Texto, Carta

Descripción generada automáticamente

FACULTAD: ECONOMÍA, EMPRESA Y NEGOCIOS.

ASIGNATURA: PROGRAMACIÓN DE COMPUTADORAS 4

DOCENTE: ALEXANDER ALBERTO SIGÜENZA

ACTIVIDAD: INVESTIGACIÓN PATRÓN DE ARQUITECTURA MVC PARA JAVA

ESTUDIANTE:

JOSÉ ALEJANDRO BAUTISTA MIRÓN 202001096

ANTIGUA CUSCATLÁN, SÁBADO 03 DE JUNIO DEL 2023

Índice

[**INTRODUCCIÓN:** 3](#_Toc136719451)

[**Patrón de arquitectura MVC en Java con Netbeans** 4](#_Toc136719452)

[**El patrón MVC es utilizado en múltiples frameworks:** 5](#_Toc136719453)

[**¿Por qué utilizar MVC?** 5](#_Toc136719454)

[**Ejemplo patrón MVC** 6](#_Toc136719455)

[**Conclusiones** 8](#_Toc136719456)

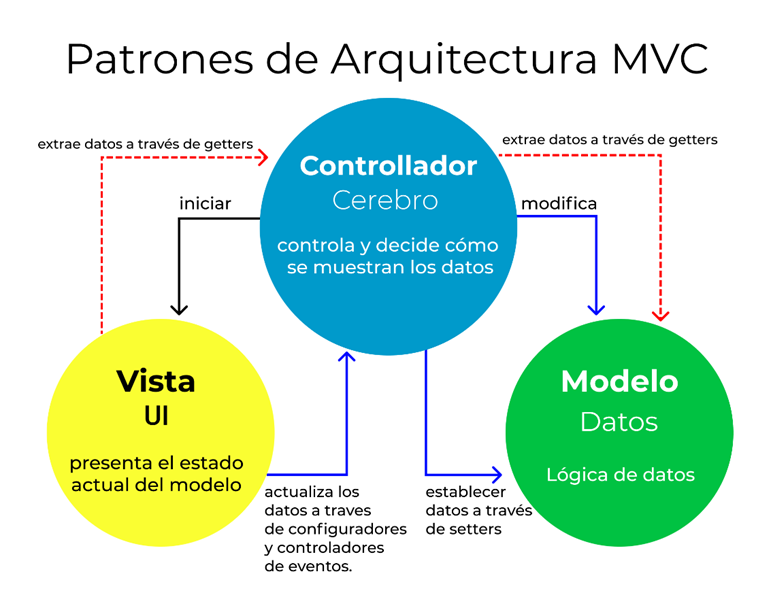
[**Bibliografía** 9](#_Toc136719457)

## **INTRODUCCIÓN:**

El siguiente trabajo de investigación es acerca de El patrón de arquitectura de MVC para el lenguaje de programación Java utilizando el IDE Netbeans, esta investigación se enfoca en estudiar el patrón MVC y de manera general los conceptos de mvc y cada una de sus partes, añadiendo que esta investigación está dedicada específicamente con el lenguaje de Java ya que es el lenguaje de estudio de la asignatura a la que corresponde este trabajo de investigación y el estudio de este concepto es importante para aportar y aumentar el conocimiento y para el desarrollo de las habilidades que permitan un mejor desempeño en el mundo laboral ya como profesional, para finalizar la investigación contiene un ejemplo desarrollado con el objeto de estudio junto a una conclusión del trabajo desarrollado.

## **Patrón de arquitectura MVC en Java con Netbeans**

Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz que ve el usuario de la lógica del negocio en tres componentes distintos. Su utilización se encuentra mas en las aplicaciones Web que en las aplicaciones para escritorio, sin embargo, es posible hacerlo para este tipo de aplicaciones, debido a que Java ya cuenta con Observer y Observable, son herramientas que ayudan a la interacción entre la interfaz y el modelo, sin embargo, el ejemplo que dejamos a continuación no hace uso de estas herramientas. El patrón se basa en separar la lógica de negocio de la interfaz de usuario lo que facilita la evolución por separado de ambos aspectos y además de eso incrementa reutilización y flexibilidad.

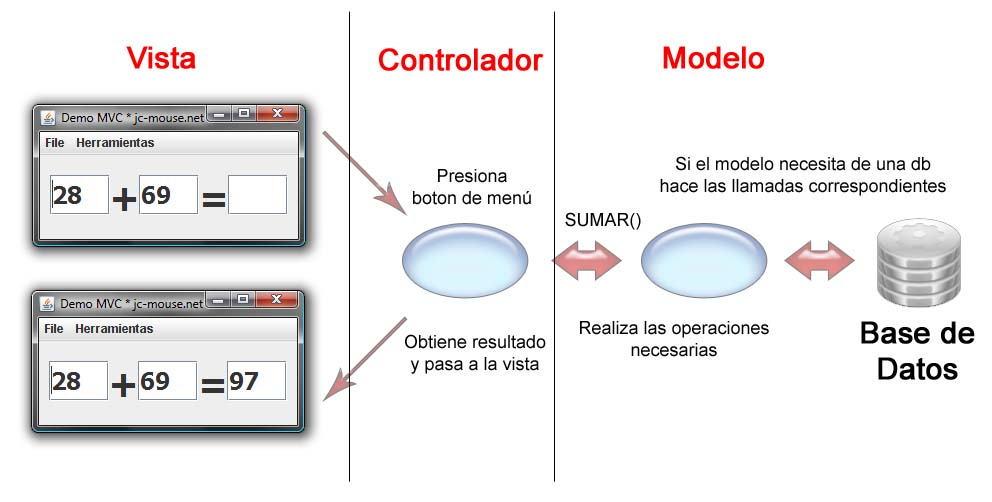


Las siglas del patrón MVC están descritas como:

Vista (view): Representa la interfaz de usuario y todas las herramientas con las que el usuario hace uso del programa.

Modelo (Model): Es donde esta toda la lógica del negocio, la representación de todo el sistema incluido la interacción con una base de datos, si es que el programa así lo requiere.

Controlador (Controller): Este componente es el que responde a la interacción(eventos) que hace el usuario en la interfaz y realiza las peticiones al modelo para pasar estos a la vista. Aun no esta del todo claro, pero con el siguiente ejemplo, lo comprenderemos mejor.



Este concepto fue introducido por primera vez por Trygve Reenskaug, que lo propuso como una manera de desarrollar el GUI de aplicaciones de escritorio.

Por lo tanto, ahora en día este patrón se utiliza para aplicaciones web modernas porque permite que las aplicaciones sean escalables permitiendo ser más fácil de expandir y que sea mejor para su mantenimiento.

## **El patrón MVC es utilizado en múltiples frameworks:**

• Java Swing

• Java Enterprise Edition (J2EE)

• XForms (Formato XML estándar del W3C para la especificación

de un modelo de proceso de datos XML e interfaces de usuario

como formularios web)

• GTK+ (escrito en C, toolkit creado por Gnome para construir

aplicaciones gráficas, inicialmente para el sistema X Window)

• ASP.NET MVC Framework (Microsoft)

• Google Web Toolkit (GWT, para crear aplicaciones Ajax con Java)

• Apache Struts (framework para aplicaciones web J2EE)

• Ruby on Rails (framework para aplicaciones web con Ruby)

• Etc., etc., etc.

## **¿Por qué utilizar MVC?**

La respuesta seria porque el patrón MVC ayuda a dividir el código frontend y el backend en componentes separados. De esta manera siendo más fácil de administrar y para realizarle cambios sin que algo pudiera interferir entre sí.

Aunque en la practica real suele ser mas complicado por la gran cantidad de desabolladores que puedan estar trabajando en el mismo sistema.

## **Ejemplo patrón MVC**

Creamos un nuevo proyecto y creamos 3 Estructuras de paquetes con los nombres Controller, Model, View.

A screenshot of a computer

Description automatically generated with medium confidence

Creamos una clase llamada Modelo en la carpeta de paquetes Model y en esa misma carpeta creamos una clase llamada Estudiantes donde agregaremos los métodos.

A screenshot of a computer program

Description automatically generated with medium confidence

Creamos una Jframe llamado vistas en el paquete view

A screenshot of a computer

Description automatically generated with medium confidence

Agregamos la variable cargar Datos de la clase Estudiante

A screenshot of a computer program

Description automatically generated with medium confidence

En Controlador agregamos una variable para inicializar Vistas.

A screenshot of a computer

Description automatically generated

Para el archivo EjemploMCV Java cargamos la variable main con todas las clases creadas anteriormente

A screenshot of a computer

Description automatically generated with medium confidence

Como podemos ver el Controlador muestra el formulario de vistas y el formulario contiene lo que pertenece a la clase Estudiantes que esta a su vez esta conectada con la base de datos de MySQL

## **Conclusiones**

La utilización del patrón MVC es ideal para cualquier desarrollador de aplicaciones ya que permite mantener un orden y además permite poder trabajar separando las clases de sus formularios y controles, así mantienen un mejor control a la hora de detectar problemas en la aplicación por lo que es recomendable implementarlo para todo tipo de proyectos, y algo que aporta en gran medida es que el patrón MVC puede ser implementado en cualquier lenguaje de programación.

## **Bibliografía**

<https://www.fdi.ucm.es/profesor/jpavon/poo/2.14.mvc.pdf>

<https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/>

<https://code.google.com/archive/p/gestion-matricula/wikis/MVC.wiki#:~:text=El%20patr%C3%B3n%20MVC%20(Modelo%2C%20Vista,negocio%20en%20tres%20componentes%20distintos>.