UANL
UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FCFM
FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

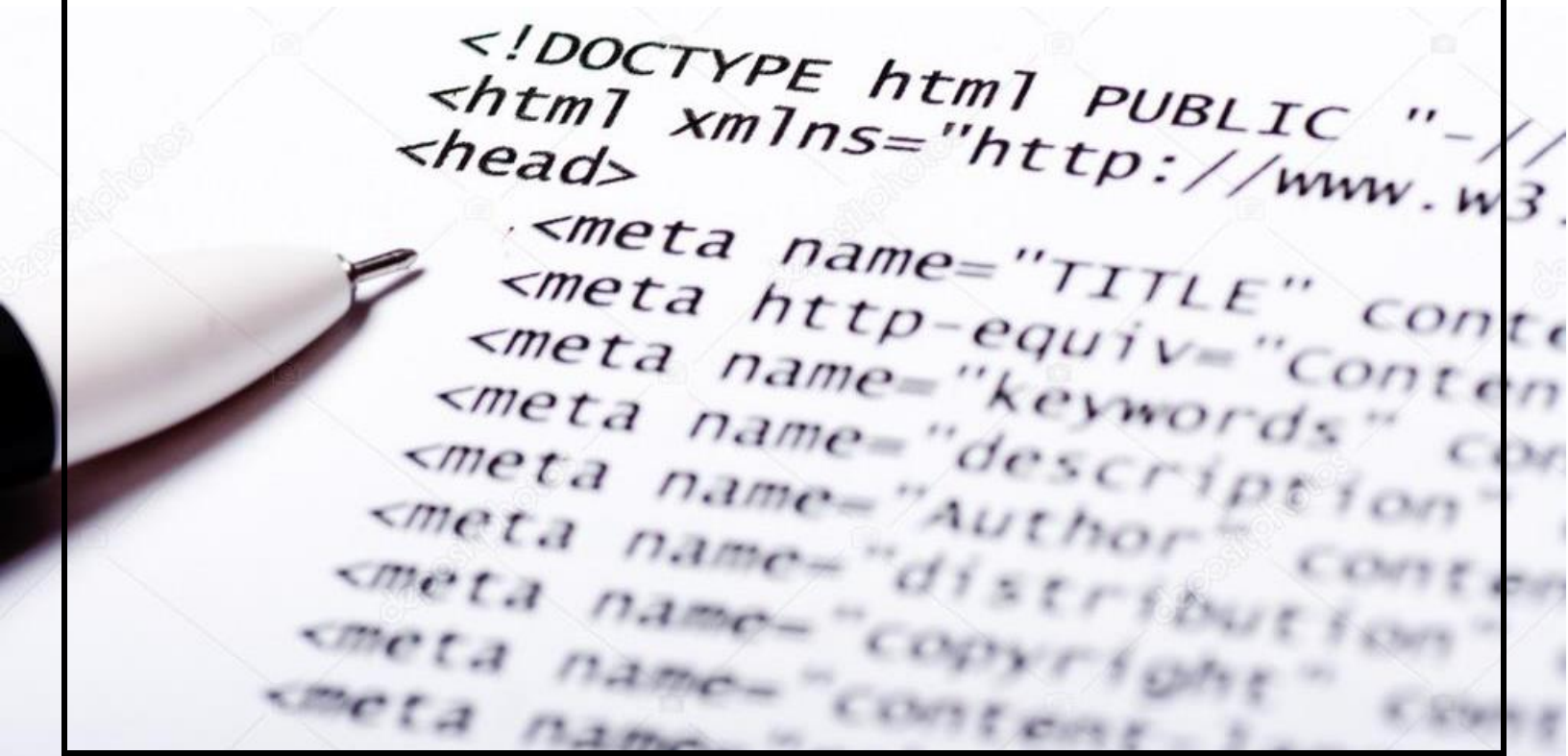
Universidad Autónoma de Nuevo León
Facultad de Ciencias Físico Matemáticas

# Diseño Orientado a Objetos

Lic. Miguel Angel Salazar S.

Tarea 6.- Patrones de Diseño

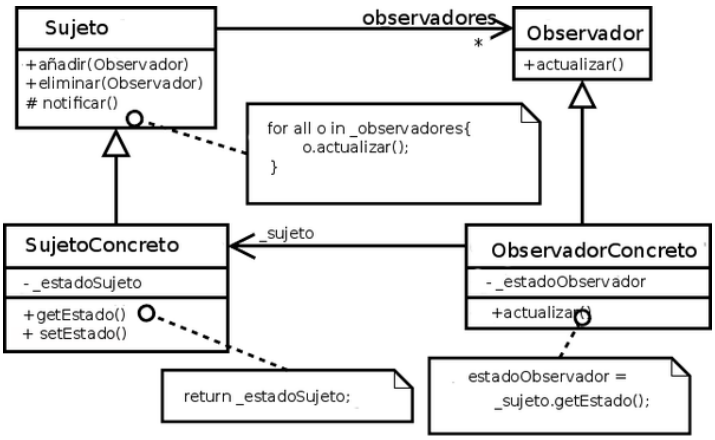
JOSE ISRAEL RESENDIZ VALENCIA
MATRICULA: 1757415



# Introducción

El presente documento, es una investigación sobre los patrones de diseño, el principal objetivo es aprender sobre que es un patrón de diseño, ¿Cuáles existen? Y para que nos sirven. Tal como lo sabemos, mantener en orden el desarrollo de nuestro software, nos beneficia a que este muy presencial. Para esto nos puede servir un patrón de diseño, ya que en este mismo nos podemos apoyar, para crear alguna estructura en general, o bien todo un mapa completo de nuestra aplicación. De cómo interacciona las clases dentro de nuestro software creado.

# Patrones de Diseño




Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software.

Es decir, son una técnica para resolver problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Debemos tener presente los siguientes elementos de un patrón:

Su nombre, el problema (cuando aplicar un patrón), la solución (descriptiva abstracta del problema) y las consecuencias (costos y beneficios).

## Los objetivos de los patrones de diseño pretenden:

- Proporcionar catálogos de elementos reusable 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 condensado conocimiento ya existente.

## ¿Cómo se documenta?

El uso de un patrón de diseño en un diseño concreto se documenta, fundamentalmente, señalando la relación que hay entre los elementos estructurales del patrón y los elementos estructurales del diseño.

# Patrones de diseño

**MVC o Modelo – Vista – Controlador**, es un patrón de arquitectura de software que, utiliza estos 3 componentes, lo que hace es separar la lógica de la aplicación de la lógica de la vista de una aplicación.


**Dato interesante...**

La razón por la que se utiliza MVC, es que permite separar los componentes de nuestra aplicación dependiendo de la responsabilidad que tiene, esto significa que cuando hacemos un cambio en alguna parte de nuestro código, esto no afecte otra parte del mismo.

## Modelo

Se encarga de los datos, generalmente (pero no obligatoriamente) consultando la base de datos. Actualizaciones, consulta, búsquedas, etc.

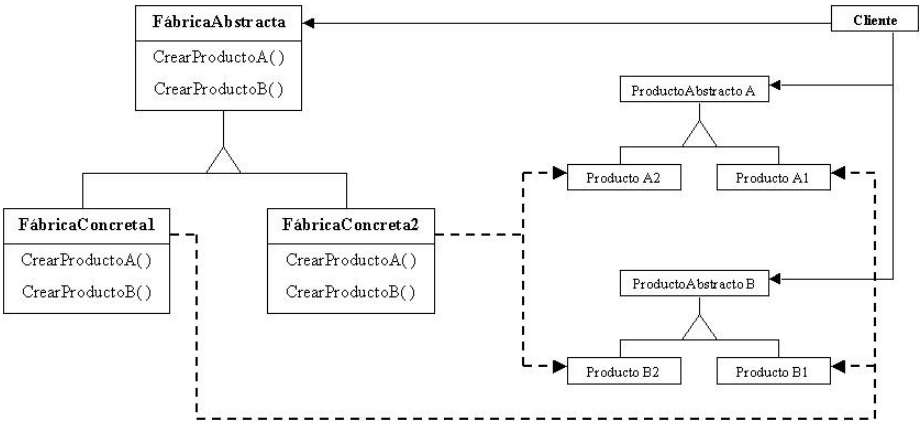
## Controlador

Se encarga de controlar los datos, recibe las órdenes del usuario y se encarga de solicitar los datos al modelo y de comunicarse a la vista.

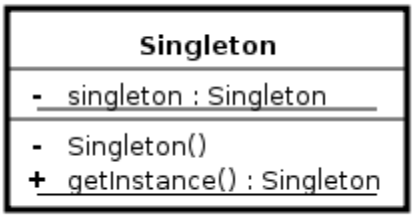
## Vistas

Son la representación visual de los datos, todo lo que tenga que ver con la interfaz gráfica. Ni el modelo, ni el controlador se preocupan de cómo se verán los datos, esa responsabilidad es únicamente de la vista.

**Abstract Factory o conocida como Fabrica Abstracta**, este patrón esta aconsejado cuando se prevé la inclusión de nuevas familias de productos, pero puede resultar contraproducente cuando se añaden nuevos productos o cambian los existentes, puesto que afectaría a todas las familias creadas.




Este patrón se utiliza para proporcionar una interfaz común para crear una serie de objetos (Productos) bajo una u otra arquitectura o framework, sin que los clientes de dichos productos tengan que tener conocimiento de la arquitectura elegida para implementar cada familia de productos.




**Singleton o instancia única,** es un patrón de diseño que permite restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto.

El diagrama UML que se encuentra arriba es una clase que implementa el patrón singleton.

Se implementa creando en nuestra clase un método que crea una instancia del objeto solo si todavía no existe alguna. Para asegurar que la clase no puede ser instanciada nuevamente se regula el alcance del constructor (con modificadores de acceso como protegido o privado).

# Conclusión

Durante el curso de Diseño Orientado a Objetos y su Laboratorio, los que me gusto más sobre patrones de diseño, fue la manera de utilizar MVC, ya que facilita en mantener el orden en la aplicación, y no ayuda a ver de una manera más clara como esta implementado todo, y que hace referencia a tales clases.

En fin los patrones de diseño son muy importantes y sin dudas, estos deben de incluir en la documentación para ver cómo está compuesto el software que creamos.

# Referencias


https://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o#Estructuras_o_plantillas_de_patrones

https://www.researchgate.net/publication/316595607_Estandar_para_documentar_el_uso_de_patrones_de_diseno_en_un_diseno_de_software

https://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o

https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador

https://codigofacilito.com/articulos/mvc-model-view-controller-explicado

https://es.wikipedia.org/wiki/Abstract_Factory

https://ociotec.com/patron-de-diseno-fabrica-abstracta/

https://es.wikipedia.org/wiki/Singleton