4.2.Ingeniería del Software / Modelos

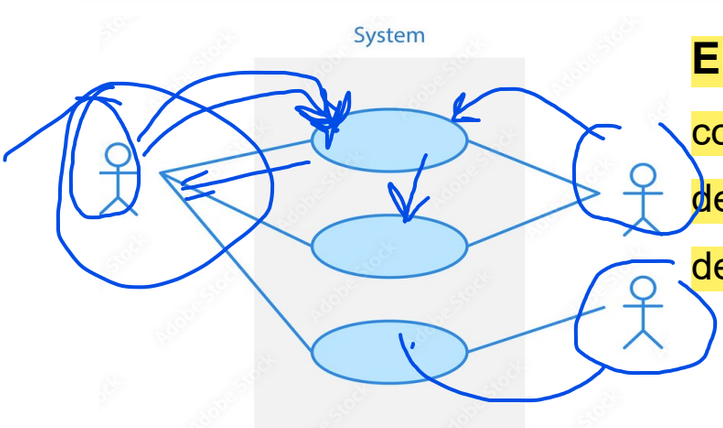
Modelos

Los modelos nos permiten formular ideas complejas en forma resumida y precisa. En los proyectos que involucran a muchos participantes, a menudo con diferentes conocimientos técnicos y culturales, la precisión y claridad son críticas cuando se incrementa rápidamente el costo de la falta de comunicación. Para que una notación permita la comunicación precisa debe tener una semántica bien definida, debe ser muy adecuada para la representación de un aspecto dado de un sistema y debe ser bien comprendida por los participantes del proyecto. En esto último se encuentra la fortaleza de los estándares y las convenciones: cuando una notación es utilizada por gran cantidad de participantes, hay poco espacio para las malas interpretaciones y la ambigüedad. Por el contrario. cuando existen muchos dialectos de una notación, o cuando se usa una notación muy especializada. los usuarios de la notación están propensos a malas interpretaciones cada vez que cada usuario impone su propia interpretación.

El desarrollo de software se enfoca en tres modelos diferentes: ***el modelo funcional, el modelo de objetos y el modelo de comportamiento.***

4.2.1. Ingeniería del Software / Modelo Funcional

Modelo Funcional

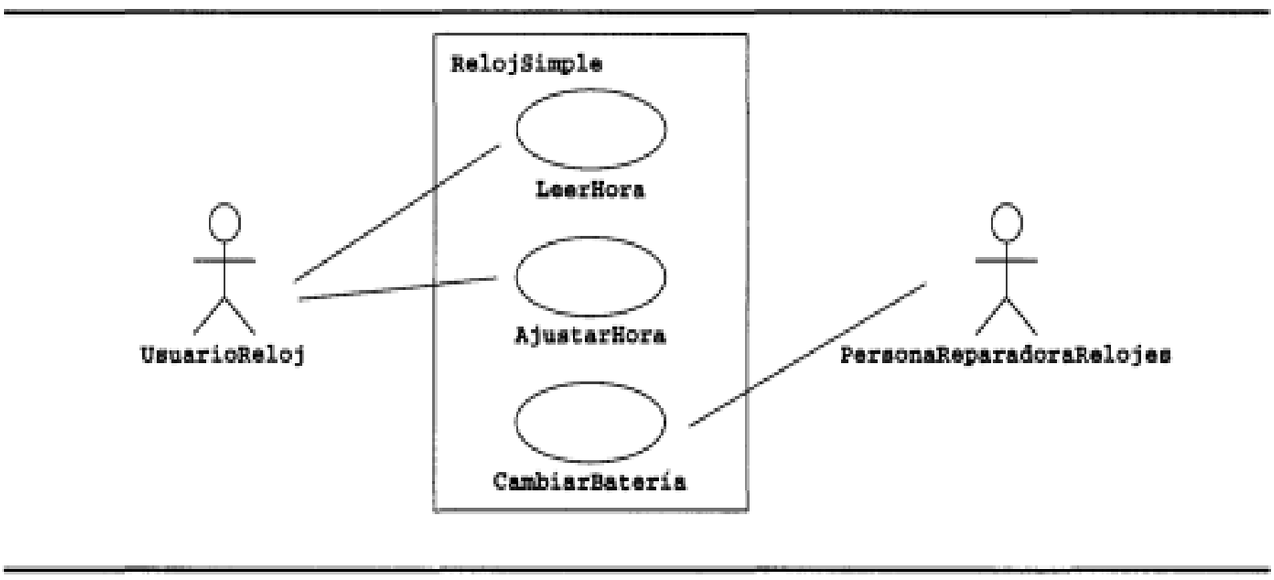


El modelo funcional, representado con ***diagramas de caso de uso***, describe la funcionalidad del sistema desde el punto de vista del usuario.

Los casos de uso se utilizan durante la obtención de requerimientos y el análisis para representar la funcionalidad del sistema. Los casos de uso se enfocan en el comportamiento del sistema desde un punto de vista externo. Un diagrama de caso de uso describe una función proporcionada por el sistema que produce un resultado visible para un actor. Un actor describe cualquier entidad que interactúa con el sistema (por ejemplo, un usuario, otro sistema, el ambiente físico del sistema). La identificación de los actores y los casos de uso da como resultado la definición de la frontera del sistema, esto es, diferencia entre las tareas realizadas por el sistema y las realizadas por su ambiente. Los actores están fuera de la frontera del sistema, mientras que los casos de uso están dentro de la frontera del sistema.

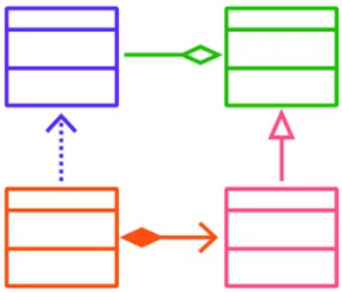
Ejemplo: ***un diagrama de caso de uso para un reloj simple.***

El actor UsuarioReloj puede consultar la hora en su reloj (con el caso de uso LeerHora) o ajustar la hora (con el caso de uso AjustarHora). Sin embargo, solo el actor PersonaReparadoraRelojes puede cambiar la bateria del reloj (con el caso de uso CambiarBateria).



4.2.2. Ingeniería del Software / El Modelo de Objetos

El Modelo de Objetos

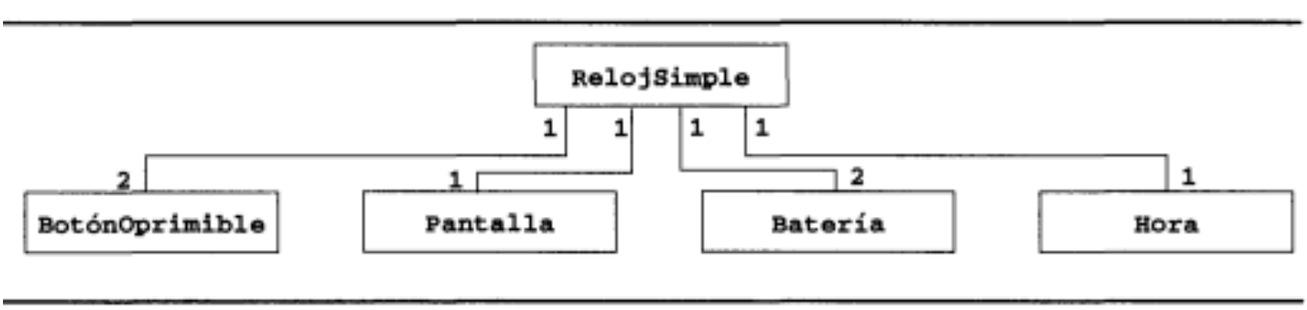


El modelo de objetos, representado con ***diagramas de clase***, describe la estructura de un sistema desde el punto de vista de objetos, atributos, asociaciones y operaciones.

Usamos diagramas de clase para describir la estructura del sistema. Las clases son abstracciones que especifican la estructura y el comportamiento comün de un conjunto de objetos. Los objetos son instancias de las clases que se crean, modifican y destruyen durante la ejecución del sistema. Los objetos tienen estados que incluyen los valores de sus atributos y sus relaciones con otros objetos. Los diagramas de clase describen el sistema desde el punto de vista de objetos, clases, atributos, operaciones y sus asociaciones.

Ejemplo: ***un diagrama de clase que describe los elementos de todos los relojes de la clase Reloj***

Todos estos objetos de reloj tienen una asociación con un objeto de la clase BotónOprimible, un objeto de la clase Pantalla, un objeto de la clase Hora y un objeto de la clase Batería. Los números que están al final de las asociaciones indican la cantidad de vínculos que puede tener cada objeto RelojSimple con un objeto de una clase dada.



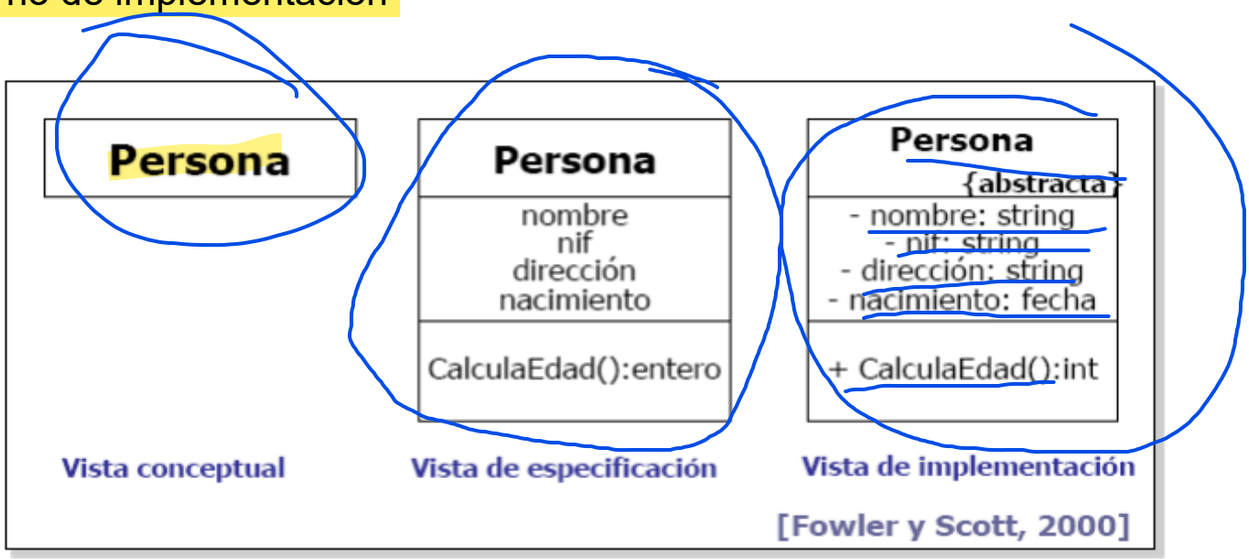
Un diagrama de clase es una representación gráfica (grafo bidimensional) de un ***modelo estructural*** que tiene un nombre. Es el diagrama principal para el análisis y diseño.

Niveles de representación de los diagramas de clases

– **Conceptual**. El diagrama de clase representa los conceptos en el dominio del problema que se está estudiando

– **Especificación.** El diagrama de clase refleja las interfaces de las clases, pero no su implementación. Aquí las clases aparecen más cercanas a los tipos de datos, ya que un tipo representa una interfaz que puede tener muchas implementaciones diferentes

– **Implementación.** Este nivel representa las clases tal cual aparecen en el entorno de implementación



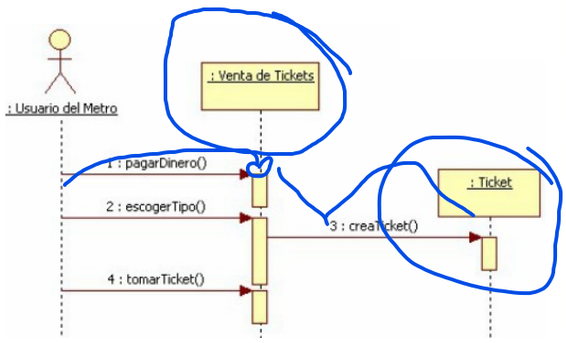
Usos en las diferentes fases del proceso software

– Modelo Conceptual, Se usan para realizar la abstracción de un dominio. Se representan conceptos del dominio del problema, incluyen atributos esenciales, restricciones y relaciones entre conceptos.

- ***Modelo del Análisis***. Los conceptos se convierten en clases. Los atributos de un concepto en atributos de la clase. Añade comportamiento (métodos).
- ***Modelo de Diseño***. Clases que corresponden a decisiones del diseño
- ***Modelo de Implementación***. Clases que corresponden a un lenguaje de programación.

4.2.3. Ingeniería del Software / El Modelo Dinámico

El Modelo de Dinámico



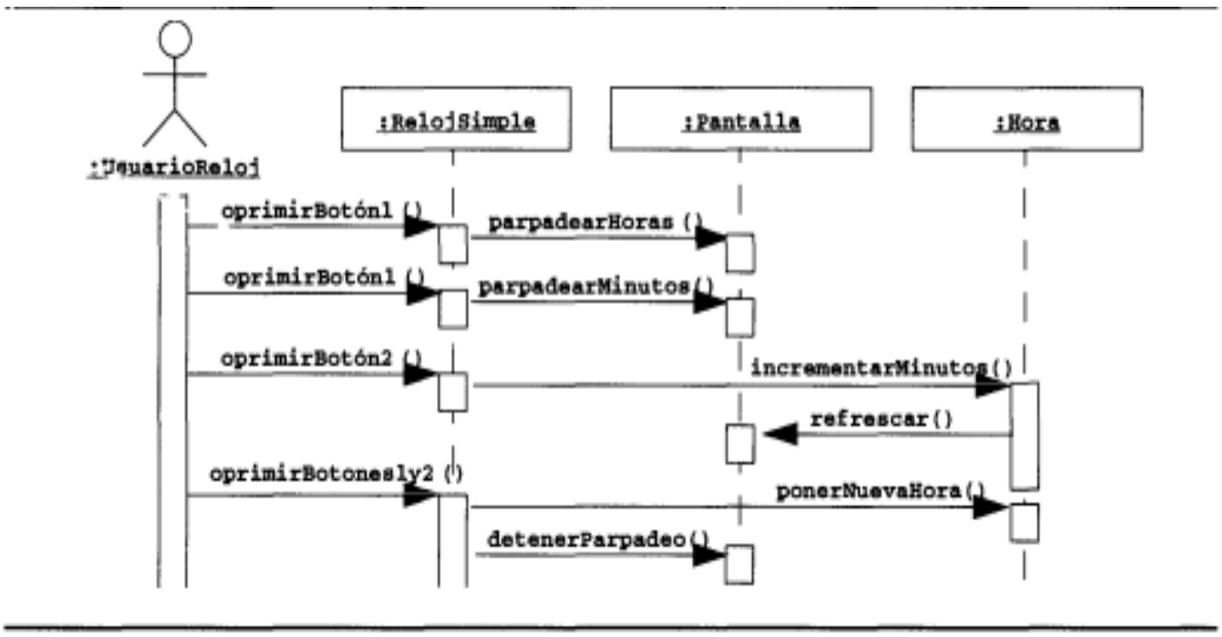
El modelo dinámico, representado en UML con ***diagramas de secuencia***, ***diagramas de gráfica de estado*** y ***diagramas de actividad***, describe el comportamiento interno del sistema.

Los diagramas de secuencia describen el comportamiento como una secuencia de mensajes intercambiados entre un conjunto de objetos, mientras que los diagramas de grafica de estado describen el comportamiento desde el punto de vista de estados de un objeto individua! y las transiciones posibles entre estados.

Los diagramas de secuencia se usan para formalizar el comportamiento del sistema y para visualizar la comunicación entre objetos. Son útiles para la identificación de objetos adicionales que participan en los casos de uso. A los objetos involucrados en un caso de uso se llama objetos participantes. Un diagrama de secuencia representa las interacciones que suceden entre esos objetos.

Ejemplo: ***un diagrama de secuencia para el caso de uso AjustarHora*** del reloj simple.

La columna de la extrema izquierda representa al actor UsuarioReloj que inicia el caso de uso. Las flechas etiquetadas representan los estimulos que un actor u objeto envla a otros objetos. En este caso, el UsuarioReloj oprime el botón 1 dos veces y el botón 2 una vez para adelantar el reloj un minuto. El caso de uso AjustarHora termina cuando el UsuarioReloj oprime ambos botones simultáneamente



4.2.4. Ingeniería del Software / Modelado

Modelado

El modelado es la actividad que se realiza cuando se diseña un sistema. El propósito del modelado es construir una abstracción del sistema que deje a un lado los detalles irrelevantes. Hay que abstraer los conceptos del dominio de aplicación (es decir, el ambiente en el que está operando el sistema) y del dominio de solución (es decir, las tecnologías para construir un sistema). El modelo resultante es más simple que el ambiente o el sistema y, por lo tanto, es más fácil de manejar.

Los diagramas de caso de uso representan la funcionalidad del sistema desde el punto de vista del usuario. Definen las fronteras del sistema.

Los diagramas de clase se usan para representar la estructura de un sistema en términos de objetos con sus atributos y relaciones.

Los diagramas de secuencia representan el comportamiento del sistema en términos de interacciones entre un conjunto de objetos. Se usan para identificar objetos en los dominios de aplicación e implementación.

4.2.5. Ingeniería del Software / Modelado / Ejemplos

Ejemplos Operaciones

Ejemplo 1: Dar de alta un nuevo cliente

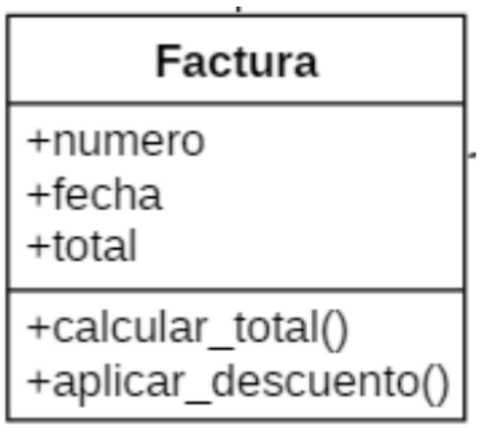
Esta funcionalidad pretende llevar un control sobre el histórico de clientes que ha tenido la empresa. Por cada cliente, la secretaria de la empresa debe registrar los siguientes datos: DNI, Nombre, Apellidos, Dirección, Código postal, Ciudad, Teléfono. Los clientes se deben poder mostrar en una lista. NO aporta información sobre operaciones

Ejemplo 2 Dar de alta una nueva ficha de reparación

Cuando el técnico regresa de hacer una reparación, la secretaria debe crear una ficha de reparación con los siguientes datos: el cliente, la fecha de la reparación, una descripción, el importe al que asciende la reparación y el nombre del responsable técnico que hizo la reparación. Al crear la ficha de reparación, ésta se considera impagada, ya que el cliente no paga al técnico. El proceso de pago se hace posteriormente. La secretaria debe poder listar todas las fichas de reparación existentes en el sistema. Así puede ver las fichas que están pagadas y las que faltan por pagar.

Ejemplo 3 Crear factura con todas las fichas de reparación impagadas

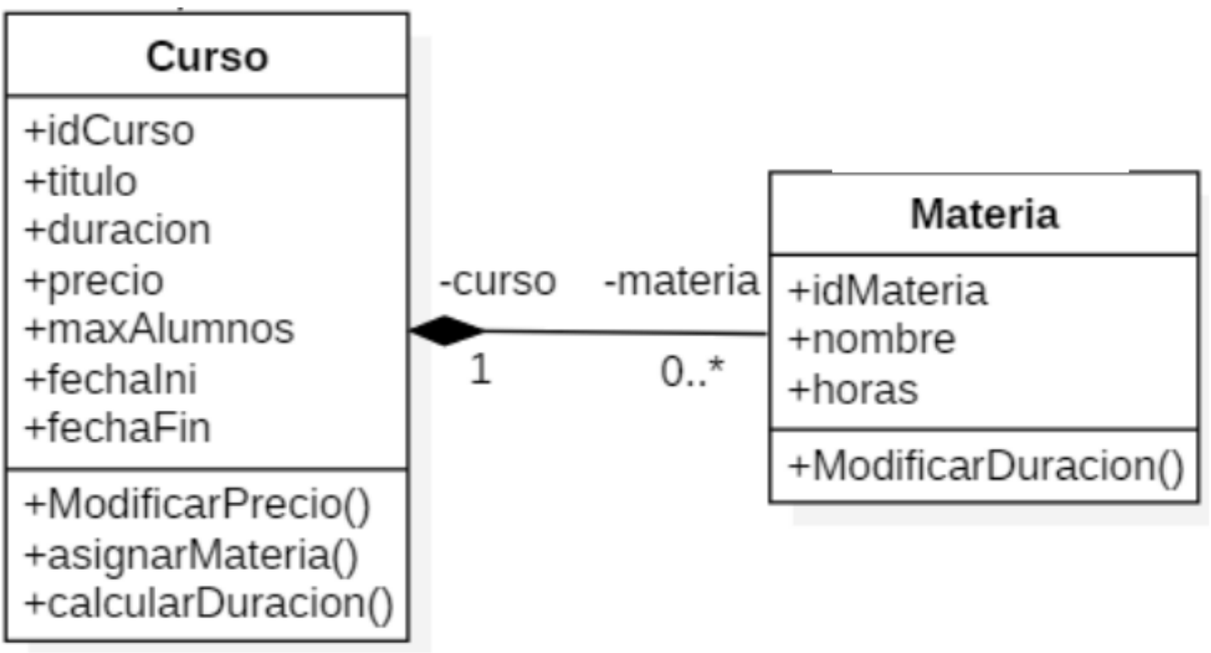
La empresa *Reparos SL* suele trabajar con hoteles, de forma que suele hacer más de una reparación por cliente en un plazo corto. suelen pagar todas las reparaciones pendientes de una vez. Cuando Estos clientes un cliente va a las oficinas a pagar, la secretaria crea una factura con todas las reparaciones que tiene pendientes hasta ese momento. Al crear la factura, se dan por pagadas todas las reparaciones que ese cliente tenía pendientes. La secretaria debe introducir en el sistema la fecha de creación de la factura y el cliente sobre el que se hará la factura. El total a pagar en la factura es la suma de todos los importes acumulados en las fichas de reparación. Este dato se debe calcular de forma automática, sin que la secretaria tenga que hacer la suma. La secretaria debe poder listar todas las facturas existentes, mostrando el total pagado.



Ejemplo 4 Gestionar cursos

La secretaria de la empresa es la encargada de gestionar (altas, bajas, modificaciones y listados) los cursos y las materias a impartir en cada uno de estos cursos. Un ejemplo de curso podría ser por ejemplo, sobre mantenimiento de lavadoras, donde se imparten las asignaturas de descalcificación y electrónica. Para cada curso, el sistema debe almacenar: el nombre del curso, la duración (en horas), el precio del curso, el número máximo de clientes que se pueden inscribir en el curso, la fecha de inicio y

la fecha de fin. Una vez creado el curso, la secretaria le debe asignar las materias que componen el curso, ya que un curso puede estar compuesto de más de una materia. Para cada materia nos interesa almacenar: el nombre y las horas de la materia. De esta forma, la duración del curso se calcula automáticamente como la suma de las horas de sus materias.



Ejemplo 5 Gestionar profesores

La secretaria de la empresa se encarga de buscar profesores que impartan las materias de los cursos. Para cada profesor, el sistema debe almacenar: nombre, apellidos, dirección, teléfono y sueldo. La secretaria puede crear, modificar, eliminar y listar los profesores que trabajan en los cursos. Una materia sólo la puede impartir un único profesor. NO aporta información sobre operaciones.

Referencias

- Universidad Politécnica de Valencia. Diagramas en Ingeniería del Software.
- Fowler y Scott, 2000. UML Gota a Gota.
- Bruegge, Dutoit. 2002. Ingeniería de Software Orientada a Objetos.