FACULTAD DE CIENCIAS FISICO MATEMATICAS

Patrones de diseño

Estudiante: Jorge Eduardo Castro Tristan

Matricula: 1640167

Materia: DOO

Fecha: 13/03/17

Patrones de diseño

¿Que es?

Los Patrones de diseño son herramientas para solucionar problemas de Diseño enfocados al desarrollo de software, estos patrones deben ser reusables permitiendo así que sean adaptados a diferentes problemáticas.

Cuando hablamos de problemáticas nos referimos a condiciones o problemas reiterativos en el desarrollo de software donde la solución se encuentra identificada mediante la aplicación de una serie de pasos, adaptando el sistema a una estructura definida por un patrón, garantizando que esta solución pueda ser aplicada cuantas veces sea necesario en circunstancias similares, evitando así la búsqueda de otras soluciones cuando estas ya se han dado anteriormente.

¿Para que sirven?

- Proporcionar catálogos de elementos reusables en el diseño de sistemas software.
- Evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente.
- Formalizar un vocabulario común entre diseñadores.
- Estandarizar el modo en que se realiza el diseño.
- Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

¿Como se documentan?

Los patrones de diseño se documentan utilizando plantillas formales con unos campos estandarizados. Existen varias plantillas ampliamente aceptadas, aunque todas ellas deben al menos documentar las siguientes partes de un patrón:

- **Nombre**: describe el problema de diseño.
- **El problema**: describe cuándo aplicar el patrón.
- **La solución**: describe los elementos que componen el diseño, sus relaciones, responsabilidades y colaboración.

La plantilla más utilizada es la plantilla GOF que consta de los siguientes campos:

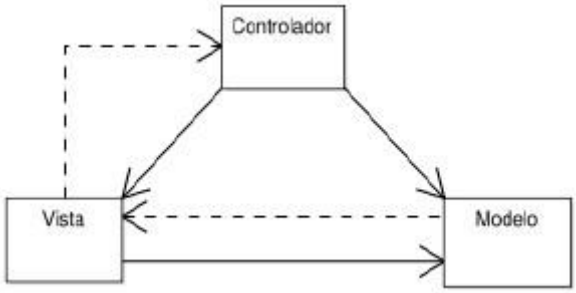
- Nombre del patrón: Ayuda a recordar la esencia del patrón.
- Clasificación: Los patrones originalmente definidos por GOF se clasifican en tres categorías "Creacional", "Estructural", "Comportamiento".
- Propósito / Problema: ¿Qué problema aborda el patrón?
- También conocido como: Otros nombres comunes con los que es conocido el patrón.
- Motivación: Escenario que ilustra el problema.
- Aplicabilidad: Situaciones en las que el patrón puede ser utilizado.
- Estructura: Diagramas UML que representan las clases y objetos en el patrón.
- Participantes: Clases y/o objetos que participan en el patrón y responsabilidades de cada uno de ellos
- Colaboraciones: ¿Cómo trabajan en equipo los participantes para llevar a cabo sus responsabilidades?
- Consecuencias: ¿Cuales son los beneficios y resultados de la aplicación del patrón?
- Implementación: Detalles a considerar en las implementaciones. Cuestiones específicas de cada lenguaje OO.
- Código de ejemplo
- Usos conocidos: Ejemplos del mundo real.

- Patrones relacionados: Comparación y discusión de patrones relacionados. Escenarios donde puede utilizarse en conjunción con otros patrones.

Ejemplos de patrones de diseño

Modelo Vista Controlador (MVC)

Se aplica el MVC donde reestructuramos nuestro código fuente separándolo en 3 partes funcionales con comportamientos definidos que posteriormente son relacionadas entre si, facilitando la mantenibilidad y flexibilidad del código, estas 3 partes son el Modelo, la Vista y el Controlador.



- Modelo : Representa la información y los datos procesados por el sistema, gestionando la forma como se accede a estos y la lógica de negocio de la aplicación.
- Controlador : Representa el puente de interacción entre el Modelo y la Vista, define la forma como se relacionan los componentes de la aplicación.
- Vista : Es la representación del modelo mediante una interfaz gráfica de usuario, es la forma como el usuario interactúa con el sistema, permitiendo la ejecución de eventos he ingreso de información que serán procesados por el Modelo.

Observer

Se aplica el patrón observador para desacoplar la clase que contenga objetos dependientes, estableciendo relaciones entre los objetos, para esto se tiene una clase SujetoConcreto que sera el objeto principal y tendremos los objetos cliente o dependientes que serían de clase ObservadorConcreto, cuando el estado del SujetoConcreto cambia, se envía una señal a cada uno de sus Observadores y estos automaticamente cambian su estado basados en el estado del objeto principal.

Data Access Object (DAO)

Se utilizan clases DAO para encapsular todos los accesos a la fuente de datos desacoplando de esta manera la lógica de negocio de la lógica de acceso a datos, estableciendo mayor organización y facilitando la mantenibilidad y extensibilidad del código fuente.

Singleton

Se utiliza el patrón Singleton como mecanismo para limitar el numero de instancias de la clase, compartiendo la misma instancia por diferentes objetos del sistema, obteniendo una variable global para toda la aplicación.

Adapter

Utilizando el patrón Adapter podemos relacionar clases incompatibles entre si, generando un mecanismo que permita extender su comportamiento por medio de una clase puente que sirva como interfaz entre la clase en cuestión y el resto de clases que quieran hacer uso de sus funcionalidades.