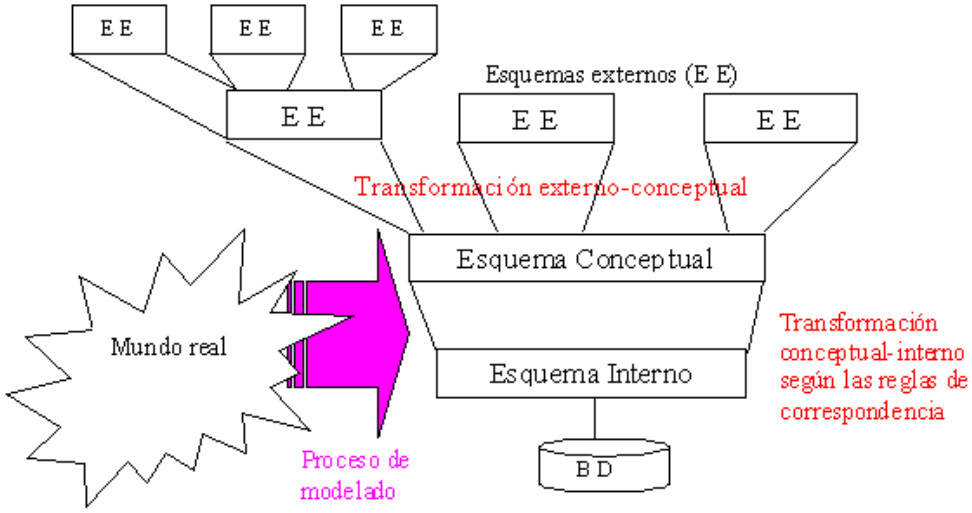


UNIVERSIDAD DE LOS ANDES

Bases de datos Unidad 2

Universidad de Los Andes
Escuela de Ingeniería de Sistemas
Departamento de Computación

Tema 2. Orientación por objetos y modelado de datos en UML

UNIVERSIDAD DE LOS ANDES

Tema 2: Orientación por objetos y modelado de datos en UML

- **Contenidos:**
  - Conceptos básicos y su notación en el lenguaje unificado de modelado (UML):
    - Elementos comunes, diagramas de casos de uso, diagramas de clases, diagramas de paquetes, diagramas de actividades, diagramas de vistas de interacción (secuencia y comunicación), diagramas de máquinas de estado, diagramas temporales, diagramas de despliegue y diagramas de componentes
- **Objetivo:**
  - Desarrollar habilidades en el modelado orientado por objetos y su representación con UML
- **Actividades:**
  - Elmasri y Navathe, cap. 12 y 20
  - Realizar ejercicio 2

UNIVERSIDAD DE LOS ANDES

Tema 2. Modelado de Datos en UML

- **Los lenguajes de modelado permiten representar y comunicar conocimiento acerca de un sistema**
  - Representan conocimientos
    - Posee constructos (símbolos) que facilitan el modelado de diferentes aspectos de un sistema
  - Comunican conocimientos
    - Los modelos que produce son utilizados, con fines diferentes, para comunicar el conocimiento representado
- **Un lenguaje de modelado es un sistema de sígnos usados para representar diferentes aspectos de un sistema**

UNIVERSIDAD DE LOS ANDES

Lenguajes de modelado

- **Un lenguaje de modelado consta de**
  - Un vocabulario:
    - Conjunto de símbolos (constructos) empleados para modelar
  - Una sintáxis:
    - Conjunto de reglas que describen como se usan los símbolos
  - Una semántica:
    - Describe el significado de los símbolos
- **Las notaciones son, también, medios para modelar; pero tienen una semántica menos rigurosa que los lenguajes**
- **Lenguajes de modelado más recientes:**
  - UML (*Unified Modeling Language*), UML Business
  - BPML (*Business Process Modeling Language*)
  - WebML (*Web Modeling Language*)

UNIVERSIDAD DE LOS ANDES

UML

- **UML (*Unified Modeling Language*):**
  - Es un lenguaje de modelado de sistemas de software que integra y unifica diferentes notaciones y lenguajes formales
  - Facilita la representación del conocimiento acerca de un sistema y la comunicación de dicho conocimiento
- **Es un estándar administrado por el consorcio OMG**
  - *Object Management Group* (www.omg.org)
- **Ha evolucionado agregando mayor poder y capacidad semántica a cada nueva versión**
  - Versiones más recientes:
    - UML 2.0 (Junio 2003)
    - UML 2.5 (Septiembre 2013)
- **Inicio de los ochenta: Métodos orientados por objetos**

UNIVERSIDAD DE LOS ANDES

Génesis de UML

- **Finales de 1994: nociones de clases y asociaciones (Rambaugh-OMT), divisiones en subsistemas (Booch) y casos de uso (Jacobson)**
- **1995: Método unificado versión 0.8.**
- **1996: versión 0.9**
- **1997: UML versión 1.0**
- **Origen: Fusión de las notaciones de Booch (1993), OMT (1996), OOSE y otras**
- **Características: simplicidad, intuitiva, homogénea, coherente, genérica, extensible y configurable**
- **Objetivo: Descripción de los artefactos de desarrollo de programas**

UNIVERSIDAD DE LOS ANDES

UML

- **Es utilizado en la industria del software para:**
  - especificar,
  - diseñar,
  - visualizar,
  - comunicar y
  - documentar sistemas de software y aplicaciones
- **Consta de un conjunto de notaciones gráficas y un lenguaje formal**
  - Las notaciones gráficas son usadas para:
    - Modelar la **estructura**, **funcionalidad**, **comportamiento** e **implementación** de un sistema
    - Organizar los modelos producidos
  - El lenguaje formal es usado para expresar formalmente restricciones acerca de los elementos modelados de un sistema
    - OCL (*Object Constraint Language*) permite representar:
      - Invariantes
      - Pre y postcondiciones
      - Referencias nulas
      - Otras restricciones

- **Permite modelar un sistema de software desde diferentes perspectivas**
  - Funcional
    - Usos del sistema
  - Estructural
    - Abstracciones del sistema
      - Clases, objetos, relaciones e interacciones
  - Comportamiento
    - Dinámica del sistema
  - Implementación
    - Componentes del sistema
    - Despliegue (instalación) de los componentes del sistema

UNIVERSIDAD DE LOS ANDES

UML 1.0, 1.4 y 1.5

- **Consta de nueve (9) notaciones gráficas (tipos de diagramas) y un (1) lenguaje formal (OCL):**
  - Notaciones estructurales:
    - Diagramas de clase
    - Diagramas de objetos
  - Notaciones orientadas al usuario o funcionales:
    - Diagramas de casos de uso
  - Notaciones de comportamiento:
    - Diagramas de secuencias
    - Diagramas de colaboración
    - Diagramas de estado
    - Diagramas de actividad
  - Notaciones orientadas a la implementación:
    - Diagramas de componentes
    - Diagramas de despliegue (*deployment*)

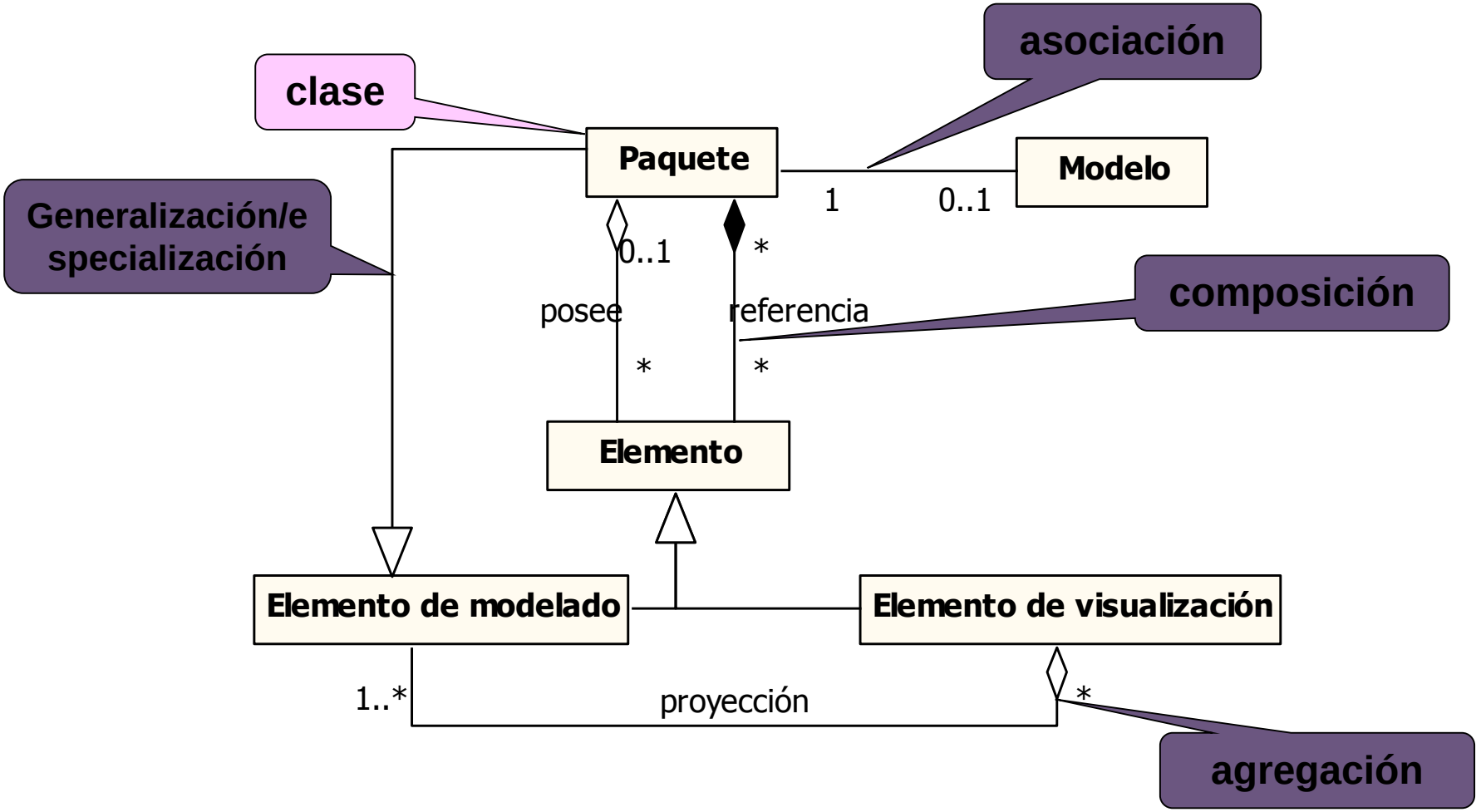
UNIVERSIDAD DE LOS ANDES

UML 2.0 y siguientes

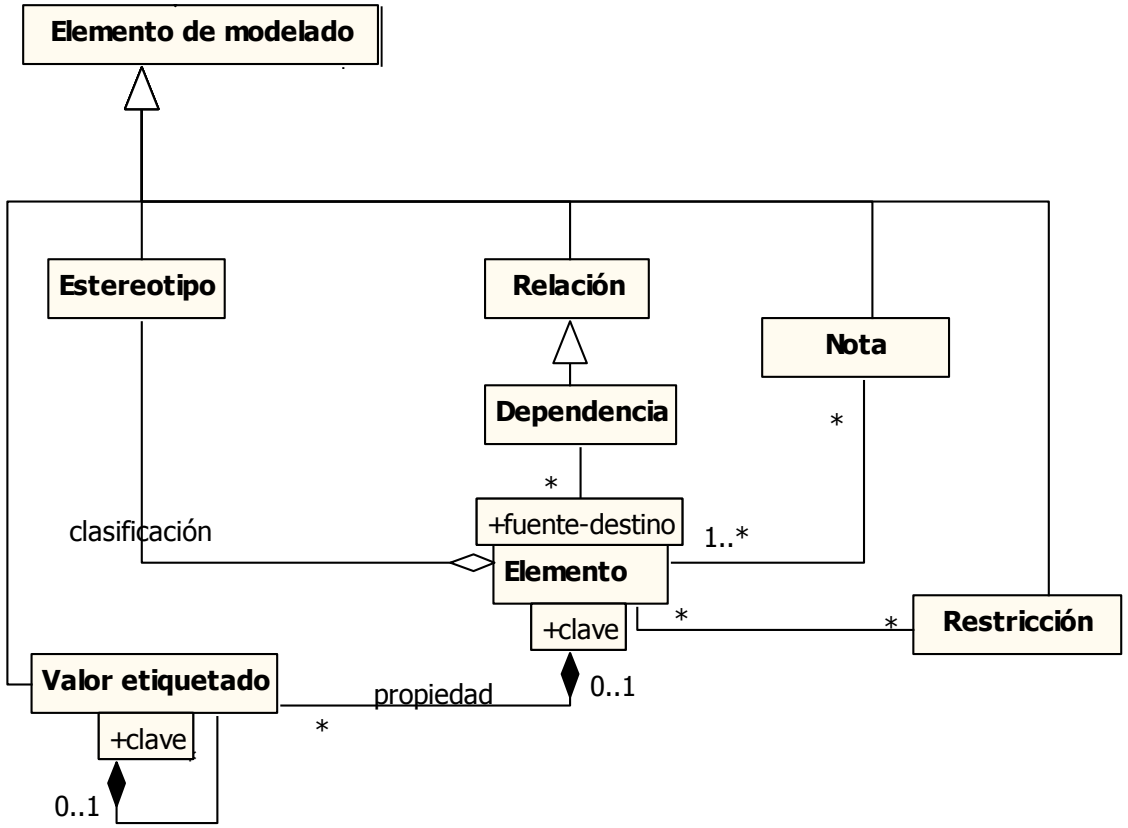
- **Los 13 tipos de diagramas se clasifican en dos grupos:**
  - **Diagramas para Modelado Estructural**
    - Diagramas de clase
    - Diagramas de objetos
    - Diagramas de componentes
    - Diagramas de estructura compuesta
    - Diagramas de despliegue (*deployment*)
    - Diagramas de paquetes
  - **Diagramas para Modelado del Comportamiento:**
    - Diagramas de casos de uso
    - Diagramas de interacción:
      - Secuencias, Comunicación, Temporización, Vistas de Interacción
    - Diagramas de máquinas de estado
    - Diagramas de actividad

- **Documentos oficiales de UML 2.5 (www.omg.org/uml)**
  - UML 2.5 OMG, Septiembre 2013
  - UML 2.4 Object Constraint Language, Febrero 2014. Define el lenguaje formal OCL

Elementos comunes

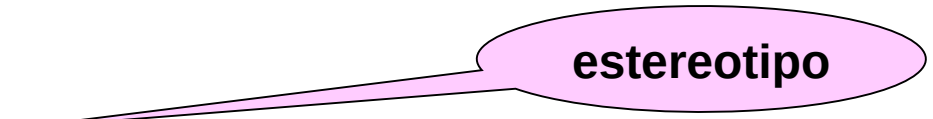


Mecanismos comunes



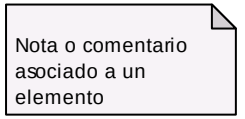
Elementos comunes

- **Estereotipos:**
  - especifican la semántica del elemento de modelado <<semántica>>
- **Perfiles <<profile>> :**
  - paquete con estereotipo que contiene elementos de modelado que han sido personalizados para una plataforma (J2EE, .NET, etc.), dominio (modelado de procesos del negocio) o propósito (modelado de hilos de seguridad) particular



Elementos comunes

- **Templates:**
  - descriptor para los elementos de modelado, que pueda ser usado para crear una familia de elementos de modelado relacionados
  - Puede aplicarse también a paquetes y a las colaboraciones
- **Notas:**
  - comentarios asociados a un elemento de modelado
- **Restricciones:**
  - condiciones o aserciones



UNIVERSIDAD DE LOS ANDES

Tipos básicos

- **No son elementos de modelado ⇒No poseen ni estereotipos, ni etiquetas, ni restricciones**
- **Booleano (*bool*):**
  - tipo enumerado: Verdadero y falso
- **Expresión:**
  - cadena de caracteres
- **Lista:**
  - contenedor que puede ser ordenado e indizado
- **Multiplicidad:**
  - conjunto no vacío de enteros positivos extendido con *
- **Cadena (*string*):**
  - cadena de caracteres con nombre
- **Tiempo (*time*):**
  - cadena que representa un tiempo absoluto o relativo

UNIVERSIDAD DE LOS ANDES

Tipos básicos

- **Nombre (*name*):**
  - cadena de caracteres que designa un elemento

  Nombre compuesto = nombre simple {'.' nombre compuesto}.
  Nombre calificado con el paquete al que pertenece = paquete '::' nombre
- **Punto (*point*):**
  - tupla (x, y, z) posición en el espacio, por omisión z = 0
- **No interpretado:**
  - blob, su significado depende del dominio

UNIVERSIDAD DE LOS ANDES

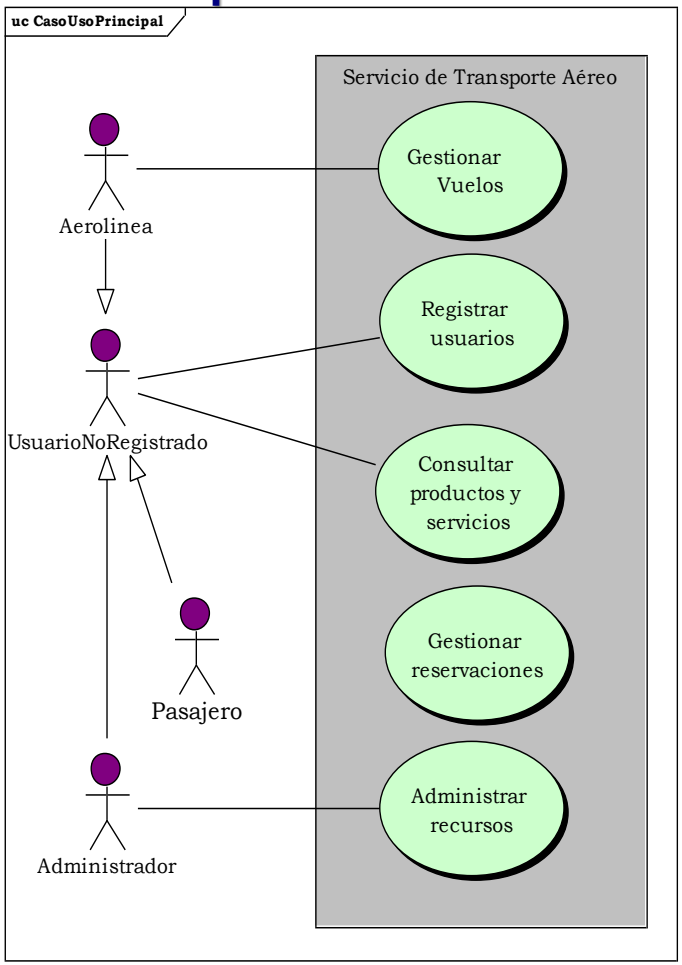
Diagramas

- **Permiten visualizar y manipular los elementos de modelado**
- **Son grafos donde los nodos y los arcos tienen diferentes representaciones dependiendo del tipo de diagrama**
- **Muestran todo o parte de las características de los elementos de modelado, según el nivel de detalle útil en el contexto del diagrama**
  - Cada diagrama se utiliza para representar un determinado aspecto de la aplicación, como por ejemplo: el diagrama de estados muestra los posibles estados en los que se pueden encontrar los objetos de una o varias clases cuando hay interacciones entre ellos

Vista de usuarios o vista funcional

- **Diagramas de casos de uso**
  - Descripción de un proceso fin-a-fin relativamente largo que típicamente incluye varias etapas o transacciones, **NO es una etapa o actividad individual de un proceso**
  - Describen bajo la forma de acciones y reacciones el comportamiento de un sistema desde el punto de vista de un usuario; permiten definir los límites del sistema y las relaciones entre el sistema y su entorno

Modelado del comportamiento



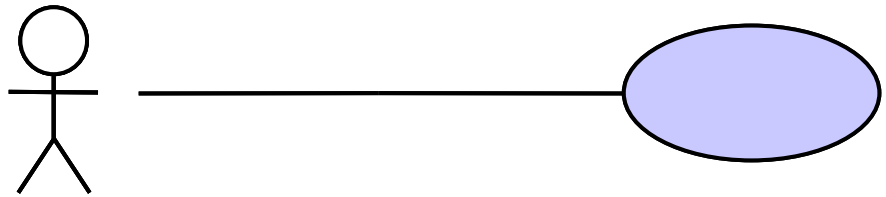
UNIVERSIDAD DE LOS ANDES

Interesados (stakeholders)

- **Son todos aquellas personas que tienen algún interés en el sistema o aplicación**
  - Usan el sistema
  - Participan en su desarrollo o mantenimiento
  - Toman decisiones con respecto a su desarrollo o mantenimiento
- **Tipos:**
  - Personas que usan o usarán el sistema
    - Usuarios directos o indirectos
  - Personas que desarrollan o mantienen el sistema
    - Miembros del grupo de desarrollo:
      - líder del proyecto,
      - analistas,
      - diseñadores,
      - programadores, etc.
    - Consultores o asesores del proyecto
  - Personas o entes cuyos intereses se ven afectados por el sistema

Casos de uso

- **Es una manera específica de utilizar el sistema**
- **Es la imagen de una funcionalidad del sistema, desencadenada en respuesta a la estimulación de un actor externo**



- ***Un actor representa el rol jugado por una persona o cosa que interactúa con el sistema***
- ***La misma persona puede jugar varios roles y varias personas pueden jugar el mismo rol***

UNIVERSIDAD DE LOS ANDES

Casos de uso

- **Categorías de actores:**
  - ***Principales:*** *las personas que utilizan las funciones principales*
  - ***Secundarios:*** *las personas que efectúan tareas administrativas o de mantenimiento*
  - ***Externos:*** *los dispositivos que forman parte del dominio de aplicación y que deben ser utilizados*
  - ***Otros sistemas:*** *los sistemas con los que debe interactuar*
- **Los casos de uso se determinan observando y precisando, actor por actor, las secuencias de interacción (escenarios) desde el punto de vista del usuario**
- **La descripción se debe concentrar en lo que DEBE HACERSE y NO en la manera de hacerlo**
- **Si un caso de uso es muy complejo (por ejemplo más de diez páginas) es posible dividirlo en casos más pequeños**

UNIVERSIDAD DE LOS ANDES

Casos de uso

- ***El enfoque orientado por objetos materializa un caso de uso por medio de la colaboración entre los objetos***
- ***Los escenarios se representan con los diagramas de interacción***
- ***Antes de hacerlos, se debe tratar de entender los requerimientos del sistema, expresando lo que el sistema debe hacer, a saber: El sistema debe hacer 1, 2, 3, ..., etc.***
- **Para su realización es conveniente responder las preguntas:**
  - ¿Cuáles son las tareas del actor?
  - ¿Cuáles datos o información el actor debe crear, guardar, modificar, destruir o leer?
  - ¿Debe el actor informar al sistema de los cambios externos?
  - ¿Debe el sistema informar al actor las condiciones internas?

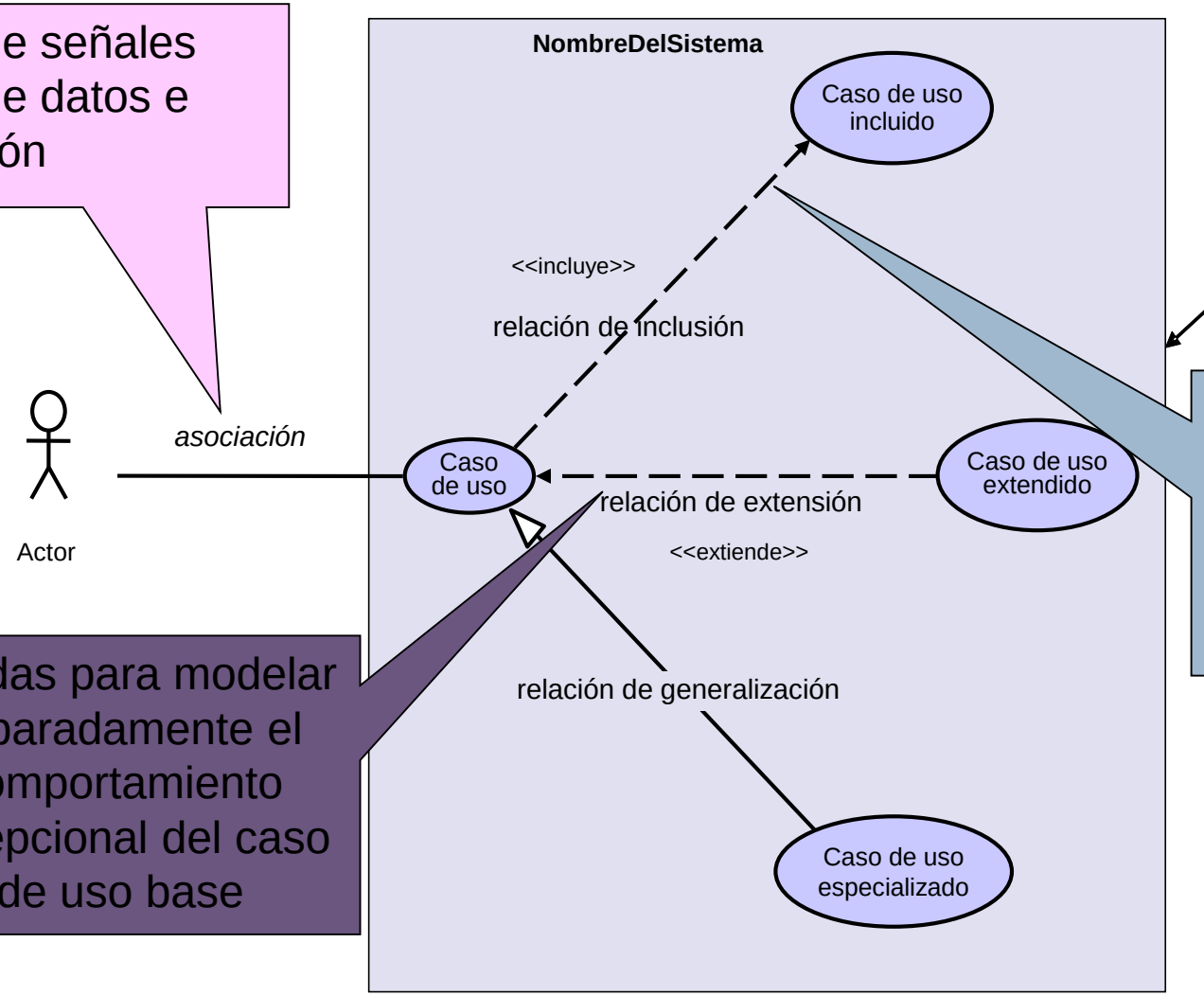
UNIVERSIDAD DE LOS ANDES

Casos de uso

- **Los diagramas de casos de uso modelan:**
  - Los actores de un sistema
  - Los casos de uso
  - Las relaciones entre actores
  - Las relaciones entre casos de uso
  - Las relaciones de comunicación entre actores y casos de uso
  - Los límites del sistema
  - El refinamiento o descomposición de los casos de uso

Forma general de un diagrama de casos de uso

- Envío de señales
- Envío de datos e información

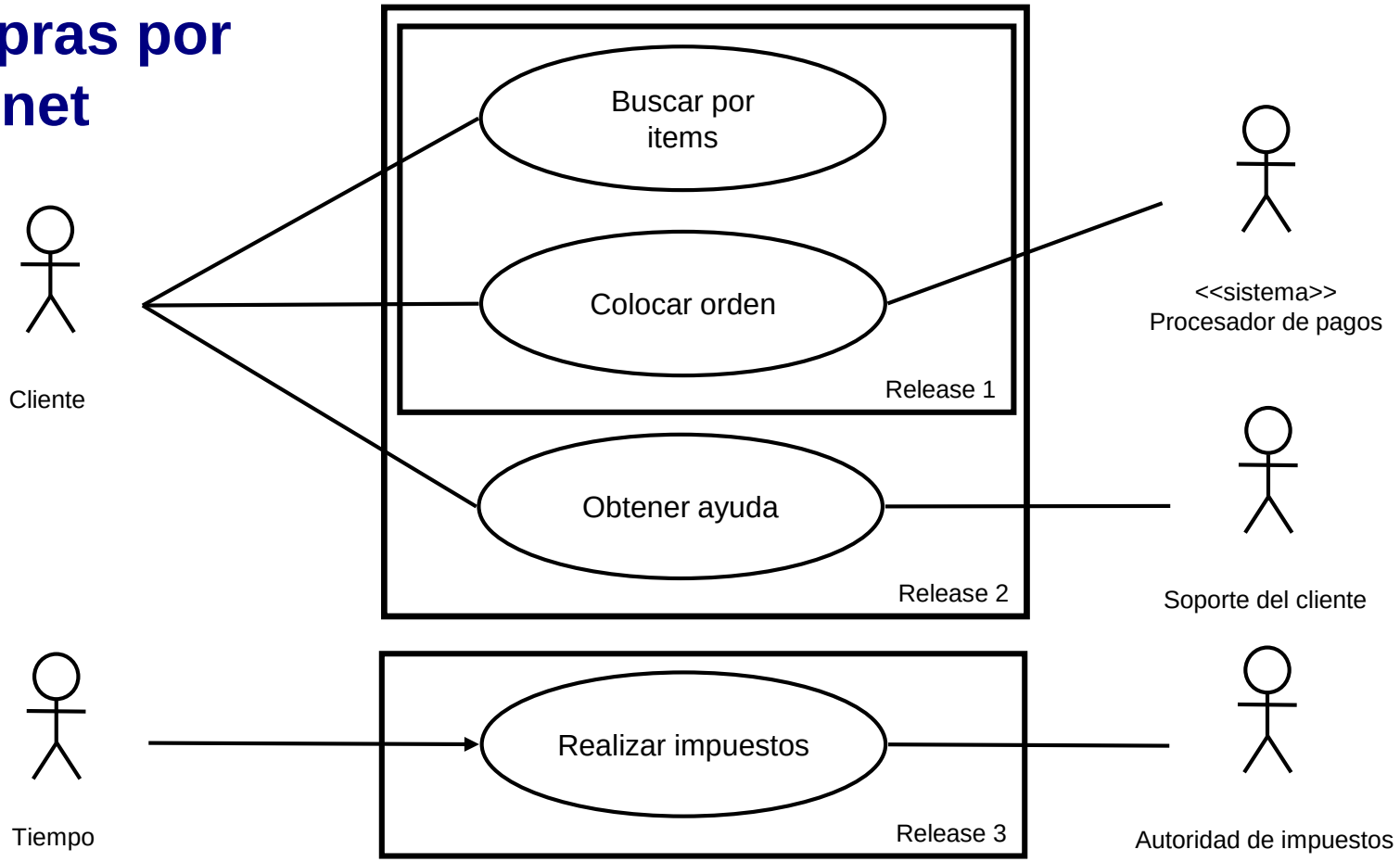


usados para compartir comportamiento común entre varios casos de uso

Usadas para modelar separadamente el comportamiento excepcional del caso de uso base

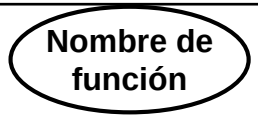
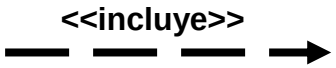
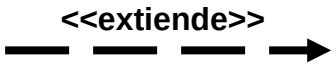
Ejemplo de diagrama de casos de uso

Sistema de compras por internet



UNIVERSIDAD DE LOS ANDES

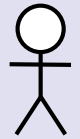
Casos de uso

Nombre	*Símbolo*	*Descripción*
Actor		**Representa el Rol que juega un objeto externo (persona o entidad que interactúa con el sistema)**
Caso de uso		**Función que el sistema pone a la disposición de los actores. Produce un resultado de valor para los actores**
Relación de comunicación		**Usado para establecer una asociación bidireccional entre un actor y un caso de uso**
Relación de *inclusión*		**Usado para establecer una relación entre dos casos de uso, en la cual un caso de uso contiene (incluye) el comportamiento de otro**
Relación de extensión		**Usado para establecer una relación entre dos casos de uso, en la cual un caso de uso extiende el comportamiento de otro**
Relación de generalización		**Usado para establecer relaciones del tipo "ES_UN(A)" entre elementos (actores o casos de uso) generales y específicos**
Límite del sistema		Indica el contenido del sistema o aplicación

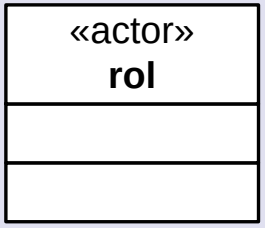
Actor

- **Símbolo usado para representar el rol que objetos externos, de una misma clase, juegan cuando interactúan con el sistema**
  - Un objeto externo puede ser una persona interesada (*stakeholder*), un dispositivo u otro sistema
  - No se refieren a un individuo particular, sino a una clase de individuos que tienen un rol común
- **Representan a entes externos al sistema**
- **Intercambia señales y datos con el sistema**
- **Es un clasificador y no una ocurrencia**

Representación icónica



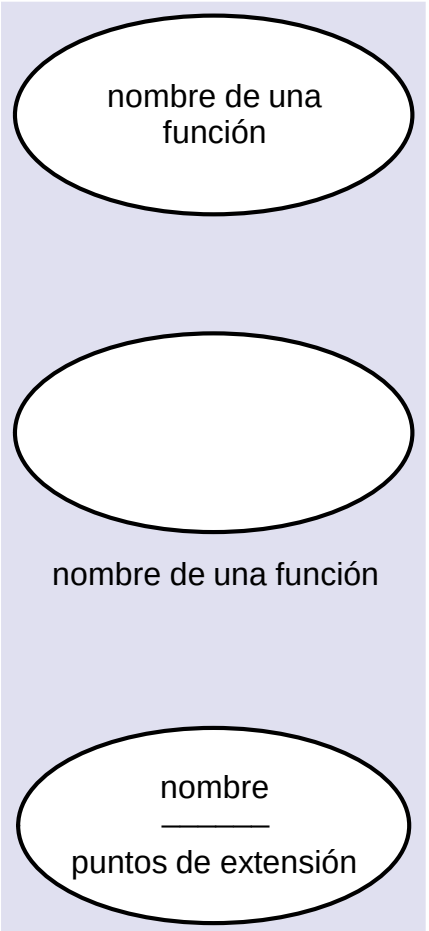
Clasificador con estereotipo



rol

Caso de uso

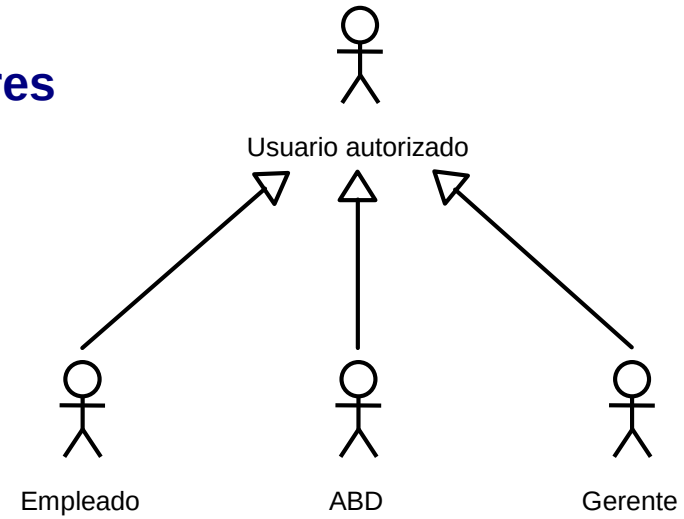
- **Símbolo usado para representar una función que ejecuta el sistema**
  - Un conjunto de acciones que cuando son ejecutadas por el sistema producen un resultado observable
- **Su resultado es de valor para uno o más actores**
- **Un caso de uso es iniciado por un actor**
- **Es un clasificador y no una instancia**



Relaciones entre los actores

- **Se pueden establecer relaciones de generalización entre los actores**
- **Un rol general puede ser heredado por uno o más roles específicos**

Jerarquía de actores



Super-rol

Sub-roles

Reglas de estilo para Diagramas de Casos de Uso

1. **Cada actor y caso de uso debe tener un nombre único**
2. **Los actores deben tener nombres y/o íconos representativos**
   - Los nombres deben indicar roles
3. **El nombre de un caso de uso debe indicar acción y debe ser claro y conciso**
   - Forma general: Verbo + predicado
   - Ejemplos: Imprimir reporte de ventas
4. **Los casos de uso de un diagrama deben estar todos a un mismo nivel de abstracción**
5. **Evite el cruce de líneas**
6. **Evite tener demasiados casos de uso en un mismo diagrama**
   - Use la regla de 5±2
7. **Evite el uso complejo de relaciones de extensión e inclusión**
   - No más de tres niveles de relaciones consecutivas

Descripciones textuales de los casos de uso

- La simplicidad de los diagramas de casos de uso tienen un costo asociado por baja expresividad
- Las Descripciones Textuales complementan los diagramas de casos de usos
- Cada caso de uso debe tener asociado una descripción textual

UNIVERSIDAD DE LOS ANDES

Plantilla para la descripción textual

Caso de uso:	**<nombre del caso de uso>**
Actores participantes:	**<lista de actores que participan>**
Condiciones de entrada:	**<precondiciones>**
Flujo de eventos:	**<evento 1>** **<evento 2>** **… (los eventos son del tipo: "El actor hace esto" o "El sistema hace aquello")**
Condiciones de salida:	**<postcondiciones>**
Flujos alternativos:	**<evento i.1>** **<eventos en caso de excepciones>**
Requisitos especiales:	**<requerimientos no-funcionales asociados al caso de uso>**
Notas:	***<notas adicionales o aclaratorias>***

UNIVERSIDAD DE LOS ANDES

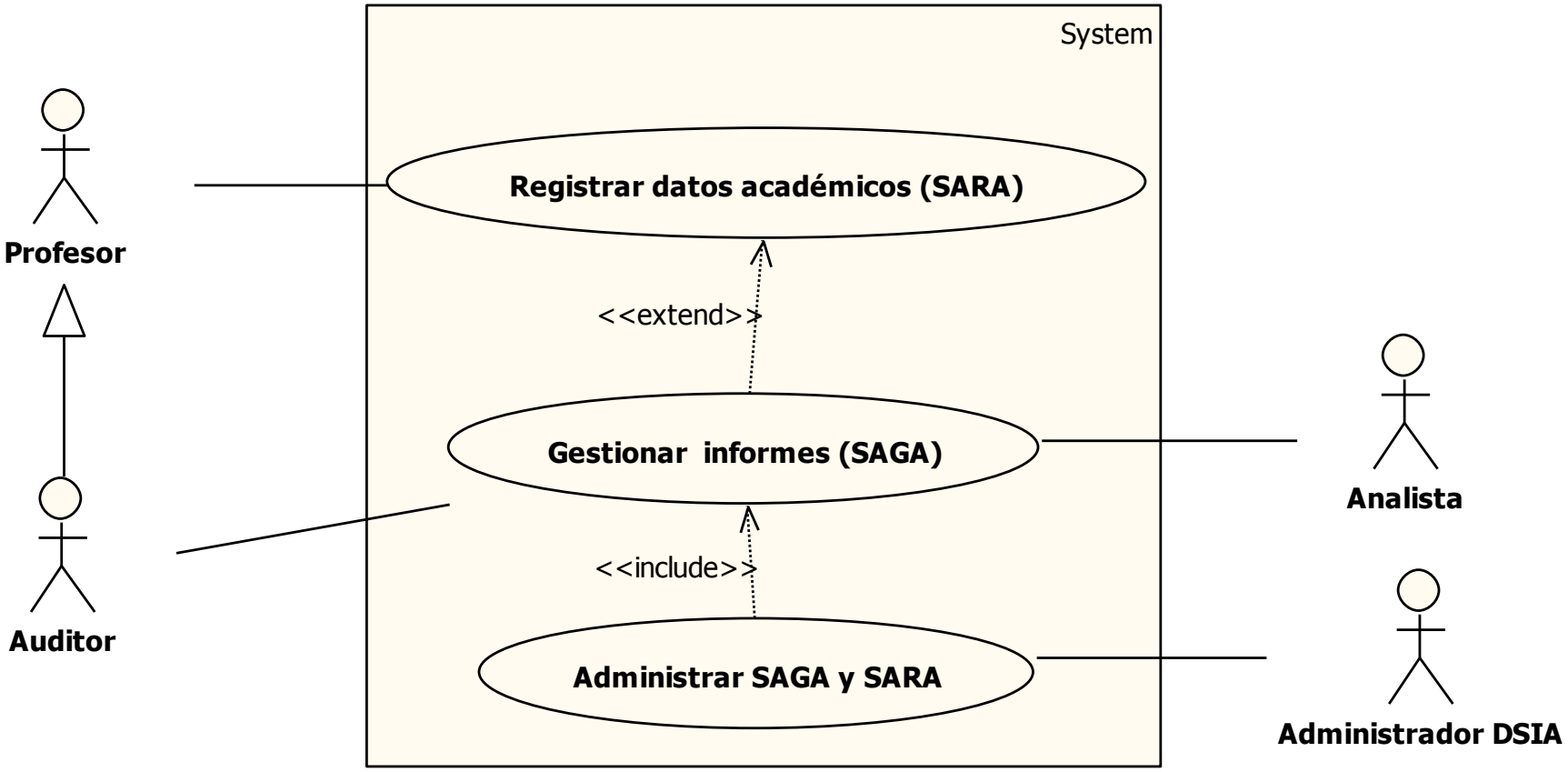
Reglas para la descripción textual

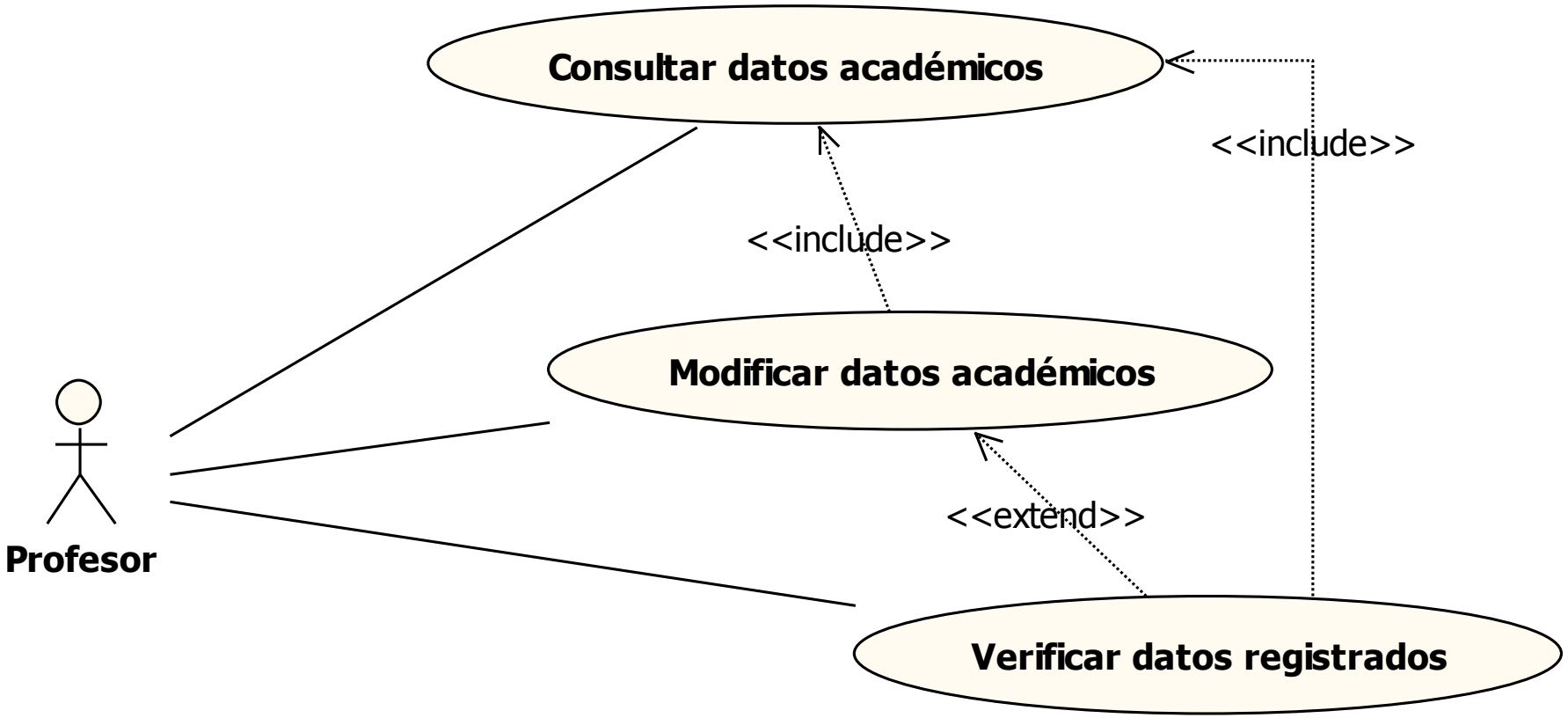
1. **Narrar el flujo de eventos usando voz activa, en tiempo presente y desde la perspectiva del actor**
   - Evite el uso de voz pasiva:
     - "La clave es introducida por el usuario"
   - Prefiera el uso de la voz activa:
     - "El usuario introduce la clave"
     - "El sistema valida la clave"
2. **Exprese cada paso del flujo usando la forma "llamada y respuesta":**
   - El texto debe reflejar el hecho de que al actor ejecuta algo y el sistema responde a la solicitud del actor
     - "El [actor] hace [esto]" y "El sistema hace [aquello]"

Reglas para la descripción textual

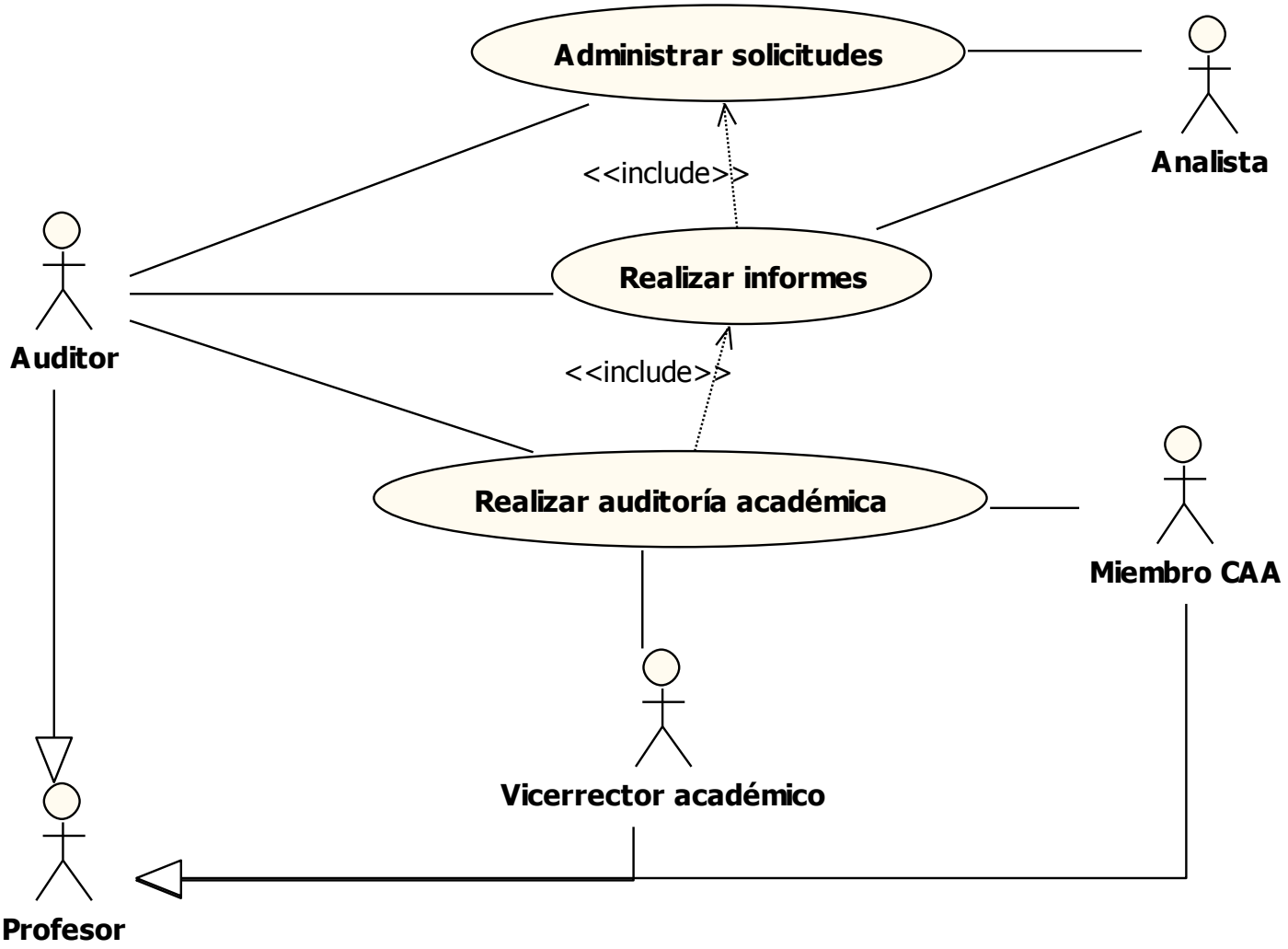
3. **El caso de uso que se describe debe expresar un solo requisito funcional**
   - Pero, pueden haber uno o más requisitos no-funcionales asociados al requisito funcional descrito
4. **Establezca el contexto inicial del actor**
   - Especifique la ubicación del actor en el contexto de la interfaz de usuario del sistema
     - Ej. El Cliente introduce su clave de identificación en la ventana de identificación al inicio del sistema
5. **Asegúrese que el caso de uso produzca un resultado visible y de valor para el cliente**
6. **Establezca todos los posibles flujos alternativos al flujo principal (flujo feliz)**

Ejemplo


System
Registrar datos académicos (SARA)
<<extend>>
Gestionar informes (SAGA)
<<include>>
Administrar SAGA y SARA
Profesor
Auditor
Analista
Administrador DSIA


Consultar datos académicos
<<include>>
<<include>>
Modificar datos académicos
<<extend>>
Profesor
Verificar datos registrados

SAGA


Administrar solicitudes
Analista
<<include>>
Realizar informes
Auditor
<<include>>
Realizar auditoría académica
Miembro CAA
Vicerrector académico
Profesor

Plantilla para la descripción textual

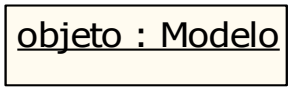
Caso de uso:	**Administrar solicitudes**
Actores participantes:	**Analista, Auditor**
Condiciones de entrada:	**Que se tenga al menos una solicitud pendiente**
Flujo de eventos:	**1. El actor solicita al sistema que liste el contenido de la cola de auditorías pendientes** **2. El actor ingresa las nuevas solicitudes en la cola** **3. El sistema actualiza la cola de solicitudes pendientes** **4. El actor verifica la prioridad de las auditorías en la cola y selecciona la de mayor prioridad que esté primero y que tenga todos sus recaudos para eliminarla de la cola** **5. El sistema elimina la solicitud seleccionada de la cola** **6. El actor coloca la solicitud seleccionada para procesarla**
Condiciones de salida:	**La cola de solicitudes cambia de tamaño con el número de solicitudes insertadas menos la solicitud seleccionada**

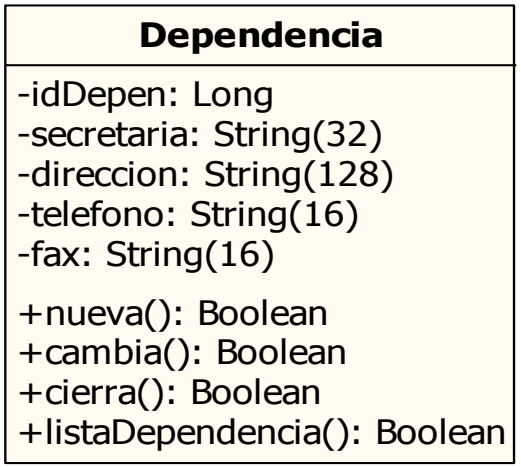
Plantilla para la descripción textual

Caso de uso:	**Administrar solicitudes (Continuación)**
Flujos alternativos:	**Ninguno**
Requisitos especiales:	**Mantener los tiempos de espera de las solicitudes en cola actualizados**
Notas:	**La cola de solicitudes es una cola por prioridad, donde aquellas solicitadas por el Consejo Universitario tienen la máxima prioridad**

Vista lógica o estructural

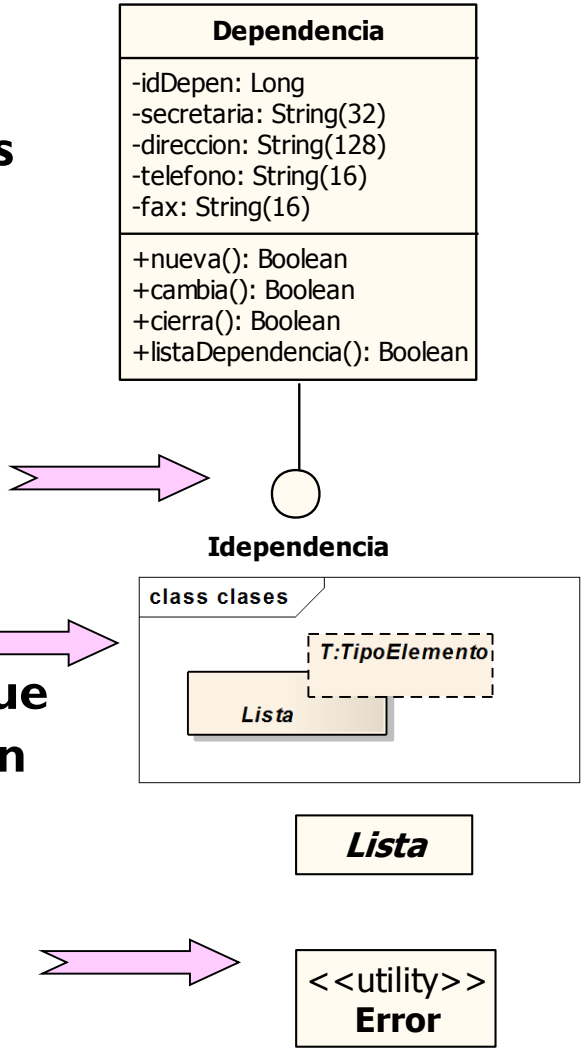
- **Objeto:** representación de algo que se describe mediante un *identificador*, una *estructura* y un *comportamiento*
- **Atributos:** propiedades relevantes de un objeto que representa su estructura. **Simples o compuestos**
- **Clasificación:** agrupación de objetos con propiedades y comportamiento similares dentro de una clase
- **Clase:** objeto que define 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





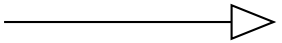
Vista lógica o estructural

- **Extensión de clase:** conjunto de todas las instancias de una clase
- **Instanciación:** proceso de generación o creación de las instancias de una clase
- **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

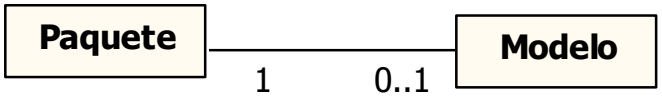


Vista lógica o estructural

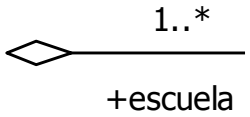
- **Jerarquía de clases:** **relación ES-UN(A), abstracciones de generalización/especialización de clases**



- **Herencia:** **propiedad que tienen las clases de heredar de sus superclases estructura y/o comportamiento. Simple o múltiple.**
- **Asociaciones:** **relaciones estructurales entre las clases, pueden ser: reflexivas, binarias, etc.**



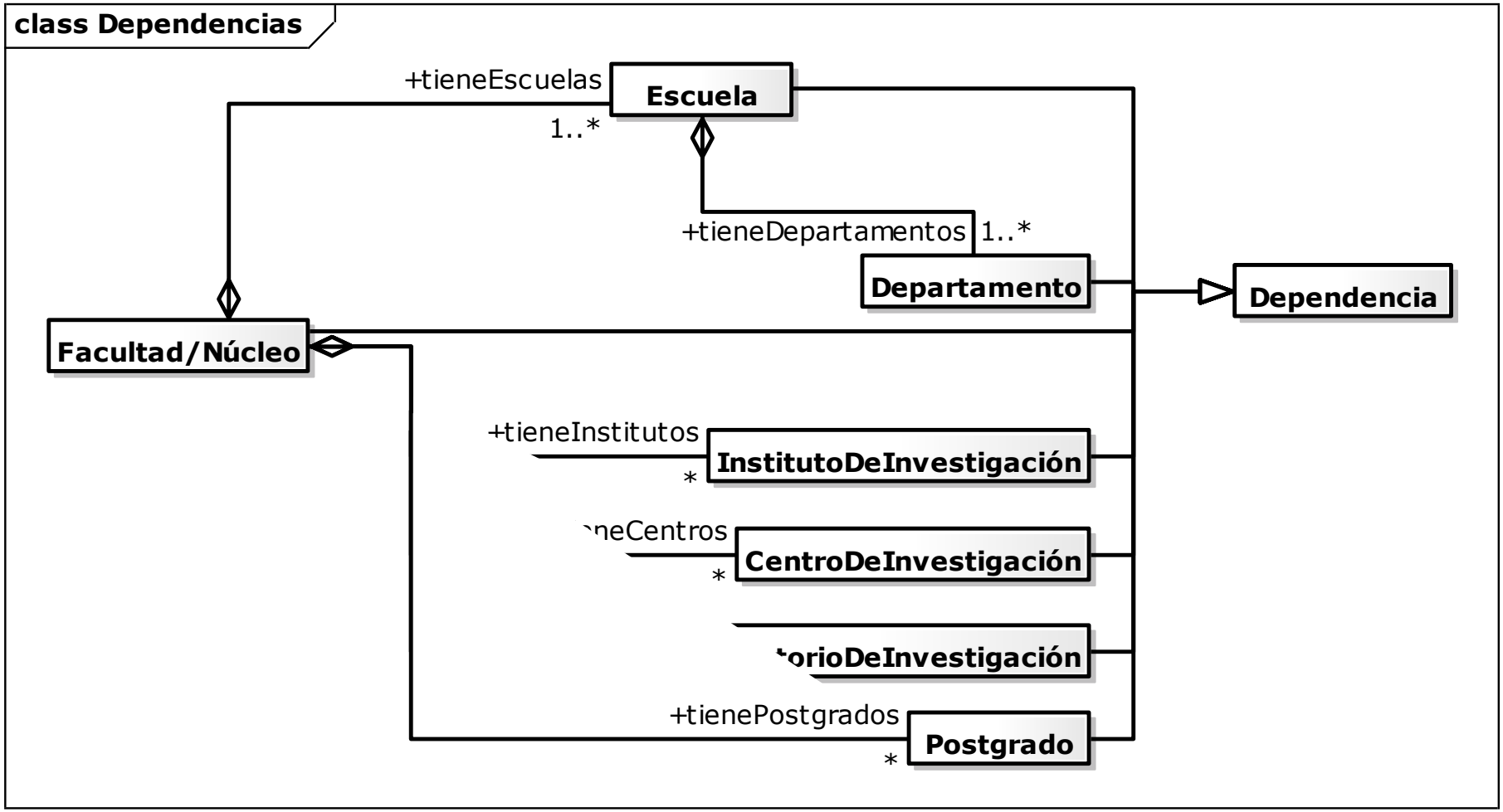
- **Agregaciones:**



  - Una clase forma parte de otra
  - Los valores de los atributos de una clase se propagan en los valores de los atributos de otra
  - Una acción sobre una clase implica una acción sobre otra
  - Los objetos de una clase están subordinados a los objetos de otra

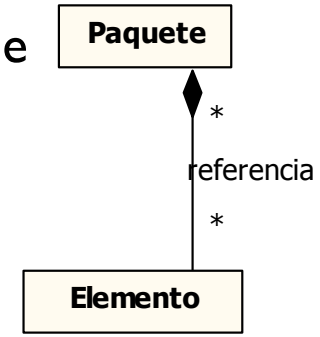
Ejemplo de diagrama de clases

Modelado estructural



Vista lógica o estructural

- **Composición:** relación ES-PARTE-DE, asimétrica, una de las extremidades juega un rol predominante en relación a la otra, por lo cual sólo lleva un rol. Indica clases de objetos compuestos



- **Mensajes:** especificación de un objeto junto con la invocación de uno de sus métodos

  objetoDestino.nombreDelMétodo(listaDeArgumentos)

- **Encapsulación:** propiedad que permite que los objetos sean definidos en su estructura y su comportamiento, obligando al uso del pase de mensajes o de la invocación de sus métodos, si se quiere acceder al objeto

Vista lógica o estructural

- **Polimorfismo:** se puede usar el mismo nombre para la definición de un método en varias clases sin importar la relación entre las mismas
- **Reescritura o sobrecarga:** permite nombrar código diferente con el mismo nombre para más de una clase de objetos
- **Encadenamiento tardío:** permite seleccionar el código adecuado al objeto definido en la invocación del método

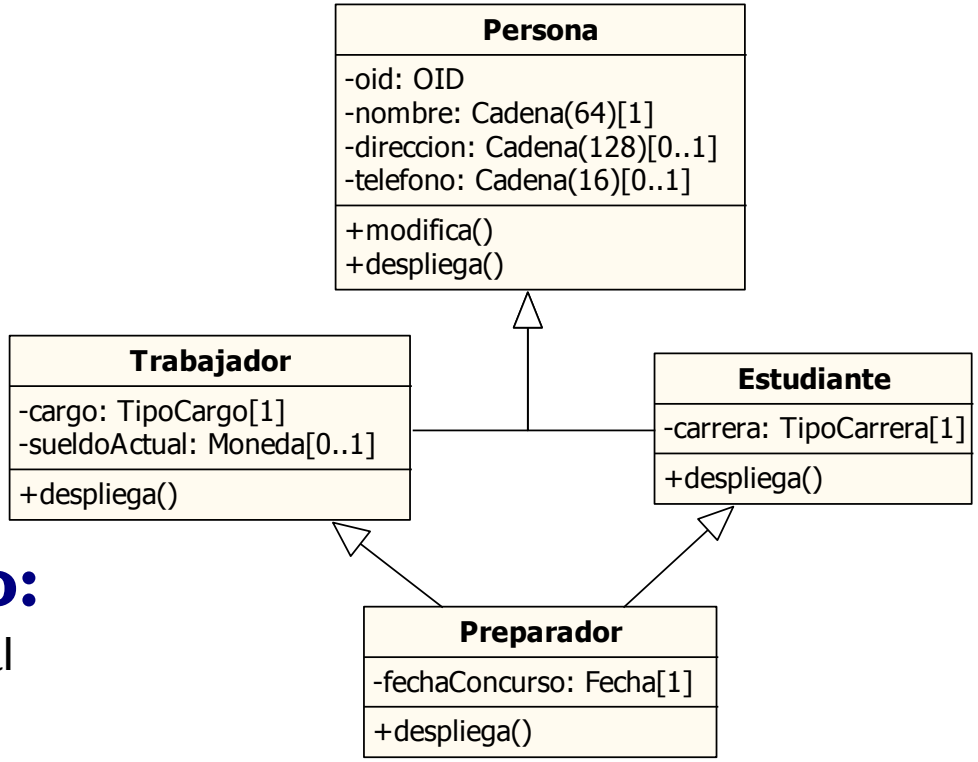


Diagrama de clases

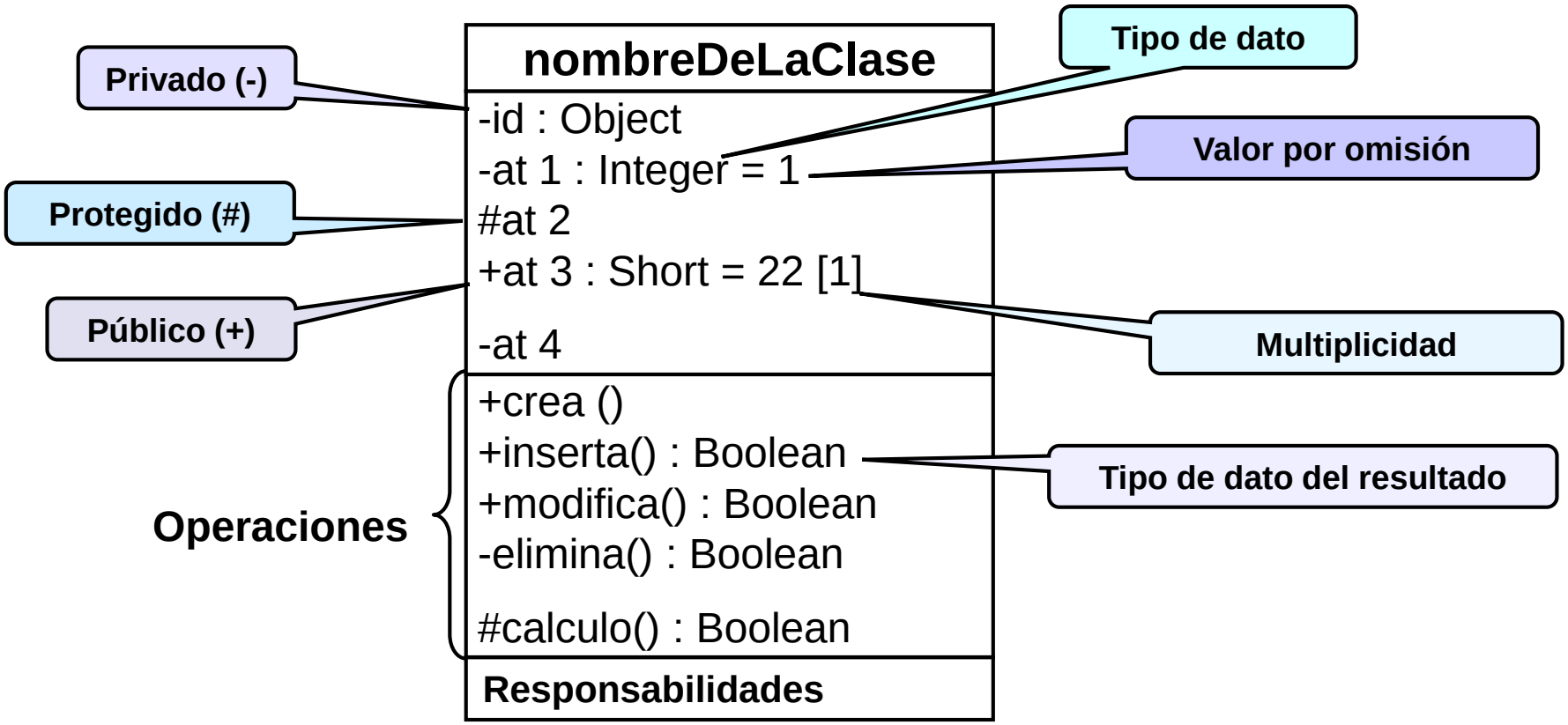


Diagrama de clases formato para los atributos

- **[visibilidad] [/] nombre [:tipo] [multiplicidad] [=valor por omisión] [{propiedad}]**
  - visibilidad: ~ en el paquete, + público, # protegido, - privado
  - /: el valor del atributo puede ser derivado de los valores de 1 o más atributos
  - tipo: es el tipo de dato del atributo
  - multiplicidad:
    - Un valor fijo: 1, 4, 6
    - Muchos: *
    - Rango de valores, inferior..superior: 1..3, 3..*
    - Un conjunto de valores: [1, 3, 5, 7]
  - valor por omisión: cualquier valor inicial para el atributo
  - propiedad:
    - No cambia el valor del atributo: ***readOnly***

Diagrama de clases formato para las operaciones

- **[visibilidad] nombre [(lista de parámetros)] [{propiedad}]**
  - **visibilidad:** ~ en el paquete, + público, # protegido, - privado
  - **lista de parámetros** de una operación separados por comas, cada uno con
    [dirección] nombre: tipo [multiplicidad] [=valor por omisión]
    - **dirección:**
      - Solo entrada: ***in***
      - Solo salida: ***out***
      - Entrada y salida: ***inout***
    - **multiplicidad:** igual que para los atributos
    - **valor por omisión:** igual que para los atributos
  - **propiedad:**
    - No cambia el valor de los atributos: ***isQuery***
    - Cambia el valor de algunos o todos los atributos:
      - Ejecución de las invocaciones en forma concurrente: ***concurrent***
      - Varias invocaciones pero ejecución secuencial: ***guarded***
      - Una invocación a la vez: ***sequential***

Diagrama de clases

- **Condiciones para las operaciones**
  - **Precondición:** aserción que se debe cumplir **antes** de iniciar la ejecución de la operación
  - **Postcondición:** aserción que se debe cumplir **al finalizar** la ejecución de la operación

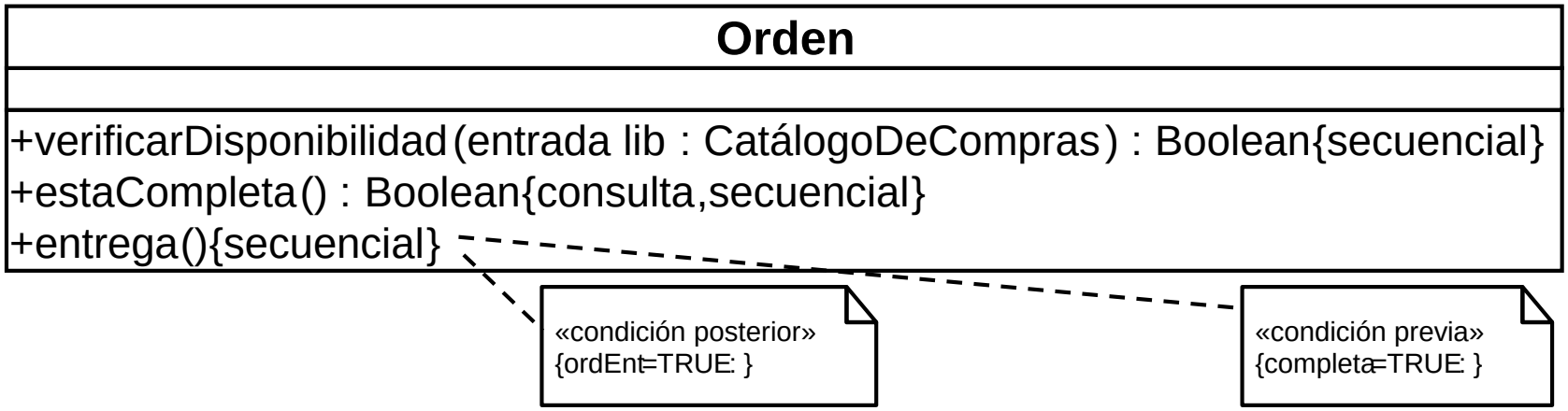


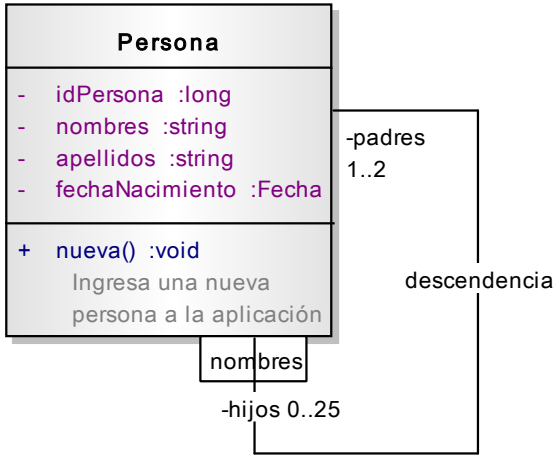
Diagrama de clases

- **Navegación:**
  - **La ausencia de flecha indica navegación en los dos sentidos**
  - **Sintaxis:**
    - destino ::= conjunto '.' selector

destino: es un conjunto de valores u objetos

selector: nombre de atributo, nombre de asociación o nombre de función

- destino ::= conjunto '.' '~' selector
- destino ::= conjunto '[' expresiónLógica ']'
- destino ::= conjunto '.' selector '[' valorDeClave ']'



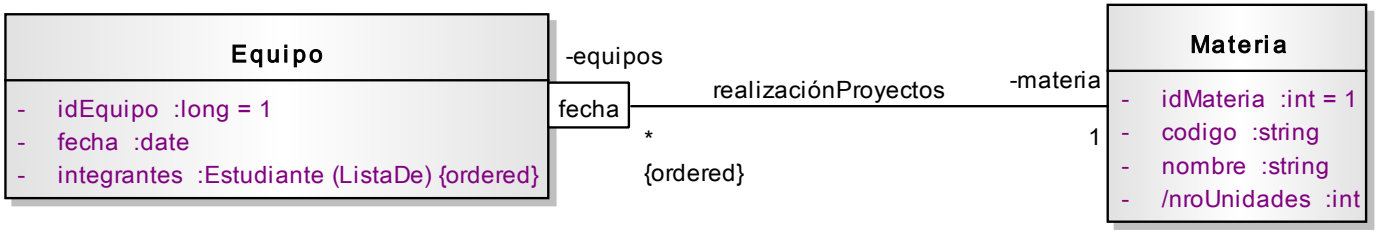


Diagrama de clases

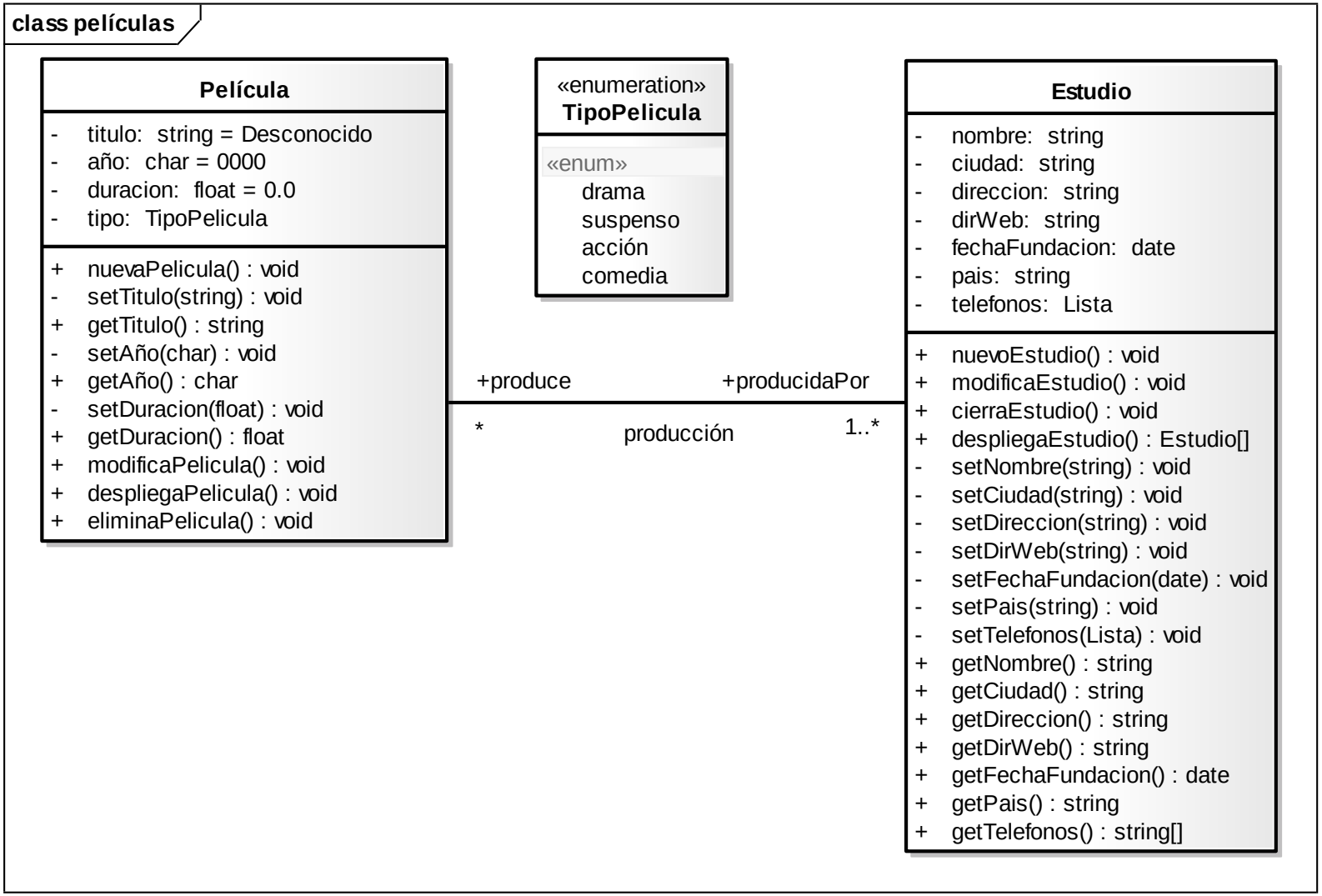


Diagrama de clases

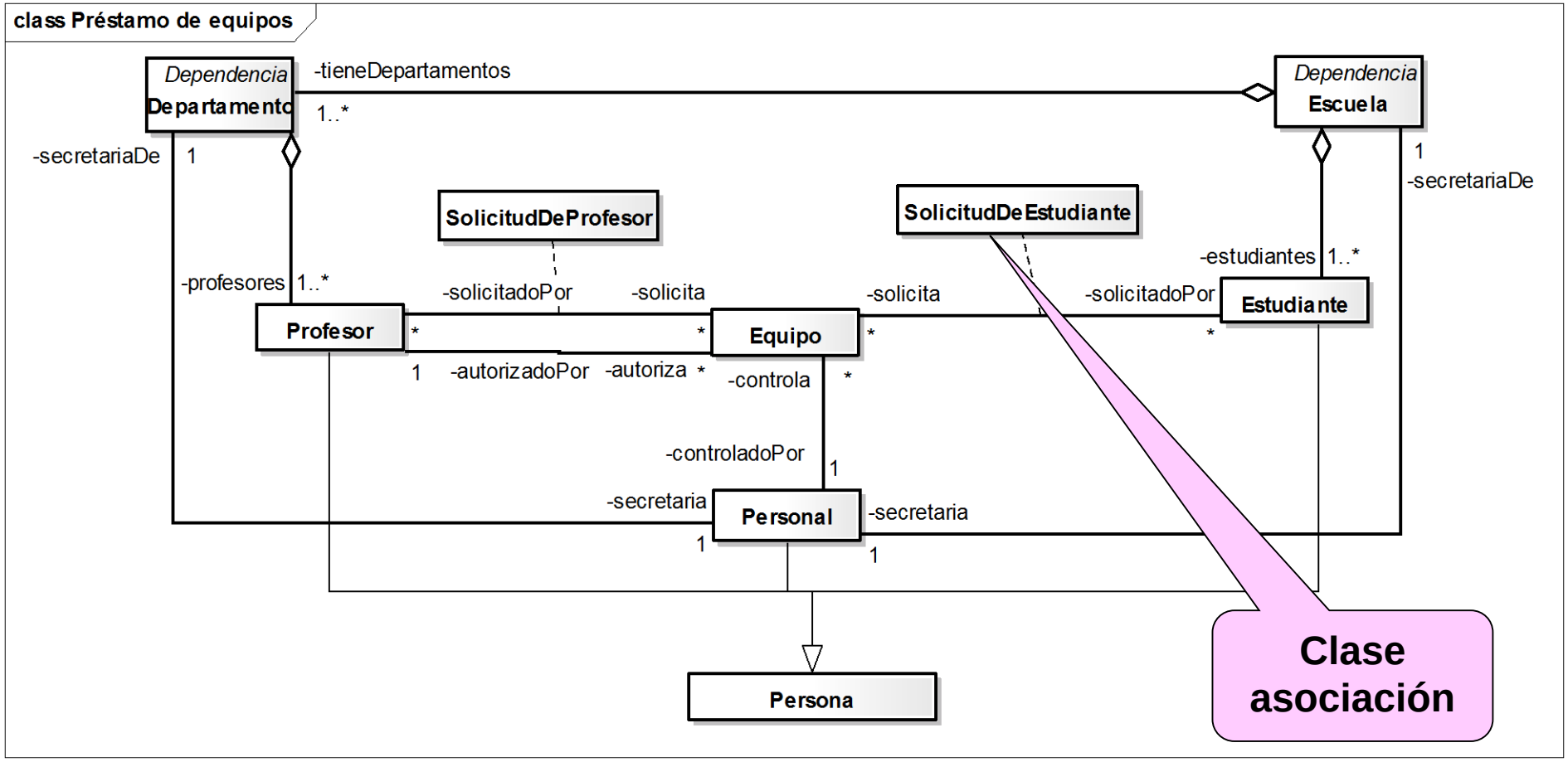


Diagrama de clases

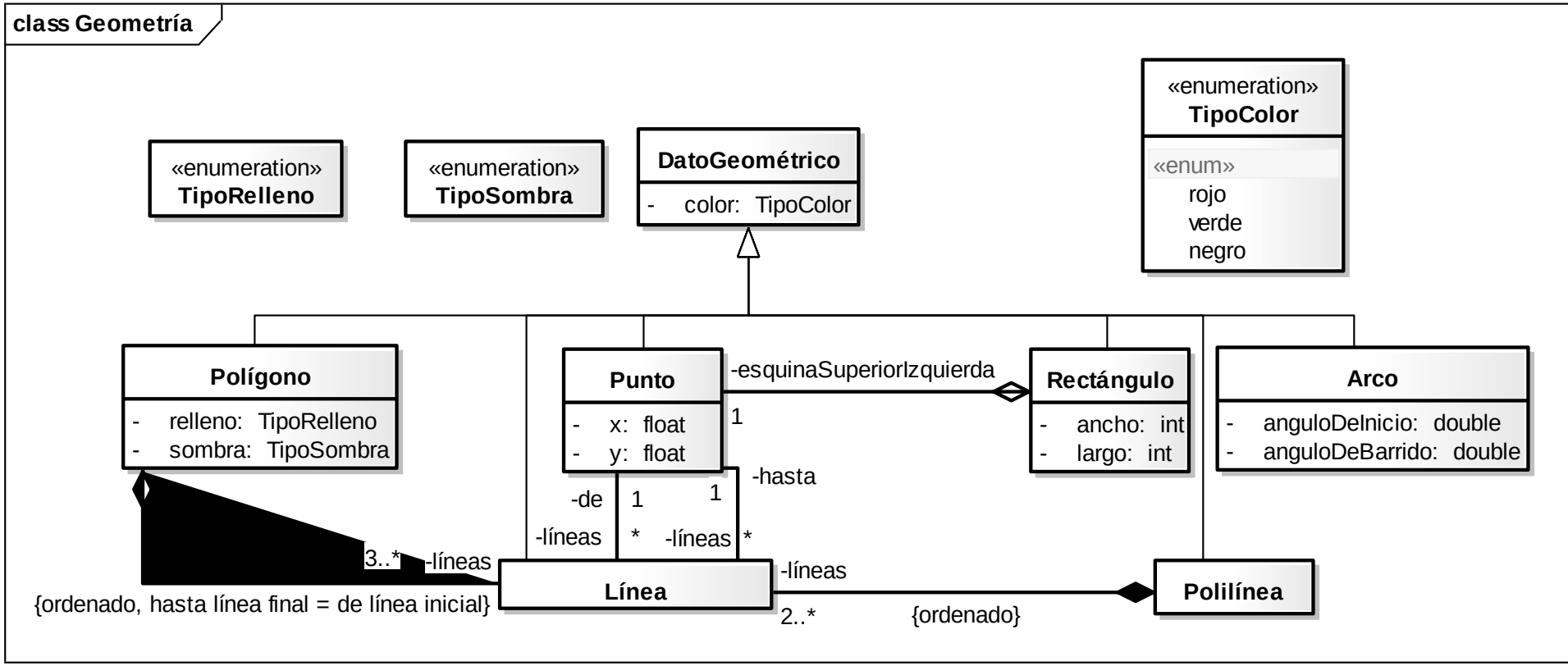
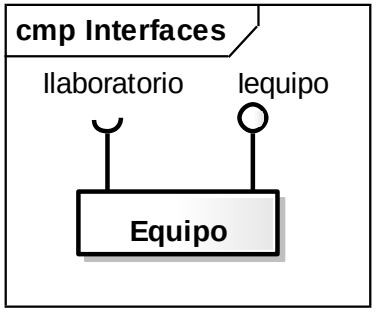
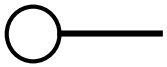
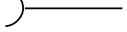


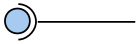
Diagrama de clases

- **Interfaces:**
  - Colección de operaciones que representan los servicios ofrecidos por una clase o componente
  - Visibilidad pública
  - Tipos:
    - Provistas: interfaz dada y lista para usarse
      - *Lollipop*: círculo asociado a la clase
      - Clase con el estereotipo <<interface>> o <<interfaz>>
    - Requeridas: interfaz que la clase necesita para cumplir con sus responsabilidades
    - Provistas/requeridas: combinación de las anteriores









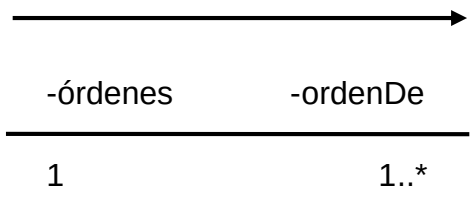
UNIVERSIDAD DE LOS ANDES

Diagrama de clases

- **Representa la estructura estática en términos de clases y asociaciones, las cuales pueden o no ir en paquetes y/o tener estereotipos (<<señal>>, <<interfaz>>, <<metaclase>> y <<utilidad>>)**
- **Realización:**
  - *Listar los conceptos candidatos relacionados con los requerimientos considerados*
  - *Dibujar el diagrama con ellos*
  - *Anexar las asociaciones necesarias que se detecten*
  - *Anexar los atributos necesarios para cumplir los requisitos*
- **Se recomienda hacer esto con el espíritu de un cartógrafo, es decir:**
  - *Usar solamente los nombres existentes en el territorio*
  - *Excluir las cosas irrelevantes*
  - *No anexar cosas que **no** están allí*

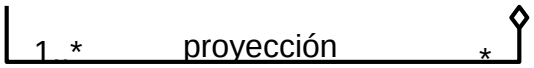
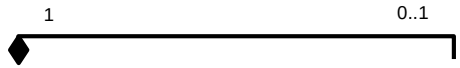
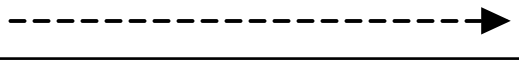
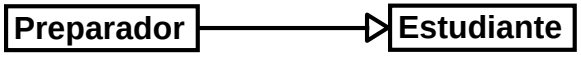
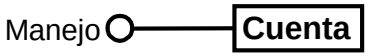
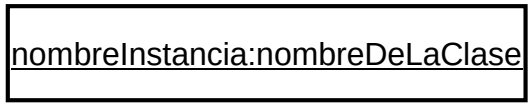
UNIVERSIDAD DE LOS ANDES

Diagrama de clases

Nombre	Notación	Descripción
Clase	**nombreDeLaClase**	Clase en su representación simple
Clase	**nombreDeLaClase** -id : Object -at 1 : Integer = 1 #at 2 +at 3 : Short = 22 -at 4	Clase con atributos y visibilidad
Clase	**nombreDeLaClase** -id : Object -at 1 : Integer = 1 #at 2 +at 3 : Short = 22 -at 4 +crea () +inserta() : Boolean +modifica() : Boolean -elimina() : Boolean #calculo() : Boolean	Clase con atributos, operaciones y visibilidad
Asociación		Asociación bidireccional entre clases con roles y multiplicidad

UNIVERSIDAD DE LOS ANDES

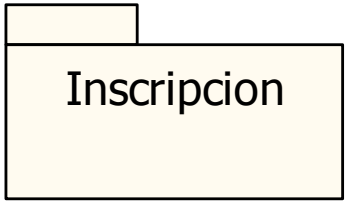
Diagrama de clase

Nombre	Notación	Descripción
Agregación		Relación de subordinación
Composición		Relación es-parte-de
Dependencia		Relación de dependencia
Generalización/ Especialización		Relación Es-un(a)
Clase con estereotipo		Clase con semántica específica
Clase interfaz		Operaciones públicas de una clase
Clase abstracta		Clase sin instancias
Clase activa		Clase que representa un proceso
Instancia de una clase		Indicación de una instancia de una clase

UNIVERSIDAD DE LOS ANDES

Diagrama de paquetes

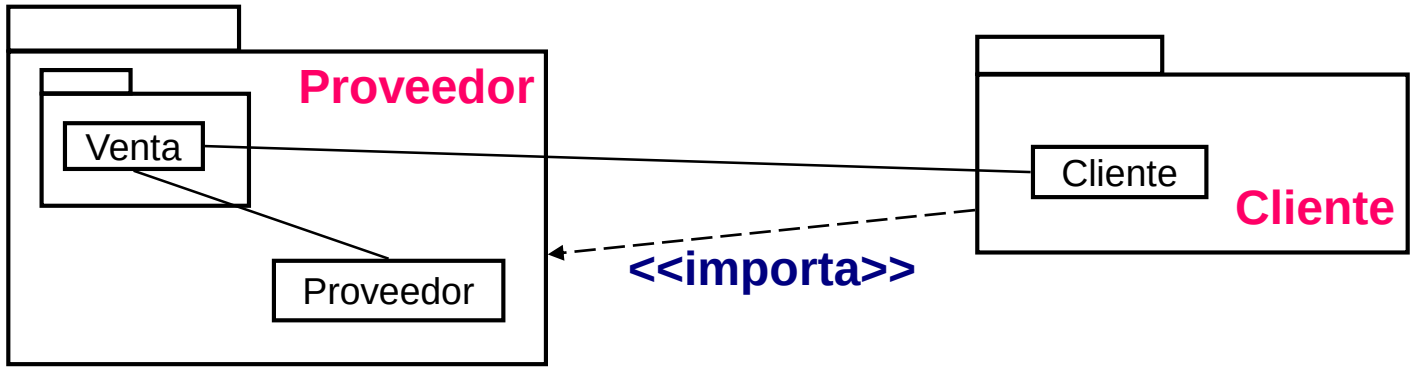
- **Un Paquete es un mecanismo de agrupamiento usado para:**
  - organizar los elementos modelados en UML
  - facilitar el manejo de los modelos de un sistema



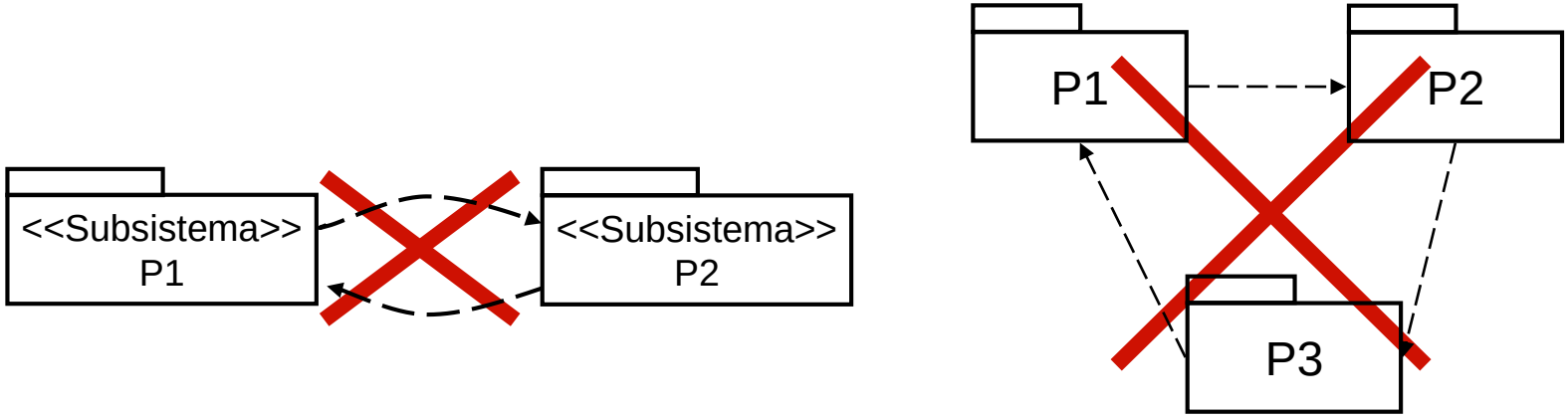
- **Define un espacio de nombres. Dos paquetes diferentes pueden contener elementos con el mismo nombre**
- **Puede contener otros paquetes y elementos de modelado, sin límite de anidamiento**
- **Permiten dividir un modelo y reagrupar y encapsular los elementos de modelado**
- **Poseen una interfaz y una realización**

Modelado estructural

Diagrama de paquetes



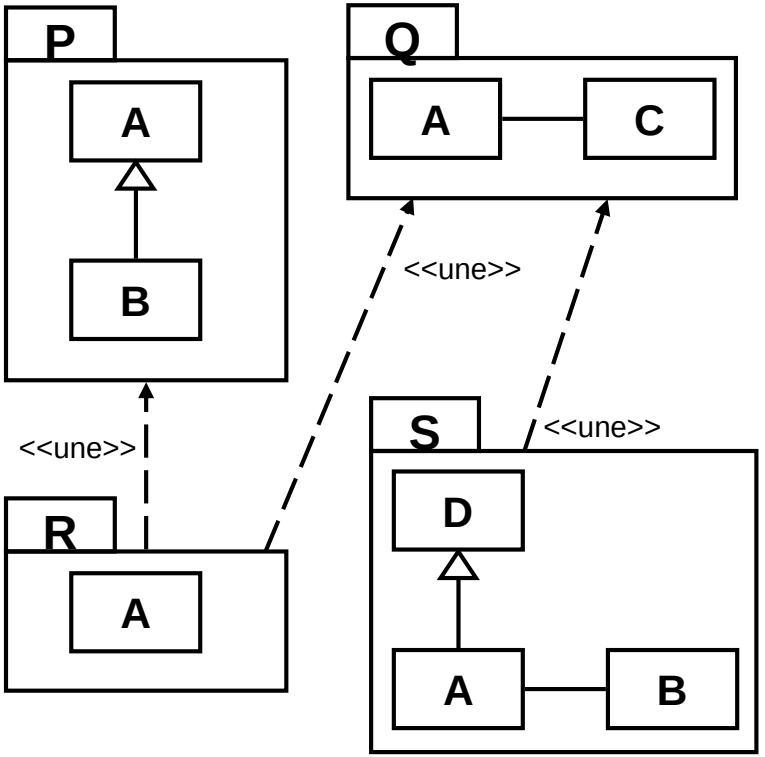
Cada elemento indica si es visible o no (<<import>> o <<access>>)



dependencias circulares

Mezcla de paquetes

Operación de mezcla



[OMG, 2003a]

Resultado de la mezcla

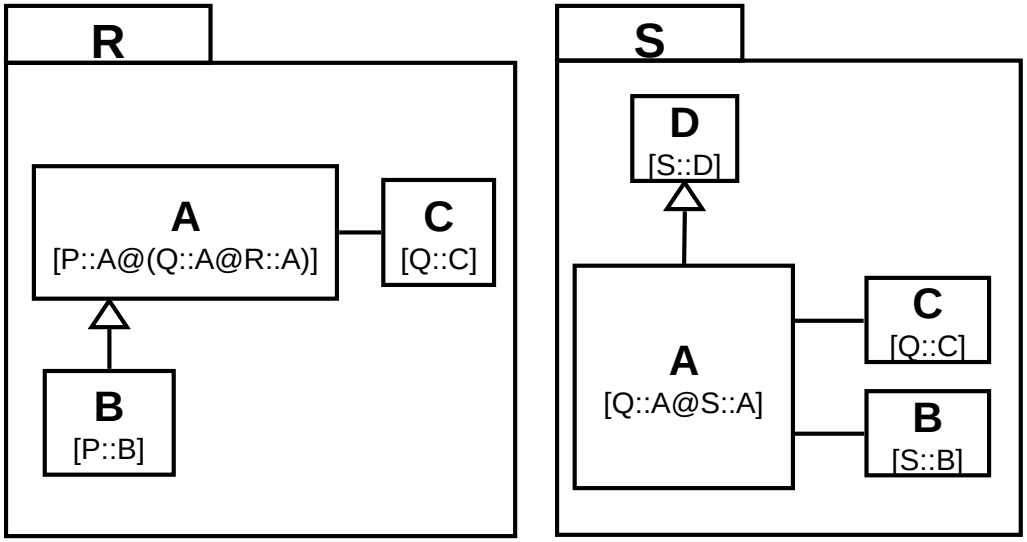


Diagrama de paquetes

- **Arquitectura del sistema:**
  - se expresa con la jerarquía de paquetes y su red de relaciones de dependencia
- **El paquete de más alto nivel es el raíz**
- **Una clase contenida en un paquete puede también aparecer en otro, bajo la forma de un elemento importado**

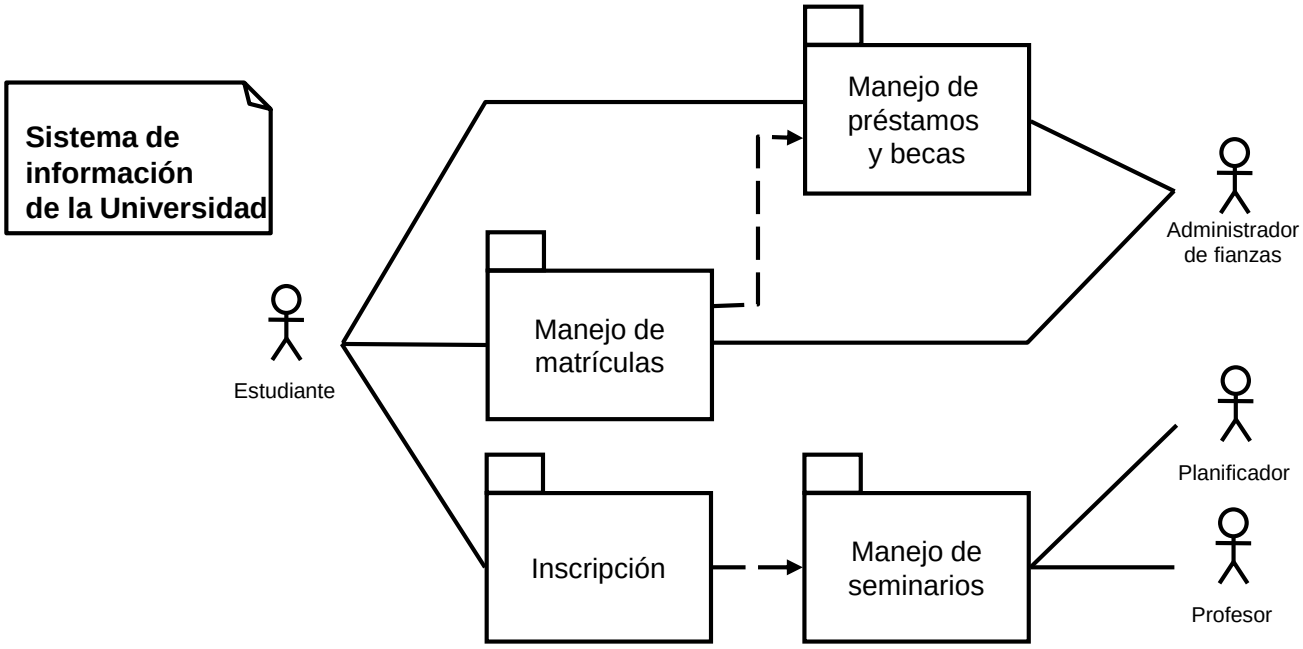
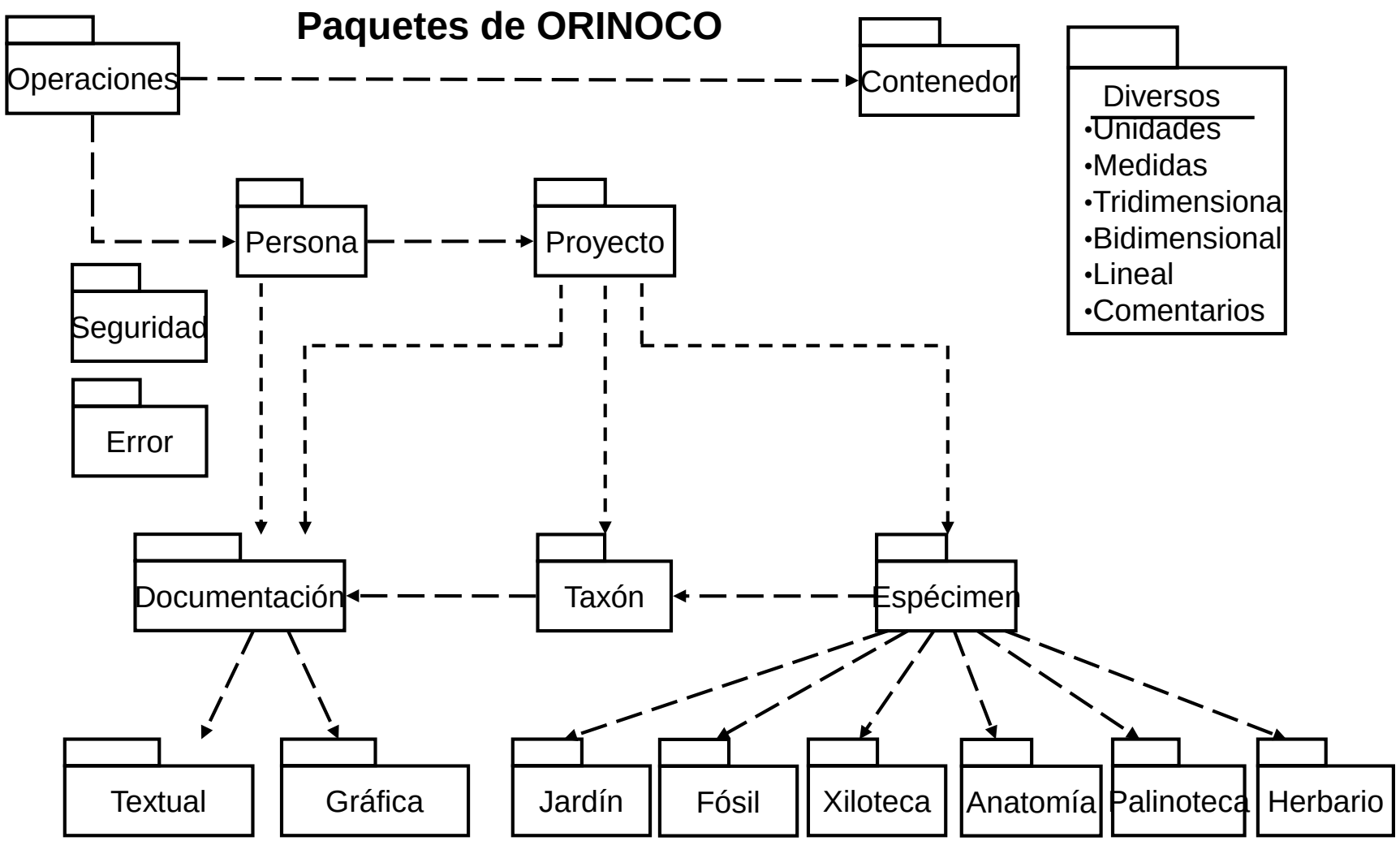
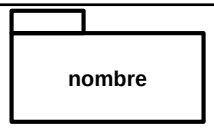
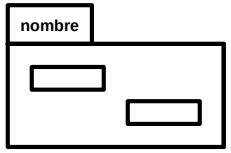


Diagrama de paquetes



UNIVERSIDAD DE LOS ANDES

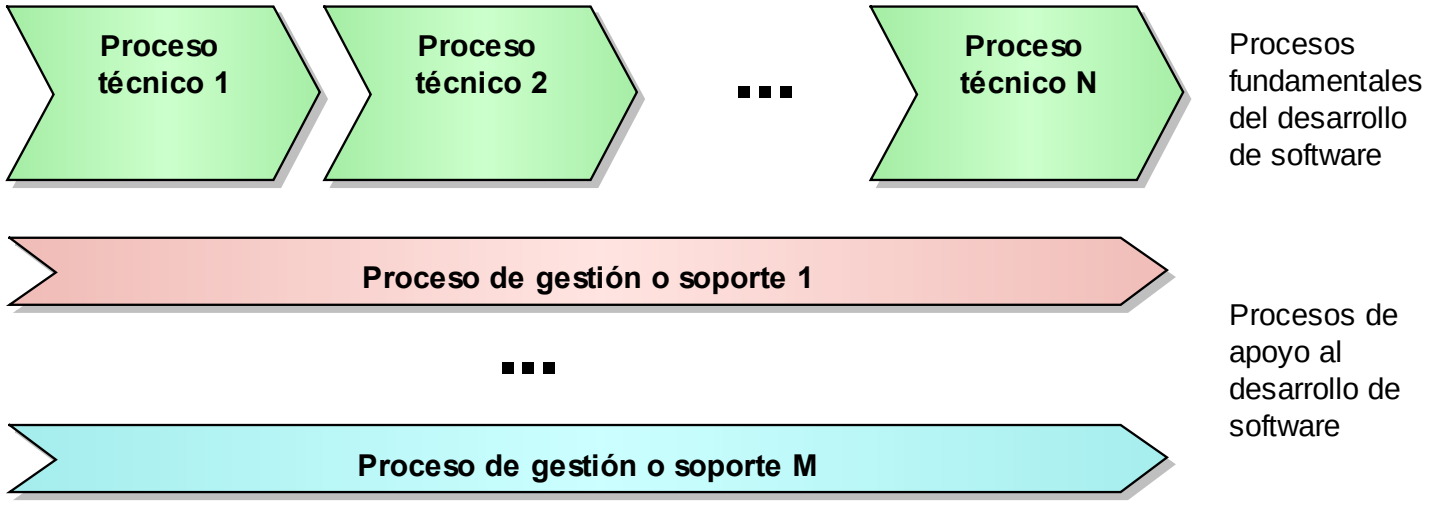
Diagrama de paquetes

Nombre	Notación	Descripción
Paquete		Paquete sin el contenido mostrado explícitamente
		Paquete que muestra sus contenido
Relaciones entre paquetes	<<importar>> 	Usado para indicar que un paquete importa otro
	<<accesar>> 	Usado para indicar que un paquete tiene acceso a otro
	<<unir>> 	Mezcla el contenido de un paquete con el de otro
		Usado para indicar la composición de paquetes

analysis Cadena de valor

Diagrama de la Extensión UML Business - Cadena de Valor

Identifica los procesos técnicos, de gestión y de soporte del método, al más alto nivel de abstracción, y los clasifica en procesos fundamentales (procesos técnicos) y procesos de apoyo (procesos de gestión y soporte).



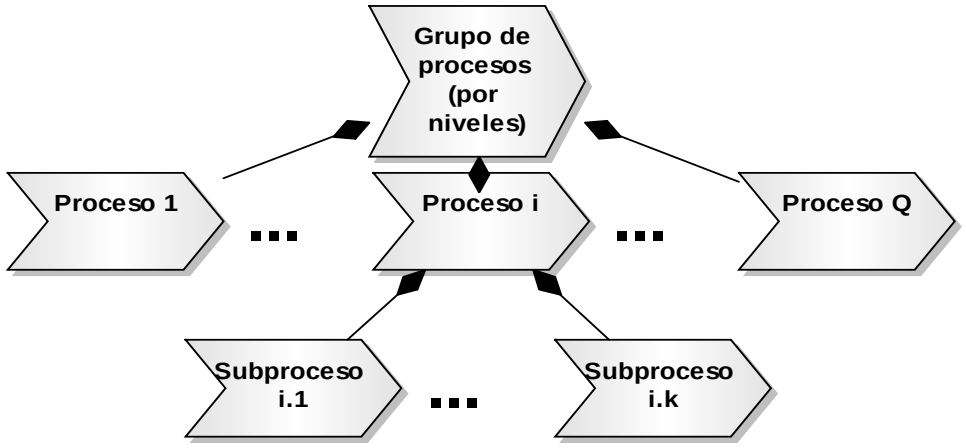
Modelado del negocio

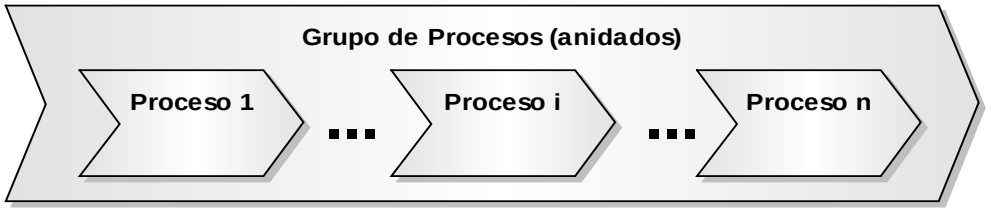
analysis Diagrama de jerarquía de procesos

Extensión UML Business
Diagrama de Jerarquía de Procesos

Usados para mostrar la descomposición de un proceso en subprocesos.
Los niveles de descomposición dependen de la complejidad y extensión del proceso que se descompone.
Se pueden usar dos maneras para diagramar la descomposición jerárquica de un proceso:

- Por niveles (de arriba hacia abajo)
- Anidada



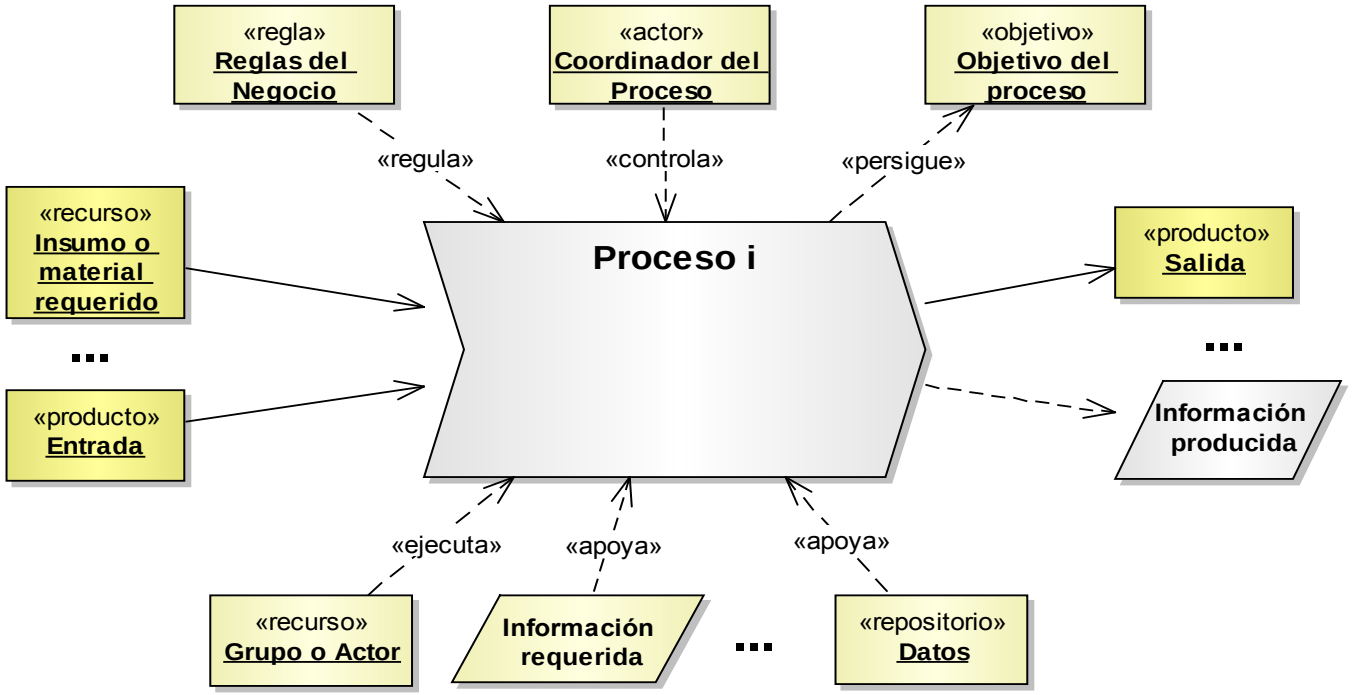


Modelado del negocio

analysis Diagrama del proceso

Diagrama de la Extensión UML Business - Diagrama del Proceso

Describen los objetivos, entradas, salidas, controles, reglas de negocio y recursos que un proceso utiliza durante su ejecución. Se usan para describir cualquiera de los procesos de una cadena de valor o de un diagrama de jerarquía de procesos.



UNIVERSIDAD DE LOS ANDES

Autoevaluación

1. ¿Qué es un lenguaje de modelado y qué elementos lo componen?
2. ¿Cuáles son los diagramas de UML 2?
3. ¿Cuáles son los elementos y mecanismos comunes de UML?
4. ¿Qué es una diagrama de casos de uso y cuáles son sus elementos de modelado?
5. ¿Qué es un diagrama de clases y cuáles son sus elementos de modelado?
6. ¿Qué es un diagrama de paquetes y cuáles son sus elementos de modelado?
7. ¿Qué relación existe entre el modelado estructural y el de comportamiento?
8. ¿Cuáles son los diagramas para modelado del negocio en UML Business?

UNIVERSIDAD DE LOS ANDES

Ejercicios

Realice para la descripción dada:

1. Modelado del negocio
2. Al menos dos diagramas de casos de uso
3. El diagrama de clases

1. **La base de datos debe contener información sobre los pacientes, los médicos y los exámenes realizados a los pacientes. Los pacientes tienen número de historia, nombre, dirección, teléfono, y fecha de nacimiento. Los médicos tienen nombre, dirección, teléfono, especialidad y número del colegio de médicos. Los exámenes tienen número de examen, tipo, fecha de realización, resultado y fecha de entrega. Los exámenes son realizados a los pacientes por solicitud del médico que tratan al paciente. Los pacientes acuden al médico previa cita. Durante la cita, el médico anota en la historia del paciente el motivo de la consulta, la fecha, el diagnóstico, el tratamiento y los exámenes que le solicitó. Un paciente sólo consulta a un médico de la clínica, pero puede hacerlo en varias consultas. Un médico puede solicitar el mismo examen a un paciente, pero en fechas diferentes**

UNIVERSIDAD DE LOS ANDES

Ejercicios

2. **La base de datos debe contener información sobre los libros, revistas, manuales, carpetas, fotocopias, mapas y DVDs de una librería. Cada libro tiene ISBN, título, autores, editorial, año de publicación, costo y precio de venta. Las revistas tienen ISBN, título, editorial, número, año, costo y precio de venta. Los manuales tienen ISBN, título, autor, editorial, año, costo y precio de venta. Las carpetas tienen tipo, tamaño, color, costo y precio de venta. Las fotocopias tienen tipo de papel y precio por copia. Los mapas tienen código, título, escala, tamaño, costo y precio de venta. Los DVD tienen código, título, autores, productora, distribuidora, año, costo y precio de venta. Cada uno de los productos se venden a los clientes generando una factura que tiene un número, fecha, nombre, dirección, teléfono y cédula de identidad o RIF del cliente, descripción y cantidad de cada producto vendido, el monto total, el IVA y el monto a pagar.**

UNIVERSIDAD DE LOS ANDES

Ejercicios

3. **Los organizadores del mundial de Futbol desean tener un sistema de BD para registrar los datos del evento. Se deben almacenar los partidos jugados, los resultados de cada partido, los jugadores de cada partido y las estadísticas individuales de cada jugador en cada partido. Cada jugador tiene número, nombre, apellido, edad y nacionalidad. Cada partido tiene un identificador, día y hora del encuentro, lugar donde se celebra, equipos que se enfrentan, árbitros que lo regulan, alineación de cada equipo, resultado del partido con los detalles del resultado (jugador que anotó y en que tiempo lo hizo). Los árbitros tienen nombre, apellido, edad, nacionalidad y nivel de experticia. Entre las estadísticas se tienen: por cada equipo los goles a favor y en contra, el número de tarjetas amarillas y rojas, el jugador con mayor número de goles por cada equipo y en el mundial, etc. Realice cualquier consideración que crea conveniente en forma razonada.**