**Nombre:** Cornejo Macias Luis Fernando  **Fecha:** 20 de junio del 2020

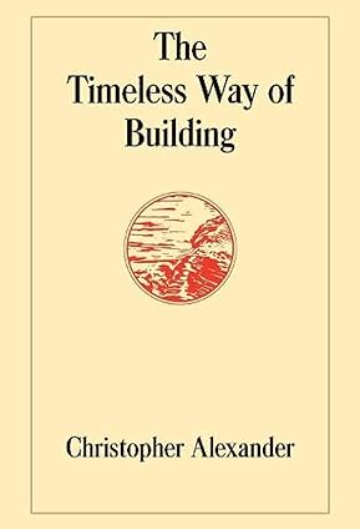
**Patrones de diseño**

**Patrón de diseño**

Breve definición:

Los patrones de diseño son una especie de técnicas para resolver problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema.

Un poco de historia…

En 1979 el arquitecto Christopher Alexander aportó al mundo de la arquitectura el libro The Timeless Way of Building; en él proponía el aprendizaje y uso de una serie de patrones para la construcción de edificios de una mayor calidad, en la que esa mayor calidad se refería a la arquitectura antigua y la menor calidad correspondía a la arquitectura moderna, que el romper con la arquitectura antigua había perdido esa conexión con lo que las personas consideraban que era calidad.

En palabras de este autor, "Cada patrón describe un problema que ocurre infinidad de veces en nuestro entorno, así como la solución al mismo, de tal modo que podemos utilizar esta solución un millón de veces más adelante sin tener que volver a pensarla otra vez."

Entonces entendemos que los patrones de diseño se crean con el fin de solucionar los problemas que se presentan, identificándolo por medio de los síntomas que va llevando el inconveniente.

Y en base a las características que presenta el problema, existen muchos patrones de diseño referente a el desarrollo de software en este caso cual estamos inclinándonos, y así mismo categorizando esas características que se presenta tenemos como base 3 tipos de patrones de diseño, que son:

1. Creacional: Los patrones de creación proporcionan diversos mecanismos de creación de objetos, que aumentan la flexibilidad y la reutilización del código existente de una manera adecuada a la situación. Esto le da al programa más flexibilidad para decidir qué objetos deben crearse para un caso de uso dado.

* Abstract Factory
* Builder Patterns
* Factory Method
* Prototype
* Singleton

1. Estructural: Facilitan soluciones y estándares eficientes con respecto a las composiciones de clase y las estructuras de objetos. El concepto de herencia se utiliza para componer interfaces y definir formas de componer objetos para obtener nuevas funcionalidades.

* Adapter
* Bridge
* Composite
* Decorator
* Facade
* Flyweight
* Proxy

1. Comportamiento: El patrón de comportamiento se ocupa de la comunicación entre objetos de clase. Se utilizan para detectar la presencia de patrones de comunicación ya presentes y pueden manipular estos patrones.

* Chain of responsibility
* Command
* Interpreter
* Iterator
* Mediator
* Memento
* Observer
* State
* Strategy
* Template method
* Visitor

Como se mencionó cada uno en base en como se presenta y se va desarrollando podemos de tal forma filtrar el problema y encontrar un patrón de diseño que vaya acorde, para así ofrecernos la solución o llegar a ella en un mayor porcentaje.

**Patrón de arquitectura**

Definición:

Antes de comenzar el desarrollo de algún software o solución tecnológica, conviene elegir la arquitectura adecuada que proporcione la funcionalidad y atributos de calidad deseados. En ese particular, cobra importancia el conocer los distintos patrones de arquitecturas, o maneras de abordar los temas de diseño de aplicaciones complejas, esto con el propósito de obtener el mejor diseño antes de iniciar algún esfuerzo en el desarrollo de software.

Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones. En comparación con los patrones de diseño, los patrones arquitectónicos tienen un nivel de abstracción mayor.

La elaboración de una arquitectura para una solución concreta puede verse como un proceso de selección, adaptación y combinación de patrones. El arquitecto de software debe decidir cómo reutilizar un patrón, hacer los ajustes necesarios al contexto específico, a las restricciones del problema, y a la solución que está diseñando.

Seguidamente, se abordarán los patrones arquitectónicos más comunes aplicados a desarrollos de software, con sus principales características.

* Capas (Layer pattern): Este patrón suele ser el más utilizado, y se enfoca en estructurar aplicaciones que pueden descomponerse en grupos de subtareas o responsabilidades, las cuales se encuentran en un nivel concreto de abstracción. Cada capa proporciona servicios a la capa inmediatamente superior.
* Cliente Servidor (Client-Server pattern): Este patrón consta de dos (2) partes: un servidor y múltiples clientes. El componente servidor proporciona servicios a múltiples componentes cliente. Los clientes solicitan servicios al servidor, y éste proporciona los servicios pertinentes a dichos clientes.
* Maestro-Esclavo (Master-Slave pattern): Este patrón consta de dos (2) partes: maestro y esclavos. El componente maestro distribuye el trabajo entre componentes esclavos idénticos, y determina un resultado final a partir de los resultados que devuelven los esclavos.
* Filtros-Tuberías (Pipe-Filter pattern): Este patrón puede utilizarse para estructurar sistemas que producen y procesan un flujo de datos. Cada paso de procesamiento está encerrado dentro de un componente de filtro. Los datos que se van a procesar pasan a través de tuberías. Estas tuberías pueden utilizarse para almacenar o sincronizar datos.

Referencias

Wikipedia. (s.f.). Patrón de diseño. Recuperado de <https://es.wikipedia.org/wiki/Patrón_de_diseño>

Alexander, C. (1979). The Timeless Way of Building. Nueva York: Oxford University Press. Recuperado de <https://www.amazon.com/Timeless-Way-Building-Christopher-Alexander/dp/0195024028>

Profile. (s.f.). Patrones de diseño de software. Recuperado de <https://profile.es/blog/patrones-de-diseno-de-software/>

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Boston: Addison-Wesley. Recuperado de <https://www.amazon.com/Design-Patterns-Elements-Reusable-Object-Oriented/dp/0201633612>

Google. (s.f.). Patrones arquitectónicos. Recuperado de <https://www.google.com/search?client=opera-gx&q=patrones+arquitectonicos&sourceid=opera&ie=UTF-8&oe=UTF-8>

Wikipedia. (s.f.). Patrones de arquitectura. Recuperado de <https://es.wikipedia.org/wiki/Patrones_de_arquitectura>

Castro Ramírez, E. R. (s.f.). Patrones en la arquitectura de software. Recuperado de <https://es.linkedin.com/pulse/patrones-en-la-arquitectura-de-software-elmo-renato-castro-ramirez>