Perseverance

Solitario Apolo

Arquitectura del Sistema

Autores:

- ➢ Espósito, Javier
- ➢ Sartori, Gastón Emilio
- ➢ Tántera, Julián
- ➢ Zanotti, Emiliano

Fecha: 4 de Junio de 2021

Versión: 1.0.1

Historial de Cambios

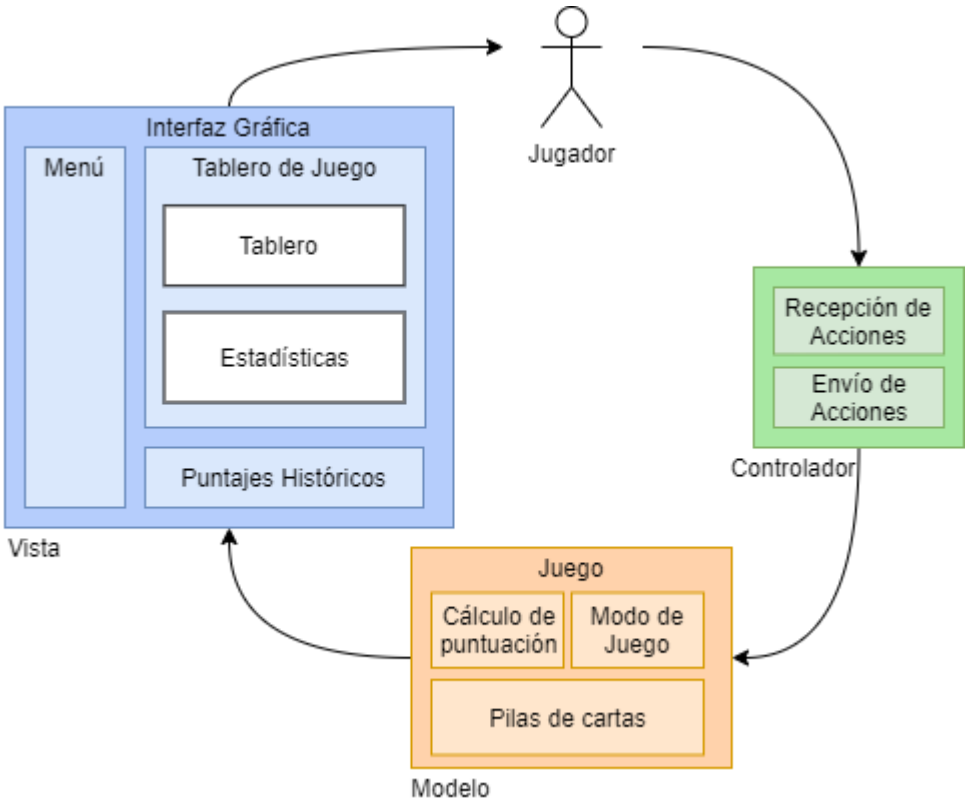
Versión	Fecha	Cambios
1.0.0	4 / 6 / 2021	Documento inicial
1.0.1	13 / 6 / 2021	Cambios en sección 2
1.0.2	17 / 6 / 2021	Cambios en sección 2.1

1. Introducción

A lo largo de este documento, se determinará en detalle la implementación del sistema, describiendo qué patrón de arquitectura se utilizará, junto con los componentes que lo integran. Además se definirán aquellos casos de pruebas de integración que verifican la correcta interacción entre los componentes del sistema.

2. Patrón de Arquitectura

A continuación se muestra el diagrama de arquitectura del sistema, el cual describe un patrón de arquitectura MVC (Model, View, Controller). Luego se describirán con mayor detalle, cómo se aplica este patrón en nuestro sistema.



Para este proyecto, utilizaremos el patrón de arquitectura MVC. En el diagrama anterior, se pueden observar los módulos que componen este patrón, el modelo, la vista y el controlador y las relaciones entre los mismos.

Las ventajas que nos ofrece la aplicación de este patrón, es poder separar la lógica de la aplicación y los datos, de la visualización de los mismos. Esta separación en módulos, nos permite un mayor desacople entre los componentes del sistema, y por consiguiente, una mayor mantenibilidad del mismo, permitiendo realizar cambios en cada uno de los módulos, sin necesidad de modificar los otros. Además, nos da la posibilidad de mostrar los mismos datos de diferentes maneras de forma sincronizada.

Se eligió este patrón, ya que además, nos permite cumplir con los requerimientos no funcionales del proyecto. En primera instancia, necesitamos una forma de comunicarnos con el usuario de manera clara para que al momento de iniciar una partida se haga de manera sencilla, para cumplir con el requerimiento RN1, por lo que la utilización de una interfaz gráfica (Vista) nos permite realizarlo.

Luego necesitamos un módulo en el cual se desarrolle la partida y contenga la información de la misma en todo momento, por lo que el módulo de Juego (Modelo), nos brinda las características necesarias para llevarlo a cabo de manera correcta. En este módulo se realizan todas las acciones necesarias para dar comienzo a la partida, por lo que unificarlas, produce que estas se hagan de manera rápida, en pos de cumplir el requerimiento RN3.

Finalmente, necesitamos un modelo para conectar Vista y Modelo, el cual será el Controlador. Este recibe información de la Vista, y se la comunica al Modelo, por lo que unifica estas tareas en un único módulo logrando un bajo tiempo de respuesta a las acciones realizadas por el jugador, lo que permite cumplir con el requerimiento RN2.

La aplicación de este patrón a nuestro proyecto en particular, se da de la siguiente manera:

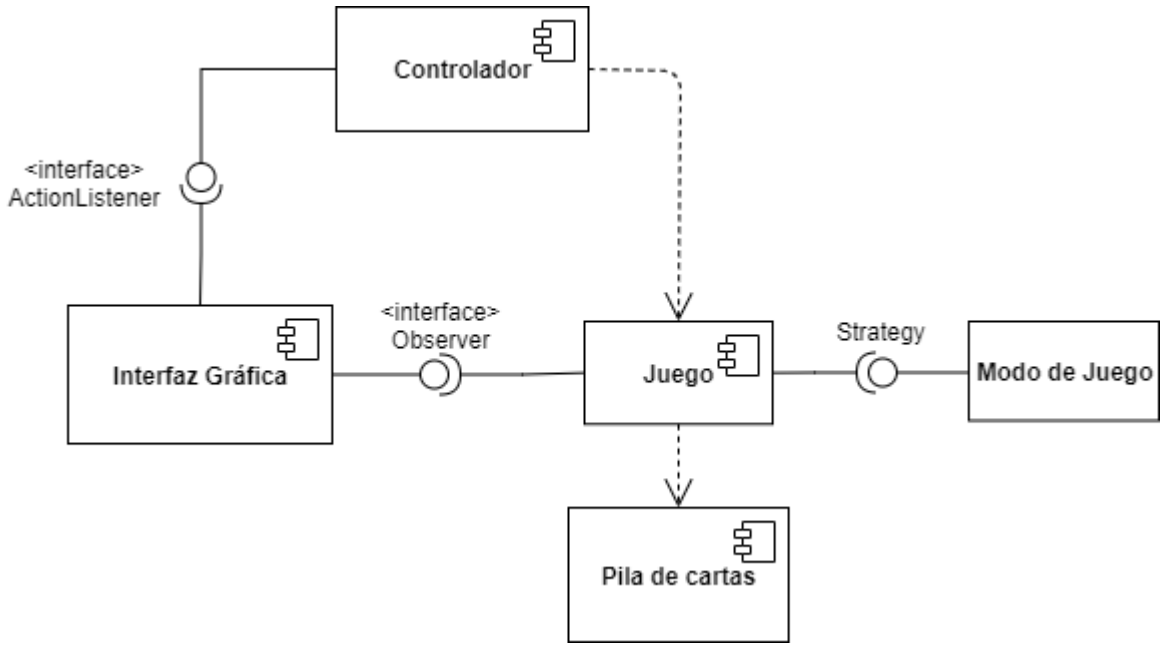
El jugador realiza una interacción con el sistema, la cual es recibida por el controlador. Esta interacción se realiza a través de clicks en las cartas y pilas de cartas para determinar qué movimiento quiere realizar. A partir de esta interacción recibida por el controlador, éste se comunica con el modelo, el cual controla que se respeten las reglas del juego, determinando si el movimiento pretendido por el jugador es válido o no, realiza los cálculos de la puntuación y lleva el control de las pilas de juego y su estado a través de la partida.

El modelo actualiza la vista, para dar información al jugador del estado actual de la partida y que en base a esto, él tome las decisiones correspondientes.

En complemento de este patrón de arquitectura, se harán uso de los patrones de diseño Observer y Strategy. Cuya aplicación se explicará con mayor detalle en el Documento de Diseño.

Diagrama de componentes

A continuación se muestra el diagrama de componentes:



En este diagrama podemos observar los diferentes componentes que integran nuestro proyecto. Podemos observar los tres componentes básicos del patrón de arquitectura MVC, en donde el Juego es el Modelo, la interfaz gráfica es la Vista y el Controlador es el Controlador. El componente Juego, hace uso de un componente nombrado como Pila de cartas, el cual representa a las pilas de juego y de las pilas de escalera que componen nuestra partida.

En el Juego debe establecerse el Modo de Juego que se quiera, lo cual se hace a través del patrón de diseño Strategy. Además el Juego, se comunica con la Interfaz Gráfica, a través de la aplicación de interfaces del patrón Observer. La comunicación entre el Controlador y la Interfaz Gráfica, se realiza a través de la interfaz ActionListener de Java, en la cual el Controlador espera a las acciones realizadas por el jugador en la Interfaz Gráfica y las comunica al Juego.

2.1 Pruebas de Integración

A continuación, se describirán test ideados para ser corridos de manera automática.

Test case ID: TI 001

Descripción: Se probará el funcionamiento de Estadísticas, quien implementa la interfaz Observer y cuyos valores se verán modificados por los movimientos del Juego.

Función a testear: Integración de Estadísticas (Observador) con Juego (sujeto).

Preparaciones previas: N/A

Ejecución del test (pasos):

1. Crear un objeto Partida
2. Registrar un objeto Estadísticas como observer del juego
3. Modificar el puntaje del jugador al ubicar una carta en zona de escaleras
4. Dejar que pasen 5 minutos de juego.

Resultado esperado: El puntaje mostrado en la ventana de Estadísticas, en el apartado Mov. a zona de escaleras se deberá ver aumentado en 100 puntos. Además el puntaje por tiempo disminuirá 2400 puntos.

Test case ID: TI 002

Descripción: Se probará el funcionamiento de la puntuación, tiempo jugado y movimientos exitosos, mostrados en el Tablero del juego, quien implementa la interfaz Observer y cuyos valores se verán modificados por los movimientos del Juego y el paso del tiempo.

Función a testear: Integración de Tablero (Observador) con Juego (sujeto)

Preparaciones previas: N/A

Ejecución del test (pasos):

1. Crear un objeto Juego.
2. Registrar un objeto Tablero como observer del juego
3. Realizar algunos movimientos exitosos y ubicar una carta en zona de escaleras

Resultado esperado: El tiempo de juego se actualizará de manera correcta y constante, se contarán correctamente los movimientos exitosos y la puntuación subirá en 100 puntos por carta ubicada en zona de escaleras**.**

Test case ID: TI 003

Descripción: Se probará si al iniciar la partida se reparten las cartas del mazo en el tablero, y si el jugador puede interactuar con las cartas.

Función a testear: Integración del Mazo, Tablero, Pilas de cartas y el Jugador

Preparaciones previas: N/A

Ejecución del test (pasos):
1. Crear un objeto Mazo
2. Crear un objeto Tablero
3. Repartir cartas
4. Comprobar distribución de las cartas
5. Sacar una carta del mazo

Resultado esperado: En el tablero se deberá contar con 7 pilas de cartas distribuidas como se especifica en el Documento de Requerimientos, con las correspondientes cartas boca arriba y abajo. El mazo restante deberá mostrarse en la zona del Mazo, y al clickear sobre el mismo, se deberá sacar una carta y colocarse en la zona de cartas a ubicar.

Test case ID: TI 004

Descripción: Se probará si funciona la integración del modo de juego de la partida (1 o 3 cartas). El modo de juego se implementa con el patrón de diseño Strategy y se puede cambiar dinámicamente durante la partida.

Función a testear: Integración de partida con elección del modo de juego. (Strategy)

Preparaciones previas: N/A

Ejecución del test (pasos):
1. Crear un objeto Juego
2. El modo de juego por defecto es Una Carta
3. Pedir cartas del mazo
4. Cambiar modo de juego a Tres Cartas
5. Volver a pedir carta del mazo

Resultado esperado: Al pedir cartas por primera vez, el mazo debió disminuir en uno su cantidad de cartas y aumentar en uno la zona de cartas de ubicar. Al pedir cartas por segunda vez, el mazo debió disminuir en tres su cantidad de cartas y aumentar en tres la zona de cartas de ubicar.
