Patrones de diseño

Camilo Martínez – John López

11/12/2013

Un patrón de diseño se puede definir como la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Los patrones de diseño provienen del mundo de la arquitectura en el año 1979 el arquitecto Christopher Alexander el lanza un libro llamado The Timeless Way of Building; En el que abordaba la temática sobre planteamiento urbano y la construcción de edificios. En él se proponía el aprendizaje y uso de una serie de patrones para la construcción de edificios de una mayor calidad. El autor planteaba que cada patrón describe a un problema que se repetía innumerables veces en nuestro entorno.

En el año 1987 Ward Cunningham y Kent Beck aplican las ideas de Christopher para desarrollar un pequeño lenguaje de patrones, para aprender Smalltalk: "Using Pattern Languages for Object-Oriented Programs". Finalmente en el año 1990 se inicia un trabajo por un grupo denominado "Gang of Four" que finalmente saca a la luz un libro llamado "Design Patterns, Elements of Reusable Object-Oriented Software". En este libro se plantea tres clasificaciones de patrones la creacional, estructural y de comportamiento.

El primer patrón de creación permite:

- Abstraer el proceso de creación de objetos.
- Ayudar a crear sistemas independientes de cómo los objetos son creados, compuestos y representados.
- El sistema conoce las clases abstractas.
- Flexibilidad en qué se crea, quién lo crea, cómo se crea y cuándo se crea.

Los patrones de estructurales:

- Cómo clases y objetos se combinan para formar estructuras más complejas.
- Patrones basados en herencia.
- Patrones basados en composición.

Los patrones de comportamiento permiten:

- Enfatizan la colaboración entre objetos.

- Caracterizan un flujo de control más o menos complejo que será transparente al que utilice el patrón.

- Basados en herencia: Template Method e Interpreter.

A continuación ahondaremos en el patrón de diseño Chain of Responsibility que es un patrón de comportamiento que evita acoplar el emisor de una petición a su receptor dando a más de un objeto la posibilidad de responder a una petición. Para ello, se encadenan los receptores y pasa la petición a través de la cadena hasta que es procesada por algún objeto. Este patrón es utilizado a menudo en el contexto de las interfaces gráficas de usuario donde un objeto puede contener varios objetos. Según si el ambiente de ventanas genera eventos, los objetos los manejan o los pasan.

Imaginemos un contexto gráfico donde se puede obtener información de ayuda de cada elemento clickeando sobre él. La información dependerá de la parte de la interfaz donde se pinche. Puede darse la situación que dos "botones" iguales difieran en la información a mostrar. Para este tipo de problemas necesitamos un patrón que permita manejar dónde se produce un evento, quién es el responsable del mismo y cuál es la respuesta adecuada al mismo. Para implementar el patrón deben de tomarse las siguientes consideraciones:

- Implementar la cadena sucesora: Hay dos posibilidades de implementación. La primera es definir nuevos enlaces, ya sea en la clase Manejador o en las clase ManejadorConcreto. La segunda posiblidad es emplear enlaces ya existentes, por ejemplo empleando el patrón Composición.

- Conexión de los sucesores. Los propios ManejadoresConcretos son los encargados de propagar incondicionalmente la petición. Las referencias deben de estar definidas.

- Representación de las peticiones. Se puede emplear el paso por parámetros o variables mediante una función manejadora, o hacer uso de invocaciones a operaciones insertadas en el código.

La mayoría de los usos conocidos de este patrón se pueden usar en editores gráfico, protocolos industriales a niveles bajos y manejadores de evento. Pero en la vida cotidiana como se puede ejemplificar este patrón. Un ejemplo podría ser : Don Juan es un trabajador de un empresa internacional de tecnología y quiere cobrar todos los viáticos que se le adeudan. Don Juan realiza la solicitud de cobranza del viatico (a él solamente le interesa que se le cancele la deuda no le importa el medio y la forma en que se le pague) el al enviar la solicitud genera una cadena de responsabilidad. En esa cadena pueden existir múltiples personas (objetos) que son responsables de la aprobación o reprobación de cada viatico eso incluye a jefes y gerentes de la área de recurso humanos. Según las reglas de la empresa si el viatico es inferior a $1000000 solo necesita la autorización de la jefa de RRHH, en cambio si el viatico es superior a el monto mencionado el gerente de área es el que da la autorización y por ultimo si el monto es superior a $2000000 el gerente general es el que autoriza los pagos. Por lo tanto la forma

en que se genera los pagos de viáticos es una cadena de responsabilidad donde cada objeto responde a lo que es responsable. Es muy importante que la cadena de responsabilidad parta siempre de menor a mayor por la razón que si es un monto es muy bajo no tendría sentido que el gerente general lo revise esto sirve absolutamente para la agilizar el proceso.

References

[http://slimcode.blogspot.com/2009/10/cadena-de-responsabilidad-chain-of.html]

[http://geeks.ms/blogs/lontivero/archive/2007/10/10/patterns-chain-of-responsibilites-pattern.aspx]

[http://patronesdediseno.blogspot.com/2009/05/patron-chain-of-responsability.html]

[http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/182]