

Tarea No. 4. MVC

Barrera Pérez Carlos Tonatihu
Profesor: José Asunción Enríquez Zárate
Web Application Development
Grupo: 3CM9

27 de enero de 2019

Índice

1. Introducción	3
2. Desarrollo	3
2.1. Modelo	4
2.2. Vista	5
2.3. Controlador	5
3. Conclusión	5

1. Introducción

El modelo vista controlador es una patron de arquitectura de software que se usa para implementar interfaces de usuario, por lo que es una buena opción en el desarrollo de aplicaciones web.

Separa la lógica de la aplicación en tres partes, promoviendo la modularidad, la facilidad de colaboración y la reutilización.[1]

2. Desarrollo

Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones y sobre multitud de lenguajes y plataformas de diseño [2].

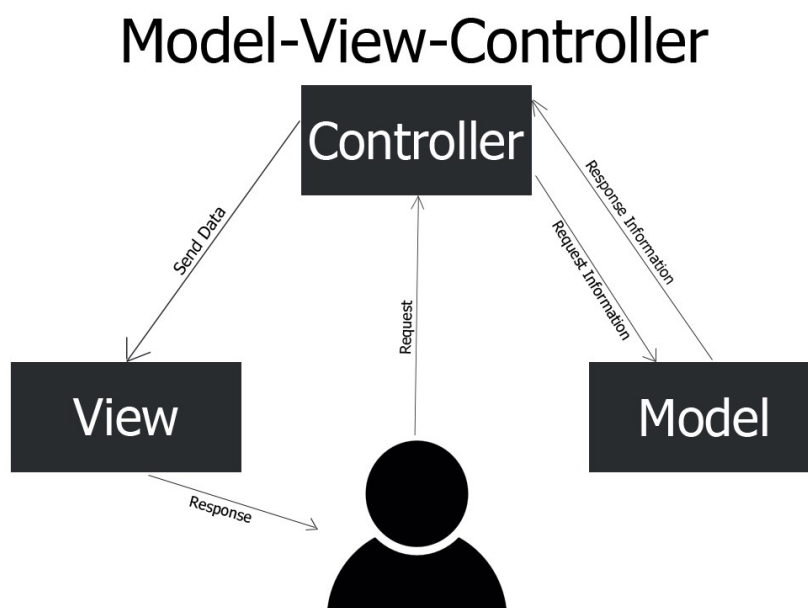


Figura 1: Flujo dentro del MVC

El flujo de control como se muestra en la figura 1 suele ser el siguiente:

1. El usuario interactúa con la interfaz de usuario de alguna forma (por ejemplo, el usuario pulsa un botón, enlace, etc.)
2. El controlador recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador

gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback.

3. El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario (por ejemplo, el controlador actualiza el carro de la compra del usuario). Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.
4. El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se refleja los cambios en el modelo (por ejemplo, produce un listado del contenido del carro de la compra). El modelo no debe tener conocimiento directo sobre la vista. Sin embargo, se podría utilizar el patrón Observador para proveer cierta indirección entre el modelo y la vista, permitiendo al modelo notificar a los interesados de cualquier cambio. Un objeto vista puede registrarse con el modelo y esperar a los cambios, pero aun así el modelo en sí mismo sigue sin saber nada de la vista. El controlador no pasa objetos de dominio (el modelo) a la vista aunque puede dar la orden a la vista para que se actualice. Nota: En algunas implementaciones la vista no tiene acceso directo al modelo, dejando que el controlador envíe los datos del modelo a la vista.
5. La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

2.1. Modelo

El Modelo que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia. El modelo es responsable de:

- Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
- Define las reglas de negocio (la funcionalidad del sistema). Un ejemplo de regla puede ser: "Si la mercancía pedida no está en el almacén, consultar el tiempo de entrega estándar del proveedor".
- Lleva un registro de las vistas y controladores del sistema.

- Si estamos ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo (por ejemplo, un fichero por lotes que actualiza los datos, un temporizador que desencadena una inserción, etc.).

2.2. Vista

La Vista, o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste. La vista es responsable de:

- Recibir datos del modelo y los muestra al usuario.
- Tienen un registro de su controlador asociado (normalmente porque además lo instancia).
- Pueden dar el servicio de `.Actualización()`, para que sea invocado por el controlador o por el modelo (cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes).

2.3. Controlador

El Controlador, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno. El controlador es responsable de:

- Recibe los eventos de entrada (un clic, un cambio en un campo de texto, etc.).
- Contiene reglas de gestión de eventos, del tipo "Si Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas.

3. Conclusión

Es un hecho que el Modelo Vista Controlador se ha vuelto todo un estándar en la industria, por lo cual se debe de entender el como se utiliza y con ello hacer un buen uso de este y acoplarlo de acuerdo a las necesidades que se presenten en el desarrollo del proyecto.

Referencias

- [1] L. Earl, “MVC architecture.” https://developer.mozilla.org/en-US/docs/Web/Apps/Fundamentals/Modern_web_app_architecture/MVC_architecture, 2018. [Online; consultado: 27-Enero-2019].
- [2] S. de Informatica, “Modelo vista controlador (MVC).” <https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html>, 2012. [Online; consultado: 27-Enero-2019].