

# **Unified Modeling Language (UML)**

# Unified Modeling Language (UML)

- Introducción
- Toma de requisitos
  - Historias de usuario
  - UML: Casos de uso
- UML: Diagramas de clases
- UML: Diagramas de comportamiento
  - Actividad
  - Secuencia

# Introducción a UML

- Análogo a los planos de una casa
- Necesidad de mostrar de manera rápida y comprensible la construcción de un software
- Utilizado en fase de análisis y diseño
  - Antes de escribir el código
- UML está dirigido a la POO

# Introducción a UML

- Lenguaje Unificado de Modelado
  - Lenguaje gráfico para visualizar, especificar, construir y documentar un sistema
- Convertido en estándar de la industria del software
- Útil como documentación

# Introducción a UML

- Cuanto más complejo sea un sistema más necesario es el uso de los diagramas que lo describen
  - Visión global:
    - Detectar dependencias entre diferentes partes del sistema
    - Iterar sobre el análisis y el propio diseño, redifiniendo y detallando el comportamiento
    - Cuanto antes se detecten problemas, más barato será corregirlo

# Unified Modeling Language (UML)

- Introducción
- **Toma de requisitos**
  - Historias de usuario
  - UML: Casos de uso
- UML: Diagramas de clases
- UML: Diagramas de comportamiento
  - Actividad
  - Comportamiento

# Toma de requisitos

- Coste de los errores (James Martin)
  - Más del 50% surgen en los requisitos
    - No entender lo que quiere el cliente
    - El cliente no sabe lo que quiere
    - Requisitos mal documentados

# Toma de requisitos

- Coste de los errores (James Martin)
  - Más del 50% surgen en los requisitos



# Toma de requisitos

- Coste de los errores (James Martin)
  - Un error detectado comparado si se detecta en la fase de toma de requisitos:
    - Al final del proyecto: 100 veces más caro
    - En producción: 1000 veces más caro
- De ahí la importancia del detalle en la definición de los requisitos y en la iteración sobre ellos.

# Toma de requisitos

- Coste de los errores
  - Requisitos → 1 – 2
  - Diseño → 5
  - Codificación → 10
  - Pruebas unitarias → 20
  - Pruebas sistema → 50-100
  - En producción → 1000

# Toma de requisitos

- Todo problema consiste en configurar una máquina M para que ejerza unos efectos R en un dominio D
  - R: requisitos
  - El dominio D es el contexto.
  - La máquina M es el software en su conjunto

# Toma de requisitos

- Requisitos: desde la perspectiva del sistema y no en la interacción del usuario; características en estado puro.
- Historias de Usuario: describen lo que el usuario desea ser capaz de hacer. Además, las historias de los usuarios se centran en el valor
- Casos de Uso (UML): interacciones entre el usuario y el sistema. Hacen hincapié en el contexto orientado al usuario

# Requisitos

- Los requisitos deben contener:
  - Información acerca del problema
  - Propiedades y comportamiento del sistema
  - Restricciones de diseño y fabricación del producto
- Hay requisitos
  - Funcionales: lo que debe realizar el sw
  - No funcionales: rendimiento, disponibilidad, seguridad, ...

# Requisitos

- Requisitos Funcionales
  - Concretos
  - Cortos
  - Completos
  - A veces, es conveniente definir lo que NO debe hacer el programa
  - Relacionados con el comportamiento externo y en lenguaje natural
  - Priorizables

# Requisitos

- Definición de requisitos
  - MAL
    - El sistema será lo más fácil de utilizar posible.
    - El sistema proporcionará una respuesta rápida al usuario.
    - El sistema se recuperará automáticamente tras producirse un fallo.
  - BIEN
    - Cuando haya hasta 100 usuarios accediendo simultáneamente al sistema, su tiempo de respuesta no será en ningún momento superior a 2 segundos.

# Unified Modeling Language (UML)

- Introducción
- Toma de requisitos
  - Historias de usuario
  - UML: Casos de uso
- UML: Diagramas de clases
- UML: Diagramas de comportamiento
  - Actividad
  - Secuencia

# Historia de usuarios

- Historia de usuarios
  - Representación de un requisito escrito en una o dos frases utilizando el lenguaje común del usuario.
  - Utilizadas en las metodologías ágiles, discutidas con los usuarios y con pruebas de validación.
  - Cada historia de usuario debe ser limitada, pudiéndose escribir sobre una nota adhesiva pequeña (post-it)
  - Dentro de la metodología XP las historias de usuario deben ser escritas por los usuarios.

# Historia de usuarios

- Historia de usuarios
  - Forma rápida de administrar los requisitos de los usuarios sin elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos.
  - Permiten responder rápidamente a los requisitos cambiantes.

# Historia de usuarios

- Historia de usuarios (INVEST)
  - Independientes unas de otras
  - Negociables
  - Valoradas por los clientes o usuarios
  - Estimables
  - Pequeñas (Small)
  - Verificables (Testable)

# Historia de usuarios

- Historia de usuarios
  - Debe responder a tres preguntas:
    - ¿Quién se beneficia?,
    - ¿qué se quiere? y
    - ¿cuál es el beneficio?
  - Como (rol) quiero (algo) para poder (beneficio).
    - Como usuario quiero subir fotos para publicarlas
    - Como administrador quiero dar de alta usuarios

# Toma de requisitos

- Historia de usuarios. Beneficios
  - Representa requisitos del modelo de negocio que pueden implementarse en días/semanas
  - Necesitan poco mantenimiento
  - Mantiene una relación cercana con el cliente
  - Divisar los proyectos en pequeñas entregas
  - Estimar fácilmente el esfuerzo de desarrollo
  - Es ideal para proyectos con requisitos volátiles o no muy claros

# Toma de requisitos

- Ejercicio: Instagram (wikipedia)

- Instagram es una red social y aplicación. Su función es subir fotos y vídeos. Está disponible para dispositivos Android y iOS. Sus usuarios también pueden aplicar efectos fotográficos como filtros, marcos, similitudes térmicas, áreas subyacentes en las bases cóncavas, colores retro y posteriormente compartir las fotografías en la misma red social o en otras como Facebook, Tumblr, Flickr y Twitter. (...). Hoy en día, las fotos pueden estar en horizontal y en vertical sin el uso de los bordes blancos, aunque estas son recortadas parcialmente. También hay un medio de comunicación privado para hablar llamado Instagram Direct. Actualmente se han creado una nueva función llamada Stories donde todas las personas pueden colgar fotografías y vídeos a su perfil con una duración máxima de 24 horas, la cual todos sus seguidores pueden visualizar. Dicha función solo pueden tener las fotos y vídeos en formas rectangulares debido a la forma del diseño de las Stories, que se adapta a la forma del dispositivo móvil.
- Actualmente, Instagram cuenta con más de 900 millones de usuarios activos. La cuenta oficial de Instagram cuenta con más de 341 millones de seguidores.
- Extrae las historias de usuario

# Unified Modeling Language (UML)

- Introducción
- Toma de requisitos
  - Historias de usuario
  - **UML: Casos de uso**
- UML: Diagramas de clases
- UML: Diagramas de comportamiento
  - Actividad
  - Secuencia

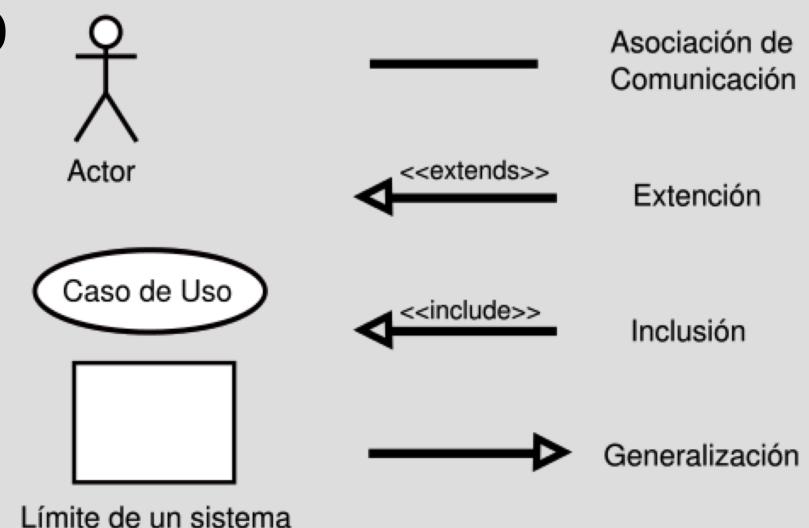
# UML: Casos de uso

- Un caso de uso es la descripción de una acción o actividad
  - Se utilizan diagramas en los que aparecen:
    - Actores: personajes o entidades
    - Procesos o sistemas
- Un diagrama de caso de uso (UC) especifica la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas

# UML: Casos de uso

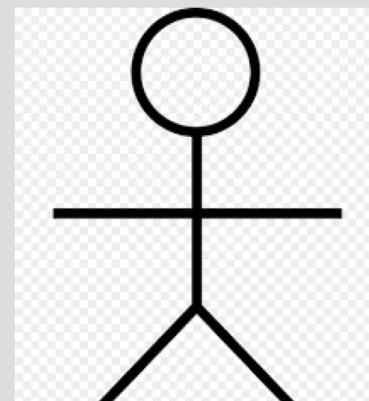
- Se utilizan para ilustrar los requisitos del sistema al mostrar cómo reacciona a eventos producidos en su ámbito o en él mismo

- Una relación es una conexión entre los elementos del modelo
  - especialización y
  - generalización



# UML: Casos de uso

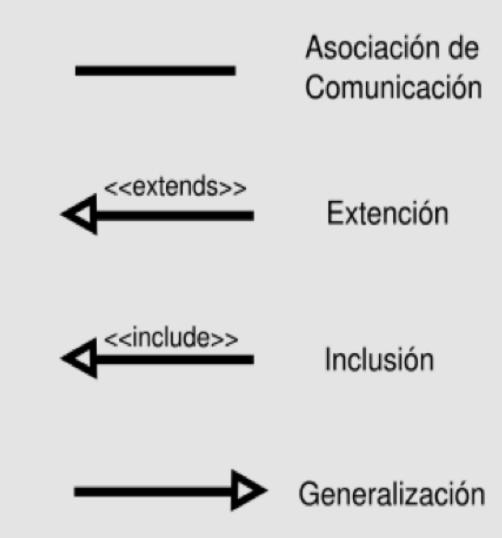
- **Actores:** entidad externa al sistema que guarda una relación con éste y que le demanda una funcionalidad
  - Operadores humanos (usuarios, roles, ...)
  - Entidades abstractas
- Se representa con un monigote



# UML: Casos de uso

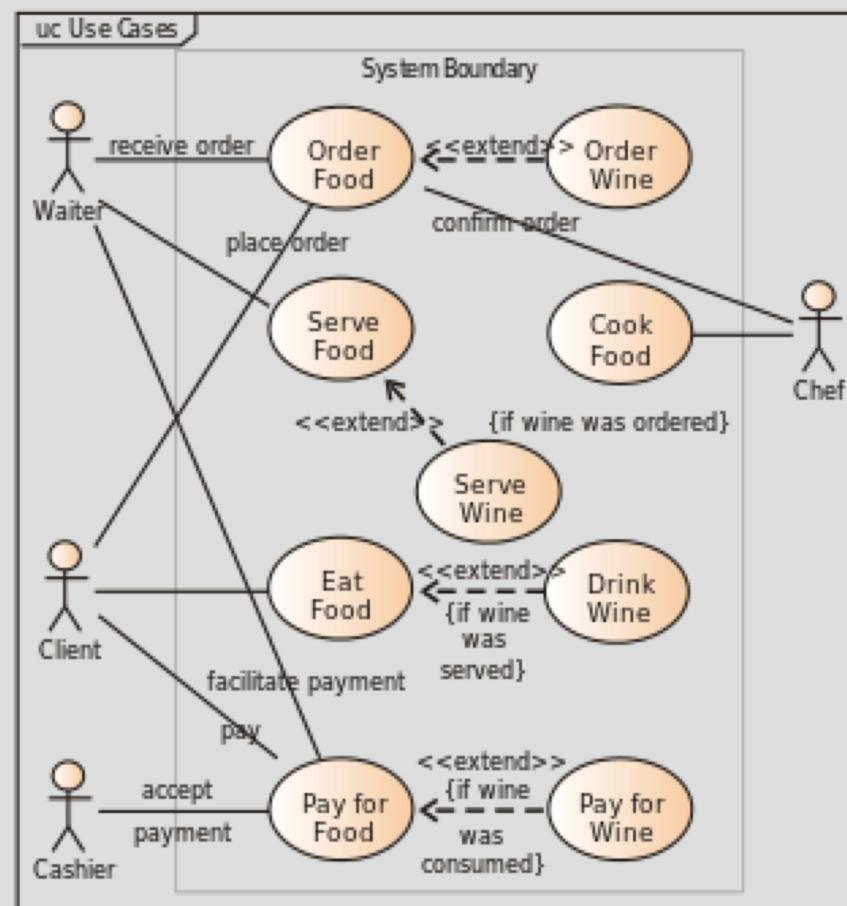
- Relaciones

- **Comunica:** asociación entre un actor y un UC en el que participa
- **Extiende:** dependencia en dos UC, solo se puede alterar una parte del comportamiento
- **Usa (o include):** dependencia entre dos UC donde uno está incluido en el otro
- **Generalización:** un UC es una variante de otro, puede variar cualquier aspecto del UC



# UML: Casos de uso

- Diagrama de caso de uso



# UML: Casos de uso

- Redacción (ejemplo en Wikipedia)
  - Elementos en la descripción de un UC:
    - Nombre del caso de uso
    - Nivel: a partir del nivel 0 se va aumentando el detalle
    - Resumen, descripción
    - Actores
    - Postcondiciones
    - Precondiciones
    - Disparadores (triggers)
    - Flujo básico
    - Alternativos
    - Excepciones

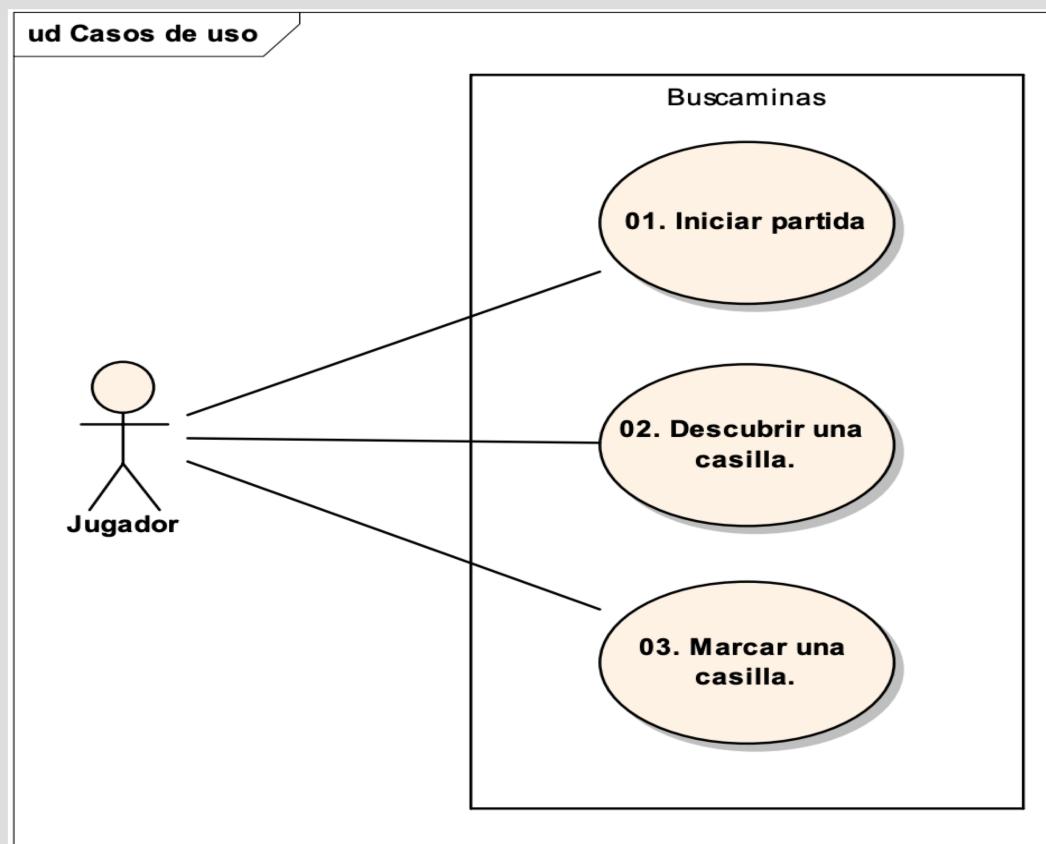
# UML: Casos de uso

- Redacción

<b>Name</b>	Save item for purchase.
<b>ID</b>	UC_001
<b>Description</b>	While browsing items in the eStore, a user finds an item he is not ready to purchase yet, but he wants to save it to a list so that he can later find the item that he was previously interested in.
<b>Actors</b>	eStore customer.
<b>Organizational Benefits</b>	Increase sales by helping the customer remember products he was previously interested in.
<b>Frequency of Use</b>	20% of users save an item to be bought later each time they visit the site. 50% of saved items are purchased within one year of the saved date.
<b>Triggers</b>	The user selects an option to save an item.
<b>Preconditions</b>	User is viewing an item in the catalog.
<b>Postconditions</b>	The item selected to be saved is visible to the user when he views his saved items. The item selected to be saved is reflected as a saved item when the user views his eStore search and browse results.
<b>Main Course</b>	<ol style="list-style-type: none"><li>1. System prompts user to confirm saving selected item instead of purchasing it right away.</li><li>2. User confirms to save now (see EX1).</li><li>3. System determines user is not logged in and redirects user to log on (see AC1).</li><li>4. User logs on (see AC2, AC3).</li><li>5. System stores the saved item (see EX2).</li><li>6. System redirects the user to their saved items list to view the full list.</li></ol>
<b>Alternate Courses</b>	<p>AC1 System determines user is already logged on. 1. Return to Main Course step 5.</p> <p>AC2 User logs off again. 1. Return user to Main Course step 3.</p> <p>AC3 User does not have an account already. 1. User creates an account. 2. System confirms account creation. 3. Return user to Main Course step 4.</p>
<b>Exceptions</b>	<p>EX1 User decides to purchase the item now. 1. See "Purchase item" Use Case.</p> <p>EX2 System fails on saving item to list. 1. System notifies user that an error has occurred. 2. Return user to Main Course step 1.</p>

# UML: Casos de uso

- Ejemplo: Buscaminas



# UML: Casos de uso



- Ejercicio

- Realiza un diagrama de caso de uso general (nivel 0) para la realización de una tortilla de patatas
- Describe el caso de uso con una tabla como las vistas en clase

Nota: para la realización del diagrama de caso de uso puedes utilizar la herramienta draw.io (en web y aplicación de escritorio)

# UML: Casos de uso



- Ejercicio

- Ve detallando aumentando la granularidad del ejercicio anterior (nivel 1 y nivel 2)
- Describe todos los caso de uso con una tabla como las vistas en clase

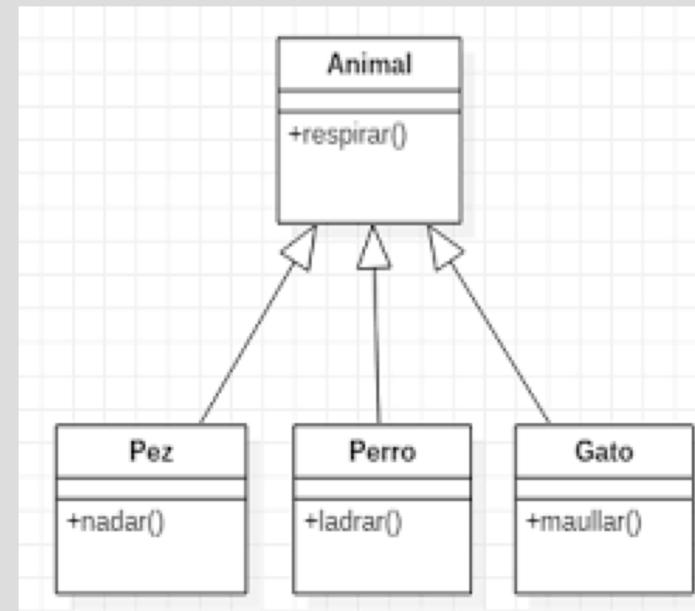
Nota: para la realización del diagrama de caso de uso puedes utilizar la herramienta draw.io (en web y aplicación de escritorio)

# Unified Modeling Language (UML)

- Introducción
- Toma de requisitos
  - Historias de usuario
  - UML: Casos de uso
- **UML: Diagramas de clases**
- UML: Diagramas de comportamiento
  - Actividad
  - Secuencia

# UML: Diagramas de clases

- Dibuja la estructura de un sistema mostrando las clases del sistema, sus atributos, operaciones (o métodos), y las relaciones entre los objetos.



# UML: Diagramas de clases

- Miembros
  - Los miembros de una clase son los atributos de clase y los métodos
  - Visibilidad

+	Público
-	Privado
#	Protegido
  - Para los atributos se debe indicar, además del nombre, su tipo
  - Miembro estático: se subraya su nombre

# UML: Diagramas de clases

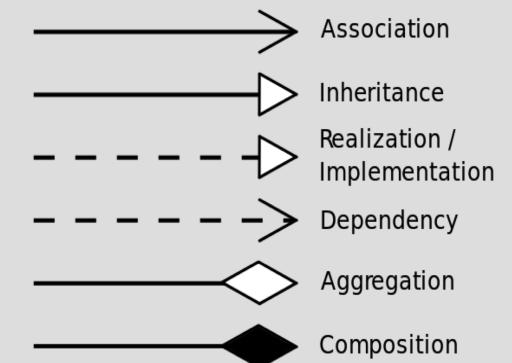
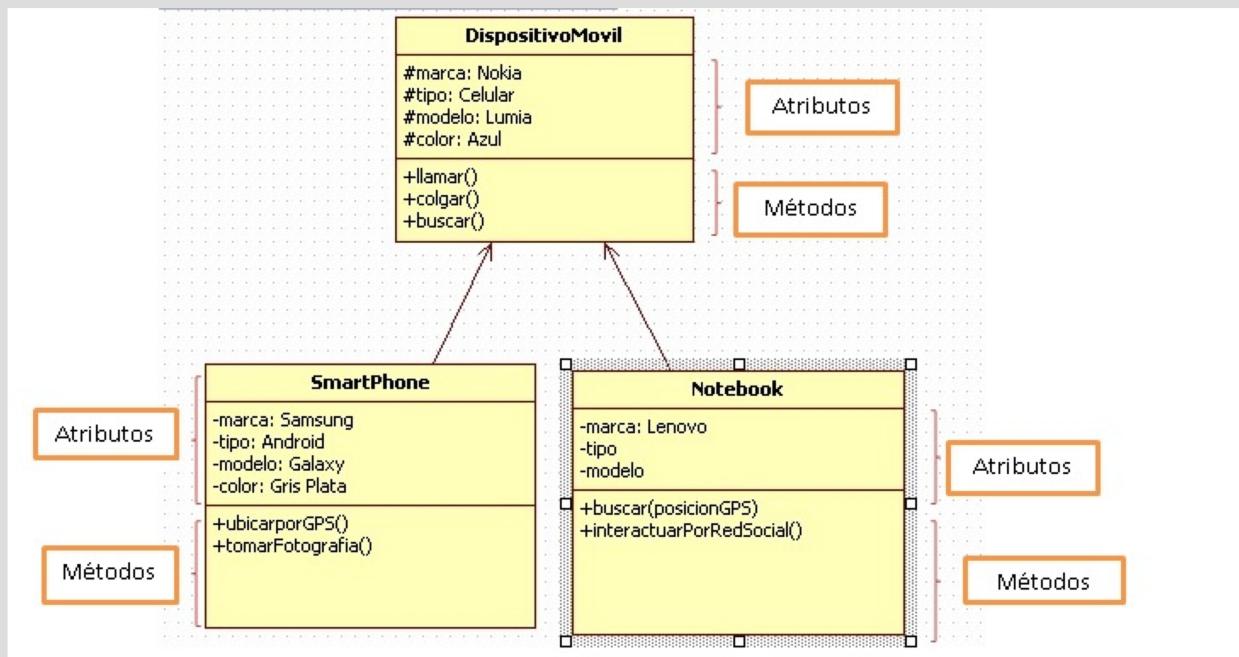
- Multiplicidad
  - Número de objetos que participan en una relación entre clases

<b>0</b>	No instances (rare)
<b>0..1</b>	No instances, or one instance
<b>1</b>	Exactly one instance
<b>1..1</b>	Exactly one instance
<b>0..*</b>	Zero or more instances
<b>*</b>	Zero or more instances
<b>1..*</b>	One or more instances

# UML: Diagramas de clases

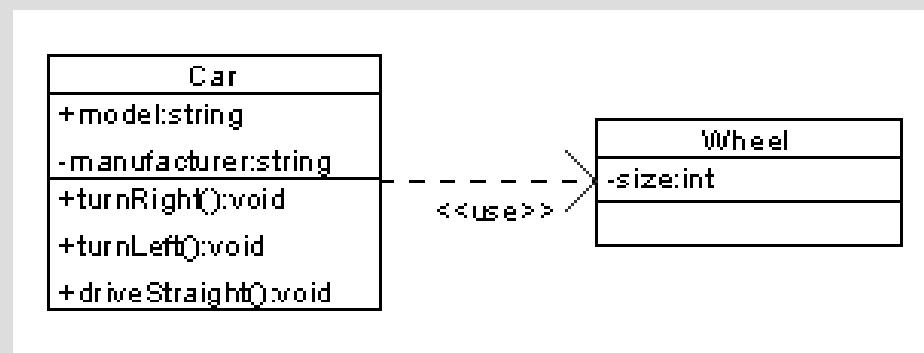
- Relación entre clases:

Se deben mostrar las diferentes relaciones que hay entre las clases de un sistema



# UML: Diagramas de clases

- Relaciones generales: **Dependencia**
  - Forma débil de relación, una clase depende de otra porque la usa en algún momento
  - Se da cuando una clase tiene un atributo de un tipo de otra clase o algún parámetro en sus métodos



# UML: Diagramas de clases

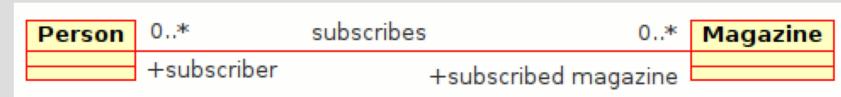
- Relación entre clases:
  - Relación entre instancias
    - Los valores de los atributos pueden ser diferentes para cada las instancias
    - Invocar a un método puede cambiar el estado de la clase
  - Relación entre clases (static)
    - Los valores de los atributos son iguales para todas las instancias
    - Invocar a un método no cambia el estado de la clase

# UML: Diagramas de clases

- Relación entre instancias: **Dependencia**
  - Conexión semántica entre elementos del modelo independientes y dependientes
  - Si se cambia la definición de la clase independiente puede afectar a la clase dependiente (unidireccional)
  - Se dibuja como una línea discontinua con una flecha abierta: de la independiente a la dependiente.

# UML: Diagramas de clases

- Relación entre instancias: **Asociación**
  - Conjunto de enlaces entre clases (binaria, ternaria, ...)
  - Tienen nombre, multiplicidad, visibilidad y otras propiedades
  - Tipos de asociación:
    - bi-direccional
    - uni-direccional
    - aggregación (que incluye composición)
    - reflexiva



# UML: Diagramas de clases

- Relación entre instancias: **Asociación**

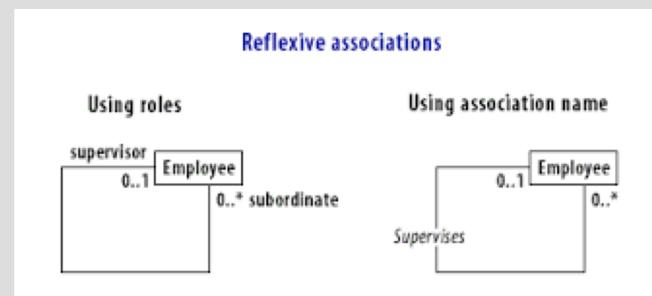
- Unidireccional



- Bidireccional:



- Reflexiva



# UML: Diagramas de clases

- Relación entre instancias: **Agregación**
  - Es un tipo de asociación pero más específica, siempre binaria
  - Variante de la relación “tiene un” (de la parte al todo)
  - Tienen nombre, multiplicidad, visibilidad y otras propiedades
  - Una clase es una colección o contiene a objetos de otra clase que no tienen un ciclo de vida dependiente de la primera clase, la contenedora.

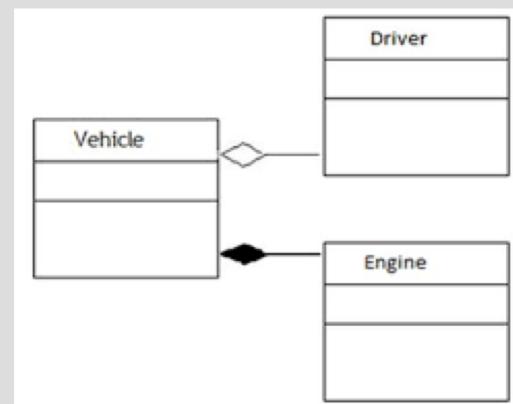
# UML: Diagramas de clases

- Relación entre instancias: **Agregación**
  - Se representa con una forma de diamante vacío en la clase contenedora unida con una línea a la clase contenida
  - Ejemplo:
    - Un profesor tiene que impartir muchas clases



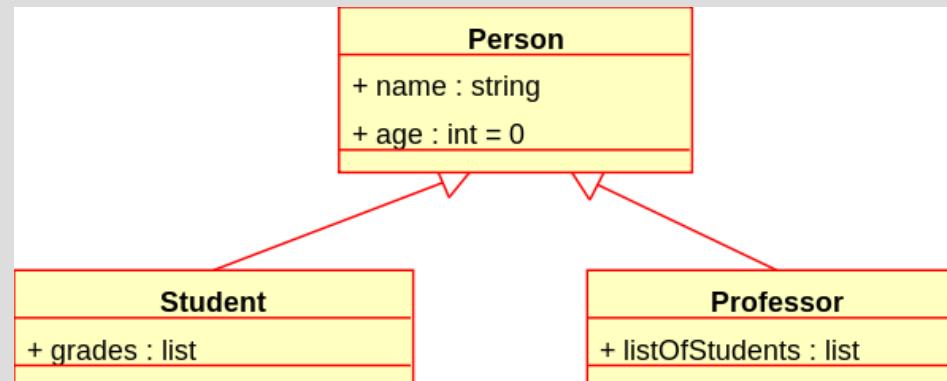
# UML: Diagramas de clases

- Relación entre instancias: **Composición**
  - La relación entre las dos clases está ligada a su ciclo de vida: la clase dependiente no existe sin la clase independientes
  - Se representa con una forma de diamante relleno en la clase contenedora unida con una línea a la clase contenida
  - Ejemplo:



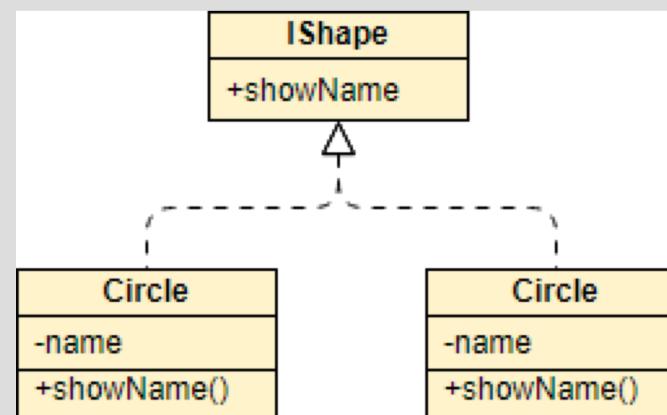
# UML: Diagramas de clases

- Relación entre clases:  
**Generalización/Herencia**
  - La subclase es una forma especializada del súper tipo (la otra clase).
  - La súper clase es una generalización de la subclase
  - Relación “es un”, padre-hijo



# UML: Diagramas de clases

- Relación entre clases:  
**Realización/Implementación**
  - Un elemento del modelo (cliente) realiza (implementa) el comportamiento de otro elemento que lo especifica (proveedor - interfaz).



# UML: Diagrama de clases

- Ejercicio
  - Modeliza el siguiente sistema:
    - Una empresa tiene varios departamentos que están localizados en una o más oficinas. Cada departamento tiene un gestor que es siempre uno de los empleados de la oficina.



Nota: para la realización de los diagramas puedes utilizar la herramienta draw.io (en web y aplicación de escritorio)

# UML: Diagramas de clases

- Ejercicio



- Diseña un sistema para un banco que almacena información de sus empleados, sus clientes y de sus cuentas
- Cada cuenta puede ser de dos tipos: o bien de ahorro o bien de inversión (acciones en bolsa)
- Las acciones se compran mediante pedidos a un precio dado y una cantidad limitada de acciones

Nota: para la realización de los diagramas puedes utilizar la herramienta draw.io (en web y aplicación de escritorio)

# Unified Modeling Language (UML)

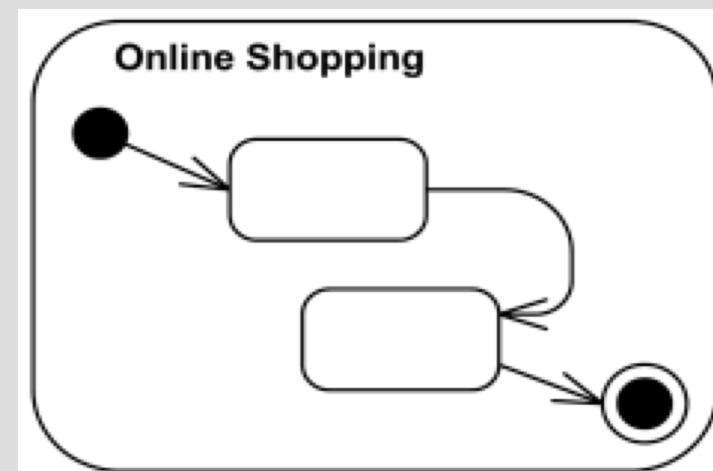
- Introducción
- Toma de requisitos
  - Historias de usuario
  - UML: Casos de uso
- UML: Diagramas de clases
- **UML: Diagramas de comportamiento**
  - Actividad
  - Secuencia

# UML: Diagramas de comportamiento

- Muestran el comportamiento dinámico de los objetos en el sistema.
- Los casos de uso son diagramas de comportamiento
  - Diagramas de actividad
  - Diagramas de secuencia
  - Diagramas de estado

# UML: Diagramas de actividad

- También llamado “Diagrama de flujo”
- Representación gráfica de un algoritmo o proceso paso a paso.
- Enfatiza en la secuencia y las condiciones del flujo

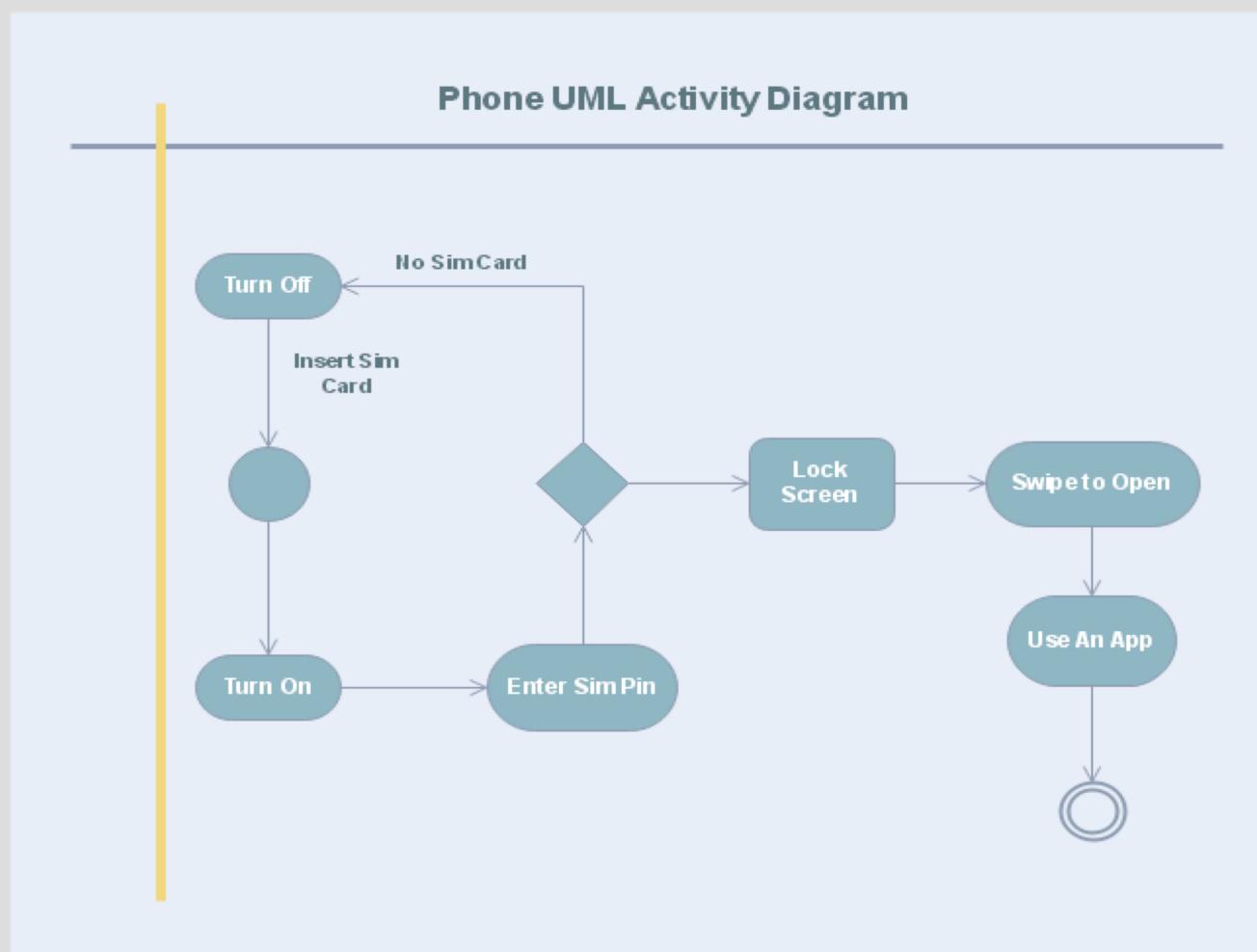


# UML: Diagramas de actividad

- Formas utilizadas en los diagramas de flujo
  - Elipses: representan acciones
  - Diamantes-rombos: representan decisiones
  - Barras horizontales: comienzo (separación) o fin (unión) del flujo de trabajo
  - Círculo negro: punto inicial del flujo
  - Círculo negro con circunferencia: final del flujo

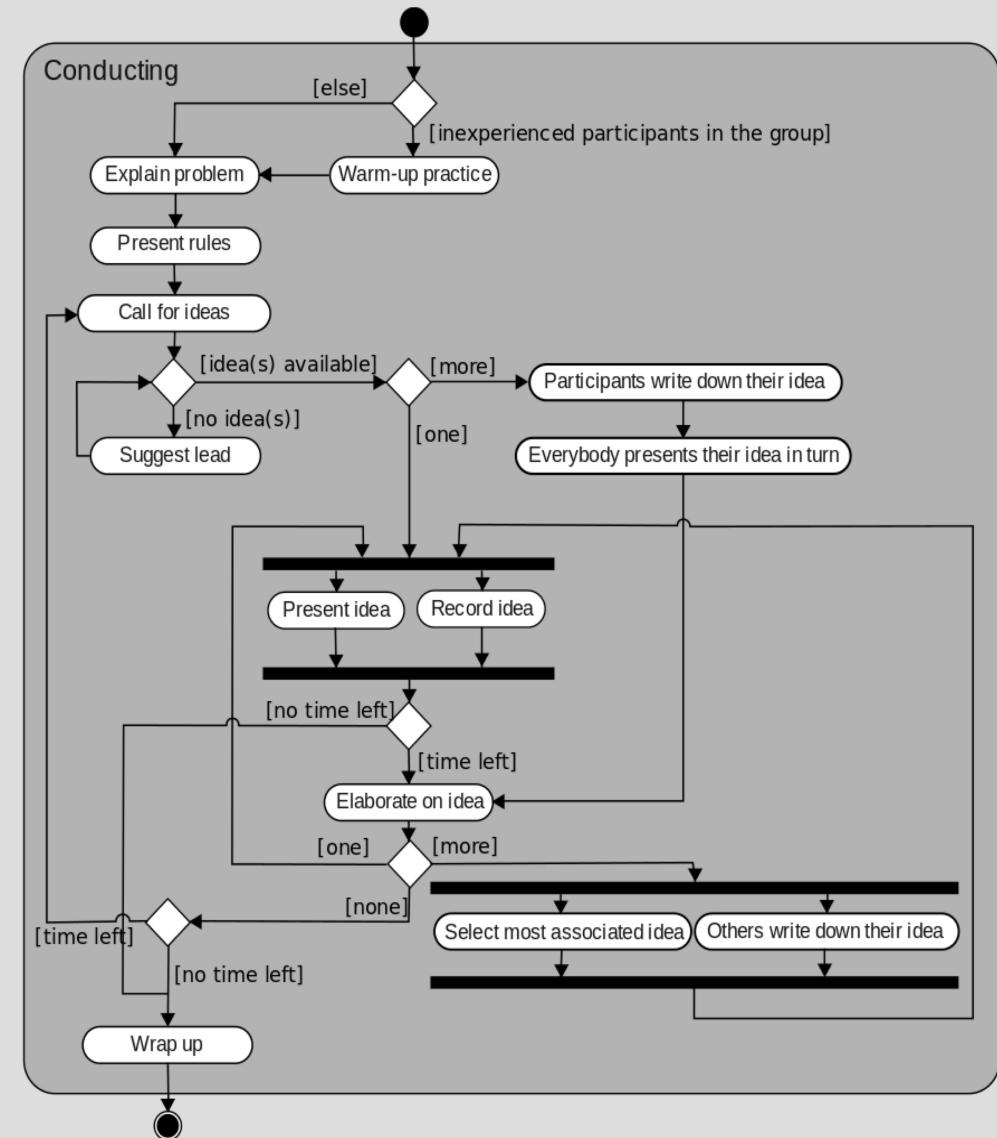
# UML: Diagramas de actividad

- Diagramas de flujo: uso de una App



# UML: Diagramas de actividad

- Diagramas de flujo:  
lluvia de ideas



# UML: Diagramas de actividad

- Diagramas de flujo:

- Realiza un diagrama de flujo de la realización de un huevo frito.
- Ten en cuenta las diversas situaciones a las que te puedes enfrentar
- Si no sabes freír un huevo, Google te ayudará.



# UML: Diagramas de actividad

- Diagramas de flujo:

- Dado un conjunto finito de números enteros, realiza un diagrama de flujo/actividad que muestre un algoritmo de ordenación.
  - Si conoces algún algoritmo de ordenación, indica cuál es



# Unified Modeling Language (UML)

- Introducción
- Toma de requisitos
  - Historias de usuario
  - UML: Casos de uso
- UML: Diagramas de clases
- UML: Diagramas de comportamiento
  - Actividad
  - Secuencia

# Diagramas de secuencia

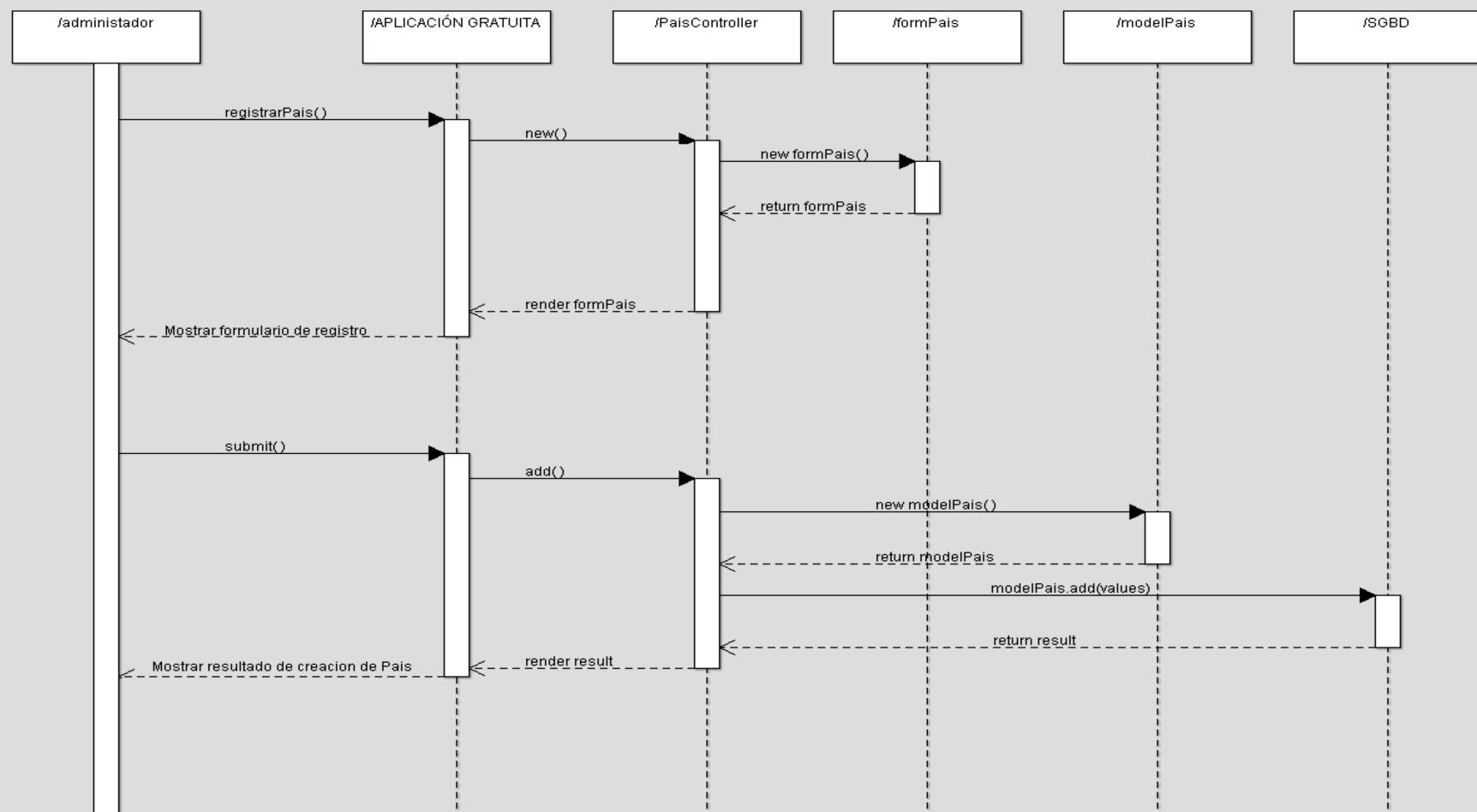
- Modela la interacción entre objetos en un sistema
- Para cada caso de uso se puede diseñar un diagrama de secuencia
- Se ve la evolución a lo largo de un periodo, habitualmente corto, de tiempo
- “Diagrama de clases en movimiento”

# Diagramas de secuencia

- Detalla la implementación del UC
- Incluye objetos y clases utilizados para implementarlo (líneas discontinuas verticales)
- Se indican los mensajes intercambiados (flechas horizontales)

# Diagramas de secuencia

- Ejemplo diagrama de secuencia



# Diagramas de secuencia

- Tipos de mensajes
  - Síncronos
    - Cuando se envía una petición se espera la respuesta inmediata
    - Representados con flechas con la punta rellena
  - Asíncronos
    - Se envía una petición pero no se espera una respuesta.
    - Representados con flechas con la punta hueca

# Diagramas de secuencia

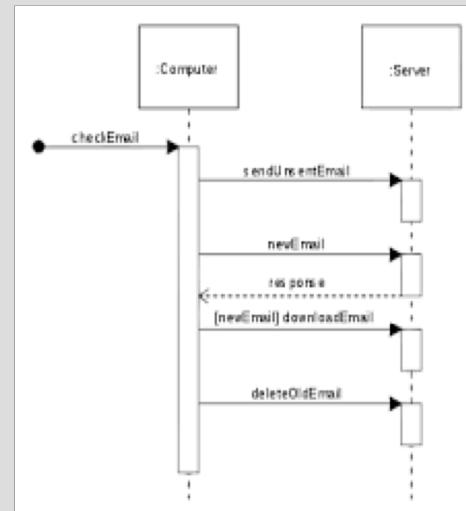
- Tipos de mensajes
  - Síncronos
    - Cuando se envía una petición se espera la respuesta inmediata
    - Representados con flechas con la punta rellena
  - Asíncronos
    - Se envía una petición pero no se espera una respuesta.
    - Representados con flechas con la punta hueca
  - Respuesta a un mensaje: flecha discontinua

# Diagramas de secuencia

- Tipos de mensajes
  - Instancia
    - Describe un escenario específico
      - un escenario es una instancia de la ejecución de un caso de uso.
  - Genérico
    - Describe la interacción para un caso de uso.
    - Utiliza ramificaciones ("branches"), condiciones y bucles.

# Diagramas de secuencia

- Gráfico
  - Mensajes: de arriba a abajo cronológicamente
  - Objetos/Clases: la distribución es arbitraria aunque se suelen dibujar en orden de uso



# UML: Diagramas de secuencia

- Diagramas de secuencia:
  - Realiza un diagrama de flujo de la realización de un huevo frito.
  - Te puedes ayudar del diagrama de actividad ya realizado



# UML: Diagramas de secuencia

- Diagramas de secuencia:

- Tomando como partida el diagrama de clases del ejercicio del banco, realiza diagramas de secuencia:
    - de retirada de dinero de un cajero por parte de un cliente
    - de compra de acciones de un cliente de inversión



# Bibliografía

- Coste de los errores software, Jose Huerta
- Wikipedia
- Javier Garzas
- Documentación de las Universidades de Sevilla, Complutense y Granada