Índice

INTRODUCCIÓN.

El software ha evolucionado en gran medida gracias a la ingeniería del software. Se define ingeniería del software como la aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software.

En el paradigma de la orientación al objeto, un sistema se concibe como un conjunto de objetos que se comunican entre sí mediante mensajes. El concepto de objeto es: una entidad percibida por el sistema que se está desarrollando. A nivel de implementación se define como un **encapsulamiento** de un conjunto de operaciones (servicios) que pueden ser invocadas de manera externa y de un **estado** que recuerda el resultado de los servicios.

El encapsulamiento es un principio de abstracción que agrupa datos y procesos permitiendo ocultar a los usuarios de un objeto los aspectos de implementación, ofreciéndoles una interfaz externa mediante la cual poder interactuar con el objeto.

Un objeto se describe por sus sus propiedades, también llamadas atributos -estructura del objeto- y por los servicios que puede proporcionar -comportamiento del objeto-. El estado de un objeto viene determinado por los valores que toman los atributos. Se denomina clase a la implementación de un tipo de objeto (considerando los objetos como instancias de las clases).

En el desarrollo orientado al objeto se pueden distinguir las tres fases tradicionales: análisis, diseño e implementación. Sin embargo, hay que resaltar que estas fases se solapan y presentan unas fronteras mas difuminadas que en el desarrollo tradicional.

Existen precedentes de múltiples notaciones diferentes para especificar un sistema software. Finalmente, se impuso UML (Unified Modeling Language), un lenguaje gráfico de modelado para visualizar, especificar, construir y documentar las especificaciones de un sistema software. La metodología Métrica 3 tiene en cuenta la mayoría de las técnicas que contempla UML.

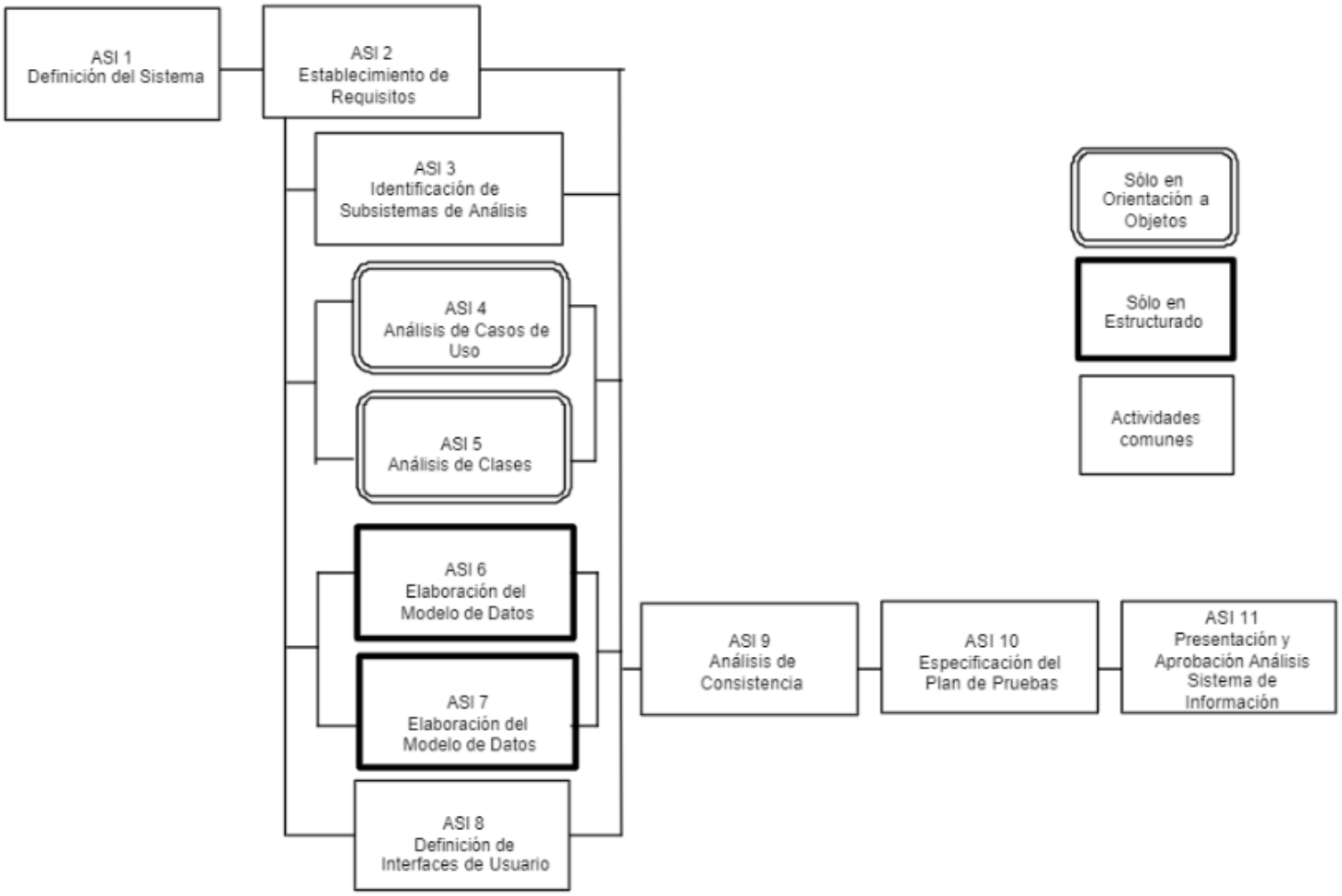
1. ANÁLISIS ORIENTADO A OBJETOS.

La metodología MÉTRICA Version 3 descompone el ciclo de desarrollo de software en procesos, cada uno de estos a su vez en actividades, y éstas a su vez en tareas.
Los procesos de la estructura principal de MÉTRICA Versión 3 son los siguientes:

PLANIFICACIÓN DE SISTEMAS DE INFORMACIÓN.
DESARROLLO DE SISTEMAS DE INFORMACIÓN.
MANTENIMIENTO DE SISTEMAS DE INFORMACIÓN.

El Proceso de Desarrollo de Sistemas de Información, para facilitar la comprensión y dada su amplitud y complejidad se ha subdividido en cinco procesos:

ESTUDIO DE VIABILIDAD DEL SISTEMA (EVS).
ANÁLISIS DEL SISTEMA DE INFORMACIÓN (ASI).
DISEÑO DEL SISTEMA DE INFORMACIÓN (DSI).
CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI).
IMPLANTACIÓN Y ACEPTACIÓN DEL SISTEMA (IAS).



1.1. Actividad ASI 1: definición del sistema.

Efectúa una descripción del sistema, estableciendo las interfaces con otros sistemas e identificando a los usuarios representativos.

Se parte de los productos obtenidos en el proceso de Estudio de Viabilidad del Sistema – EVS.

1.2. Actividad ASI 2: establecimiento de requisitos.

Aquí se lleva a cabo la definición, análisis y validación de los requisitos, completándose el catálogo de requisitos obtenidos en la actividad de Definición del Sistema (ASI 1). El objetivo de esta actividad es obtener un catálogo detallado de los requisitos.

Se propone como técnica de obtención de requisitos la especificación de los casos de uso de la orientación a objetos ya que ofrece un diagrama simple y una guía de especificación en las sesiones de trabajo con el usuario.

1.3. Actividad ASI 3: identificación de subsistemas de análisis.

Facilita el análisis del sistema de información llevando a cabo la descomposición del sistema en subsistemas.

1.4. Actividad ASI 4: análisis de los casos de uso.

Identifica las clases cuyos objetos son necesarios para realizar un caso de uso y describir su comportamiento mediante la interacción de dichos objetos.

Se lleva a cabo para cada uno de los casos de uso contenidos en un subsistema de los definidos en la actividad Identificación de Subsistemas de Análisis (ASI 3), obteniéndose los diagramas de clases y el diagrama de interacción de objetos.

La realización de esta actividad no se realizan de forma secuencial sino en paralelo, con continuas realimentaciones entre ellas y con las realizadas en las actividades Establecimiento de Requisitos (ASI 2), Identificación de Subsistemas de Análisis (ASI 3), Análisis de Clases (ASI 5) y Defi8nición de Interfaces de Usuario (ASI 8).

Tarea		Productos	Técnicas y Prácticas	Participantes
ASI 4.1	Identificación de Clases Asociadas a un Caso de Uso	- Modelo de Clases de Análisis	- Diagrama de Clases	- Analistas
ASI 4.2	Descripción de la Interacción de Objetos	- Análisis de la Realización de los Casos de Uso	- Diagrama de Interacción de Objetos (secuencia o colaboración)	- Analistas

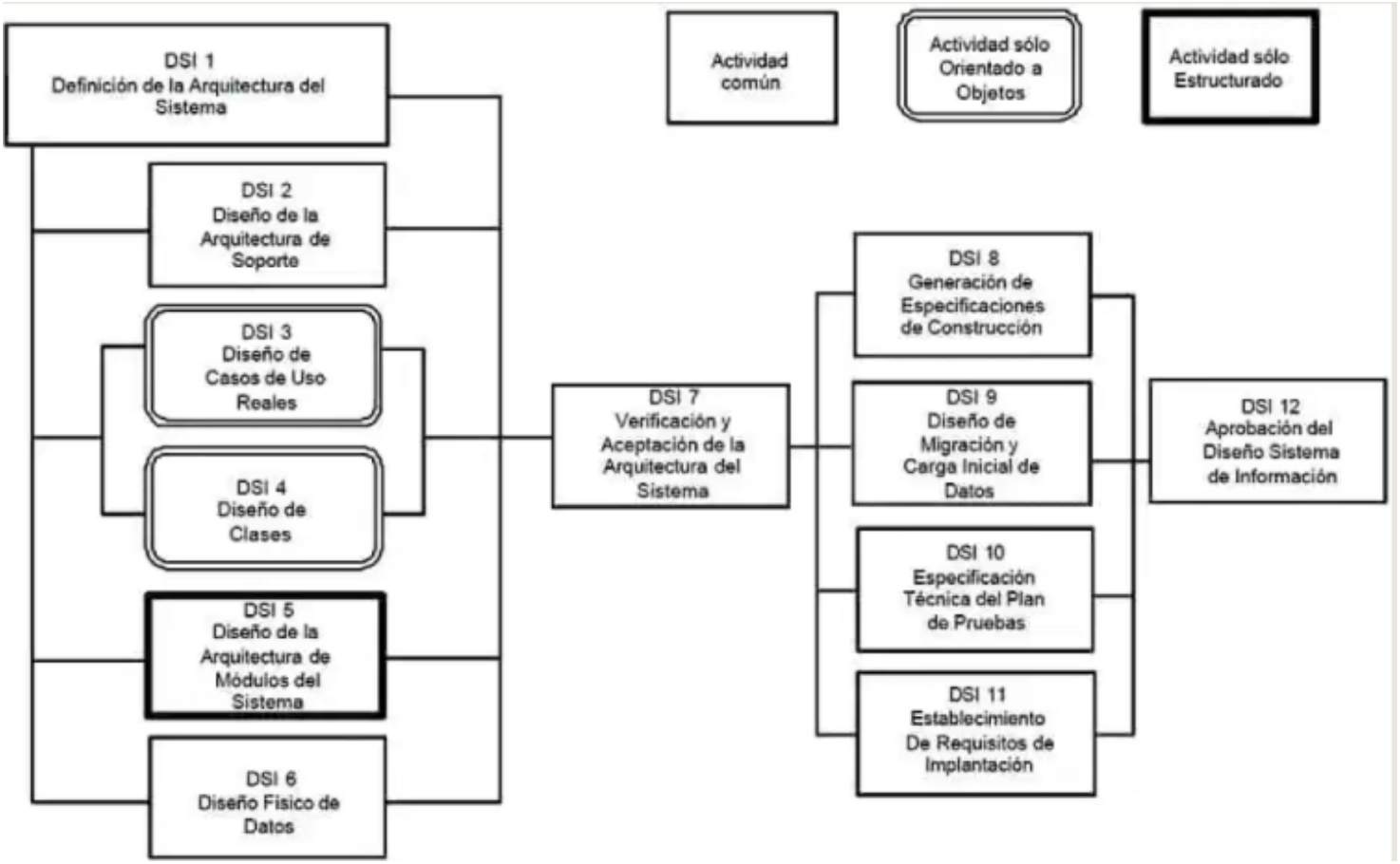
1.5. Actividad ASI 5: análisis de clases.

A partir de las clases identificadas en la actividad Análisis de los casos de Uso (ASI 4), se elabora el modelo de clases para cada subsistema. A medida que avanza el análisis, dicho modelo se va completando con las clases que vayan apareciendo, tanto del estudio de los casos de uso, como de la interfaz de usuario necesaria para el sistema de información.

Tarea		Productos	Técnicas y Prácticas	Participantes
ASI 5.1	Identificación de Responsabilidades y Atributos	- Modelo de Clases de Análisis - Comportamiento de Clases de Análisis	- Diagrama de Clases - Diagrama de Transición de Estados	- Analistas
ASI 5.2	Identificación de Asociaciones y Agregaciones	- Modelo de Clases de Análisis	- Diagrama de Clases	- Analistas
ASI 5.3	Identificación de Generalizaciones	- Modelo de Clases de Análisis	- Diagrama de Clases	- Analistas

2. DISEÑO ORIENTADO A OBJETOS.

El objetivo del proceso de Diseño del Sistema de Información (DSI) es la definición de la arquitectura del sistema y del entorno tecnológico que le va a dar soporte.



Las actividades de este proceso se agrupan en dos grandes bloques:

1. En un **primer bloque** de actividades, que se llevan a cabo en paralelo, se obtiene el diseño de detalle del sistema de información.

En la actividad **Definición de la Arquitectura del Sistema (DSI 1)**, se establece el particionamiento físico del sistema de información, la especificación del entorno tecnológico, y sus requisitos de operación, administración, seguridad y control de acceso.

El **Diseño de la Arquitectura de Soporte (DSI 2)**, incluye el diseño detallado de los subsistemas de soporte.

La actividad **Diseño de Casos de Uso Reales (DSI 3)**, obtiene el diseño del comportamiento del sistema de información para los casos de uso, el diseño de la interfaz de usuario y la validación de la división en subsistemas.

La **Actividad Diseño de Clases (DSI 4)**, obtiene el diseño detallado de cada una de las clases que forman parte del sistema.

Una vez que se tiene el modelo de clases, se comienza el diseño físico en la actividad **Diseño Físico de Datos (DSI 6)**, que incluye el diseño y optimización de las estructuras de datos del sistema.

En la actividad **Verificación y Aceptación de la Arquitectura del Sistema (DSI 7)**, se revisa y valida el diseño de detalle.

2. En el **segundo bloque** de actividades se generan todas las especificaciones necesarias para la construcción del sistema de información.

- Generación de Especificaciones de Construcción (DSI 8).
- Diseño de la Migración y Carga Inicial de Datos (DSI 9).
- Especificación Técnica del Plan de Pruebas (DSI 10).
- Establecimiento de Requisitos de Implantación (DSI 11).

En la actividad de **Presentación y Aprobación del Diseño del Sistema de Información (DSI 12)**, se aprueban los distintos productos del diseño.

3. TÉCNICAS DE ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS.

Disponemos de técnicas de modelado expuestas a continuación.

3.1. Casos de uso.

Facilitan la comunicación entre cliente y desarrollador. Los objetivos de los casos de uso son los siguientes:

- Capturar los requisitos funcionales del sistema.
- Guiar todo el proceso de desarrollo del sistema de información.

Un caso de uso es una secuencia de acciones realizadas por el sistema, que producen un resultado observable y valioso para un usuario en particular.

Los diagramas de casos de uso presentan dos tipos de elementos fundamentales:

- **Actores**, que serán los usuarios del sistema.
- **Casos de uso,** que serán un conjunto de transacciones ejecutadas entre el sistema y los actores.

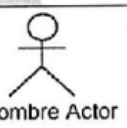
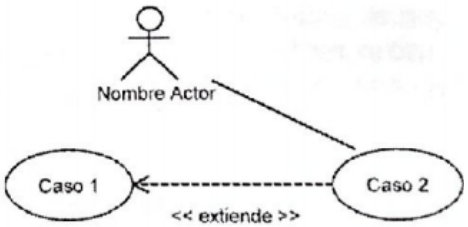
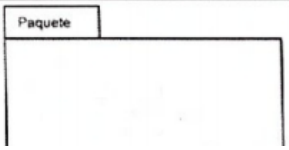
Un diagrama de casos de uso representa relaciones, que pueden ser entre actores y casos de uso o entre casos de uso.

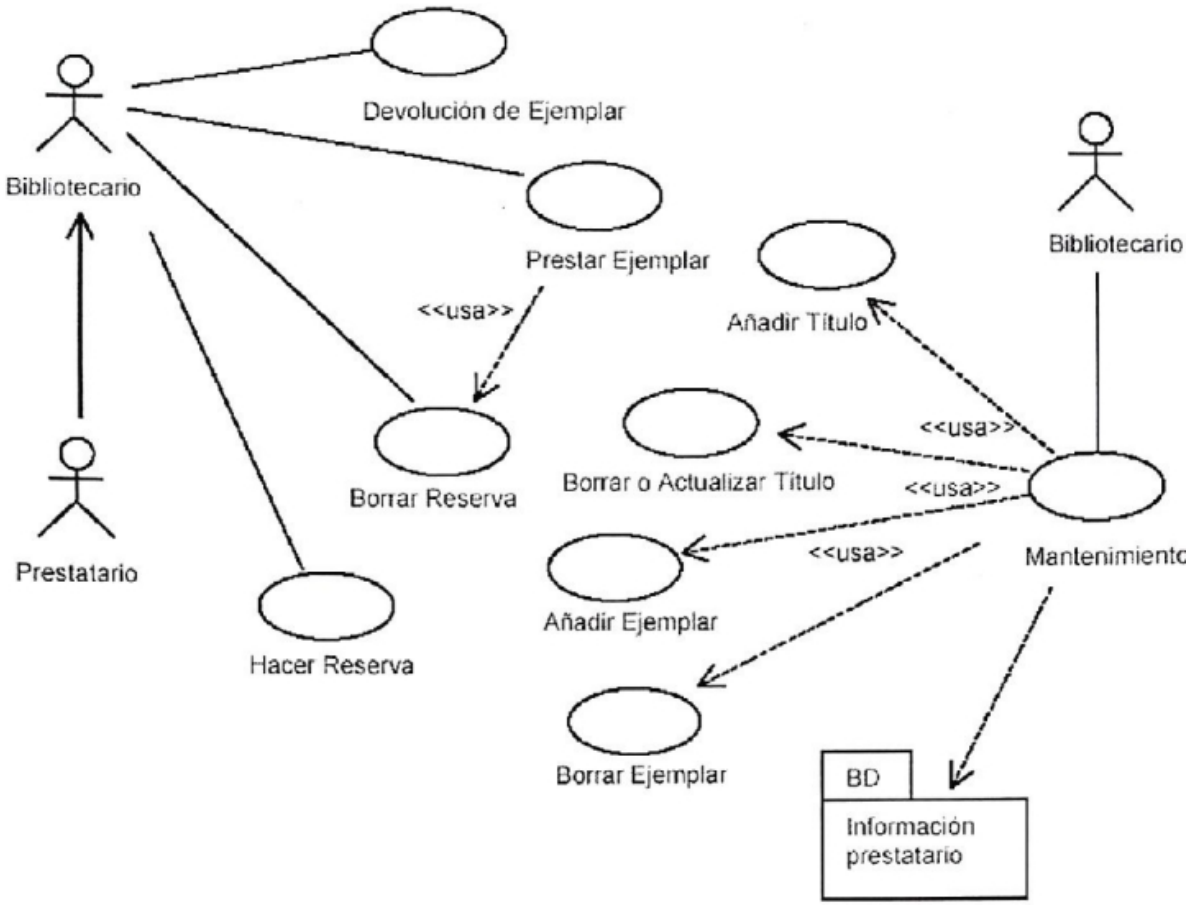
La relación entre un actor y un caso de uso es una relación de comunicación, que indica que un actor interviene en el caso de uso.

El actor aporta información para la realización de un caso de uso o recibe información como resultado de la realización del mismo, por ello, esta relación puede ser unidireccional o bidireccional.

La relación entre casos de uso es una relación unidireccional. Esta relación puede ser: "usa" cuando se quiere reflejar un comportamiento común en varios casos de uso o "extiende" cuando se quiere reflejar un comportamiento opcional de un caso de uso.

El diagrama de casos de uso es un grafo de actores, casos de uso y las relaciones entre estos elementos.

Caso de Uso: Un caso de uso se representa mediante una elipse con el nombre del caso de uso dentro o debajo.	
Actor: Un actor se representa con una figura de 'hombre de palo' con el nombre del actor debajo de la figura.	
Relación: Dependiendo del tipo de relación, la representación en los diagramas será distinta. Así pues, las relaciones entre un actor y un caso de uso se representan mediante una línea continua entre ellos. Las relaciones entre casos de uso se representan con una flecha discontinua con el nombre del tipo de relación como etiqueta. En las relaciones "extensión" la flecha parte del caso de uso con el comportamiento adicional hacia aquel que recoge el comportamiento básico y en las relaciones "usa" desde el caso de uso básico hacia el que representa el comportamiento común.	
Paquete: Un paquete se representa con un icono con forma de carpeta y con el nombre colocado en la 'pestaña'. Los paquetes también pueden formar diagramas que complementen al diagrama de casos de uso (ver *Diagrama de paquetes*).	



3.2. Diagrama de clases.

Es la representación de los aspectos estáticos del sistema. Aquí se representa la estructura y el comportamiento de cada uno de los objetos del sistema y sus relaciones con los demás objetos. Los elementos básicos del diagrama son:

3.2.1. Clases.

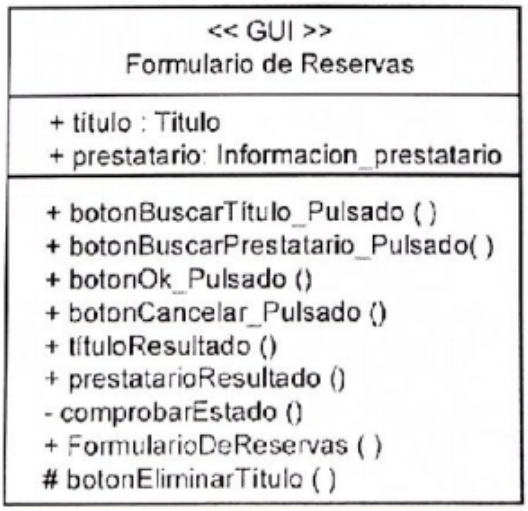
Describe un conjunto de objetos con propiedades (atributos) similares y un comportamiento común. Los objetos son instancias de las clases. Dentro de la estructura de una clase se definen:

Atributos: representan los datos asociados a los objetos instanciados por esa clase.

Operaciones o métodos: representan las funciones o procesos propios de los objetos de una clase, caracterizando a dichos objetos.

Una clase se representa como una caja, separada en tres zonas por líneas horizontales:

- En la zona superior se muestra el nombre de la clase y propiedades generales como el estereotipo. El nombre de la clase aparece centrado y si la clase es abstracta se representa en cursiva. El estereotipo, si se muestra, se sitúa sobre el nombre y entre el símbolo: << .... >>
- La zona central contiene una lista de atributos, uno en cada línea. La notación utilizada para representarlos incluye, dependiendo del detalle, el nombre del atributo, su tipo y su valor por defecto
  La visibilidad será en general publica (+), privada (-) o protegida (#), aunque puede haber otros tipos de visibilidad dependiendo del lenguaje de programación empleado.
- En la zona inferior se incluye una lista con las operaciones que proporciona la clase. Cada operación aparece en una línea.



3.2.2. Relaciones.

Los tipos más importantes de relaciones estáticas entre clases son los siguientes:

Asociación. Las relaciones de asociación representan un conjunto de enlaces entre objetos o instancias de clases. Es el tipo de relación más general, y denota básicamente una dependencia semántica. Por ejemplo, una persona trabaja para una Empresa. Cada asociación puede presentar elementos adicionales que doten de mayor detalle al tipo de relación:

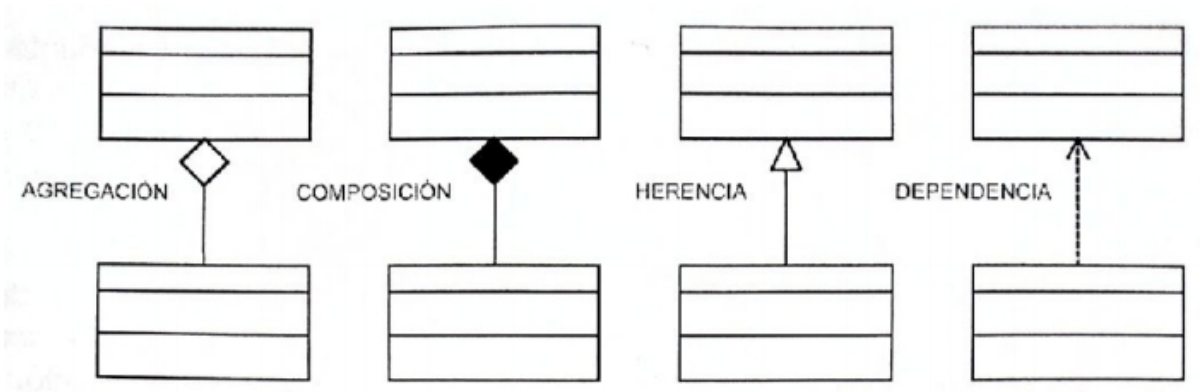
- **Rol**, o nombre de la asociación, que describe la semántica de la relación en el sentido indicado. Por ejemplo, la asociación entre Persona y Empre sa recibe el nombre de trabaja para, como rol en ese sentido.
- **Multiplicidad**, que describe la cardinalidad de la relación, es decir, especifica cuántas instancias de una clase están asociadas a una instancia de la otra clase. Los tipos de multiplicidad son: Uno a uno, uno a muchos y muchos a muchos.

Herencia. Las jerarquías de generalización especialización se conocen como herencia. Herencia es el mecanismo que permite a una clase de objetos incorporar atributos y métodos de otra clase, añadiéndolos a los que ya posee. Con la herencia se refleja una relación "es_un" entre clases. La clase de la cual se hereda se denomina superclase, y la que hereda subclase.
La generalización define una superclase a partir de otras. Por ejemplo, de las cla ses profesor y estudiante se obtiene la superclase persona. La especialización o especificación es la operación inversa, y en ella una clase se descompone en una o varias subclases. Por ejemplo, de la clase empleado se pueden obtener las subclases secretaria, técnico e ingeniero. Esta relación se representa como una línea continua con una flecha hueca en el extremo que apunta a la superclase.

Agregación. La agregación es un tipo de relación jerárquica entre un objeto que representa la totalidad de ese objeto y las partes que lo componen. Permite el agrupamiento físico de estructuras relacionadas lógicamente. Los objetos "son parte-de" otro objeto completo. Por ejemplo, motor, ruedas, carrocería son parte de automóvil. Se representa con un rombo hueco en la clase cuya instancia es una agregación de las instancias de la otra.

Composición. La composición es una forma de agregación donde la relación de propiedad es más fuerte, e incluso coinciden los tiempos de vida del objeto com pleto y las partes que lo componen. Por ejemplo, en un sistema de Máquina de café, las relaciones entre la clase máquina y producto, o entre máquina y depósi
to de monedas, son de composición. Se representa con un rombo lleno en la cla COPIA se cuya instancia contiene las instancias de la otra clase.

Dependencia. Una relación de dependencia se utiliza entre dos clases o entre una clase y una interfaz, e indica que una clase requiere de otra para proporcio nar alguno de sus servicios. Una línea discontinua con una flecha apuntando a la clase cliente. La relación puede tener un estereotipo que se coloca junto a la lí nea, y entre el simbolo: <<...>>.



Una relación de asociación se representa como una línea continua entre las clases asociadas. En una relación de asociación, ambos extremos de la línea pueden conectar con la misma clase, indicando que una instancia de una clase, está asociada a otras instancias de la misma clase, lo que se conoce como asociación reflexiva.

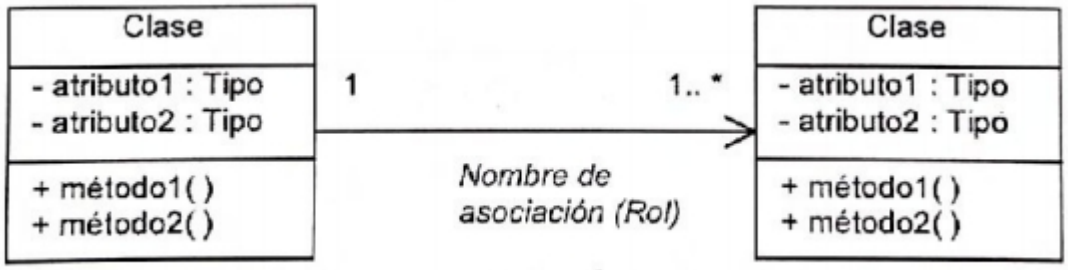
Las diferentes propiedades de la relación se pueden representar con la siguiente notación:

Multiplicidad: La multiplicidad puede ser un número concreto, un rango o una colección de números. La letra 'n' y el símbolo '* representan cualquier número.

Orden: Se puede especificar si las instancias guardan un orden con la palabra clave '{ordered}'. Si el modelo es suficientemente detallado, se puede incluir una restricción que indique el criterio de ordenación.

Navegabilidad: La navegación desde una clase a la otra se representa poniendo una flecha sin relleno en el extremo de la línea, indicando el sentido de la navegación.

Rol o nombre de la asociación: Este nombre se coloca junto al extremo de la línea que está unida a una clase, para expresar cómo esa clase hace uso de la otra clase con la que mantiene la asociación.



Asociación.

3.2.3. INTERFACES.

Una interfaz es una especificación de la semántica de un conjunto de operaciones de una clase o paquete que son visibles desde otras clases o paquetes. Normalmente, se corresponde con una parte del comportamiento del elemento que la proporciona.

Una interfaz se representa como una caja con compartimentos, igual que las clases.

3.2.4. PAQUETES.

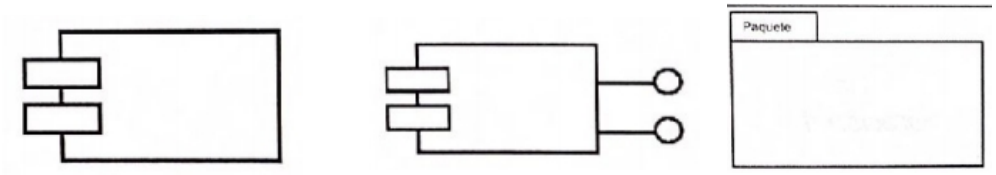
Se usan para dividir el modelo de clases del sistema de información, agrupando clases u otros paquetes según los criterios que sean oportunos. Los paquetes se representan mediante un icono con forma de carpeta y las dependencias con flechas discontinuas entre los paquetes dependientes.

3.3. DIAGRAMA DE COMPONENTES.

El diagrama de componentes proporciona una visión física de la construcción del sistema de información. Muestra la organización de los componentes software, sus interfaces y las dependencias entre ellos.

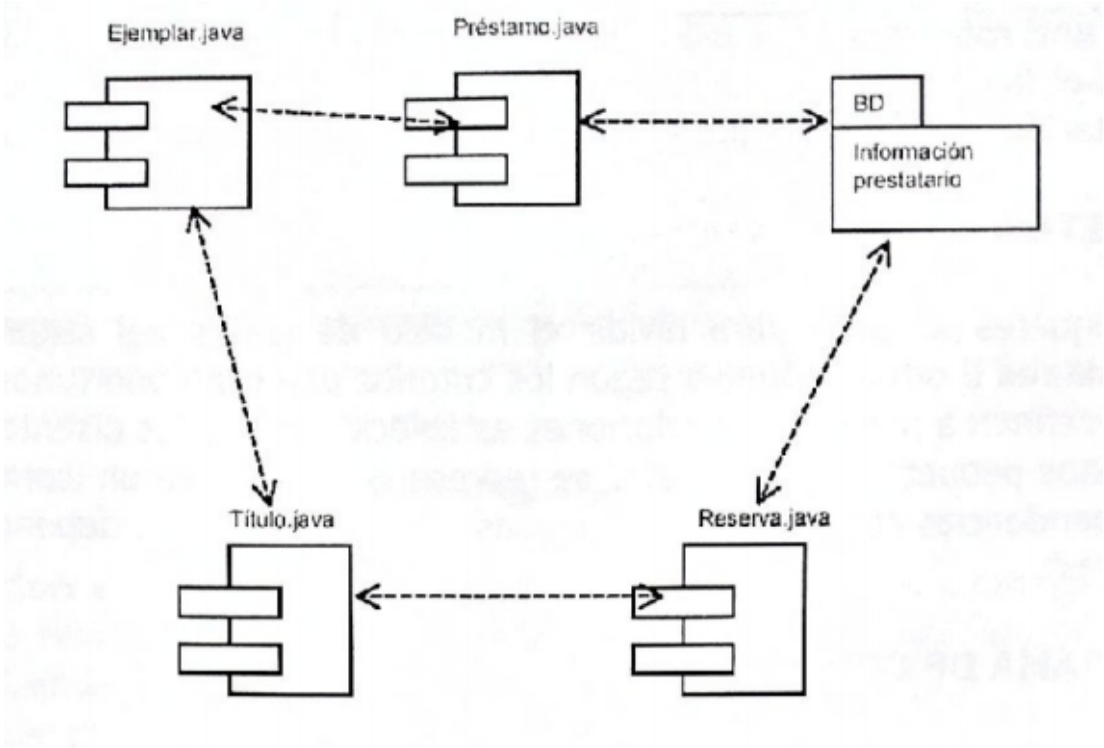
Un **componente** es un módulo de software que puede ser código fuente, código binario, un ejecutable, o una librería con una interfaz definida.

Estos diagramas pueden incluir paquetes que permiten organizar la construcción del sistema de información en subsistemas.



Componente. *Interfaz* *Paquete*

Ejemplo: Sistema encargado de la gestión de los préstamos y reservas de libros y revistas de una biblioteca. El lenguaje JAVA, y los accesos a la información del prestatario serán mediante un paquete de Base de Datos.



3.4. DIAGRAMA DE INTERACCIÓN.

Describe el comportamiento dinámico del sistema de información mediante el paso de mensajes entre los objetos del mismo. Describe en detalle un determinado escenario de un caso de uso: muestra la interacción entre el conjunto de objetos que cooperan en la realización de dicho escenario.

Elementos de un diagrama de interacción:

- Un **objeto** es una entidad que tiene un estado, un comportamiento e identidad.
- Un **mensaje** es una comunicación entre dos objetos.

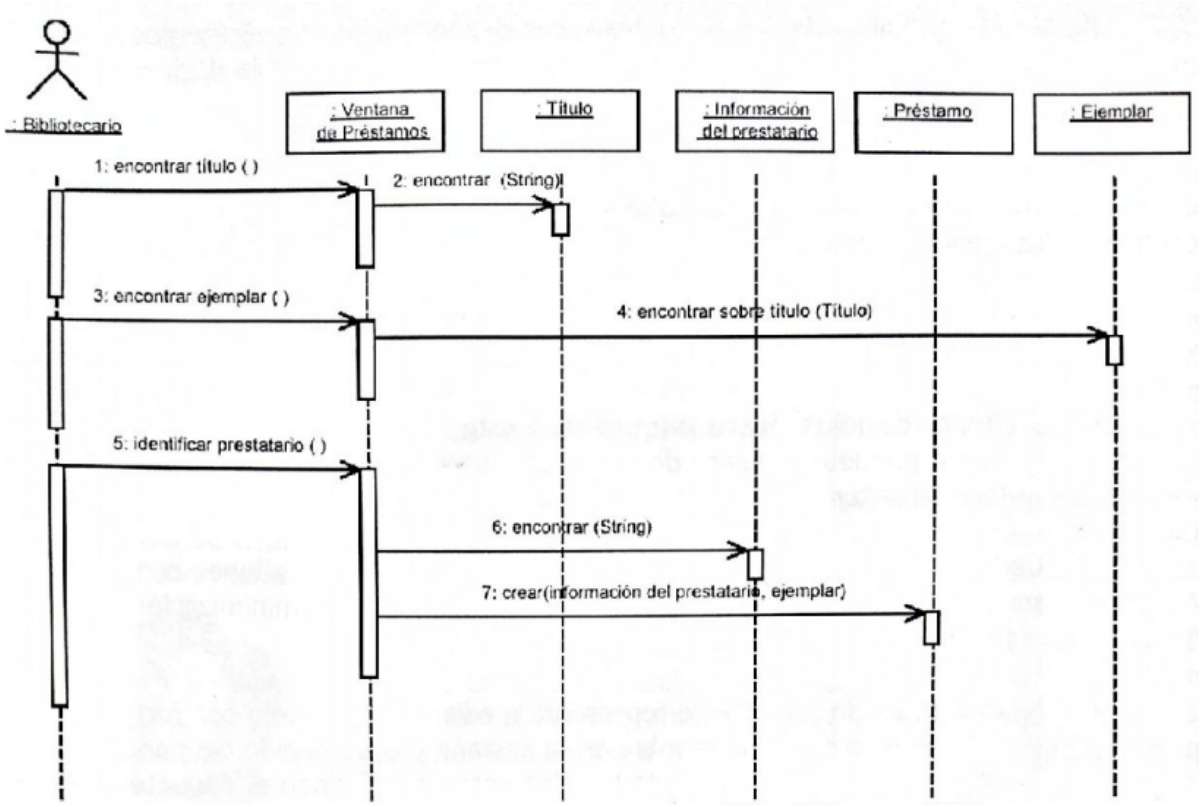
El envío de un mensaje por parte de un objeto (emisor) a otro (receptor), puede provocar que se ejecute una operación, se produzca un evento o se cree o destruya un objeto.

Hay dos tipos de diagramas de interacción:

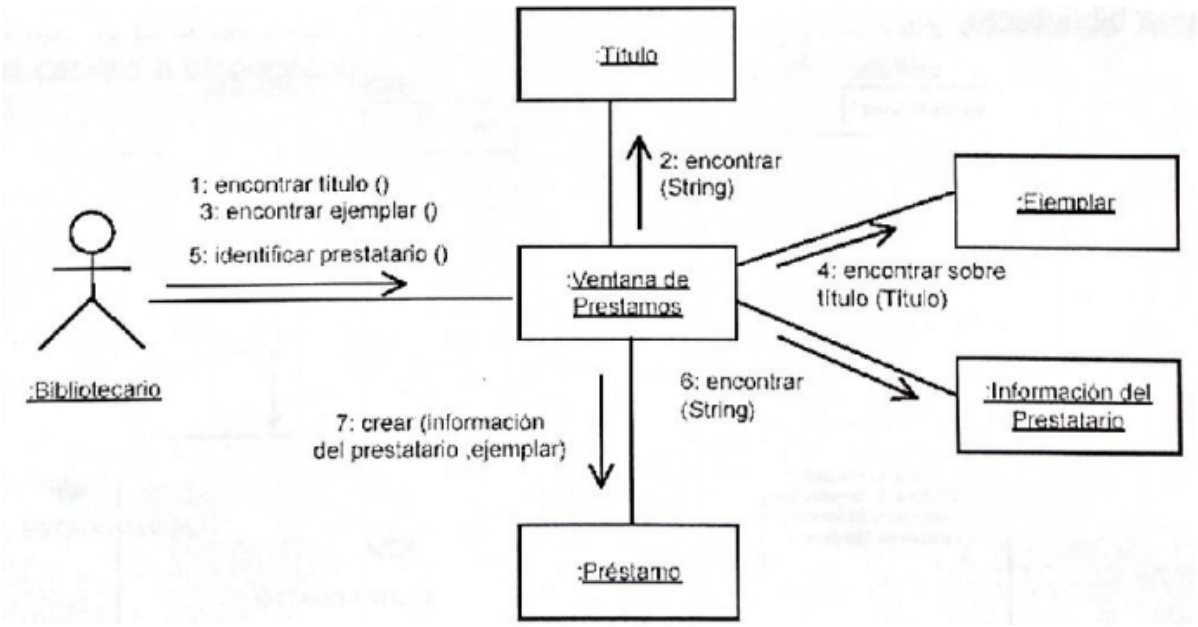
Diagramas de secuencia. Muestran de forma explícita la secuencia de los mensajes intercambiados por los objetos.

Diagramas de colaboración. Muestran con que otros objetos tiene vínculos o intercambia mensajes un determinado objeto.

Ejemplo: Diagrama de secuencia para el caso de uso: Prestar un ejemplar de una aplicación encargada de los préstamos y reservas de una biblioteca:



Ejemplo: Diagrama de colaboración para el caso de uso: Prestar un ejemplar de una aplicación encargada de los préstamos y reservas de una biblioteca:



3.5. DIAGRAMA DE PAQUETES.

Obtiene una visión más clara del sistema de información orientado a objetos, organizándolo en subsistemas, agrupando los elementos del análisis, diseño o construcción y detallando las relaciones de dependencia entre ellos. El mecanismo de agrupación se denomina Paquete.

En MÈTRICA Version 3, el diagrama de paquetes se aplica en el análisis para la agrupación de casos de uso, en el diseño de la arquitectura para la agrupación de clases de diseño y en el diseño detallado para agrupar componentes.

Estos diagramas contienen dos tipos de elementos:

Paquetes: Un paquete es una agrupación de elementos, bien sea casos de uso, clases o componentes. Los paquetes pueden contener a su vez otros paquetes anidados que en última instancia contendrán alguno de los elementos anteriores.

Dependencias entre paquetes: Cuando un elemento de un paquete requiere de otro que pertenece a un paquete distinto.

3.6. DIAGRAMA DE TRANSICIÓN DE ESTADOS.

Un diagrama de transición de estados muestra el comportamiento dependiente del tiempo de un sistema de información.

Los dos elementos principales en estos diagramas son:

El estado de un componente. Representa algún comportamiento que es observable externamente y que perdura durante un periodo de tiempo finito.

Una transición es un cambio de estado producido por un evento y refleja los posibles caminos para llegar a un estado final desde un estado inicial.

Los diagramas de transición de estados comprenden otros dos elementos:

Acciónes. Operaciones instantáneas asociadas a un evento.

Actividades. Operaciones asociadas a un estado que se ejecutan durante un intervalo de tiempo hasta que se produce el cambio a otro estado.