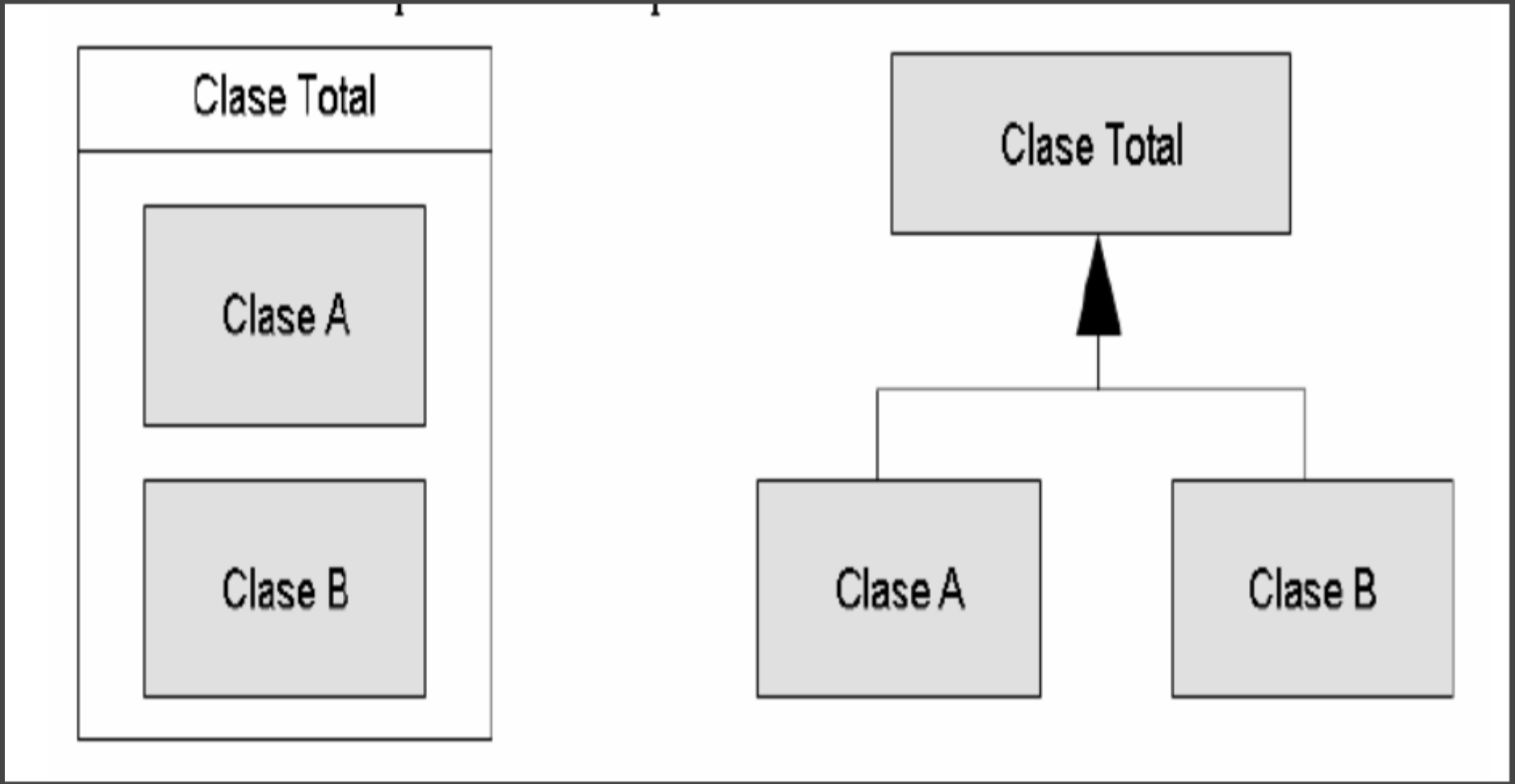


Agregación


La clase Partes compone la clase Motor en su totalidad (no existen otras clase que participan en la composición de la clase Motor (Cabeza de flecha llena)
Agregación por Composición
Motor
Partes
Contiene

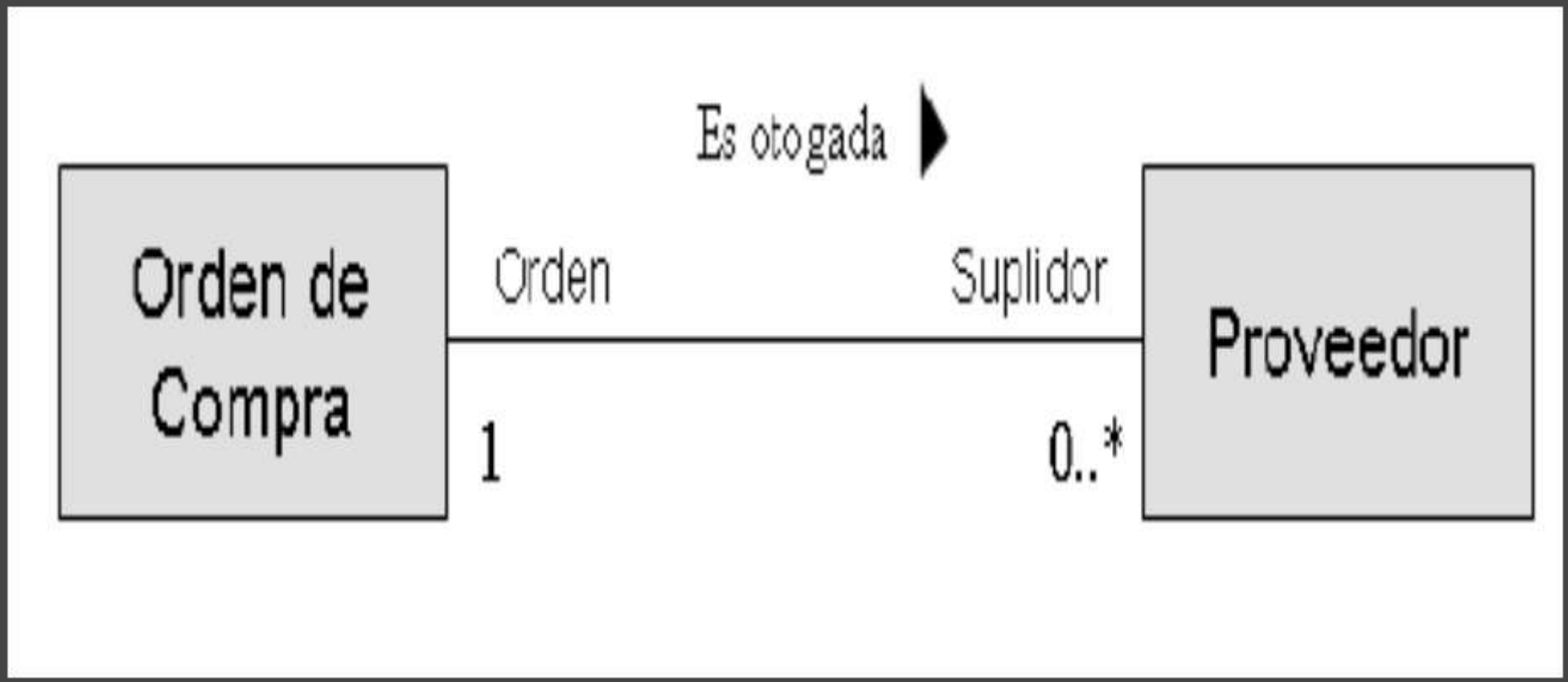
Agregación

Alternativas para mostrar Agregación por Composición



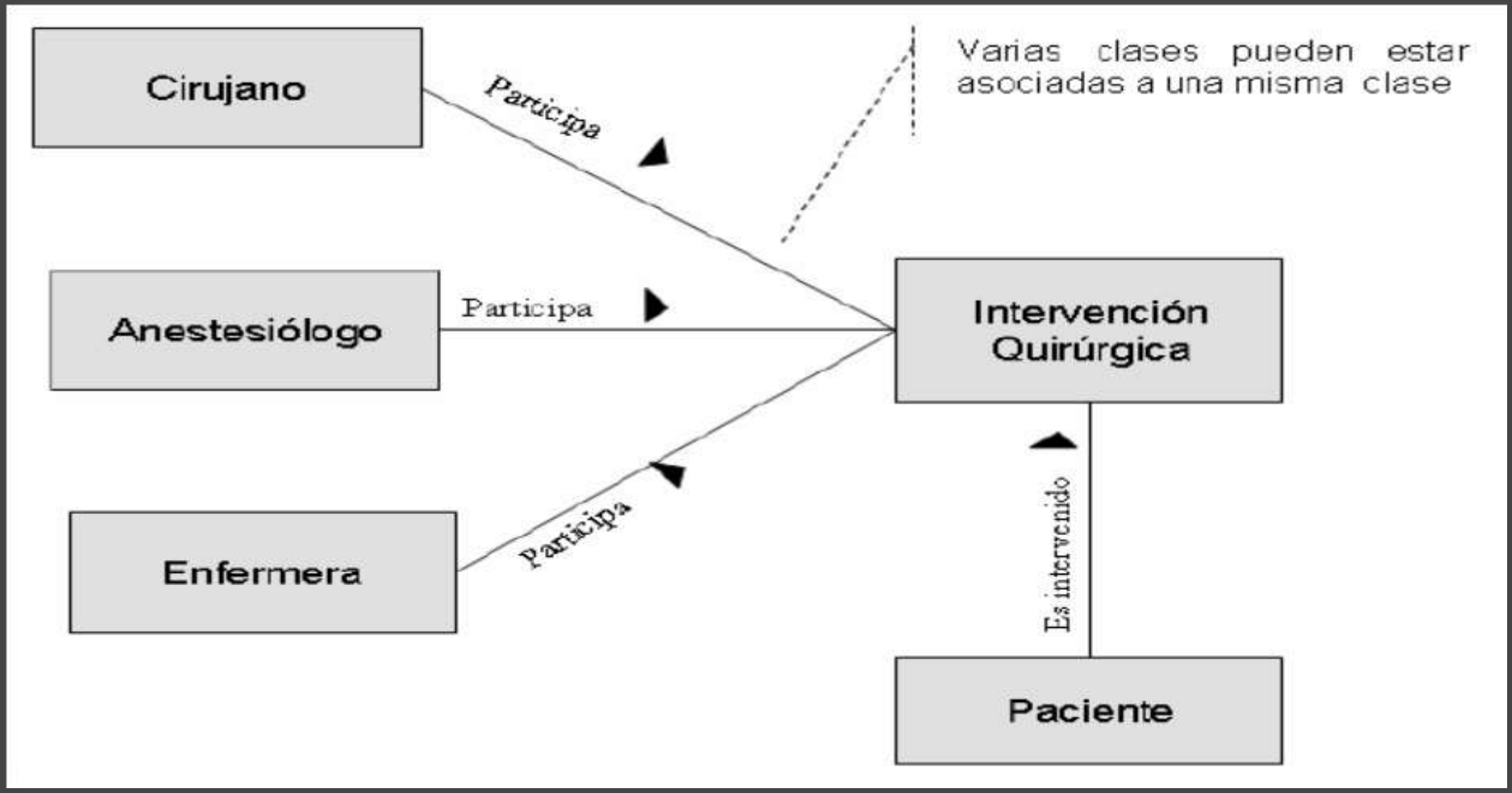
Asociación

- Si existe un vínculo entre los objetos de las clases se denomina relación de **Asociación**.



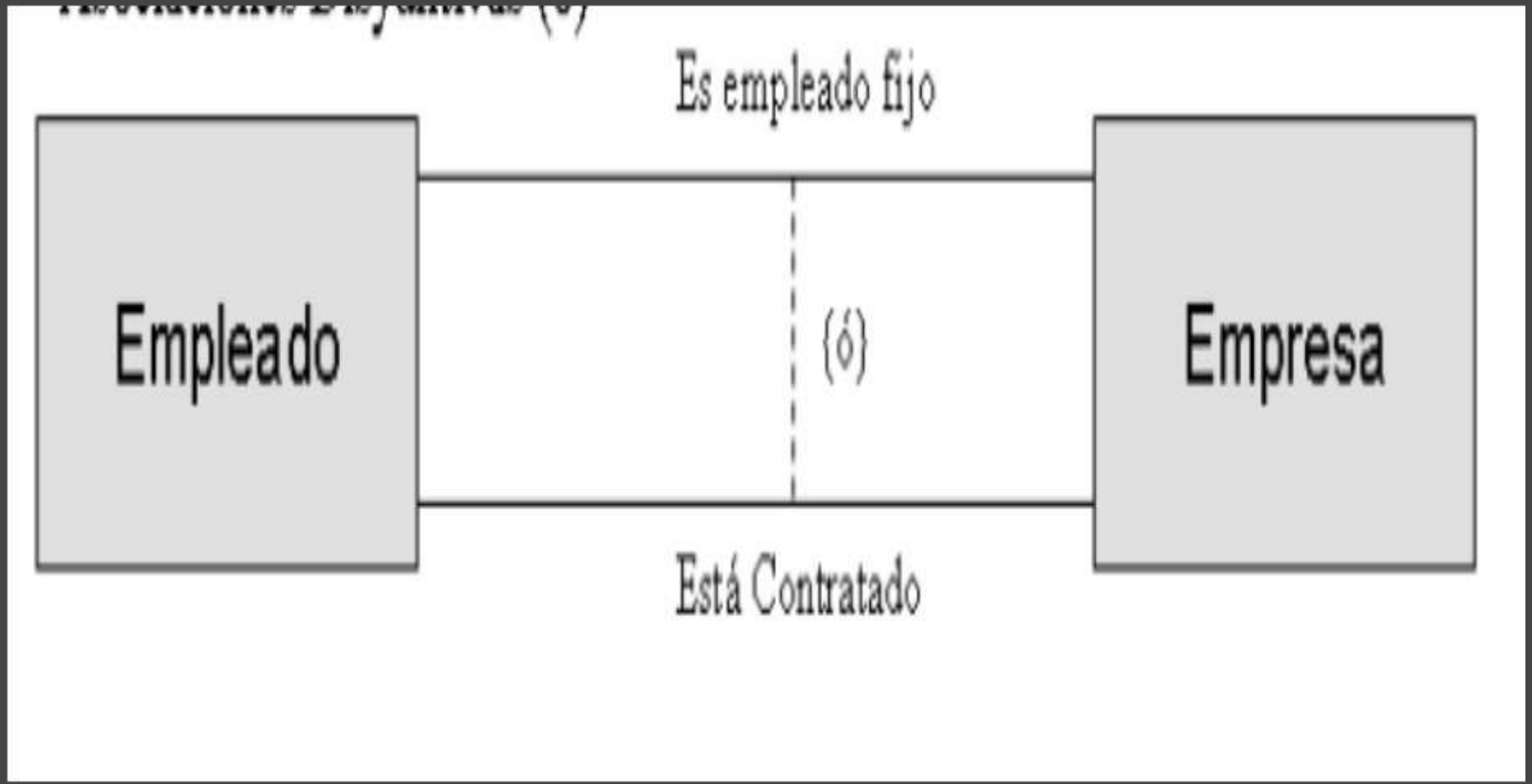
Asociación

Muchas clases pueden estar asociadas a una misma clase



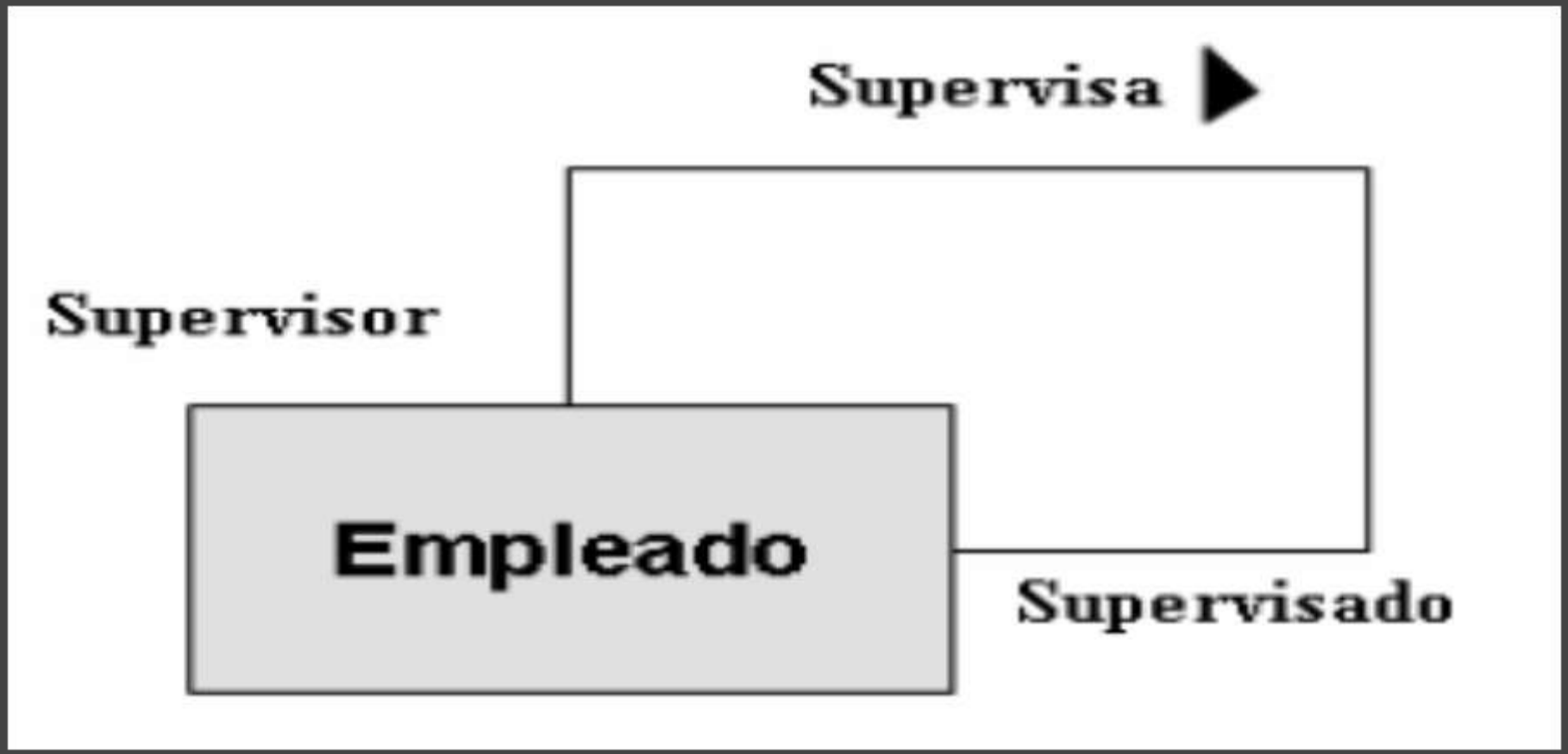
Asociaciónes Disyuntivas

Asociaciones formadas por clases que se relacionan en forma alternativa



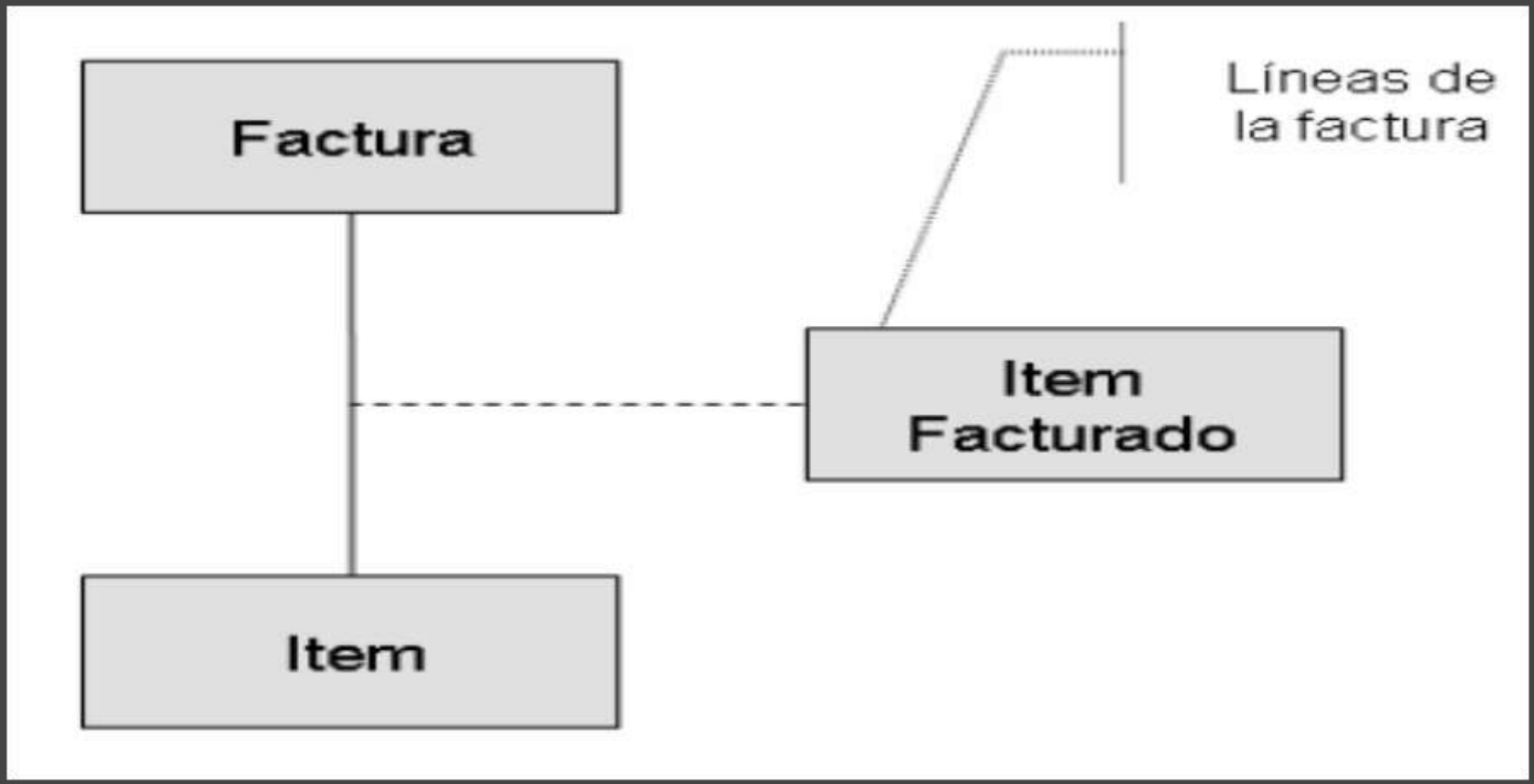
Asociación Recursiva

Para una misma clase puede existir una asociación recursiva



Clase Asociativa

Existen asociaciones que no sólo contienen información de las clases asociadas, sino que también contienen información própia de la asociación.



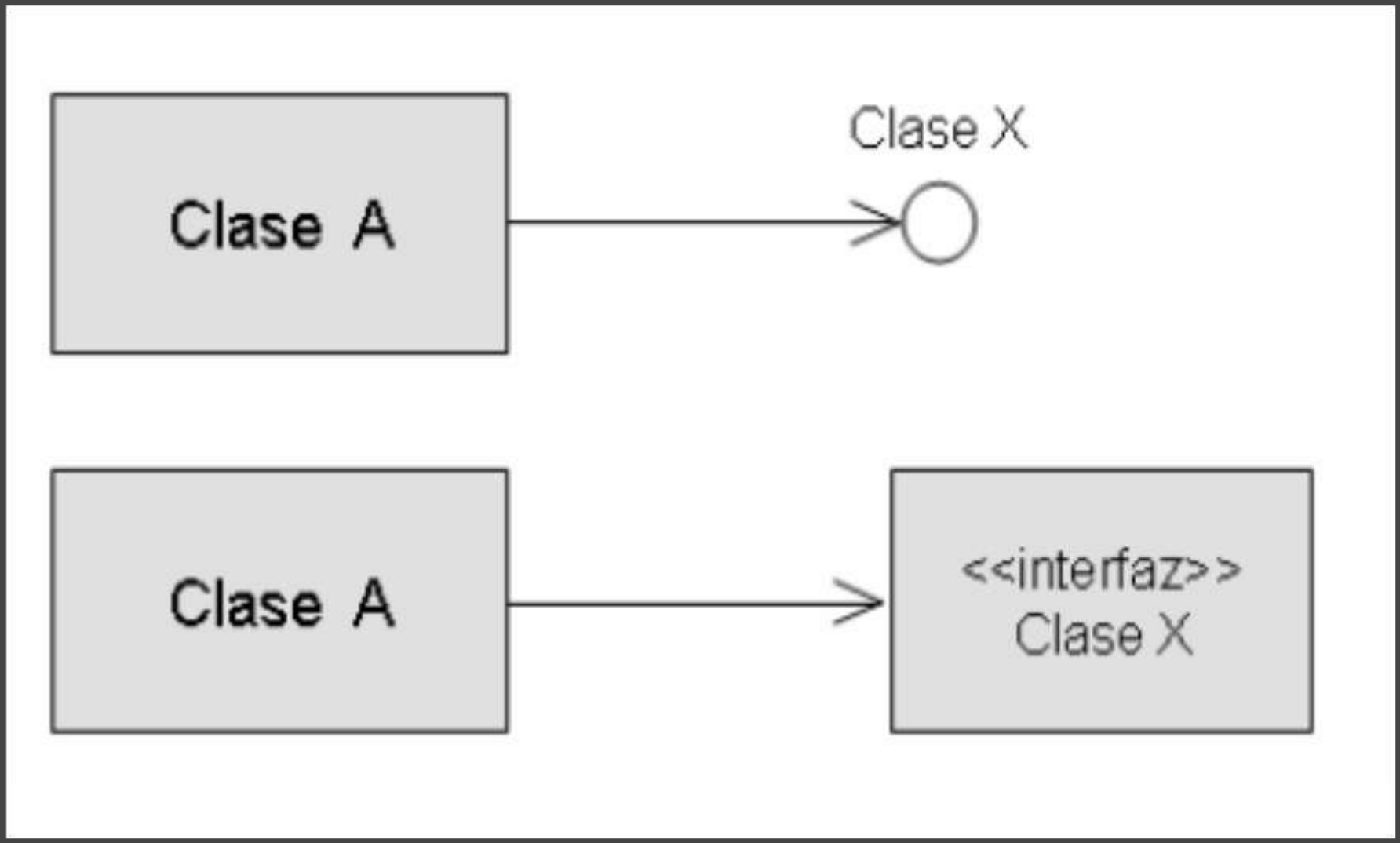
Interfaces / Realización

Existen clase que, aun siendo totalmente diferentes, tienen en común una série de métodos, a estas se les denomina Interfaces.

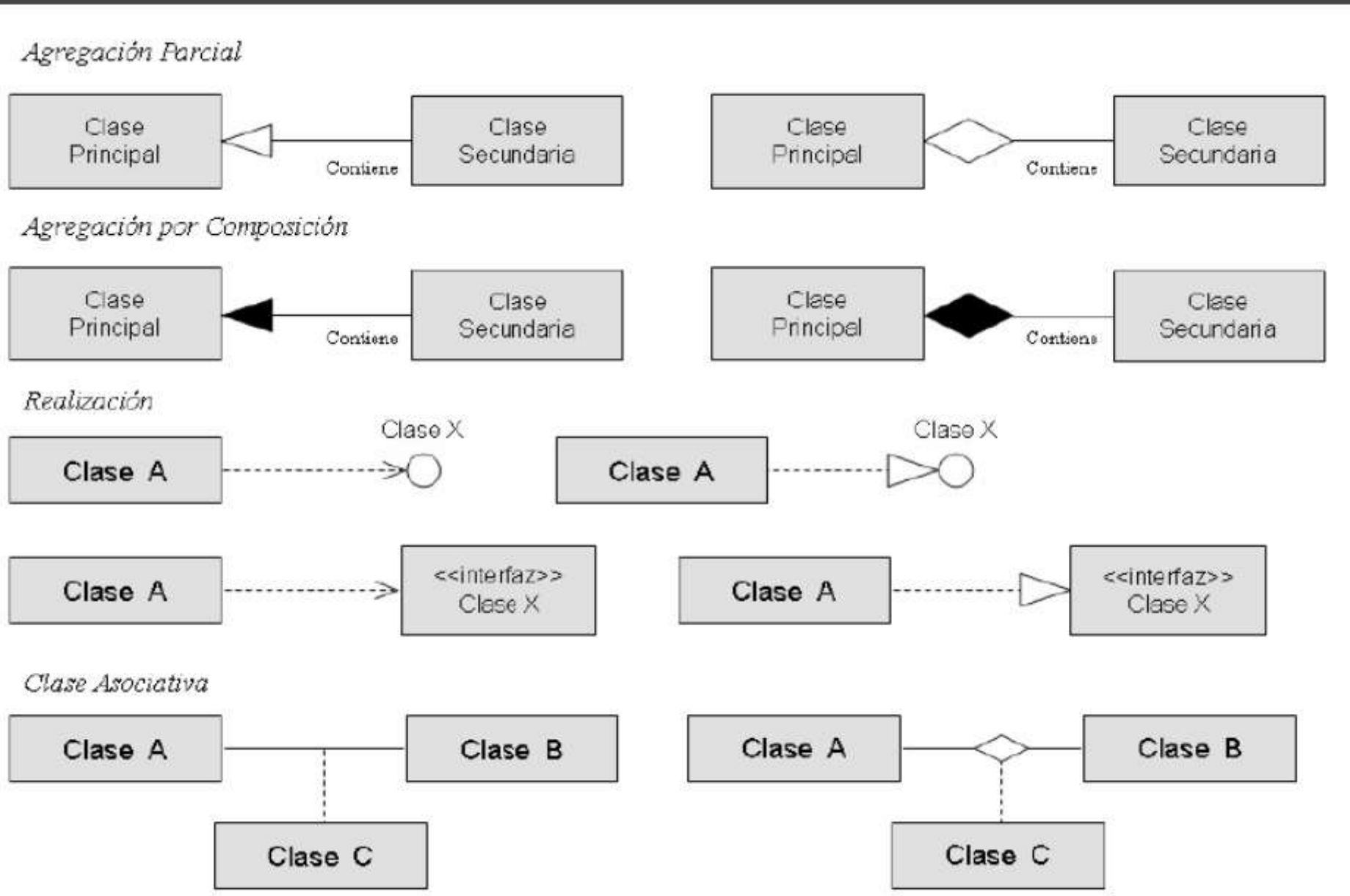
Una vez definida, una interfaz puede ser reutilizada en diversos sistemas o módulos por lo que puede desarrollarse por separado y tratarse como una clase que sólo contiene métodos.

La relación que vincula una clase con una interfaz se denomina Realización

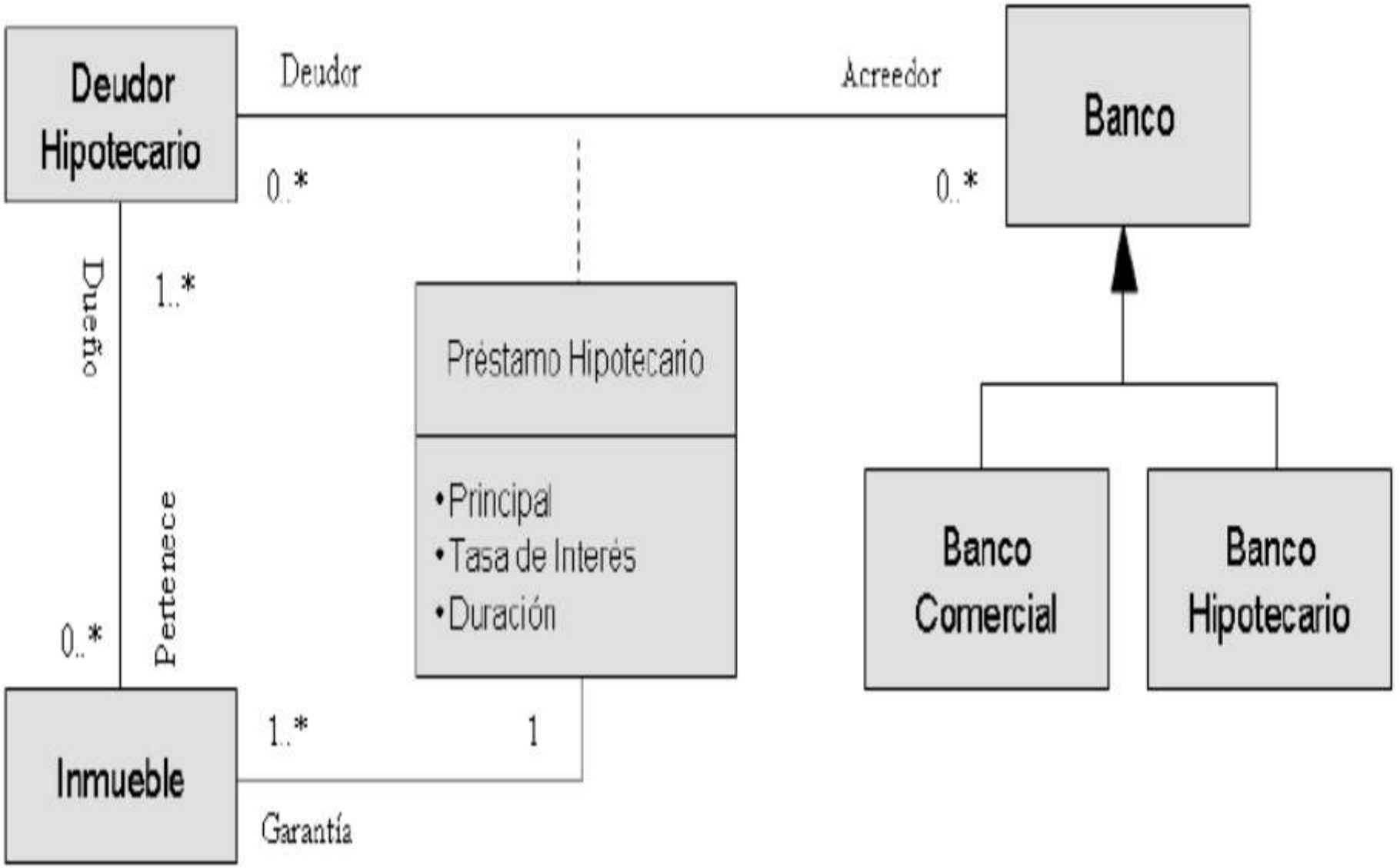
Interfaces / Realización



Notaciones Alternativas



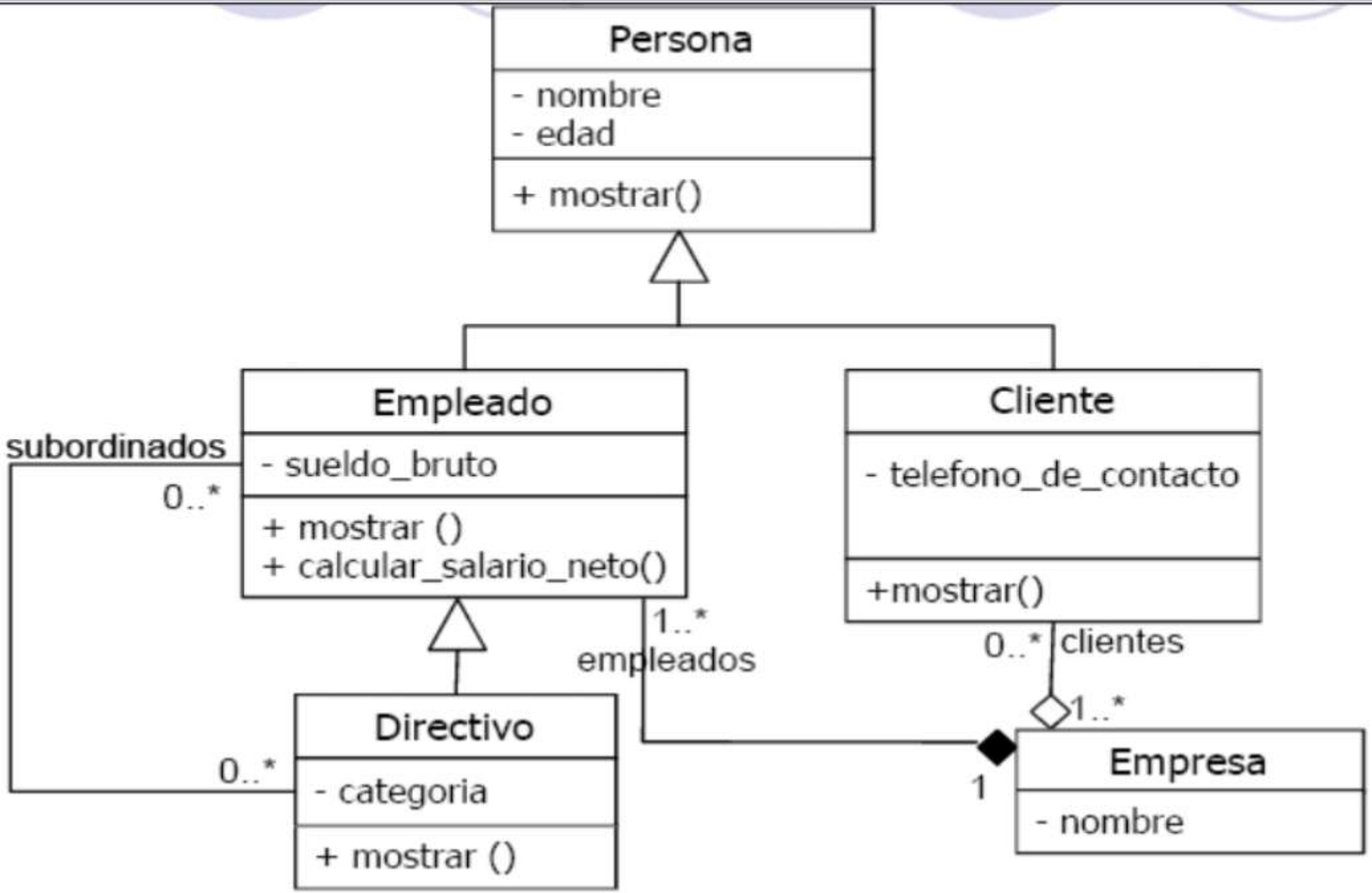
Ejemplo


Deudor Hipotecario
Deudor
Acreedor
Banco
0..*
0..*
Dueño
1..*
Préstamo Hipotecario
•Principal
•Tasa de Interés
•Duración
Pertenece
0..*
Banco Comercial
Banco Hipotecario
1..*
1
Inmueble
Garantía

Ejercicio Práctico

- Una aplicación necesita almacenar información sobre empresas, sus empleados y sus clientes.
  - Ambos se caracterizan por su nombre y edad
  - Los empleados tienen un sueldo bruto,los empleados que son directivos tienen una categoría, asi com un conjunto de empleados subordinados
  - De los clientes además se necesita conocer su teléfono de contacto
  - La aplicación necesita mostrar los datos de empleados y clientes

Solución Ejercicio



Diagramas de Estado

- Los diagramas de estado muestran los diferentes estados de un objeto o sistema durante su vida y los estímulos que provocan sus cambios de estado.

Estados como:

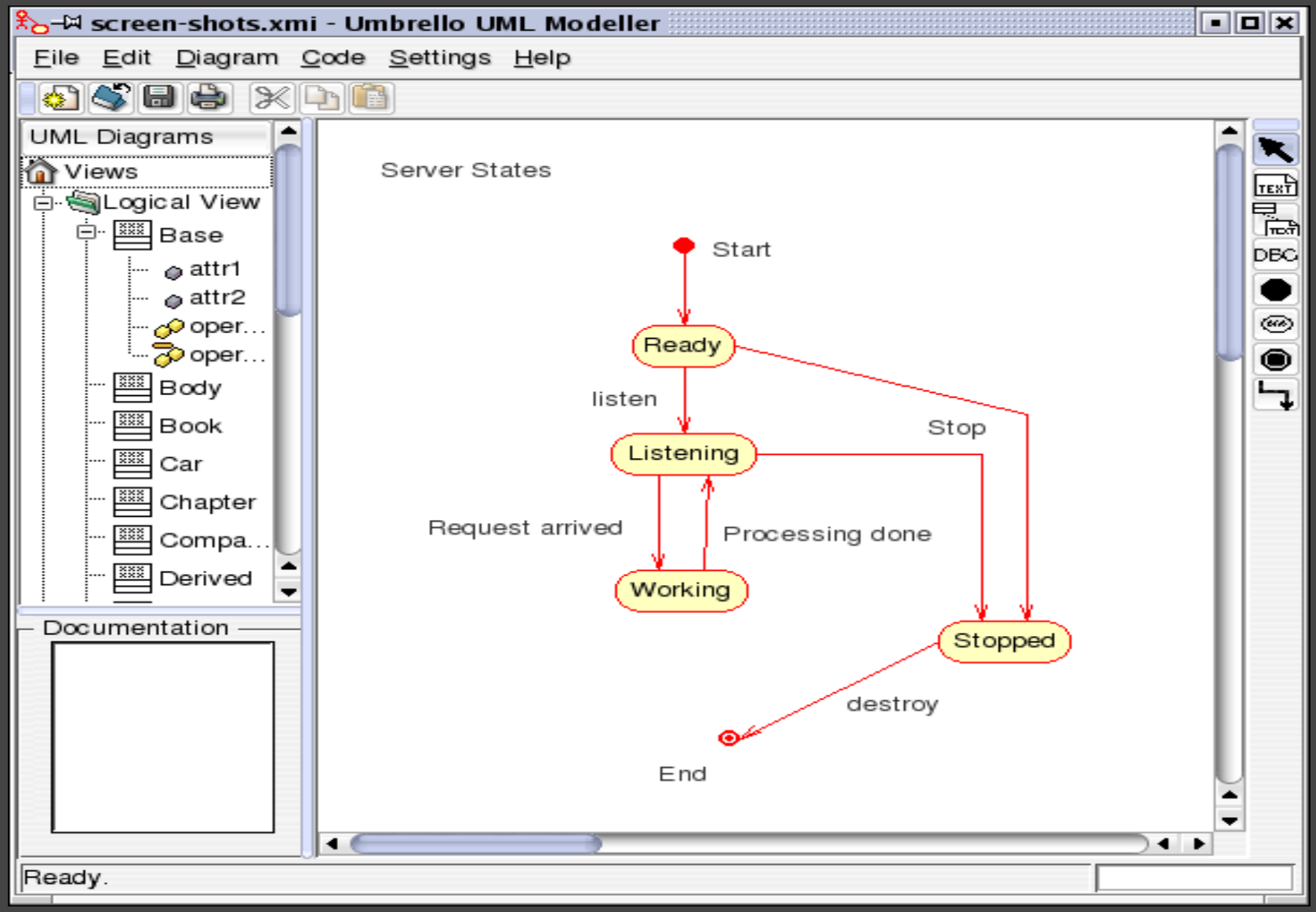
- trabajando
- detenido
- escuchando
- Listo

Diagramas de Estado

Estimulos como:

- Se crea el objeto
- El objeto recibe un mensaje de escucha
- El objeto recibe un mensaje de detención
- Un cliente solicita una conexión a través de la red
- Un cliente finaliza una solicitud
- La solicitud se ejecuta y ser termina

Diagramas de Estado



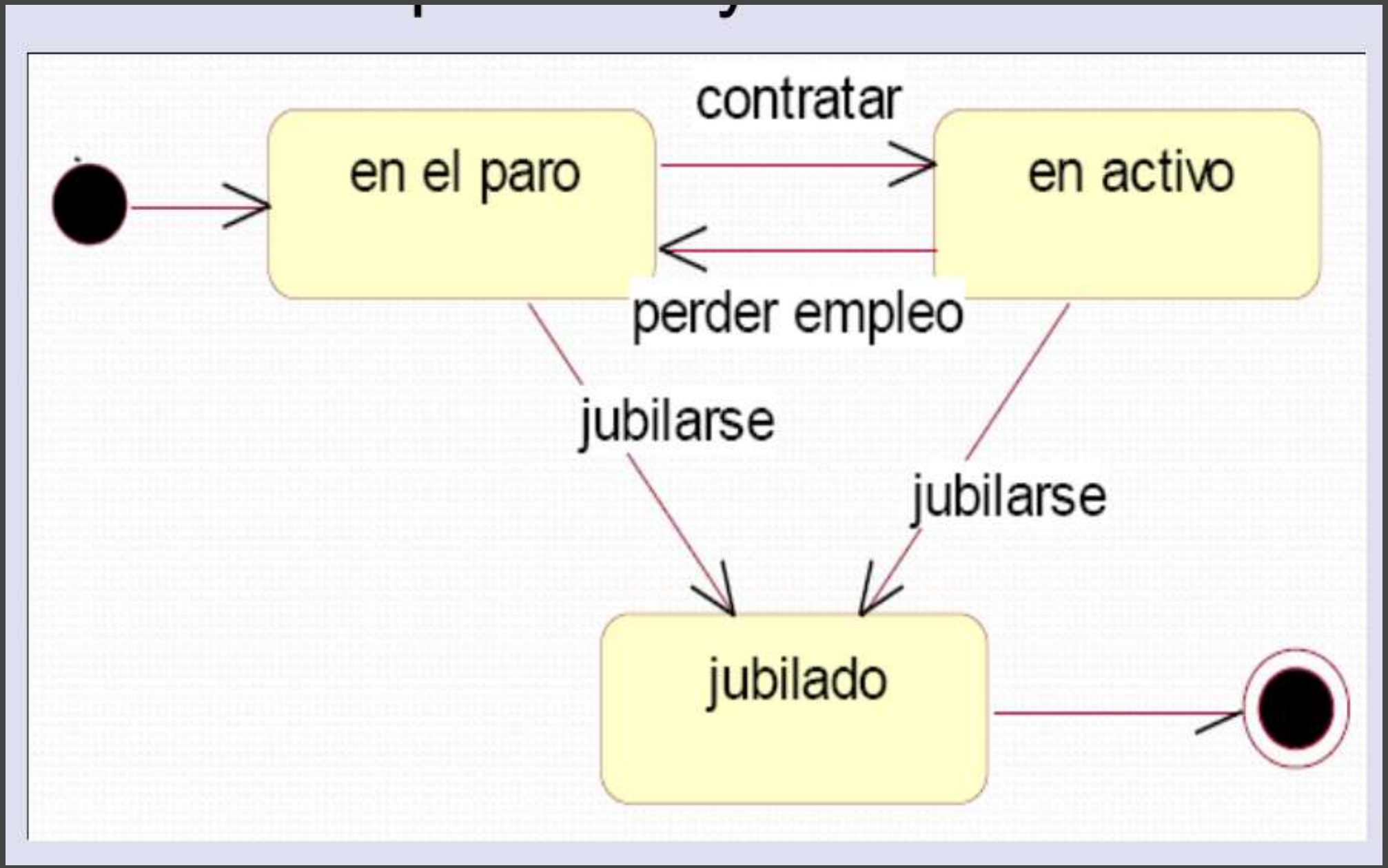
Ejercico Propuesto

Realizar el diagrama de estados para una persona y su vida laboral

Suponer solo tres estados:

- Activo
- Parado
- Jubilado

Solución Ejercicio

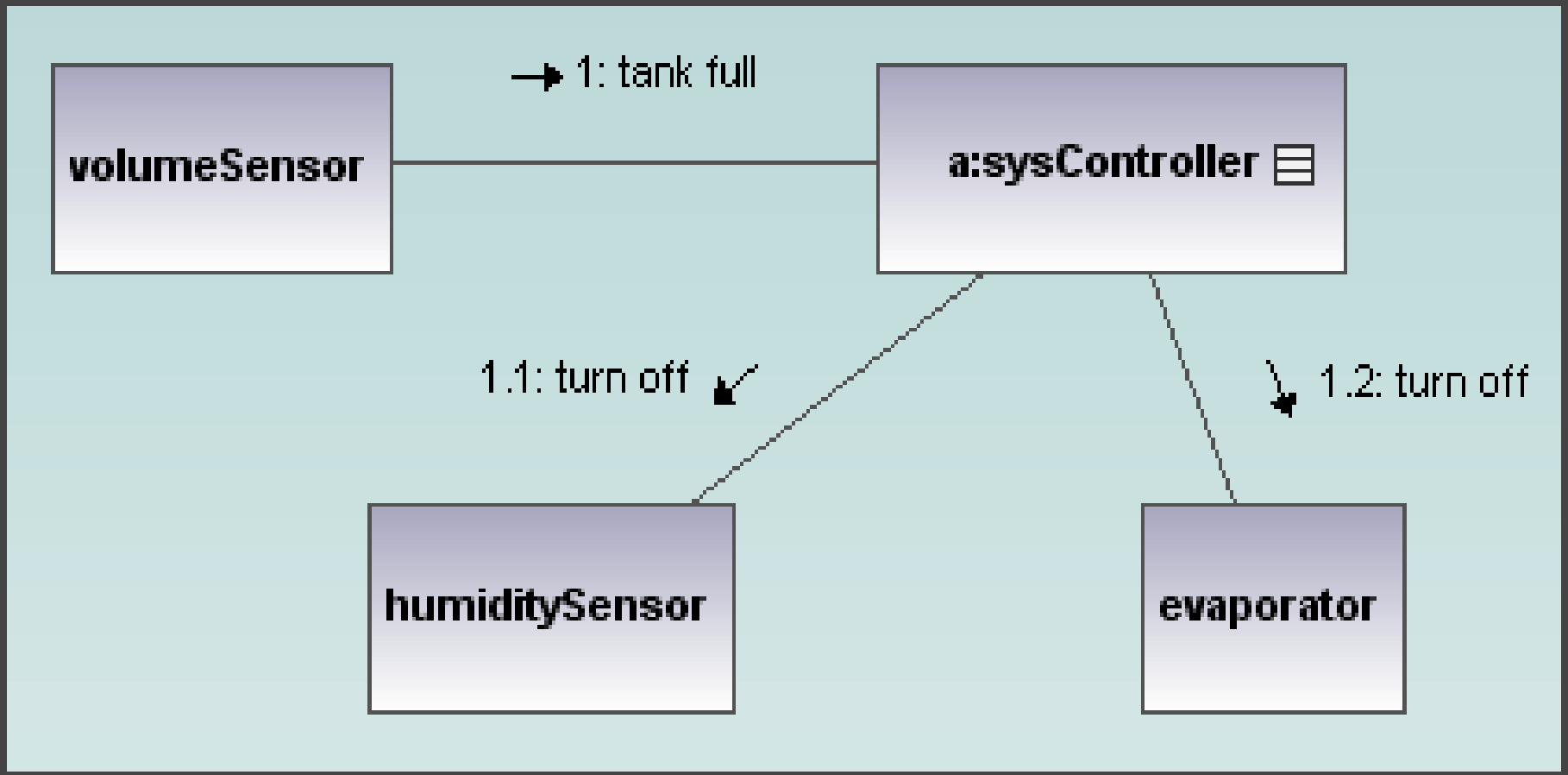


Diagramas de Comunicación

- muestran la comunicacion entre los objetos del sistema, mostrando un orden numerado de mensajes.
Destacan:

- **Mensajes** enviados entre los objetos
- **Enlaces** entre los objetos
- Un **escenario concreto,** sin condiciones

Diagramas de Comunicación



Ejercicio propuesto

Realizar un diagrama de comunicación de una llamada telefonica entre 2 usuarios através de un servidor.

Diagramas de Comunicación

Diagrama de Componentes

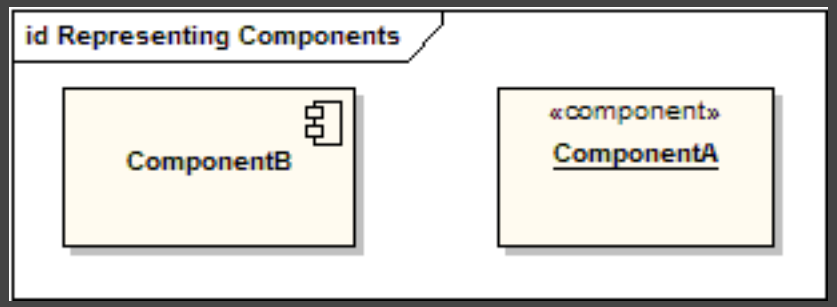
- Los diagramas de componentes representan las distintas partes del software (archivos, cabeceras, módulos, ejecutables...) que representan un sistema y las dependencias existentes entre ellas.

- Normalmente un mismo componente se puede implementar por mas de una clase, por lo que la representación de un componente abarca una gran parte del sistema.

- Los diagramas de componentes no siempre representan el sistema completo ya que se este se suele dividir en varios diagramas.

Componentes

Los componentes se representan mediante un clasificador rectangular con una clave o icono en la esquina superior derecha.

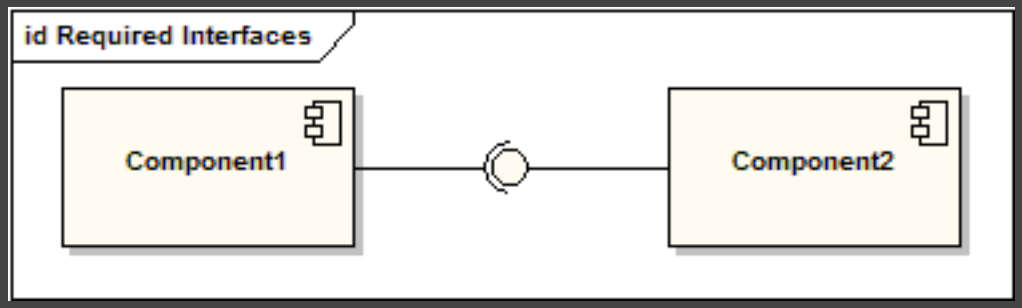


Representación de ComponenteB y ComponenteA

Interfaces requeridas

Para unir una interfaz requerida con la interfaz proporcionada correspondiente se usan los conectores de ensamble.

De esta manera conseguimos que un componente provea los servicios requeridos por otro componente del diagrama.

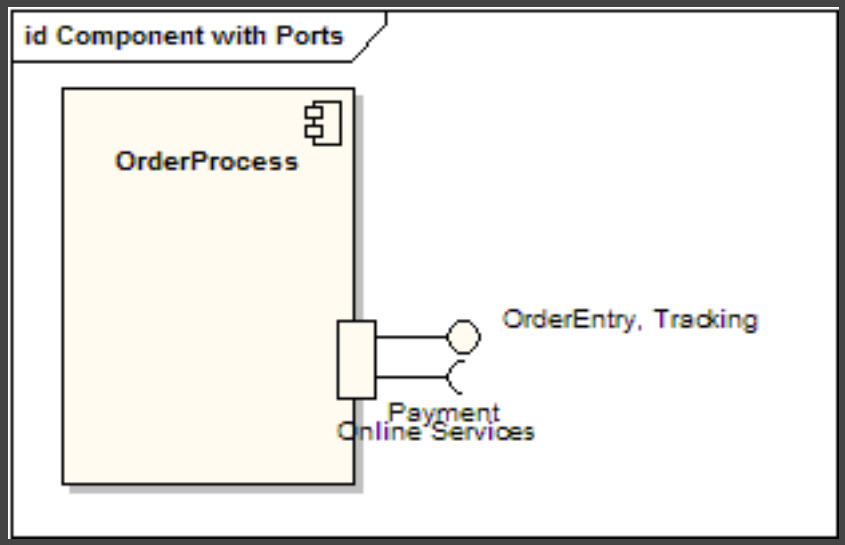


Componente1 requiere Componente2

Componentes con puertos

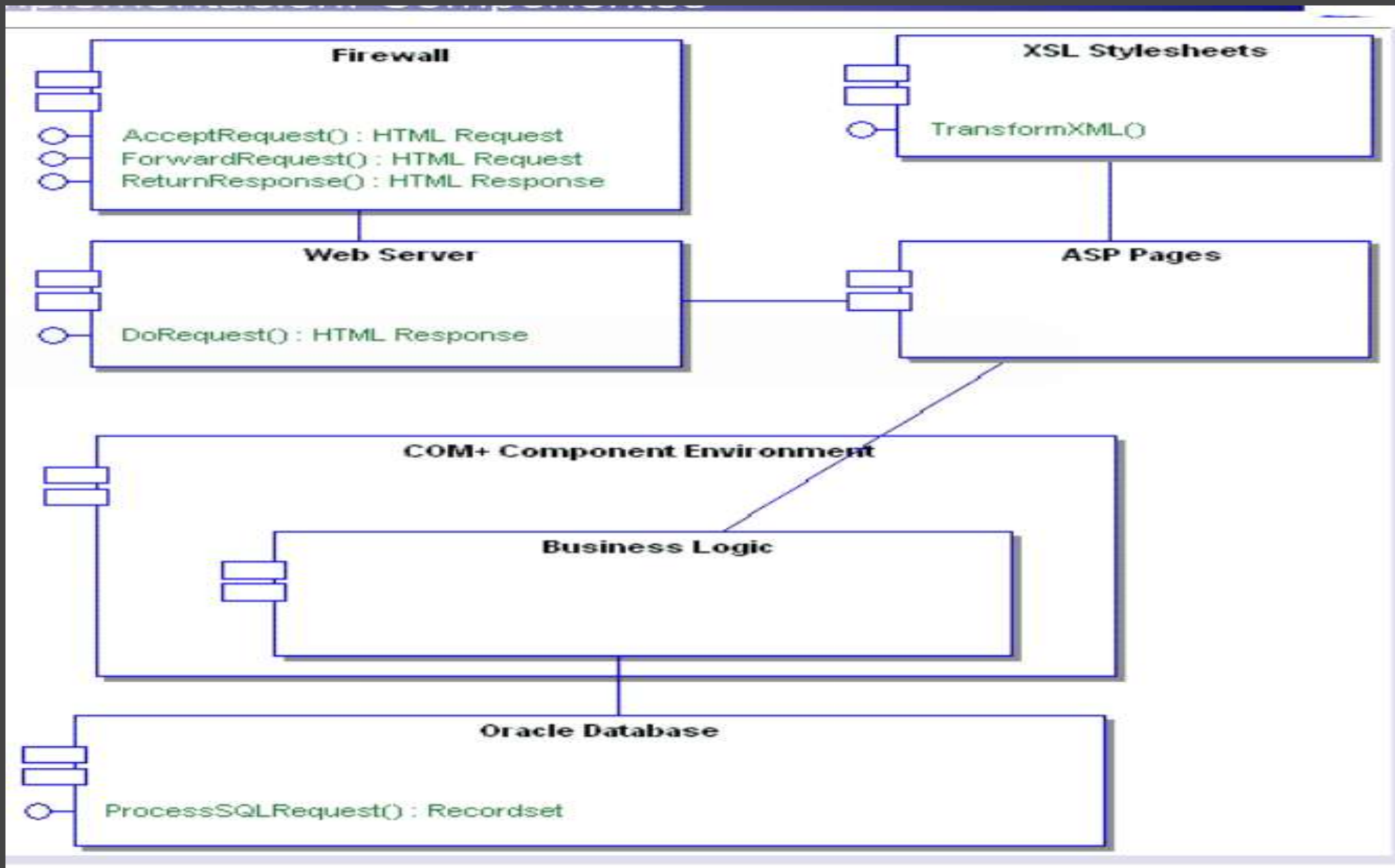
Los puertos de un componente pueden especificar entrada, salida o ambos.

Permiten representar un servicio o comportamiento a su entorno o requerido por otro componente.



Puerto para servicios con interface Tracking proporcionada y Pago requerida

Ejemplo


Firewall
AcceptRequest() : HTML Request
ForwardRequest() : HTML Request
ReturnResponse() : HTML Response
XSL Stylesheets
TransformXML()
Web Server
DoRequest() : HTML Response
ASP Pages
COM+ Component Environment
Business Logic
Oracle Database
ProcessSQLRequest() : Recordset

Ejercicio Propuesto

Realizar el digrama de componentes para un software de seguridad con los siguientes elementos:

-Autoridad Certificadora (Certificate Authority

-Navegador (Browser)

-Servidor Web

-Otros Elementos (Firewall, etc..)

Solución Ejercicio

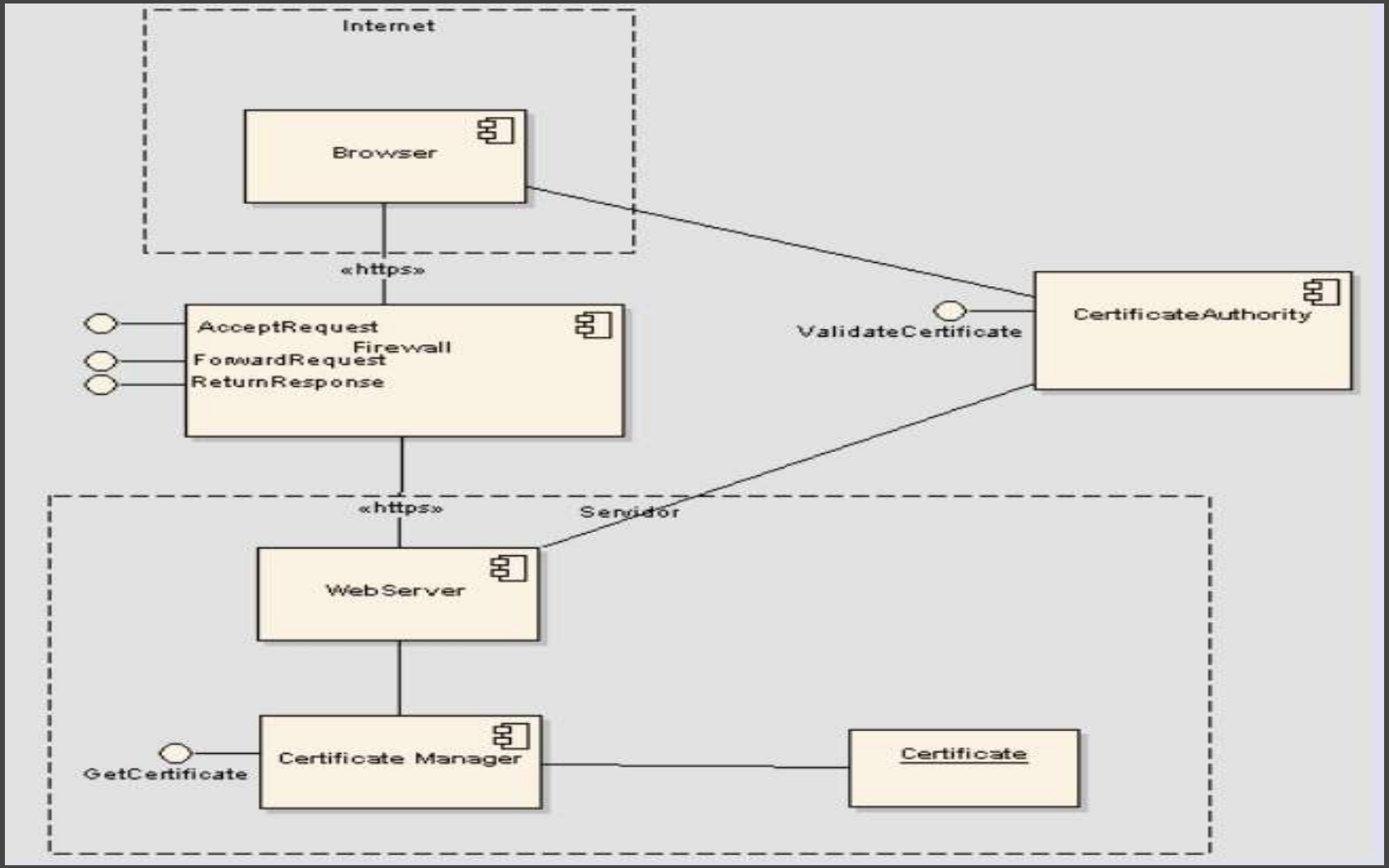


Diagrama de despliegue

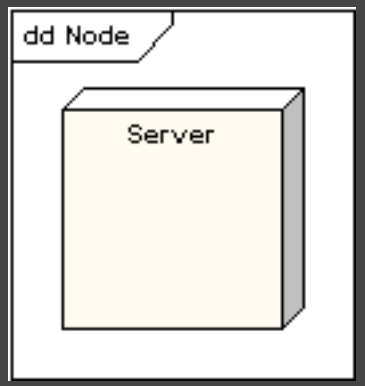
El diagrama de despliegue es usado para el diseño de la arquitectura de un sistema en tiempo de ejecución.

Muestra la configuración hardware del sistema mediante los nodos y la relación de los artefactos software entre los distintos nodos.

Nodo

Un nodo es la representación de un elemento software o hardware en un diagrama de despliegue.

Se representa gráficamente mediante un rectángulo tridimensional con el nombre del nodo en su interior.

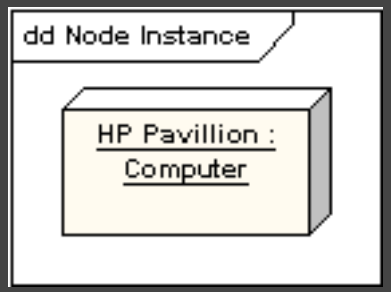


Nodo Server

Instancia de nodo

Una instancia de nodo es la creación de un subnodo del tipo de un nodo existente.

Este se representa de igual manera que el nodo pero con el nombre subrayado seguido de dos puntos y el tipo de nodo.

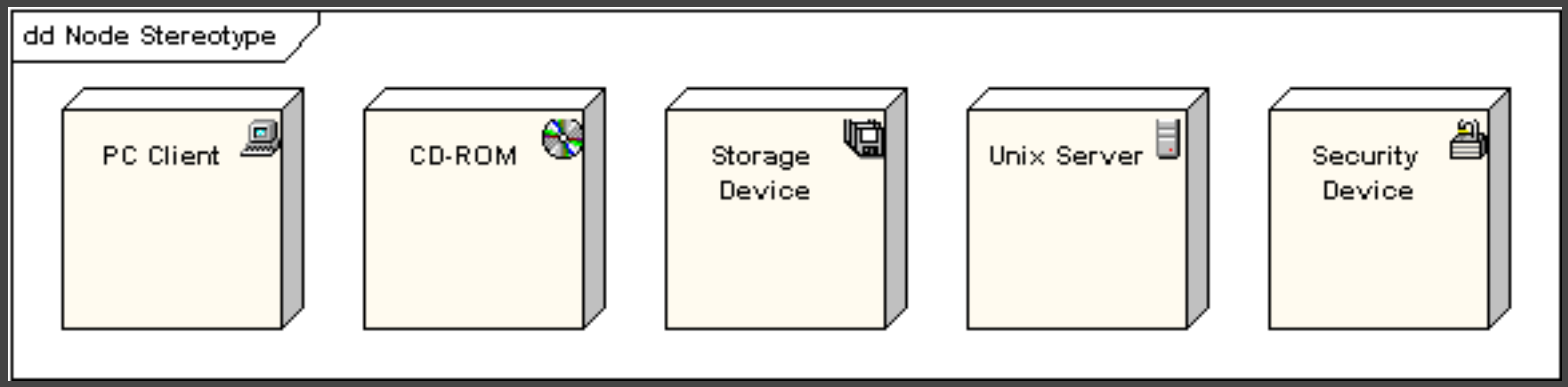


Instancia HP Pavilion del nodo Computer

Estereotipo de nodo

Un estereotipo de nodo es un nodo "genérico" frecuentemente usado.

Estos estereotipos muestran en su esquina superior derecha un icono apropiado a el tipo de nodo.



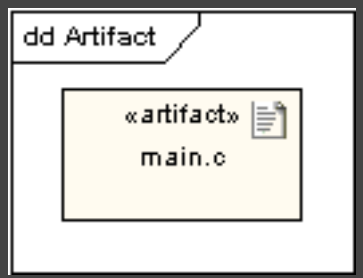
Diferentes estereotipos de nodo

Artefacto

Un artefacto es un elemento que representa un producto obtenido en el proceso de desarrollo software.

Este puede incluir modelos de proceso, archivos fuente, ejecutables, manuales, etc.

Se representa mediante un rectángulo con el nombre del estereotipo, el nombre del documento y un icono identificativo en la esquina superior derecha.

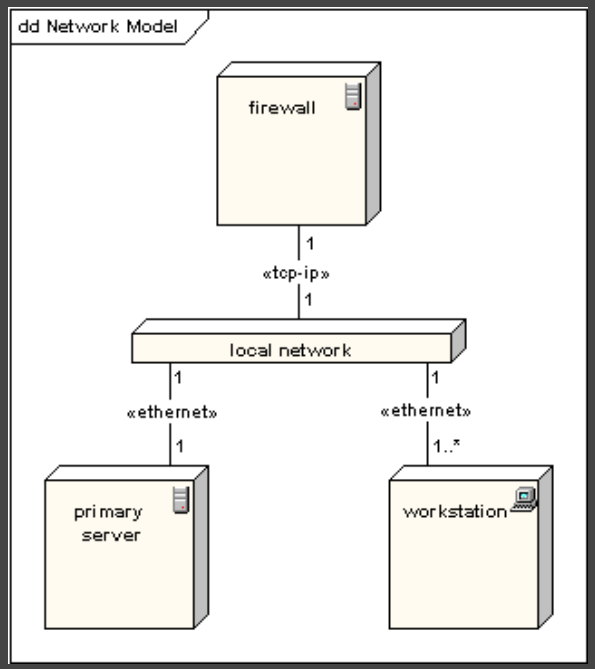


Artefacto del código fuente main.c

Asociación

El elemento asociación en el diagrama de despliegue representa una ruta de comunicación entre los distintos nodos del diagrama.

Se representa mediante una línea con el nombre de la asociación en medio de esta.

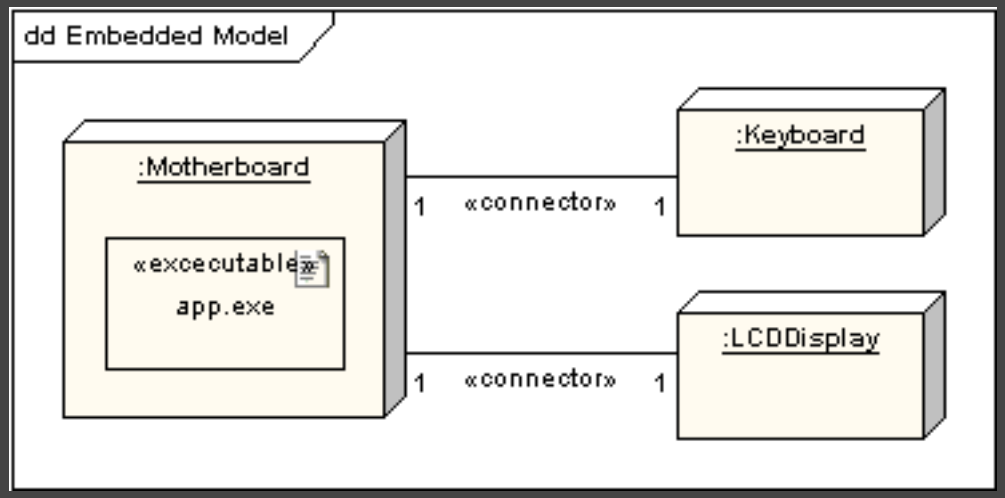


Asociaciónes entre nodos de una red

Nodo contenedor

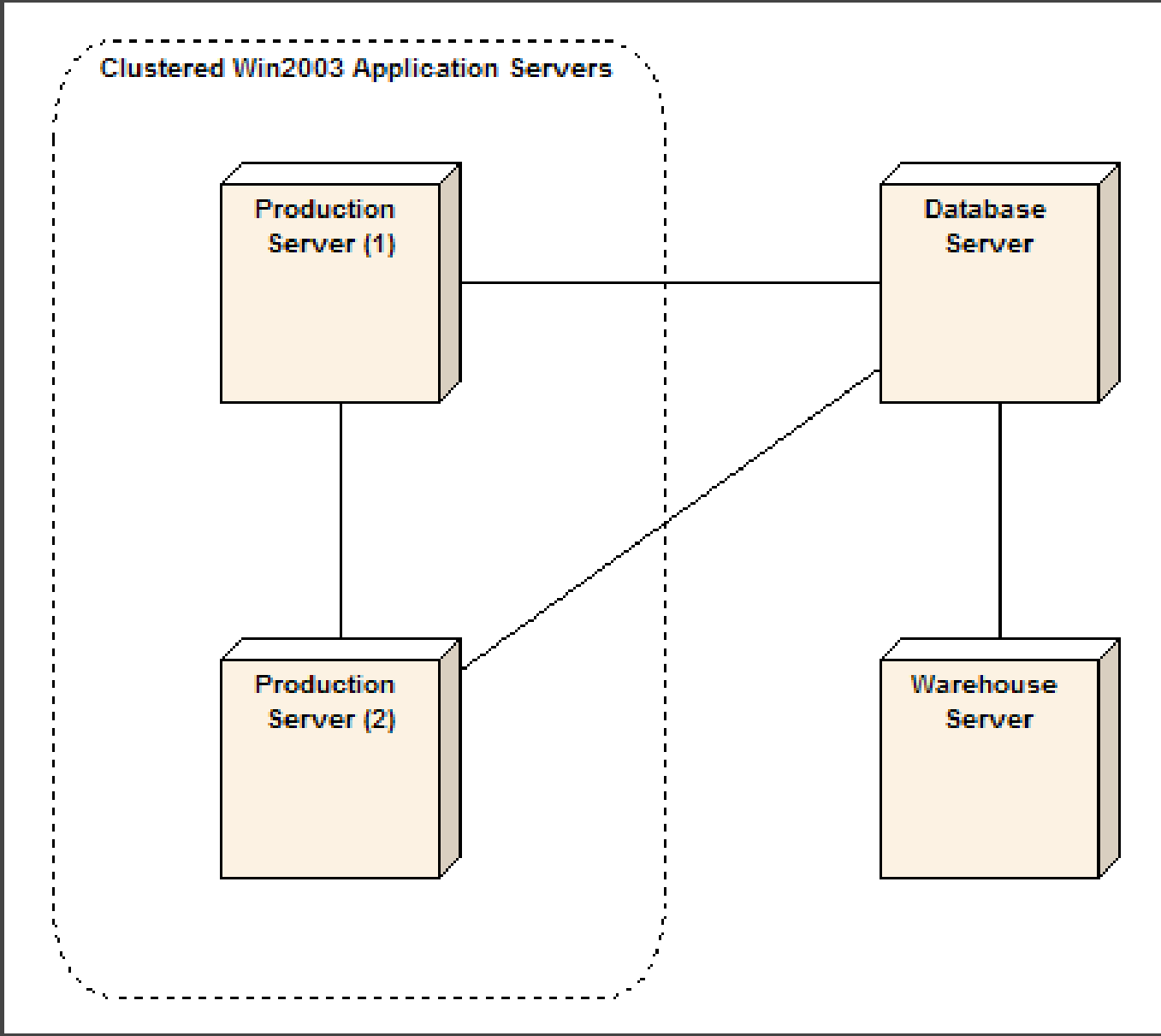
Un nodo puede contener en su interior diferentes elementos como artefactos o componentes.

Para representarlo simplemente se sitúa en el interior del nodo correspondiente el componente o artefacto tal cual.



Artefacto tipo ejecutable app.exe dentro del nodo Motherboard (Placa base)

Ejemplo


Clustered Win2003 Application Servers
Production Server (1)
Database Server
Production Server (2)
Warehouse Server

Ejercicio Propuestp

Realizar el diagrama de despliegue de una aplicación Web Cliente-Servidor

- Cliente: Browser Convencional

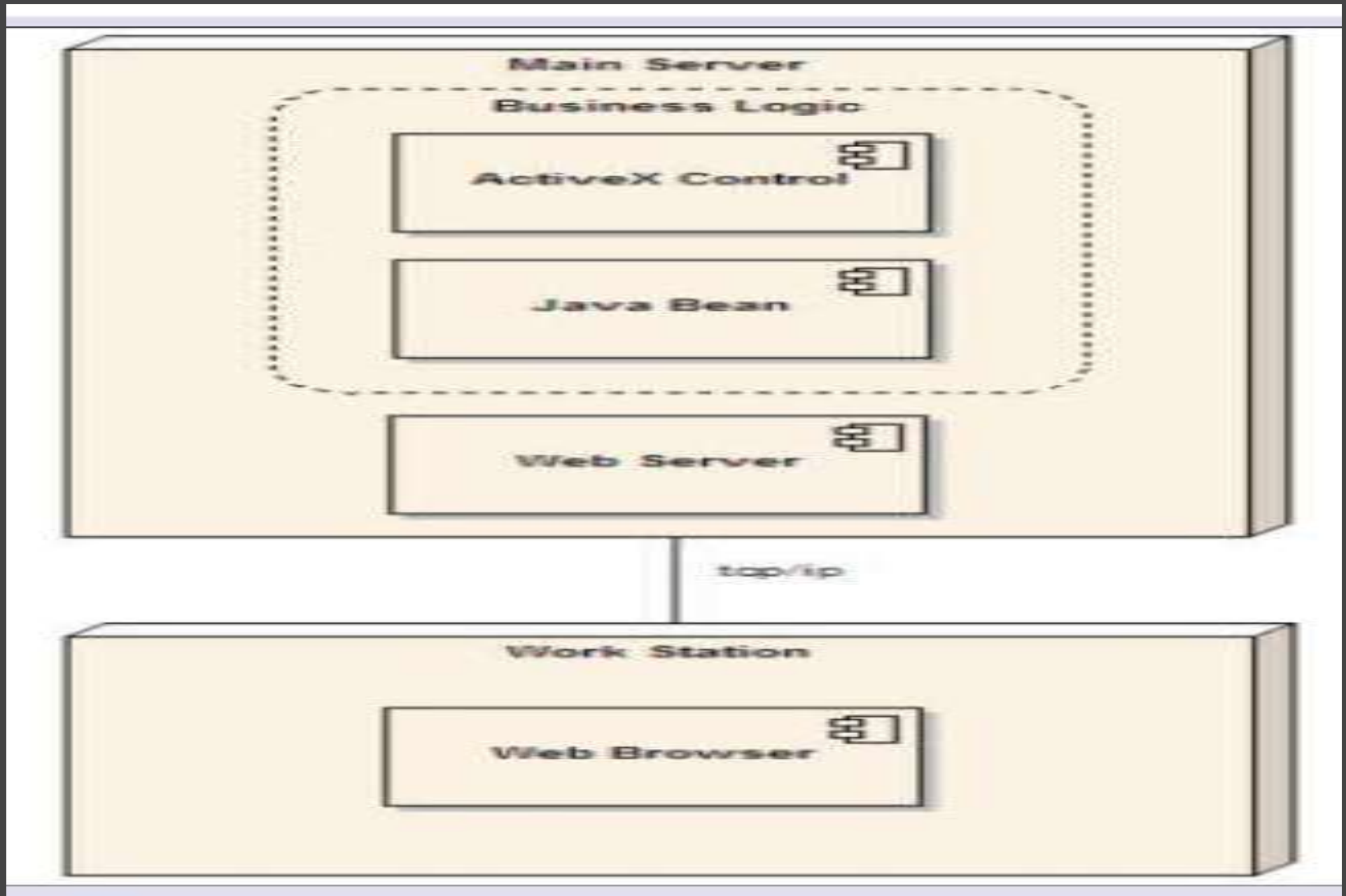
-Servidor:

Web tier: Web Server

Business tier: JavaBeans + Controles ActiveX

-Comunicación: tcp/ip

Solución Ejercicio



Bibliografía

- *http://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado*
- *http://tvdi.det.uvigo.es/~avilas/UML/node37.html*
- *http://docs.kde.org/kde3/es/kdesdk/umbrello/uml-elements.html*
- *http://www.geocities.com/j_ll_fabregas/ADPTI-09-DClase.pdf*
- *http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html*
- *http://sparxsystems.com.es/resources/tutorial/uml2_componentdiagram.html*
- *http://es.wikipedia.org/wiki/Diagrama_de_componentes*

FIN