2019

Herramientas de productividad HDP115

UNIDAD III: HERRAMIENTAS PARA ANÁLISIS Y DISEÑO

BLADIMIR DIAZ CAMPOS

UNIVERSIDAD DE EL SALVADOR | Facultad de Ingeniería y Arquitectura, Escuela de Sistemas Informáticos

Contenido

1. Programación orientada a objetos

El paradigma de programación vigente es el paradigma de desarrollo orientado a objetos, sucesor del paradigma de la Programación Estructurada (PE), que se implementaba a través de los llamados lenguajes procedimentales.

En los lenguajes procedimentales un programa es un conjunto de instrucciones o sentencias que cumplen un objetivo específico. Donde en general, tanto la estructura del software y el código son fácilmente comprensibles. Sin embargo, el problema más importante de la PE es la reutilización de código, que está ligada a la estructura del software que se está construyendo.

Como resultado, la cantidad de código (usualmente medida en líneas de código) es proporcional al tamaño y la complejidad del problema que se está resolviendo.

La POO persigue, además de brindar una visión más aproximada a la forma en cómo vemos los fenómenos en el mundo real, resolver el problema de la reutilización de código.

La POO es un paradigma de programación que se basa en modelar el mundo real como objetos, sus características y relaciones.

Podemos definir un paradigma como “la forma en la que son concebidas las cosas en un momento determinado”. Esta forma en la que comprendemos las cosas, está determinada generalmente por un conjunto de conceptos que son utilizados para representar los elementos más importantes del fenómeno que se estudia. Cuando este conjunto de conceptos es organizado de manera ordenada y consistente se le llama modelo.

Más formalmente, un modelo es una representación abstracta y conceptual (generalmente gráfica o visual) que trata de describir las partes esenciales de un fenómeno, compuesto de una notación (símbolos) y reglas.

Las reglas pueden ser semánticas, es decir; el significado de los símbolos, y sintácticas, es decir; cómo se utilizan los símbolos.

En resumen, un paradigma es un marco teórico que proporciona un modelo de representación de los fenómenos en un periodo determinado de tiempo. El paradigma POO define un modelo de representación de los elementos de un fenómeno, que requieren ser descritos, para posteriormente ser programados (codificados).

Actualmente, estos conceptos son generalmente representados a través del Lenguaje Unificado de Modelado o UML por sus siglas en inglés (Unified Modeling Language) que estudiaremos posteriormente.

Como se ha mencionado anteriormente, la POO persigue lograr el máximo de reutilización de código. Una de las características más importantes a través de la que se logra esta reutilización es el polimorfismo, que se logra a través de la herencia, una de las formas de implementar la abstracción.

1.1. La abstracción

La abstracción es el proceso de aislar los componentes fundamentales de un fenómeno para conservar los rasgos más relevantes. Aislar estos componentes requiere del reconocimiento de los elementos del fenómeno que se estudia, identificar los más importantes y las relaciones entre ellos.

Dependiendo del fenómeno, estas características y relaciones pueden resultar complejas de identificar. Desde el punto de vista intelectual, se puede describir como un proceso de inducción-deducción.

El método de razonamiento deductivo, es el proceso elemental del análisis, que consiste en obtener una conclusión a partir de una serie de premisas. Este modo de razonamiento nos lleva de lo general a lo particular, o de lo complejo a lo simple.

Por ejemplo, si sabemos que todos los mamíferos lactan (se alimentan de la leche materna). Además, sabemos que los humanos son mamíferos, podemos deducir que todos los humanos lactan.

Por otro lado, la forma inductiva es un modo de razonar que nos lleva de lo particular a lo general, o de una parte a un todo.

Por ejemplo, después de hacer una prueba de conocimientos y aptitudes a una muestra representativa de los estudiantes de 5° de la carrera de Ingeniería de Sistemas Informáticos, todos mostraron aptitudes y conocimiento para desarrollar software, de forma sobresaliente. Aunque no se les haya realizado la prueba a todos los estudiantes, podemos inferir que todos los estudiantes de la carrera, tienen aptitudes y conocimientos para desarrollar software. Es decir, podemos establecer una característica generalizada a un conjunto de objetos, partiendo de la información particular de cada algunos de ellos.

La abstracción en el ámbito de la POO consiste en definir los elementos fundamentales del fenómeno para el que se realiza el software, las características que definen a esos elementos y sus relaciones más relevantes.

1.2. Objeto

El término objeto es un concepto que resulta en sí mismo bastante abstracto, puede referirse a cualquier cosa hasta que hayamos definido concretamente de qué se trata. Son las características de ese objeto lo que nos ayudarán a darle un “nombre” o definirlo en una categoría de objetos.

1.3. Clase

Al acercarnos a ese objeto e identificar sus características, lo podemos definir en alguna categoría de objetos. Una clase por lo tanto es una definición de una categoría de objetos. Esta definición describe las características fundamentes de los objetos que pertenecen a dicha clase: atributos y operaciones.

De forma análoga que, en la PE, una clase describe las características de los objetos, al igual que los TAD (Tipos Abstractos de Datos). Sin embargo, a diferencia de los TAD, las clases describen también definen el comportamiento a través de las operaciones.

A partir de esta definición, los objetos son concebidos como instancias (casos concretos u ocurrencias) de una clase.

Supongamos que intentamos definir las clases para el conjunto de objetos que se muestran en la Figura 1

Figura 1: Objetos, medios de transporte

Nos interesa reconocer las características más relevantes de estos objetos para clasificarlos de acuerdo a ellas. Algunas de esas características se muestran en la Tabla 1

Objeto	**Atributos**	**Operaciones**
1	Desplazamiento: Autónomo por motor de combustión Medio: Terrestre Se mueve sobre: Ruedas (4) Mando: Volante	Acelerar Frenar Virar (cambiar de dirección)
2	Desplazamiento: Autónomo a través de turbinas Medio: Aéreo Se mueve sobre: N/A Mando: Volante	Acelerar Frenar Virar (cambiar de dirección)
3	Desplazamiento: Autónomo a través de turbinas / Hélices Medio: Acuático Se mueve sobre: Su obra viva Mando: Volante	Acelerar Frenar Virar (cambiar de dirección)
4	Desplazamiento: Por fuerza proporcionada a través de un sistema de piñones Medio: Terrestre Se mueve sobre: Ruedas (2, 3 o 4) Mando: Volante	Acelerar Frenar Virar (cambiar de dirección)
5	Desplazamiento: Autónomo por motor de combustión Medio: Terrestre Se mueve sobre: Ruedas (4) Mando: Volante	Acelerar Frenar Virar (cambiar de dirección)
6	Desplazamiento: Autónomo por motor de combustión Medio: Terrestre Se mueve sobre: Ruedas (2) Mando: Volante	Acelerar Frenar Virar (cambiar de dirección)
7	Desplazamiento: Por fuerza proporcionada a través de remos Medio: Acuático Se mueve sobre: Su obra viva Mando: Remos	Acelerar Frenar Virar (cambiar de dirección)
8	Desplazamiento: Autónomo por motor de combustión Medio: Terrestre Se mueve sobre: Ruedas (4) Mando: Volante	Acelerar Frenar Virar (cambiar de dirección)

Ya hemos definido una clase como una categoría de objetos que define las características que deben tener los objetos que pertenezca a dicha clase.

Podríamos crear una clase para cada uno de los objetos que hemos visto (8 en total). A priori podríamos pensar en 8 clases. Sin embargo, podemos ver que debido a que hemos identificado las características más relevantes de los objetos, la clase Carro define las características tanto del objeto #1 como del objeto #8.

El resultado serán 7 clases que definen los objetos que hemos visto: Carro, Avión, Barco, Bicicleta, Autobus, Motocicleta y Balsa.

Como ejemplo, veamos las definiciones de las clases Bicicleta y Balsa.

Clase: Bicicleta
Definición: Un vehículo terrestre que se mueve por la fuerza provista sobre sus ruedas a través de un sistema de piñones y pedales.
Instancias: Bicicleta

Clase: Balsa
Definición: Un vehículo acuático que se mueve por la fuerza provista a través de remos
Instancias: Balsa

Cualquier objeto que cumpla con las características definidas en cada una de las clases podrá ser reconocido como una instancia de una de ellas.

Si generalizamos más las características de estos objetos para crear clases con un nivel más alto de abstracción, podemos definir una nueva clase que podemos llamar Automóvil, que defina las características básicas que deben tener los vehículos que se mueven de forma autónoma.

Clase: Automóvil
Definición: Todos aquellos vehículos que se mueven de forma autónoma
Instancias: Carro #1, Carro #8, Autobús, Avión, Barco y Motocicleta

Dado que no se nos ha especificado cual es el objetivo que se persigue con este ejercicio, cabe hacer las siguientes reflexiones:

1. Para que un objeto sea considerado una instancia de la clase Bicicleta no es importante el color. A menos que se nos hubiera especificado que se trata de, por ejemplo; modelar un sistema para un almacén, el color resulta irrelevante.
2. Definir la clase Avión, al igual que una clase Barco y la clase Autobús está determinado por el problema a resolver, lo que en el proceso de desarrollo de software se llama requerimiento.
3. De igual manera podemos advertir que la clase Automóvil generaliza las características de las clases Autobús, Carro (o Coche), Barco, etc. Podemos también advertir que una clase Vehículo podría generalizar las características comunes y más generales de las clases Automóvil, Bicicleta y Balsa.
4. La relación entre estas clases será de herencia, en la que las clases inferiores, que son las que definen los conceptos más específicos, serán subclases de las clases que definan las características más generales (superclases).

2. UML (lenguaje unificado de modelado)

UML (Unified Modeling Language) es un lenguaje de modelado de software. Consiste de un conjunto de símbolos y reglas para visualizar, especificar, construir y documentar software. Está compuesto por tres categorías de diagramas:

1. De estructura.
2. De comportamiento.
3. De interacción.

Los diagramas de estructura enfatizan en los elementos de los que debe consistir el software que estamos construyendo. En esta categoría de diagramas se encuentra:

- Diagrama de clases.
- Diagrama de componentes.
- Diagrama de objetos.

- Diagrama de estructura compuesta (UML 2.0)
- Diagrama de despliegue.
- Diagrama de paquetes

Los diagramas de comportamiento enfatizan en lo que debe suceder en el sistema modelado:

1. Diagrama de actividades.
2. Diagrama de casos de uso.
3. Diagrama de estados.

Los diagramas de interacción, son un subtipo de diagramas de comportamiento. Enfatiza sobre el flujo de control y de datos entre los elementos del sistema modelado:

1. Diagrama de secuencia.
2. Diagrama de comunicación. (UML 2.0). Una variante del diagrama de colaboración de UML 1.x.
3. Diagrama de tiempos (UML 2.0).
4. Diagrama de vista de interacción (UML 2.0).

En esta asignatura nos centraremos en los diagramas de casos de uso, de clase y de secuencia.

2.1 Análisis orientado a objetos (aoo)

El AOO no se centra inicialmente en los objetos, sino más bien en qué es lo que debe hacer el sistema (el problema) y cuáles son las entidades externas con las que interactuará.

El proceso de Análisis y Diseño Orientado a Objetos es realmente un proceso cíclico evolutivo, que inicia con aproximaciones de los requerimientos y el diseño de clases. El proceso se detiene cuando se alcanza una especificación completa de los requerimientos, el análisis y el diseño.

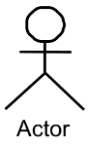
2.1.1 Diagrama de casos de uso

Un caso de uso es un flujo completo de eventos que ocurren entre el sistema y las entidades externas a él. Dichas entidades externas son llamadas actores. Un caso de uso es por lo tanto es una descripción de la interacción que se da entre los actores y el sistema.

Los eventos a los que se refiere la definición anterior se deben entender como aquellas acciones que son llevadas a cabo por los actores, que requieren de una respuesta del sistema. Los casos de uso, por lo tanto, representan requerimientos (requisitos) funcionales del sistema.

Simbología:

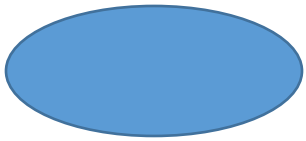
Actor



Es una entidad externa al sistema que necesita intercambiar información con él.

La notación utilizada en UML para representar un actor es una figura humana con el nombre del actor.

Caso de uso



Como ya se ha dicho, los casos de uso son flujos completos de eventos que ocurren entre los actores y el sistema. Se representan por una elipse con el nombre del caso de uso.

Relaciones

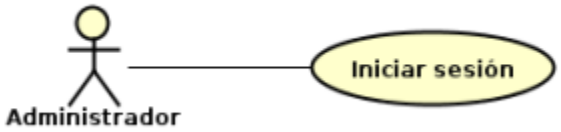
Existen 4 tipos de relaciones en los diagramas de casos de uso:

1. Asociación
2. Inclusión
3. Extensión
4. Generalización

Asociación

Describe la invocación de un caso de uso por parte de un actor. Se representa por una línea entre el actor y el caso de uso.

Como ya se ha mencionado, una de los principales objetivos de la POO es la reusabilidad. Este principio se aplica desde la fase de análisis, en la que es posible identificar flujos de eventos que pueden aislarse para que puedan ser utilizados directamente por los actores o por otros casos de uso.

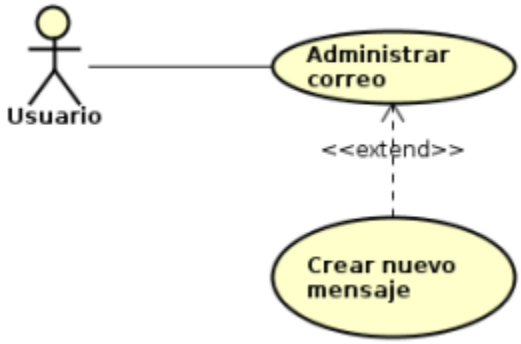


Extensión

Describe una relación condicional entre dos casos de uso en la que el caso de uso que extiende al caso de uso primario se realizará siempre que se cumplan las condiciones establecidas en el caso de uso primario. Normalmente estas condiciones son eventos, es decir; acciones que debe realizar el actor para que se realice el caso de uso extendido.

La extensión es representada por una flecha discontinua que va desde el caso de uso que extiende al caso de uso primario (el que es extendido), con el estereotipo *extend*.

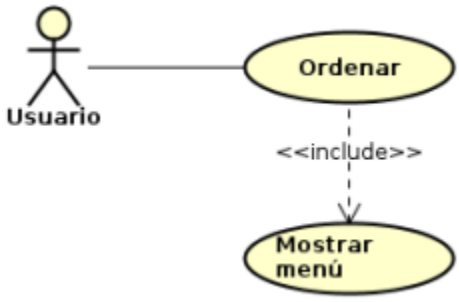
En el ejemplo que se muestra, el caso de uso *Crear nuevo mensaje* extiende al caso de uso *Administrar correo* de un cliente de correo electrónico.



El evento que podría iniciar el caso de uso *Crear nuevo mensaje* podría ser un botón *Nuevo o Redactar* en la interfaz del caso de uso *Administrar correo.*

Inclusión

Describe una relación obligatoria entre dos casos de uso del tipo hace uso, en el que el caso de uso que es incluido en el caso de uso primario será realizado siempre. La inclusión es representada por una flecha discontinua que va desde el caso de uso primario (el que incluye) al caso de uso que es incluido.



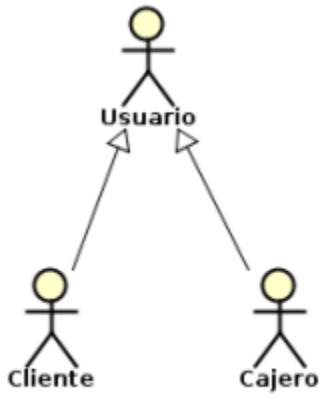
En el ejemplo que se muestra, el caso de uso *Ordenar*, que se refiere a hacer una orden de comida rápida en línea, incluye el caso de uso *Mostrar menú*. El menú se refiere al menú de platillos

disponible. El caso de uso *Mostrar menú* será realizado siempre que se inicie el caso de uso *Ordenar.*

Generalización

Describe una relación de herencia entre actores o casos de uso. Comúnmente la generalización se da de lo particular a lo general, es decir; se crean casos de uso o actores que sirvan para describir características generales de los ya identificados. La generalización se describe a través de una flecha continua con su punta vacía, desde el actor o caso de uso particular al que lo generaliza. En el ejemplo, el actor Usuario es una generalización de los actores Cliente y Cajero.

La generalización es poco utilizada en casos de uso. En su lugar, es más frecuente utilizar una relación de extensión.



Proceso:

1. Hacer una lista de potenciales actores.
2. Hacer una lista de potenciales casos de uso.
3. Asociar los actores con los casos de uso en los que participa.
4. Identificar la relación entre los casos de uso.

Para identificar las relaciones entre los casos de uso, deben tomarse en cuenta las siguientes consideraciones:

- Es posible que algunos casos de uso no hayan sido identificados hasta ese momento. Estos casos de uso faltantes pueden surgir al momento de identificar las relaciones entre los actores y los casos de uso o entre los casos de uso.
- Es posible que un caso de uso haya sido definido de forma muy extensa, y requiera ser divido en casos de uso separados que tendrán una relación entre ellos. En este punto el criterio que debe prevalecer es que el caso de uso describa un flujo de eventos completo.
- El flujo de los eventos determinará si un caso de uso requiere ser extendido o incluir otro caso de uso.
- Con el fin de buscar la reutilización, es posible que resulte más útil aislar un fragmento de un caso de uso, que pueda ser invocado desde otros casos de uso.

2.1.2 Diccionario de Casos de Uso

El Diccionario de casos de uso es la descripción narrada de los casos de uso identificados para el software que se está construyendo. La Descripción de casos de uso también se conocen como Casos de uso narrados. Sirven para detallar el flujo de eventos en un caso de uso, que posteriormente pueden ser utilizados en los diagramas de actividades, de secuencia o el prototipado.

En general, cada autor incorpora elementos a la platilla de descripción de casos de uso. En la tabla 2 se muestran los más importantes.

Tabla 2 Elementos más importantes de los casos de uso narrados.

Caso de uso:	**Nombre del caso de uso**
Descripción:	Breve descripción del caso de uso
Actores:	Lista de actores que participan

Precondiciones:	Condiciones que deben de cumplirse para que el caso de uso se realice
Flujo nombre de los eventos:	Descripción paso por paso del flujo de lo que se espera ocurra en el sistema
Flujo alternativo:	Descripción de los pasos que pueden ocurrir ante eventos no descritos en el flujo normal
Poscondiciones:	Condiciones que deben de cumplirse posterior a la realización del caso de uso