PATRÓN DE DISEÑO

Un patrón de diseño es una abstracción de una solución en un nivel alto. Los patrones solucionan problemas que existen en muchos niveles de abstracción. Hay patrones que abarcan las distintas etapas del desarrollo; desde el análisis hasta el diseño y desde la arquitectura hasta la implementación.

Debido a las características que proporcionaron los lenguajes orientados a objetos (como herencia, abstracción y encapsulamiento) les permitieron relacionar entidades de los lenguajes de programación a entidades del mundo real fácilmente, los diseñadores empezaron a aplicar esas características para crear soluciones comunes y reutilizables para problemas frecuentes que exhibían patrones similares.

Un grupo llamado GoF clasificaron los patrones en 3 grandes categorías basadas en su PROPÓSITO: creacionales, estructurales y de comportamiento.

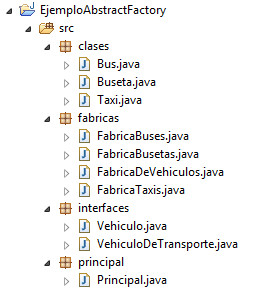
* Creacionales: Patrones creacionales tratan con las formas de crear instancias de objetos. El objetivo de estos patrones es de abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados.
* Estructurales: Los patrones estructurales describen como las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades. Estos objetos adicionados pueden ser incluso objetos simples u objetos compuestos.
* Comportamiento: Los patrones de comportamiento nos ayudan a definir la comunicación e iteración entre los objetos de un sistema. El propósito de este patrón es reducir el acoplamiento entre los objetos.

|  |  |  |  |
| --- | --- | --- | --- |
| Propósito - Ámbito | Creación | Estructural | Comportamiento |
| Clase | Factory Method | Adapter | Interpreter  Template Method |
| Objetos | Abstract Factory  Builder  Prototype  Sigleton | Adapter  Bridge  Composite  Decorator  Facade  Flyweight  Proxy | Chain of Responsablity  Command  Iterator  Mediator  Memento  Observer  Strategy  Visitor |

Tabla de Clasificación de Patrones según GoF (gang of Four)

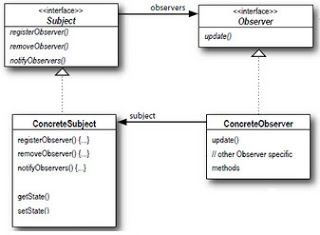
Patrones de Fabrica Abstracta

Los patrones Abstract Factory trabajan alrededor de una super-fábrica que crea otras fábricas. Esta fábrica también es llamada fábrica de fábricas. Este tipo de diseño cae en la categoría de creational patter ya que nos proporciona una de las mejores formas de crear un objeto



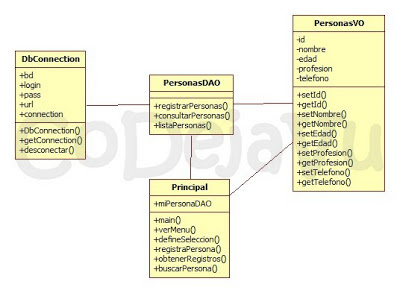
Observer

Es un patrón de comportamiento permite actualizar automáticamente el estado de un objeto dependiendo del estado de un objeto principal, cuando el objeto principal cambia sus objetos dependientes se actualizan.



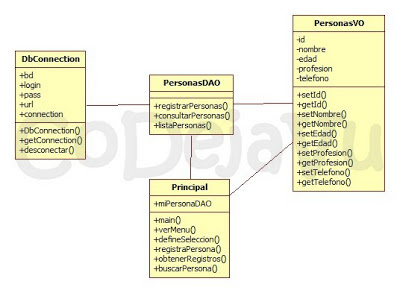
Value Object

Consiste básicamente en la agrupación de datos dentro de un objeto, estos datos representan los campos de una tabla o entidad de la BD y facilitan su mantenimiento y transporte dentro del sistema.



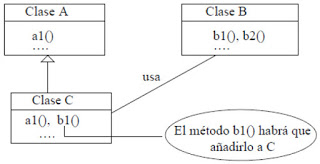
Data Acess Object

Facilita y estructura el acceso a la información de una Base de Datos, separando la persistencia de objetos de la lógica de acceso a datos, brindando mayor flexibilidad ya que por ejemplo al momento de hacer un cambio en la lógica de negocio, esto seria transparente para la lógica de acceso a la información.



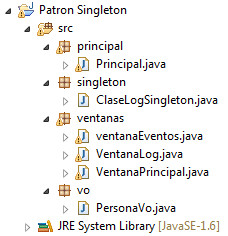
Delegate

Permite extender y reutilizar la funcionalidad de una clase sin utilizar el mecanismo de herencia.



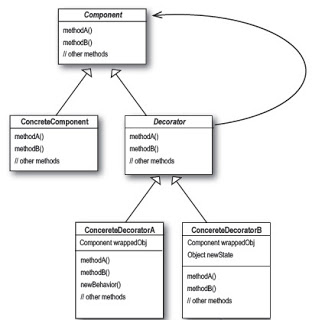
Sigleton

Permite la creación de una única instancia de clase permitiendo que sea global para toda la aplicación.



Decorator

Este patrón permite agregar funcionalidades o responsabilidades a objetos de forma transparente y dinámica, sin ser dependiente de la herencia en su totalidad.



Adapter

Permite la cooperación entre clases para extender sus funcionalidades a clases de diferentes tipos, que no pueden usarlas por mecanismos comunes como la herencia.

