

**UNIVERSIDAD TECNOLÓGICA DE**

**SAN LUIS RIO COLORADO**

**MODELO-VISTA-CONTROLADOR**

**MTRO. SERGIO AMILLANO**

**ALUMNO: VICTOR MANUEL GALVAN COVARRUBIAS**

San Luis Rio Colorado, Sonora Octubre, 2020

Existen varios tipos de estructuras o modelos que pueden ser utilizados por los desarrolladores para la creación de software.

El modelo-vista-controlador es uno de los más populares gracias a que es muy fácil de manejar porque está dividido en tres secciones principales. Modelo, Vista y Controlador.

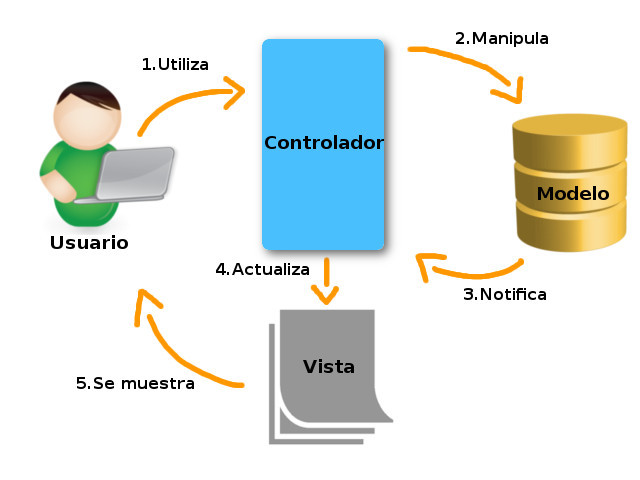
A continuación, pasaremos a analizar cada uno de estos elementos que conforman al MVC (modelo-vista-controlador) y también al final se presentaran algunas ventajas y desventajas de este.

Es importante mencionar que se deben tener conocimientos básicos de desarrollo web para entender la información que se describe a continuación.

¿Qué es el patrón MVC?

MVC es un patrón arquitectura, un modelo o guía que expresa cómo organizar y estructurar los componentes de un sistema software, sus responsabilidades y las relaciones existentes entre cada uno de ellos.

Su nombre, MVC, parte de las iniciales de Modelo-Vista-Controlador (Model-View-Controller, en inglés), que son las capas o grupos de componentes en los que organizaremos nuestras aplicaciones bajo este paradigma.



Es a menudo considerado también un patrón de diseño de la capa de presentación, pues define la forma en que se organizan los componentes de presentación en sistemas distribuidos.

La arquitectura MVC propone, independientemente de las tecnologías o entornos en los que se base el sistema a desarrollar, la separación de los componentes de una aplicación en tres grupos (o capas) principales: el modelo, la vista, y el controlador, y describe cómo se relacionarán entre ellos para mantener una estructura organizada, limpia y con un acoplamiento mínimo entre las distintas capas.

Modelo:

En la capa Modelo encontraremos siempre una representación de los datos del dominio, es decir, aquellas entidades que nos servirán para almacenar información del sistema que estamos desarrollando.

Asimismo, encontraremos la lógica de negocio de la aplicación, es decir, la implementación de las reglas, acciones y restricciones que nos permiten gestionar las entidades del dominio. Será por tanto el responsable de que el sistema se encuentre siempre en un estado consistente e íntegro.

Por último, el Modelo será también el encargado de gestionar el almacenamiento y recuperación de datos y entidades del dominio, es decir, incluirá mecanismos de persistencia o será capaz de interactuar con ellos.

La Vista:

Los componentes de la Vista son los responsables de generar la interfaz de nuestra aplicación, es decir, de componer las pantallas, páginas, o cualquier tipo de resultado utilizable por el usuario o cliente del sistema. De hecho, se dice que la Vista es una representación del estado del Modelo en un momento concreto y en el contexto de una acción determinada.

Cuando las vistas componen la interfaz de usuario de una aplicación, deberán contener los elementos de interacción que permitan al usuario enviar información e invocar acciones en el sistema, como botones, cuadros de edición o cualquier otro tipo de elemento, convenientemente adaptados a la tecnología del cliente.

En el caso de las aplicaciones para la Web, normalmente en la Vista se encontrarán los componentes capaces de generar el lenguaje de marcado de la página que será enviada al usuario.

Controlador:

La misión principal de los componentes incluidos en el Controlador es actuar como intermediarios entre el usuario y el sistema. Serán capaces de capturar las acciones de éste sobre la Vista, como puede ser la pulsación de un botón o la selección de una opción de menú, interpretarlas y actuar en función de ellas. Por ejemplo, retornando al usuario una nueva vista que represente el estado actual del sistema, o invocando a acciones definidas en el Modelo para consultar o actualizar información.

Realizarán también tareas de transformación de datos para hacer que los componentes de la Vista y el Modelo se entiendan. Así, traducirán la información enviada desde la interfaz. Y de la misma forma, el Controlador tomará la información procedente del Modelo y la adaptará a formatos o estructuras de datos que la Vista sea capaz de manejar.

En el Controlador se encuentran los componentes capaces de procesar las interacciones del usuario, consultar o actualizar el Modelo, y seleccionar las Vistas apropiadas en cada momento.

Ventajas y desventajas de patrón MVC

El uso del patrón MVC ofrece múltiples ventajas sobre otras maneras de desarrollar aplicaciones con interfaz de usuario, y en especial para la Web.

* La clara separación de responsabilidades impuesta por el uso del patrón MVC hace que los componentes de nuestras aplicaciones tengan sus misiones bien definidas. Por lo tanto, nuestros sistemas serán más limpios, simples, más fácilmente sostenible y, a la postre, más robustos.
* Mayor velocidad de desarrollo en equipo, que es consecuencia de lo anterior, ya que, al estar separado en tres partes tan diferenciadas, diferentes programadores pueden ocuparse de cada parte en paralelo. Esto la hace ideal para el desarrollo de aplicaciones grandes.
* Múltiples vistas a partir del mismo modelo, pudiendo reaprovechar mucho mejor los desarrollos y asegurando consistencia entre ellas. Facilidad para realización de pruebas unitarias.

Pero, por supuesto, no todo es siempre maravilloso, así que su uso presenta también algunas desventajas:

* Hay que ceñirse a las convenciones y al patrón. El uso de las convenciones impuestas por el framework y la estructura propuesta por el patrón arquitectural MVC nos obliga a ceñirnos a las mismas, lo que puede resultar a veces algo tedioso si lo comparamos con la forma habitual de trabajar con otros frameworks que dan más libertad al desarrollador. La división impuesta por el patrón MVC obliga a mantener un mayor número de archivos, incluso para tareas simples.
* Curva de aprendizaje. Dependiendo del punto de partida, el salto a MVC puede resultar un cambio radical y su adopción requerirá cierto esfuerzo. Además, utilizarlo implica conocer bien las tecnologías con las que se implemente.

El Modelo-Vista-Controlador será la estructura que se utilizará en este cuatrimestre por lo cual el necesario tenerla bien definida, para que después el desarrollo de aplicaciones web sea sencillo.

Para concluir acerca del Modelo-Vista-Controlador podríamos considerar el Controlador como un coordinador general del sistema, que regula la navegación y el flujo de información con el usuario, ejerciendo también como intermediario entre la capa de Vista y el Modelo. El Modelo contiene principalmente las entidades que representan el dominio, la lógica de negocio, y los mecanismos de persistencia de nuestro sistema. Y en la Vista encontraremos los componentes responsables de generar la interfaz con el exterior, por regla general, aunque no exclusivamente, el UI de nuestra aplicación.