

TEMA 2: INGENIERÍA DE REQUISITOS

1. ¿QUÉ ES LA INGENIERÍA DE REQUISITOS?

Los **requerimientos (o requisitos)** para un sistema son descripciones de lo que el sistema debe hacer: el servicio que ofrece y las restricciones en su operación. Tales requerimientos reflejan las necesidades de los clientes por un sistema que atienda cierto propósito, como sería controlar un dispositivo, colocar un pedido o buscar información. Al proceso de descubrir, analizar, documentar y verificar estos servicios y restricciones se le llama **ingeniería de requerimientos (IR)**.

La **Ingeniería de Requisitos (IR)** cubre las tareas y proporciona las técnicas y mecanismos apropiados para:

1. Entender y analizar las necesidades del cliente.
2. Evaluar la viabilidad de las necesidades.
3. Negociar una solución razonable.
4. Especificar la solución sin ambigüedades. Como resultado se tendrá un documento que describa la solución acordada.
5. Validar y analizar la especificación reflejada en el documento de especificación de requisitos. Como resultado se obtendrá el modelo del análisis.
6. Administrar y controlar los requisitos a lo largo del proceso de desarrollo.

El proceso de construcción de una “especificación de Requisitos” es un proceso iterativo, en el que partimos de especificaciones iniciales incompletas, poco claras o ambiguas y llegamos a especificaciones finales completas, claras, documentadas y validadas.

Definición de requisito:

- **Condición o capacidad** que debe tener un producto software para resolver una necesidad expresada por un usuario.
- **Representación** en forma de documento de una capacidad o condición que debe tener un producto software.
- **Característica** de un producto software que es condición para su **aceptación** por parte del cliente.
- **Propiedad o restricción**, determinada con precisión, que un producto software debe satisfacer.

2. TIPOS DE REQUISITOS

REQUISITOS FUNCIONALES: describen la interacción entre el sistema y su entorno, proporcionando servicios que proveerá el sistema o indicando la manera en que éste reaccionará ante determinados estímulos. Enunciados acerca de servicios que el sistema debe proveer, de cómo debería reaccionar el sistema a entradas particulares y de cómo debería comportarse el sistema en situaciones específicas. En algunos casos, los requerimientos funcionales también explican lo que no debe hacer el sistema.

Los requerimientos funcionales del sistema varían desde requerimientos generales que cubren lo que tiene que hacer el sistema, hasta requerimientos muy específicos que reflejan maneras locales de trabajar o los sistemas existentes de una organización. Estos requerimientos funcionales del usuario definen las actividades específicas que debe proporcionar el sistema.

La especificación de los requerimientos funcionales de un sistema debe ser completa y consistente. Totalidad significa que deben definirse todos los servicios requeridos por el usuario. Consistencia quiere decir que los requerimientos tienen que evitar definiciones contradictorias. Aunque es casi imposible lograrlo.

REQUISITOS NO FUNCIONALES: son requerimientos que no se relacionan directamente con los servicios específicos que el sistema entrega a sus usuarios. Pueden relacionarse con propiedades emergentes del sistema, como fiabilidad, tiempo de respuesta y uso de almacenamiento. De forma alternativa, pueden definir restricciones sobre la implementación del sistema, como las capacidades de los dispositivos I/O o las representaciones de datos usados en las interfaces con otros sistemas.

Los requerimientos no funcionales, como el rendimiento, la seguridad o la disponibilidad, especifican o restringen por lo general características del sistema como un todo. Los requerimientos no funcionales a menudo son más significativos que los requerimientos funcionales individuales.

En resumen:

- Restringen los tipos de soluciones que podemos tomar y suelen restringir el diseño que se realice.
- No describen funciones sino propiedades (rendimiento, fiabilidad, seguridad, capacidad de almacenamiento...).
- Son los que garantizan la calidad del software.

- Pueden ser requisitos del producto, requisitos de la organización o requisitos externos.

Dificultades para determinarlos:

- Las metodologías no proveen herramientas ni formas de abordar de forma directa su obtención.
- Suelen aparecer al estudiar los posibles diseños.
- Aumentan la complejidad del diseño.
- Uso del lenguaje natural para su especificación.

REQUISITOS DE INFORMACIÓN: Describen necesidades de almacenamiento de información en el sistema.

3. FURPS

En el UP, los requisitos se clasifican de acuerdo con el modelo FURPS+ [Grady92], un útil nemotécnico que significa los siguientes cinco tipos de requisitos':

- **Funcional (Functional)**: características, capacidades y seguridad. (**CUIDADO ES PARA REQUISITOS FUNCIONALES (ES TROLL))**)
- **Facilidad de uso (Usability)**: factores humanos, ayuda, documentación.
- **Fiabilidad (Reliability)**: frecuencia de fallos, capacidad de recuperación de un fallo y grado de previsión.
- **Rendimiento (Performance)**: tiempos de respuesta, productividad, precisión, disponibilidad, uso de los recursos.
- **Soporte (Supportability)**: adaptabilidad, facilidad de mantenimiento, internacionalización, configurabilidad.

El ' + ' en FURPS+ indica requisitos adicionales, tales como:

- **Implementación**: limitación de recursos, lenguajes y herramientas, hardware...
- **Interfaz**: restricciones impuestas para la interacción con sistemas externos.
- **Operaciones**: gestión del sistema en su puesta en marcha.
- **Empaquetamiento**: Formas de distribución, restricciones de instalación, etc.
- **Legales**: licencias, etcétera.

Resulta útil utilizar las categorías del FURPS + (o algún esquema de clasificación) como una lista para comprobar que se cubren los requisitos, de manera que reducimos el riesgo de no considerar alguna faceta importante del sistema.

4. TAREAS DE LA INGENIERÍA DE REQUISITOS

(0) Estudio de Viabilidad: Técnico, Económico y Jurídico

- ¿Es conveniente realizar el desarrollo del Sistema/Software?
 - ¿Soluciona el Software los problemas existentes?
 - ¿Se puede desarrollar con la tecnología actual?
 - ¿Se puede desarrollar con las restricciones de costo y tiempo?
 - ¿Puede integrarse con otros existentes en la organización?

(1) Obtención de Requisitos (Elicitación).

Trabajo con los clientes y usuarios para:

- Estudiar el funcionamiento del sistema
- Descubrir las necesidades reales
- Consensuar los requisitos entre las distintas partes
- Proceso difícil apoyado por técnicas:
 - Entrevista
 - Escenarios/Puntos de vista
 - Casos de uso
 - Prototipado
 - Análisis Etnográfico

(2) Análisis de Requisitos: Actividad más importante de todas

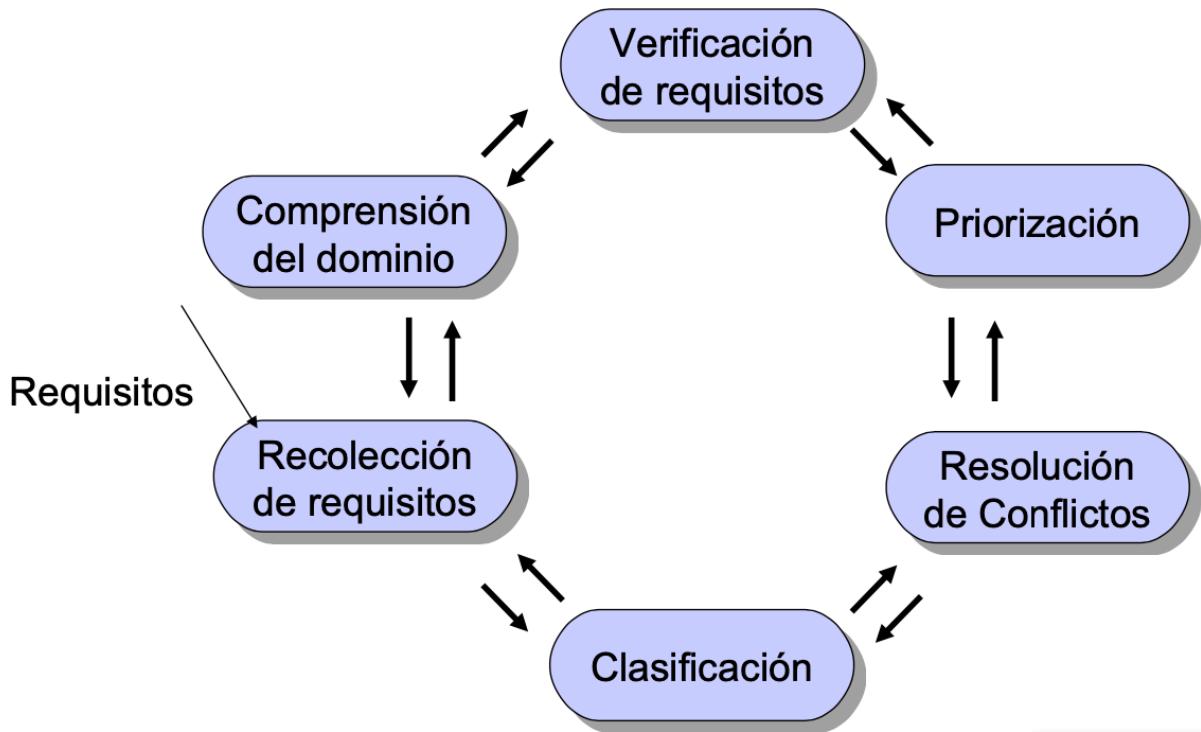
Objetivos:

- Detectar conflictos entre los requisitos
- Profundizar en el conocimiento del sistema
- Establecer las bases para el diseño
- Construcción de modelos abstractos

Sistema -----> Software

"Estudio de Soluciones"

Actividades del Análisis de Requisitos



(3) Especificación de Requisitos

Representación de los requisitos en base al modelo creado en la etapa de análisis (documento escrito, conjunto de diagramas, modelo matemático, simulación, prototipo)

Utilización de herramientas y de estándares

Manual preliminar del usuario

“la idea es correcta pero no es la forma en lo que yo me imaginaba que se iba a poder realizar”

(4) Revisión de requisitos

Validación (1): Ver que los requisitos documentados representan el problema que se desea representar

Verificación (2): La representación es correcta

Proceso continuo durante todo el desarrollo

Facilitar la revisión:

- Crear prototipos
- Crear simulaciones

- Revisión automática (técnicas formales)
- Apoyarse en herramientas

5. PROBLEMAS DE LA INGENIERÍA DE REQUISITOS

Podemos agruparlos en 3 áreas:

- Dificultades para obtener información
- Manejo de la complejidad del problema
- Dificultades para la integración de los cambios

Posibles causas

- Pobre comunicación
- Uso de técnicas inapropiadas
- Tendencias a acortar el análisis
- No considerar alternativas

6. CASOS DE USO

Definición: los casos de uso son un mecanismo ampliamente utilizado para descubrir y registrar los requisitos (especialmente los funcionales); influencian muchos aspectos de un proyecto, incluyendo el A/DOO. Merece la pena tanto saber sobre los casos de uso como crearlos.

La escritura de casos de uso —historias del uso de un sistema— es una técnica excelente para entender y describir los requisitos. Este capítulo explora los conceptos claves de los casos de uso y presenta casos de uso de ejemplo para la aplicación NuevaEra.

El UP define el Modelo de Casos de Uso en la disciplina Requisitos. Básicamente, es el conjunto de todos los casos de uso; es un modelo de la funcionalidad y entorno del sistema.

Elementos que componen el modelo de CU:

- **Actores:** algo con comportamiento, como una persona (identificada por un rol), sistema informatizado u organización; por ejemplo, un cajero.

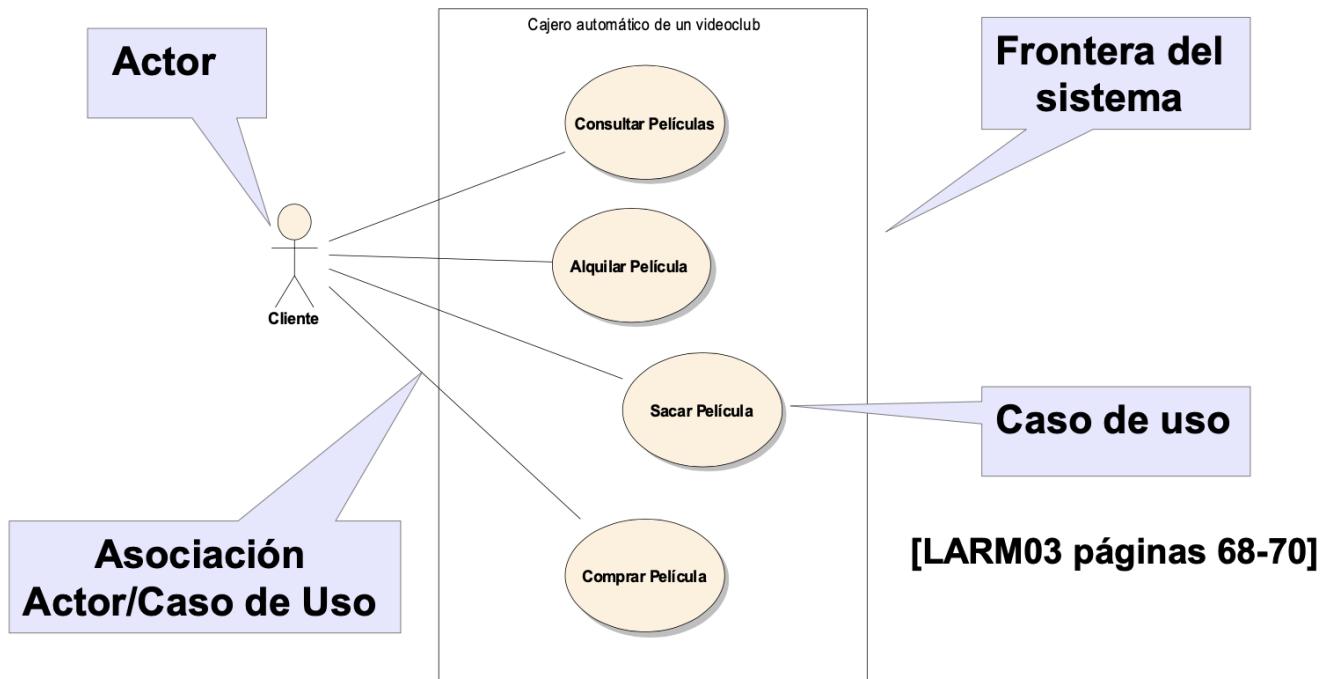
- **Escenario:** secuencia específica de acciones e interacciones entre los actores y el sistema objeto de estudio; también se denomina instancia de caso de uso. Es una historia particular del uso de un sistema, o un camino a través del caso de uso; por ejemplo, el escenario de éxito de compra de artículos con pago en efectivo, o el escenario de fallo al comprar debido al rechazo de la transacción de pago con la tarjeta de crédito.
- **Casos de Uso:** colección de escenarios con éxito y fallo relacionados, que describe a los actores utilizando un sistema para satisfacer un objetivo.
- Relaciones entre:
 - Actores
 - Actores y Casos de Uso
 - Casos de Uso

Para la representación y descripción de estos elementos usamos:

- Diagrama de Casos de Uso de UML.
- Plantillas estructuradas para los Actores y Casos de Uso.

6.1 Diagrama de Casos de Uso

UML proporciona notación para los diagramas de casos de uso con el fin de ilustrar los nombres de los casos de uso y los actores, y las relaciones entre ellos



Los diagramas de caso de uso y las relaciones entre los casos de uso son secundarios en el trabajo con los casos de uso. Los casos de uso son documentos de texto. Trabajar con los casos de uso significa escribir texto. Los expertos mundiales en casos de uso, como Anderson, Fowler, Cockbum, entre otros, minimizan la importancia de los diagramas de casos de uso y las relaciones entre los casos de uso, y en lugar de eso se centran en escribir. Con eso como advertencia, un simple diagrama de casos de uso proporciona un conciso diagrama de contexto visual del sistema, que muestra los actores externos y cómo utilizan el sistema.

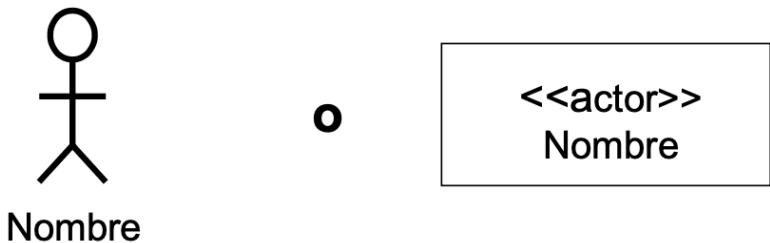
6.2 Actores

Definición: Abstracción de **entidades externas** al sistema que interactúan directamente con él.

Especifican roles que adoptan esa entidades externas cuando interactúan con el sistema. Una entidad puede desempeñar varios roles simultáneamente a lo largo del tiempo. Un rol puede ser desempeñado por varias entidades. El nombre del rol debe ser breve y tener sentido desde la perspectiva del negocio. Es frecuente que coincidan con

áreas de la empresa (vendedor, gestor de almacén) o distintos niveles de la jerarquía (jefe, empleado, aprendiz).

Representación:



Dos tipos:

Principales: Además de interactuar con el caso de uso son los que lo activan.

Secundarios: Interactúan con el caso de uso, pero no lo activan.

Los actores pueden ser:

- Personas con el rol de usuario en el sistema.
- Dispositivos de E/S, como sensores o medidores, siempre que sean independientes de la acción de un usuario.
- Sistemas informáticos externos con los que se tiene que comunicar.
- Temporizador o reloj cuando se hace algo como respuesta a un evento de

tiempo de tipo periódico o en un momento determinado, sin que haya un actor que lo active.

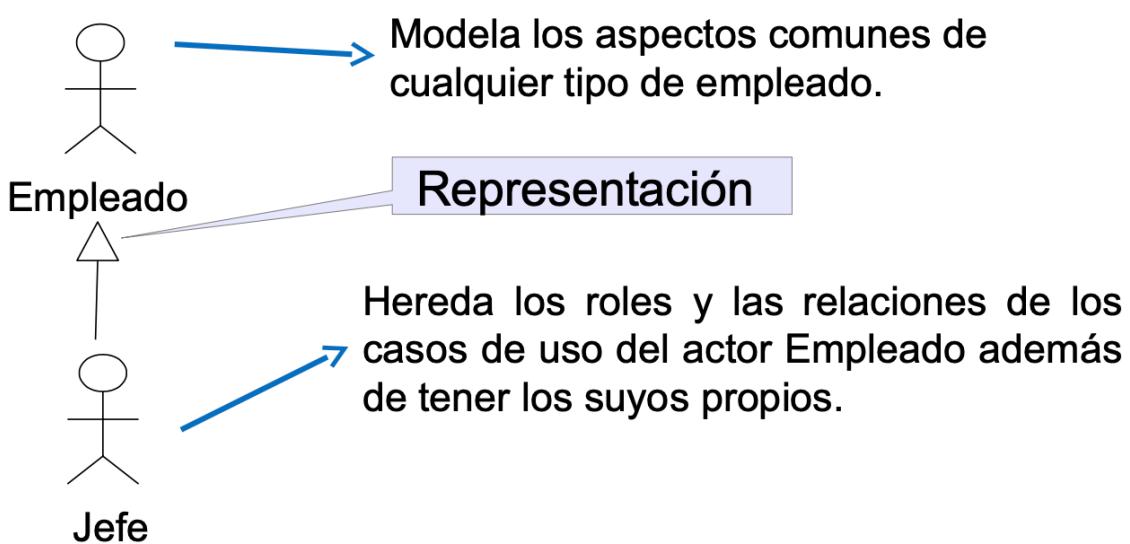
Guía para identificarlo:

Respondiendo a las siguientes preguntas:

- ¿Quién y qué utiliza el sistema?
- ¿Qué roles desempeñan en la iteración?
- ¿Quién instala el sistema?
- ¿Quién o qué inicia y cierra el sistema?
- ¿Quién mantiene el sistema?
- ¿Qué otros sistemas interactúan con este sistema?
- ¿Quién o qué consigue y proporciona información al sistema?

Relación entre actores

Generalización: Expresa un comportamiento común entre actores, es decir se relacionan de la misma forma con los mismos casos de uso.



Un actor Jefe siempre puede ser usado en lugar de un actor Empleado

6.3 Casos de Uso

Definición

Los casos de uso son requisitos; ante todo son requisitos funcionales que indican qué hará el sistema. En términos de los tipos de requisitos FURPS +, los casos de uso se refieren fundamentalmente a la “F” (funcional o de comportamiento), pero también pueden utilizarse para otros tipos, especialmente cuando esos otros tipos están estrechamente relacionados con un caso de uso. En el UP —y métodos más

modernos— los casos de uso son el mecanismo principal que se recomienda para su descubrimiento y definición. Los casos de uso definen una promesa o contrato de la manera en la que se comportará un sistema. Una idea clave de los casos de uso es (por lo general) reducir la importancia o el uso de listas de características detalladas al estilo an- tiguo y más bien, escribir casos de uso para los requisitos funcionales. Veremos más so- bre este punto en una sección posterior.

Los casos de uso. son documentos de texto, no diagramas, y el modelado de casos de uso es, sobre todo, una acción de escribir texto, no dibujar. Sin embargo, UML define un diagrama de casos de uso para ilustrar los nombres de casos de uso y actores, y sus re- laciones.

Guía para identificarlos

Respondiendo a las siguientes preguntas:

- ¿Qué objetivos o necesidades tendrá un actor específico del sistema?
- ¿Qué sucede cuando el sistema cambia de estado (por ejemplo, al iniciar o detener el sistema)? ¿Se notifica a algún actor?
- ¿El sistema almacena y recupera información? Si es así, ¿qué actores activan este comportamiento?
- ¿Afecta algún evento externo al sistema? ¿Qué notificará sobre estos eventos?
- ¿Interactúa el sistema con algún sistema externo?
- ¿Genera el sistema algún informe?

Descripción del caso de uso: Contenido

La descripción de un CU comprende:

- El inicio: Cuándo y qué actor lo produce.
- El fin: Cuándo se produce y qué se obtiene.
- La interacción: Qué mensajes intercambian los actores y el sistema.
- El objetivo: Para qué se usa o qué intenta el CU.
- Cronología y origen de las interacciones.
- Repeticiones de comportamiento: Qué acciones se repiten.
- Situaciones opcionales o de error: Qué situaciones alternativas se presentan en el CU.

Descripción del caso de uso: Tipos

Dependiendo del procesamiento:

- **Básico:** Descripción general del procesamiento. Poco detalle
- **Extendido:** Descripción de la secuencia de acción completa entre actores y sistema. Mucho detalle

Dependiendo de su nivel de abstracción:

- **Esencial:** Expresado de forma abstracta, contiene poca tecnología y pocos detalles de diseño. Muy abstracto
- **Real:** Expresado en base al diseño actual, en el que aparecen relaciones con la Interfaz de Usuario. Muy concreto

Plantilla Básica



Descripción del caso de uso: Plantilla Básica

Caso de uso	<<Nombre del caso de uso>>			<<identificador>>
Actores	<< Actores que participan en el CU, indicando si son principal o secundario>>			
Tipo	<<tipo de CU, primario, secundario u opcional básico o extendido esencial o real>>			
Referencias	<<Requisitos funcionales incluidos en el CU>>		<<CU's con los que se relaciona>>	
Precondición	<< Condiciones sobre el estado del sistema que tienen que cumplirse para que se pueda realizar el CU>>			
Poscondición	<<Efectos que de forma inmediata tiene la realización del CU sobre el estado del sistema>>			
Autor		Fecha	Versión	

Propósito
<<Descripción del objetivo que cubre el CU (suficiente con una línea)>>

Resumen
<<Descripción a alto nivel de la secuencia de acciones realizadas en el CU (suficiente con un párrafo)>>

Plantilla Extendida

Para la descripción extendida del caso de uso recurrimos a **escenarios** que son secuencias específicas y concretas de acciones e interacciones entre los actores y el sistema objeto de estudio (Historia particular).

Dos tipos de escenarios:

- Básicos: situaciones normales y comunes de actuación.
- Secundarios: situaciones alternativas y de error.

Añade a la descripción básica los siguientes apartados:

Curso Normal (básico) de eventos	
Actor	Sistema
1 <<Acción realizada por un actor>>	2 <<Acción realizada por el sistema>>
...	...
m <<Acción realizada por un actor>>	n <<Acción realizada por el sistema>>
...	...
<<se describe la secuencia de acciones realizadas por los actores que intervienen en el CU, se usarán frases cortas>>	<<Se describe la secuencia de acciones realizadas por el sistema como respuestas a las realizadas por los actores>>

Añade a la descripción básica los siguientes apartados (continuación):

Curso Alterno (secundario) de eventos
1.a <<Alternativa a la acción 1 del curso normal de eventos>>
1.<<Acción realizada por actor>> j. <<Acción realizada por el sistema>>
m.a <<alternativa 1 a la acción m del curso normal de eventos>>
1. <<Acción realiza por el actor>> k. <<Acción realizada por el sistema>>
m.b <<alternativa 2 a la acción m del curso normal de eventos>>
1. <<Acción realiza por el actor>> l. <<Acción realizada por el sistema>>

Añade a la descripción básica los siguientes apartados (continuación):

Otros datos			
Frecuencia esperada	<<Número de veces que se realiza el CU por unidad de tiempo>>	Rendimiento	<<Rendimiento esperado de la secuencia de acciones del CU>>
Importancia	<<vital, alta, moderada o baja>>	Urgencia	<<Urgencia en su desarrollo puede ser: alta, media o baja>>
Estado	<<Estado de desarrollo en el que se encuentra>>	Estabilidad	<<Estabilidad de los requisitos asociados al caso de uso>>

Comentarios

<<otros aspecto que necesiten aclararse del CU>>

Las plantillas para la descripción de casos de uso, tanto la básica como la extendida, la tenéis en prado.

6.4 Relaciones Casos de Uso

La finalidad de las relaciones es:

- Organizar los Casos de uso.
- Mejorar la comprensión.
- Reducir la redundancia de texto.
- Mejorar la gestión de los documentos generados.

Tipos

Inclusión

Ésta es la relación más común e importante. Es habitual tener algún comportamiento parcial común a varios casos de uso. Por ejemplo, la descripción del pago a crédito tiene lugar en varios casos de uso, entre los que se encuentran Procesar Venta, Procesar Alquiler, Contribuir a Plan de Ahorro, et- cétera. En lugar de duplicar este texto, es conveniente separarlo en su propio caso de uso de subfunción, e indicar su inclusión.

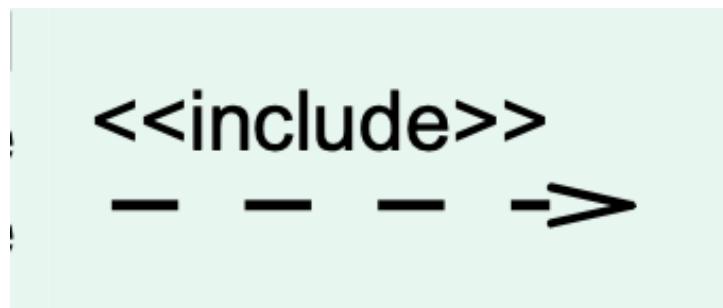
Características:

- Es una relación de dependencia entre dos casos de uso que permite incluir el

comportamiento de un caso de uso en el flujo de otro caso de uso.

- Denominamos al caso de uso que incluye como el caso de uso base y al caso de uso incluido como caso de uso de inclusión.
- El caso de uso base se realiza hasta que se alcanza el punto donde se encuentra la referencia al caso de uso de inclusión, donde la ejecución pasa al caso de uso de inclusión. Cuando éste se termina el control regresa de nuevo al caso de uso base.
- El caso de uso de inclusión es utilizado completamente por el caso de uso base.
- El caso de uso base no está completo sin todos sus casos de uso de inclusión.
- El caso de uso de inclusión puede ser compartido por varios casos de uso base.
- El caso de uso de inclusión no es opcional y es necesario para que tenga sentido el caso de uso base.

Notación:



Extensión

Características:

- Es una relación de dependencia que especifica que el comportamiento del caso de uso base puede ser extendido por otro caso de uso (caso de uso de extensión) bajo determinadas condiciones.
- El caso de uso base declara uno o más puntos de extensión que son como enganches o anclajes en los que se pueden añadir nuevos comportamientos.
- El caso de uso de extensión define segmentos de inserción los cuales pueden ser insertados en los puntos de enganche cuando se cumpla una determinada condición.
- El caso de uso base no sabe nada de los casos de uso de extensión y está completo sin sus extensiones. De hecho los puntos de extensión no tienen numeración en el flujo de eventos del caso de uso base.
- El caso de uso de extensión no tiene sentido de forma separada de un caso de uso base.

<<extend>>
— — — >

Extensión o Inclusión

Heurística

- Usar relaciones de *extensión* para comportamientos excepcionales, opcionales o que rara vez suceden.
- Usar relaciones de inclusión para comportamientos que se comparten entre dos o más casos de uso, o bien para separar un caso de uso en subunidades.

Generalización

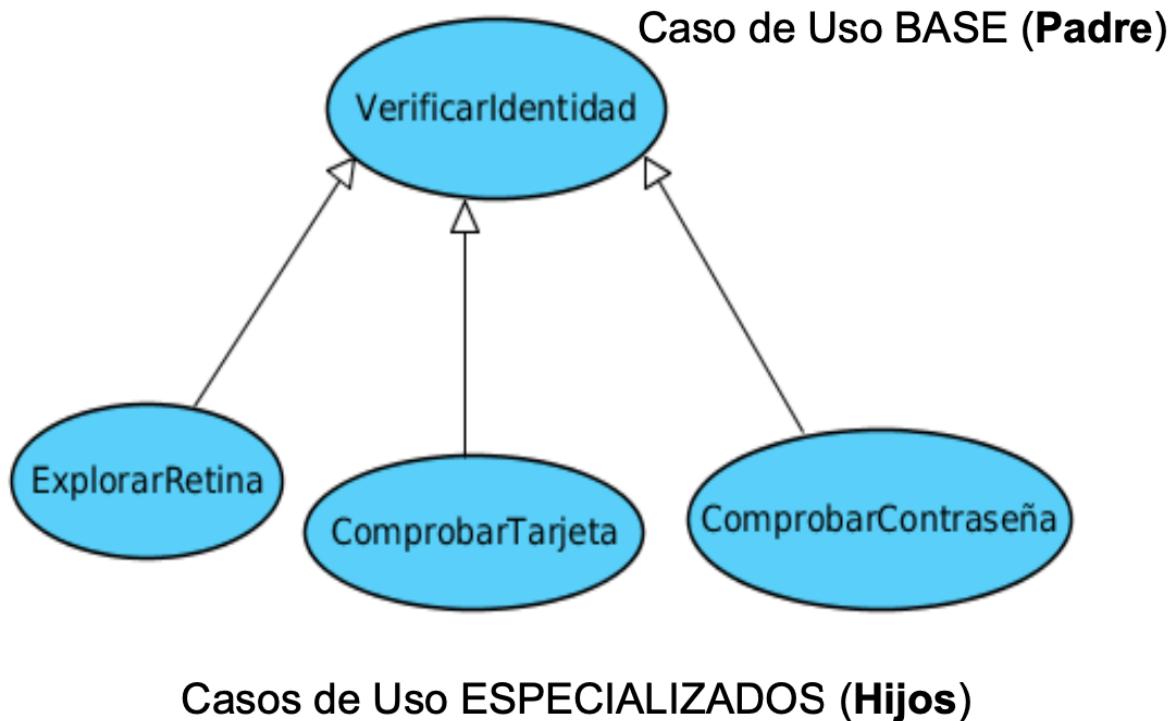
Características:

Es una relación entre un caso de uso general (Caso de uso **padre**) y otros más especializados (**hijos**).

- Los casos de uso hijos:
 - Heredan todas las características del caso de uso general.
 - Pueden añadir nuevas características.
 - Pueden anular o reescribir características del caso de uso general, salvo relaciones, puntos de extensión y precondición.



Notación



Asociación

Relación entre un actor y el caso de uso en el que participa.



Recomendaciones de uso

- Usar las relaciones entre casos de uso cuando simplifiquen el modelo.
- Un sencillo modelo de casos de uso es preferible a uno con demasiadas relaciones, ya que son más fáciles de entender.
- El uso de muchos <> hace que tengamos que ver más de un caso de uso para tener una idea completa.
- Las relaciones extensión son complejas y difíciles de entender por la comunidad de usuarios/clientes.
- La generalización de casos de uso debería evitarse, a menos que se utilicen casos de uso abstractos.

Proceso de construcción del modelo de CU

Pasos a seguir

1. Identificar actores (principales y secundarios).
2. Identificar los principales CU de cada actor, identificando sus objetivos y necesidades, para ello nos preguntamos:
 1. ¿Cuáles son las tareas principales que realiza cada actor?
 2. ¿Qué información del sistema debe adquirir, producir o cambiar?
 3. ¿Tiene que informar el actor sobre cambios producidos en el exterior del sistema?
 4. ¿Qué información desea adquirir el actor del sistema?
 5. ¿Desea el actor ser informado de cambios producidos en el sistema?
3. Identificar nuevos CU a partir de los existentes, para ello analizar las siguientes situaciones:
 1. Variaciones significativas de los CU existentes. Actor que lo realiza / objeto sobre el que actúa
 2. Acciones opuestas → CU opuestos a los existente. Funciones opuestas, Negación de la acción principal
 3. Acciones que deben realizarse antes o después de CU existentes.
4. Hacer el/los Diagrama/s de CU y diagrama de paquetes en el que se muestre las relaciones lógicas entre diagramas de CU
5. Hacer la descripción Básica de cada CU, usando la plantilla básica.
6. Definir prioridades y seleccionar CU primarios, para ello ver:
 - Requisitos imprescindibles.
 - Requisitos importantes.
 - Requisitos deseables.Categorizar los requisitos --> Evaluar costos y complejidad
7. Hacer la descripción Extendida de cada CU, completando la descripción básica con la plantilla extendida.
8. Realizar los diagramas de actividad.
9. Desarrollar prototipos de la Interfaz de Usuario.

Forma de Estructurar Diagramas de CU

Diagrama de paquetes: Diagrama de UML usado para describir la estructuración de un sistema en base a agrupaciones lógicas. También muestra las dependencias entre las agrupaciones. El diagrama de paquetes es usado en el modelado de CU para agrupar de forma lógica los diferentes diagramas de casos de uso.

Diagrama de actividad: Diagrama de UML para la descripción del comportamiento que tienen un conjunto de tareas o procesos.

Se usan para representar:

- Los flujos de actividades de los procesos de negocio de una empresa.
- Los flujos de acciones de uno o varios casos de uso de forma gráfica.

(Se supone que hay seminarios de estos dos tipos de diagramas)

7. Análisis y especificación de requisitos

Un modelo del dominio muestra (a los modeladores) clases conceptuales significativas en un dominio del problema; es el artefacto más importante que se crea durante el análisis orientado a objetos (Los casos de uso son un importante artefacto del análisis de requisitos, pero no son orientados a objetos, Ponen de relieve una vista de procesos del dominio).

Idea Clave: Un modelo del dominio es una representación de las clases conceptuales del mundo real, no de componentes software. No se trata de un conjunto de diagramas que describen clases software, u objetos software con responsabilidades.

Proceso de análisis

Se ha usado tradicionalmente como sinónimo de Ingeniería de Requisitos, pero es una de sus fases, en la que hay que:

- Descubrir los conflictos existentes entre los requisitos.
- Profundizar en el conocimiento del sistema (realización de modelos)
 - Más fáciles de entender por los desarrolladores (lenguaje de los desarrolladores)
 - Servir de base para el diseño e implementación
- Aumentar la formalización del conocimiento existente sobre el sistema, para facilitar el mantenimiento.

Objetivo principal del análisis

Refinar, estructurar y describir los requisitos para conseguir una comprensión más precisa, más fácil de mantener y que nos ayude a estructurar el sistema completo (modelos del análisis).

Es importante rastrear los requisitos de usuario a través de los requisitos del software

Modelos del dominio

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés

Modelo CU	Modelo del Análisis
<ul style="list-style-type: none">• Lenguaje del cliente	<ul style="list-style-type: none">• Lenguaje del desarrollador
<ul style="list-style-type: none">• Vista externa del sistema estructurado en CU	<ul style="list-style-type: none">• Vista interna del sistema estructurado por clases y subsistemas
<ul style="list-style-type: none">• Contrato Cliente/Desarrolladores	<ul style="list-style-type: none">• Con vistas a la solución
<ul style="list-style-type: none">• Puede contener redundancias e inconsistencias entre requisitos	<ul style="list-style-type: none">• No debe contenerlos
<ul style="list-style-type: none">• Captura la funcionalidad del sistema	<ul style="list-style-type: none">• Esboza cómo llevar a cabo esta funcionalidad (primera aproximación a la arquitectura)
<ul style="list-style-type: none">• Se definen CU que luego serán analizados en mayor profundidad	<ul style="list-style-type: none">• Define relaciones entre casos de uso

7.1 Análisis Orientado a Objetos

El AOO examina y representa los requisitos desde la perspectiva de los objetos que nos encontramos en el dominio del problema. Existen una gran variedad de métodos AOO, aunque todos ellos se centran en la obtención de modelos: Estáticos o de estructura, y dinámicos o de comportamiento

El lenguaje o herramienta más usada para representar esos modelos es UML.

¿Por qué usar AOO?

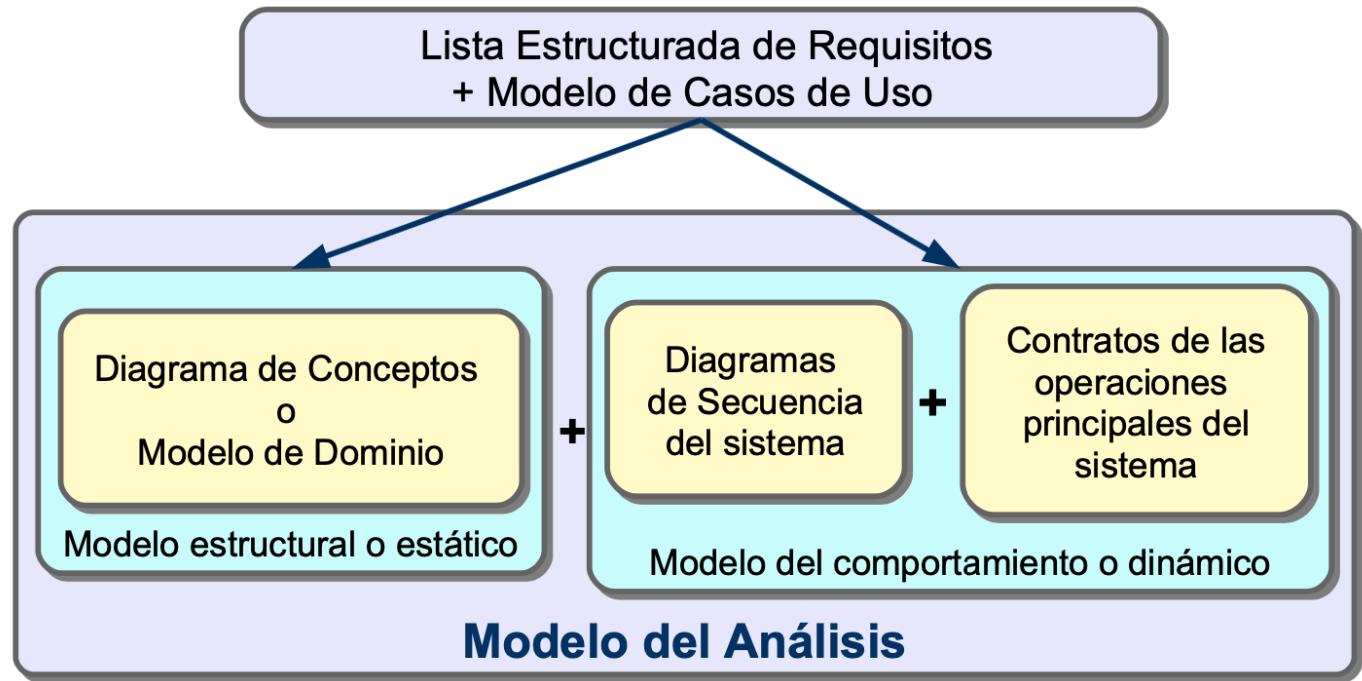
Los términos usados en los modelos están cercanos a los del mundo real facilitando y mejorando la obtención de requisitos y acercando el espacio del problema al espacio de la solución.

Se modelan tanto elementos y propiedades estáticas como dinámicas del ámbito del problema.

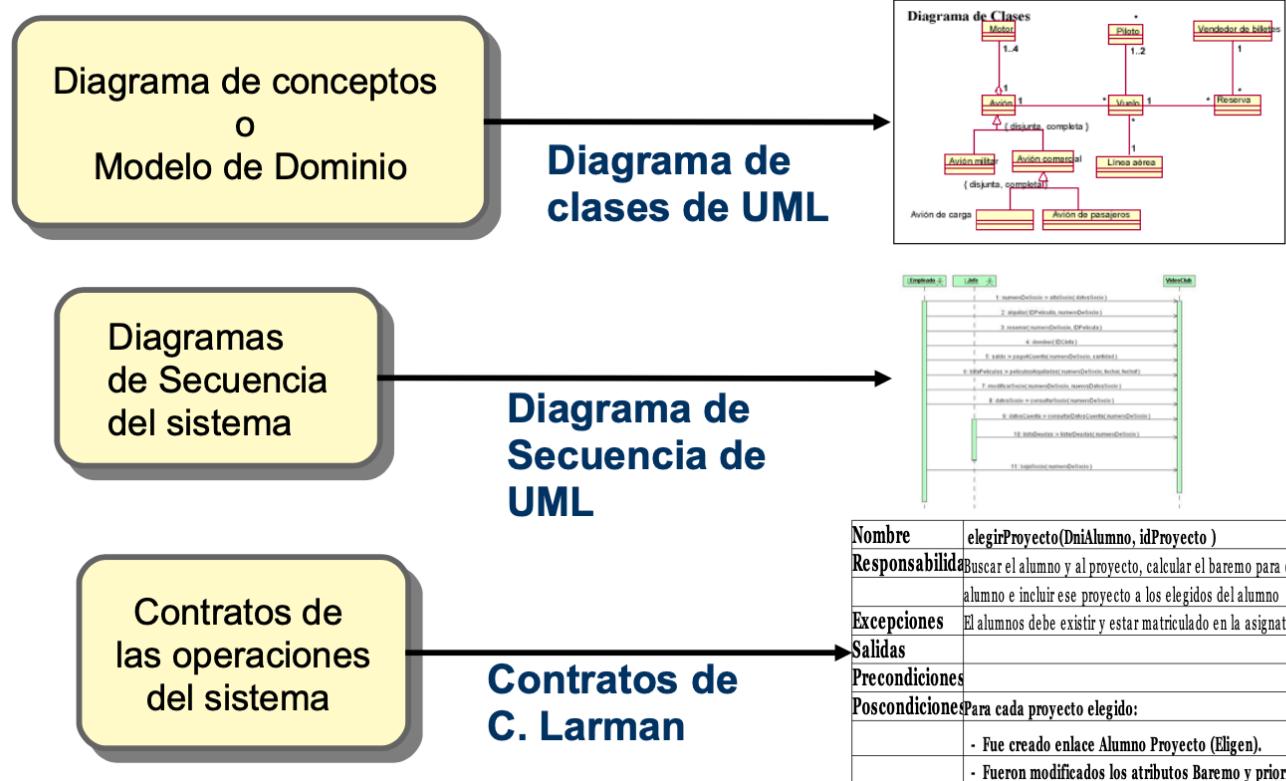
Manejamos conceptos comunes durante el análisis, diseño e implementación del software proporcionando una mejor transición entre fases, facilitando el desarrollo iterativo y difuminando la barrera entre el “Qué” y el “Cómo”.

Modelos del AOO

Identificados y comprendidos los requisitos del sistema software a desarrollar se procede a representarlos mediante modelos.



Usamos las siguientes **herramientas** para representar los Modelos del Análisis.



Modelo estático, Diagrama de conceptos, Modelo de Dominio

En él se representa los principales conceptos del dominio del problema, sus propiedades y relaciones entre ellos.

El modelo de casos de uso es la base para obtener la información necesaria para este modelo.

Se representa usando como herramienta los diagrama de clases de UML, en el que podrá haber:

- Clases que se corresponderán con los conceptos del dominio del problema.
- Asociaciones entre los conceptos.
- Generalizaciones de conceptos.
- Atributos de los conceptos.

1. Elaboración

Pasos a seguir para su obtención:

1. Identificar e incorporar conceptos.

Pasos a seguir:

1. Identificar los conceptos.
2. Seleccionar los conceptos relevantes en el dominio problema.
3. Representarlos, como clases, en el diagrama de conceptos.

Estrategias para identificar conceptos:

- Establecer una lista de categorías de conceptos y rellenarla a partir de la información de la que disponemos.
- Encontrar los términos que se correspondan con sustantivos o frases nominales, éstos van a ser candidatos a conceptos.

Lista de categoría de conceptos

Tipo de categorías		Ejemplos
Cosas no tangibles	<ul style="list-style-type: none"> • Línea de crédito • Beca • Clasificación • Acción en bolsa • Expediente • Matrícula 	
Documentos físicos o virtuales	<ul style="list-style-type: none"> • Catálogo de artículos • Lista de alumnos • Cuenta corriente • Recibo • Contrato laboral 	
Especificaciones, reglas, diseño o Descripciones	<ul style="list-style-type: none"> • Especificación de un producto • Regla de negocio(Devoluciones/Cancelación) • Reglas de creación de producto/servicios. • Manual de procedimientos y seguridad 	
Transacciones	<ul style="list-style-type: none"> • Venta • Matrícula • Reserva • Prestamo 	

Tipo de categorías	Actores y agentes participantes	<ul style="list-style-type: none"> • Cajero • Cliente • Usuario • Supervisor • Proveedor • Transportista 	Ejemplos
	Lugares	<ul style="list-style-type: none"> • Establecimiento • Oficina de atención al público • Despacho del profesor • Almacén de artículos • Centro académico 	
	Organizaciones	<ul style="list-style-type: none"> • Compañía aérea • Universidad • Entidad Bancaria • Departamento 	
	Cosas tangibles	<ul style="list-style-type: none"> • Cajón de máquina registradora • Cajero automático • Producto • Terminal Punto de Venta 	

Tipo de categorías	Items de una transacción	<ul style="list-style-type: none"> • línea de una venta • Importe de la matrícula • Fechas de la reserva • Periodo de vencimiento del préstamo 	Ejemplos
	Eventos	<ul style="list-style-type: none"> • Venta • Compra • Matrícula • Certificación académica • Autorización de pago • Cancelación de reserva • Ingreso hospitalario 	
	Contenedores de cosas	<ul style="list-style-type: none"> • Recipiente • Autocar • Unidad de Urgencia. • Plan de estudios 	
	Items del contenedor	<ul style="list-style-type: none"> • Elementos del recipiente • Pasajero • Box de Urgencias • Asignaturas 	

Tipo de categorías	<ul style="list-style-type: none"> • Tipo de impuesto aplicable • Tipo de conservación del producto • Tipo de préstamo • Tipo de subasta • Tipo de procedimiento terapéutico • Tipo de contrato de trabajo 	Ejemplos
Tipo o categoría de cosas		
	<ul style="list-style-type: none"> • Sistema de pago a crédito • Sistema de Expedientes • Sistema de autorización de pago con tarjetas • Sistema de control de temperatura • Sistema de envío de pedidos. 	
Otros sistemas externos al SuD		

Sustantivos ---> Conceptos

Problemas: No se puede hacer de forma mecánica y puede haber ambigüedad del lenguaje natural.

Ejemplo: Este caso de uso comienza cuando un cliente llega a una caja de TPDV con productos que desea comprar.

El cajero registra el código universal de producto (CUP) en cada producto. Si el producto se repite, el cajero también puede introducir la cantidad.

2. Identificar e incorporar asociaciones entre conceptos.

Una asociación es una conexión significativa y relevante entre conceptos.

Pasos a seguir:

a. *Identificar las posibles asociaciones*

Siguiendo una lista de categoría de relaciones entre Conceptos

Categoría	Ejemplos
A es una parte física de B	Ala-Avión
A es una parte lógica de B	TramoDeVuelo-RutaDeVuelo
A está contenido físicamente en B	Asiento-Avión
A está contenido lógicamente en B	Vuelo-ProgramaDeVuelo
A es una descripción de B	DescripciónDeVuelo-Vuelo
A es un elemento de línea en una transacción B	TrabajoDeMantenimiento-Mantenimiento
A se conoce/ introduce/ registra/ presenta/ captura en B	Reserva-ListaDePasajeros
A es miembro de B	Piloto-Tripulación
A es una subunidad organizacional de B	UnidadMantenimiento-CompañíaAérea
A usa o dirige a B	Piloto-Avión
A se comunica con B	AgenteDeReserva-Pasajero
A se relaciona con una transacción B	Pasajero-Billete
A es una transacción relacionada con otra transacción B	Reserva-Cancelación
A está contiguo a B	Ciudad-Ciudad
A es propiedad de B	Avión-CompañíaAérea

Identificando conceptos relacionados (ejemplo)

1. En un plan de estudios de una titulación universitaria, hay una asignatura denominada "proyectos".

Plan Estudios ----- Asignaturas

A es parte lógica de B

2. Para aprobar dicha asignatura el alumno tiene que desarrollar un trabajo práctico, en el que resuelva un determinado problema aplicando los conocimientos adquiridos durante su formación.

Alumno ----- Proyecto

A se conecta con B

3. El alumno recibe la dirección tutelada de un profesor.

Alumno ----- Profesor

A dirige B

4. Los profesores definen una serie de proyectos.

Profesor ----- Proyecto

A es propiedad de B

5. los alumnos indican sus preferencia (proyectos)

Alumno ----- Proyecto

A conoce a B

6. se les(Alumno) adjudica un proyecto determinado, de entre sus elegidos,

Alumno ----- Proyecto

A es parte lógica de B

7. Los alumnos se matriculan de dicha asignatura “proyecto”.

Alumno ----- Asignatura

A conoce a B

8. Nota media del Expediente del Alumno

Expediente ----- Alumno

A es propiedad de B

9. Asignaturas recomendadas en el proyecto

Proyecto ----- Asignatura

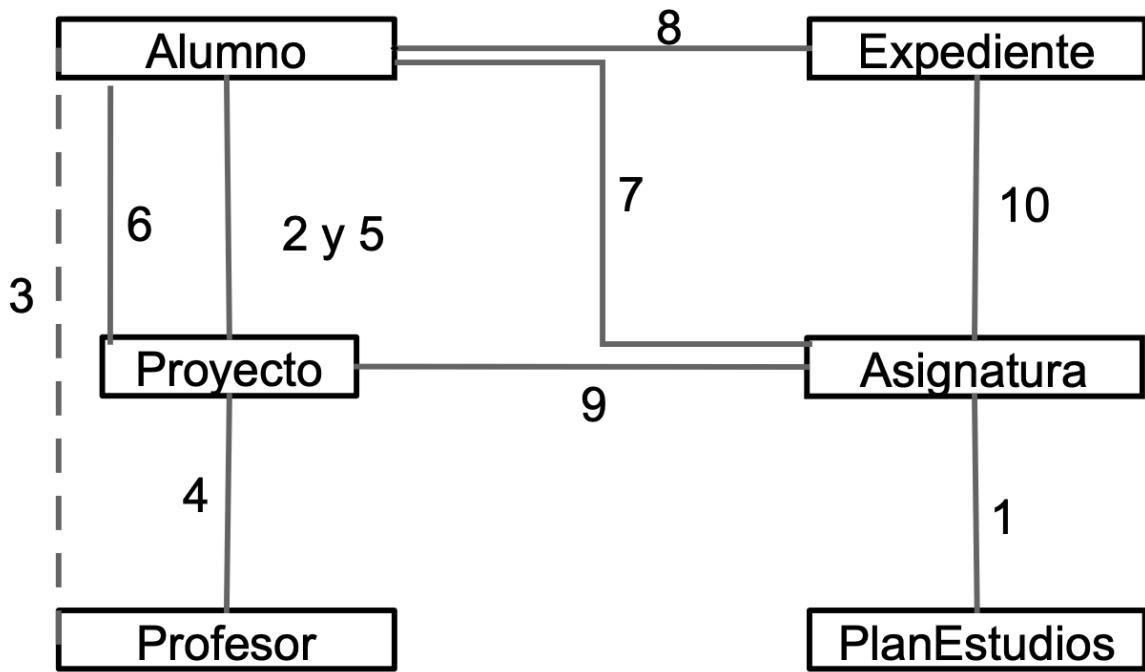
A conoce a B

10. Del enunciado del problema (punto 4º del proceso a seguir) se deduce que: El Expediente está formado por Asignaturas y sus notas.

Expediente ----- Asignatura

A es parte lógica de B

b. Representarlas en el diagrama y seleccionar las asociaciones válidas.

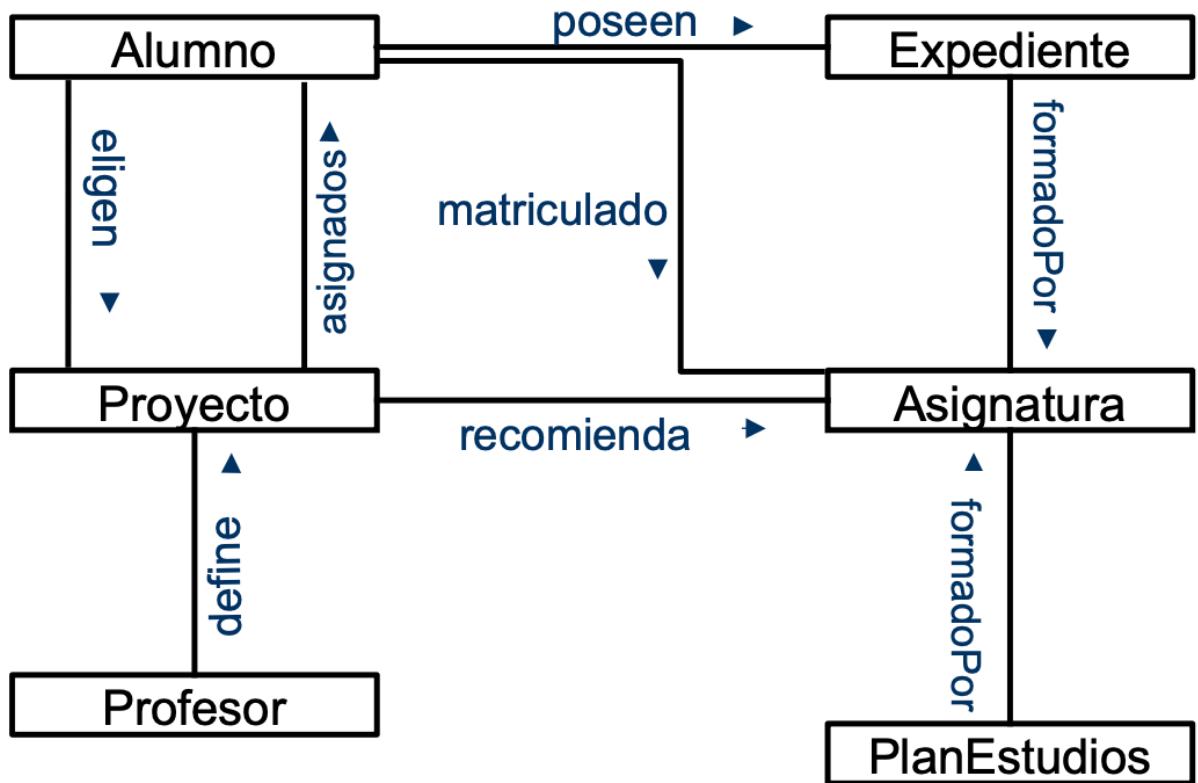


— — — — **Asociación redundante o derivada**

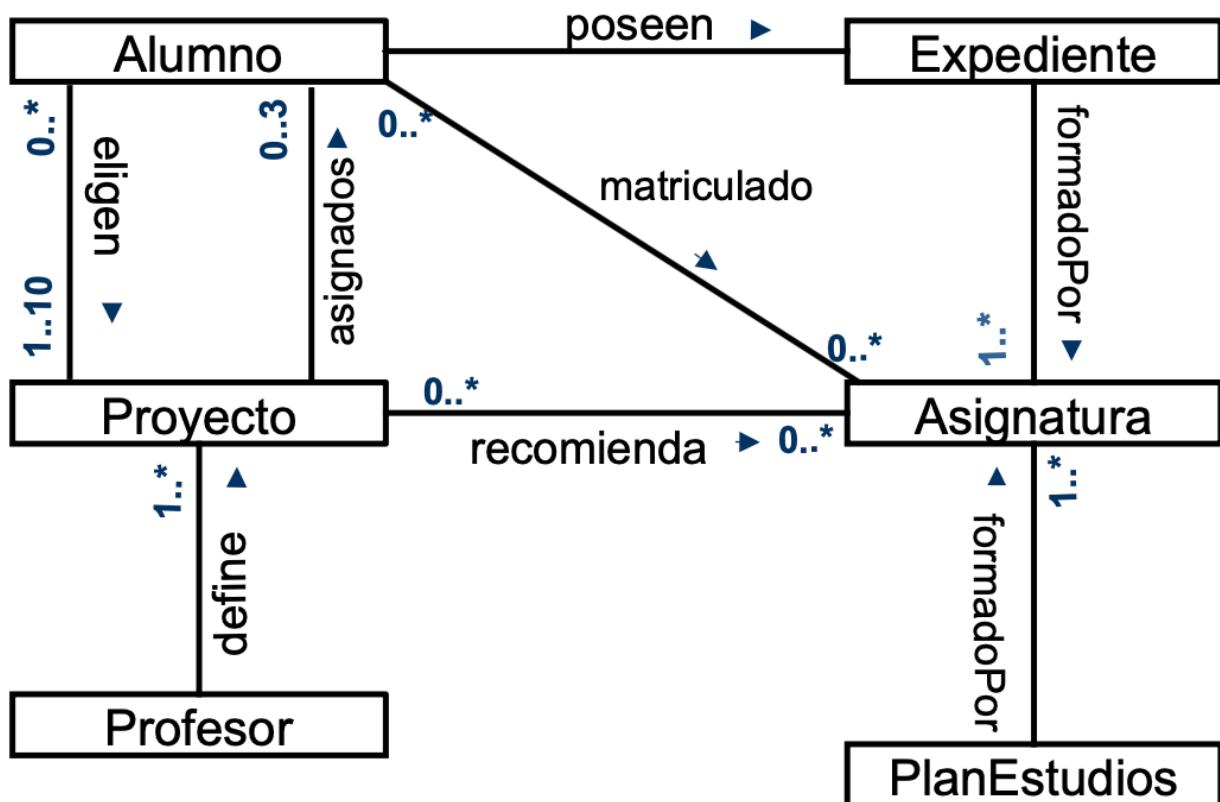
c. Asignarles nombre.

Asociaciones:

- 1 -> Un Plan de Estudios está formado por Asignaturas.
- 6 -> Los alumnos eligen proyectos.
- 2 y 5 -> Los Proyectos son asignados/realizados a/por Alumnos.
- 4 -> Los profesores definen Proyectos.
- 7 -> Alumnos matriculados de asignaturas
- 8 -> Los alumnos poseen expedientes.
- 10 -> Los expedientes están formados por Asignaturas y su nota.
- 9 -> Los proyecto recomiendan asignaturas.



d. Incorporar multiplicidades.



3. Identificar e incorporar generalizaciones de conceptos.

Pasos a seguir

1. Identificar posibles generalizaciones

- A partir de la descripción del problema y de las clases conceptuales identificadas, encontrar clases conceptuales con elementos comunes
 - Definir las relaciones de superclase (concepto general) y subclase (concepto mas específico)
2. **Validar las estructuras encontradas.** Una subclase potencial debería estar de acuerdo con:
- La regla del “100%” (conformidad con la definición de la superclase)
 - La regla “es-un” (conformidad con la pertenencia al conjunto que define la superclase)
3. **Representarlas en el modelo conceptual**
-

Directrices para obtenerlas:

Para **crear subclases** a partir de superclases:

- La subclase tiene atributos adicionales de interés
- La subclase tiene asociaciones adicionales de interés
- La subclase funciona, reacciona o se manipula de manera diferente a la superclase o a alguna subclase

Para **crear superclases** a partir de subclases potenciales:

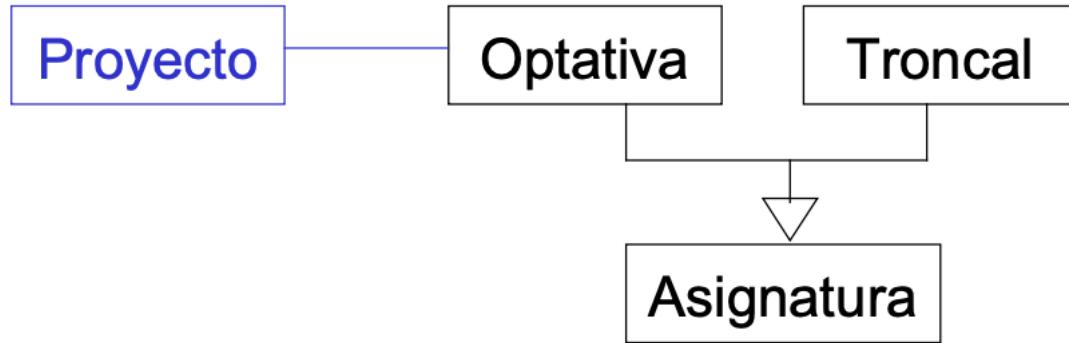
- Cuando las subclases presentan variaciones de un concepto similar
- Las subclases cumplen con las reglas del “100%” y “es-un”
- Todas las subclases tienen el mismo atributo que se puede factorizar en la superclase
- Todas las subclases tienen la misma asociación que se puede factorizar en la superclase

En el ejemplo:

Supongamos que se dice:

“Los profesores definen los contenidos de sus proyectos, ... las asignaturas optativas recomendadas...”

- Hay dos tipos de asignaturas: Optativas y ¿Troncales?
- Esta justificada la subclasificación: Sí, ya que asignatura optativa tiene una asociación relevante con proyecto.
- Incorporación al modelo conceptual.



4. Identificar e incorporar atributos de conceptos.

Pasos a seguir:

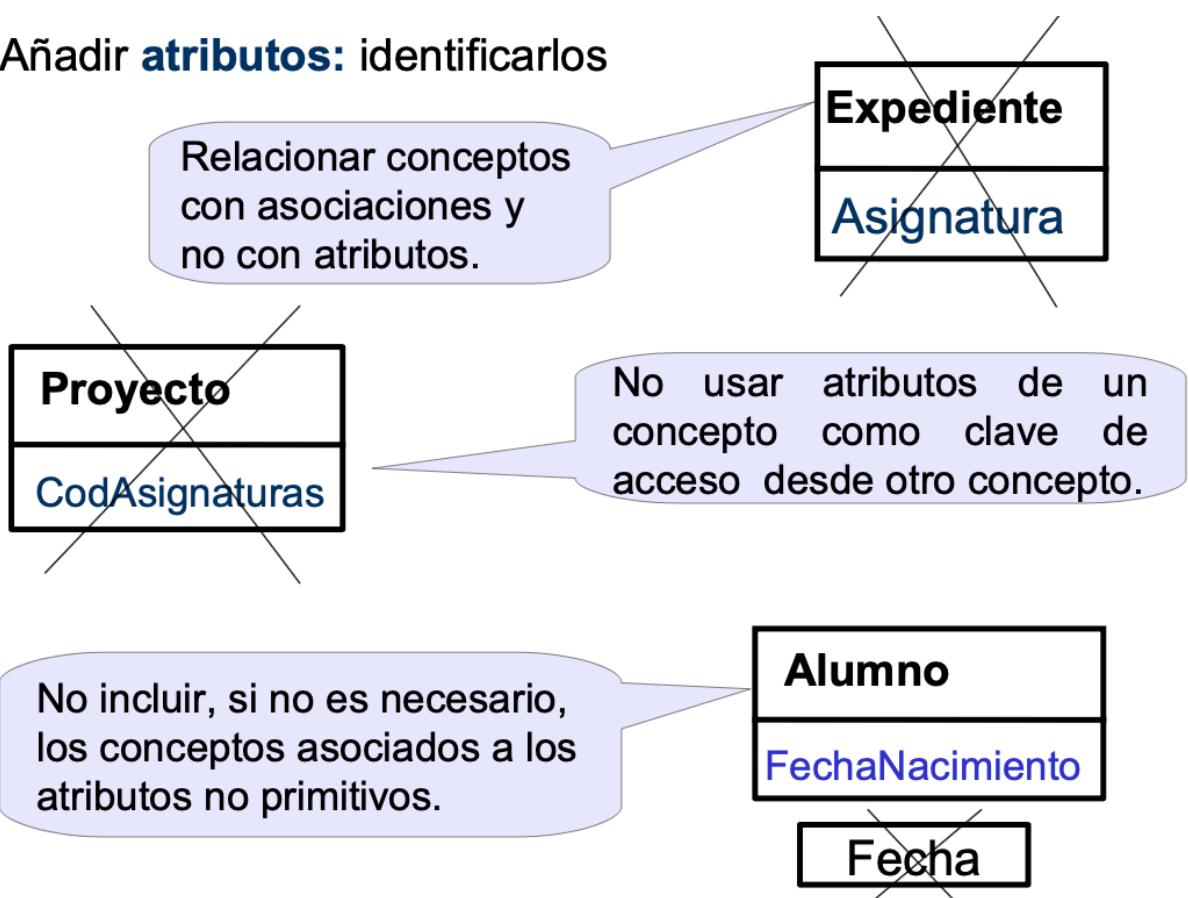
1. Identificar Atributos desde
 - Casos de uso y lista de requisitos
 - Otras fuentes de información (Documentos, impresos, ...)
2. Representarlos en el diagrama, en los conceptos o las relaciones que correspondan.

Tipos de atributos válidos:

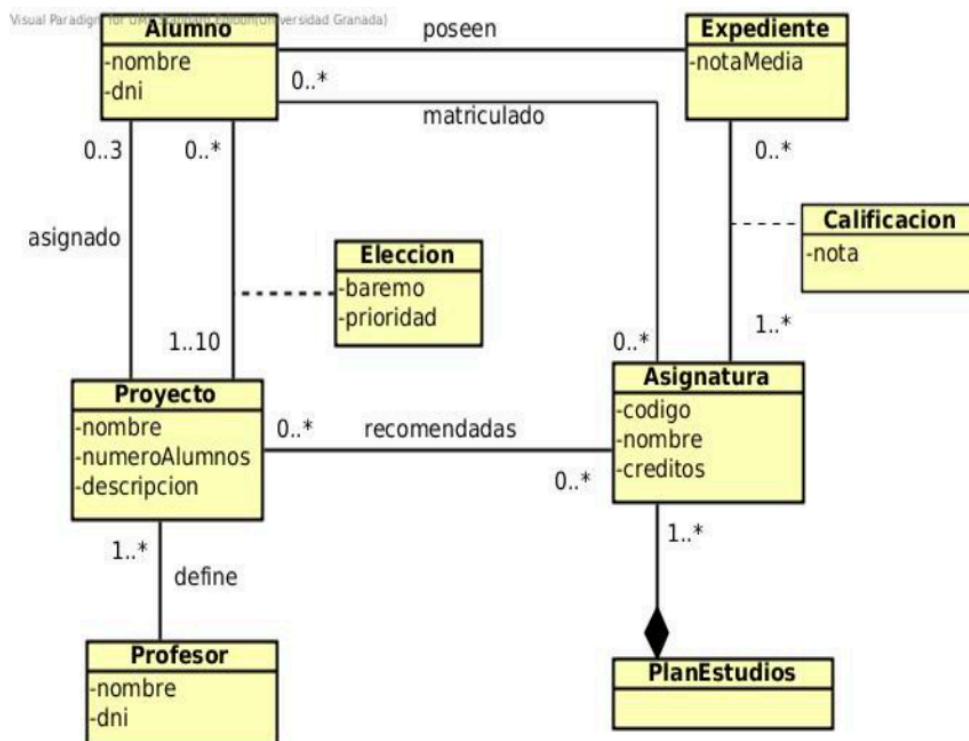
- Primitivos o valores puros de datos:(Entero, Real, Carácter, Boolean, Cadena).
- No primitivos: (Nombre de persona, Número de teléfono, Hora, Fecha, Dirección, Punto,...)

Identificarlos:

4. Añadir atributos: identificarlos



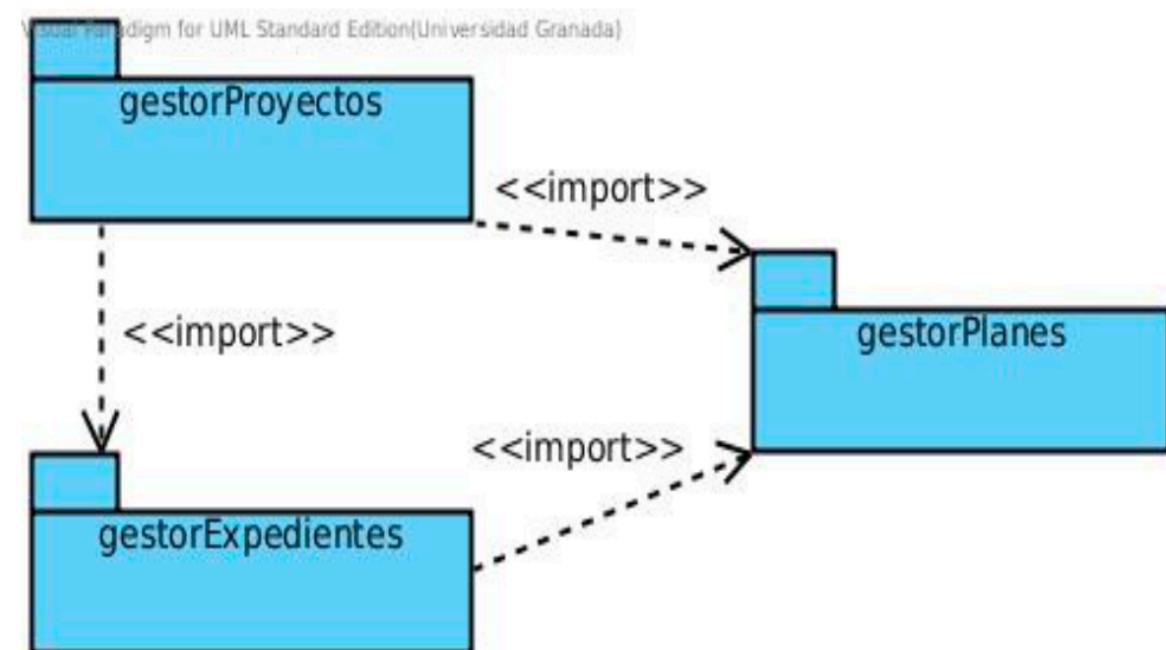
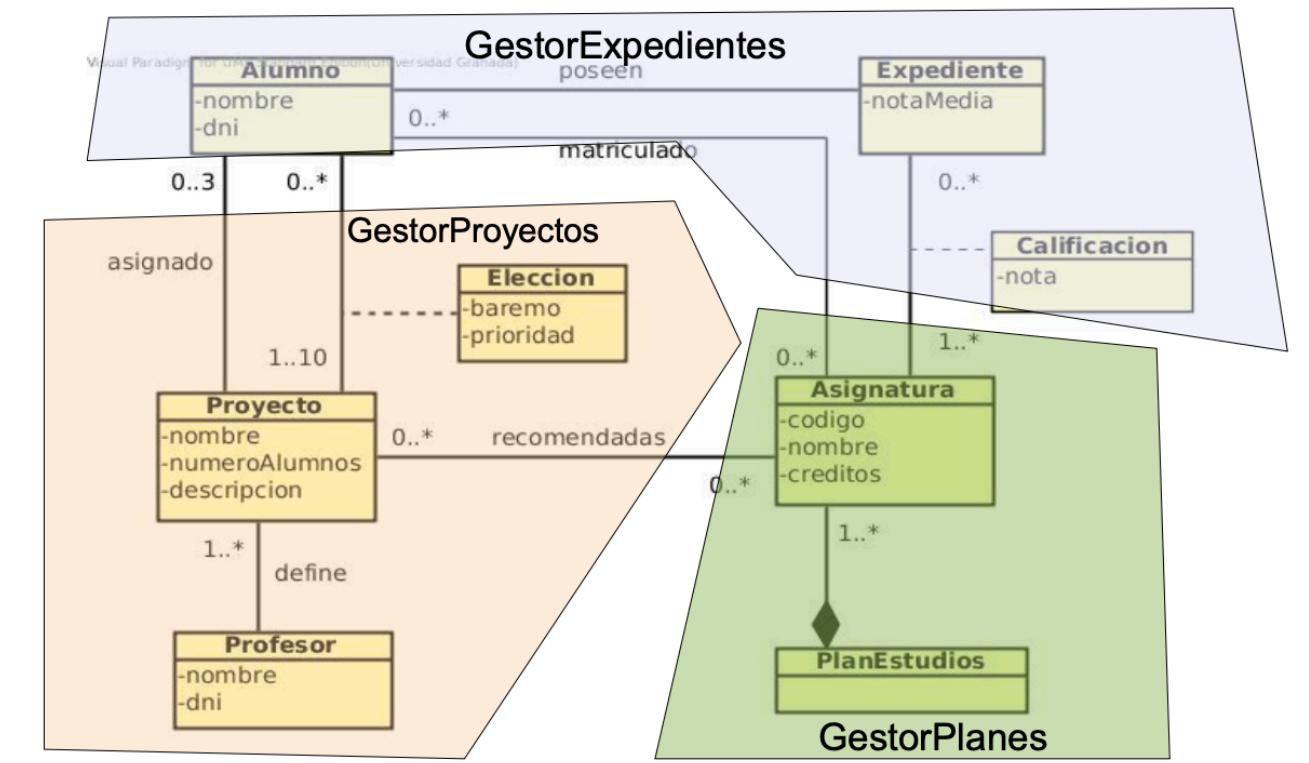
4. Añadir atributos: Representarlos

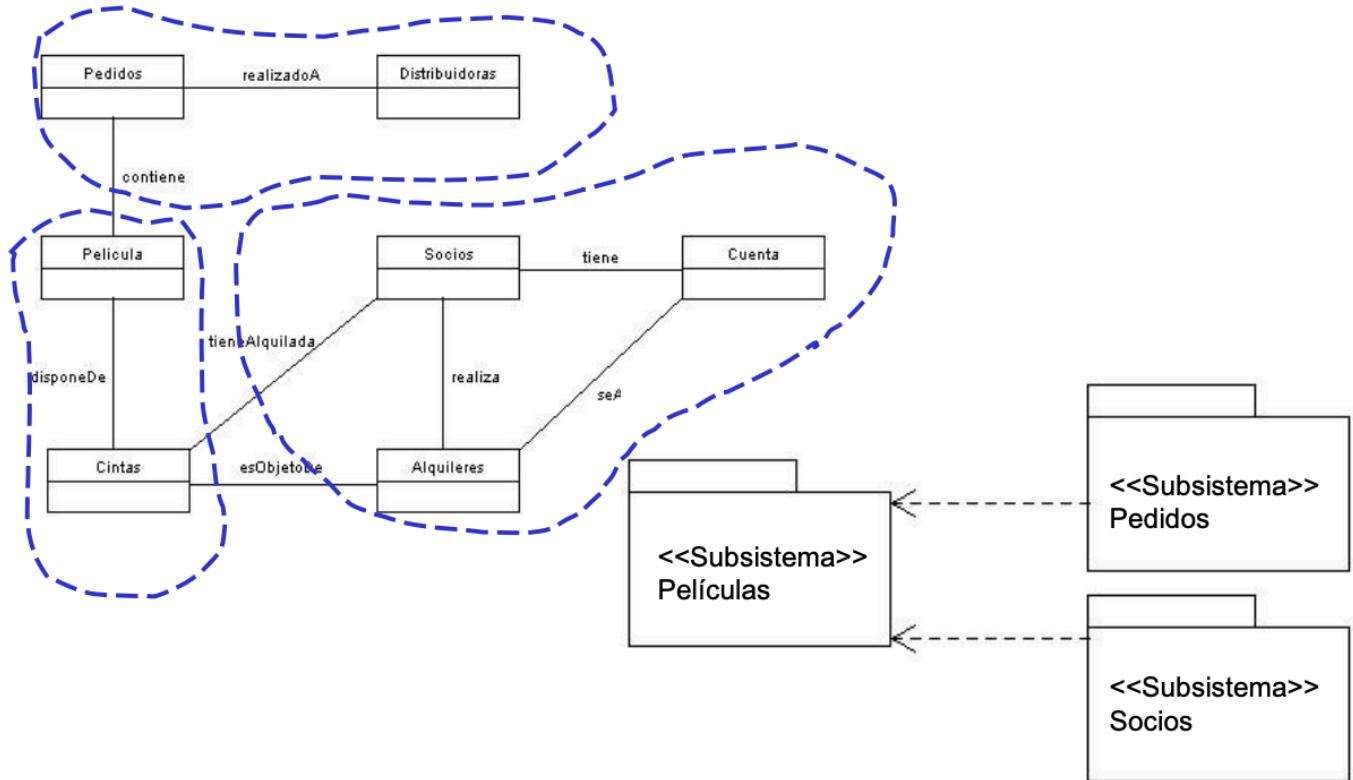


5. Estructurar y empaquetar el modelo.

Usando un diagrama de paquetes de UML Guía para estructurar el diagrama de conceptos o modelo de dominio

- Elementos que están en el mismo área de interés (relacionados por conceptos)
- Están juntos en una jerarquía de clases
- Participan en los mismos casos de uso
- Están fuertemente asociados

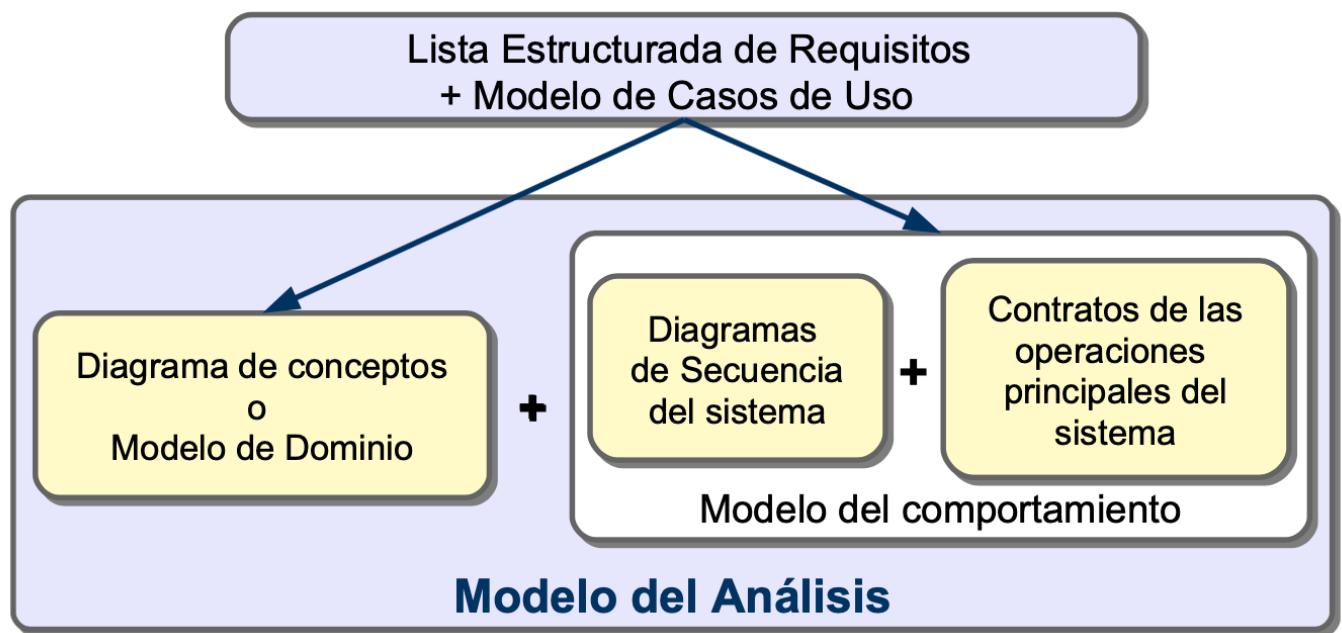


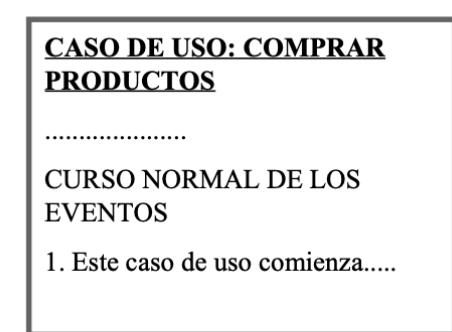


Modelo del comportamiento

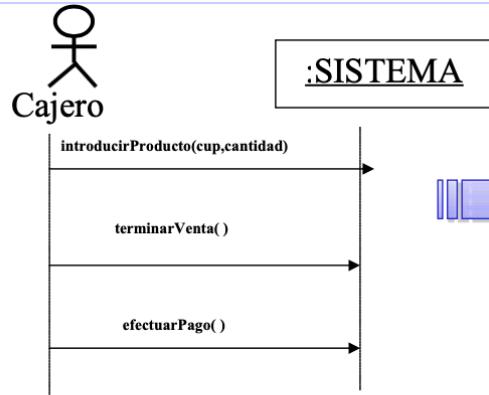
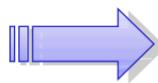
Estudio adicional del dominio del problema en el que añadimos los requisitos funcionales al modelo del análisis

“Qué hace el sistema sin explicar cómo lo hace”

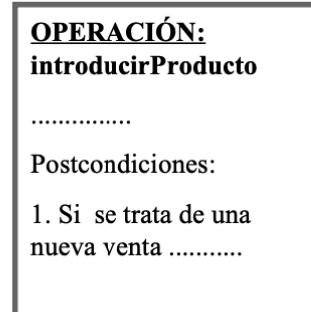
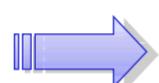
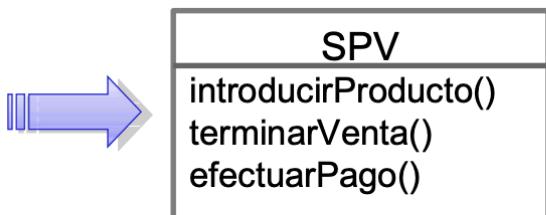




CASOS DE USO



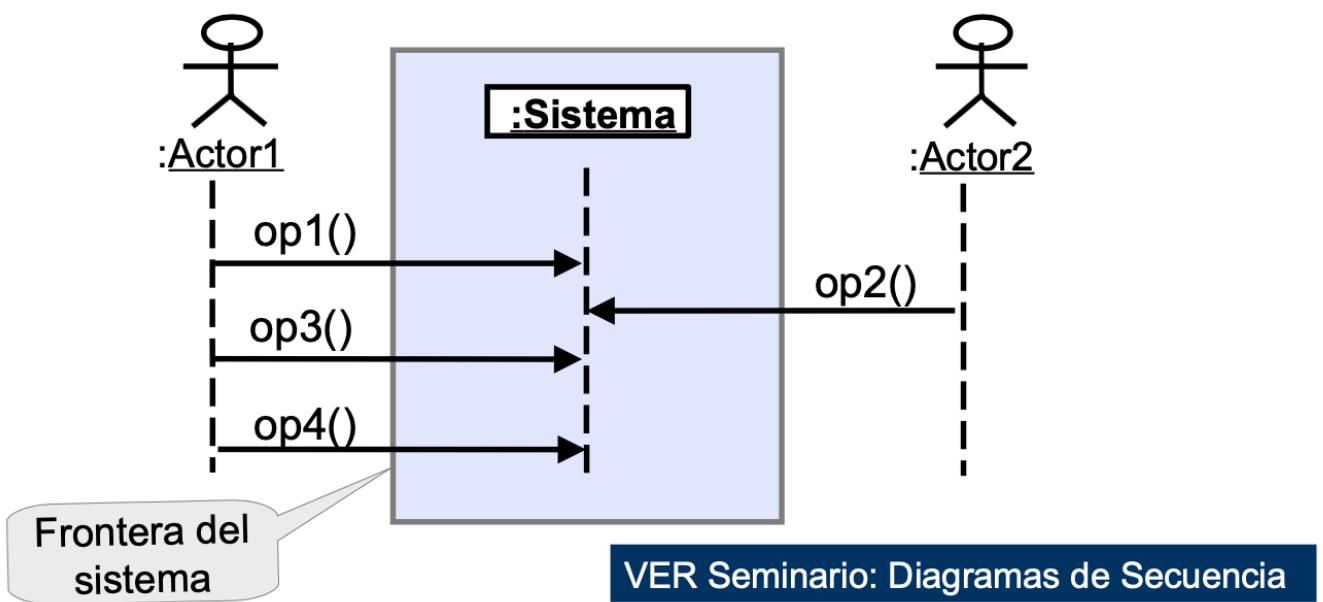
DIAGRAMAS DE SECUENCIA DEL SISTEMA (DSS)



OPERACIONES DEL SISTEMA

CONTRATOS

Un **Diagrama de Secuencia del Sistema (DSS)** es un Diagrama de Secuencia UML en el que se muestran como los eventos generados por los actores van a provocar la ejecución de una operación por el Sistema, siendo visto éste como una caja negra.



Pasos a seguir, para todos los casos de uso:

1. Identificar los **actores** que inician dichas operaciones
2. Asignar un **nombre** a todo el **sistema**

3. Identificar y nombrar las **operaciones principales del sistema** de las descripciones de los Casos de Uso
4. Ver cuáles serían los **parámetros de las operaciones**
5. Representarlas en el diagrama de secuencia del sistema (DSS)
6. Incluir las operaciones en la clase que identifica a todo el sistema del Diagrama conceptual.

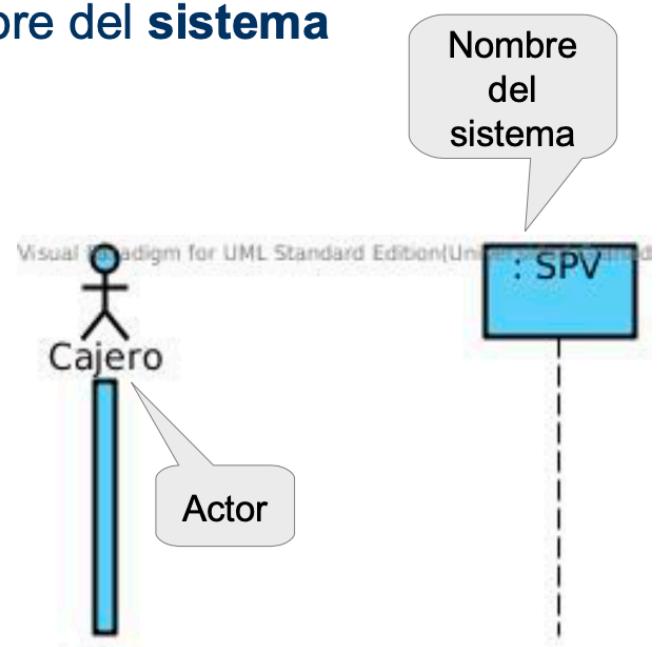
Podemos tener **un DSS para cada CU** o **un solo DSS** con todas las operaciones del sistema o **un DSS por diagrama de casos de uso**.

Elaboración DSS

1 y 2. Identificar actores y nombre del sistema

CU01: Procesar Venta con pago efectivo

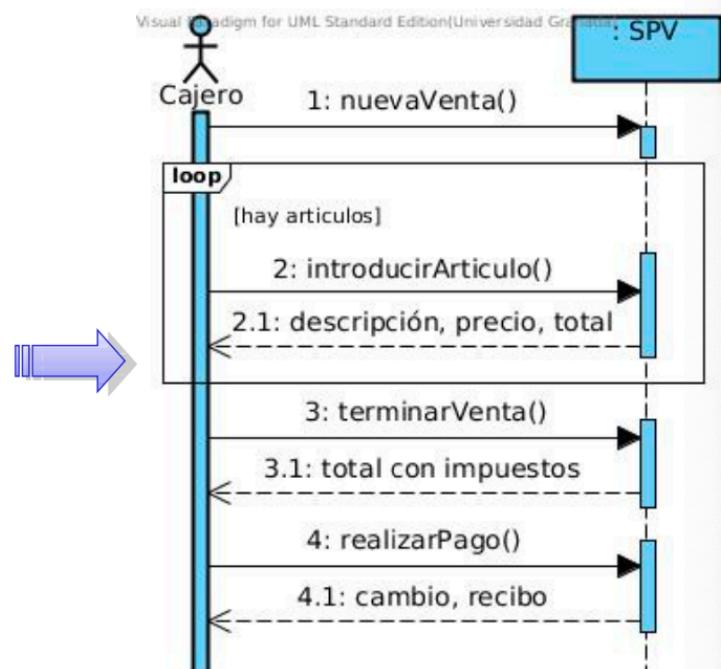
1. El Cliente llega al terminal
2. El Cajero **inicia una nueva Venta**
3. El Cajero **inserta el identificador de artículo**
4. El Sistema registra la línea de venta y presenta la descripción del artículo, precio y suma parcial
El Cajero **repite 3 y 4** hasta que se indique **fin de venta**
5. El Sistema muestra el total con los impuestos calculados
6. El Cajero indica al Cliente el total y pide que le pague
7. El Cliente Paga y el sistema **gestiona el pago**



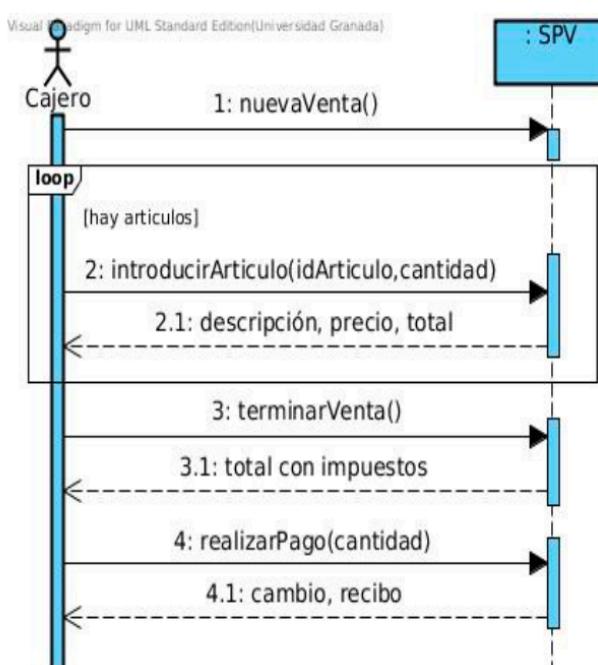
3, 4 y 5. Identificar operaciones y parámetros y representarlas en el DSS

CU01: ProcesarVenta con pago efectivo

1. El Cliente llega al terminal
2. El Cajero **inicia una nueva Venta**
3. El Cajero **inserta el identificador de artículo**
4. El Sistema registra la línea de venta y presenta la descripción del artículo, precio y suma parcial
- El Cajero **repite 3 y 4 hasta que se indique fin de venta**
5. El Sistema muestra el total con los impuestos calculados
6. El Cajero indica al Cliente el total y pide que le pague
7. El Cliente Paga y el sistema **gestiona el pago**



6. Incluir las operaciones en el Diagrama conceptual

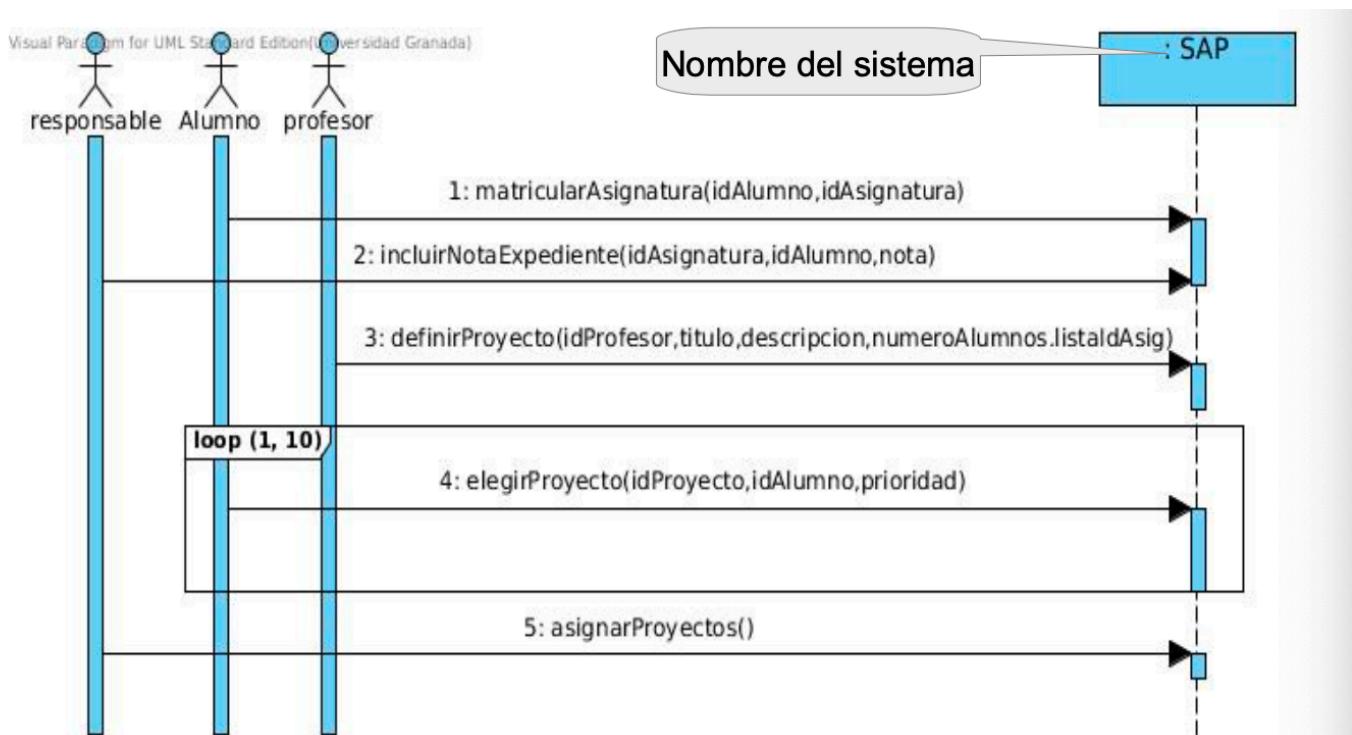
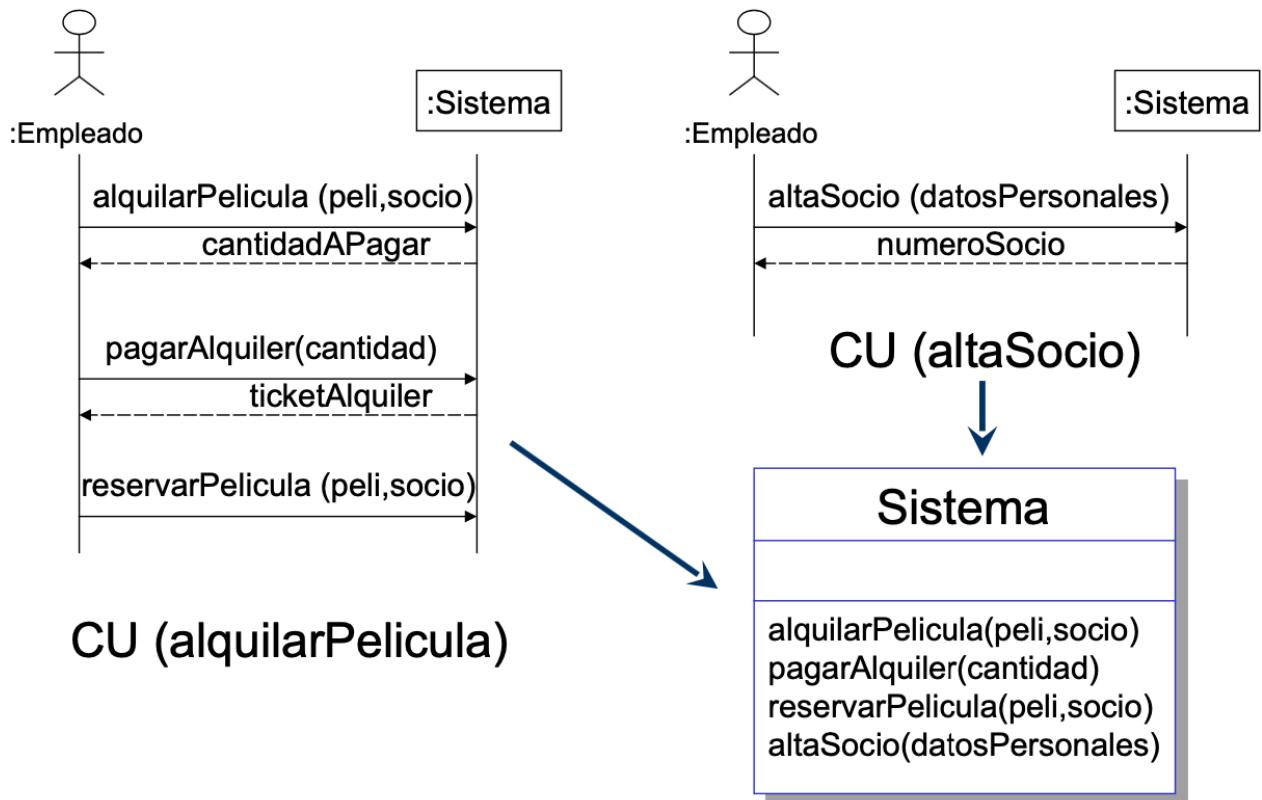


DSS del caso de uso ProcesarVenta en efectivo

SPV
+nuevaVenta()
+introducirArticulo(idArticulo, cantidad)
+terminarVenta()
+realizarPago(cantidad)

Clase que representa a todo el sistema

6. Incluir las operaciones en el Diagrama conceptual (continuación)



Ejercicio: representar la clase SAP junto con sus operaciones en el Diagrama Conceptual

Contratos

Contenido del contrato

Nombre	<i><<Nombre de la operación y sus parámetros>></i>
Responsabilidad	<i><<Descripción informal de las responsabilidades que debe cumplir la operación>></i>
Tipo	<i><<Concepto, clase o interfaz responsable de la operación>></i>
Notas	<i><<Notas de diseño, algoritmo...>></i>
Excepciones	<i><<Casos excepcionales>></i>
Salida	<i><<Mensajes o datos que proporciona>></i>
Precondiciones	<i><<Suposición acerca del estado del sistema o de los objetos del modelo conceptual antes de ejecutar la operación>></i>
Poscondiciones	<i><<Estado del sistema o de los objetos del modelo conceptual después de la ejecución de la operación>></i>

Directrices generales

- El **nombre de la operación** viene del DSS correspondiente. Comenzar con las **responsabilidades**, describiendo informalmente el propósito de la operación.
- Continuar con las **poscondiciones** y finalizar con las demás secciones, especialmente con las **precondiciones** y **excepciones**.
- Las **poscondiciones** deben describir los cambios de estado de un sistema no sus acciones (espíritu escenario-telón), éstos son:
 - Creación y destrucción de objetos.
 - Creación y destrucción de enlaces.
 - Modificación de atributos.
- Los *objetos y enlaces que se pueden crear y destruir son los que están en el Modelo Conceptual*.
- Las **poscondiciones** deben expresarse mediante una frase verbal en pretérito.

Ejemplo: contrato de la operación MatricularAlumno(*idAlumno,idAsignatura*)

Nombre	matricularAsignatura(idAlumno,idAsignatura)
Responsabilidad	Matricular al alumno identificado por idAlumno en la asignatura identificada por idAsignatura
Tipo	SAP
Notas	
Excepciones	<ul style="list-style-type: none"> Si el alumno identificado por idAlumno no existe Si la asignatura identificada por idAsignatura no existe
Salida	
Precondiciones	
Poscondiciones	???

Poscondiciones del contrato

Para especificar **las poscondiciones**, hay que identificar en el diagrama de conceptos los objetos que intervienen en la operación

En la operación matricularAsignatura intervienen los siguientes objetos de las clases conceptuales: Alumno y Asignatura.

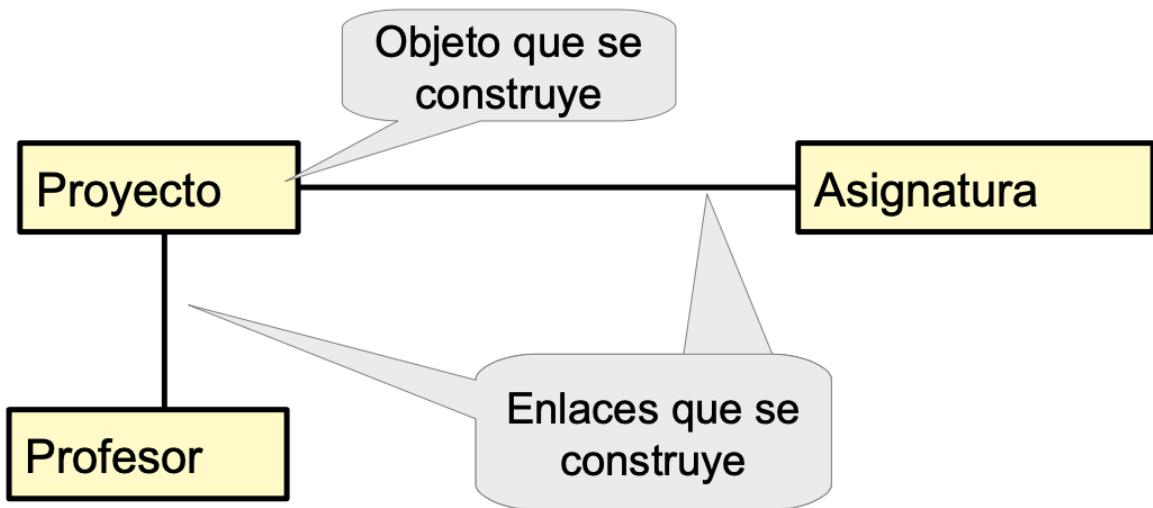
Poscondiciones matricularAsignatura: Fue creado un enlace entre el objetos de la clases Alumno (identificado por idAlumno) y el objetos de la clase Asignatura (identificado por idAsignatura)

+ Ejemplos:

Contrato de la operación:

definirProyecto(idProfesor,titulo,descripcion,numeroAlumnos,listIdAsig)

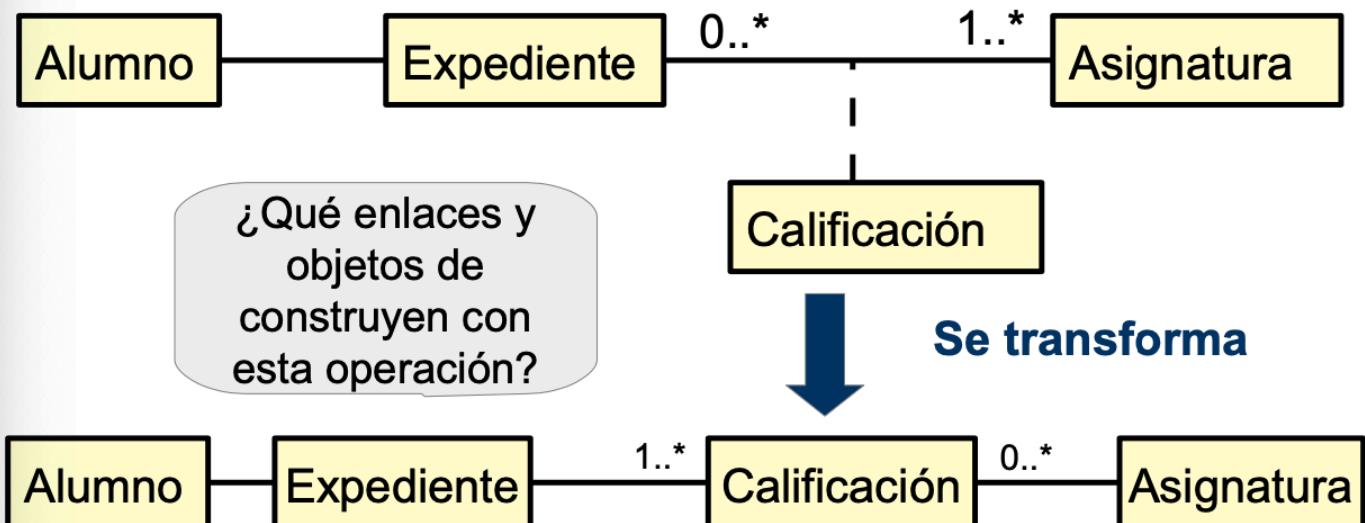
En esta operación intervienen objetos de las clases conceptuales: **Profesor, Proyecto y Asignatura**.



Nombre	definirProyecto(idProfesor,titulo,descripcion, numeroAlumnos,listIdAsig)
Responsabilidad	Crea un nuevo proyecto inicializando su estado y asignandole el profesor que lo define y las asignaturas recomendadas
Tipo	SAP
Notas	
Excepciones	<ul style="list-style-type: none"> Si el profesor identificado por idProfesor no existe Si algunas de las asignaturas identificadas por alguno de los elementos de listIdAsig no exista
Salida	
Precondiciones	
Poscondiciones	<ul style="list-style-type: none"> Fue creado un objeto, pro, de la clase Proyecto debidamente inicializado. Fue creado un enlace entre pro y el objeto Profesor, identificado por idProfesor. Para todos los elementos de listIdAsig Fue creado un enlace entre pro y el objeto de la clase Asignatura identificado por el correspondiente elemento de listIdAsig

Contrato de la operación:
incluirNotaExpediente(idAsignatura,idAlumno,nota)

En esta operación intervienen objetos de las clases conceptuales:
Alumno, Expediente y Asignatura.



Nombre	incluirNotaExpediente(idAsignatura,idAlumno,nota)
Responsabilidad	Incluye una asignatura con su nota en el expediente de un alumno.
Tipo	SAP
Notas	
Excepciones	<ul style="list-style-type: none"> Si el alumno identificado por idAlumno no existe Si la asignatura identificada por idAsignatura no existe
Salida	
Precondiciones	
Poscondiciones	<ul style="list-style-type: none"> Fue creado un objeto, calificacion, de la clase Calificacion debidamente inicializado. Fue creado un enlace entre calificacion y el objeto Expediente, identificado por idAlumno. Fue creado un enlace entre calificacion y el objetos Asignatura, identificado por idAsignatura.