Instituto Tecnológico de Costa Rica

Algoritmos y Estructuras de Datos II

Estudiantes:
Sergio Martínez Bonilla 2020039334
Luis David Delgado Jimenez 2020072813

Prof. Antonio Gonzales Torres

Grupo 3

Trabajo Extraclase II

I Semestre 2021

Resumen: Impacto de los patrones de diseño orientados a objetos en el desarrollo de software

1 Introducción

Desde que la introducción de la programación orientada a objetos, esta se considera un estándar en lo que corresponde al "buen" desarrollo de software. Hay muchas maneras de desarrollar mediante estos métodos, pero debido a su complejidad se han creado patrones como "soluciones comunes para problemas comunes". Representan soluciones recurrentes para algún problema en particular, como procesamiento de datos, interfaces, etc. Pueden funcionar como planos en la solución de un problema, describen el problema, solución y en que situaciones conviene aplicarlos, junto con las consecuencias de hacerlo. En el libro "Gang of four", los patrones poseen estos elementos: intención, motivación aplicabilidad, estructura, participantes, colaboración, implementación, ejemplo de código, problemas conocidos y patrones relacionados.

2 Trabajo Relacionado

Los patrones de diseño orientados a objetos son un área de investigación activa, entre ellas se ha concluido que tienen gran cantidad de usos, pueden ser reutilizados constantemente, proveen las soluciones más extendibles, pero no se puede aplicar a todos los patrones, cada uno tiene sus puntos fuertes y puntos débiles. Aplicarlos definitivamente otorga ventajas, pero también tienen sus limitaciones por lo que se debe valorar bien la situación en la que se deseen utilizar.

Sin embargo, también se explican estos mismos como una manera de enseñanza para los desarrolladores para construir y revisar software, y esta misma razón también los convierte en una herramienta valiosa para los desarrolladores profesionales, quienes ya sea que aprenden o promueven el uso de estos mismos en su trabajo.

3 Catálogo de Patrones de Diseño

Existen 3 clases básicas de patrones de diseño:

- **Estructurales:** Estos patrones se encargan de identificar una manera sencilla de que se den las relaciones entre las entidades. Se basan en la composición de la clase y el objeto. Algunos utilizan las herencias, mientras que otros se basan en la manera de componer los objetos para obtener ciertas funcionalidades.
- **Creacionales:** Se componen de dos ideas principales. La primera es encapsular el conocimiento sobre las clases que usa el sistema. La otra corresponde a esconder como son creadas y combinadas las instancias de estas clases. Pueden ser utilizadas como un objeto en la creación de otro objeto. Algunos ejemplos: builder, abstract factory, singleton, etc.
- **De comportamiento:** Se identifican comúnmente como patrones que conectan las funciones entre objetos, incrementando la flexibilidad y comunicación del programa, no están en control. Por ejemplo el observer, iterator, command, etc.

4 ¿Cómo seleccionar un patrón de diseño?

Existen 5 principales maneras para buscar y seleccionar patrones de diseño: nombre, repositorios y catálogos, sistemas recomendados, lenguajes formales, motores de búsqueda, entre otros. Se crean repositorios para aumentar la disponibilidad de estos mismos, así como documentos que describen como implementarlos en lenguajes formales, también para cuál patrón se adapta mejor a los tipos de problemas, buscar también a través de palabras clave para dar con la información necesitada más rápido.

Un consejo puede ser buscar mediante la clasificaci'on de estos mismos, ya que cada uno tiene su propia finalidad, puede utilizarse un esquema de clasificación. También puede ser útil el estudiar las relaciones entre patrones, para así llegar al patrón o grupo de patrones correctos. Otra manera puede ser identificarlos mediante los tipos previamente explicados en la sección anterior. También buscar las causas del rediseño para averiguar que es lo que implica el problema y comparar con los patrones, a su vez otro consejo sería ver en el proyecto que es lo que puede cambiar constantemente, conectar con lo que debe ser cambiado sin rediseñar, considerando aspectos de los patrones de diseño que permitirían la variabilidad independiente para que el diseñador lo cambie sin rediseñar.

5 ¿Cómo utilizar los patrones de diseño?

Se debe de leer la información acerca del seleccionado para tener una idea acerca de su aplicación, consecuencias e implementación, y asegurar que es el indicado para el problema que se busca resolver. Entender como se relacionan los componentes de este patrón (clases y objetos). Estudiar el código es recomendable, así como incorporar el patrón mediante un nombre que permita identificarlo al ver el código. También se deben identificar las clases existentes que el patrón va a afectar. Para lo que es la implementación en código propiamente del mismo, sirven las secciones de implementación de la guía y los ejemplos de esta, que contienen ya un código montado para una situación específica.

6 Ventajas de los patrones de diseño

Pueden servir para reducir el tiempo de desarrollo, ya que se utilizan soluciones conocidas y creadas por expertos para los desarrolladores en lugar de reinventarlas. Actualmente se busca mucho que los ingenieros entreguen calidad en un límite de tiempo, por lo que ahorrar el mismo puede ser de lo más útil.

También, los patrones de diseño proveen mayor flexibilidad y extensibilidad. Mejoran la estructura, simplifican el mantenimiento y ayudan a evitar problemas arquitectónicos. Los cambios a futuro no son caros, es probable que falle el sistema, pero se tendría mejor conocimiento del mismo para arreglarlo.

Ayudan a mantener una mejor comunicación entre los equipos de desarrollo, y en general entre todos los desarrolladores de software al ser un tema tan estudiado, y también puede ayudar a que los menos experimentados creen productos con alta calidad.

Se encuentran subespecificados, lo que les permite la flexibilidad en las soluciones a las que son aplicados, para que se ajusten a estas. Implementar uno puede ser la mejor manera de aprenderlo gracias a esto, ya que otra de sus ventajas es que capturan el conocimiento que se entiende de manera implícita. Codifican el conocimiento que se entiende de manera intuitiva, se puede mejorar la manera de documentar y explicar el funcionamiento de un sistema, esto debido a que también ayudan a promover una manera estructurada de documentar la arquitectura del software, captura las interacciones esenciales ignorando detalles innecesarios, para que así su mantenimiento a futuro sea más eficiente.

Como se ha mencionado anteriormente, son soluciones conocidas para problemas recurrentes en el desarrollo de software. Capturan las experiencia y la hacen accesible para desarrolladores sin ella sin mucha dificultad, sin embargo se debe comprender el problema de antemano. También pueden ser utilizados para otorgar un punto de adaptabilidad, donde todo se une y se determina la colaboración de lo que compone al programa. Agregar que también hacen mas sencillo el reutilizar diseños y evitar alternativas que disminuyen su reutilziación. Permiten el cambio independiente de algún aspecto sin afectar otros. Los patrones pueden ser utilizados reactiva y proactivamente a través de la fragmentación y abstracción del diseño, y convertir en situaciones de ganancia muchas facetas de la calidad que se ven como mutuamente exclusivas.

7 Desventajas de los patrones de diseño

Se puede cometer el error de implementar un patrón en donde no es requerido. Muchas de las personas que utilizan estos patrones no han recibido ningún curso o recurso educativo acerca del uso de los mismos. En algunos casos pueden incrementar la complejidad de un problema, se hace más sofisticado de lo que debe ser. También puede ocurrir lo contrario, se simplifica más de lo que debe, por lo que puede ser menos óptimo de lo requerido y se debe cambiar en los cambios futuros. Esta última es mucho más frecuente que hacer el problema más complejo. Incluso en algunas ocasiones se da que se ataca el problema equivocado, por lo que nos lleva al siguiente punto de que no necesariamente mejoran la calidad del programa. Por ejemplo, una implementación enredada de estos mismos provoca el efecto contrario en la calidad y es incierto el impacto de los patrones en algunos aspectos de la calidad.

También, el aprender una colección de patrones no reemplaza a las habilidades de implementación y diseño, puede provocar que se crea saber más de la solución a un problema a lo que en realidad conocen.

Está la posibilidad de que los patrones de diseño aumenten la complejidad de un programa, sin beneficiar los resultados. También los catálogos de estos mismos pueden ser muy abstractos, lo que los hacen difíciles de entender e implementar. El uso excesivo de patrones tampoco es beneficioso, en vez de mejorar los presentes, crear de nuevo no es beneficioso, no son la solución mágica para todos los problemas del desarrollo. Por último, los lenguajes. Existen muchos lenguajes para cualquier contexto, ya sean funcionales o no.

8 Conclusión y trabajo a futuro

Los patrones de diseño no son la solución perfecta para todo, así como poseen ventajas, poseen desventajas si no son aplicados de manera correcta. Hace falta realizar mucho trabajo sobre estos mismos para así hacerlos entendibles a los nuevos desarrolladores, ya que su uso depende del conocimiento de cada desarrollador.

Referencia del Artículo Original

[1]S. Ramasamy, G. Jekese and C. Hwata, Impact of Object Oriented Design Patterns in Software Development. 2016.