Unidad 6. Establecimiento de las claves abstractas

Objetivo

Conocer el proceso y técnicas para identificar las claves abstractas en el proceso de análisis de un sistema.

Cuando analizamos sistemas, creamos modelos del área de aplicación que nos interesa. Un modelo puede incorporar un sistema, centrarse en un área concreta de la empresa o abarcar toda la empresa. El modelo de empresas es importante para la planificación de la automatización de las mismas. Dicho modelo, además, representa un aspecto de la realidad y se construye de modo que nos ayude a comprenderla. También, resulta mucho más sencillo que la realidad misma, al igual que un coche o una casa a escala es mucho más simple que un coche o una casa de verdad. Al utilizar modelo podremos idear sistemas o rediseñar áreas de una empresa. Con el análisis orientado a objetos, la forma de modelar la realidad difiere del análisis convencional. Modelamos el mundo en términos de tipos de objetos y lo que le ocurre a éstos. Esto implica también diseñar y construir sistemas de forma orientada a objetos. El modelo debe representar la forma en que los usuarios finales perciben el área de dominio en cuestión, aquella en la que se encuentran involucrados de alguna manera: laboral, ocasional, etc. En la medida de lo posible, el modelo debe ser presentado de forma que sea comprensible para los usuarios finales. Los modelos construidos en el análisis orientado a objetos reflejan ese mundo real de forma más natural que en el análisis tradicional de sistemas, ya que la realidad está formada por objetos y por eventos, es decir, por situaciones que cambian el estado de los susodichos objetos.

Así, cuando el mundo real cambie, nuestro software será más fácil de cambiar, lo que es una gran ventaja.

Siguiendo con el razonamiento, la programación orientada a objetos se basa en que una situación puede ser modelada y los problemas que se presenten en su interior resueltos, mediante la utilización de objetos de clases existentes o que deben ser creadas. Esta unidad estará dedicada a la identificación objetos, cuyo comportamiento, creación y relaciones se definirá a través de clases que se aprenderán a definir en él.

Al hacer referencia a un objeto, se habla de una unidad que tiene identidad propia, ciertas características y, además, capacidad de prestar servicios.

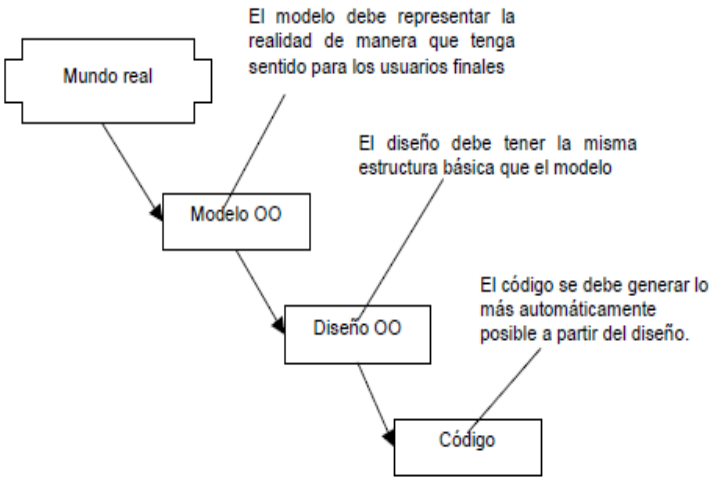
Modelar software a partir del mundo real.

Cuando se analizan sistemas se crean modelos del área de la aplicación en cuestión. El modelo representa un aspecto del mundo real y se construye de modo que ayude a comprenderlo. Por tal razón, debe ser mucho más sencillo que la realidad, una mera abstracción de ésta, y, además, el modelo permite recrear muchas situaciones imposibles de hacer en el mundo real.

En el análisis orientado a objetos se modela desde el mundo real en términos de tipos de objetos y lo que les ocurre a éstos

En la ilustración siguiente podemos ver de manera gráfica la forma de construir sistemas:

El análisis crea un modelo en el dominio de la aplicación.
El modelo se convierte en un diseño
El diseño se convierte en código.

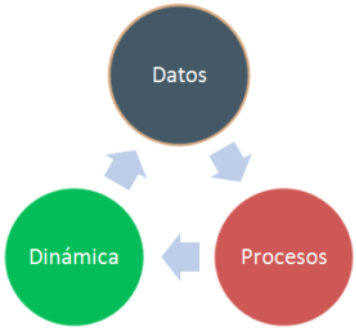


En el proceso de análisis se captura el punto de vista de los usuarios con respecto al área de dominio del mundo real y posteriormente se traduce a software de la manera más automáticamente posible, por lo tanto, al cambiar las necesidades de los usuarios, el software cambiará con ellas.

El análisis orientado a objetos es análisis, pero también contiene una cierta cantidad de síntesis. La abstracción de requisitos del usuario y la identificación de los objetos clave del área de dominio van seguidas por el ensamblaje de estos objetos en estructuras, de una forma que admita el diseño físico en alguna fase posterior. El aspecto de síntesis aparece precisamente porque se está utilizando un sistema. Se estás sistematizando la información lo que, en otras palabras, acaba imponiendo una estructura al área de dominio.

Los sistemas pueden ser descritos en tres dimensiones:

*a) Los **datos**, objetos o conceptos y su estructura.*
*b) Arquitectura o **proceso** no temporal.*
*c) La dinámica, comportamiento del sistema o el **control** del sistema.*



Para validar la congruencia de estos aspectos, se ha de realizar una comprobación cruzada, de tal manera que se identifiquen qué procesos usan determinados datos y cómo estos están siendo utilizados. Aunque estas verificaciones incrementan los costes en términos de documentación, sin embargo, se asegura que se tratan todos los aspectos del sistema, siempre y cuando no haya sido deficiente la extracción de los mismos.

También se tiene la ventaja de que se utilizan dos técnicas para llegar a la misma conclusión; por lo tanto, si concuerdan los resultados, el grado de confianza será mayor.

La orientación a objetos, combina esos dos aspectos - los datos y los procesos-, de tal modo que encapsulando los datos, también resulte posible encapsular su control. Por consiguiente, un análisis orientado a objetos se puede considerar como una manera de trasladar de silogismos racionales o deducciones que partan de lo particular (las clases), atraviesen lo individual (las instancias) y lleguen a lo universal (el control u objeto).

La solución de un problema comprende su comprensión y conceptualización, así como la ejecución e implementación de esa misma solución. Conceptualizar un problema involucra su representación usando construcciones representativas (nociones mentales o ideas). De hecho, no podemos afrontar un problema directamente, desmenuzándolo en partes y manejando las construcciones representativas del dominio del problema. Igualmente, desmenuzado debe estar el dominio de la solución para abordar su ejecución e implementación en una serie de pasos. La realización de la solución involucra el mapeo de las construcciones representativas. El uso de las construcciones representativas es un proceso muy natural que a menudo ocurre sutilmente y a veces inconscientemente. Subyacente a este esquema está el uso de un paradigma en la determinación de los posibles tipos de representaciones usadas en los esfuerzos de resolución del problema. El paradigma orientado al objeto deriva de la convergencia de otros paradigmas fundamentales, y se reduce a los otros paradigmas tanto como sea requerido por su aplicación a través de un lenguaje o método. ¿Cuáles son los elementos involucrados? Podemos dividirlos en:

El Mundo Real Es el dominio que abarca problemas, soluciones y esfuerzos para resolver los problemas.

Este mundo real incluye: Cosas o entidades que tienen un propósito o rol dentro del mundo. Las entidades tienen características estructurales que representan lo que las cosas "saben" y características de comportamiento que representan lo que las cosas "hacen". Una entidad agregada puede contener a su vez otras entidades componentes que serán características estructurales y de comportamiento. Por ejemplo, un coche puede estar compuesto de diferentes piezas como ruedas, puerta, pedales, volantes… De este modo, a la entidad principal le quedan agregadas otras entidades que aportarán características a aquella e incluso determinarán su comportamiento.

Relaciones entre entidades. Las relaciones pueden ser tratadas como características estructurales compartidas entre múltiples entidades que tienen una relación con alguna otra. Una entidad agregada puede contener entidades componentes que están relacionadas vía un atributo de la entidad agregada.

Concurrencias entre entidades. Las concurrencias pueden ser tratadas como características de comportamiento compartidas entre múltiples entidades que interactúan con alguna otra. Una entidad agregada puede contener entidades componentes que son manipuladas vía una operación de la entidad agregada. Estos conceptos son fundamentales para la interacción con el mundo real.

Determinar las abstracciones clave del modelo.

Demos un paso más en la aclaración de conceptos y hablemos de las abstracciones. Una abstracción es una clase u objeto que forma parte del vocabulario del dominio de un problema. De esta forma, podríamos decir que la abstracción es un objeto que tiene responsabilidades y también hay otros objetos que pueden usarla. Estos se conocen como colaboradores.

¿Cómo podemos identificar en el mundo real estás abstracciones y hacerlas candidatas a convertirse en claves del modelo? Podemos recomendar algunas pautas generales para su localización, pero siempre teniendo en cuenta de que no existen fórmulas mágicas y cada caso concreto es particular. Aun así, podemos detectar claves abstractas en:

Todos los nombres y sustantivos en los requerimientos del proyecto son muy útiles para obtener el dominio del problema.
Los casos de uso y sus escenarios porque contienen detalles y responsabilidades del dominio del problema. Normalmente los nombres de los casos de uso se escriben usando un verbo y un nombre: Crear Reserva, el nombre puede ser un candidato.
Los "expertos de dominio", usuarios que están directamente involucrados en el problema a solucionar pueden ayudar a diferenciar objetos, atributos u operaciones.

Una vez establecido el corpus de abstracciones clave, debemos seleccionar las que resultan pertinentes. Para ello, una vez que se tiene una lista se eliminarán aquellas que sean caracterizaciones (atributos) de algo, por ejemplo calle sería un atributo dirección. Es decir,

tendremos que discernir aquellos elementos que son partes de aquellos otros que son el todo al que están vinculados.

También deberán sustituirse las que se refieran a algo particular y no a algo general, por ejemplo en lugar de apellido, seleccionaremos cliente.

Igualmente, pueden encontrarse sinónimos y seleccionar sólo uno de los términos.

Como vemos, las abstracciones seleccionadas aparecerán ante nuestros ojos como cúmulos de elementos que se suelen denominar responsabilidades y colaboradores. Vienen a identificar en la narrativa de los casos de uso y en sus escenarios.

Las responsabilidades son cualquier atributo, operación ó especificación de los valores del rango de datos para los atributos.

Los colaboradores son otros objetos, normalmente otra abstracción, con la que aquella estaría asociada. Es decir, que las abstracciones pueden colaborar entre sí y subordinarse en determinados casos.

Una cuestión importante es que si no detectamos responsabilidades dentro de una abstracción, deberemos rechazar esta última. Del mismo modo, si se encuentra que una responsabilidad (un atributo) es otra abstracción, también habrá que eliminar esta última candidata.

¿Cómo detectar estas partes? Bueno, si típicamente los sustantivos de los casos de uso son una buena pista para encontrar candidatos a abstracciones clave, los verbos de los casos de uso son candidatos a sus responsabilidades correspondientes.

Por ejemplo, en un sistema de reserva de habitaciones de un hotel:

Un empleado debe crear, recuperar, actualizar y borrar una reserva. Por lo tanto, una reserva tiene una fecha de llegada, de salida y un ID. Estos últimos tres datos son las responsabilidades en forma de atributos de la abstracción clave "reserva".
Una reserva está asociada con un solo cliente. En este caso hay una colaboración entre la abstracción clave Reserva y la abstracción clave Cliente.

Las tarjetas CRC

Una vez que se identificaron las abstracciones clave se creará una tarjeta CRC.

A fines de la década de 1980, uno de los centros más grandes de tecnología de objetos era el laboratorio de investigación de Tektronix, en Portland, Oregon, Estados Unidos. Este laboratorio tenía algunos de los principales usuarios de Smalltalk y muchas de las ideas clave de la tecnología de objetos se desarrollaron allí. Dos de sus programadores renombrados de Smalltalk eran Ward Cunningham y Kent Beck.

Tanto Cunningham como Beck estaban y siguen preocupados por cómo enseñar los profundos conocimientos de Smalltalk que habían logrado. De esta pregunta sobre cómo enseñar objetos surgió la sencilla técnica de las tarjetas de Clase-Responsabilidad-Colaboración (CRC).

En lugar de utilizar diagramas para desarrollar modelos, como lo hacían la mayoría de los metodólogos, Cunningham y Beck representaron las clases en tarjetas 4 x 6 [pulgadas]. Y en lugar de indicar atributos y métodos en las tarjetas, escribieron responsabilidades.

Ahora bien, ¿qué es una responsabilidad? En realidad es una descripción de alto nivel del propósito de una clase. La idea es tratar de eliminar la descripción de pedazos de datos y procesos y, en cambio, captar el propósito de la clase en unas cuantas frases. El que se haya seleccionado una tarjeta es deliberado. No se permite escribir más de lo que cabe en una tarjeta

En consecuencia, un modelo CRC es un conjunto de tarjetas índice estándar que representan las abstracciones clave antes seleccionadas. En este punto, las responsabilidades son los atributos y operaciones relevantes para la abstracción clave. Mientras que los colaboradores son las abstracciones subsidiarias necesarias para proveer a una abstracción clave de la información necesaria con la cual pueda completar una responsabilidad determinada.

Las colaboraciones identifican relaciones entre abstracciones clave. Cuando un conjunto de abstracciones clave colabora para satisfacer algún requisito es posible organizarlas en un subsistema (elemento de diseño).

¿Cómo organizamos toda esta información dentro de cada tarjeta? Del modo en el que queda reflejado en la siguiente ilustración:

Nombre de la Abstracción (Clase)	
Responsabilidades (operaciones y atributos)	**Colaboradores (relaciones)**
Mantener un estado Llevar a cabo alguna tarea	Elementos con los que va a interactuar la abstracción

Podemos ver a continuación un ejemplo de aplicación de la tarjeta CRC para el caso de uso de una reserva de habitación:

Reserva	
Responsabilidades (operaciones y atributos)	**Colaboradores (relaciones)**
Reservar una habitación Status ("nueva", "confirmada") Fecha de llegada Fecha de salida Forma pago Número reserva	Habitación Cliente

Las colaboraciones se identifican determinando si una clase puede satisfacer cada responsabilidad.

Para identificar colaboradores, se pueden examinar tres relaciones genéricas diferentes entre clases:

- es-parte-de
- tiene-conocimiento-sobre
- depende-de

El nombre de la clase colaboradora se registra en la tarjeta índice del modelo CRC al lado de la responsabilidad que ha generado dicha colaboración. Veamos un ejemplo:

Nombre de la Clase: Pago	
Responsabilidades: Guardar información de cada transacción de pago realizada. Guardar el lugar donde se realiza el pago y el tipo (efectivo o tarjeta de crédito).	Colaboradores: Contrato Banco Secretaria
Atributos: idPago fecha tipoPago efectivoTarjeta idCajero idContrato	

Aquí tenemos el pago a una entidad bancaria. Las abstracciones clave sería pago con sus correspondientes responsabilidades y atributos. A su vez, colaboran en el pago el contrato, el banco y secretaría.

También, se deriva una estructura de generalización-especialización para las clases identificadas. Un objeto puede estar compuesto de un número de partes las cuales pueden definirse a su vez como objetos. Estos objetos agregados pueden representarse como estructura todo-partes.

Las representaciones estructurales proveen al analista de los medios para fraccionar el modelo CRC y para representar esta partición gráficamente.

Los subconjuntos de clases que colaboran entre sí para llevar a cabo un conjunto de responsabilidades cohesionadas, se les llama temas o subsistemas. Un subsistema se puede tratar como una caja negra que contiene un conjunto de responsabilidades y que posee sus propios colaboradores (externos).

También, un subsistema implementa uno o más contratos con sus colaboradores externos. Un contrato es una lista específica de solicitudes que los colaboradores pueden hacer a un subsistema.

Los subsistemas pueden representarse dentro del contexto del modelo CRC creando una tarjeta índice del subsistema.

A partir de todo este proceso podemos implementar dos modelados:

Modelo objeto-relación. Se deriva en tres pasos o etapas:

- Usando las tarjetas índice CRC se dibuja una red de objetos colaboradores conectados por líneas sin etiquetar.
- Revisando el modelo CRC se evalúan responsabilidades y colaboradores y cada línea recibe un nombre. Una punta de flecha indica la dirección de la relación.
- Se evalúa cada extremo de la relación para determinar la cardinalidad (0:1; 1:1; 0:n; 1:n).

Modelo objeto-comportamiento. Indica cómo responderá un sistema orientado a objetos a eventos externos o estímulos. Se deriva en los pasos siguientes:

- Evaluar todos los casos de uso para comprender la secuencia de interacción dentro del sistema.
- Identificar eventos que dirigen la secuencia de interacción y comprender cómo estos eventos se relacionan con objetos específicos.
- Crear una traza de eventos para cada caso de uso.
- Construir un diagrama de transición de estados para el sistema.
- Revisar el modelo objeto-comportamiento para verificar exactitud y consistencia.

Ver Video: *Determinando las abstracciones claves,* en la Unidad 6, en el Módulo 3, en la plataforma elearning

Actividades

"Recuerde que para un seguimiento óptimo de la unidad es imprescindible realizar las actividades que encontrará en la unidad correspondiente de la plataforma eLearning".