

# ADSI – TEMA 4

---

Captura de requisitos

---

# Índice

- Definición
  - Proceso
  - Artefactos a obtener en la captura de requisitos
    - Modelo de Casos de Uso
    - Jerarquía de actores
    - Prototipos de interfaces de usuario
    - Glosario
    - Modelo del dominio
    - Descripción de la arquitectura
-

---

# Definición

- Consiste en saber qué sistema debe construirse
  - No es fácil:
    - El cliente no sabe lo que quiere
    - El cliente no sabe cómo expresar lo que quiere
  - Hay que usar un lenguaje común y sencillo para facilitar la comunicación
-

# Proceso

- Listar los requisitos candidatos
- Entender el contexto del sistema
- Capturar los requisitos funcionales
- Capturar los requisitos no funcionales

---

# Proceso

- Listar los requisitos candidatos
    - Aportar ideas de cómo cada uno ve el sistema y apuntarlas en una lista
    - Indicar si deben incorporarse al sistema o no
-

---

# Proceso

- Entender el contexto del sistema

- Modelado del dominio

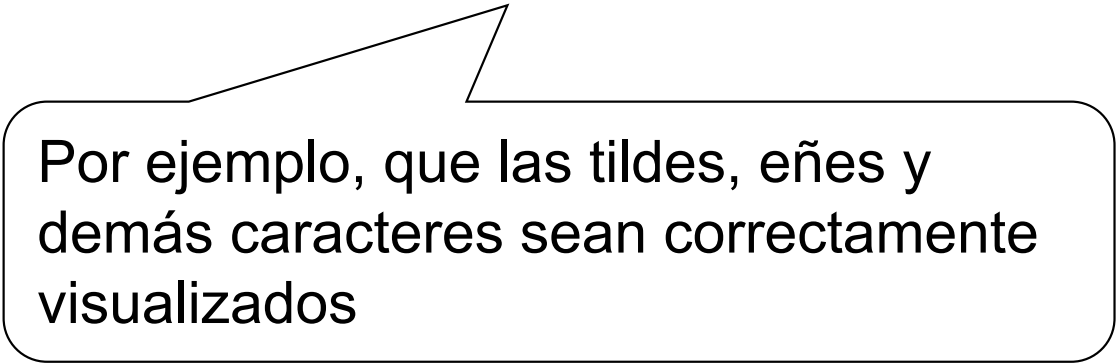
- Describir los “objetos” del dominio
    - Construir un glosario de términos

- Modelado del negocio

- Describir los procesos
-

# Proceso

- Capturar los requisitos funcionales
  - Encontrar los Casos de Uso
- Capturar los requisitos no funcionales
  - Son propiedades o restricciones del sistema
  - NO dicen qué hay que hacer, sino cómo hay que hacerlo



Por ejemplo, que las tildes, eñes y demás caracteres sean correctamente visualizados

---

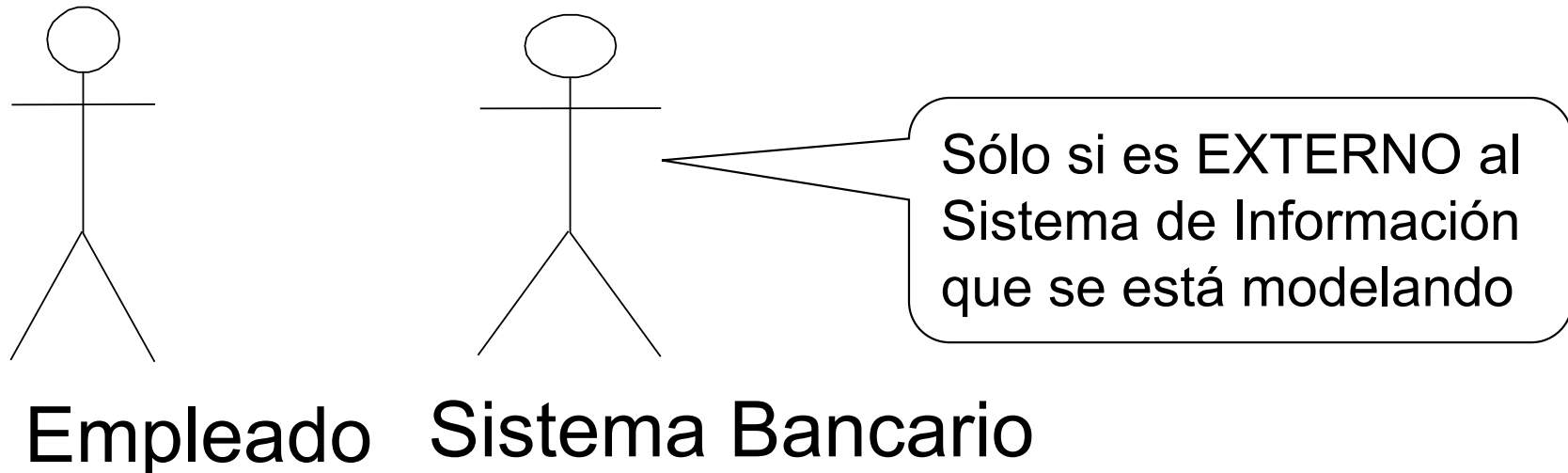
# Artefactos a obtener en la captura de requisitos

- Modelo de Casos de Uso
  - Jerarquía de Actores
  - Prototipos de interfaces de usuario
  - Casos de Uso extendidos
  - Glosario
  - Modelo del dominio
  - Descripción de la arquitectura
-



# Modelo de Casos de Uso

- Actor : Tipo de usuario o sistema EXTERNO que interactúa con el sistema a desarrollar



---

# Modelo de Casos de Uso

- Los actores representan **roles**, no personas
  - Es el sistema el que tiene que poder clasificar a las personas en esos roles
    - En base a los datos que proporcione el usuario
    - En base a qué parte del sistema se esté usando
    - ...
-

# Modelo de Casos de Uso

- Una misma persona puede usar el sistema bajo distintos roles
  - Por ejemplo:
    - Antes y después de identificarse
    - Como usuario y como administrador
    - ....
- Se suelen usar definiciones significativas, pero cuidado, que pueden llevar a error

No confundir qué es realmente una persona (por ejemplo, estudiante) con los roles bajo los que use el sistema

# Modelo de Casos de Uso

- Ejemplo: actores en eGela
  - **Usuario:** cualquier persona que accede a <https://egela.ehu.eus>
  - **Docente:** persona que se ha identificado correctamente en el sistema y tiene asignado un perfil de docente
  - **Estudiante:** persona que se ha identificado correctamente en el sistema y tiene asignado un perfil de estudiante

Cuando llegamos al sistema, todos somos **usuarios** y solo tras identificarnos pasamos a ser **estudiantes** o **docentes**

# Modelo de Casos de Uso

## ■ Tipos de actores

- Primario: el que interactúa con el sistema y aporta la información
- Secundario: el que recibe información del sistema

# Modelo de Casos de Uso

- Caso de Uso: Cada una de las formas que un actor tiene de usar el sistema
  - Tienen que ser acciones que por sí solas tengan sentido en el sistema a desarrollar

Lo que en un sistema tiene sentido, en otro puede no tenerlo

Identificarse

Introducir usuario

Introducir Contraseña

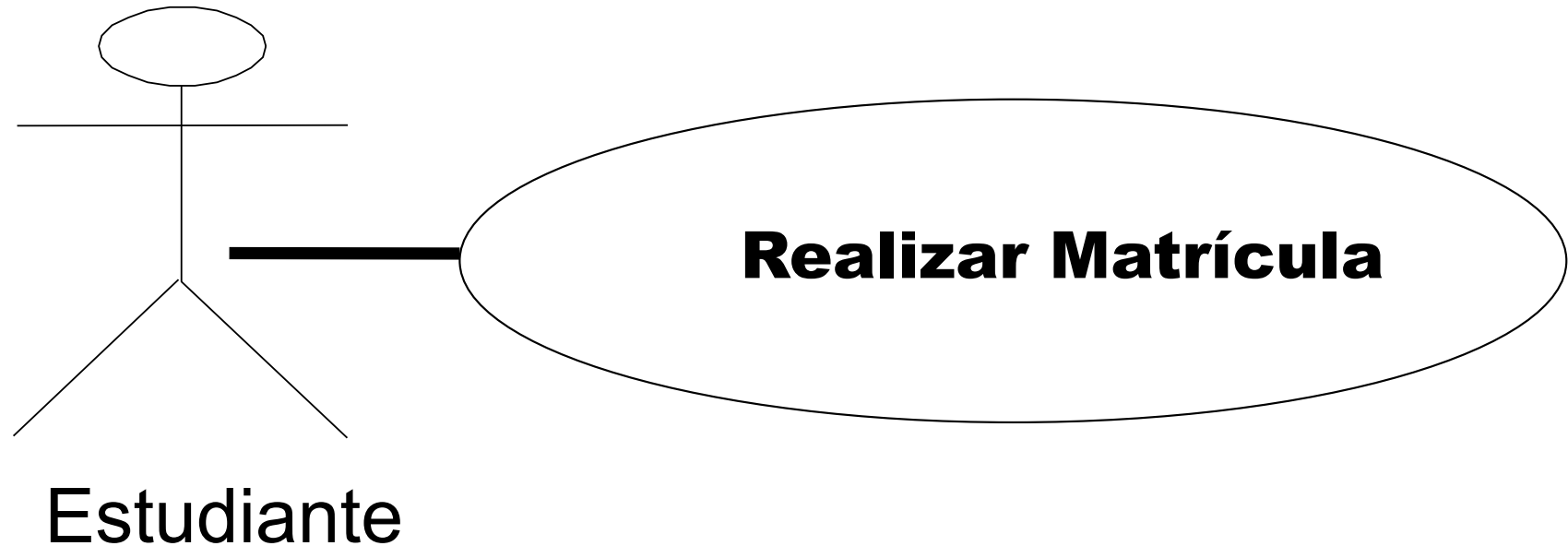
# Modelo de Casos de Uso

- Cada caso de uso tiene asociado:
  - Un Flujo de eventos: Descripción de cómo funciona el caso de uso paso a paso
  - Requisitos especiales: Descripción textual de los requisitos no funcionales

No se hace referencia a la tecnología, la descripción de la funcionalidad es independiente de cómo se implemente finalmente

# Modelo de Casos de Uso. Ejemplo

- Un estudiante tiene que poder matricularse





# Modelo de Casos de Uso. Ejemplo

- Flujo de eventos (o sucesos)
  - El estudiante proporciona su DNI
  - Si el DNI no existe se muestra un aviso de error
  - Si el DNI existe
    - El sistema muestra todas las asignaturas en las que puede matricularse y que, de momento, no están completas
    - El estudiante escoge las asignaturas que desea
    - El sistema almacena los datos de la matrícula

---

# Modelo de Casos de Uso. Ejemplo

## ■ Requisitos no funcionales

- ❑ Tiene que permitir la ejecución de 500 matrículas simultáneamente
- ❑ Los datos de matriculación y plazas libres en cada asignatura tienen que actualizarse en tiempo real

# Modelo de Casos de Uso

- El actor primario de un caso de uso siempre es el que aporta la información

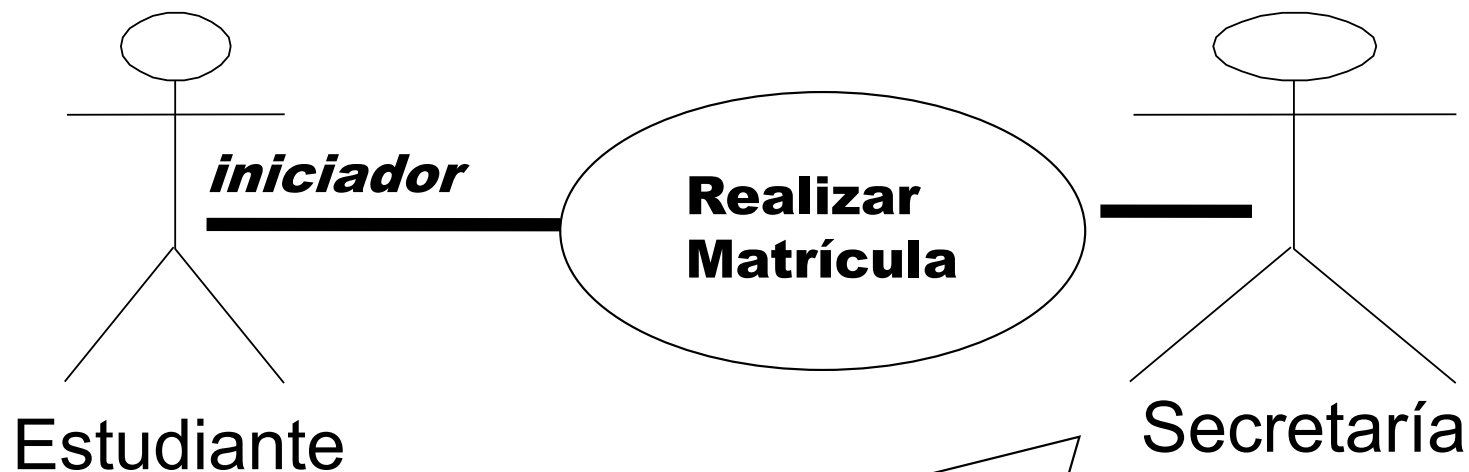
Aportar ≠ Teclear

- Quién aporta la información es un hecho, quién teclea la información es una decisión

En un supermercado, ¿se ponen cajas de autopago?  
En la universidad, ¿se permite la automatrícula?

# Modelo de Casos de Uso

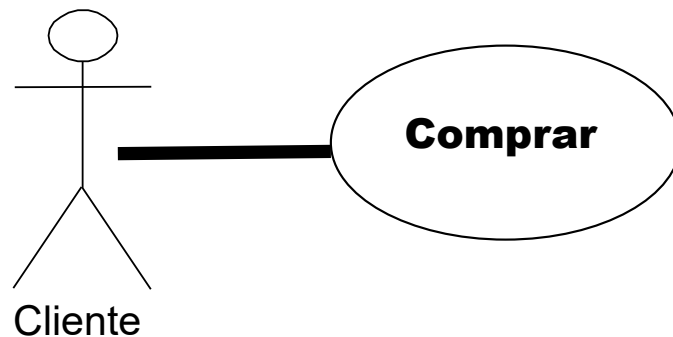
- 2 o más actores primarios en un Caso de Uso
- Se indica quién lo inicia



Solo si el personal de Secretaría aporta **información** al caso de uso

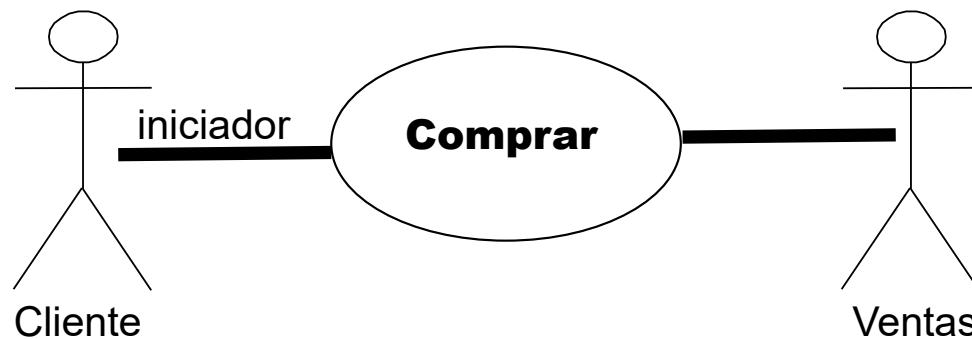
# Modelo de Casos de Uso. Ejemplo

## ■ Tienda sin comisión para quien vende



El caso de uso se relaciona con el actor que proporciona la **información**, no con quien la teclea. Sin cliente no hay compra, sin personal de ventas, sí.

## ■ Tienda con comisión para quien vende



Para cobrar la comisión el personal de ventas debe introducir algún tipo de identificación. Es imprescindible que participe.

---

# Modelo de Casos de Uso

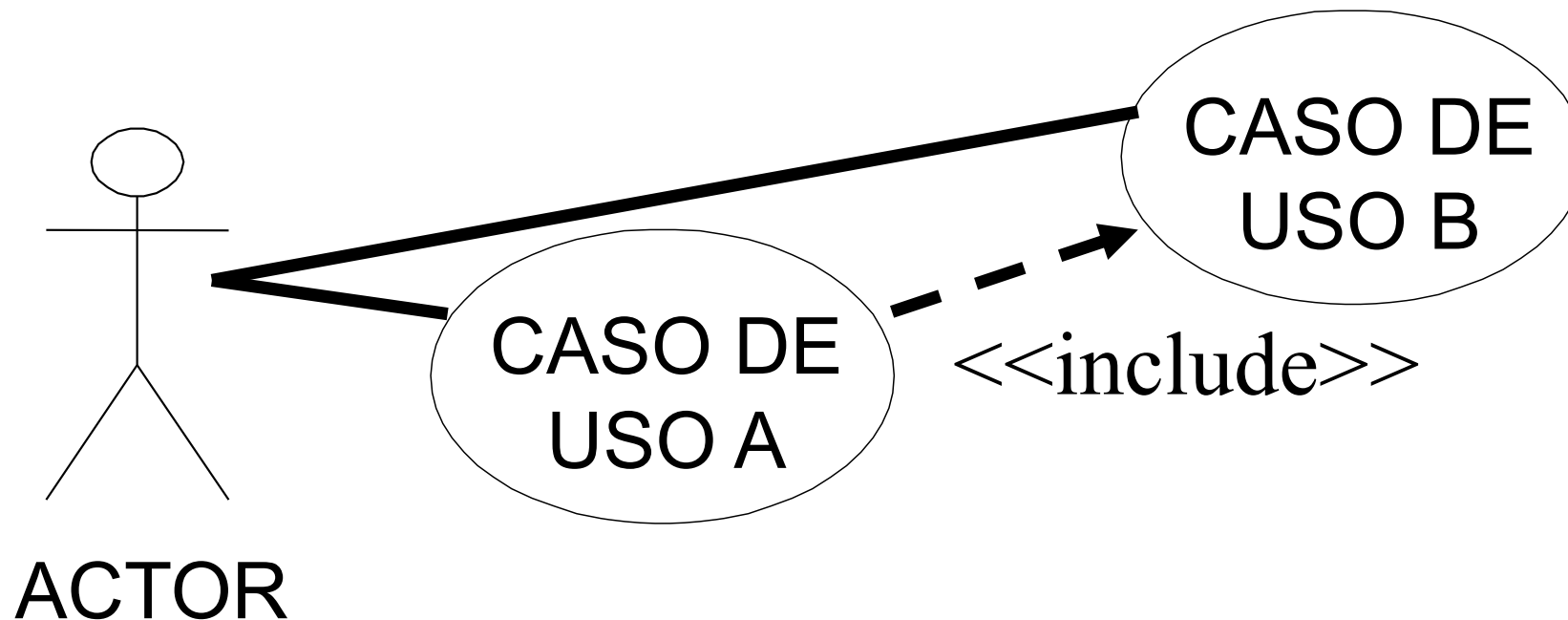
## ■ Relaciones entre Casos de Uso

- Ambos Casos de Uso deben tener sentido en el sistema
  - 2 tipos de relaciones:
    - Include (anteriormente llamado uses o includes)
    - Extend (anteriormente llamado extends)
-

# Modelo de Casos de Uso

## ■ Include

- El Caso de Uso A incluye al Caso de Uso B, si **SIEMPRE** que se ejecuta A, se ejecuta B



# Modelo de Casos de Uso

## ■ Include

- En el código de la funcionalidad A habrá una llamada al código de la funcionalidad B
  - La llamada puede estar en cualquier punto

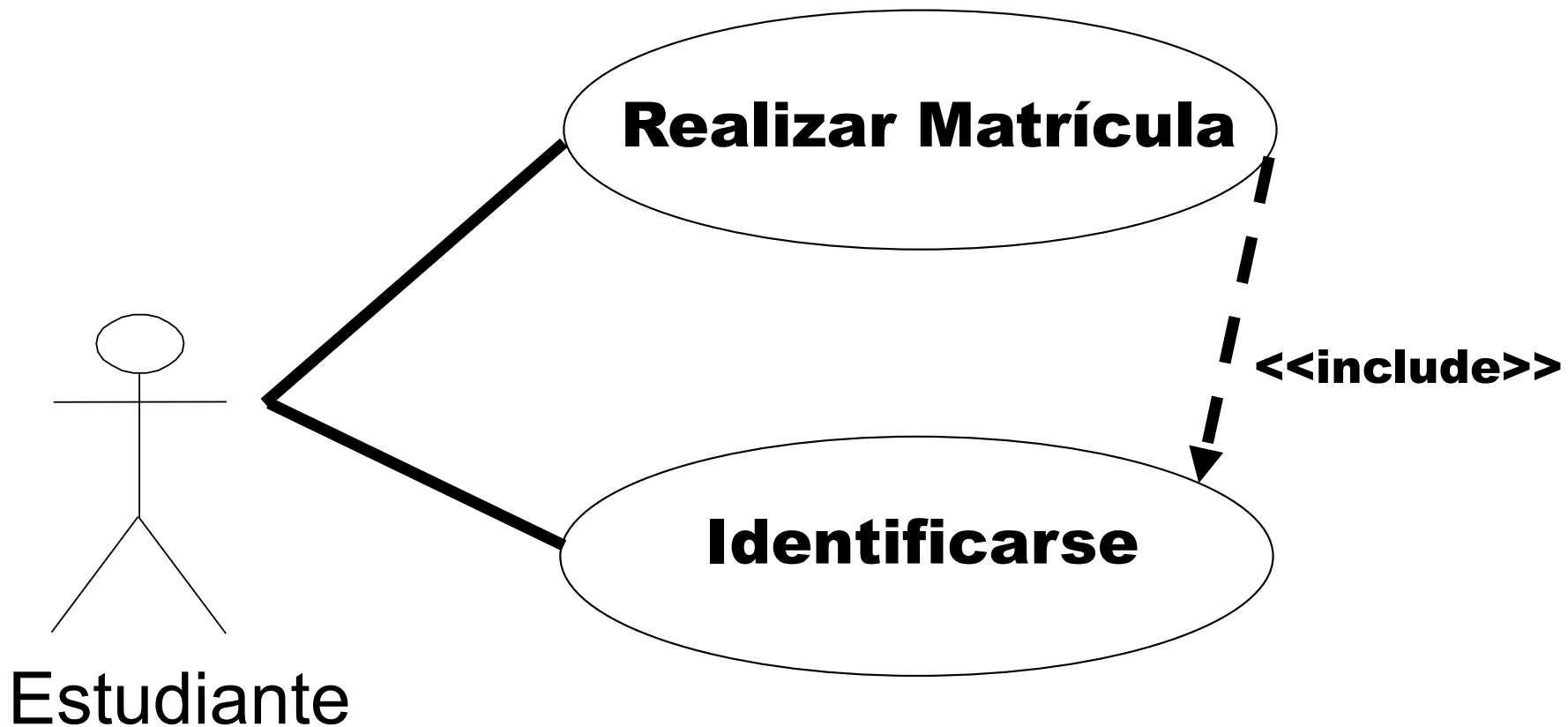
```
A {  
    B()  
    ...  
    ...  
}
```

```
A {  
    ...  
    B()  
    ...  
}
```

```
A {  
    ...  
    ...  
    B()  
}
```



# Modelo de Casos de Uso. Ejemplo



---

# Modelo de Casos de Uso. Ejemplo

- Flujo de eventos (Identificarse)
    - El/la estudiante introduce su número de identificación y su contraseña y pulsa “Aceptar”
    - Si no son correctos se muestra pantalla de error
-

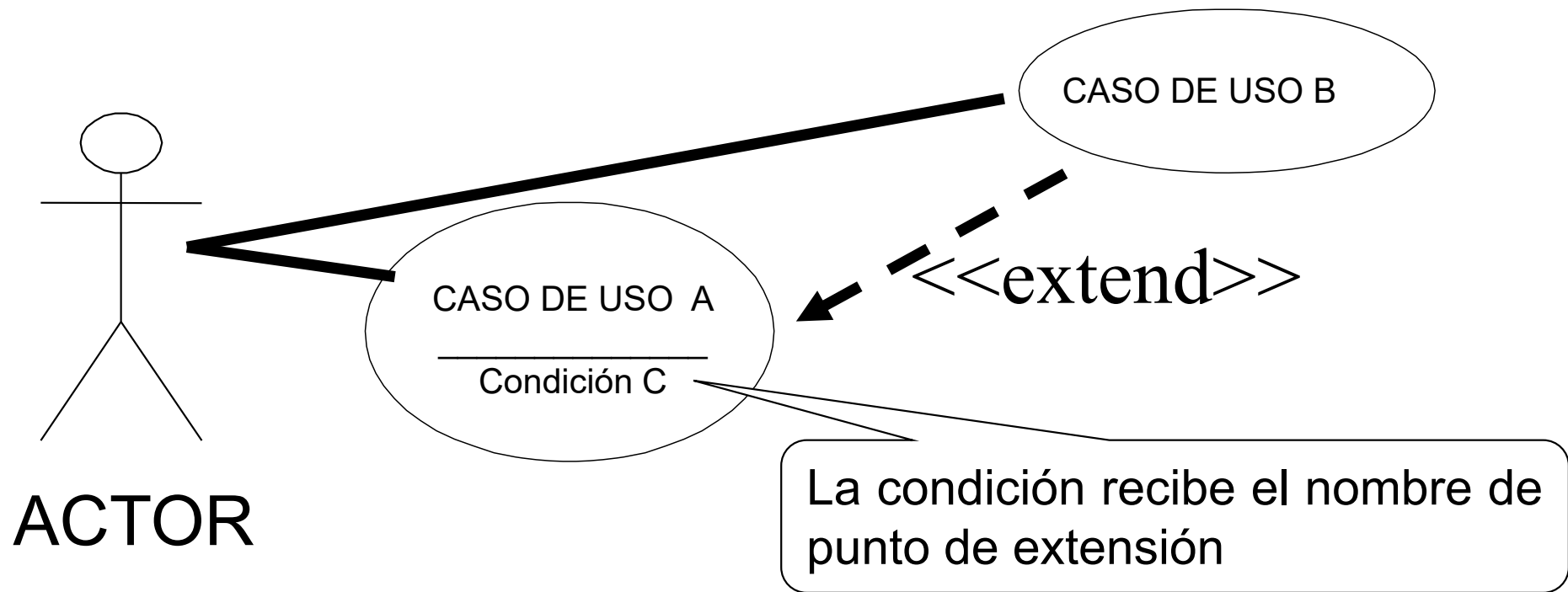
# Modelo de Casos de Uso. Ejemplo

- Flujo de eventos (Realizar Matrícula)
  - El/la estudiante pincha en “Ver Asignaturas”
  - Se muestra el listado de todas las asignaturas
  - El/la estudiante selecciona una asignatura y pulsa “Matricular”
  - INCLUDE “Identificarse”
  - Se almacenan los datos de la matrícula

# Modelo de Casos de Uso

## ■ Extend

- El Caso de Uso B extend al Caso de Uso A, si al ejecutar A y cumplirse la condición C, se ejecuta B



# Modelo de Casos de Uso

## ■ Extend

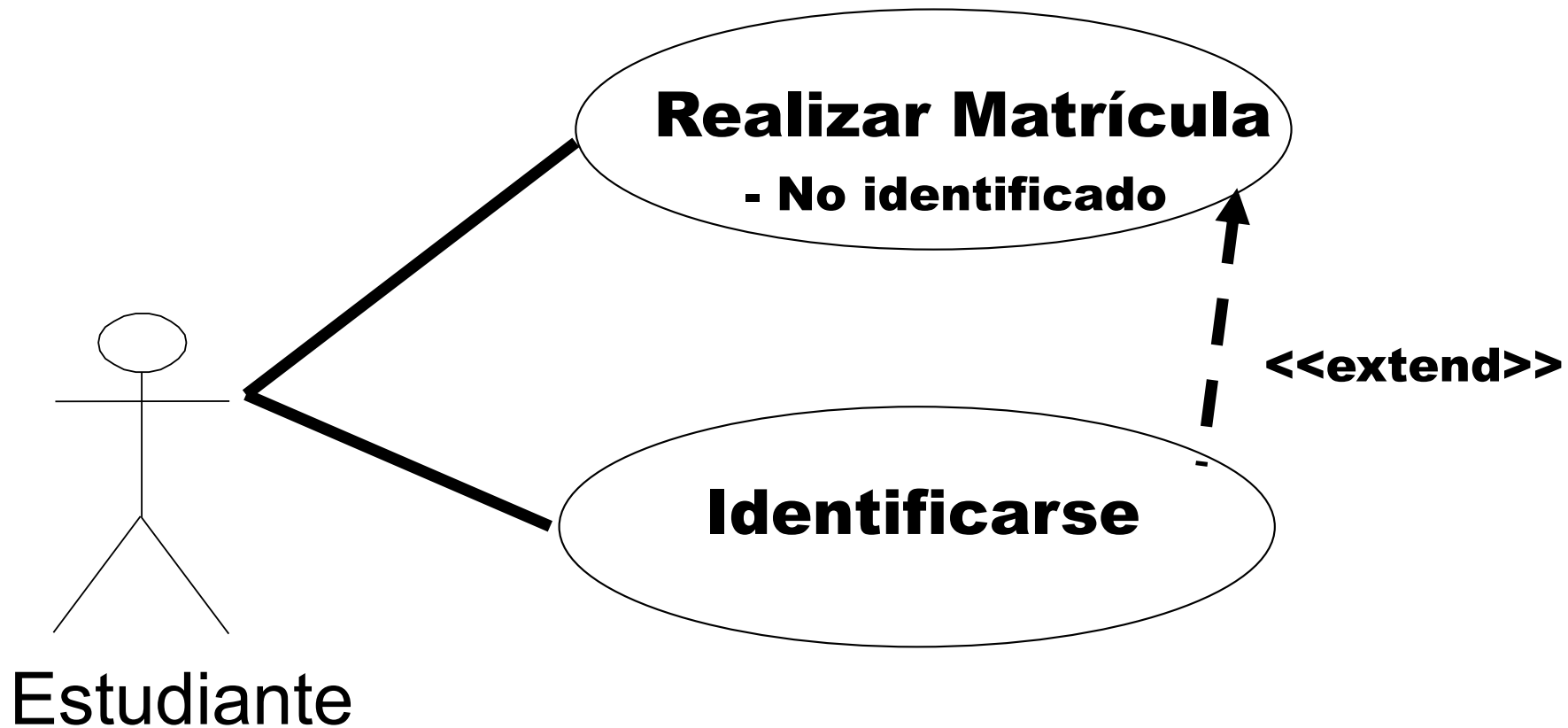
- En el código de la funcionalidad A habrá una condición con llamada al código de la funcionalidad B
- La condición puede estar en cualquier punto

```
A {  
  if (C) {  
    B()  
  }  
  ...  
  ...  
}
```

```
A {  
  ...  
  if (C) {  
    B()  
  }  
  ...  
}
```

```
A {  
  ...  
  ...  
  if (C) {  
    B()  
  }  
}
```

# Modelo de Casos de Uso. Ejemplo



---

# Modelo de Casos de Uso. Ejemplo

- Flujo de eventos (Identificarse)
    - El/la estudiante introduce su número de identificación y su contraseña y pulsa “Aceptar”
    - Si no son correctos se muestra pantalla de error
-

# Modelo de Casos de Uso. Ejemplo

- Flujo de eventos (Realizar Matrícula)
  - El/la estudiante pincha en “Ver Asignaturas”
  - Se muestra el listado de todas las asignaturas
  - Si el/la estudiante no está identificado
    - EXTEND “Identificarse”
  - Almacenar los datos de la matrícula



# Modelo de Casos de Uso

- Cuando se relacionan dos casos de uso, se comportan como si fuera un único caso de uso
  - Deben ejecutarse de manera secuencial
  - La ejecución debe depender de una única persona (aunque sea bajo distintos roles) o involucrar otros sistemas externos

Puede involucrar distintas personas si podemos asegurar que la segunda persona va a estar 24x7 esperando para ejecutar su parte en el mismo instante que le corresponda.

# Modelo de Casos de Uso

## ■ Subcasos de Uso

- Funcionalidades que NO son Casos de Uso, pero que por comodidad se van a tratar como tal

- Mayor comprensión del modelo
- Reutilización

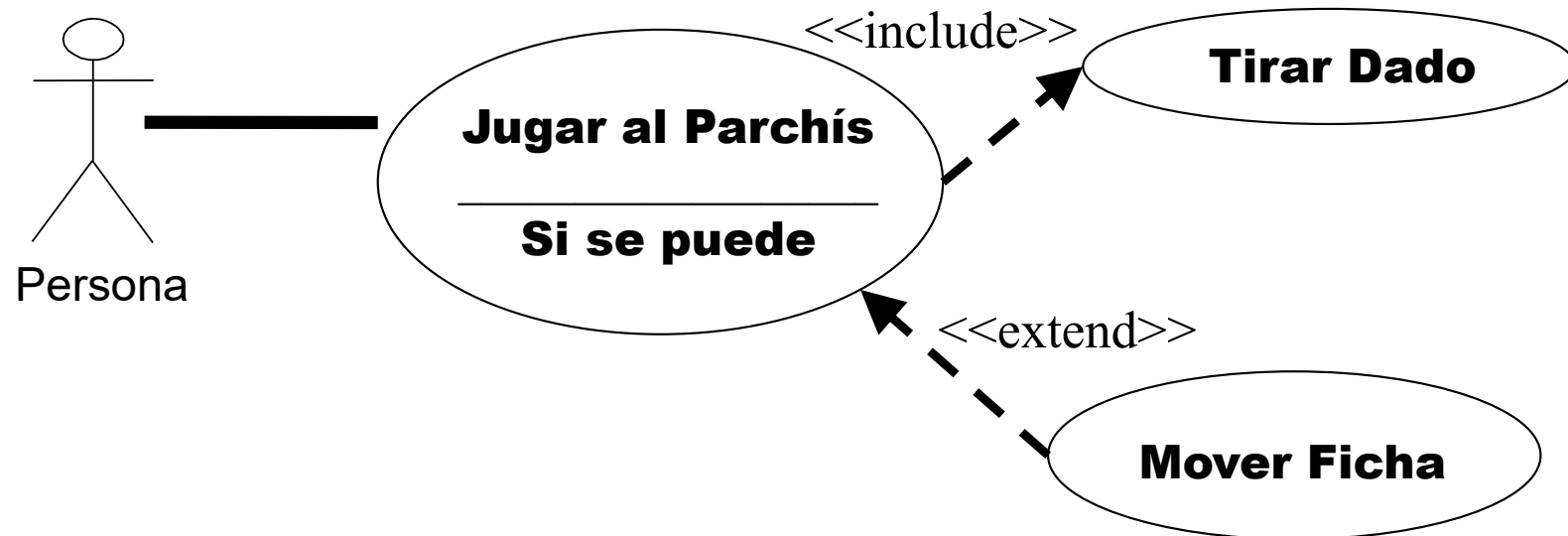
Se documentarán como si fueran casos de uso

- No se pueden iniciar de manera independiente
- No estarán relacionados con ningún actor primario

# Modelo de Casos de Uso. Ejemplo

## ■ Subcasos de Uso

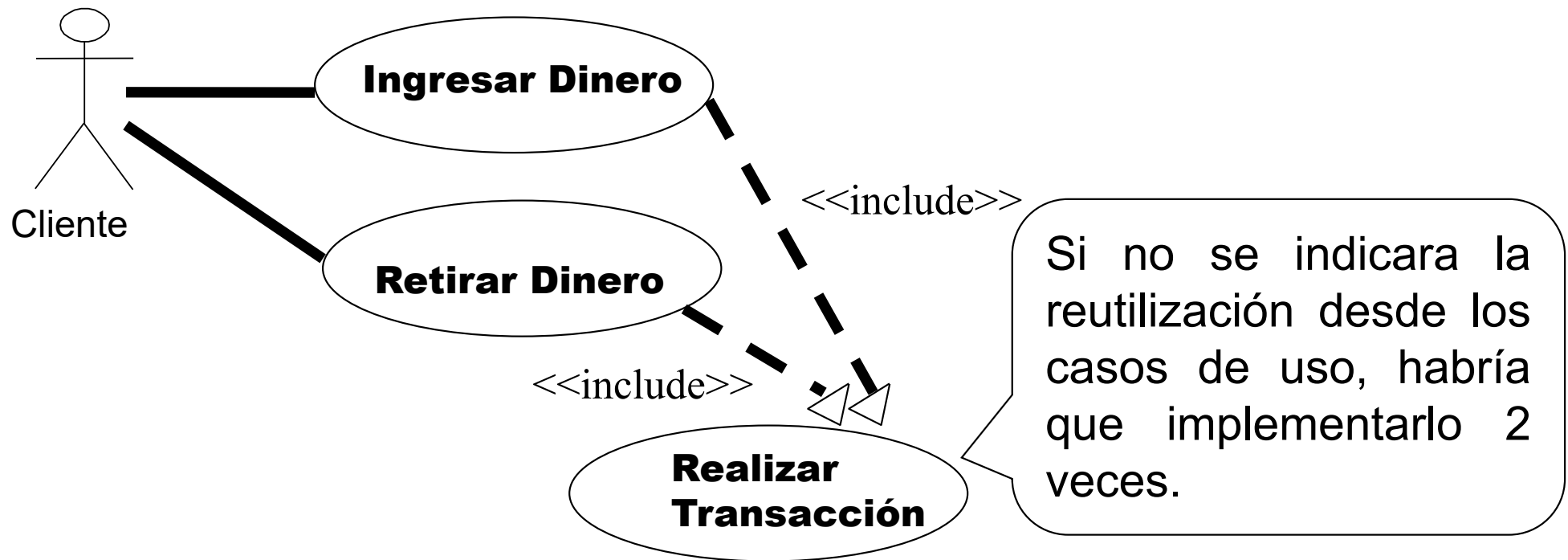
- Para entender mejor el sistema y simplificar la documentación



# Modelo de Casos de Uso. Ejemplo

## ■ Subcasos de Uso

- Para reutilizar parte de la funcionalidad



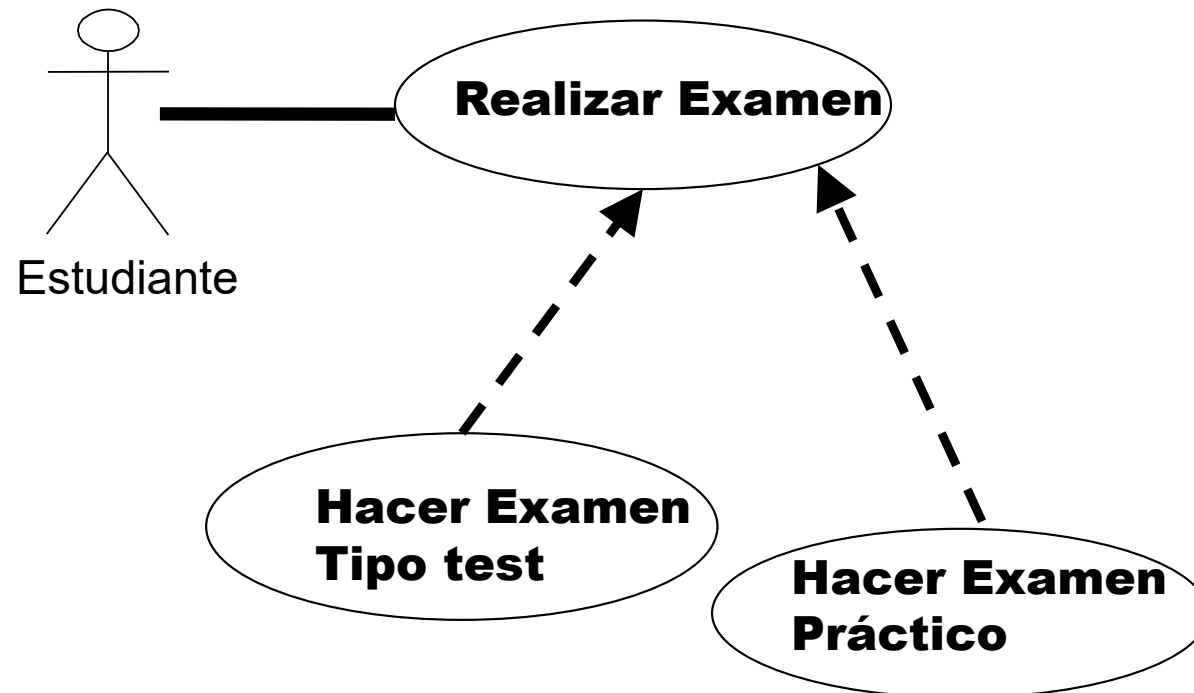
# Modelo de Casos de Uso

## ■ Casos de Uso abstractos

- ❑ Caso de Uso con especializaciones para indicar ejecuciones alternativas
- ❑ Las especializaciones no son Casos de Uso como tal, pero se tratarán de igual modo
- ❑ El estereotipo de la relación es la generalización

# Modelo de Casos de Uso. Ejemplo

- Casos de Uso abstractos
  - Compartiendo parte de la funcionalidad (que no es lo mismo que reutilizarla)



# Modelo de Casos de Uso. Ejemplo

- Flujo de eventos (Realizar Examen)
  - El/la estudiante pincha en “Hacer Examen”
  - Se muestra el listado con los exámenes pendientes
  - El/la estudiante selecciona un examen y pulsa “Realizar”
  - Si el examen es de tipo test
    - CU “Hacer Examen Tipo Test”
  - Si el examen es práctico
    - CU “Hacer Examen Práctico”

# Modelo de Casos de Uso. Errores

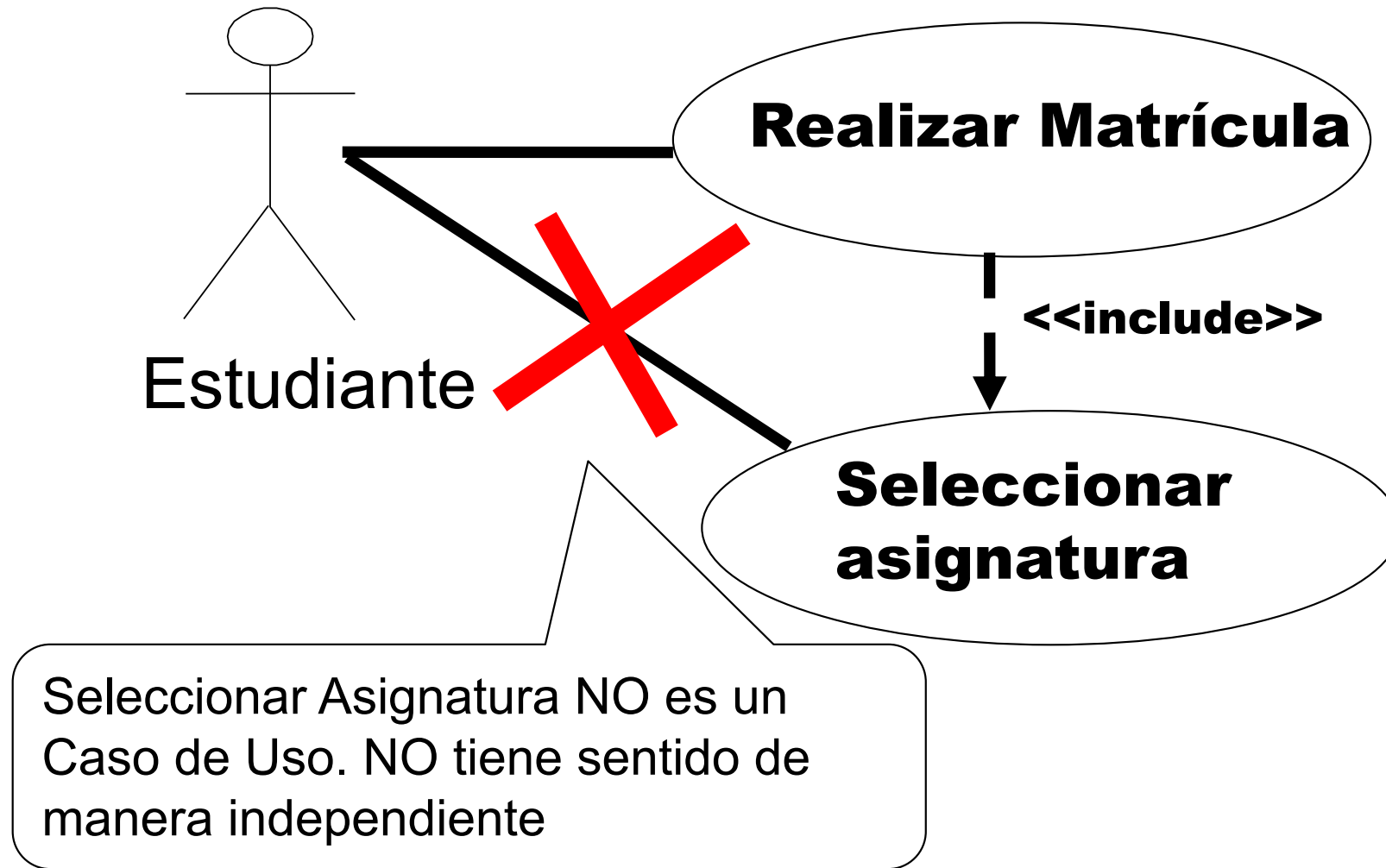
- Definir CU que no lo son
  - ❑ No hay actor que lo ejecute
  - ❑ Es un procedimiento interno del sistema
  - ❑ Es un procedimiento ajeno al sistema
  - ❑ Ocurre normalmente al “buscar” include o extend

**REGLA DE ORO: Un CU es una funcionalidad DEL SISTEMA que proporciona algún RESULTADO o VALOR a por lo menos un ACTOR**



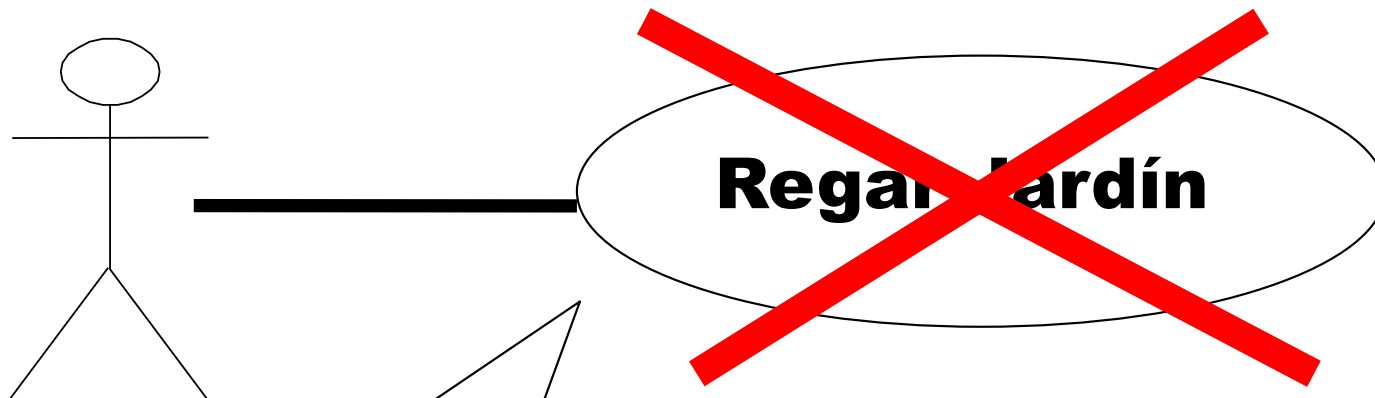
# Modelo de Casos de Uso. Errores

- Definir CU que no lo son



# Modelo de Casos de Uso. Errores

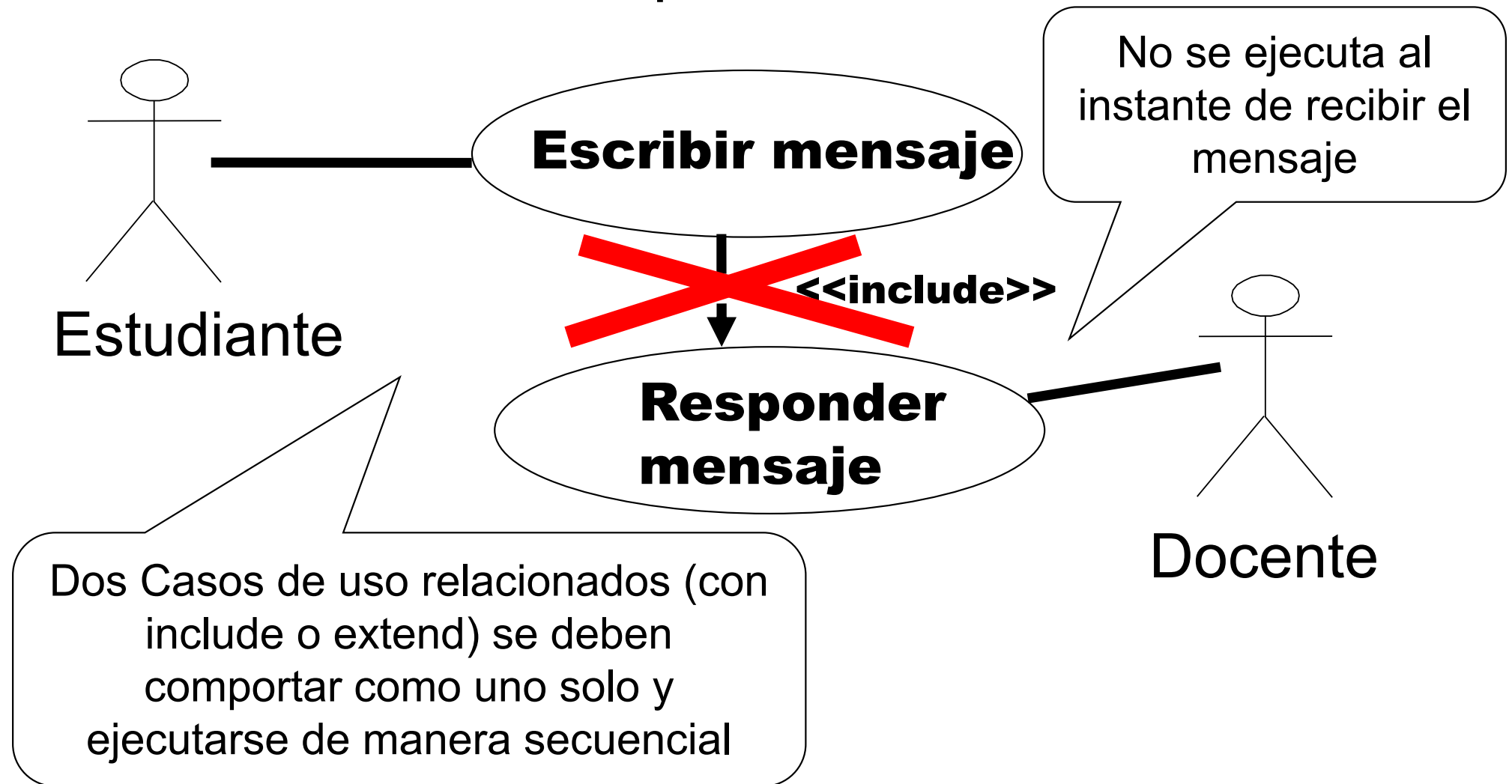
- Definir CU que no lo son
  - “...el usuario coge la manguera y riega el jardín”



Regar Jardín NO es un Caso de Uso. No tiene nada que ver con el sistema

# Modelo de Casos de Uso. Errores

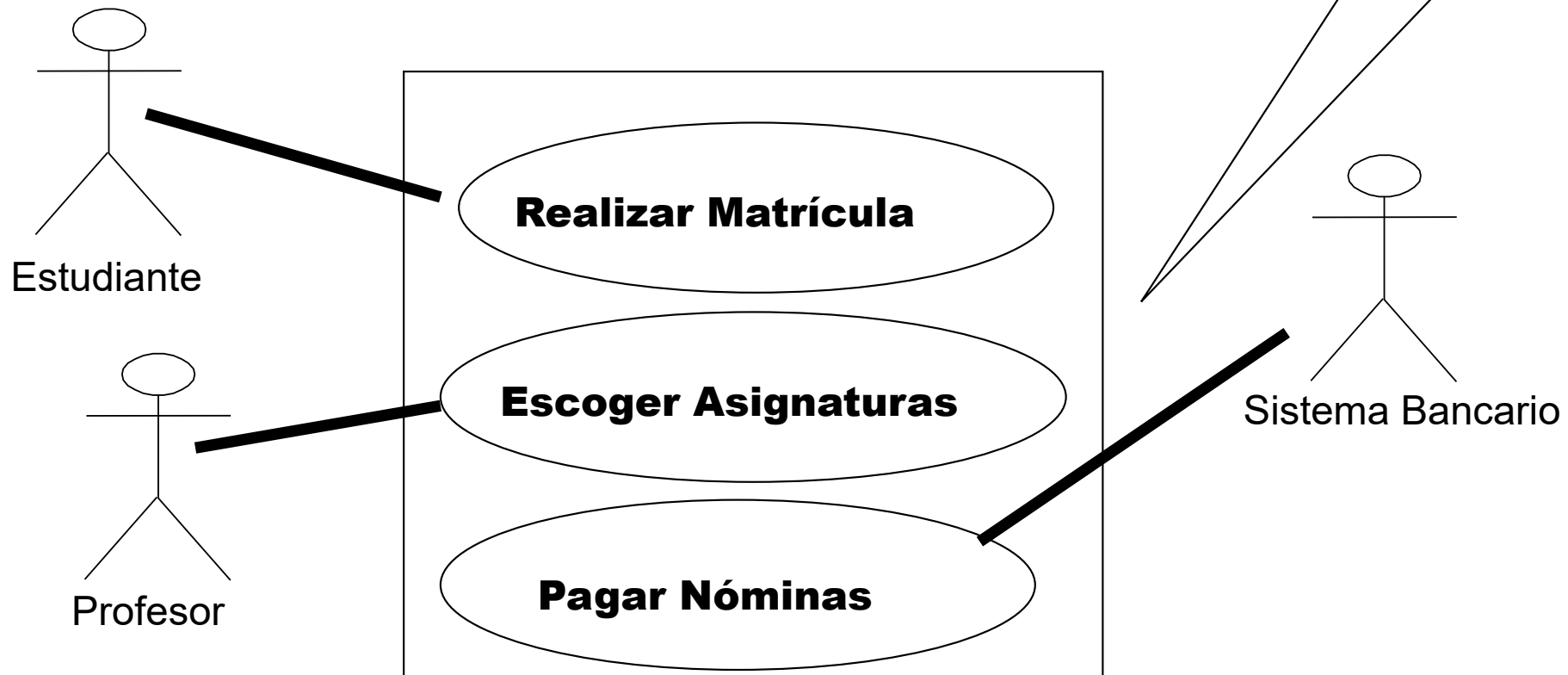
- Definir relaciones que no lo son



# Modelo de Casos de Uso

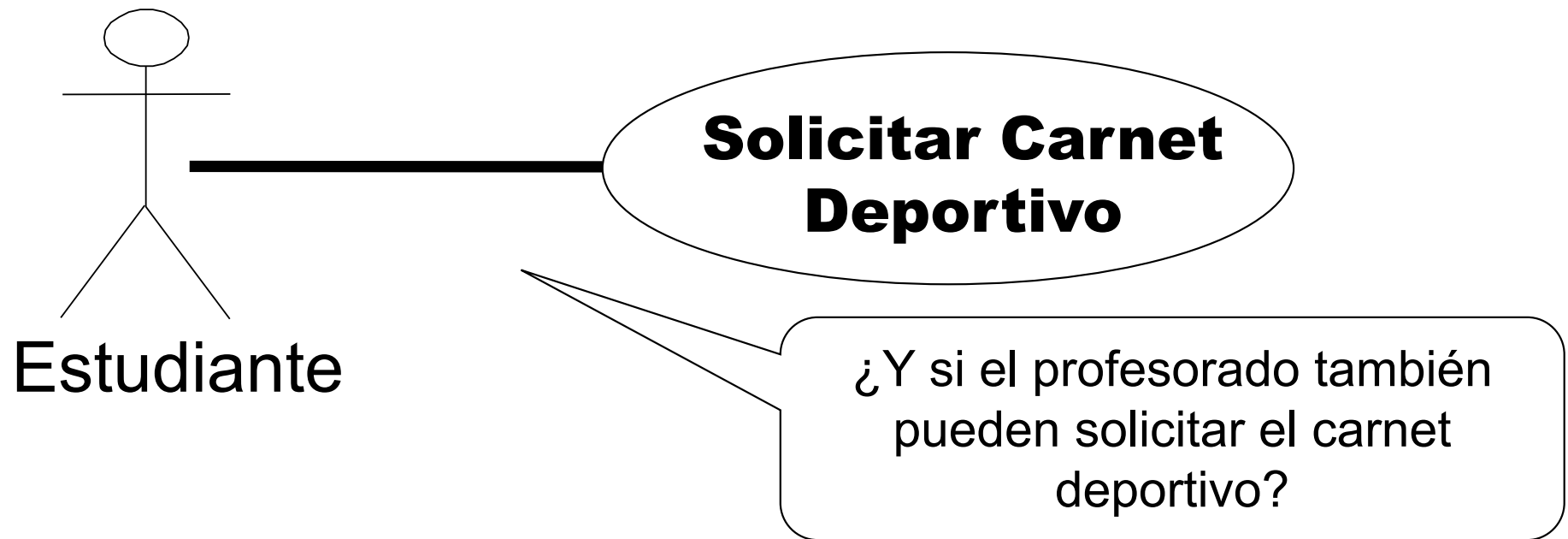
## ■ Modelo de Casos de Uso

- Casos de Uso para todos los actores

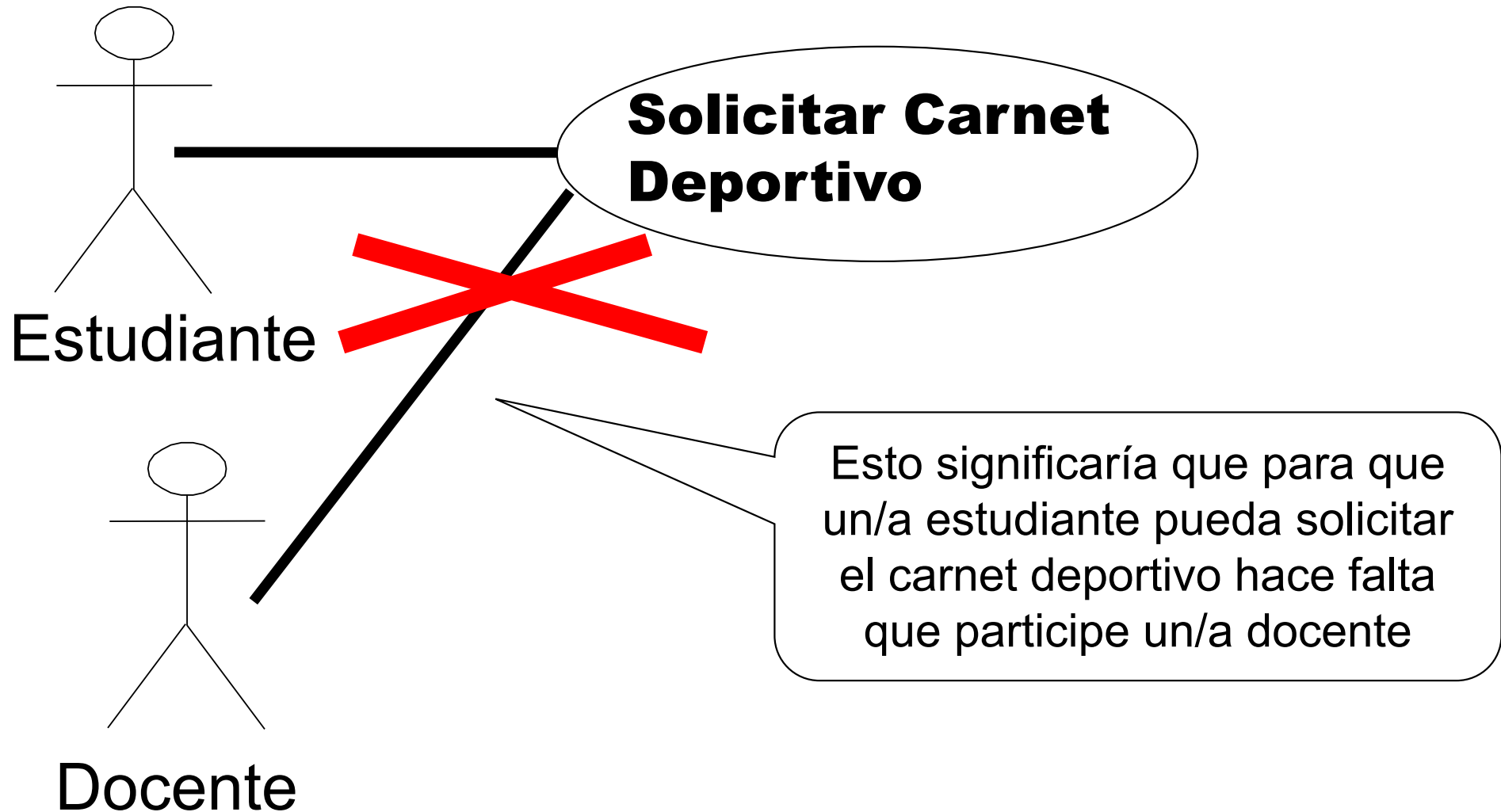


# Jerarquía de actores

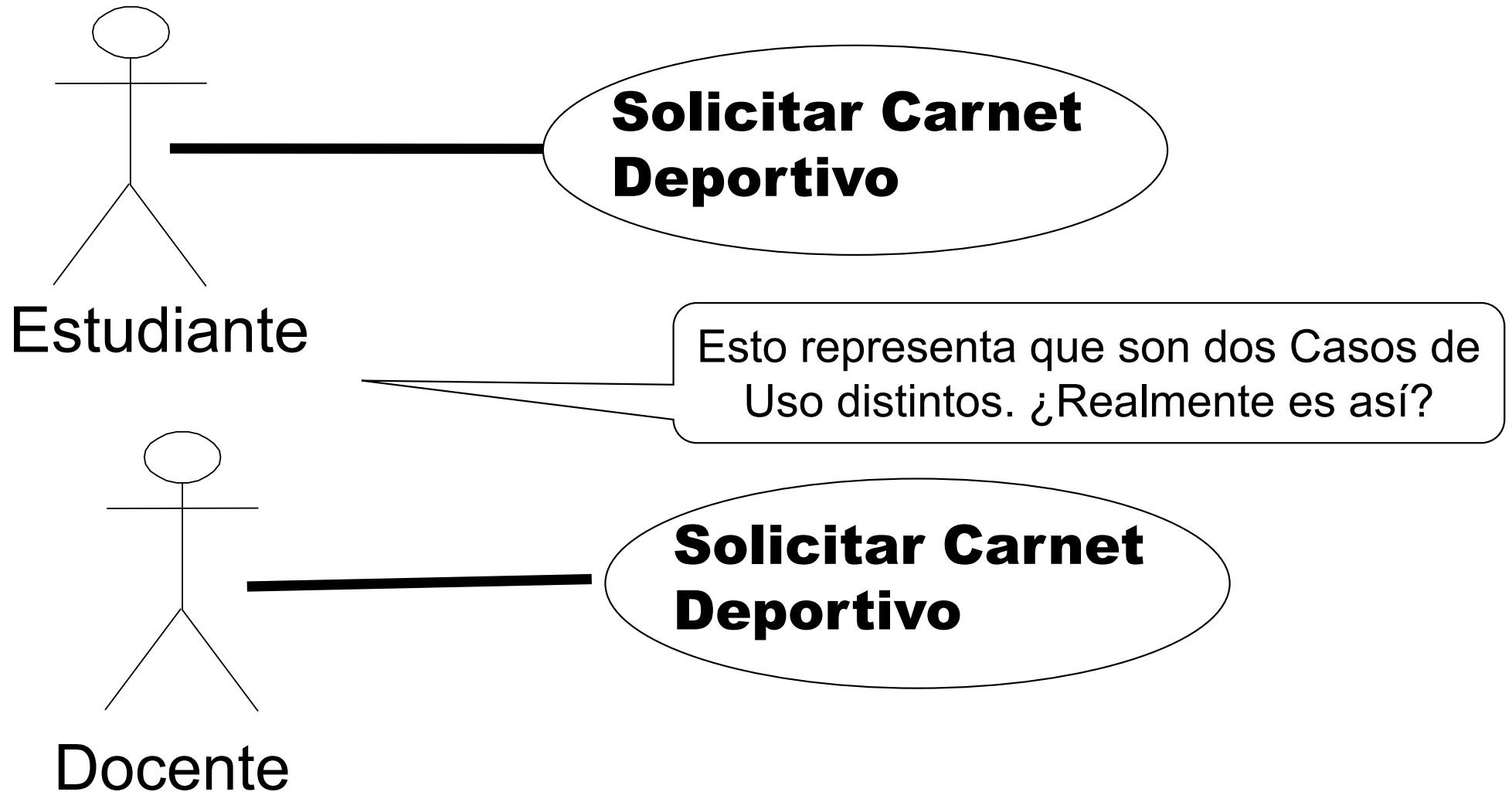
- Modelo que expresa cómo se relacionan los actores en el sistema EN BASE a los Casos de Uso de cada actor



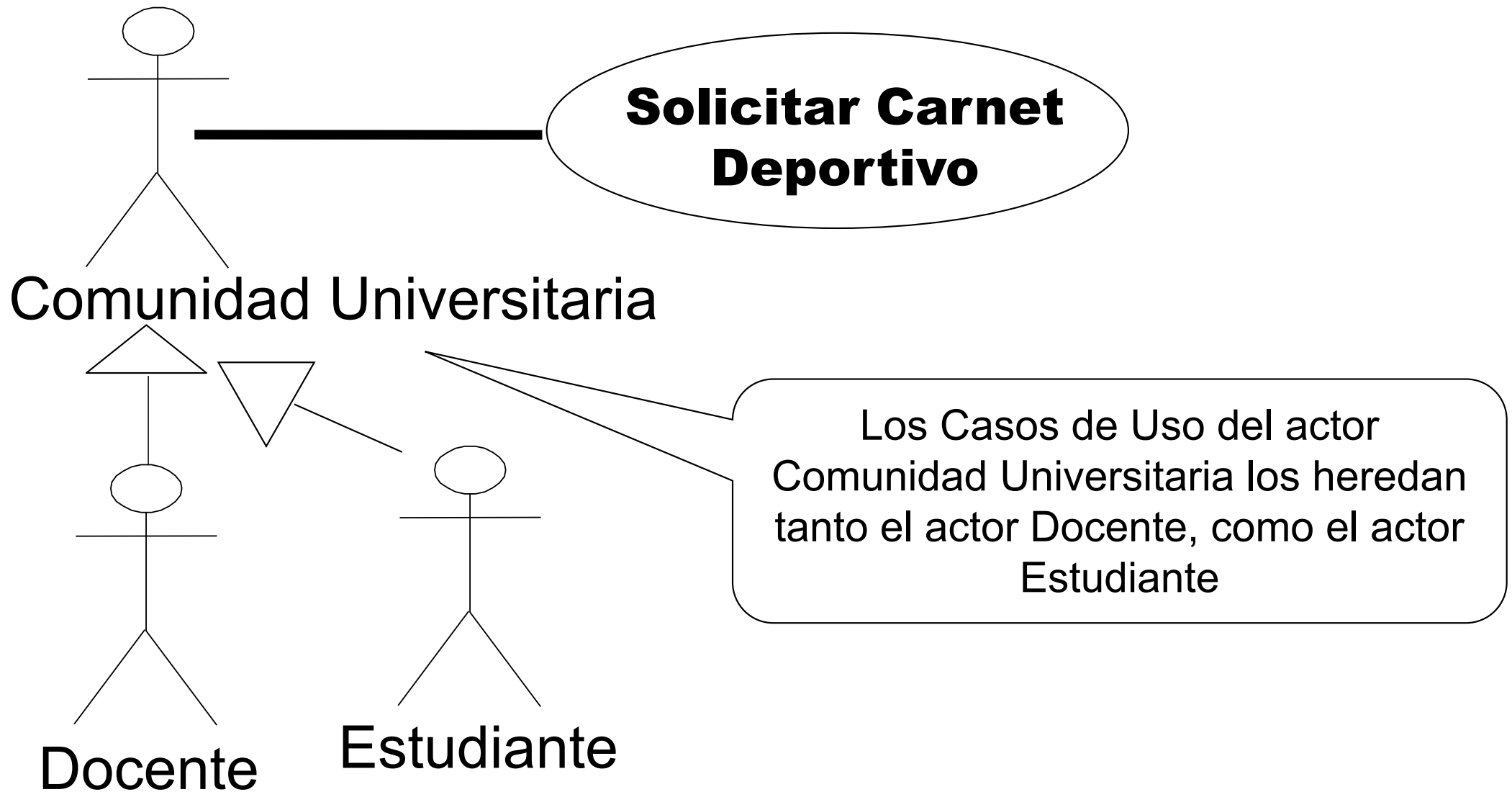
# Jerarquía de actores



# Jerarquía de actores



# Jerarquía de actores





---

# Prototipos de interfaces de usuario

- Imagen aproximada de la interfaz gráfica del sistema
  - Ayuda a entender la interacción con el sistema y permite obtener mejores interfaces gráficas
  - Ayuda a capturar los Casos de Uso
-

# Prototipos de interfaces de usuario.

## Ejemplo

**CASO DE USO: TOMAR COPIA LIBRO EN PRÉSTAMO**

SIGNATURA LIBRO:

NÚMERO SOCIO:

Área de texto donde aparecerá el número de copia del libro que se ha tomado en préstamo.

Si no hay ninguna libre o si el socio ha sobrepasado su número máximo de préstamos entonces se indicará aquí mismo.

TOMAR EN PRÉSTAMO

RESERVAR LIBRO

Cancel

# Casos de Uso extendidos

- Documento donde por cada Caso de Uso se definen
  - ❑ Descripción
  - ❑ Flujo de eventos
  - ❑ Requisitos no funcionales
  - ❑ Precondición / Postcondición
  - ❑ Prototipo de la interfaz

# Casos de Uso extendidos

## ■ Precondición

- Qué tiene que ocurrir para que el Caso de Uso pueda ejecutarse

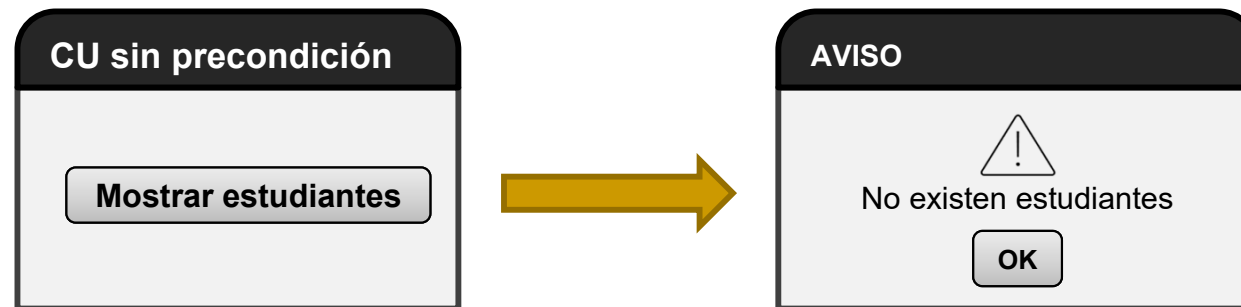
Mientras la precondición no se cumpla,  
el CU “no existe”.  
No confundir con dar error o avisar

## ■ Postcondición

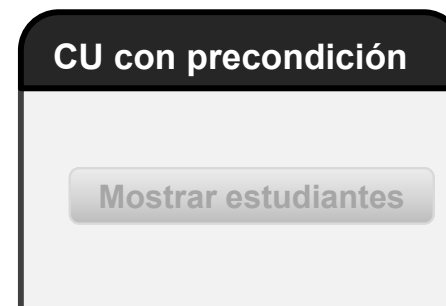
- Refleja el estado en el que se queda el sistema (los datos) tras la ejecución del Caso de Uso

# Casos de Uso extendidos

- Sin precondition

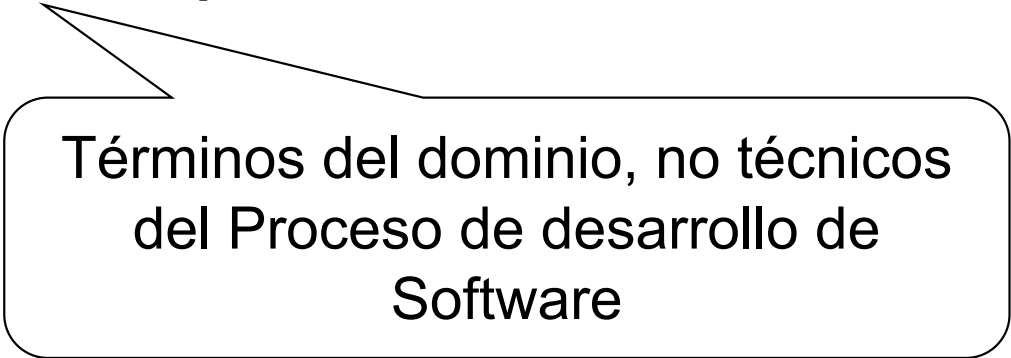


- Con precondition “existen estudiantes”, mientras no haya estudiantes, el botón está deshabilitado / oculto



# Glosario

- Documento donde se definen los términos más comunes e importantes utilizados



Términos del dominio, no técnicos  
del Proceso de desarrollo de  
Software

- Su objetivo es que todo el mundo sepa con exactitud a qué se refiere cada término

# Glosario. Ejemplo

- **ASIGNATURA:** ...
- **ESTUDIANTE:** es una persona que está estudiando una carrera en la universidad UnivX. Necesariamente debe estar matriculada en por lo menos una ASIGNATURA.
- **MATRÍCULA:** es el resultado de un proceso administrativo por el cual un/a ESTUDIANTE adquiere el derecho a ser evaluado/a en dos convocatorias de una ASIGNATURA. Se le asocia a un GRUPO. Tiene derecho a asistir a las clases del DOCENTE responsable de dicha ASIGNATURA en el GRUPO asignado.
- **DOCENTE:** es una persona que trabaja en UnivX y que imparte al menos una asignatura de una determinada TITULACIÓN. Se encarga de evaluar a todo el alumnado matriculado en la asignatura y asignado a sus grupos. El/la docente no puede ser estudiante en la misma carrera en la que imparte clases, pero sí en otras.

---

# Un Caso de Uso especial: “Identificarse”

---

Diferencia entre “include”, “extend” y  
precondición



# Sistema de ejemplo

- Las funcionalidades que el sistema tiene que ofertar son:
    - Modificar Datos Personales
    - Realizar Operación
    - Consultar Información General
  - Las dos primeras exigen tener identificado al usuario.
  - “Identificarse” puede ser Caso de Uso o no.
-

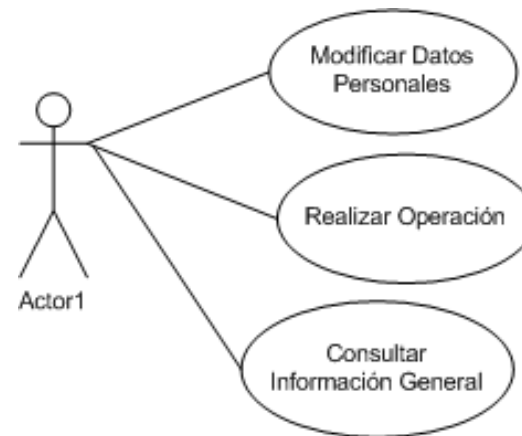
---

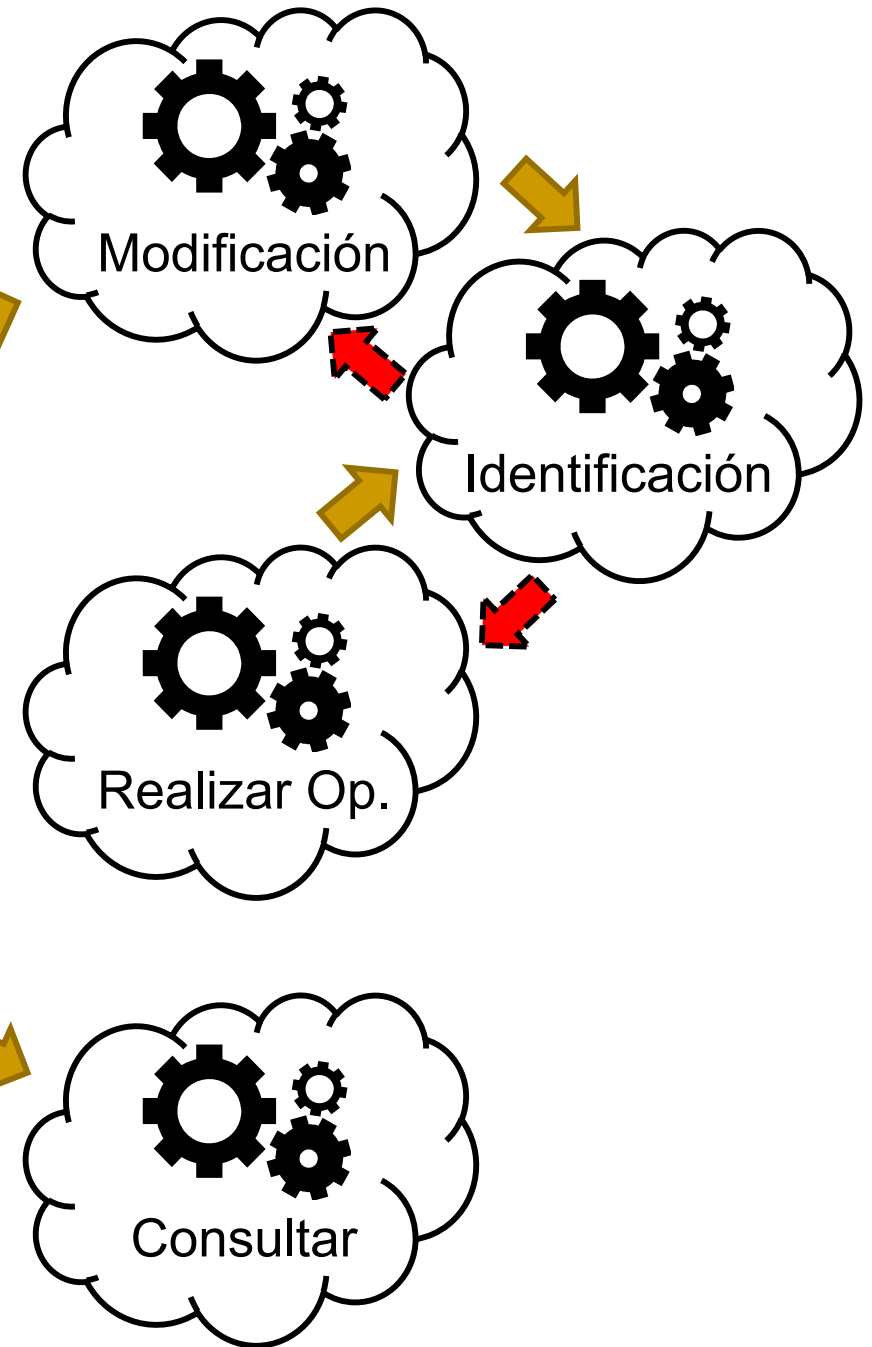
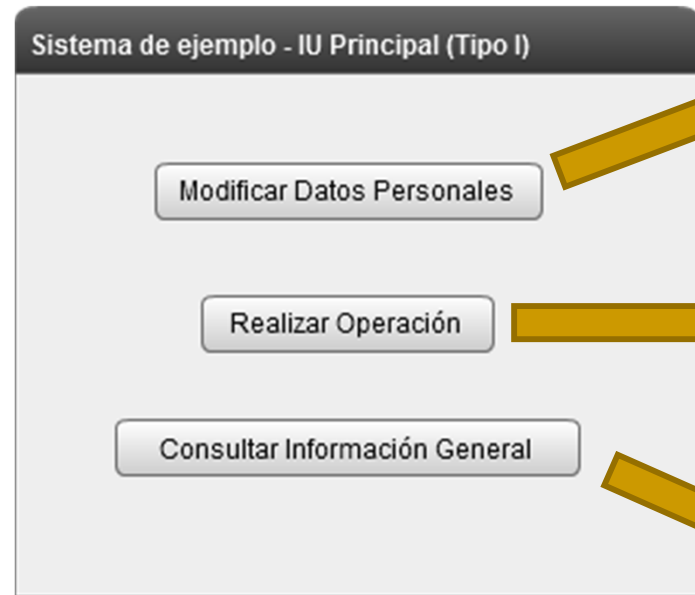
¿Cómo queremos que  
funcione el sistema?

---

# Posibilidad I

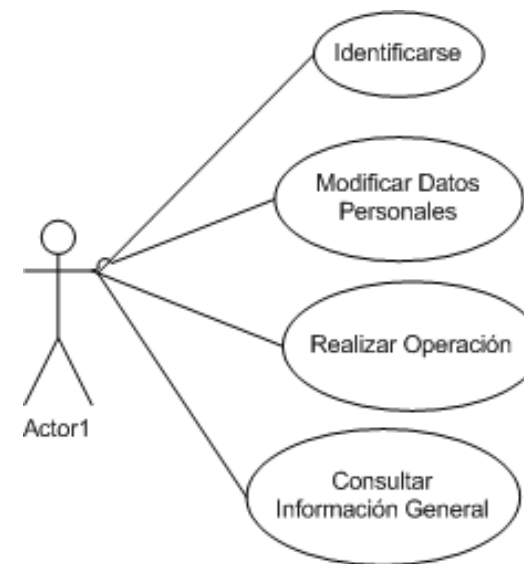
- “Identificarse” NO será Caso de Uso si:
  - ❑ No se puede ejecutar de manera independiente al resto de Casos de Uso.
  - ❑ No existe una opción para ejecutar la identificación sin entrar en la ejecución de otro Caso de Uso.





# Posibilidades II, III y IV

- “Identificarse” será Caso de Uso si:
  - ❑ Se puede ejecutar de manera independiente al resto de Casos de Uso.
  - ❑ Debe existir una opción para “Identificarse” sin tener que hacer nada más



Sistema de ejemplo - IU Principal (Tipo II)

Modificar Datos Personales

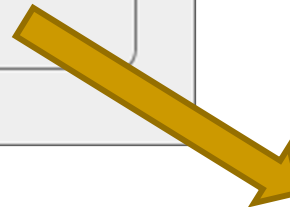
Realizar Operación

Consultar Información General

Usuario

Contraseña

Aceptar

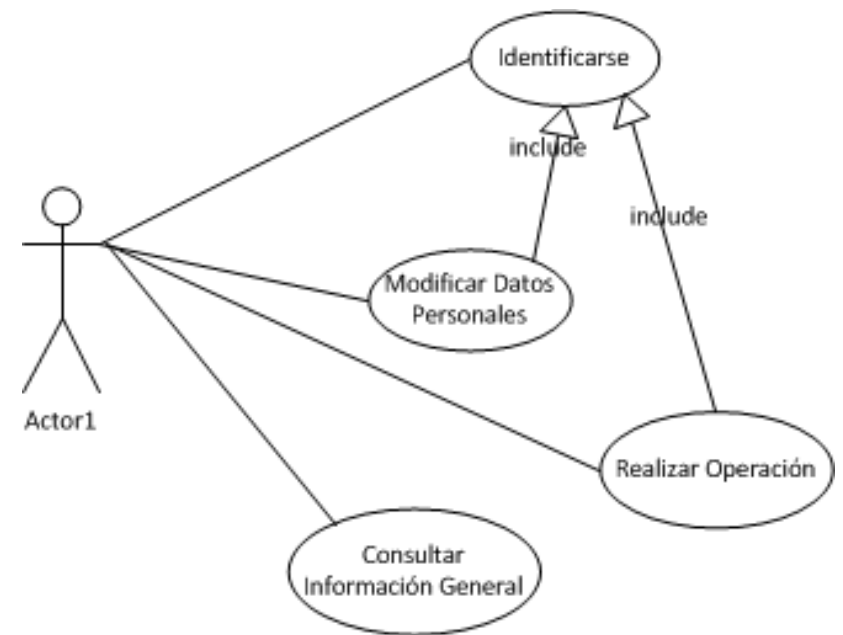


# Posibilidades II, III y IV

- Si “Identificarse” es Caso de Uso, su relación con “Modificar Datos Personales” y con “Realizar Operación” puede ser de 3 tipos:
  - ❑ Como “include”
  - ❑ Como “extend”
  - ❑ Como Precondición

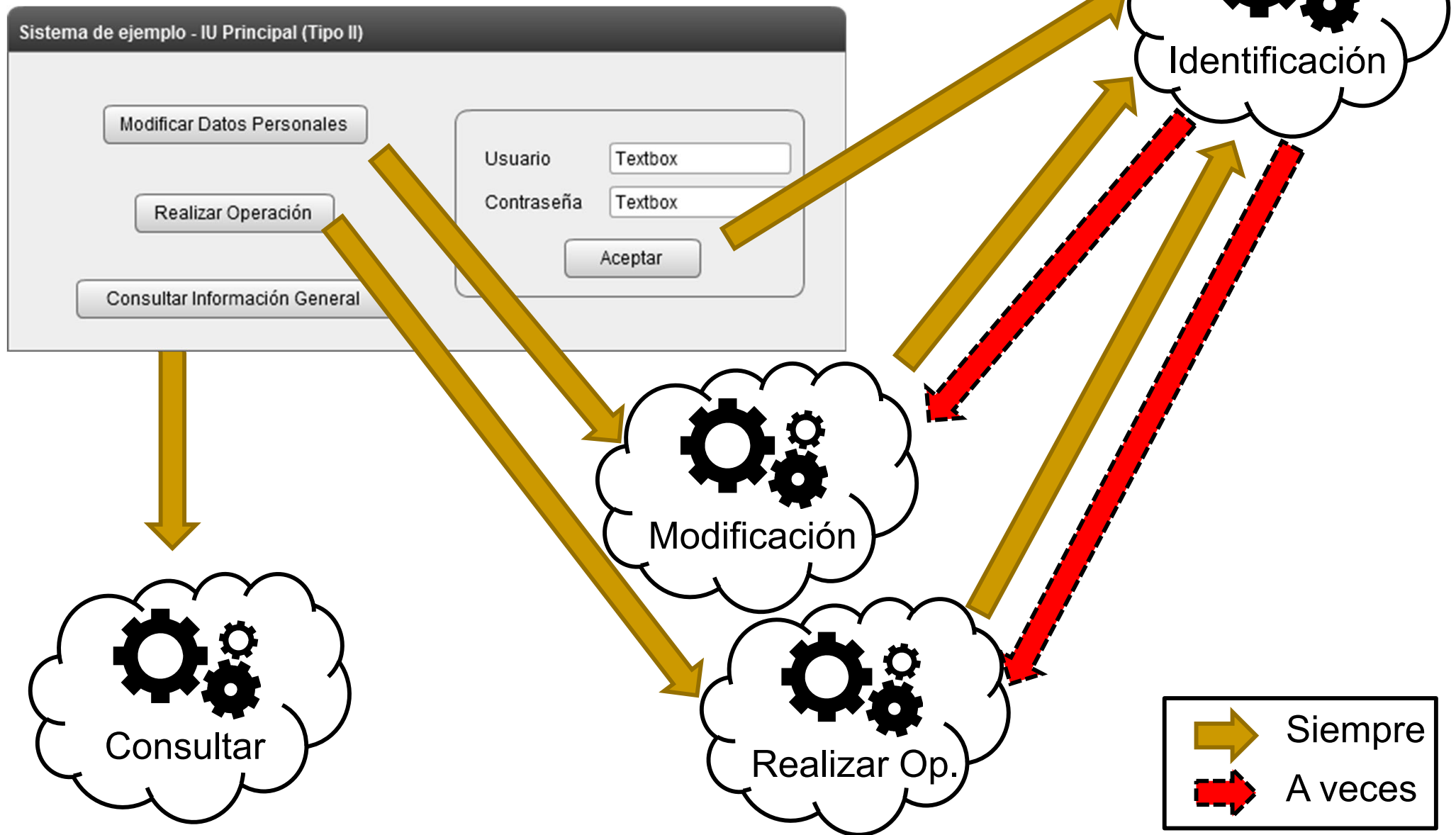
# Posibilidad II

- Como “include” implica:
  - ❑ Cada vez que se ejecuta un Caso de Uso, hay que ejecutar “Identificarse”.
  - ❑ Ventajas:
    - Seguridad Extrema
  - ❑ Desventajas:
    - Poca usabilidad
    - Complejidad en los procesos.



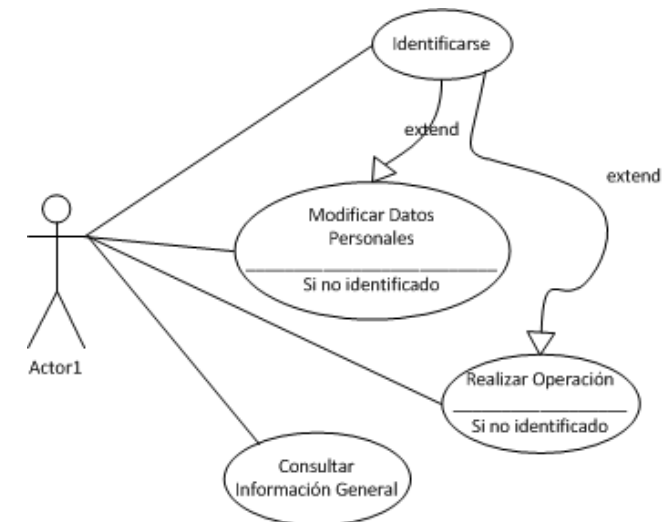


# Posibilidad II

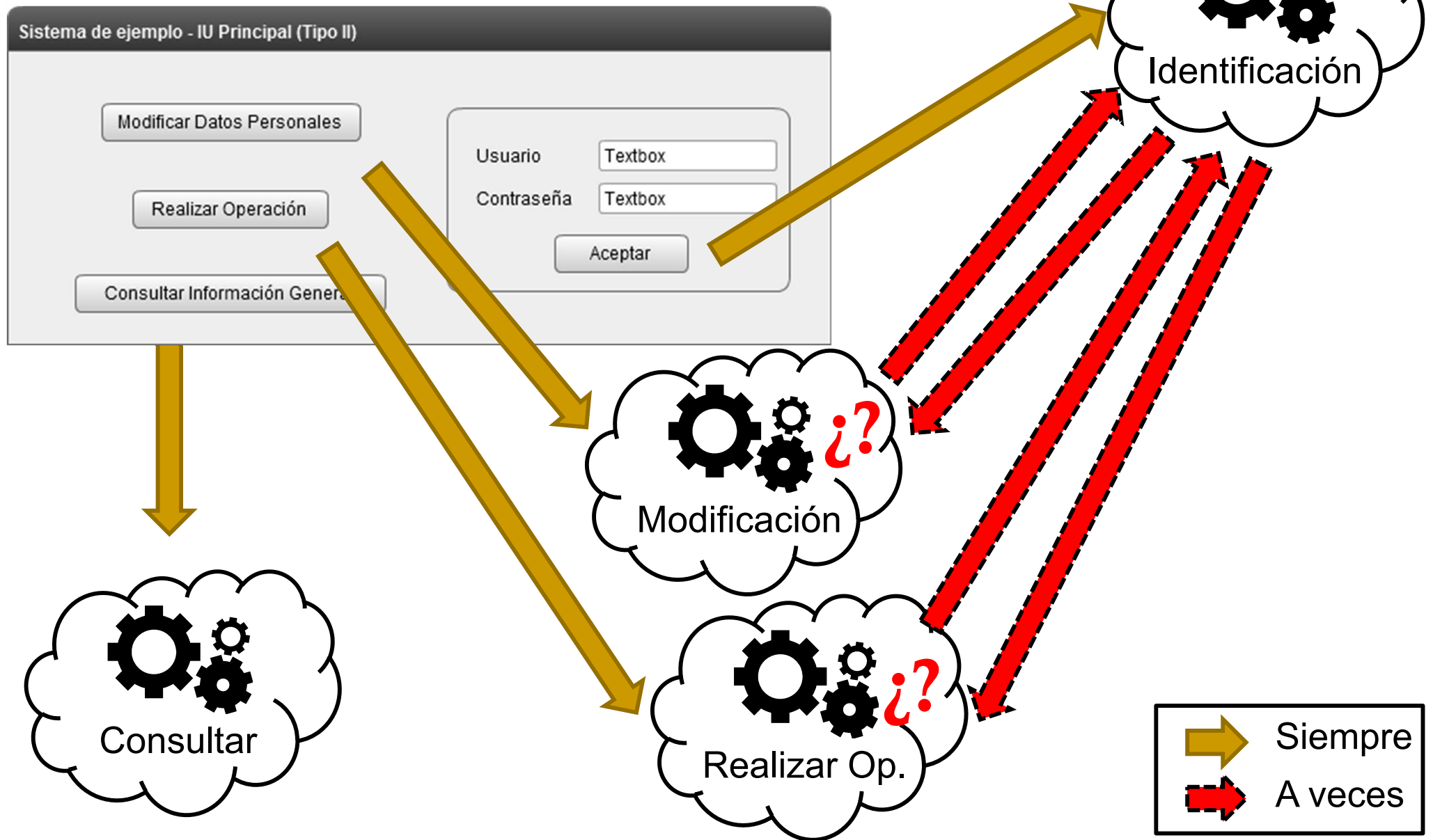


# Posibilidad III

- Como “extend” implica:
  - ❑ Cada vez que se ejecuta el Caso de Uso, hay que comprobar si se está identificado o no. Si no se está, se ejecuta la identificación.
  - ❑ Ventajas:
    - Mayor usabilidad
  - ❑ Desventajas:
    - Mayor complejidad en los procesos
    - Hay que comprobar en todos si se está identificado

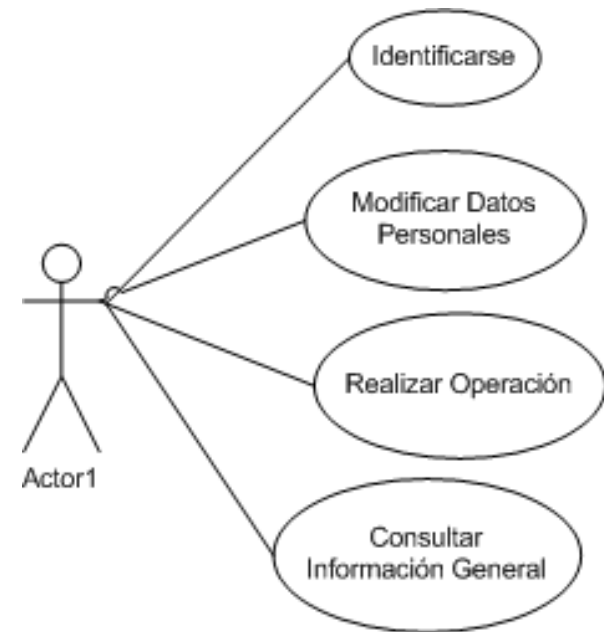


# Posibilidad III



# Posibilidad IV

- Como Precondición implica:
  - Hasta que la identificación no se ha ejecutado, no se pueden ejecutar los Casos de Uso que lo tengan como precondición.
  - Ventajas:
    - Mayor usabilidad
    - IUs más intuitivas y lógicas
    - Procesos más simples



Sistema de ejemplo - IU Principal (Tipo III - Sin Identificar)

Consultar Información General

Usuario

Textbox

Contraseña

Textbox

Aceptar

Identificación

Consultar

Modificación

Realizar Op.

Sistema de ejemplo - IU Principal (Tipo III - Identificado)

Consultar Información General

Modificar Datos Personales

Realizar Operación



Siempre



A veces

---

# Un Caso de Uso especial: “Identificarse”

---

Influencia en la jerarquía de actores

# Sistema de ejemplo

- Tenemos dos tipos de usuarios:
  - Profesores
  - Alumnos
- Ambos tipos de usuarios deben identificarse para acceder al sistema y realizar sus funcionalidades
- Ambos tipos de usuarios pueden “Consultar Información General” sin falta de Identificarse

---

¿Cómo queremos que  
funcione el sistema?

---



# Posibilidad I

- Ambos usuarios se identifican en la misma interfaz y de la misma manera. La información sólo se puede consultar antes de identificarse



# Posibilidad I

- ¿Qué actor ejecuta la identificación y la consulta de información?
- El sistema no puede saberlo
- Hay que definir un actor más general
- NO HAY HERENCIA
  - ❑ En el menú del profesor y del alumno no existe “Identificarse”
  - ❑ En el menú del profesor y del alumno no existe “Consultar Información General”

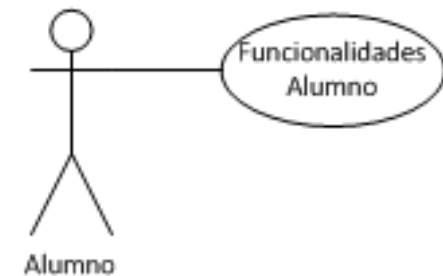
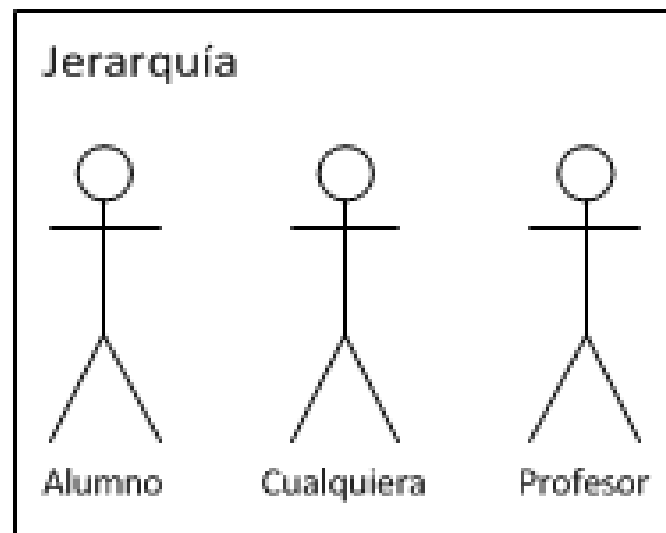
# Posibilidad I

- Cualquiera: persona que “abre” el sistema
- Profesor: persona identificada correctamente como profesor
- Alumno: persona identificada correctamente como alumno



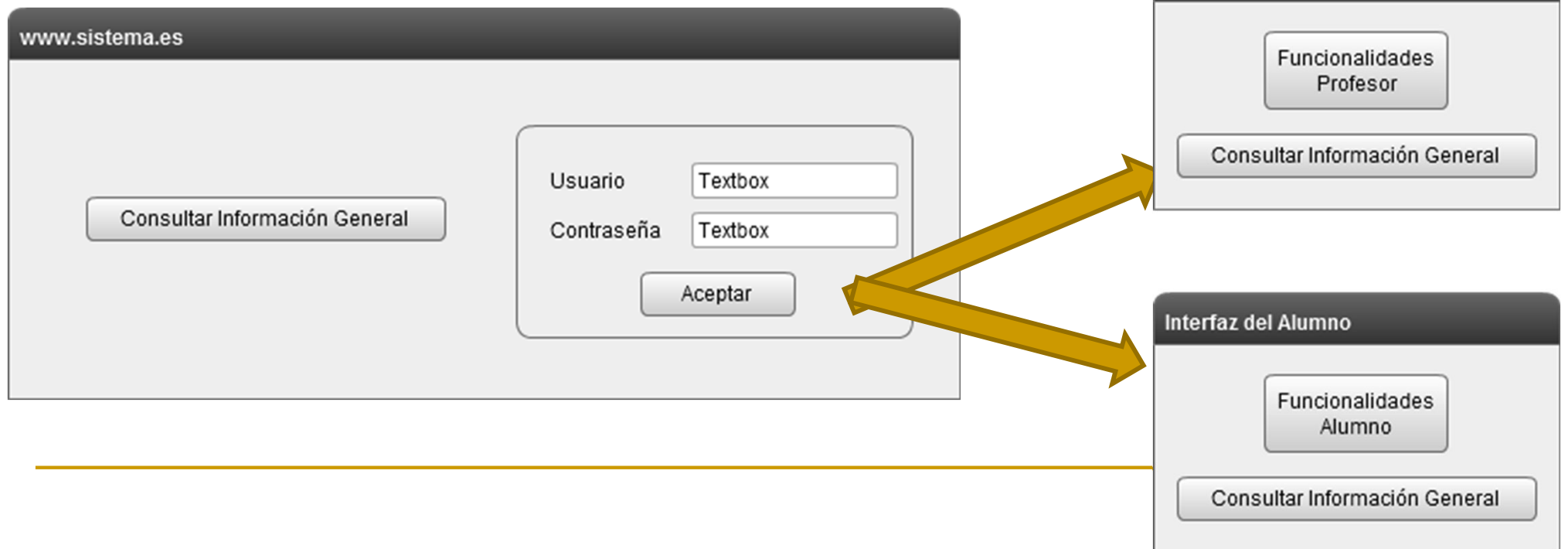
No hace falta la precondition de “estar identificado” en los casos de uso del Profesor y el Alumno

# Posibilidad I



# Posibilidad II

- Ambos usuarios se identifican en la misma interfaz y de la misma manera. La información se puede consultar antes y después de identificarse



## Posibilidad II

- HAY HERENCIA ya que “Consultar Información General” tienen que poder hacerlo ambos actores
- “Identificarse” NO tiene que heredarse
- Hay que definir otro actor más general que “Cualquiera”
  - No tiene una correspondencia real
  - Su uso es por razones de organización

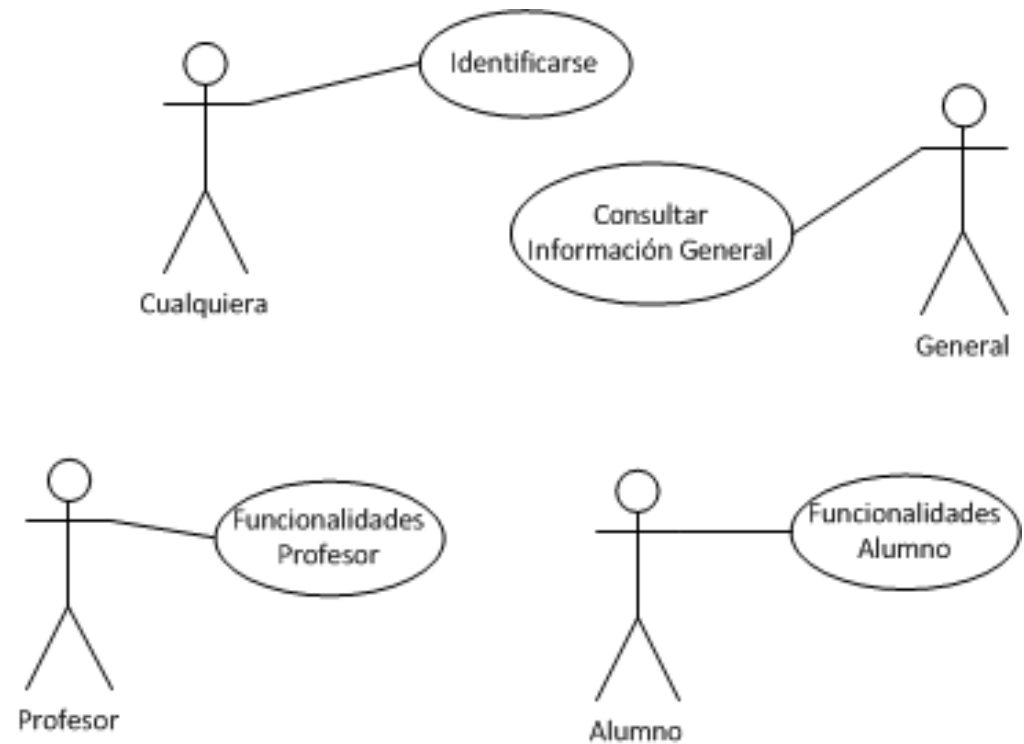
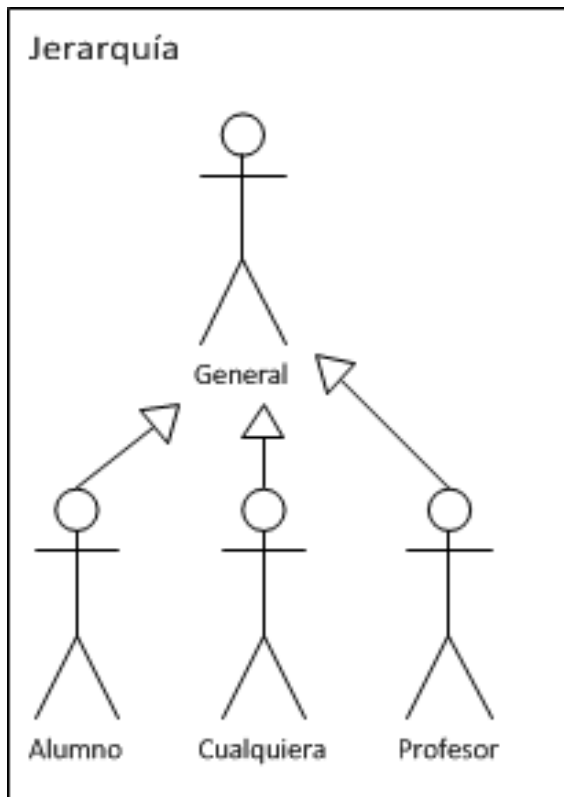
# Posibilidad II

- General: Actor definido por razones organizativas
- Cualquiera: persona que “abre” el sistema
- Profesor: persona identificada correctamente como profesor
- Alumno: persona identificada correctamente como alumno



No hace falta la precondition de “estar identificado” en los casos de uso del Profesor y el Alumno

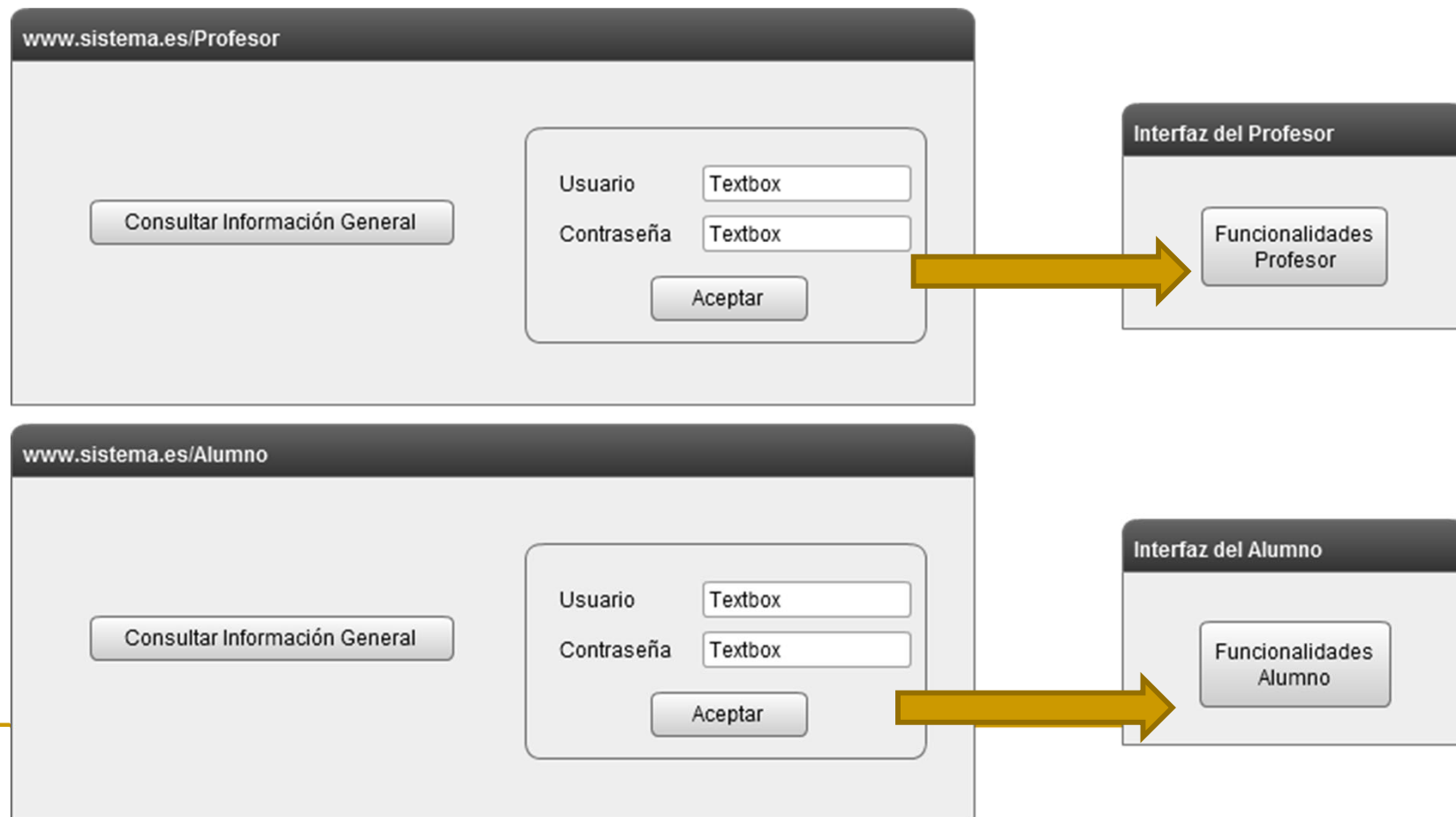
# Posibilidad II





# Posibilidad III

- Ambos usuarios se identifican de la misma manera, pero en interfaces distintas



# Posibilidad III

- ¿El sistema puede distinguir entre Profesor y Alumno antes de identificarse?
  - ❑ Sí, en función de la interfaz a la que acceden
- Se pueden poner los casos de uso directamente al Profesor y al Alumno
- HAY HERENCIA
  - ❑ La identificación es la misma en ambos casos
  - ❑ La consulta de información es la misma en ambos casos

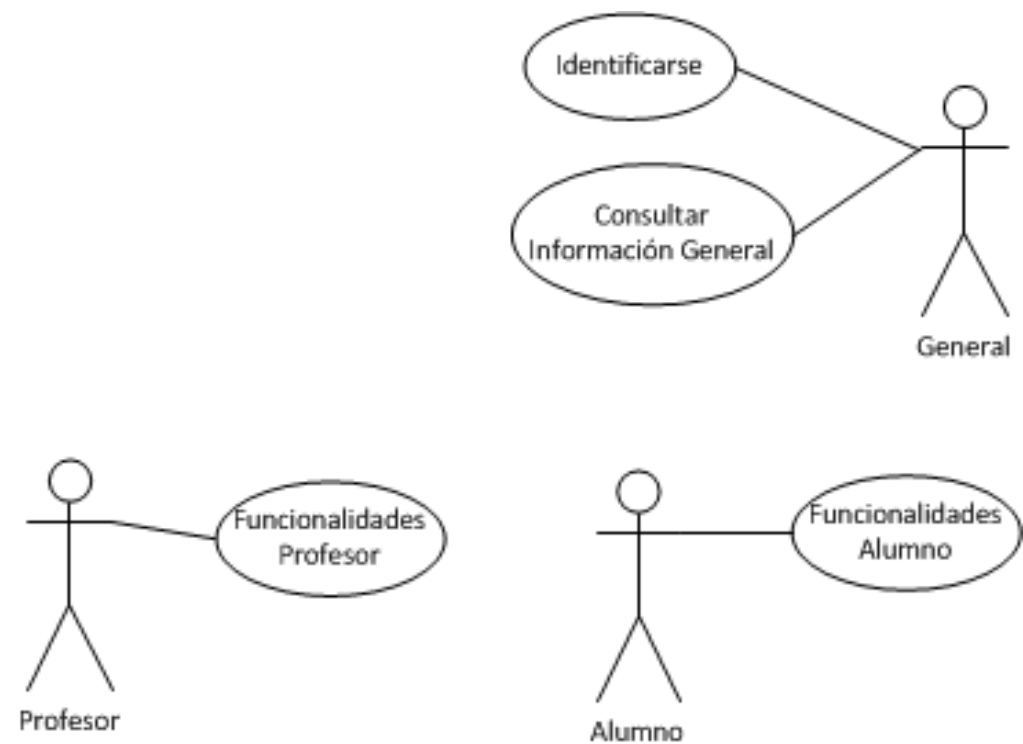
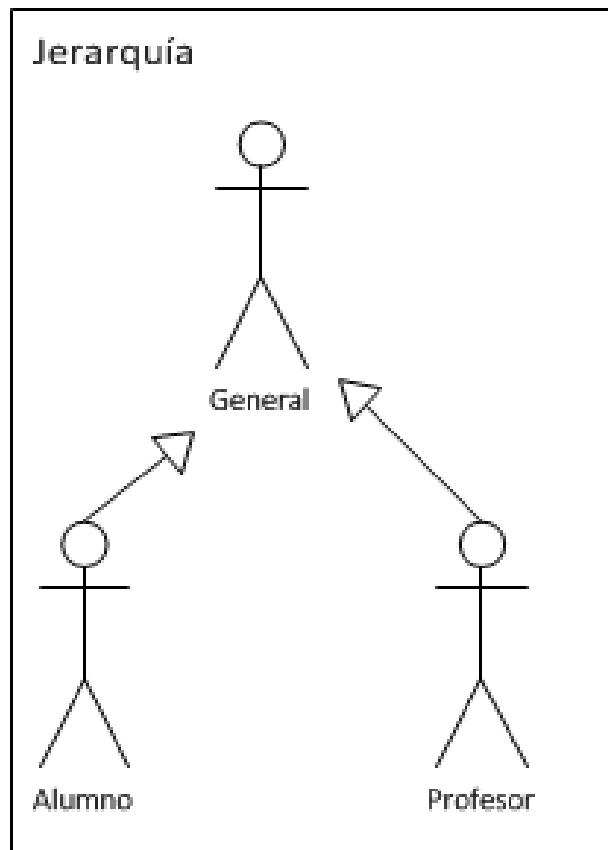
# Posibilidad III

- General: actor definido por razones organizativas
- Profesor: persona que “abre” el sistema en `www.sistema.es/Profesor`
- Alumno: persona que “abre” el sistema en `www.sistema.es/Alumno`



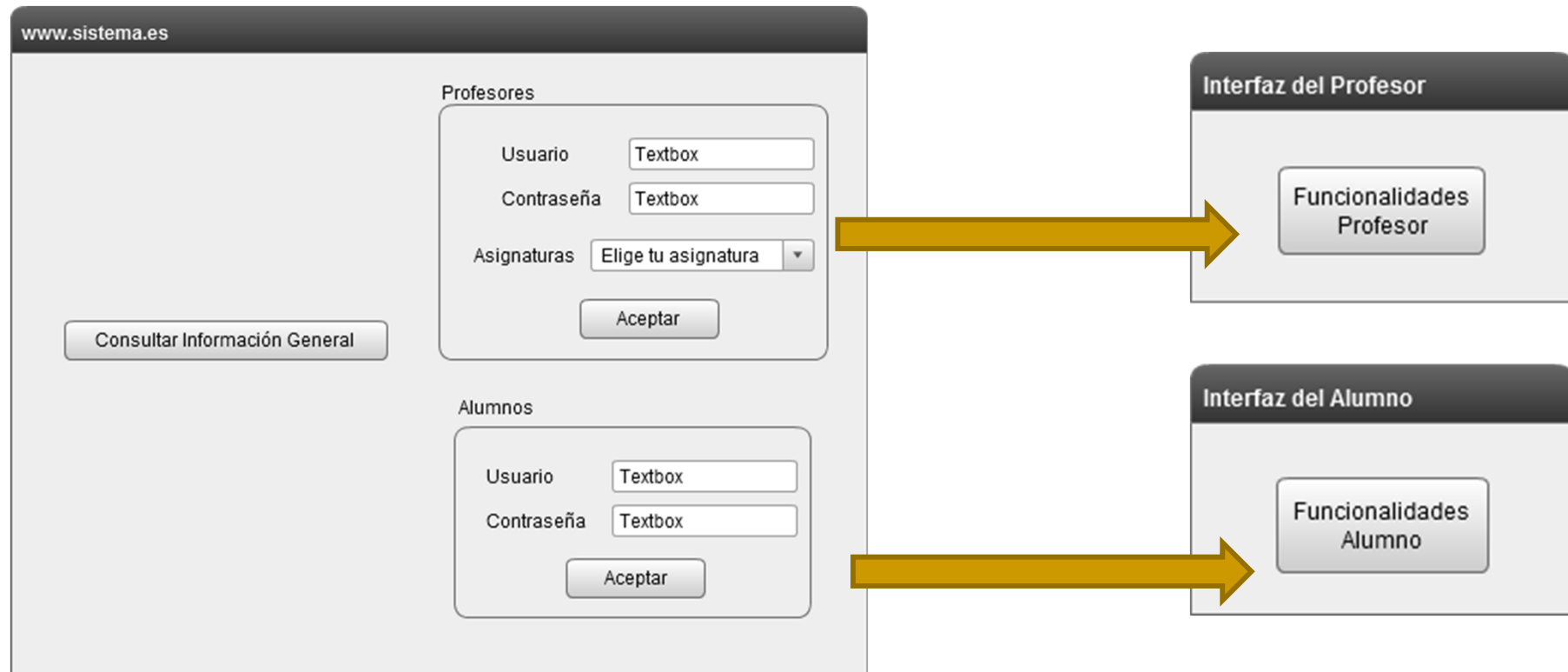
Hace falta poner la precondition “estar identificado” en los casos de uso del Profesor y el Alumno

# Posibilidad III



# Posibilidad IV

- La identificación es distinta



---

## Posibilidad IV

- Son casos de uso **DISTINTOS**
  - En este ejemplo concreto, no se puede saber quién los ejecuta
  - Se necesita el actor “Cualquiera”
-

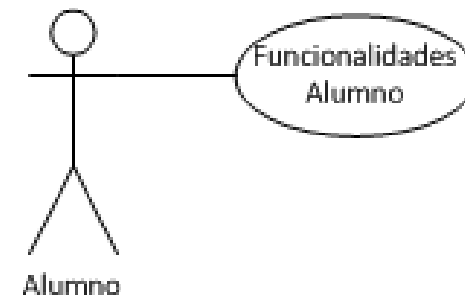
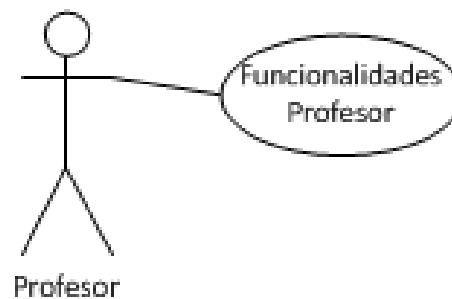
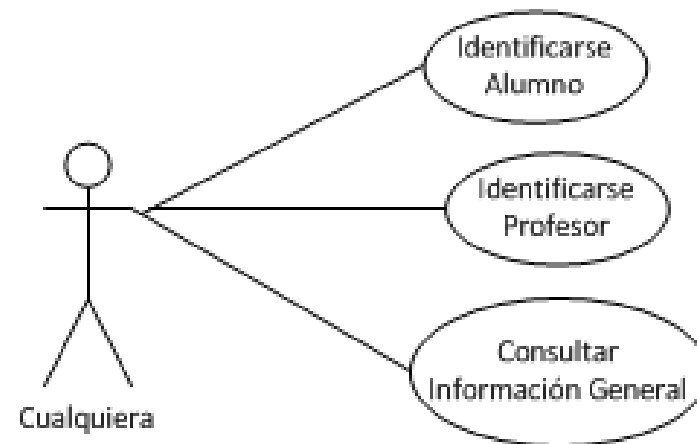
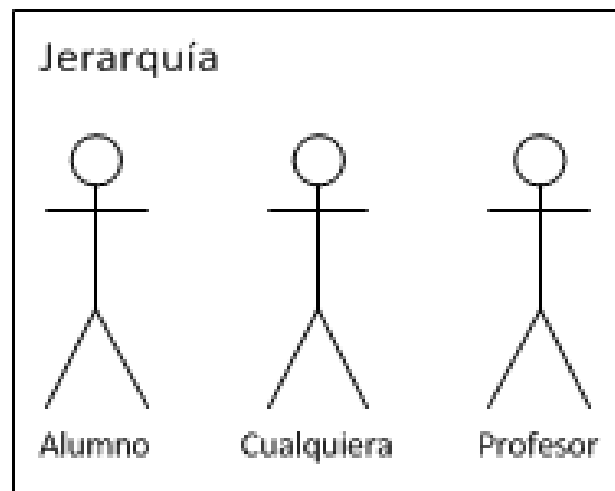
# Posibilidad IV

- Cualquiera: persona que “abre” el sistema
- Profesor: persona identificada correctamente como profesor
- Alumno: persona identificada correctamente como alumno



No hace falta la precondition de “estar identificado” en los casos de uso del Profesor y el Alumno

# Posibilidad IV





# Conclusiones

- El diseño de la interfaz gráfica define
  - Qué casos de uso existen
  - Cómo se relacionan esos casos de uso
  - Qué actores se definen
  - La jerarquía entre dichos actores