FUNDAMENTOS DE LA VISTA DE CASOS DE USO

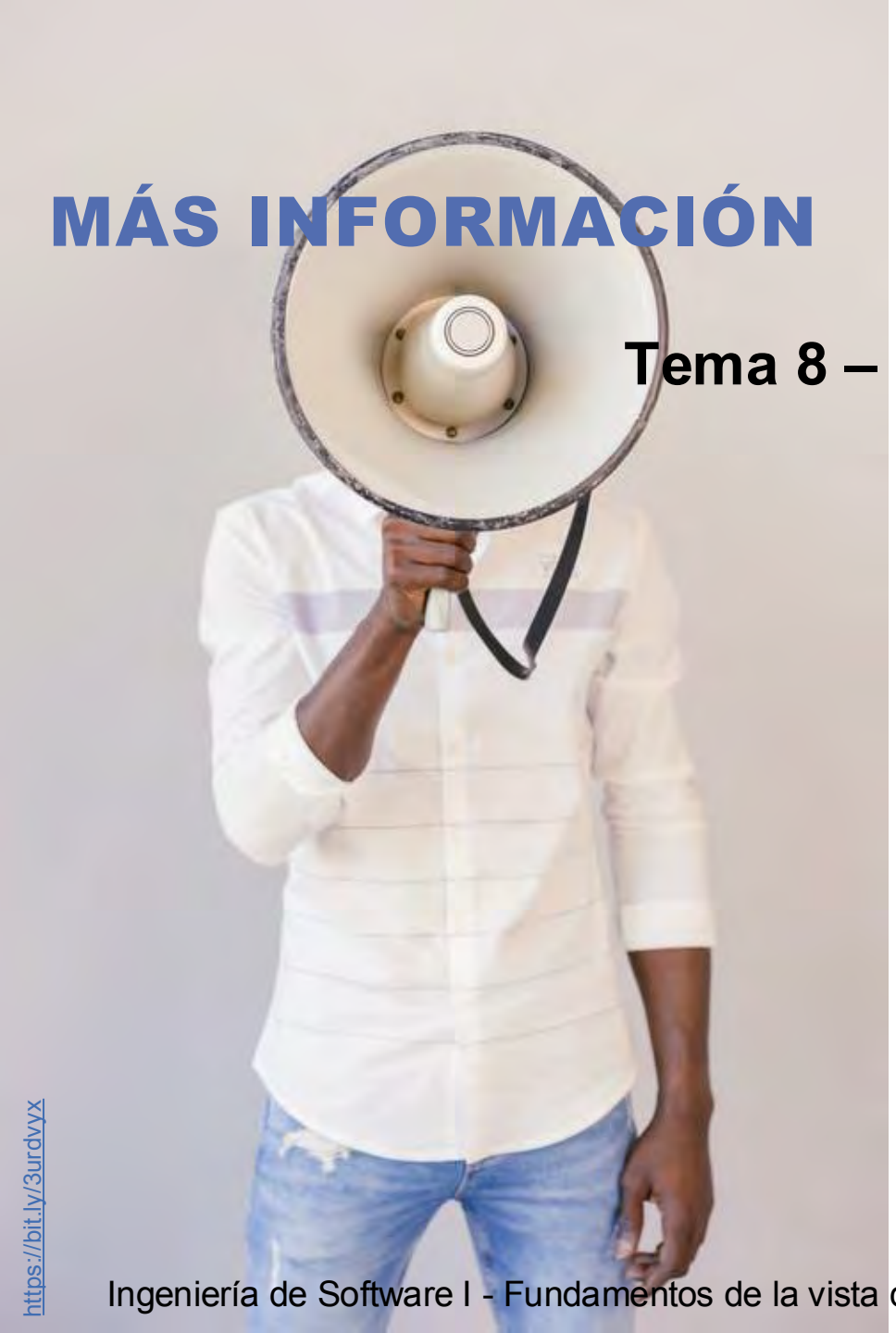
INGENIERÍA DE SOFTWARE I

2º DE GRADO EN INGENIERÍA INFORMÁTICA
CURSO 2024/2025

Dr. Francisco José García-Peñalvo / fgarcia@usal.es
Dra. Alicia García-Holgado / aliciagh@usal.es
Dra. Andrea Vázquez-Ingelmo / andreavazquez@usal.es

Departamento de Informática y Automática
Universidad de Salamanca

MÁS INFORMACIÓN



Tema 8 – UML [1]

https://bit.ly/3urdvyx

ÍNDICE

Introducción a UML

CARACTERÍSTICAS

UML es un lenguaje de modelado para visualizar, especificar, construir y documentar partes de un sistema *software* desde distintos puntos de vista

- Puede usarse con cualquier proceso de desarrollo, a lo largo de todo el ciclo de vida y puede aplicarse a todos los dominios de aplicación y plataformas de implementación
- También puede usarse en tres áreas, como la ingeniería de negocio y modelado de procesos gracias a los mecanismos de adaptación/extensión mediante perfiles

Lo que no es

- UML no es una notación propietaria
- UML no es un método, ni un proceso ni una metodología

El objetivo de UML es la unificación de los métodos de modelado de objetos (Booch, OMT y OOSE) por medio de la

- Identificación y definición de la semántica de los conceptos fundamentales y elección de una representación gráfica con una sintaxis simple, expresiva e intuitiva

La especificación UML se define usando el enfoque de un metamodelo

GÉNESIS Y EVOLUCIÓN

En 1994 Rumbaugh y Booch crean El Método Unificado

En 1995 se incorpora Jacobson y los tres autores publican un documento titulado Unified Method V0.8 (Booch y Rumbaugh) [2]

El método unificado se reorienta hacia la definición de un lenguaje universal para el modelado de objetos, transformándose en UML (*Unified Modeling Language for Object-Oriented Development*)

En 1996 se crea un consorcio de colaboradores para trabajar en la versión 1.0 de UML

En 1997 se produce la estandarización de UML 1.0 por la OMG [3]

La siguiente versión oficial de UML es la versión 1.1

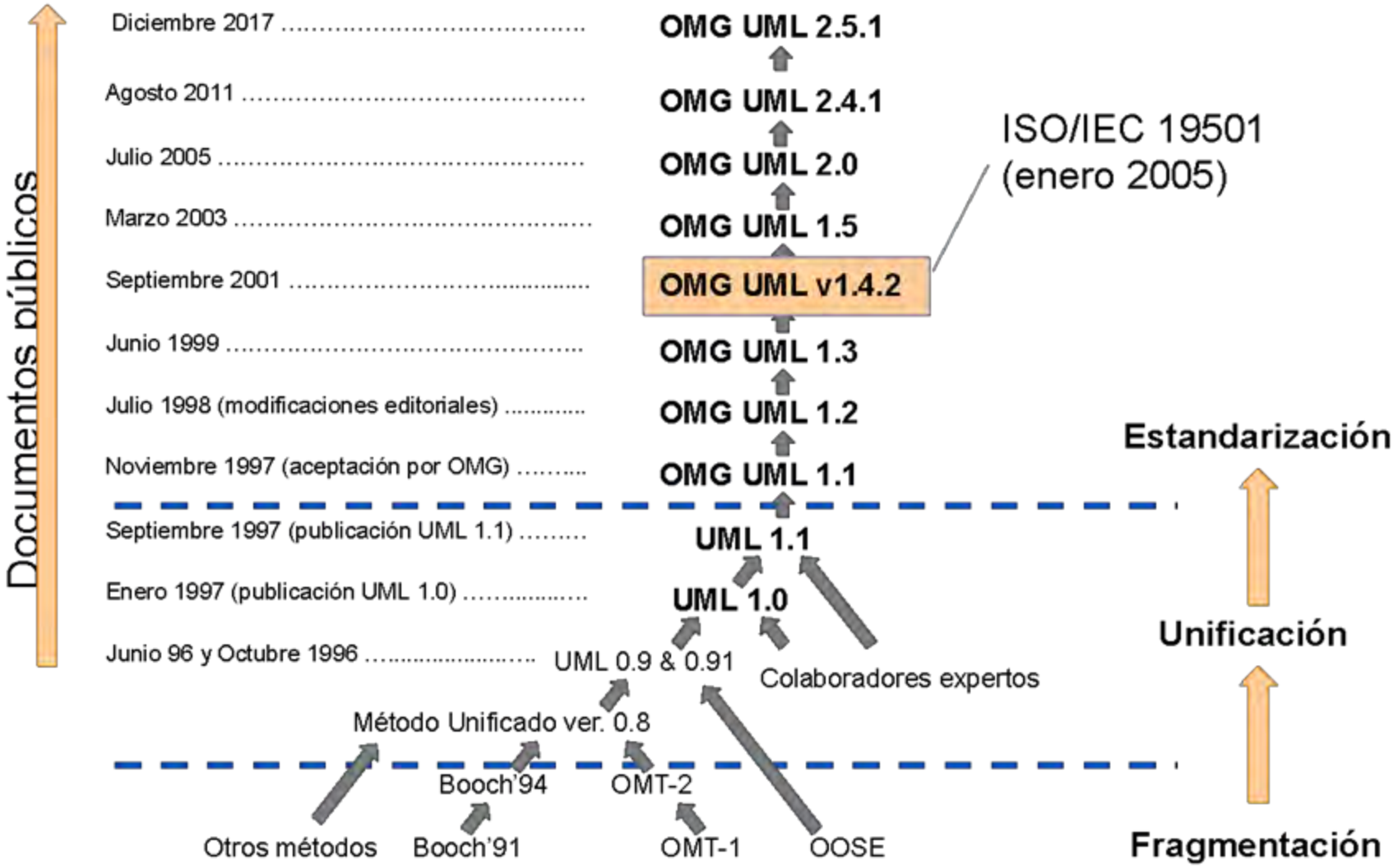
En julio de 1998 aparece una revisión interna de UML que recoge diversos cambios editoriales, pero no técnicos. Esta versión es la que se conoce como UML 1.2

Casi un año más tarde, en junio de 1999 aparece OMG UML 1.3 con algunos cambios significativos, especialmente en lo tocante a la semántica

En septiembre de 2001 aparece UML 1.4 y en enero de 2005 OMG UML 1.4.2 (OMG document: formal/05-04-01) es aceptado como un estándar ISO (ISO/IEC 19501)

En julio de 2005 se libera UML 2.0 siendo la última versión 2.5.1 [4]

GÉNESIS Y EVOLUCIÓN



DIAGRAMAS Y VISTAS

UML define varios modelos para la representación de los sistemas que pueden verse y manipularse mediante un conjunto de diagramas diferentes

- **Diagramas de estructura**
  - Diagrama de clases
  - Diagrama de estructuras compuestas
  - Diagrama de componentes
  - Diagrama de despliegue
  - Diagrama de objetos
  - Diagrama de paquetes
- **Diagramas de comportamiento**
  - Diagrama de casos de uso
  - Diagrama de actividad
  - Diagramas de interacción
    - Diagrama de secuencia
    - Diagrama de comunicación o colaboración
    - Diagrama de visión global de la interacción
    - Diagrama de tiempo
  - Diagrama de máquina de estados

DIAGRAMAS Y VISTAS

Una vista es un subconjunto de las construcciones de modelado de UML que representa un aspecto del sistema

Los diagramas UML se pueden organizar en las siguientes vistas [5]

- **Vista estática**
  - Diagramas de clases
- **Vista de casos de uso**
  - Diagramas de casos de uso
- **Vista de interacción**
  - Diagramas de secuencia
  - Diagramas de comunicación
- **Vista de actividad**
  - Diagramas de actividad
- **Vista de la máquina de estados**
  - Diagramas de máquina de estados
- **Vista de diseño**
  - Diagramas de estructuras compuestas
  - Diagramas de colaboración
  - Diagramas de componentes
- **Vista de despliegue**
  - Diagramas de despliegue
- **Vista de gestión del Modelo**
  - Diagramas de paquetes
- **Perfiles**
  - Diagramas de paquetes

DIAGRAMAS Y VISTAS

Las vistas se pueden agrupar en áreas conceptuales

Área	Vista
Estructural	Vista estática
	Vista de diseño
	Vista de casos de uso
Dinámica	Vista de máquina de estados
	Vista de actividad
	Vista de interacción
Física	Vista de despliegue
Gestión	Vista de gestión del modelo
	Perfiles

Vista de casos de uso

INTRODUCCIÓN

- La vista de casos de uso captura la funcionalidad de un sistema, de un subsistema, o de una clase, tal como se muestra a un usuario exterior
- Reparte la funcionalidad del sistema en transacciones significativas para los usuarios ideales de un sistema
- Los usuarios del sistema se denominan actores y las particiones funcionales se conocen con el nombre de casos de uso
- La técnica que se utiliza para modelar esta vista es el diagrama de casos de uso

CARACTERÍSTICAS

- Los casos de uso son una técnica para la especificación de requisitos funcionales propuesta inicialmente por **Ivar Jacobson** [6, 7] e incorporada a UML
- Modela la funcionalidad del sistema tal como la perciben los agentes externos, denominados actores, que interactúan con el sistema desde un punto de vista particular
- Sus componentes principales son
  - **Sujeto**: sistema que se modela
  - **Casos de uso**: unidades funcionales completas
  - **Actores**: entidades externas que interactúan con el sistema
- El sujeto se muestra como una caja negra que proporciona los casos de uso
- El modelo de casos de uso se representa mediante los **diagramas de casos de uso**

CARACTERÍSTICAS

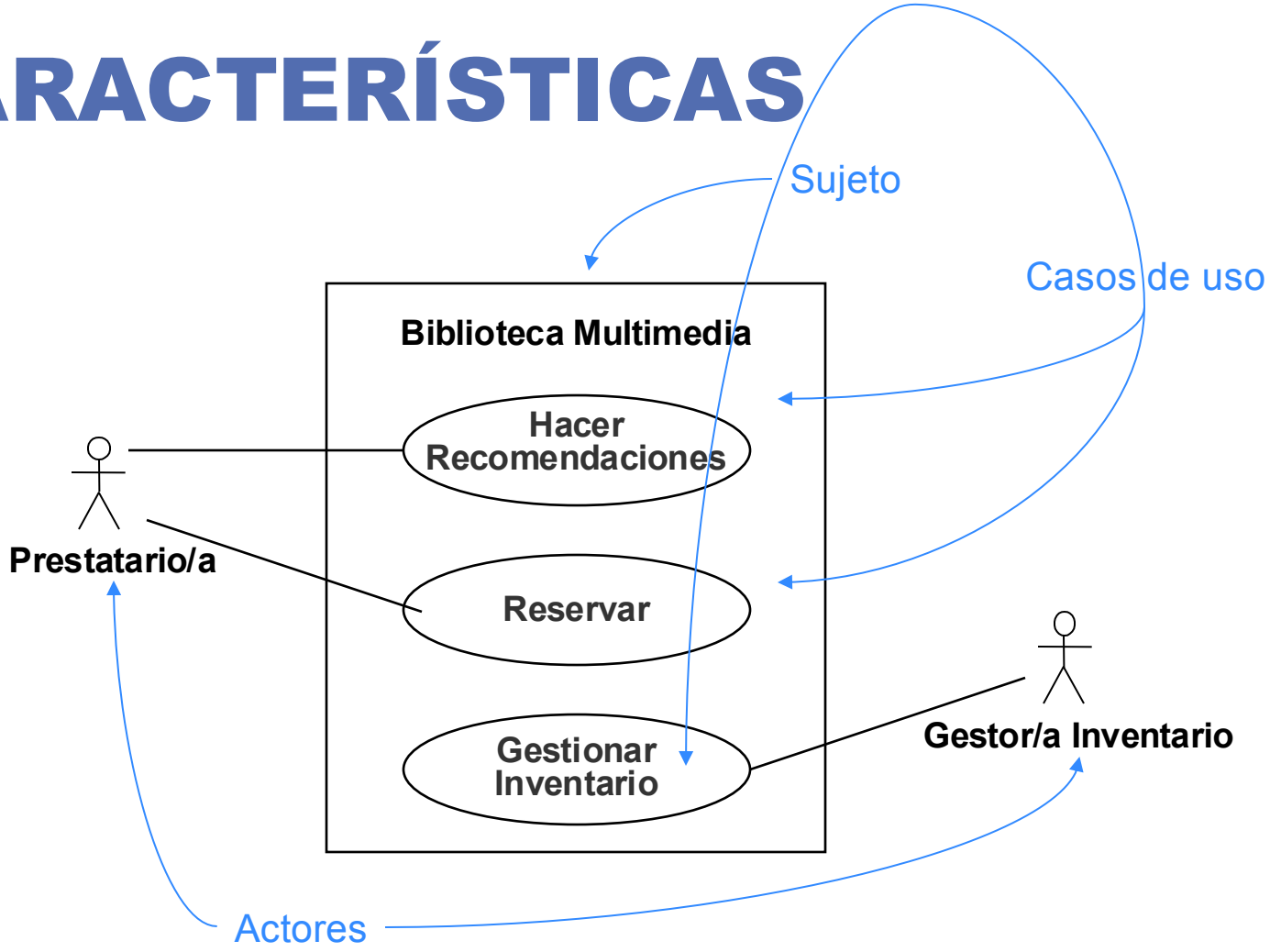


Diagrama de casos de uso

ACTORES

- Un actor es un clasificador que modela un tipo de rol que juega una entidad que interacciona con el sujeto pero que es externa a él
  - Un actor puede tener múltiples instancias físicas
  - Una instancia física de un actor puede jugar diferentes papeles
- Los actores se comunican con el sujeto intercambiando mensajes (señales, llamadas o datos)
- Notación
  - Se representan con el icono estándar de "*stick man*" o "monigote" con el nombre del actor (obligatorio) cerca del símbolo, normalmente se pone encima o debajo
  - También se puede representar mediante un símbolo de clasificador con el estereotipo **«actor»**
  - Los nombres de los actores suelen empezar por mayúscula
  - Se pueden usar otros símbolos para representar tipos de actores, por ejemplo, para representar actores no humanos

TIPOS DE ACTORES [8]

Principales

- Tiene objetivos de usuario que se satisfacen mediante el uso de los servicios del sistema
- Se identifican para encontrar los objetivos de usuario, los cuales dirigen los casos de uso

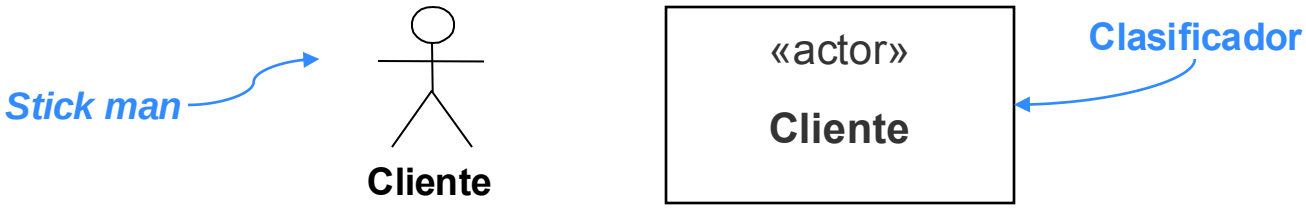
De apoyo

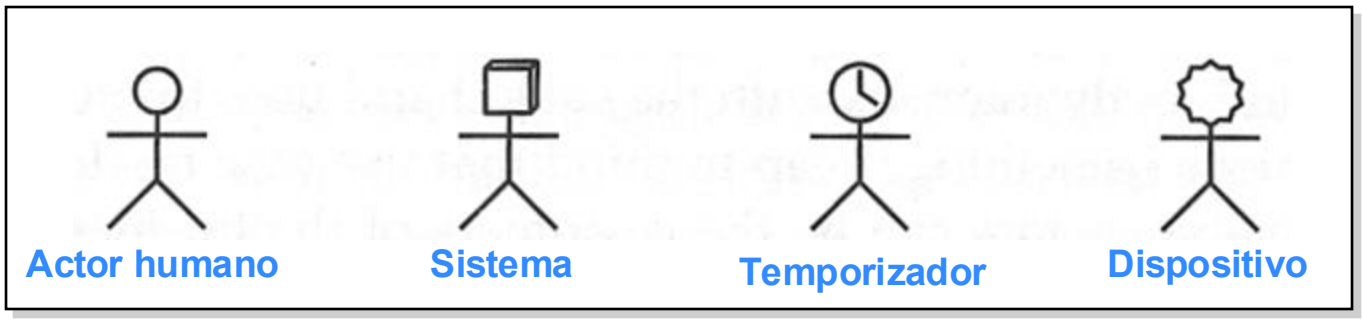
- Proporcionan un servicio al sistema
- Normalmente se trata de un sistema informático, pero podría ser una organización o una persona
- Se identifican para clarificar las interfaces externas y los protocolos

Pasivos

- Está interesado en el comportamiento del caso de uso, pero no es principal ni de apoyo
- Se identifican para asegurar que todos los intereses necesarios se han identificado y satisfecho
- Los intereses de los actores pasivos algunas veces son sutiles o es fácil no tenerlos en cuenta, a menos que estos actores sean identificados explícitamente

ACTORES



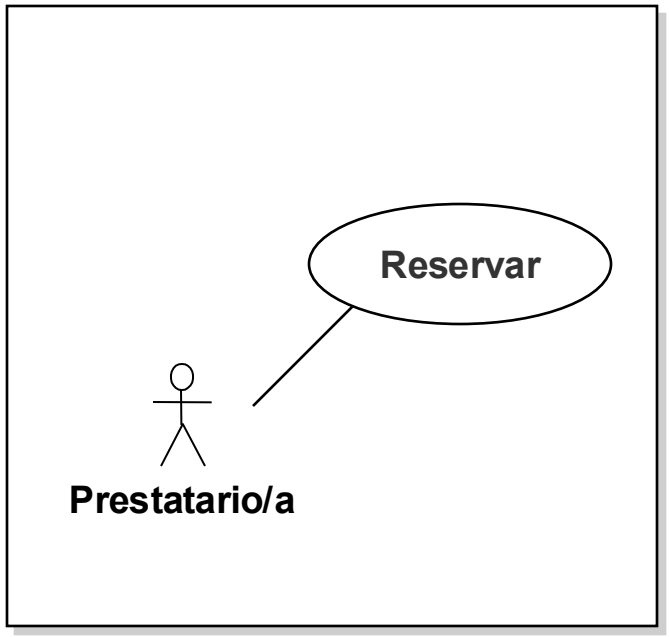


Símbolos utilizados para representar tipos de actores

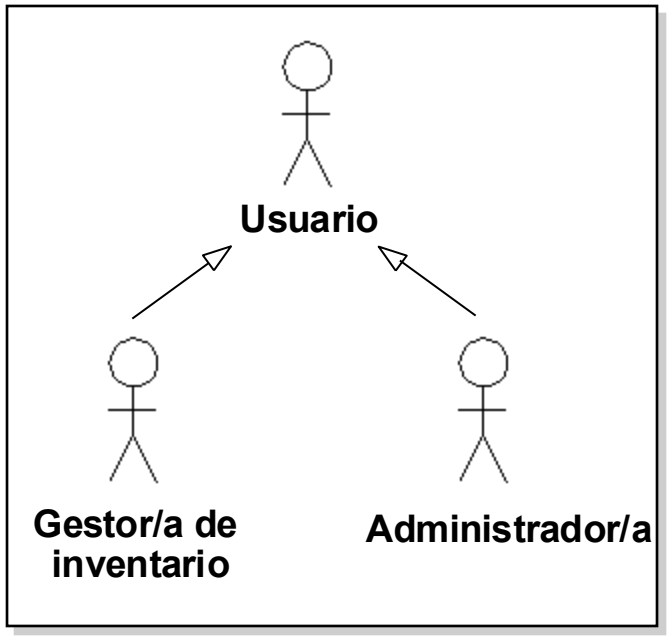
RELACIONES ENTRE ACTORES

- Los actores solo pueden tener asociaciones con casos de uso, subsistemas, componentes y clases y dichas asociaciones deben ser binarias
- Se pueden establecer relaciones de generalización entre actores
  - El actor general describirá el comportamiento de un rol más general
  - Los actores especializados heredan el comportamiento del actor general y lo extienden de alguna forma
  - Una instancia de un actor descendiente siempre se puede utilizar en aquellos casos en los que se espera una instancia del actor antecesor
  - Los actores pueden ser abstractos, en ese caso se representan con el nombre en cursiva

RELACIONES ENTRE ACTORES



Asociación entre un actor y un caso de uso



Relaciones de generalización entre actores

CASOS DE USO

- Un caso de uso se define como un conjunto de acciones realizadas por el sistema que dan lugar a un resultado observable
- El caso de uso especifica un comportamiento que el sujeto puede realizar en colaboración con uno o más actores, pero sin hacer referencia a su estructura interna
- El caso de uso puede contener posibles variaciones de su comportamiento básico incluyendo manejo de errores y excepciones
- Una instanciación de un caso de uso es un **escenario** que representa un uso particular del sistema (un camino)
- Características de los casos de uso
  - Un caso de uso se inicia por un actor
  - Los casos de uso proporcionan valores a los actores
  - La funcionalidad de un caso de uso debe ser completa
- El comportamiento de un caso de uso se puede describir mediante interacciones, actividades, máquinas de estado...

CASOS DE USO

Notación

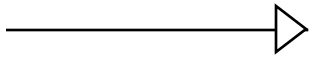
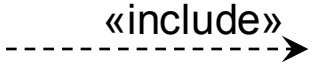
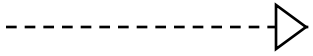
- Elipse con el nombre del caso de uso dentro o debajo de ella. Se puede colocar algún estereotipo encima del nombre y una lista de propiedades debajo
- La representación alternativa es la del símbolo del clasificador con una elipse pequeña en la esquina superior derecha

Notaciones usadas para la representación de casos de uso

RELACIONES DE LOS CASOS DE USO

- Los casos de uso pueden tener asociaciones y dependencias con otros clasificadores
- Relación **entre actores y casos de uso**
  - **Asociación**
- Relaciones **entre casos de uso**
  - **Generalización**: Un caso de uso también se puede especializar en uno o más casos de uso hijos
  - **Inclusión**: Un caso de uso puede incorporar el comportamiento de otros casos de uso como fragmentos de su propio comportamiento
  - **Extensión**: Un caso de uso también se puede definir como una extensión incremental de un caso de uso base
- Relación **entre un caso de uso y una colaboración**
  - **Realización**

RELACIONES DE LOS CASOS DE USO

Relación	Descripción	Notación
Asociación	Línea de comunicación entre un actor y un caso de uso en el que participa	
Generalización	Una relación entre un caso de uso general y un caso de uso más específico, que hereda y añade propiedades al caso de uso base	
Inclusión	Inserción de comportamiento adicional en un caso de uso base, que describe explícitamente la inserción	
Extensión	Inserción de comportamiento adicional en un caso de uso base que no tiene conocimiento sobre él	
Realización	Establece una relación entre el caso de uso y los diagramas que describen la funcionalidad del caso de uso	

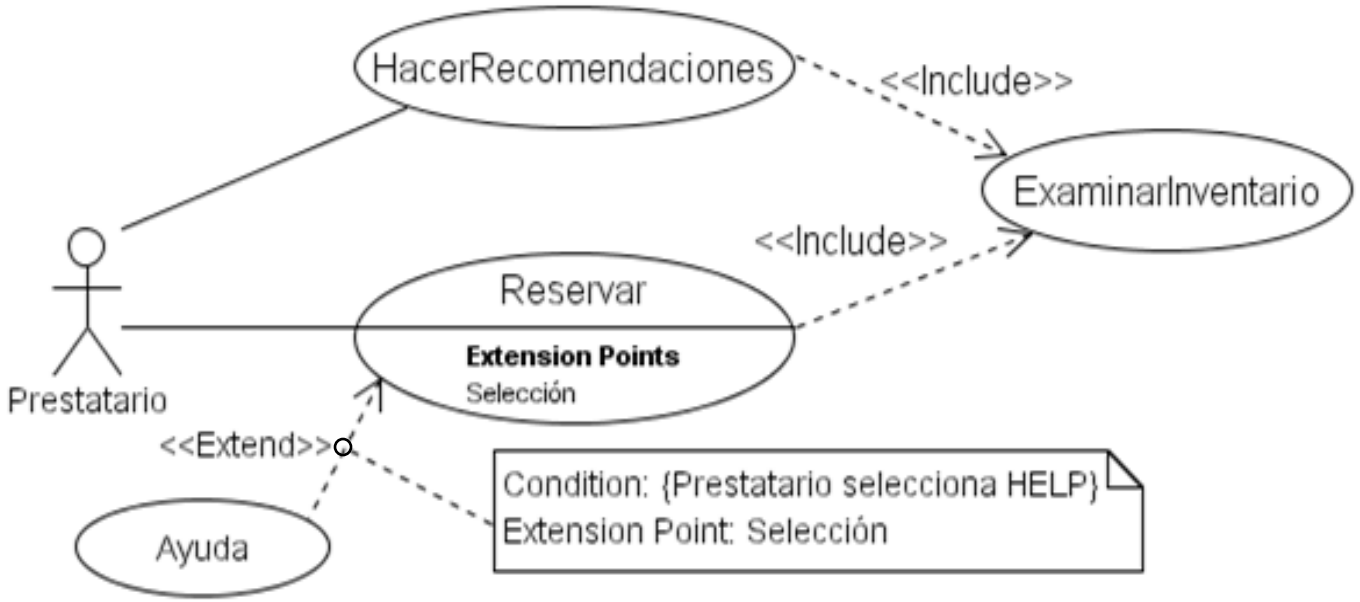
RELACIONES DE LOS CASOS DE USO

Relación de inclusión (I)

- Relación entre dos casos de uso que indica que el comportamiento de un caso de uso (**incluido**) se inserta en el comportamiento de otro caso de uso (**base** o **inclusor**) en la **localización** especificada en este último
- La inclusión no es condicional
- El propósito de la inclusión es la reutilización de porciones de comportamiento comunes a varios casos de uso
    - Un caso de uso incluido puede insertarse en varios casos de uso base y puede, a su vez, incluir otros casos de uso
    - Un caso de uso base puede tener relaciones de inclusión con varios casos de uso incluidos
- La ejecución es análoga a las llamadas a procedimientos
- Notación de la relación de inclusión: símbolo de dependencia con el estereotipo **«include»**

RELACIONES DE LOS CASOS DE USO

Relación de inclusión (II)



RELACIONES DE LOS CASOS DE USO

Relación de extensión (I)

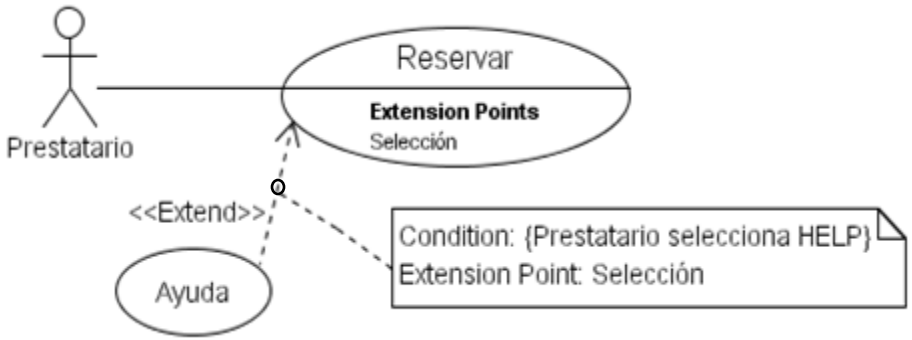
- Dependencia entre dos casos de uso que especifica que el comportamiento de un caso de uso base (**extendido**) puede ser extendido con comportamiento adicional definido en otro caso de uso (**extensor**)
- El caso de uso extendido define un comportamiento que tiene significado con independencia del caso de uso extensor
- El comportamiento del caso de uso extensor incrementa el del caso de uso base solo en determinadas condiciones
- Un caso de uso extensor puede extender varios casos de uso base y puede, a su vez, ser extendido por otro caso de uso
- La extensión tiene lugar en **puntos de extensión**
    - pertenecen al caso de uso extendido
    - Indican el lugar donde se insertan los **fragmentos de comportamiento** del caso de uso extensor

RELACIONES DE LOS CASOS DE USO

Relación de extensión (II)

- Notación de la relación de extensión: símbolo de dependencia con el estereotipo **«extend»** y opcionalmente una nota con las condiciones y las referencias a los puntos de extensión
- Notación de los puntos de extensión: se representan como una cadena de texto dentro del caso de uso conforme a la sintaxis:

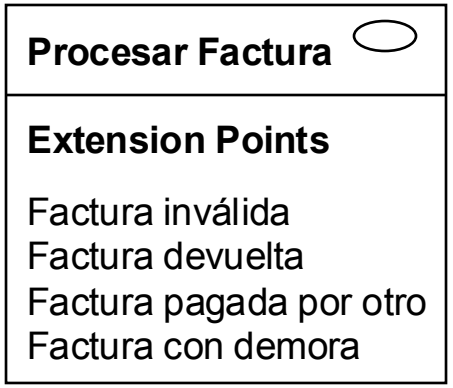
< nombre > [: <explicación>]



RELACIONES DE LOS CASOS DE USO

Relación de extensión (III)

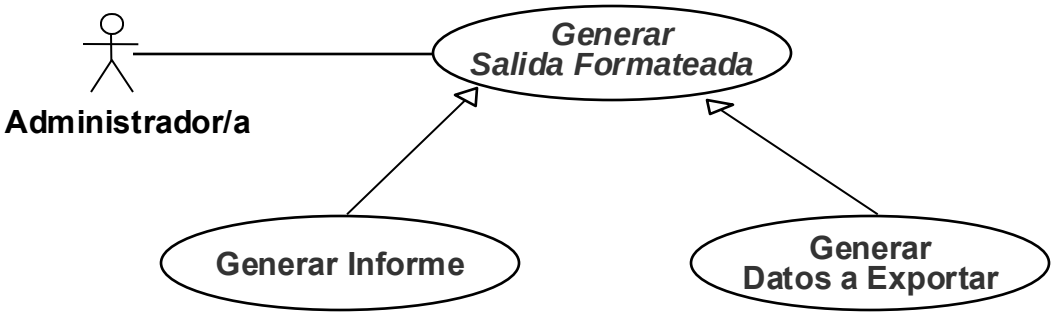
- Si hay varios puntos de extensión en un caso de uso es mejor representar el caso de uso con el símbolo del clasificador
- **Condición de la extensión**
  - Es única para todos los puntos de extensión de una relación de extensión
  - Si es verdadera cuando se alcanza el primer punto de extensión al ejecutar el caso de uso base, serán ejecutados todos los fragmentos del caso de uso extensor correspondientes a todos los puntos de extensión. Terminada la ejecución de un fragmento dado el control retorna al caso de uso base y continua su ejecución hasta el siguiente punto de extensión
  - Si la condición es falsa no se produce la extensión
  - Si no hay condición la extensión es incondicional (ocurre siempre)



RELACIONES DE LOS CASOS DE USO

Generalización de casos de uso

- Una relación de generalización relaciona un caso de uso especializado con otro caso de uso más general
- El hijo hereda las relaciones y comportamiento del padre y puede agregar atributos y operaciones propios
- El caso de uso hijo añade comportamiento al caso de uso padre **insertando secuencias de acción** adicionales en la secuencia del padre en puntos arbitrarios
- También puede **modificar algunas operaciones y secuencias heredadas**, pero debe hacerse de forma que la intención del padre se mantenga
- El caso de uso padre puede ser abstracto



Relaciones de generalización entre casos de uso

ORGANIZACIÓN DE LOS CASOS DE USO

- Los casos de uso se pueden agrupar en paquetes
  - Los paquetes se pueden organizar jerárquicamente
  - Un caso de uso puede estar en más de un paquete
  - Pueden existir relaciones entre casos de uso de diferentes paquetes
  - Se pueden agrupar actores en un paquete
- Los clasificadores pueden poseer casos de uso
  - Forma de organización alternativa permitida en UML 2
  - El conjunto completo de casos de uso de un clasificador especifica todas las distintas formas que hay de utilizar ese clasificador

ORGANIZACIÓN DE LOS CASOS DE USO

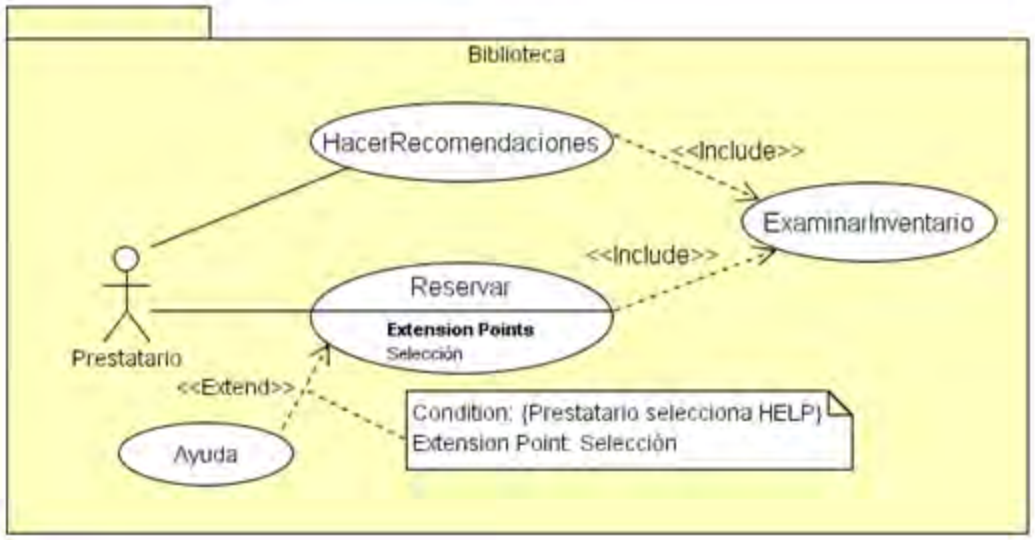
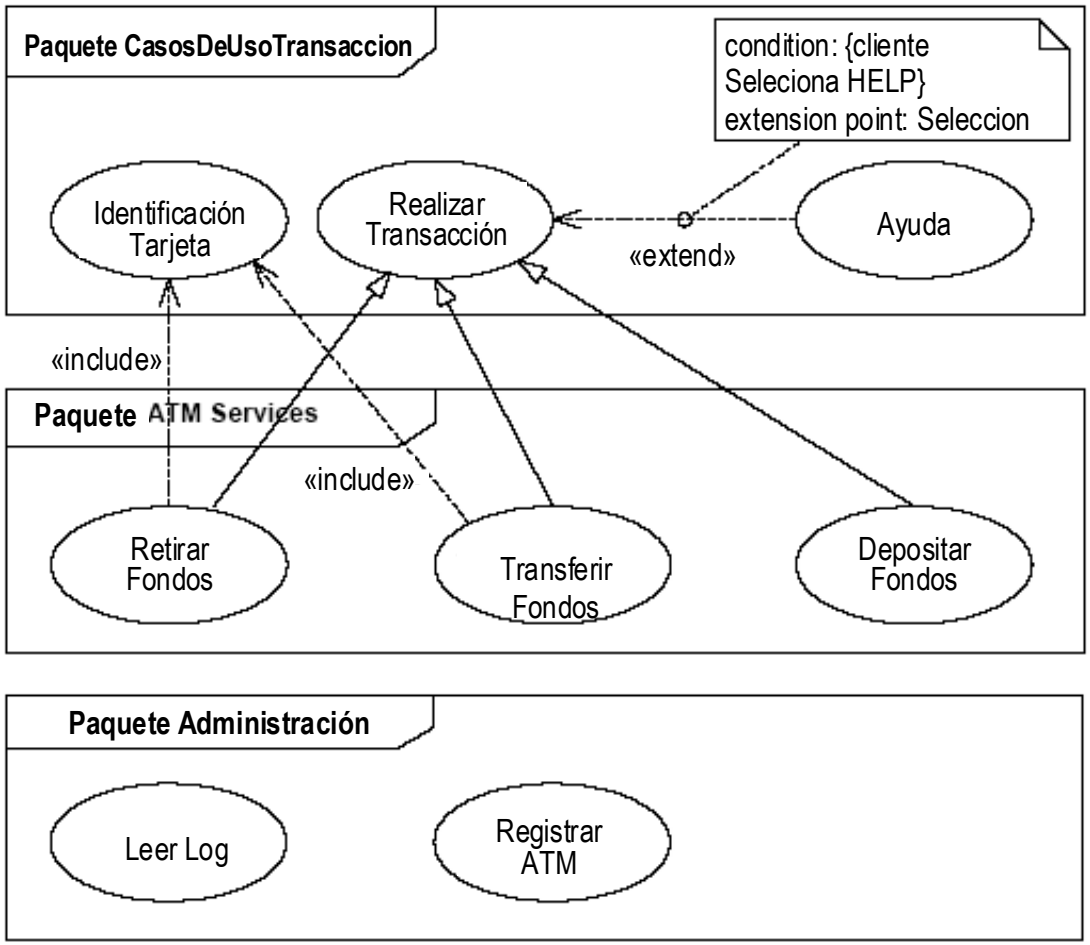
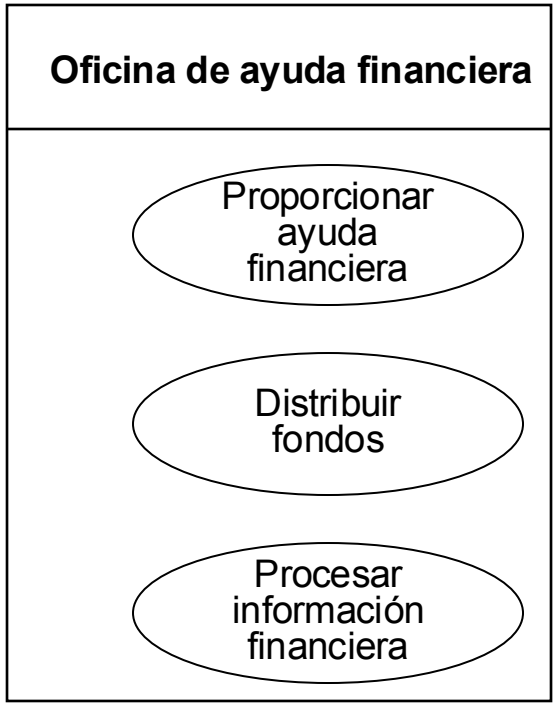


Diagrama de casos de uso del paquete Biblioteca

ORGANIZACIÓN DE LOS CASOS DE USO

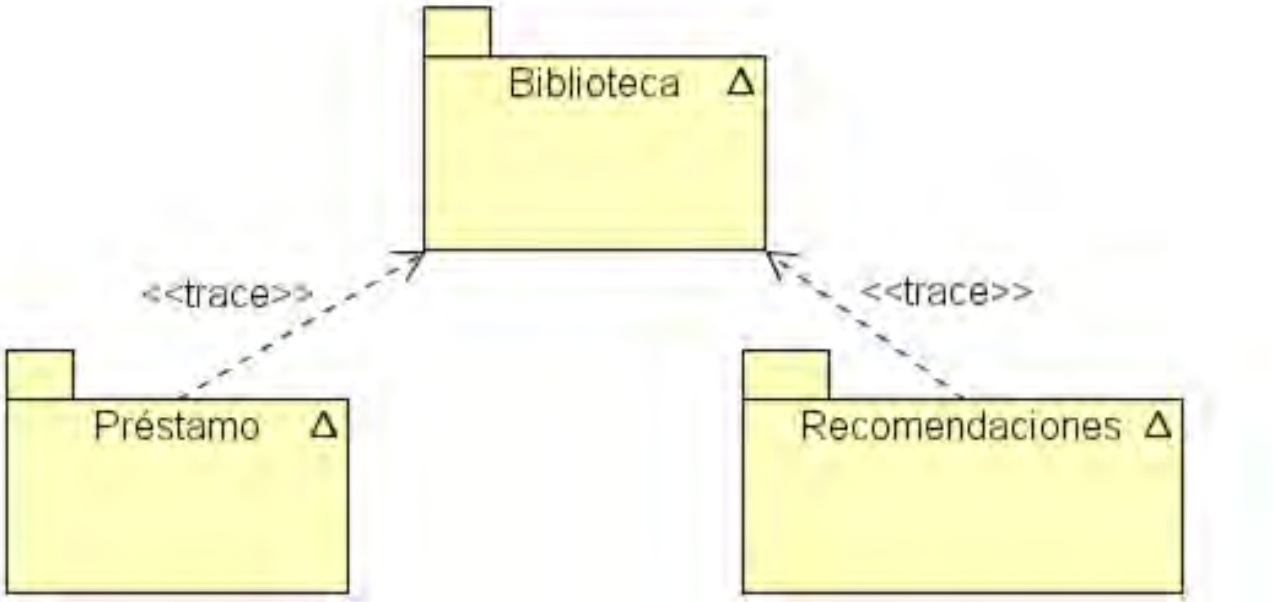


Relaciones entre casos de uso de diferentes paquetes



Casos de uso contenidos en un clasificador

ORGANIZACIÓN DE LOS CASOS DE USO



Paquetes de casos de uso en diferentes niveles de abstracción

ORGANIZACIÓN DE LOS CASOS DE USO

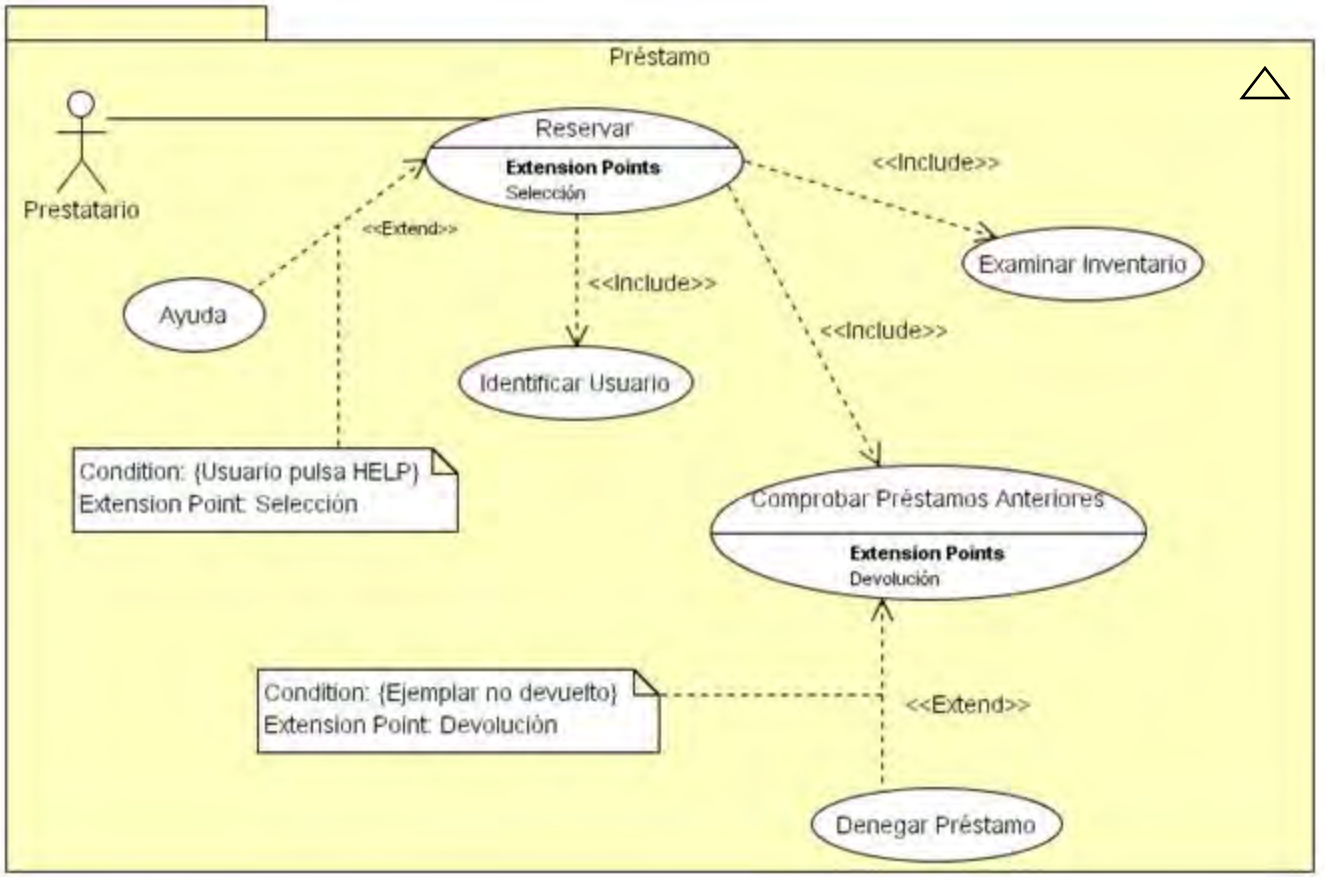


Diagrama de casos de uso del paquete Préstamo

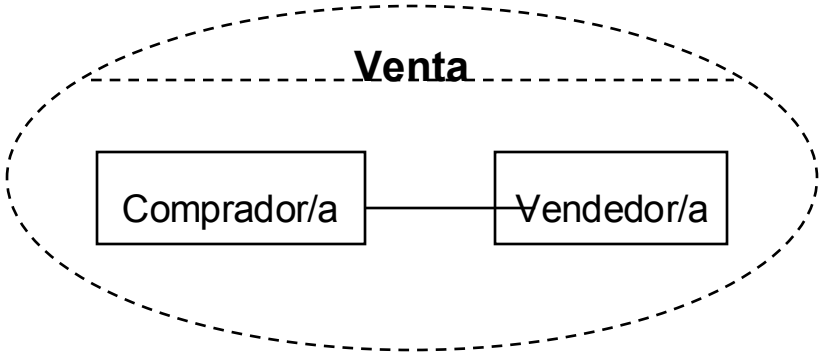
REALIZACIÓN DE LOS CASOS DE USO

- Las responsabilidades de realización de las acciones descritas en los casos de uso se asignan a objetos que colaboran e implementan la funcionalidad del caso de uso
- **Principios para la realización de los casos de uso (I)**
  - Una **colaboración** realiza un caso de uso: solución dependiente de la implementación
    - **Contexto** de la colaboración: relaciones entre clases y objetos
    - **Interacción** de la colaboración: interacciones entre ellos para alcanzar la funcionalidad deseada
  - El símbolo de la colaboración es una elipse con la línea discontinua y con el nombre en su interior
- Para explicar una colaboración se requieren diagramas que muestren el contexto y la interacción entre los elementos que colaboran: diagramas de comunicación, de secuencia, de visión global de la interacción, de actividad y de máquina de estados

REALIZACIÓN DE LOS CASOS DE USO

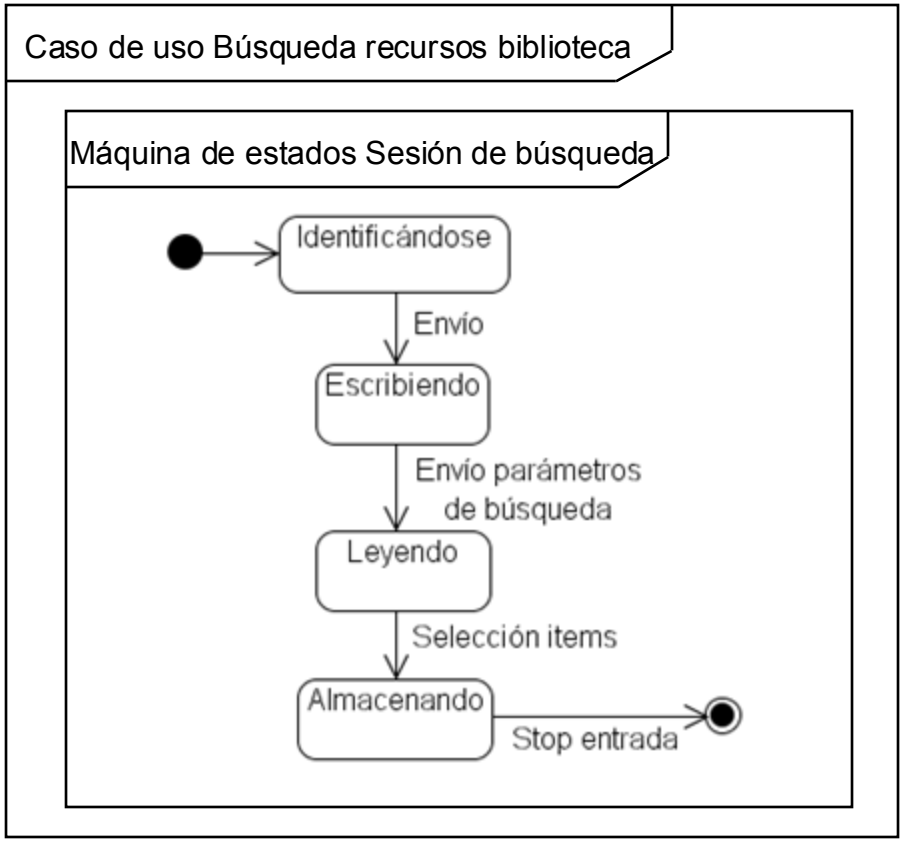
Principios para la realización de los casos de uso (II)

- Un **escenario** es una instancia de un caso de uso
  - Un escenario es un camino de ejecución específico que representa una instanciación específica de un caso de uso
  - Un escenario visto como una ocurrencia de una colaboración incluye la interacción entre las partes dentro del sistema
- Un caso de uso puede poseer diagramas que detallen su estructura interna: pueden enfatizar su estructura de tiempo de ejecución u otros elementos que surgen en la implementación del caso de uso (por ejemplo, un diagrama de máquina de estados)



Colaboración que realiza un caso de uso

REALIZACIÓN DE LOS CASOS DE USO



Caso de uso con diagramas que detallan su estructura interna

REFERENCIAS

1. F. J. García-Peñalvo, M. N. Moreno García, A. García-Holgado y A. Vázquez-Ingelmo, "UML. Unified Modeling Language," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2024-2025, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2025. [Online]. Disponible en. doi: 10.5281/zenodo.14841504.
2. G. Booch y J. Rumbaugh, "Unified Method for Object-Oriented Development," Rational Software Corporation Documentation set, version 0.8, 1995.
3. G. Booch, I. Jacobson y J. Rumbaugh, "The Unified Modeling Language for Object-Oriented Development," Rational Software Corporation Documentation set, version 1.0, 13 January 1997
4. Object Management Group, "Unified Modeling Language specification version 2.5.1," Object Management Group, Needham, MA, USA, formal/17-12-05, 2017. Disponible en: https://d66z.short.gy/RnnKjS.
5. J. Rumbaugh, I. Jacobson y G. Booch, *The Unified Modeling Language reference manual*, 2ª ed. (Object Technology Series). Boston, MA, USA: Addison Wesley, 2005.
6. I. Jacobson, "Object Oriented Development in an Industrial Environment," en *Proceedings of the 1987 OOPSLA - Conference proceedings on Object-Oriented Programming Systems, Languages and Applications. (October 4-8, 1987, Orlando, FL USA)* pp. 183-191, New York, NY, USA: ACM, 1987.
7. I. Jacobson, M. Christerson, P. Jonsson y G. Övergaad, *Object oriented software engineering: A use case driven approach*. Reading, MA, USA: Addison-Wesley, 1992.
8. C. Larman, *Applying UML and patterns. An introduction to object-oriented analysis and design and the Unified Process*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2004.

FUNDAMENTOS DE LA VISTA DE CASOS DE USO

INGENIERÍA DE SOFTWARE I

2º DE GRADO EN INGENIERÍA INFORMÁTICA
CURSO 2024/2025

Dr. Francisco José García-Peñalvo / fgarcia@usal.es
Dra. Alicia García-Holgado / aliciagh@usal.es
Dra. Andrea Vázquez-Ingelmo / andreavazquez@usal.es

Departamento de Informática y Automática
Universidad de Salamanca