MIND GAME

DOCUMENTO DE ARQUITECTURA DEL SISTEMA

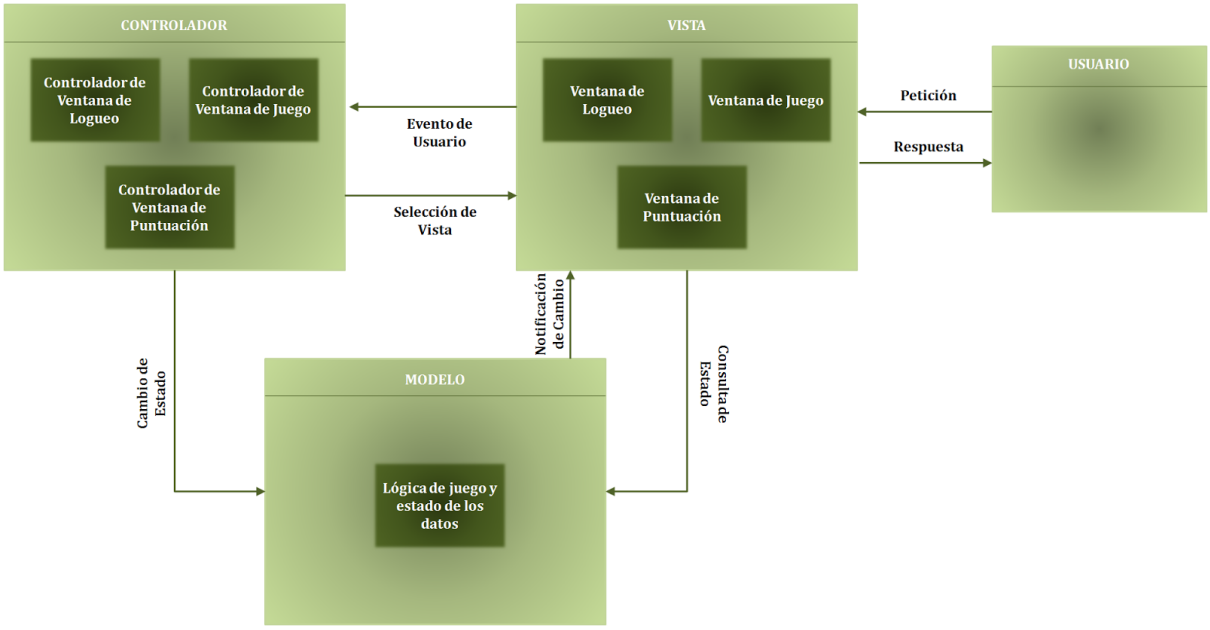
Autores: Sosa Ludueña Diego
Sleiman Mohamad
Choquevilca Gustavo

Versión del Documento: 1.0.0

Índice

1. Gráfico de Arquitectura General

A Continuación mostraremos un gráfico de arquitectura general, en el cual se visualizarán los componentes y las relaciones que existen en el sistema.



Como vemos en el gráfico, el sistema se estructura en 3 componentes lógicos que interactúan y se relacionan entre sí. Por un lado tenemos el componente MODELO que maneja los datos del sistema y las operaciones asociadas a esos datos. Otro componente importante es la VISTA que define y gestiona como se presentan los datos al USUARIO. Y por último el componente CONTROLADOR que dirige la interacción del USUARIO (que en nuestro caso son los "clics del mouse") y pasa estas interacciones a la VISTA y al MODELO.

2. Patrón de Arquitectura MVC

Sabiendo que un patrón arquitectónico describe una organización de sistema que ha tenido éxito en sistemas previos. Por lo tanto en nuestro sistema Mind Game utilizamos el patrón de arquitectura Modelo-Vista-Controlador (MVC), debido a que en general la realización de la interfaz de usuario de una aplicación resulta un problema complejo. La principal característica del diseño de la interfaz de usuario es que debe ser lo suficientemente flexible para dar respuesta a las siguientes exigencias:

- Los usuarios de esta aplicación pueden solicitar cambios a dicha interfaz para que sea más eficaz o fácil de usar.
- La aplicación puede ofrecer nuevas funcionalidades, lo cual requiere una actualización de su interfaz de usuario.
- El sistema de ventanas de la plataforma con el que trabaja la aplicación puede evolucionar e imponer modificaciones en la interfaz de usuario.
- La misma información puede representarse mediante diferentes vistas e introducirse a través de distintos medios.
- La representación debe reflejar, inmediatamente, las modificaciones de datos

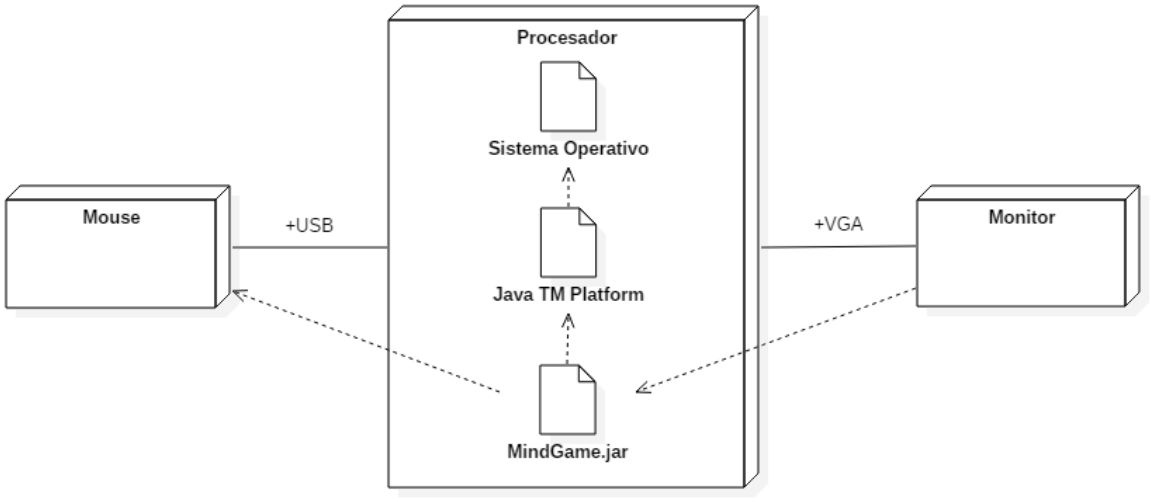
manipulados por la aplicación.

- Los datos gestionados por la aplicación pueden manipularse simultáneamente a través de varias interfaces.

Debido a estos requisitos, se adopta una solución algo más modular como la que propone el patrón de arquitectura MVC.

3. Diagrama de Despliegue

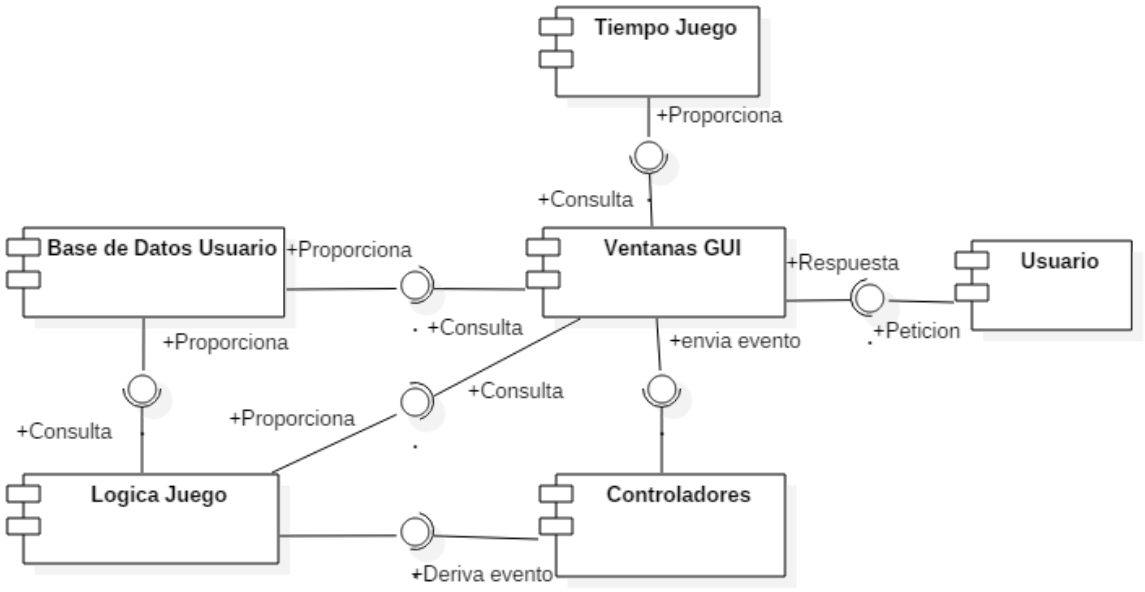
Este diagrama de despliegue describe la distribución de los artefactos de software en el sistema.



Como se observa en el diagrama se tiene tres nodos *Mouse, Monitor y Procesador*. Dentro del nodo *Procesador* tendremos los artefactos *Sistema Operativo, Java(TM) Platform y MindGame.jar*. Y por último tendremos la comunicación entre los nodos o interfaces que son *USB y VGA*.

4. Diagrama de Componentes

Este diagrama representa las relaciones entre componentes de un sistema y sus interfaces.



Como se observa en el diagrama se tienen los componentes que son *Base de Datos Usuario, Ventana (GUI), Usuario, Controladores, Tiempo Juego y Lógica Juego.* Por otro lado tenemos a las interfaces para mostrar cómo se relacionan los componentes entre sí.

5. Pruebas de Integración

En esta apartado, se generaron en total 4 test de integración que se realizaron también como en el apartado anterior, con la herramienta *JUnit Test Case* provista por la *IDE Eclipse.* Estos test de integración son:

5.1. TestTiempo

En este test de integración se prueba por medio de la interacción de dos clases (modelo y controladorLogueo) que la variable de estado de puntuación vaya progresando hasta su último estado.

5.2. TestEstadoLogueo

En este test de integración se prueba que por medio de la interacción de dos clases (modelo y controladorLogueo) la variable de estado de logueo vaya progresando hasta su último estado.

5.3. TestEstadoJuego

En este test de integración como en el caso anterior se prueba por medio de la interacción de tres clases (modelo, controladorLogueo y controladorJuego) que la variable de estado de juego vaya progresando hasta su último estado.

5.4. TestEstadoPuntuacion

En este test de integración como en el caso anterior se prueba por medio de la interacción de cuatro clases (modelo, controladorLogueo, controladorJuego y controladorpuntuacion) que la variable de estado de puntuación vaya progresando hasta su último estado.

5.5. AllTests

Como en el caso de los test unitarios se tuvo en cuenta utilizar esta test (*AllTests*) para correr todas las pruebas.