

INTRODUCCION UML 2

DISEÑO Y CONSTRUCCIÓN DE SOFTWARE



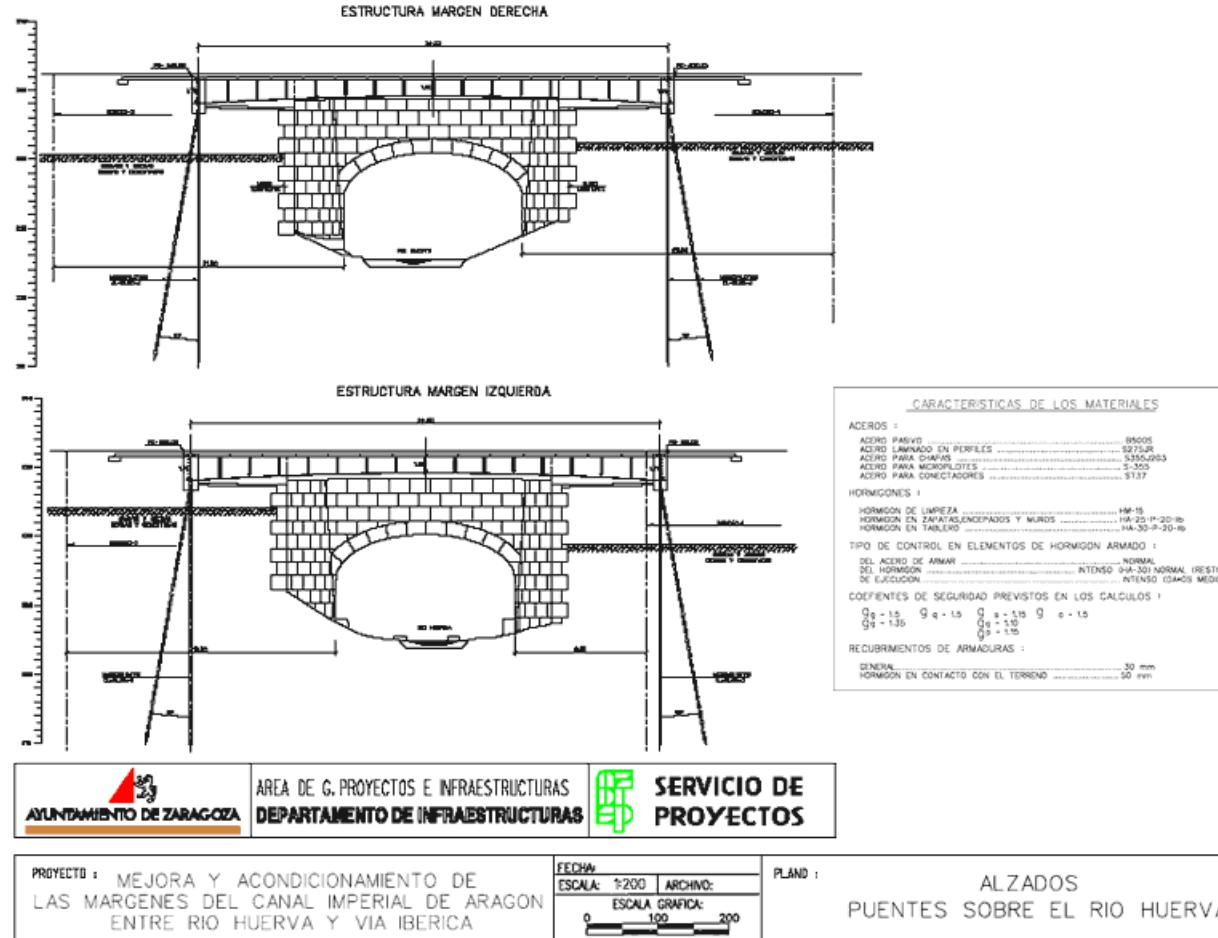
¿Qué es UML?

■ Unified Modeling Language

- ◆ Lenguaje gráfico y formal para el modelado de sistemas
 - Visualizar, especificar, construir y documentar los elementos de un sistema.
- ◆ Lenguaje, **no método**, ligado a USDP, pero sirve para otros métodos
 - Object-oriented Process, Environment, and Notation
- ◆ Orientado a objetos
 - Toma sus conceptos de los lenguajes de programación orientados a objetos
 - Aplicación más extensa

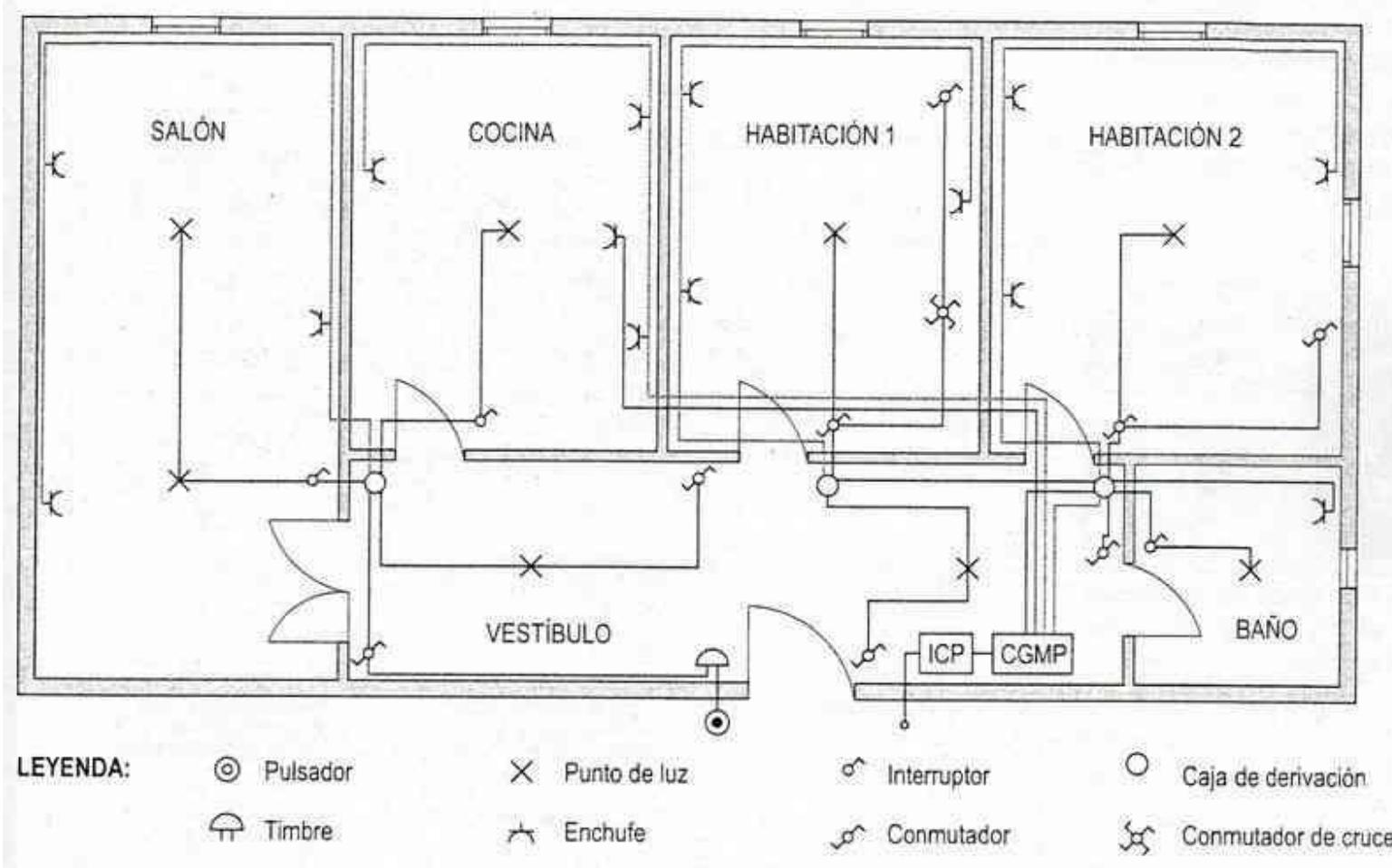
¿Gráfico? Diseños ~ Dibujos

- Representar diseño mediante dibujos es útil
 - ◆ La mayoría de las ingenierías



¿Gráfico? Diseños ~ Dibujos

- Representar diseño mediante dibujos es útil
 - ◆ La mayoría de las ingenierías



¿Gráfico? Diseños ~ Dibujos

- Representar diseño mediante dibujos es útil
 - ◆ La mayoría de las ingenierías



¿Gráfico? Diseños ~ Dibujos

- Ingeniería del Software no es una excepción
 - ◆ Uso de dibujos y modelos para representar conceptos
- ¿Por qué?
 - ◆ Forma de comunicar los modelos
 - Miembros y cualquier persona relacionada con el proyectos
 - Futuras generaciones de desarrolladores
 - ◆ Marco para pensar y analizar
 - Las ideas no están en la cabeza durante meses/años

UNIFIED
MODELING
LANGUAGE



¿Unificado?

■ Diferentes dominios

- ◆ Ciclo de vida de desarrollo
 - Sintaxis visual para modelar desde los requisitos hasta la implementación
- ◆ Dominios de aplicación
 - Desde sistemas en tiempo real a sistema de toma de decisiones
- ◆ Lenguajes y plataformas de implementación
 - Neutro tanto en lenguaje como en plataforma
 - Orientados a objetos → Smalltalk, Java, C#
 - OO híbridos → C++
 - Basados en objetos → Visual Basic

¿Unificado?

■ Diferentes dominios

- ◆ Procesos de desarrollo
 - UP es el proceso de desarrollo preferido
 - Soporta otros procesos de ingeniería del software
- ◆ Propios conceptos internos
 - UML trata de ser coherente y uniforme en su aplicación de un pequeño grupo de conceptos internos

¿Modelado?

■ Arquitectura

- ◆ ¿Tiene sentido poner ladrillos de un edificio sin hacer antes los planos?
- ◆ El modelo, los planos, ayuda a afrontar la complejidad del proyecto.
- ◆ ¿Cuál es el lenguaje adecuado para representar los planos?

■ Desarrollo de software

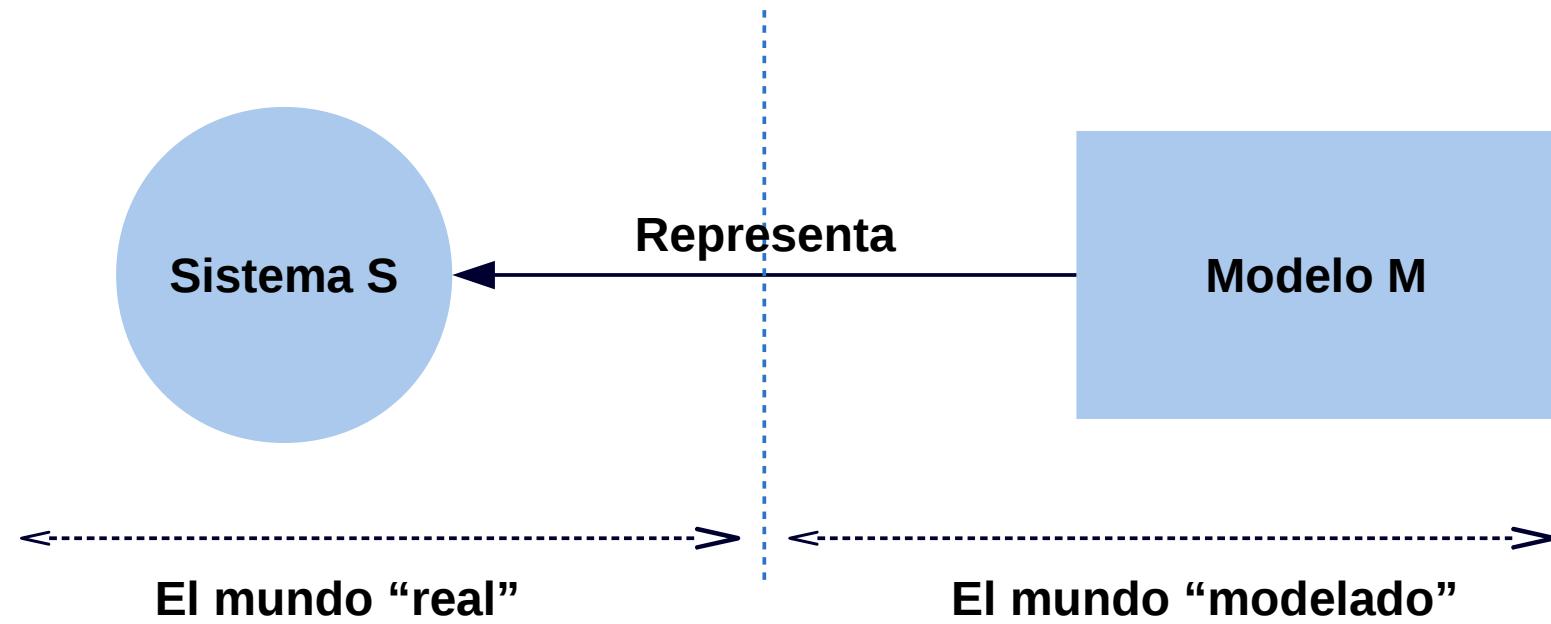
- ◆ ¿Tiene sentido “picar código” sin hacer antes de conocer el problema (análisis) y haber pensado una solución (diseño)?

¿Modelado?

- **Modelo → Representación de un sistema físico**
 - ◆ **Abstracto**
 - Simplifica de la realidad: divide y vencerás
 - ◆ **Comprensible**
 - Expresado de tal forma que se pueda entender fácilmente
 - ◆ **Preciso**
 - Representa fielmente el sistema modelado
 - ◆ **Predictivo**
 - Se puede utilizar para obtener conclusiones correctas
 - ◆ **Barata**
 - Más económico que construir y estudiar el propio sistema₀

¿Modelado?

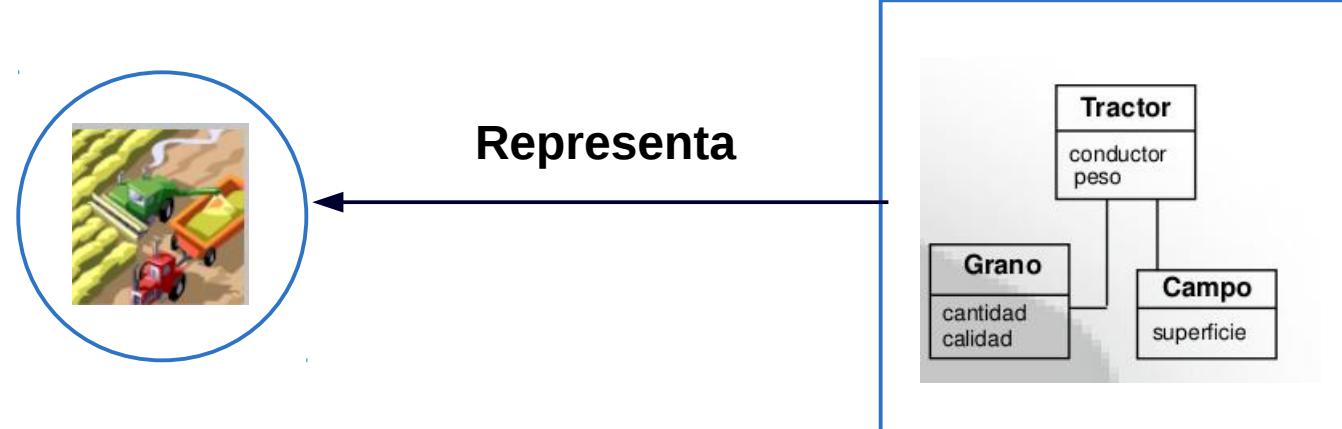
■ Modelo de un sistema



Jean Bézivin, —Model Engineering for Software Modernization||,
The 11th IEEE Working Conference on Reverse Engineering, 2004.

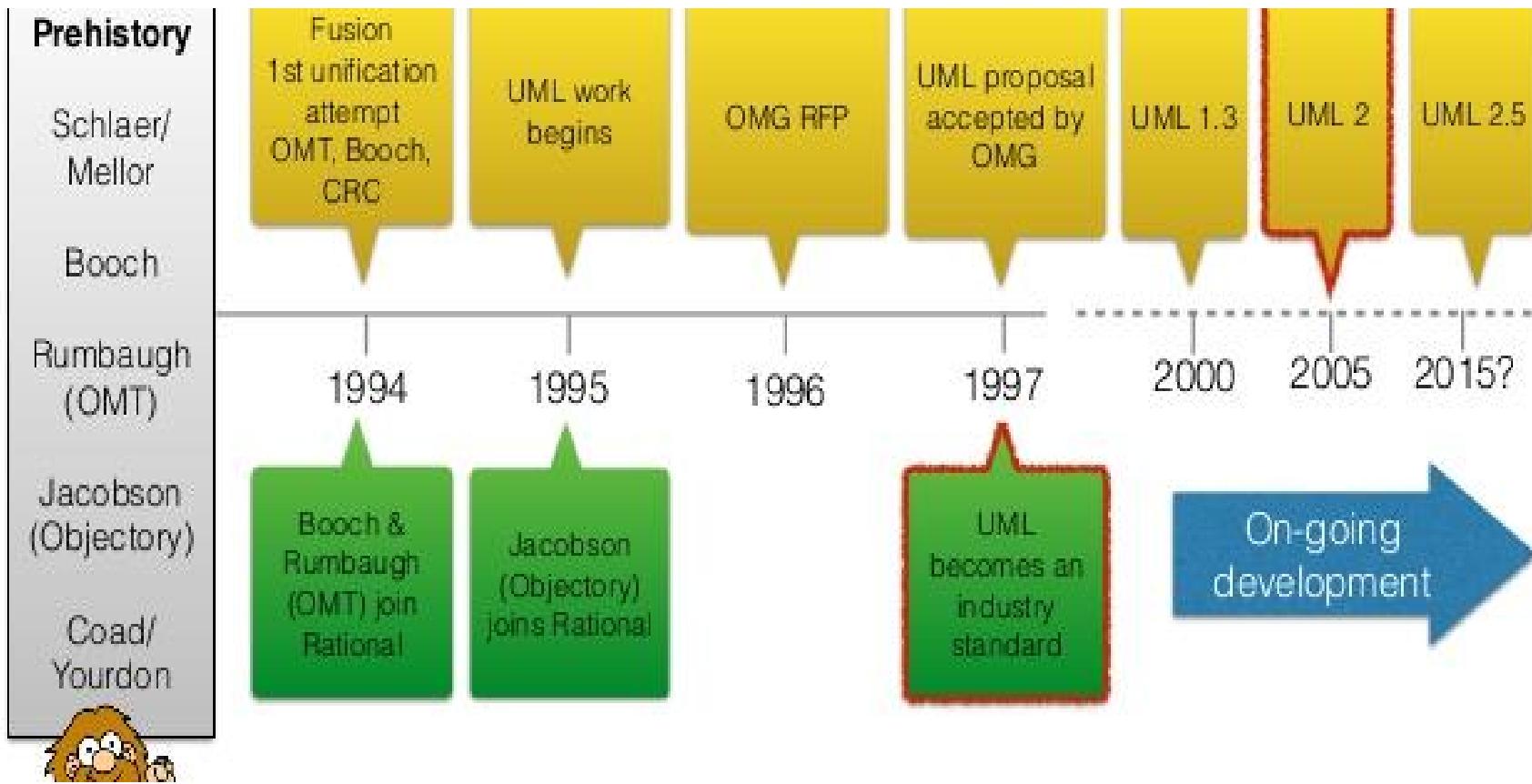
¿Modelado?

■ Modelo de un sistema



Jean Bézivin, —Model Engineering for Software Modernization||,
The 11th IEEE Working Conference on Reverse Engineering, 2004.

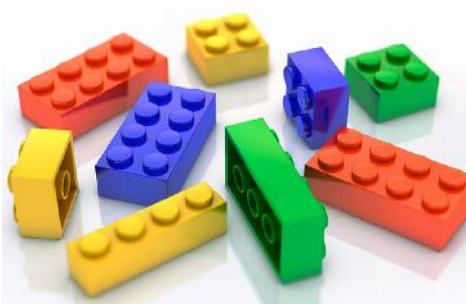
Historia de UML



Objetos y UML

- UML modela sistemas como colecciones de **objetos** que interactúan
 - ◆ Procesos de negocios y otras aplicaciones
- Modelo UML
 - ◆ Estructura estática:
 - ¿Qué objetos son importantes?
 - ¿Cómo se relacionan?
 - ◆ Estructura dinámica
 - Ciclos de vida de los objetos
 - Cómo interactúan entre sí para lograr la funcionalidad

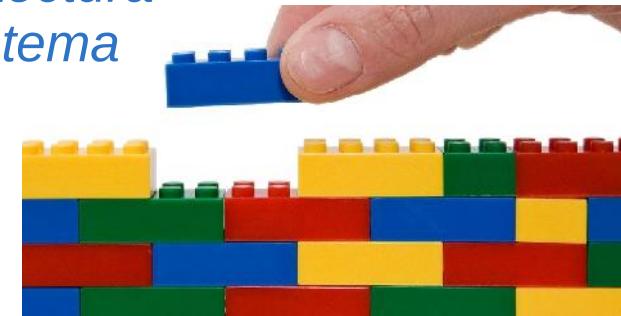
Estructura de UML



Bloques de construcción *Elementos básicos modelado*

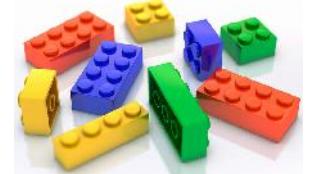


Arquitectura *Visión UML de la arquitectura del sistema*



Mecanismos comunes *Formas comunes de conseguir objetivos específicos*

Bloques de construcción



■ Elementos

- ◆ Elementos básicos del modelado
 - Los elementos propios del modelo

■ Relaciones

- ◆ Unen a los elementos entre sí
 - ¿Cómo se relacionan dos o más elementos?

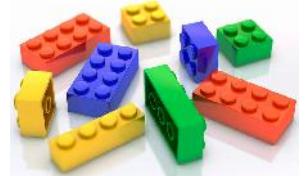
■ Diagramas

- ◆ Son vistas de los modelos UML
 - Muestran colecciones de elementos que “cuentan una historia” sobre el sistema
 - Forma de visualizar qué hará el sistema o cómo lo hará

Bloques de construcción (Elementos)

■ Cuatro tipos

- ◆ Elementos estructurales
 - Partes estáticas de un modelo
 - Representan cosas que son conceptuales o materiales
 - **Clase, interfaz, colaboración, caso de uso, clase activa, componente, nodo**
- ◆ Elementos de comportamiento
 - Partes dinámicas de un modelo
 - Representan comportamiento en el tiempo y el espacio
 - **Interacción, actividades, máquinas de estado**



Bloques de construcción (Elementos)

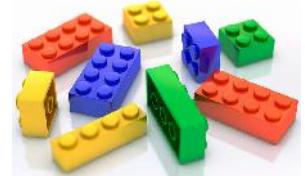
■ Cuatro tipos

- ◆ Elementos de agrupación

- Parte organizativa de los modelos UML
- Cajas en que puede descomponerse un modelo
- **Paquete**

- ◆ Elementos de anotación

- Parte aclarativa de los modelos
- Se anexa a los modelos para capturar información ad hoc

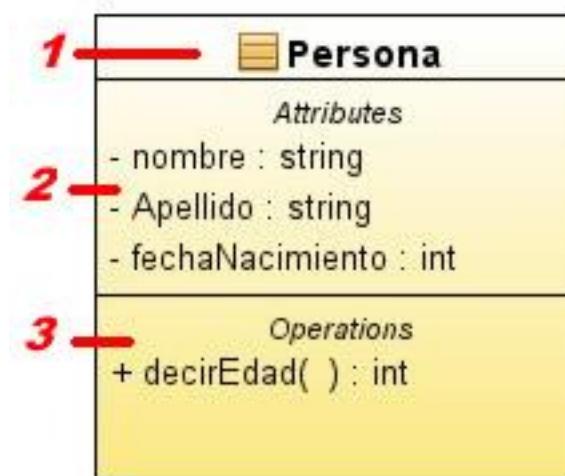
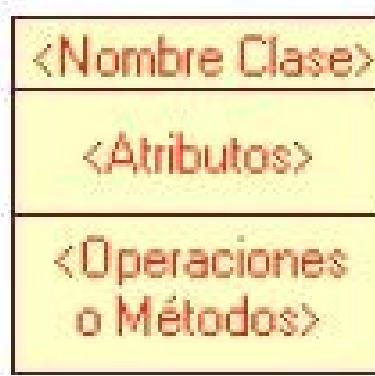
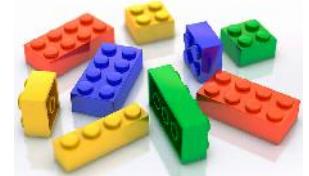


Bloques de construcción (Elementos)

■ Elementos estructurales

◆ Clase

- Descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica
- Implementa una o más interfaces

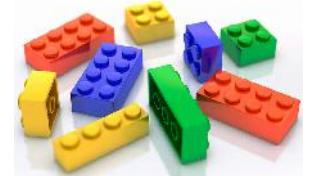


Bloques de construcción (Elementos)

■ Elementos estructurales

◆ Interfaz

- Colección de operaciones que especifican un servicio de una clase o componente
 - No la implementación
- Representa el comportamiento completo de una clase o componente.

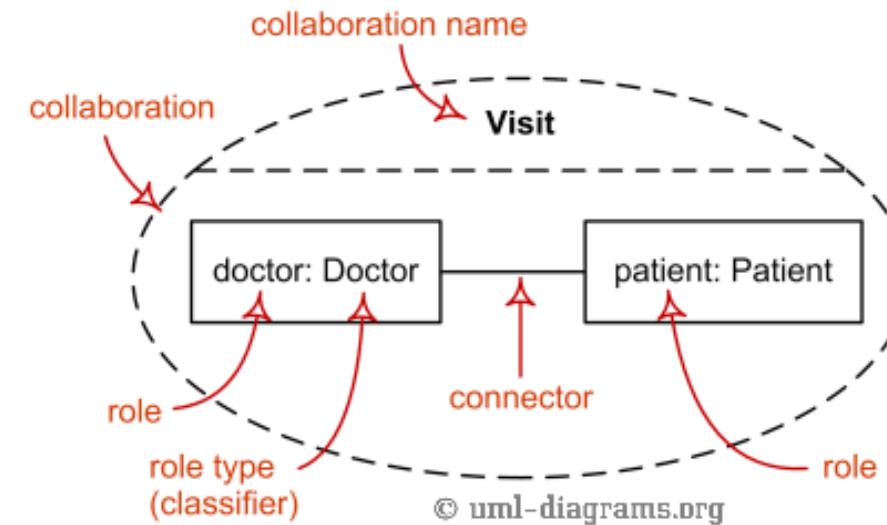
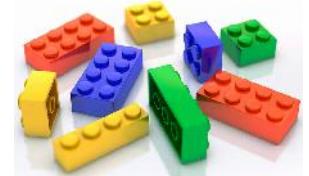


Bloques de construcción (Elementos)

■ Elementos estructurales

◆ Colaboración

- Es una sociedad de clases, interfaces y otros elementos que cooperan para dar un comportamiento mayor que la suma de los comportamientos de sus elementos.



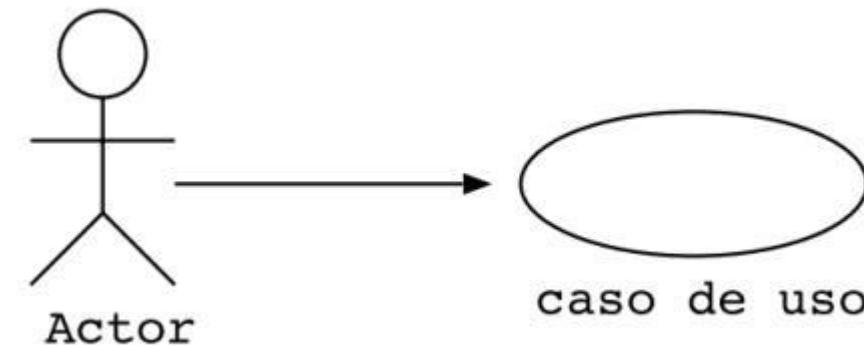
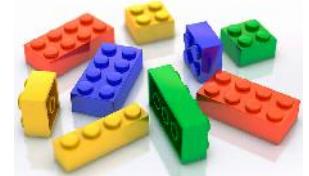
© uml-diagrams.org

Bloques de construcción (Elementos)

■ Elementos estructurales

◆ Caso de uso

- Descripción de una secuencia de acciones que un sistema ejecuta y que produce un resultado de interés para un usuario en particular.
- Se usa para estructurar el comportamiento en un modelo.

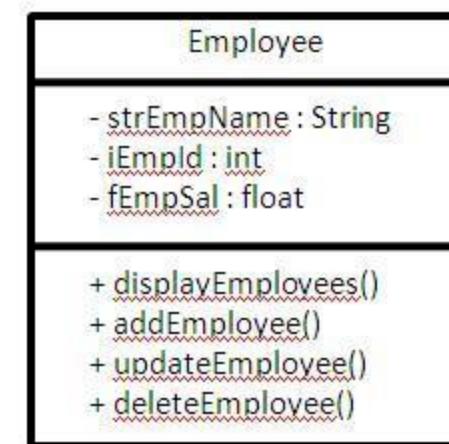
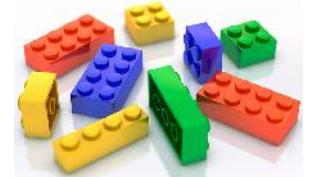


Bloques de construcción (Elementos)

■ Elementos estructurales

◆ Clase activa

- Objeto que tienen uno o más procesos o hilos de ejecución y por lo tanto pueden dar origen a actividades de control.
- Iguales a las clases excepto en que sus objetos representan a elementos cuyo comportamiento es concurrente con otros elementos.

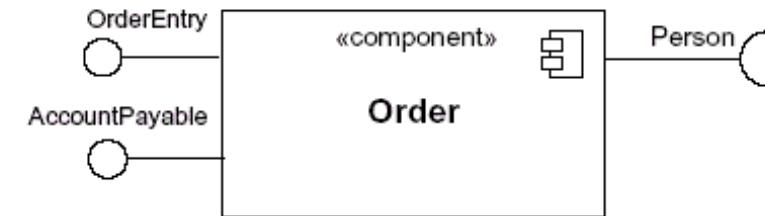
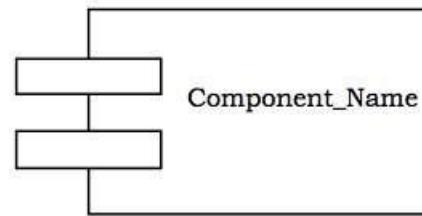
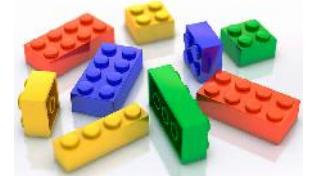


Bloques de construcción (Elementos)

■ Elementos estructurales

◆ Componente

- Parte física y reemplazable de un sistema que se ajusta a, y proporciona la realización, de un conjunto de interfaces
- Representa típicamente el empaquetamiento físico de diferentes elementos lógicos, como clases, interfaces y colaboraciones.

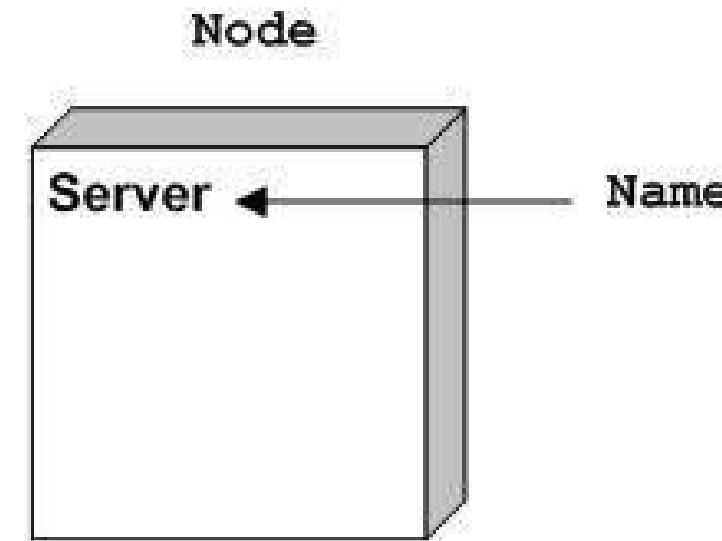
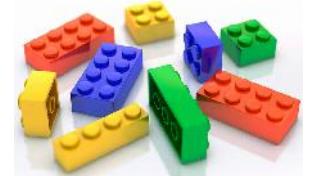


Bloques de construcción (Elementos)

■ Elementos estructurales

◆ Nodo

- Es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional que dispone de memoria y con frecuencia capacidad de procesamiento.

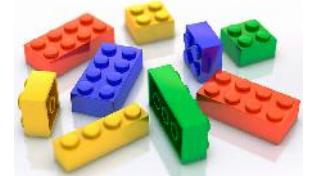


Bloques de construcción (Elementos)

■ Elementos comportamiento

◆ Interacción

- Comportamiento que comprende un conjunto de mensajes intercambiables entre un conjunto de objetos, dentro de un contexto particular para alcanzar un propósito específico

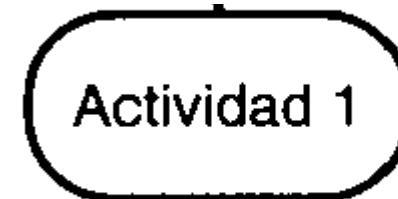


dibujar()



◆ Actividad

- El estado en el que se exhibe algún comportamiento

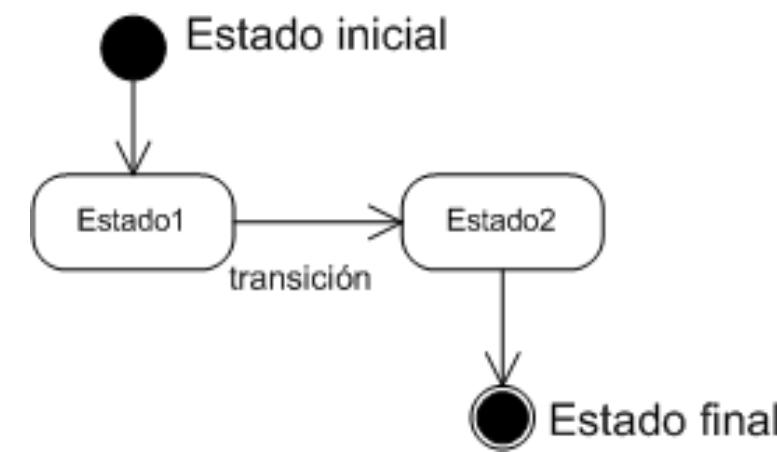
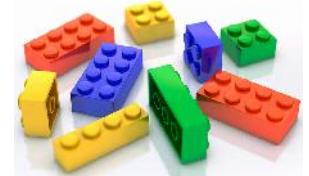


Bloques de construcción (Elementos)

■ Elementos comportamiento

◆ Maquinas de estado

- Comportamiento que especifica la secuencia de estados por las que pasa un objeto durante su vida en respuesta a eventos, junto con sus reacciones a estos eventos

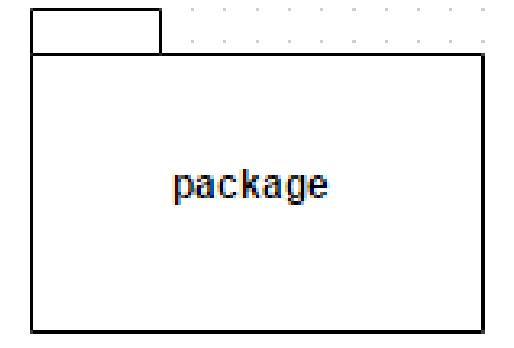
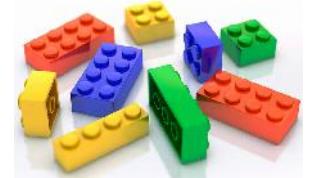


Bloques de construcción (Elementos)

■ Elementos de agrupación

◆ Paquete

- Mecanismo de propósito general para organizar elementos (estructurales, elementos de comportamiento u otros paquetes)

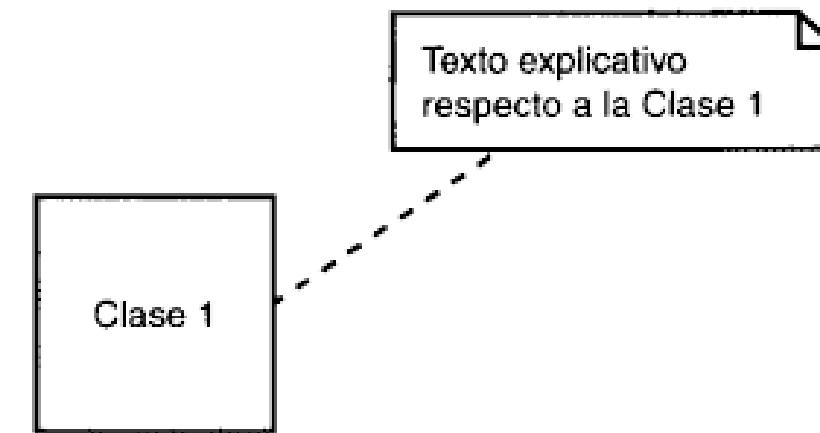
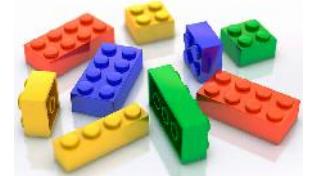


Bloques de construcción (Elementos)

■ Elementos de anotación

◆ Nota

- Parte explicativa de los modelos UML
- Comentarios que se pueden aplicar para describir, clarificar y hacer observaciones sobre cualquier otro elemento del modelo

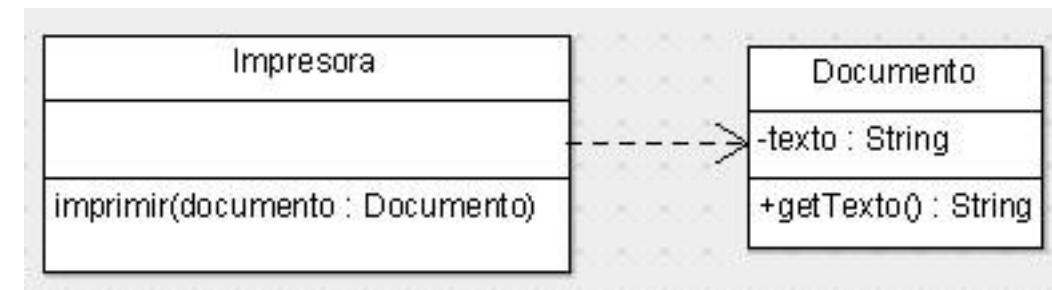
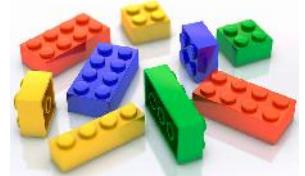


Bloques de construcción (Relaciones)

■ Tipos de relación

◆ Dependencia

- El elemento origen depende del elemento destino y se puede ver afectado por cambios en este

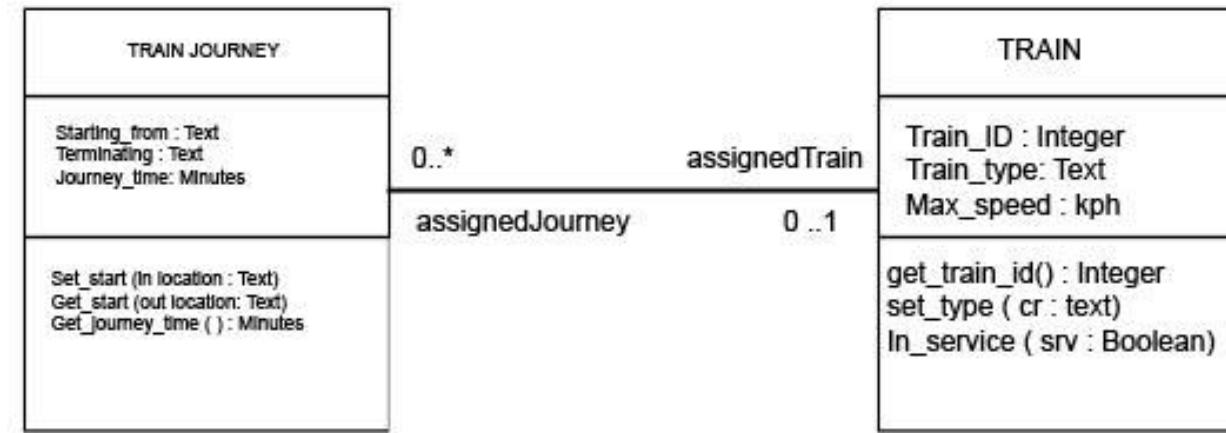
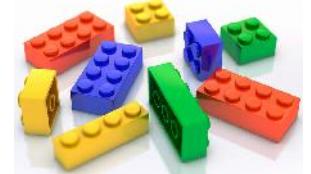


Bloques de construcción (Relaciones)

■ Tipos de relación

◆ Asociación

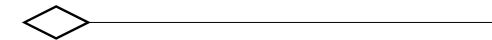
- Relación estructural que describe un conjunto de enlaces, los cuales son conexiones entre objetos
- Puede estar dirigida y contener otros elementos como multiplicidad y los nombres de roles



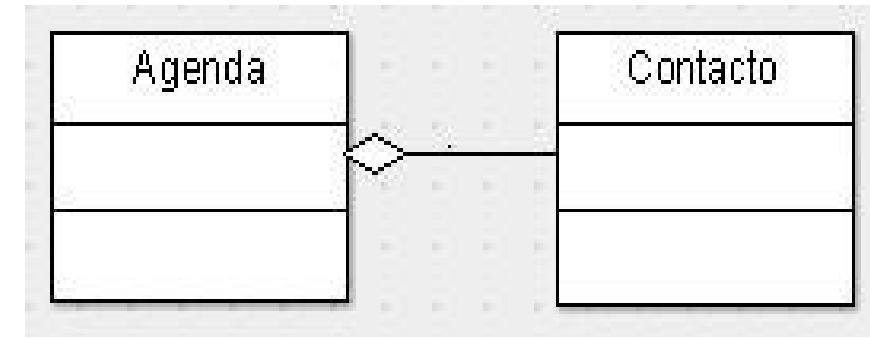
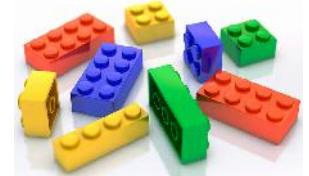
Bloques de construcción (Relaciones)

■ Tipos de relación

◆ Agregación



- El elemento origen es una parte del elemento destino

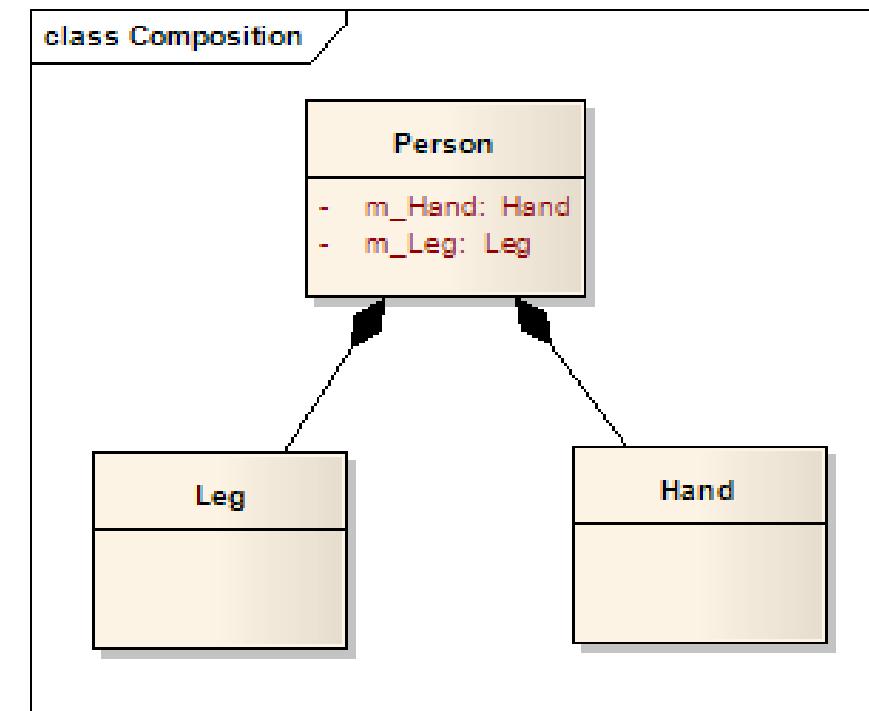


Bloques de construcción (Relaciones)

■ Tipos de relación

◆ Composición

- Agregación más restringida

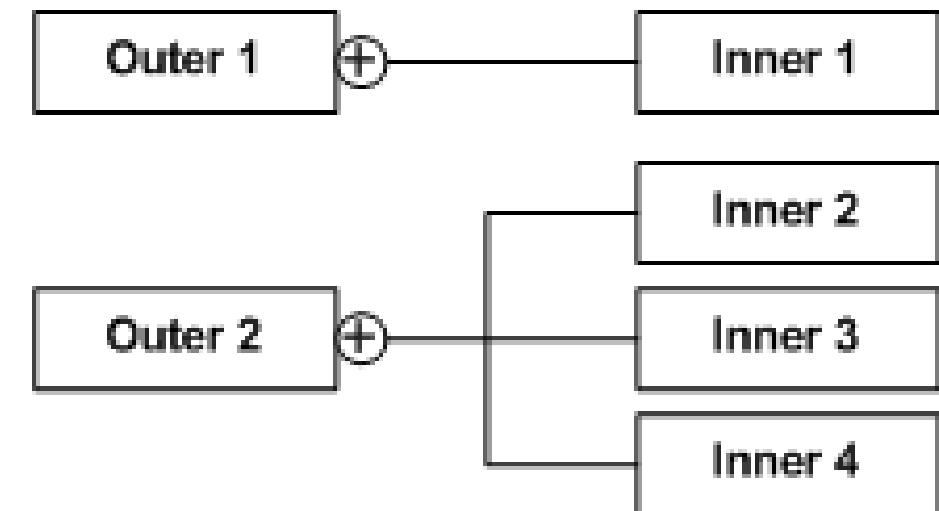
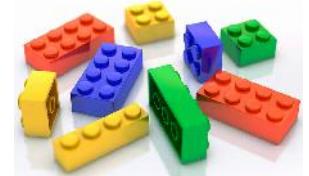


Bloques de construcción (Relaciones)

■ Tipos de relación

◆ Contención

- El elemento origen contiene el elemento destino

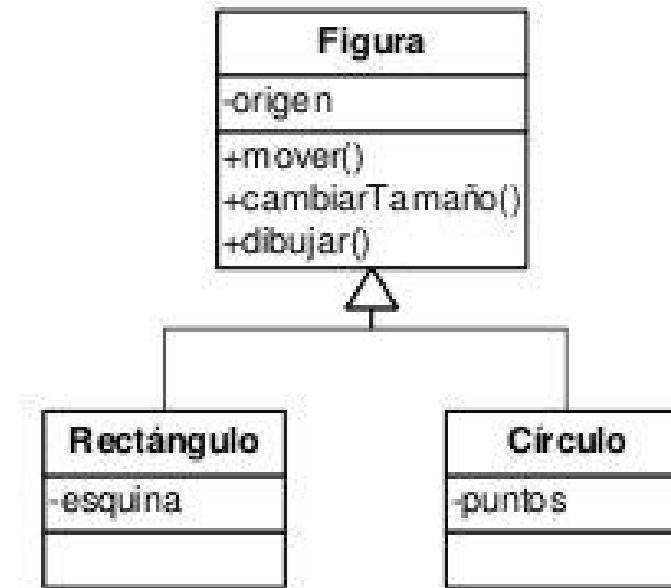
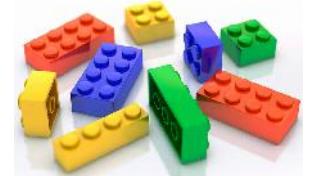


Bloques de construcción (Relaciones)

■ Tipos de relación

◆ Generalización

- El elemento origen es una especialización del elemento destino más general y se puede sustituir por éste.

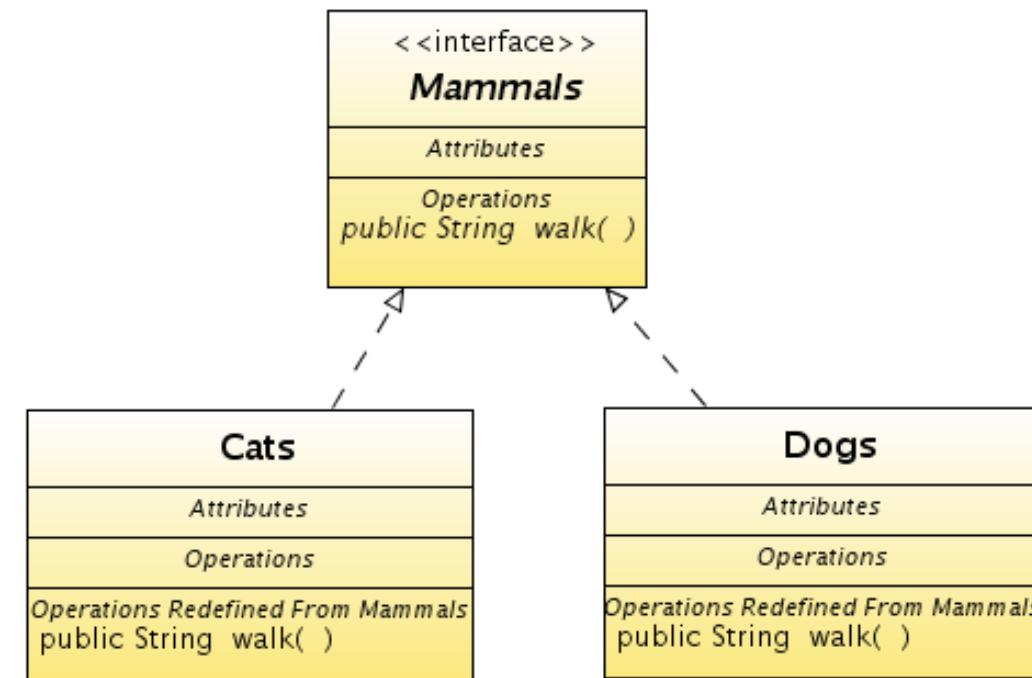
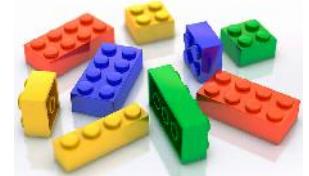


Bloques de construcción (Relaciones)

■ Tipos de relación

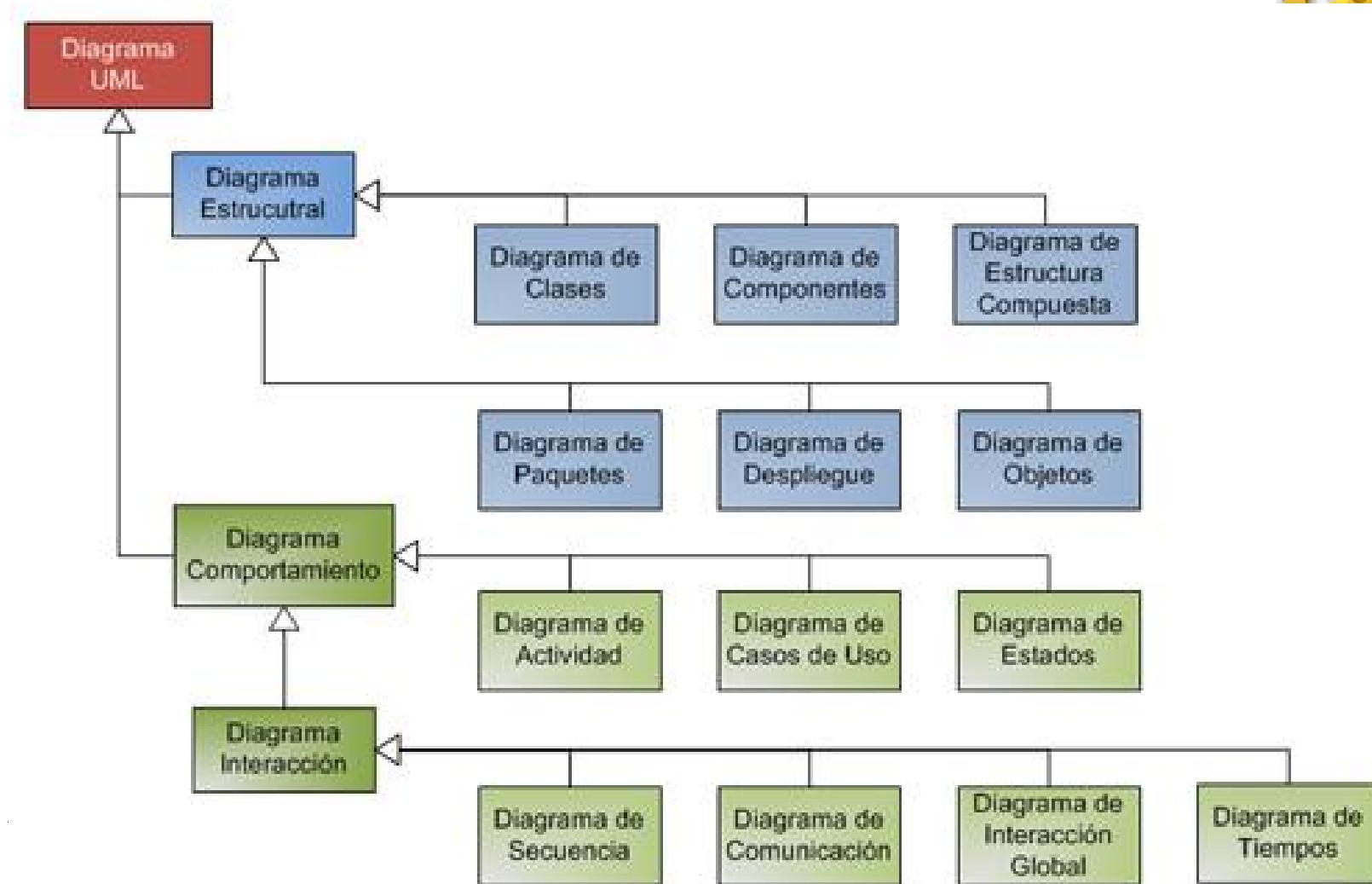
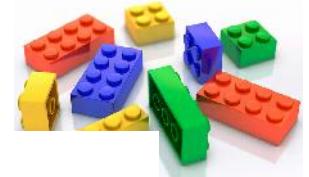
◆ Realización

- El elemento origen garantiza llevar a cabo el contrato especificado por el elemento destino.



Bloques de construcción (Diagramas)

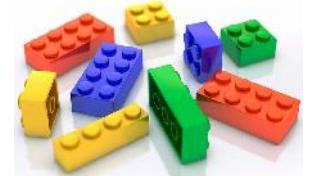
■ Tipos de diagramas



Bloques de construcción (Diagramas)

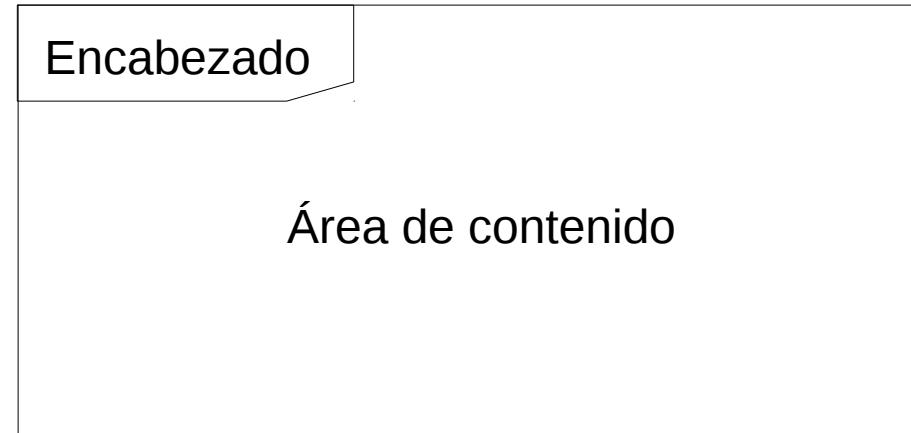
■ Tipos de diagramas

- ◆ **Diagrama estructura (modelo estático)**
 - Modelan la estructura estática del sistema
 - Captura los elementos y la relaciones estructurales entre ellos
- ◆ **Diagrama comportamiento (modelo dinámico)**
 - Modelan la estructura dinámica del sistema
 - Captura como los elementos interactúan para generar el comportamiento requerido del sistema
- ◆ **Cada diagrama da un vista diferente del modelo**
 - Modelo → conjunto de elementos y relaciones

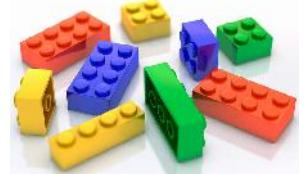


Bloques de construcción (Diagramas)

■ Sintaxis



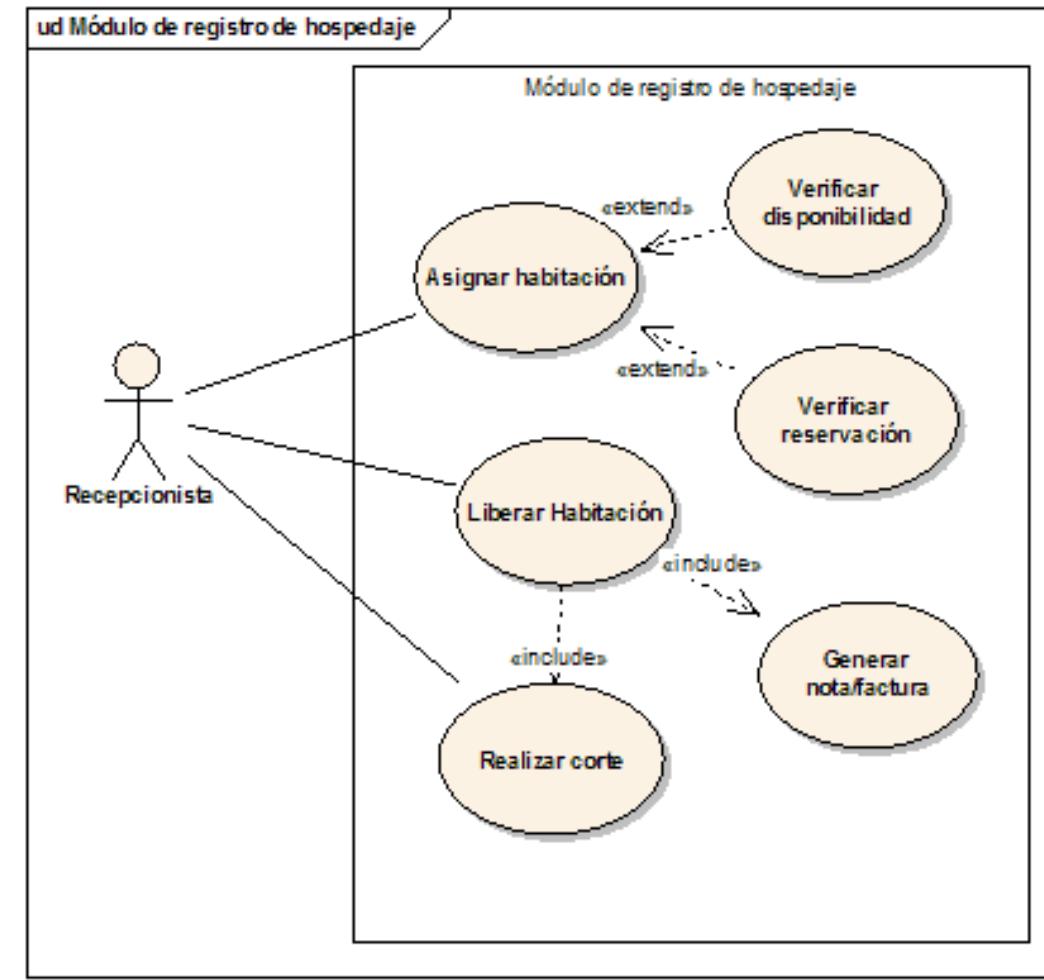
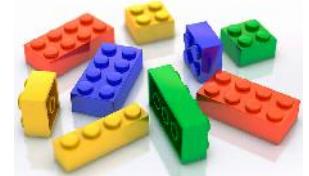
Sintaxis encabezado:
<tipo><nombre><parametros>



- ◆ **Encabezado**
 - Especifica el **tipo** de diagrama, su **nombre** y cualquier información (**parametros**) necesaria
- ◆ **Área**
 - Contiene el diagrama UML

Bloques de construcción (Diagramas)

■ Sintaxis



Bloques de construcción (Diagramas)

Sintaxis

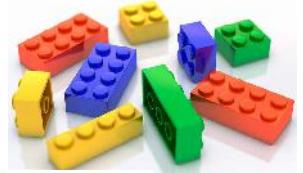
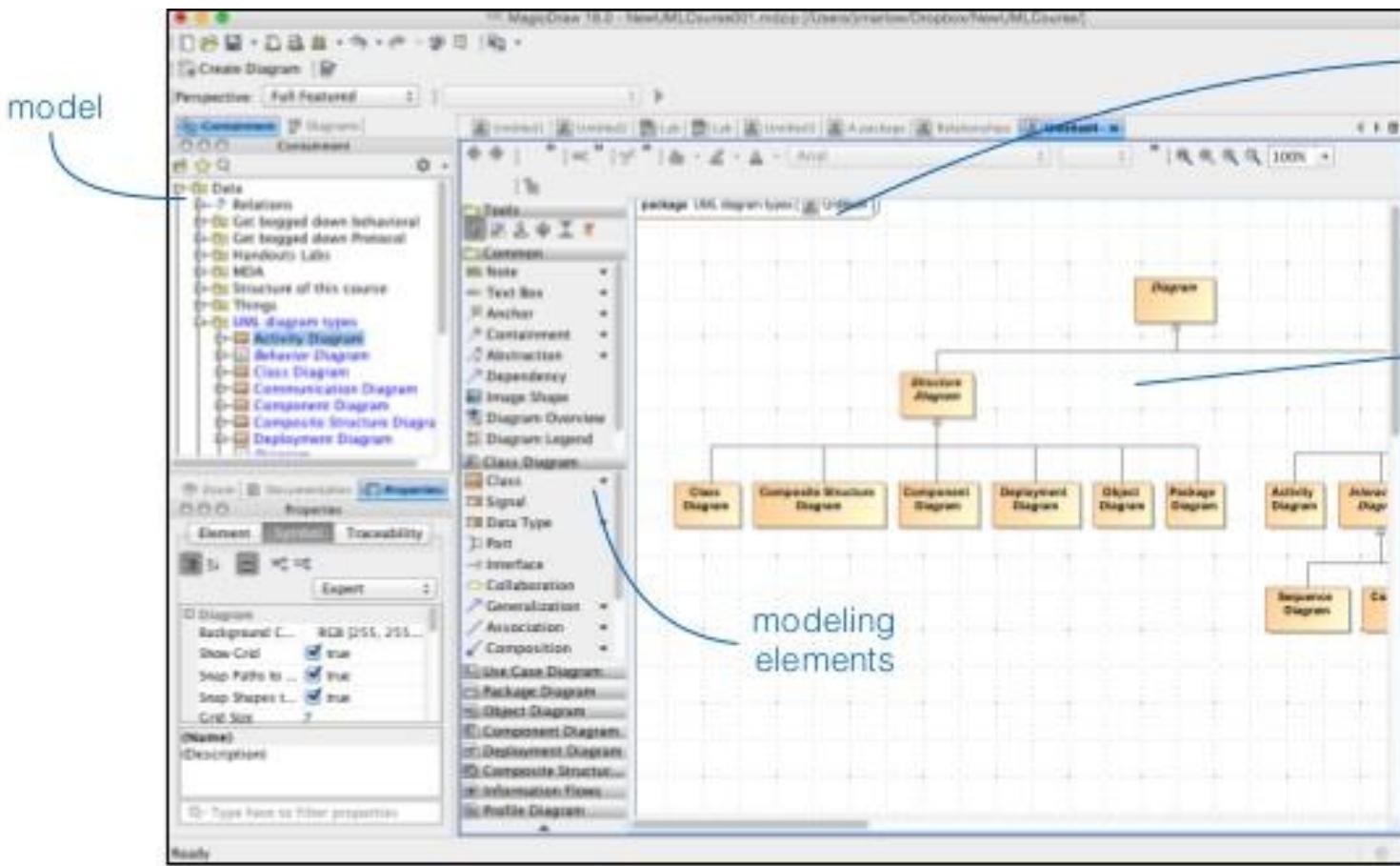


diagram frame

diagram

modeling elements



UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design
(Addison-Wesley Object Technology Series)

Mecanismos comunes

■ ¿Qué son?

- ◆ Estrategias para acercarse al modelado de objetos
- ◆ Se aplican de forma repetida y coherente en diferentes contextos en todo UML

■ Tipos

- ◆ Especificaciones
- ◆ Adornos
- ◆ Divisiones comunes
- ◆ Mecanismos de extensión



Mecanismos comunes (Especificaciones)

■ Dos dimensiones en los modelos UML

- ◆ Dimensión gráfica
 - Visualización del modelo usando diagramas e iconos
 - Que representa un elemento de modelado
- ◆ Dimensión textual
 - Especificación de los diferentes elementos del modelo
 - Descripciones textuales de la semántica de un elemento
 - Forma el plano posterior semántico del modelo
 - Le da significado
 - Los diagramas son vistas o proyecciones de este plano



Mecanismos comunes (Especificaciones)

- Dos dimensiones en los modelos UML



```

Customer
-firstName : String
lastName : String

+Customer(f : String, l : String)
+getFirstName() : String
+getLastName() : String
+getAccount() : Account
+setAccount(acct : Account)
  
```

Dimensión gráfica

Name	Type	Visibility	Is Abstract
Customer		public	false

Attribute

Dimensión textual

Mecanismos comunes (Adornos)



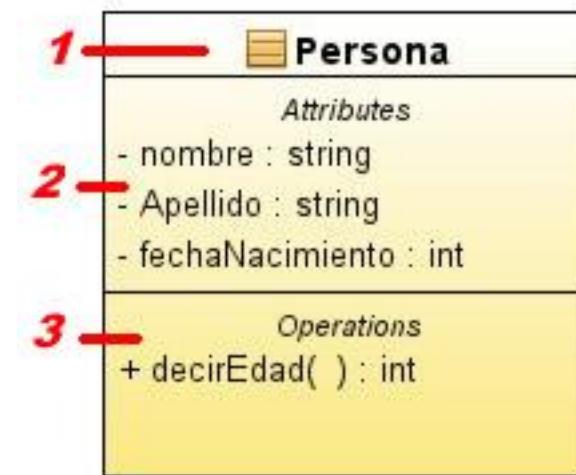
- Todo elemento del modelado
 - ◆ Un símbolo sencillo
 - ◆ Añadir un conjunto de **adornos**
 - Hacer visible aspectos de la especificación del elemento
- ¿Cuántos adornos?
 - ◆ Aquellos que resalten características importantes
 - ◆ Aumenta la claridad global y legibilidad del diagrama

Mecanismos comunes (Adornos)

- Ejemplo para una clase



Elemento sin adornos

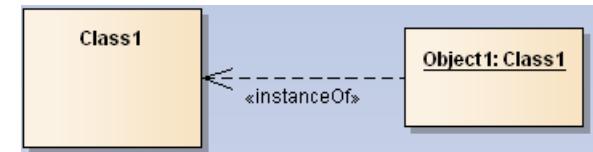


Elemento con adornos

Mecanismos comunes (Divisiones comunes)



- ¿Qué son?
 - ◆ Describe formas particulares de pensar sobre el mundo
- Clasificador e instancia
 - ◆ Clasificador → noción abstracta de un tipo
 - UML y el procesos unificado
 - Cuenta corriente
 - ◆ Instancia → elemento concreto de una abstracción
 - Libro UML 2 / UML 2 and the Unified Process
 - Tu cuenta corriente (un saldo concreto)



Mecanismos comunes (Divisiones comunes)

■ Interfaz e implementación

◆ Idea

- Separar qué hace algo (interfaz) de cómo lo hace (implementación).

➤ *Ejemplo → conducir un coche*

◆ Interfaz

- Define un contrato

➤ Botones de sonido de un móvil

◆ Implementación

- Realización concreta de ese contrato

➤ Mecanismo para subir y bajar el volumen del móvil



Mecanismos comunes (Mecanismos extensibilidad)



■ Idea

- ◆ Mecanismos que permiten personalizar UML

■ Tipos

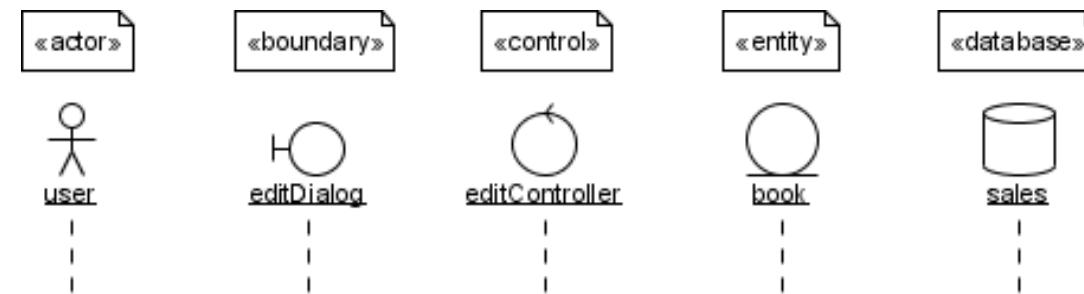
- ◆ **Restricciones** (*{restricción}*)
 - Extensión de la semántica de un elemento UML
 - Permite añadir nuevas reglas o modificar las existentes
- ◆ **Valores etiquetados** (*{label1=valor1, label2=valor2 }*)
 - Extensión de las propiedades de un elemento UML
 - Permite la creación de una nueva información en la especificación de ese elemento

Mecanismos comunes (Mecanismos extensibilidad)

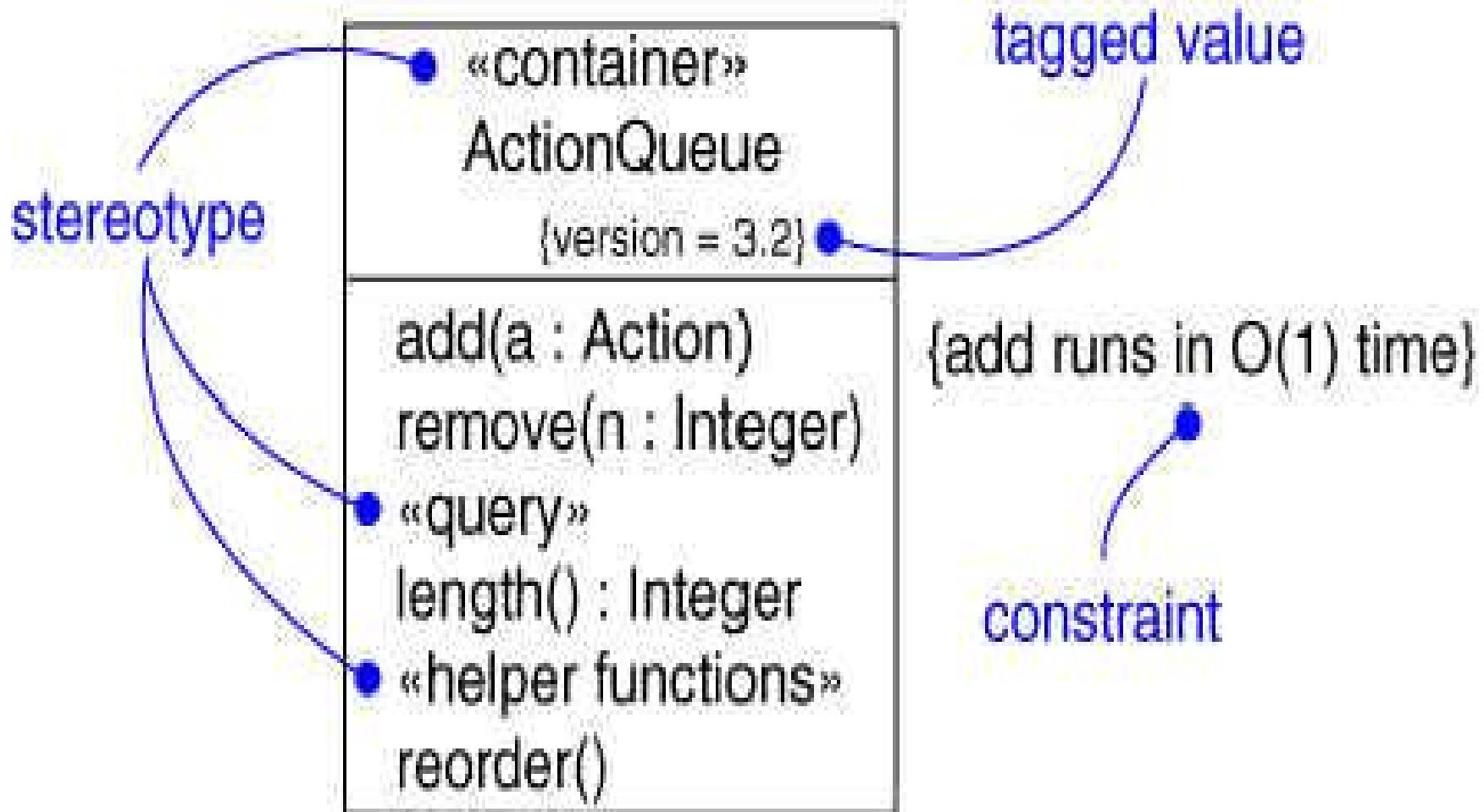
■ Tipos

◆ Estereotipos

- Extensión del vocabulario de UML
- Permite la creación de nuevos bloques de construcción basándose en uno existente
 - Añadiendo conjunto de restricciones y valores etiquetados
- Tiene su propio símbolo y propiedades
- Son específicos para un problema particular
- *<< estereotipo >> nuevo elemento*



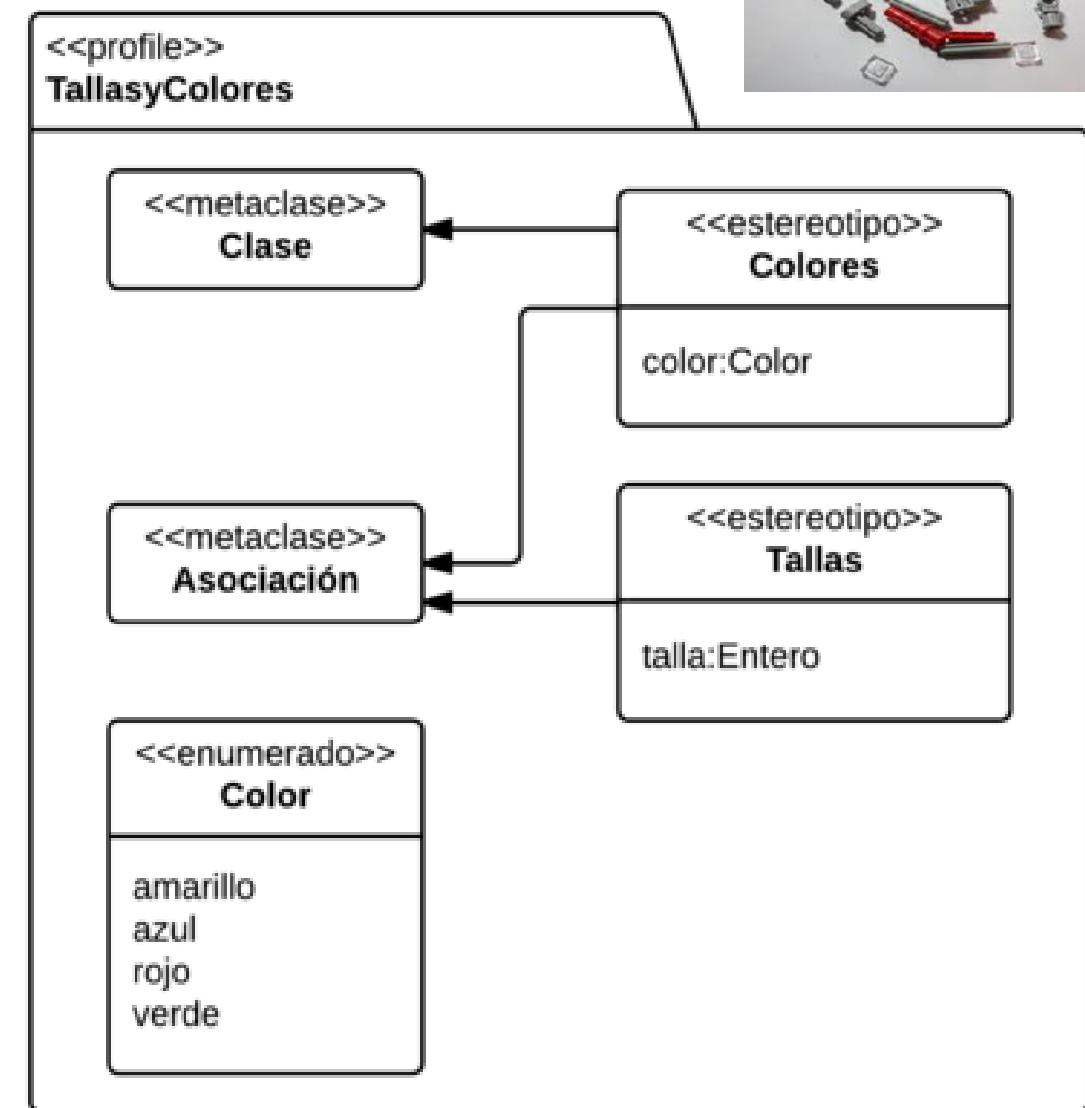
Mecanismos comunes (Mecanismos extensibilidad)



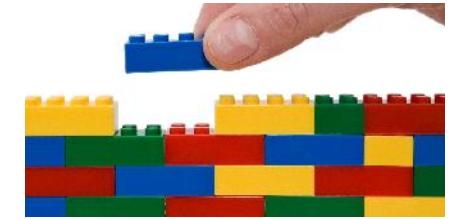
Mecanismos comunes (Mecanismos extensibilidad)

■ Perfiles UML

- ◆ Colección de estereotipos, valores etiquetados y restricciones
- ◆ Se usan para personalizar UML para una finalidad específica



Arquitectura



■ Definiciones

- ◆ *UML Reference Manual*

- Estructura organizacional de un sistema, incluida su descomposición en partes, su conectividad, interacción, mecanismos y los principios directores de un sistema software

- ◆ *IEEE*

- El concepto de más alto nivel de un sistema

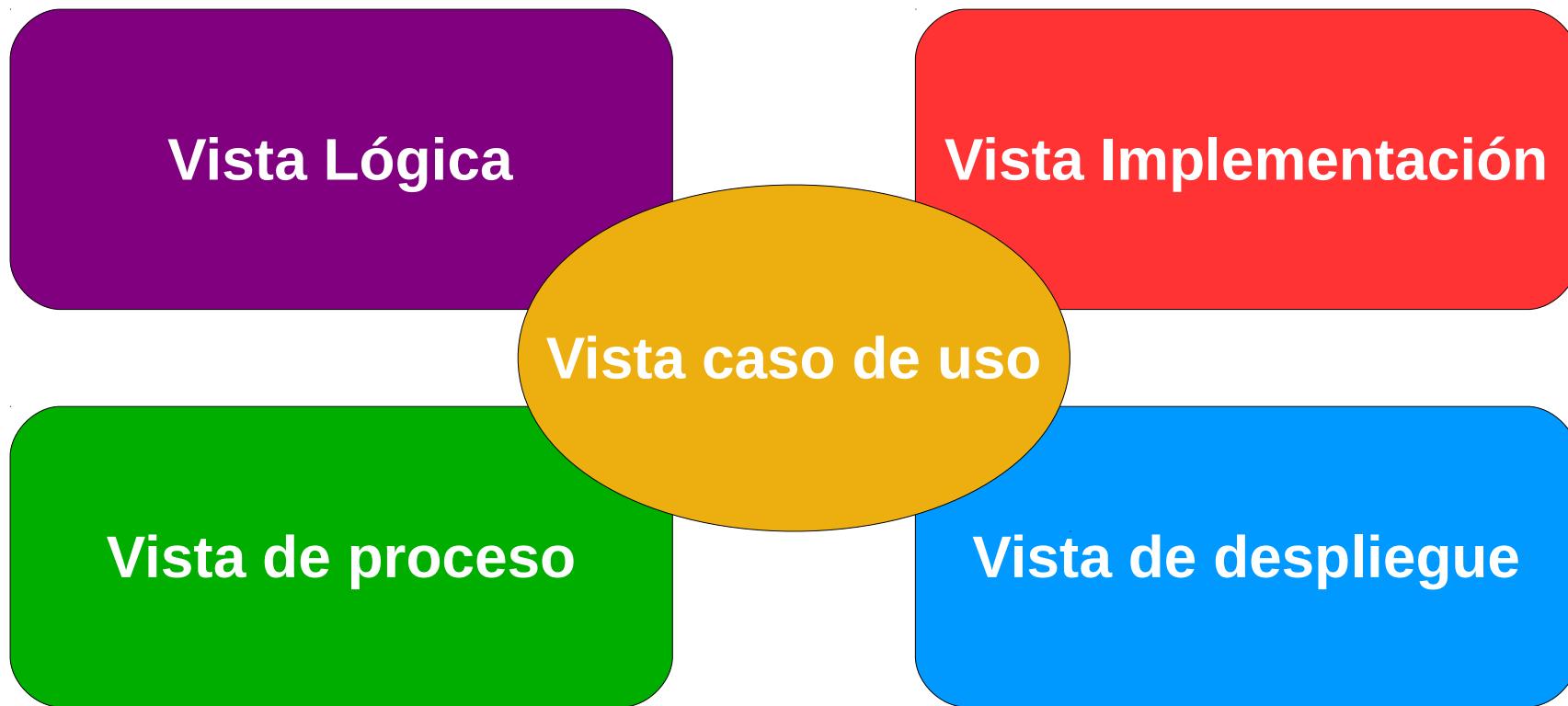
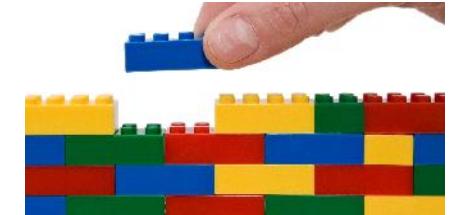
■ Forma de examinar la arquitectura

- ◆ *Vistas 4+1*

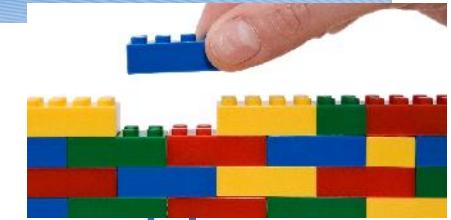
- ◆ *Cada una aborda diferentes aspectos*

Arquitectura

■ Vistas 4+1

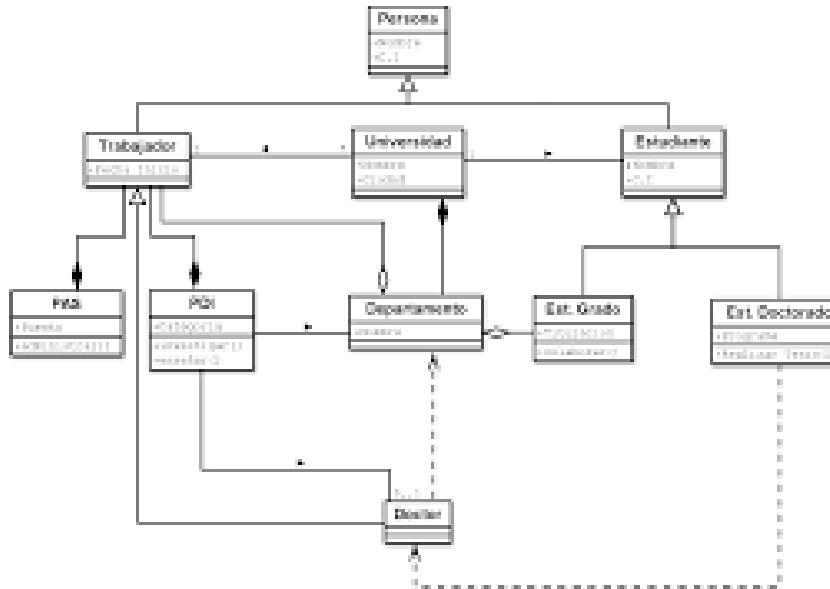


Arquitectura (Vista 4 +1)

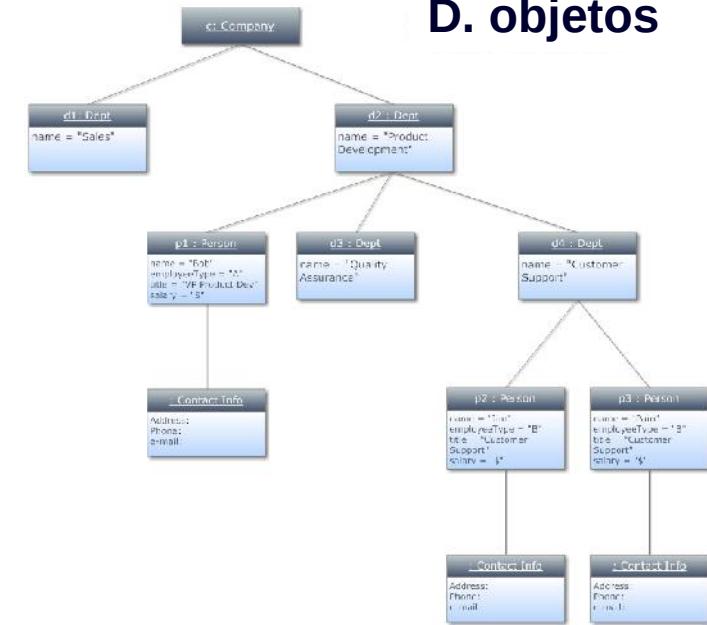


■ Vista lógica

- ◆ Captura el vocabulario del ámbito del problema como un conjunto de clases y objetos
- ◆ Describe la estructura y funcionalidad del sistema

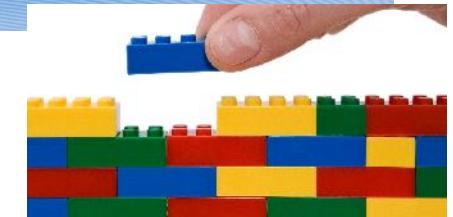


D. objetos



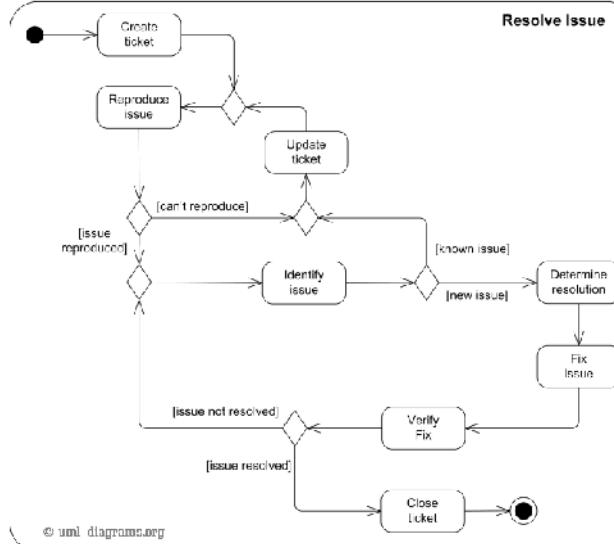
D. clases

Arquitectura (Vista 4 +1)

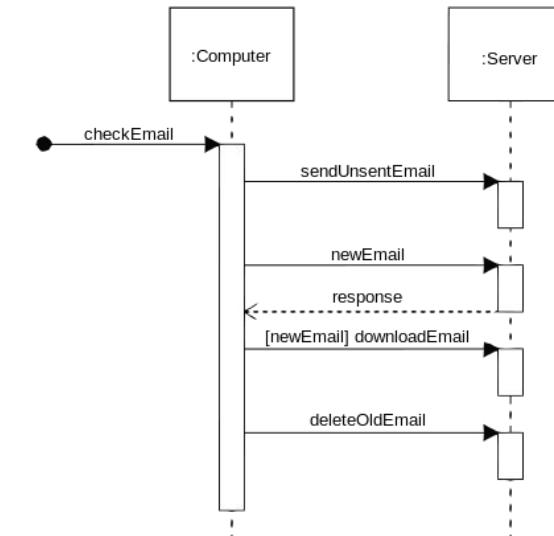


■ Vista de proceso

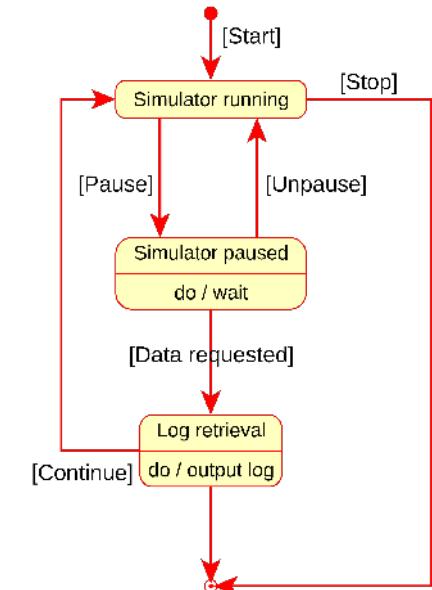
- ◆ Trata los aspectos dinámicos del sistema
- ◆ Explica los procesos del sistema y como se comunican



D. actividad



D. secuencia

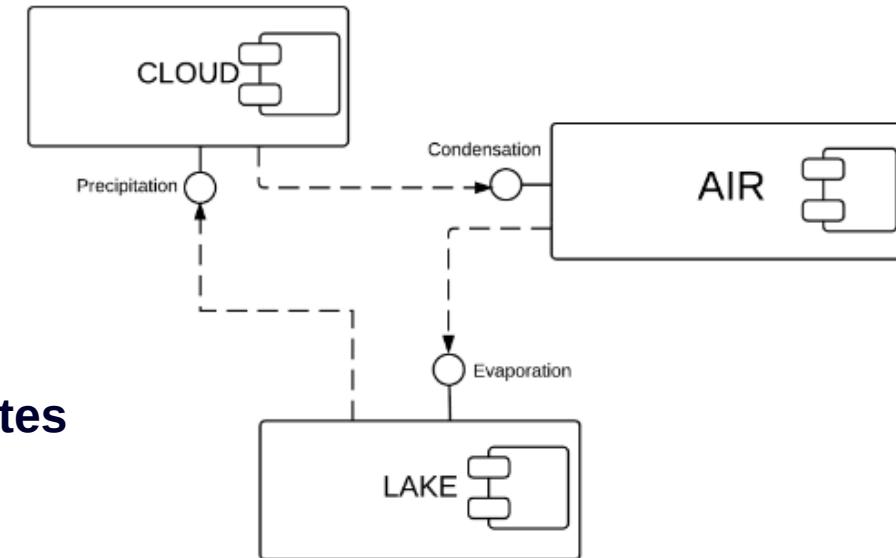
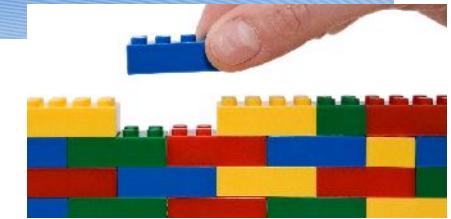


D. estados

Arquitectura (Vista 4 +1)

■ Vista implementación

- ◆ Ilustra el sistema desde el punto de vista del programador
- ◆ Modela los archivos y componentes que conforman la base de código físico del sistema

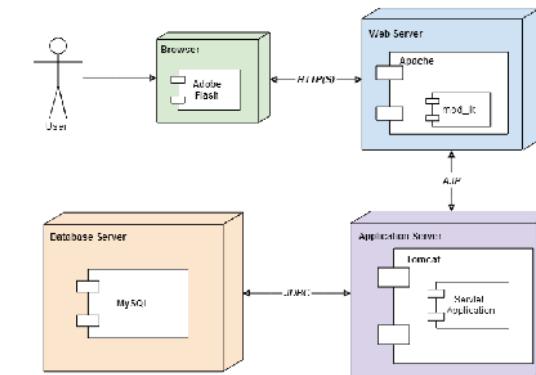


D. componentes

Arquitectura (Vista 4 +1)

■ Vista de despliegue

- ◆ Muestra el sistema desde el punto de vista de un ingeniero de sistemas
- ◆ Abarca los nodos que forman la topología hardware sobre la que se ejecuta el sistema
- ◆ Aborda la distribución, entrega e instalación de las partes que constituyen el sistema físico

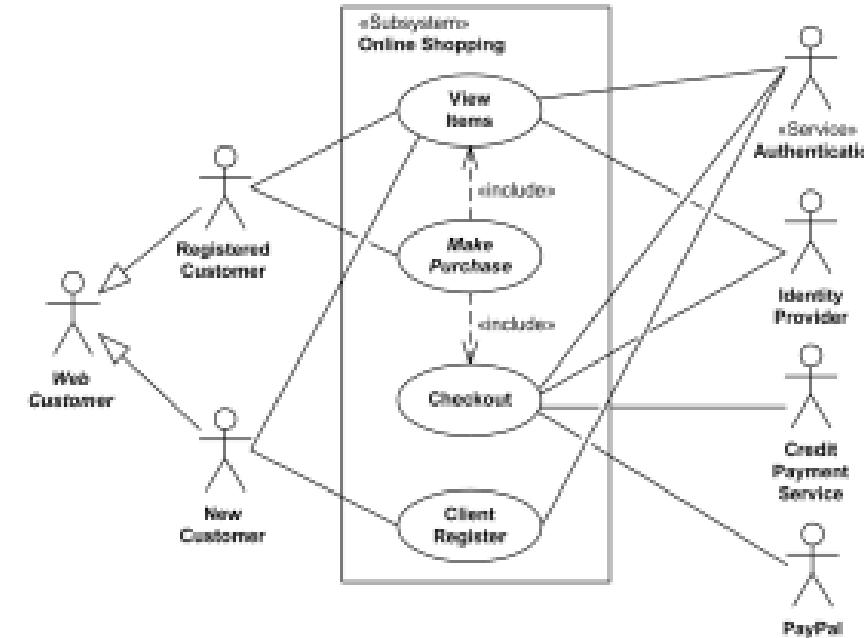


D. despliegue

Arquitectura (Vista 4 +1)

■ Vista de casos de uso

- ◆ Captura los requisitos básicos como un conjunto de casos de uso
- ◆ Proporcionan la base para la construcción de otras vistas



D. casos de uso

Testimonios empresariales

IBM:

“permite tener el control del ciclo de vida completo de un proyecto ...”

ERISSON:

“... el modelado del software es fundamental durante el desarrollo porque garantiza la construcción de la arquitectura software, permite entenderla de manera completa y reduce los riesgos ...”

ORACLE:

“podemos representar los objetos del negocio y entender todas la definiciones de la base de datos ...”

Testimonios empresariales

MICROSOFT:

“... reduce el costo de desarrollo, el tiempo y el riesgo de un proyecto, incrementa la reutilización de componentes ...”

REPUBLIC BANK:

“...provee una plantilla del sistema y ayuda a entender el comportamiento del sistema ...”

HEWLETT PACKARD

“... ayuda al equipo a entenderse y ver cual es su trabajo dentro del contexto de construcción del software. Hace viable la comunicación entre cliente, gente de análisis y diseño y sus herramientas de desarrollo ...”