Modelo-Vista-Controlador (MVC)

TUP – Programación II

UTN

Modelo-Vista-Controlador

- Este patrón fue descrito por primera vez por Trygve Reenskaug en 1979, y la implementación original fue realizada en Smalltalk en los laboratorios Xerox. Lo propuso como una forma de desarrollar el GUI de aplicaciones de escritorio.
- Hoy en día, el patrón **MVC** se utiliza para aplicaciones web modernas porque permite que la aplicación sea escalable, sustentable y fácil de expandir.
- MVC se basa en la separación de la aplicación en tres capas principales: Modelo, Vista y Controlador.
- Se usa (él o alguna de sus variantes) en la gran mayoría de las interfaces de usuario.

Modelo-Vista-Controlador

- **Modelo**: es la representación específica del dominio de la información sobre la cual funciona la aplicación.
- El modelo es otra forma de llamar a la capa de dominio.
- La lógica de dominio añade significado a los datos; por ejemplo, calculando si hoy es el cumpleaños del usuario o los totales, impuestos o resultados.

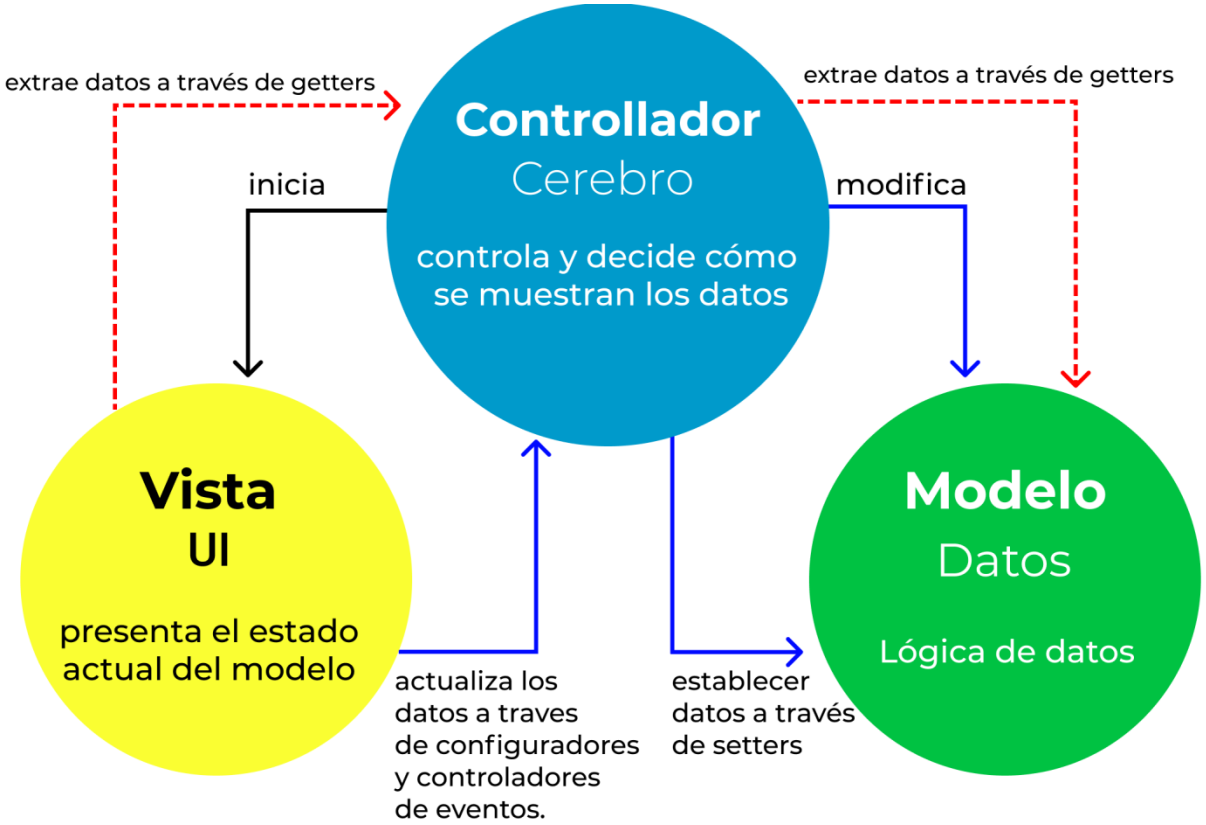
Modelo-Vista-Controlador

- **Vista**: Se presenta el modelo en un formato adecuado para interactuar, usualmente un elemento de interfaz de usuario.

- **Controlador**: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

Modelo-Vista-Controlador

En general



Modelo-Vista-Controlador

Modelo-Vista-Controlador

- Muchas aplicaciones utilizan un mecanismo de almacenamiento persistente (como puede ser una base de datos) para almacenar los datos. MVC no menciona específicamente esta capa de acceso a datos porque supone que está encapsulada por el modelo.
- El objetivo primordial del MVC es la reutilización del código ya implementado.
- Esta tarea se facilita mucho si a la hora de programar tenemos la precaución de separar el código en varias partes que sean susceptibles de ser reutilizadas sin modificaciones.

Modelo-Vista-Controlador

Ejemplos

- Los datos de una hoja de cálculo pueden mostrarse en formato tabular, con un gráfico de barras, con uno de sectores.

- Los datos son el modelo.
- Si cambia el modelo, las vistas deberían actualizarse en consonancia.
- El usuario manipula el modelo a través de las vistas. (en realidad, a través de los controladores).
- El Modelo es representado por el contenido actual, que usualmente se encuentra almacenado en una base de datos o en archivos XML.

Modelo-Vista-Controlador

Fortalezas

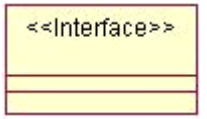
- Se presenta la misma información de distintas formas.
- Las vistas y comportamiento de una aplicación deben reflejar las manipulaciones de los datos de forma inmediata.
- Debería ser fácil cambiar la interfaz de usuario (incluso en tiempo de ejecución).
- Permitir diferentes estándares de interfaz de usuario o portarla a otros entornos no debería afectar al código de la aplicación.

Modelo-Vista-Controlador

- En UML

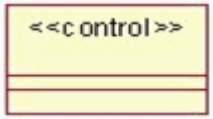
Se propone para el desarrollo del Modelo de Análisis de las aplicaciones, tres tipos de clases fundamentales, con las cuales podemos expresar todas las funciones de cualquier software, con sus respectivas responsabilidades

Clase Interfaz <<Interface>>:
Recepcionar peticiones al sistema.
Mostrar respuestas del sistema.



Clase Entidad <<Entity>>:
Gestionar datos (información) necesaria para el sistema.
Almacenar datos (información) persistentes del sistema.
Provee la funcionalidad principal de la aplicación

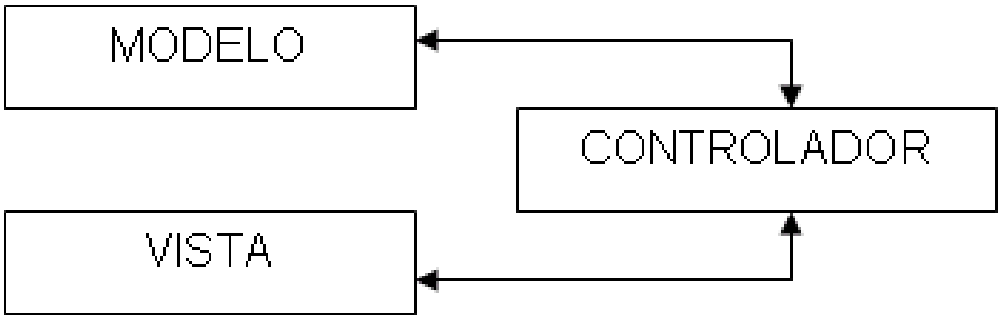
Clase Controlador <<Controller>>:
Procesar Información del sistema.
Gestionar visualización de respuesta del sistema.
Obtiene los datos del modelo.



Modelo-Vista-Controlador

Variantes del Modelo.

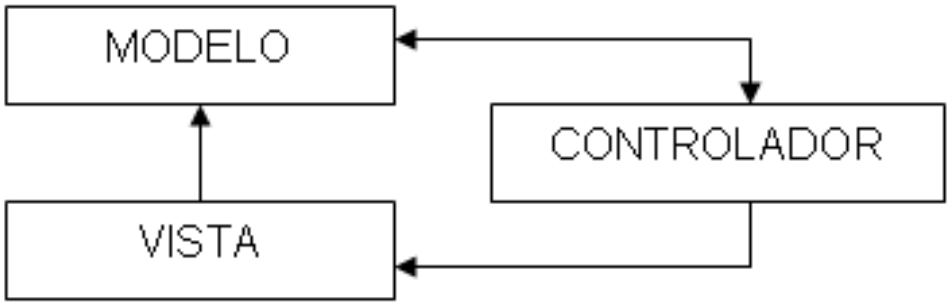
- Variante en la cual no existe ninguna comunicación entre el Modelo y la Vista y esta última recibe los datos a mostrar a través del Controlador.



- Variante inicial del Patrón MVC.

Modelo-Vista-Controlador

- Variante en la cual se desarrolla una comunicación entre el Modelo y la Vista, donde esta última al mostrar los datos los busca directamente en el Modelo, dada una indicación del Controlador, disminuyendo el conjunto de responsabilidades de este último.



Variante Intermedia del Patrón MVC.