

**Alumnos—**

Itzel Haydeé Garza López

Ian Salas López

Jesús Adrián Félix Álvarez

Hiram Rodriguez Ruiz

**IDs—**

00000216628

00000216755

00000204255

00000194923

**Asignación—**

Proyecto Final - Timbiriche

**Fecha—**

13 de diciembre de 2021

**Materia—**

Arquitectura de Software

**Profesor—**

Gilberto Borrego Soto

Índice

1. Explicación breve del juego
2. Explicación del estilo arquitectónico
3. Diagramas de subsistemas
   1. Dominio
   2. Presentación
4. Comunicación de subsistemas
5. Diagrama de componentes
6. Diagrama de despliegue

* Explicación breve del juego.

Se juega sobre una hoja de papel, a modo de tablero, por dos o más jugadores; quizás hasta cuatro, dependiendo de las características del tablero ya preparado como una matriz cuadriculada; cada jugador dibuja un punto por cada esquina de lo que será un cuadro. Los puntos en forma de cuadrícula deben ser, proporcional a la cantidad de participantes, suficientes que permita a todos jugar para hacerlo divertido, pues el objetivo es el de completar cuadros, y así reclamar la mayor cantidad de estos posibles sobre el tablero. Así que en cada jugada, de forma alternada, un jugador unirá dos puntos consecutivos horizontal o verticalmente; los cuadros se van formando lado por lado, línea por línea, y las líneas diagonales no son permitidas en el juego.

Cuando un jugador forma con estas líneas un cuadrado, se anota este, escribiendo una inicial en el centro o marcándose con el color del jugador. Después de formar un cuadrado deberá dibujar una línea más, cerrando todos los cuadros que el tablero permita. El jugador que se haga con más cuadrados gana la partida.

* Explicación del estilo arquitectónico usado

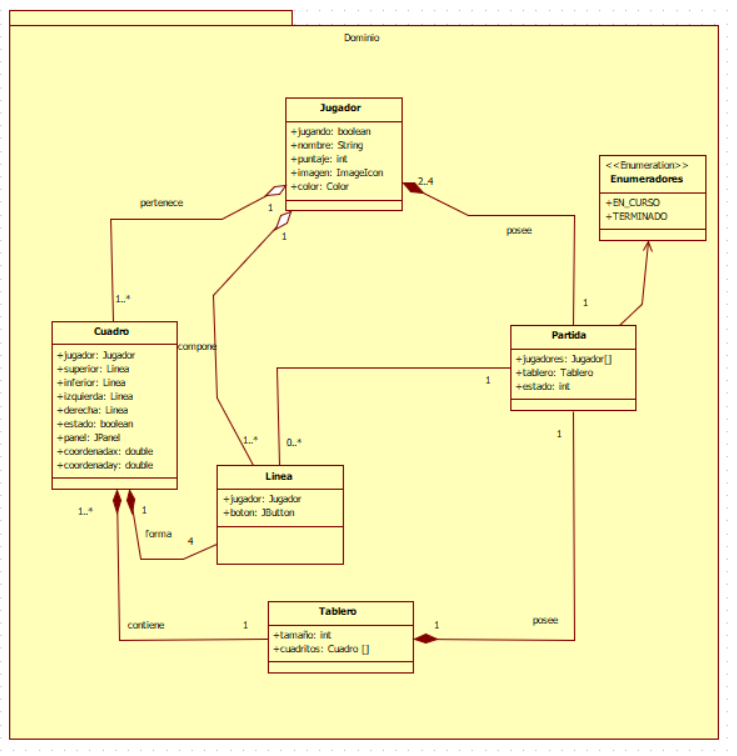
**Event Driven:**

La arquitectura de eventos es una arquitectura distribuida asincrónica, diseñada para crear aplicaciones altamente escalables.

Los componentes no se comunican de la forma tradicional, la comunicación se establece de forma sincrónica, recibe una respuesta y da el siguiente paso. En esta arquitectura, se espera que las aplicaciones provoquen diferentes "eventos" para que generen cambios dentro del sistema.

Este tipo de programación brinda a los usuarios respuestas rápidas y completas a lo que buscan, lo que se traduce en una mejor experiencia para ellos. Al realizar un seguimiento de lo que hacen los usuarios y responder a sus interacciones, se crea un sentido de comprensión entre los dos y esto beneficia a la empresa.

**Subsistema Dominio**



Este subsistema permite establecer la relación que habrá entre los elementos que conforman el sistema del juego.

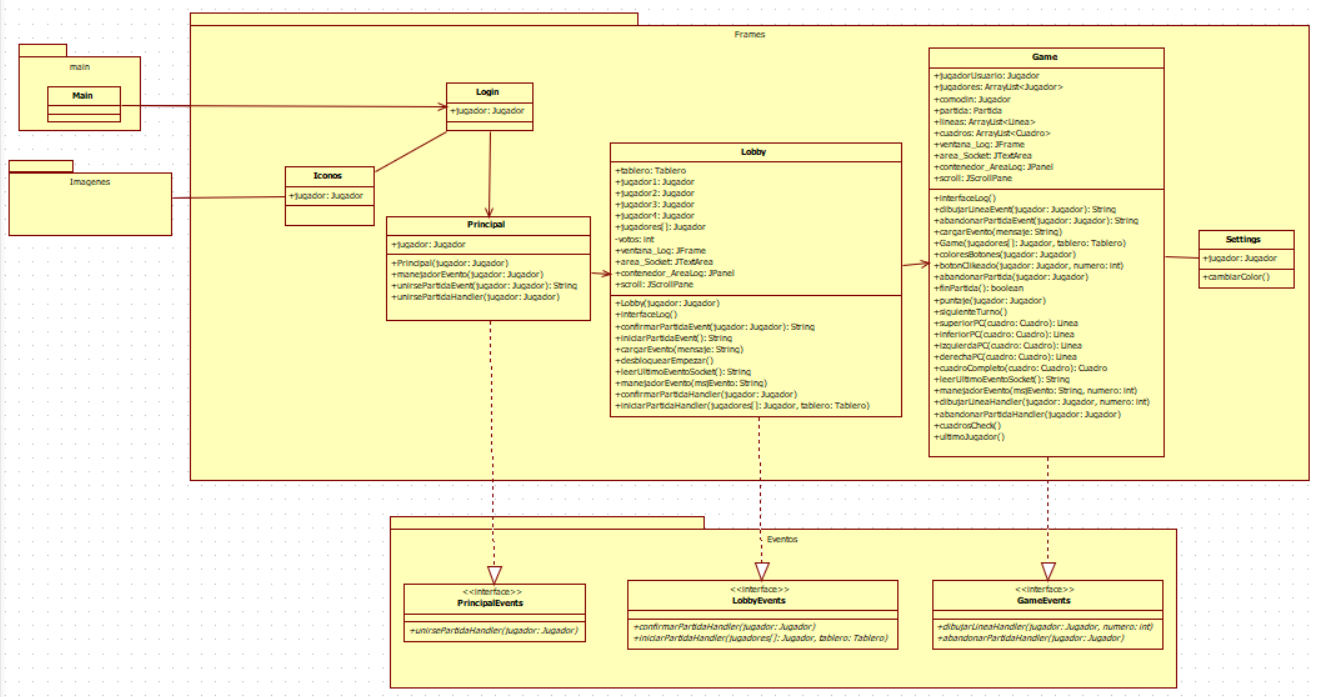
**“Jugador”** estará conformado por *jugando*, el cual es el estado en el que se encuentra (true cuando esté jugando y false cuando aún no empiece la partida o cuando ésta haya finalizado). También cuenta con nombre, el cual es el que el usuario haya escrito antes de iniciar la partida, su respectiva imagen, color y el puntaje que irá cambiando a medida que la partida avanza.

**“Partida”** a su vez, estará conformada por la lista de jugadores que se encuentren en ésta, un tablero, donde se están almacenando los valores de los cuadros y el tamaño con el que cuenta el tablero.

**“Línea”** tiene un botón, el cuál representará el trazo de la línea y un Jugador, que será quien haya seleccionado el botón, para así obtener el color representativo de dicho jugador y así colorear el botón del color correspondiente.

Cuadro cuenta con un Jugador, que será quien haga la última línea y por ende sea el dueño de dicho cuadro. Cuenta además con cuatro objetos de tipo Línea, que como se mencionó antes, incluyen el botón que representa el trazo de la línea. Tiene un estado, para saber cuándo se haya completado y así el panel sea coloreado con el color del Jugador que lo haya hecho. A su vez tiene coordenadas x y y, que indican la posición del panel que se coloreará y a su vez, funcionan para encontrar las líneas que se encuentran alrededor de dicho panel.

**Subsistema Presentación**

****

Este subsistema permite que el usuario pueda navegar por las vistas requiriendo la información necesaria para su funcionamiento. Además de poder con cada pantalla mostrada, provocar eventos y obtener eventos ya provocados por otros jugadores para su actualización cada vez que estos ocurran.

Clase “**Iconos**”: Será una clase que tenga funcionalidad de un frame para la vista del usuario, en donde el usuario pueda escoger el icono que deseé para su avatar dentro del juego.

Clase “**Login**”: Será una clase que tenga funcionalidad de un frame para la vista del usuario, y tendrá la funcionalidad de poder hacer Log In dentro del sistema para poder participar en el juego.

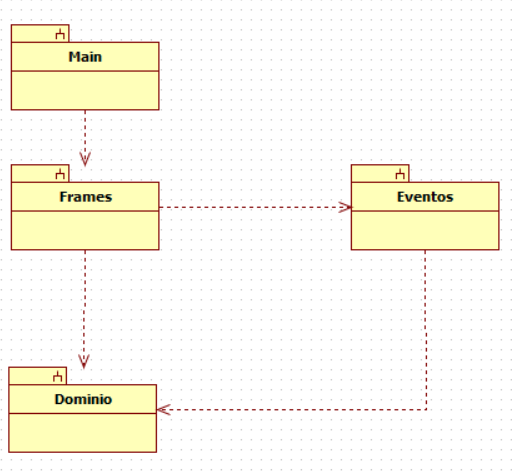
Clase “**Principal**”: Será una clase que tenga funcionalidad de un frame para la vista del usuario, en donde su función principal sea escoger unirse a una partida mediante el evento **“Unirse Partida Event”**, que ocasionará que a todos los jugadores(si se introduce los valores correctos) puedan ver al jugador que se unió en la siguiente pantalla.

Clase “**Lobby**”: Será una clase que tenga funcionalidad de un frame para la vista del usuario, en donde se mostrarán todos los jugadores unidos al lobby y donde se podrá saber quienes de ellos están listos para empezar el juego, además de sus nombres e iconos. Este frame se verá afectado cuando un jugador quiera empezar la partida, se mostrará en pantalla una señal de confirmación de parte del jugador mismo para que todos los otros jugadores se enteren que está listo para empezar, aquí intervendrá el evento **“Confirmar Partida Event”**. También se reproducirá por el administrador el evento “**Iniciar Partida Event**” lo que provocará a todos los jugadores que se cambie de pestaña a la siguiente, la cual será la pestaña **“Game”**.

Clase “**Game**”: Será una clase que tenga funcionalidad de un frame para la vista del usuario, en donde empezará el juego con sus respectivos turnos, colores, nombres y el tablero donde cada quién haya puesto su línea se verá reflejado en esta clase. En este frame también actuará el evento de “**Dibujar Linea Event**”, el cuál estará constantemente actualizándose para verificar quienes han dibujado una línea. Y por último si un jugador se llegara a salir de la partida, entraría el evento “**Abandonar Partida Event**”, el cual notificará a todos que un jugador ha desertado.

Clase “**Setting**”: Será una clase que tenga funcionalidad de un frame para la vista del usuario, en donde se podrán cambiar los colores de los jugadores en la perspectiva propia de cada jugador durante el juego.

* Diagrama que indique cómo se comunican los subsistemas, en donde se pueden agrupar en paquetes para indicar cuales subsistemas se agrupan en componentes.



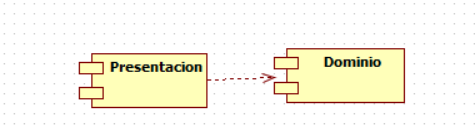
El subsistema “**Dominio**” participa en el subsistema “**Frames**” y **“Eventos”** brindándoles la información de cada elemento.

El subsistema **“Main”** se encarga de ejecutar el cliente para presentar la primera página de frames.

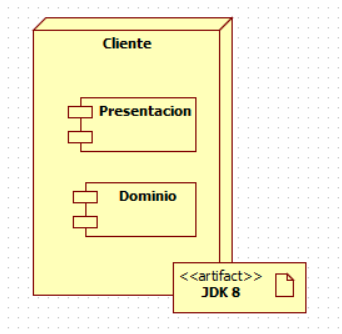
El subsistema **“Eventos”** brinda interfaces para que los frames puedan obtener los métodos de los eventos a manejar.

El subsistema **“Frames”** se encarga de mostrar al usuario la interfaz, y operar los eventos dentro del mismo para que se cumpla con el funcionamiento de la aplicación y arquitectura.

* Diagrama de componentes donde con notas se puede definir qué subsistemas están dentro de cada componente.



* **Diagrama de despliegue**, es donde se explica el porqué cada nodo contiene los componentes y artefactos que se muestran en el diagrama.



Nodo **“Cliente”**  representa la computadora personal en la que se estará corriendo el programa, junto con el artefacto JDK 8 que será necesario para correr el sistema. dentro de el nodo, podemos ver los 2 componentes que se usarán para el desarrollo y el buen funcionamiento del programa.