¿Qué es un Patrón de Diseño?

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

En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Debemos tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios).

Los patrones de diseño se clasifican como se muestra a continuación:

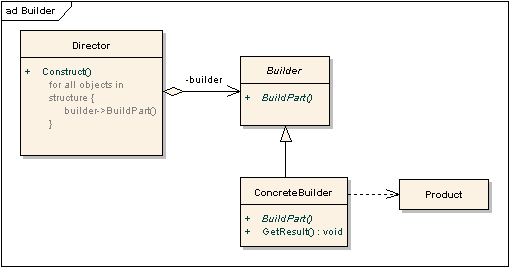
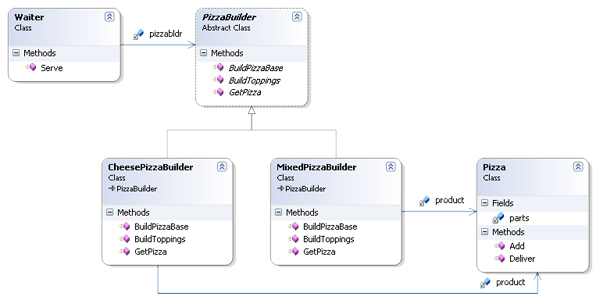
* Patrones Creacionales: Inicialización y configuración de objetos.
* Patrones Estructurales: Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes.
* Patrones de Comportamiento: Más que describir objetos o clases, describen la comunicación entre ellos.

**Builder**

 - Se necesita crear un objeto en varios pasos.

Construye una clase basada en los requisitos donde el Director pide al constructor (builder) que construya cada una de las partes. El patrón del constructor no se usa de manera independiente sino otros patrones tendrán patrones de construcción para crear partes más complicadas.

El diagram de clase se asemeja a esto

  
  
**Ejemplo**  
  
  
  
**Quien es que?**  
  
Waiter is Director  
PizzaBuilder is Builder  
CheesePizzaBuilder and MixedPizzaBuilder are Concretebuilder  
Pizza is product  
  
Cuando al Waiter (Director) se le pide que sirva crea la Pizza (Product) usando el PizzaBuilder.

# **Modelo–vista–controlador**

El modelo–vista–controlador (MVC) es un patrón de [arquitectura de software](https://es.wikipedia.org/wiki/Arquitectura_de_software) que separa los [datos](https://es.wikipedia.org/wiki/Datos) y la [lógica de negocio](https://es.wikipedia.org/wiki/L%C3%B3gica_de_negocio) de una aplicación de la [interfaz de usuario](https://es.wikipedia.org/wiki/Interfaz_de_usuario) y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres [componentes](https://es.wikipedia.org/wiki/Componentes) distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario.

Este patrón de [arquitectura de software](https://es.wikipedia.org/wiki/Arquitectura_de_software) se basa en las ideas de [reutilización de código](https://es.wikipedia.org/wiki/Reutilizaci%C3%B3n_de_c%C3%B3digo) y la [separación de conceptos](https://es.wikipedia.org/wiki/Separaci%C3%B3n_de_conceptos), características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

**La capa del Modelo**

El modelo representa la parte de la aplicación que implementa la lógica de negocio. Ésto significa que es responsable de la recuperación de datos convirtiéndolos en conceptos significativos para la aplicación, así como su procesamiento, validación, asociación y cualquier otra tarea relativa a la manipulación de dichos datos.

A primera vista los objetos del modelo pueden ser considerados como la primera capa de la interacción con cualquier base de datos que podría estar utilizando tu aplicación. Pero en general representan los principales conceptos en torno a los cuales se desea implementar un programa.

En el caso de una red social, la capa de modelo se haría cargo de tareas tales como guardar datos del usuario, el almacenamiento de asociaciones con amigos, el almacenamiento y la recuperación de fotos de los usuarios, encontrar sugerencias de nuevos amigos, etc. Mientras que los objetos del modelo pueden ser considerados como “Amigo”, “Usuario”, “Comentario” y “Foto”.

**La capa de la Vista**

La vista hace una presentación de los datos del modelo estando separada de los objetos del modelo. Es responsable del uso de la información de la cual dispone para producir cualquier interfaz de presentación de cualquier petición que se presente.

Por ejemplo, como la capa de modelo devuelve un conjunto de datos, la vista los usaría para hacer una página HTML que los contenga. O un resultado con formato XML para que otras aplicaciones puedan consumir.

La capa de la Vista no se limita únicamente a HTML o texto que represente los datos, sino que puede ser utilizada para ofrecer una amplia variedad de formatos en función de sus necesidades tales como videos, música, documentos y cualquier otro formato que puedas imaginar.

**La capa del Controlador**

La capa del controlador gestiona las peticiones de los usuarios. Es responsable de responder la información solicitada con la ayuda tanto del modelo como de la vista.

Los controladores pueden ser vistos como administradores cuidando de que todos los recursos necesarios para completar una tarea se deleguen a los trabajadores más adecuados. Espera peticiones de los clientes, comprueba su validez de acuerdo a las normas de autenticación o autorización, delega la búsqueda de datos al modelo y selecciona el tipo de respuesta más adecuado según las preferencias del cliente. Finalmente delega este proceso de presentación a la capa de la Vista.

