TEMA: POOL
Proyecto Final

Asignatura:	Programación 2.
Grupo:	N°14.
Integrantes:	- Francy Pilar Jelvez Jen. - Diego Joaquín Andrés Venegas Anabalón.
Profesor:	Geoffrey Jean-Pierre Christophe Hecht.
Fecha de Entrega:	Domingo 11 de diciembre del 2022.

Tema Elegido:

En el panel central, con vista aérea, deben aparecer bolas en posicionesrandómicas, una blanca y otras de color. El taco debe aparecer automáticamente apuntando a la bola blanca y debe ser manejado controles GUI (teclado y mouse), para golpear la bola blanca. Las bolas deben tener la física de impactos, inercia y roce. En las esquinas debe haber troneras donde pueden caer. La cantidad de bolas debe ser definible por interfaz GUI. Habrá bandas para rebote de las bolas y si caen en las troneras, otorgan puntos. Reiniciar se debe hacer por controles GUI. Si secae la bola de color y la blanca no hay puntaje, si cae la blanca se restapuntaje.

Diagrama Gráfico UML: (En el repositorio se encuentra con mayor calidad)

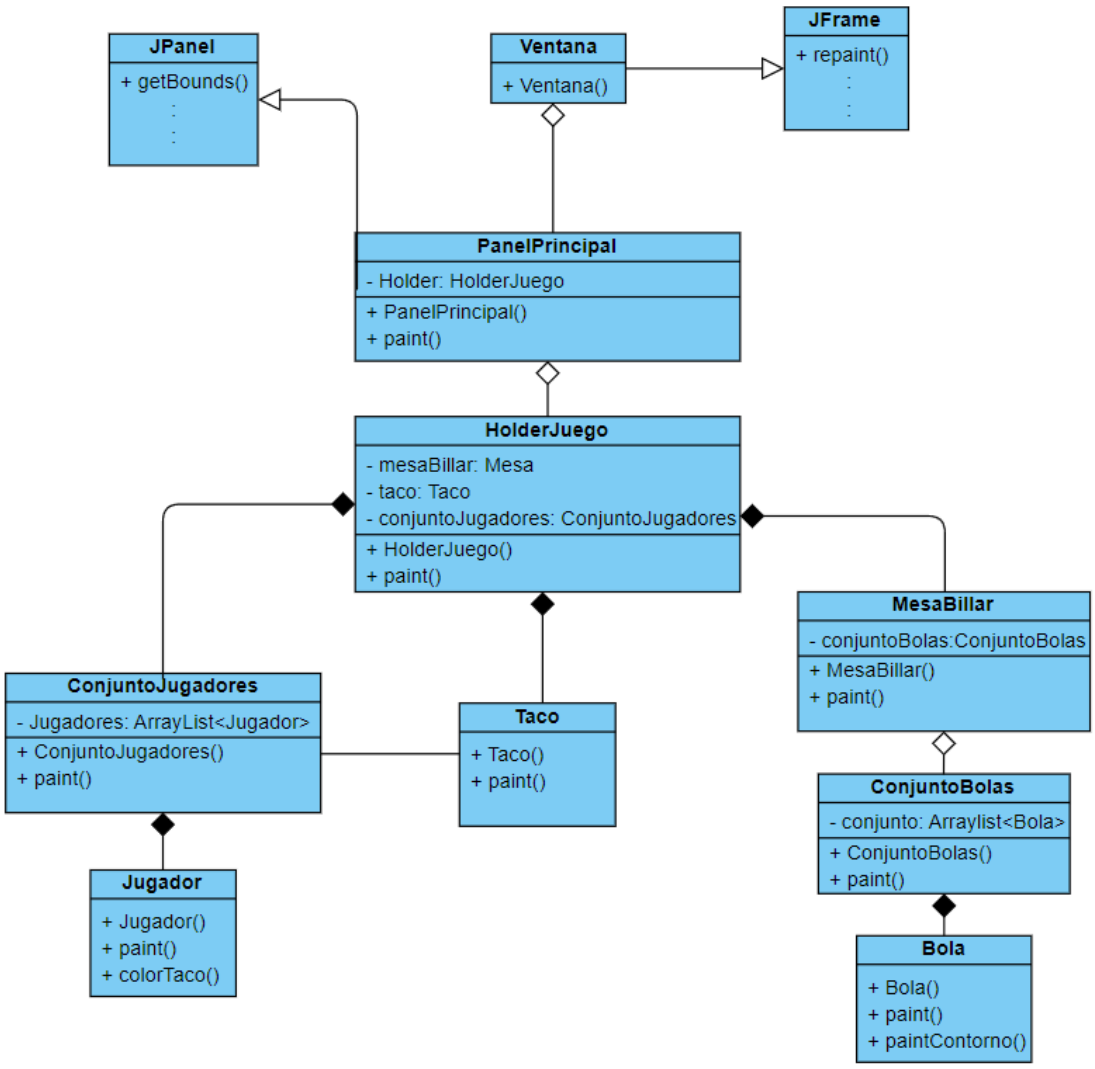


Diagrama Lógico UML: (En el repositorio se encuentra con mayor calidad)

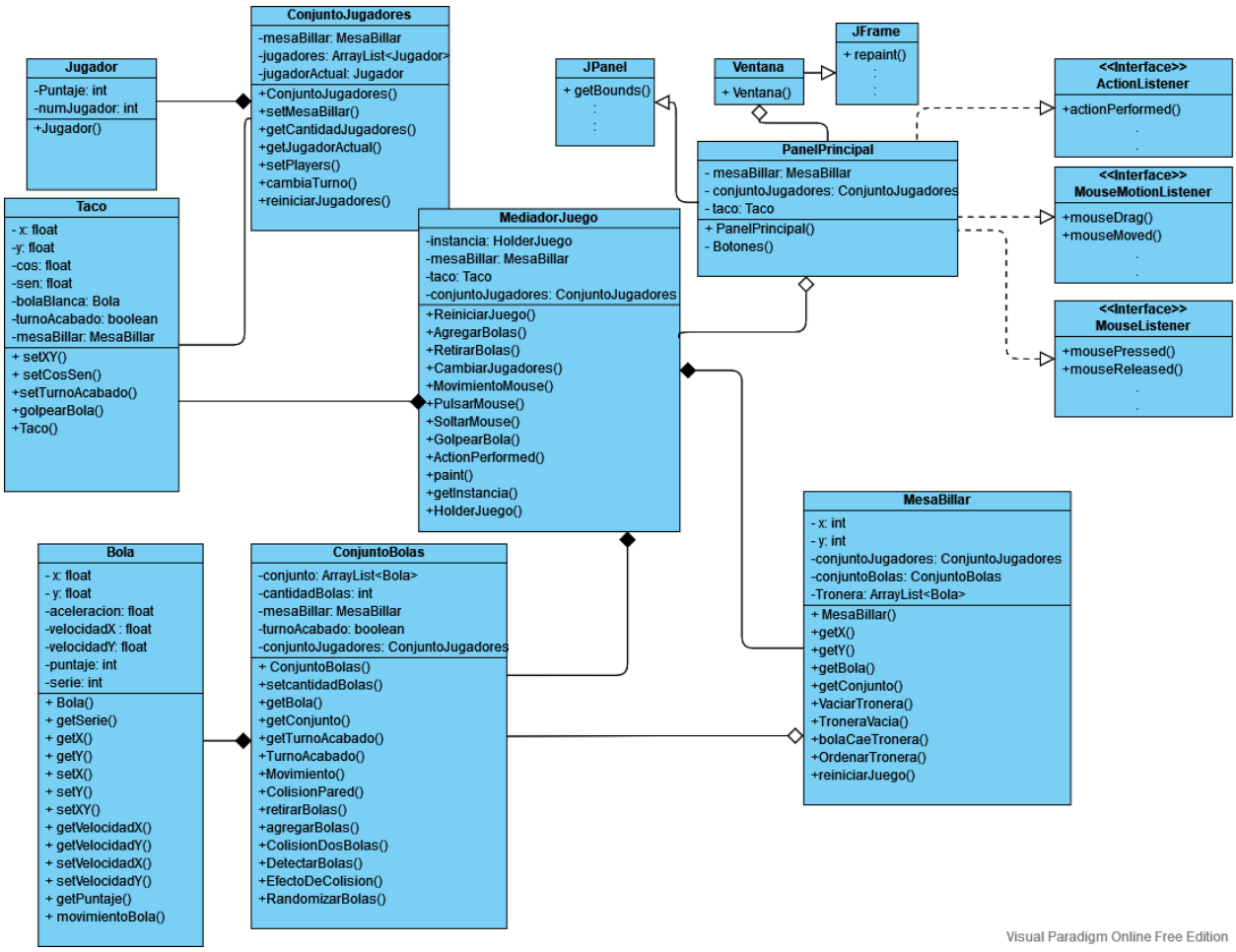
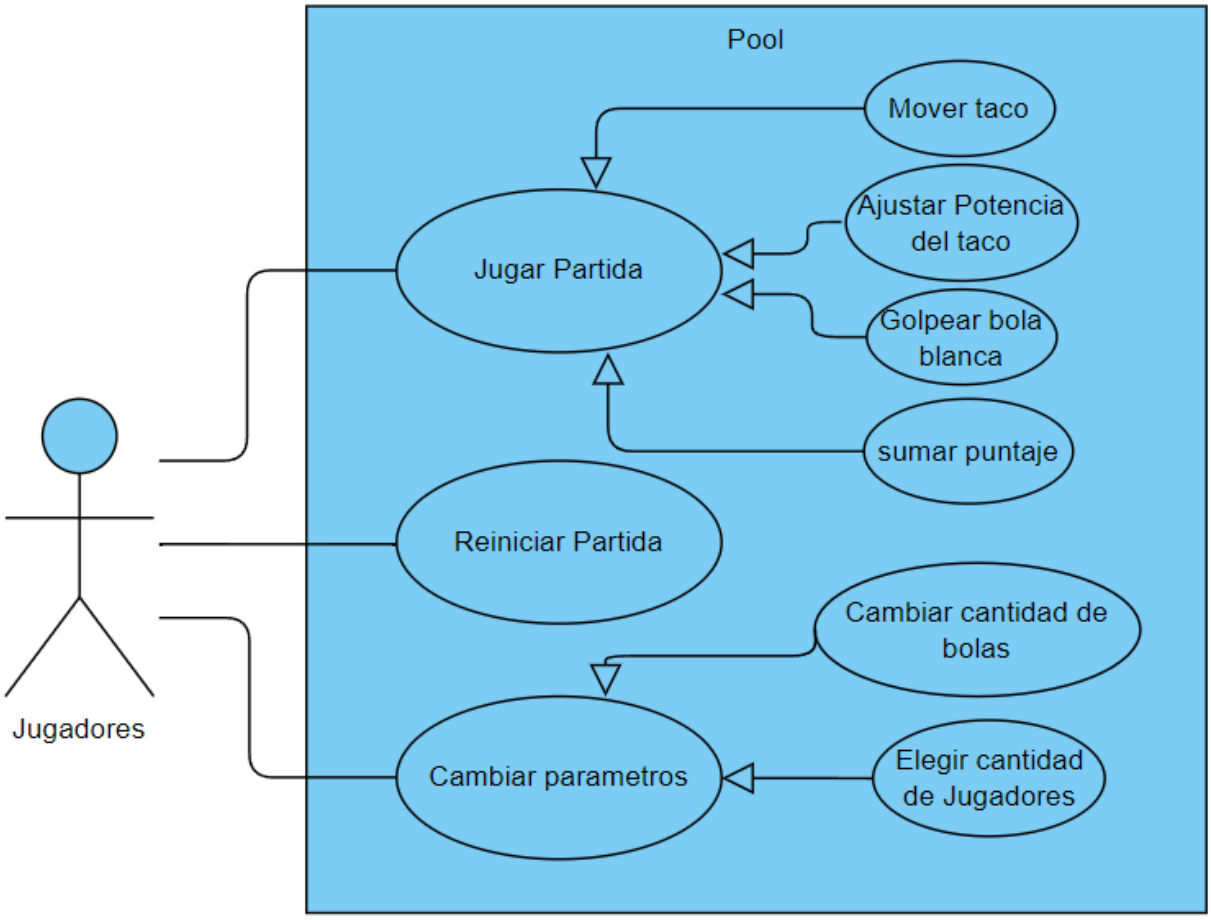


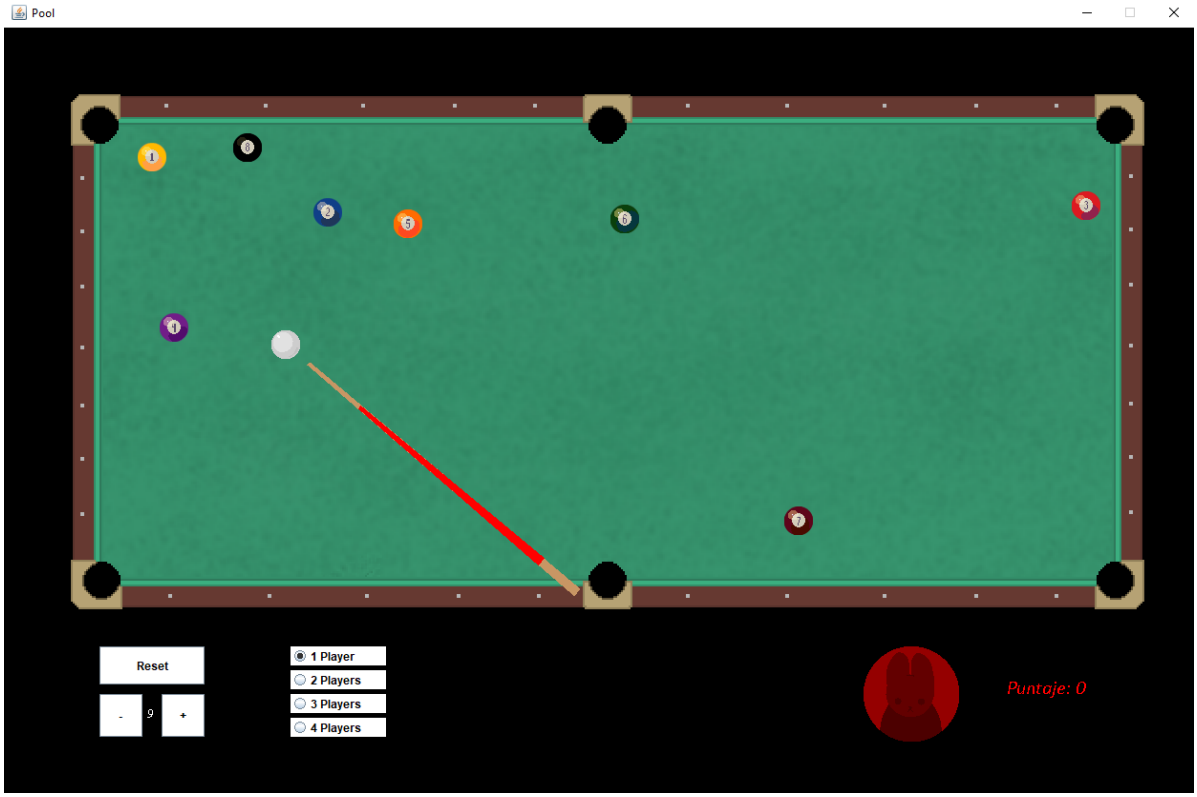
Diagrama de Casos de Uso: (En el repositorio se encuentra con mayor calidad)

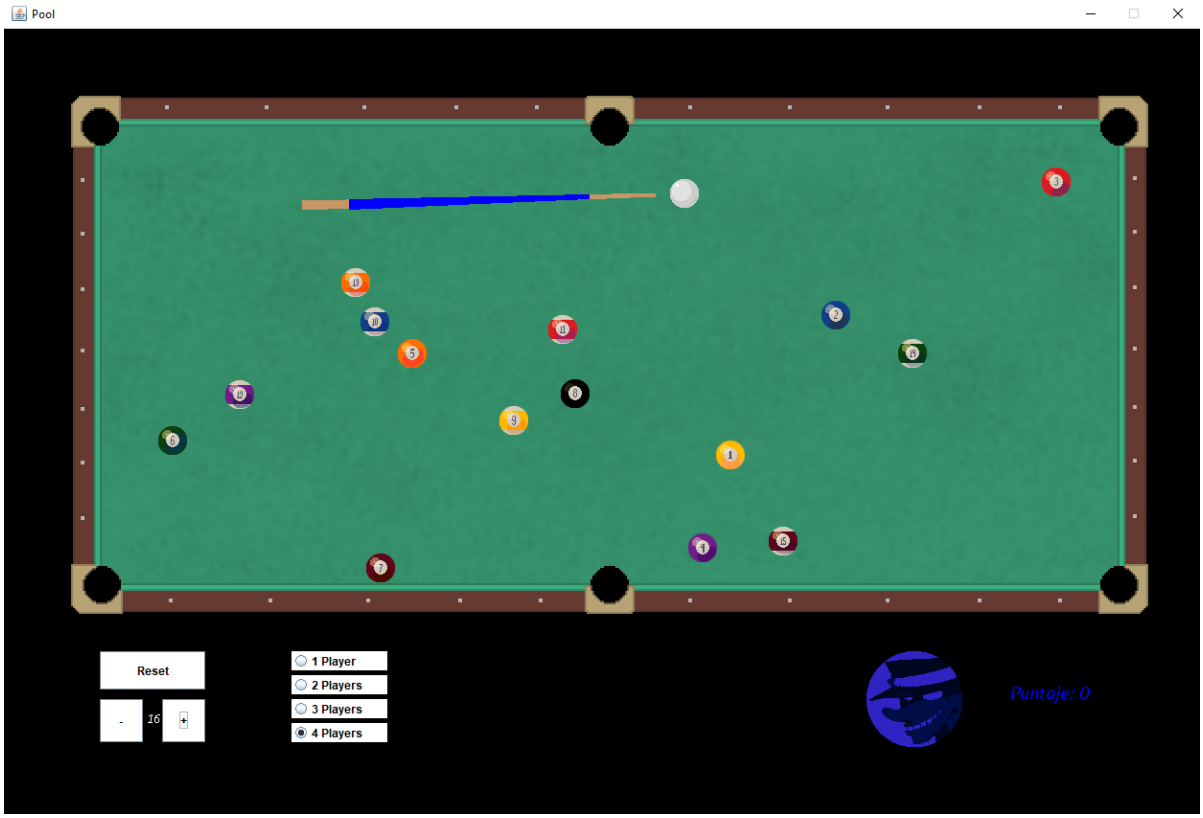


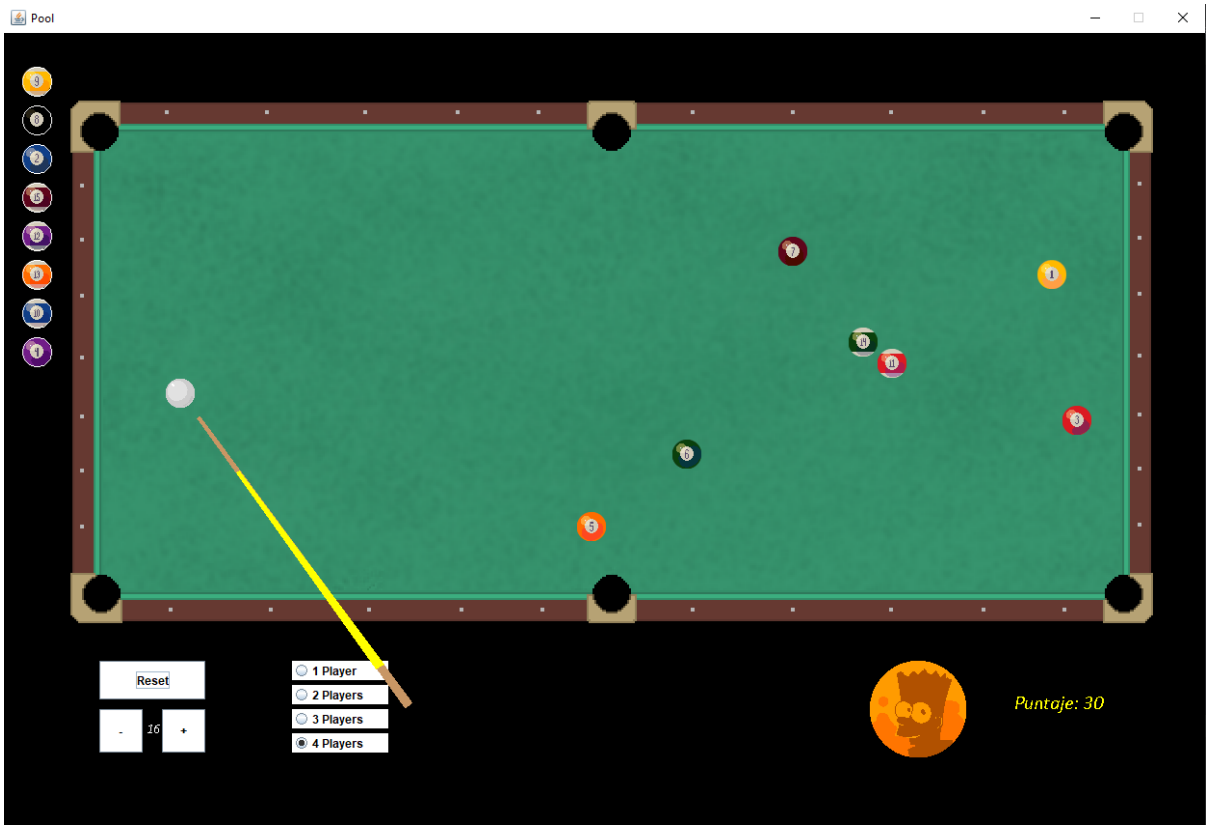
Patrones Usados:

- Patrón Mediador: Lo ocupamos para facilitar la conexión de las clases MesaBillar, Taco, ConjuntoJugadores y PanelPrincipal de modo que nuestro panel principal no tenga que tener tantas referencias y el código no sea tan complejo.
- Patron Singleton: Lo utilizamos en la clase MediadorJuego, puesto que no queremos que se generen múltiples clases de este tipo.

Imágenes del Programa: (En el repositorio se encuentran con mayor calidad)




Pool
Reset
-
16
+
1 Player
2 Players
3 Players
4 Players
Puntaje: 0


Pool
Reset
-
16
+
1 Player
2 Players
3 Players
4 Players
Puntaje: 30

Decisiones Importantes:

- En un principio todas nuestra variables numéricas eran del tipo double, pero con el tiempo, decidimos hacer uso de float, puesto que es difícil que ocupemos tal cantidad de cifras significativas, además de evitar casteos de int al usar Math.round() y un trabajo innecesario para nuestro computador.
- Cuando empezamos el proyecto, teníamos la idea de usar dos JPanel, uno para opciones que el jugador cambiaría mediante JButtons y otro para la mesa de billar, al final descartamos esta idea, puesto que no lo vimos necesario.
- Implementar modo multijugador, haciendo una clase ConjuntoJugadores en la cual se determina la cantidad de jugadores y el Jugador actual va rotando.
- Implementar iteraciones para que las colisiones sean más exactas y no ocurran errores, como el acoplamiento de las bolas, esta funciona de tal modo que mientras en la pantalla vemos un frame, el programa divide todos los movimiento en fracciones de 10 y los revisa uno a uno.
- Usar texturas para ciertos objetos, de manera que nuestro programa se vea más prolijo.
- Implementación de un “Medidor” de fuerza para el disparo del taco a la bola blanca.
- Implementación de JButtons que tienen el objetivo de disminuir o aumentar la cantidad total de bolas en la mesa.

Problemas encontrados:

- Colisiones, la detección es fácil pero la implementación de la trayectoria luego de la colisión es compleja. Entre otros grandes problemas que encontramos desarrollándose, encontramos bugs tales como el acoplamiento de las bolas, el atascamiento de una bola después de haber chocado con una, repentina detección de movimiento, etc.
- El uso de patrones nos causó problemas, puesto que nos costó ver en qué casos ocuparlos de manera eficaz, sin hacer el código confuso.
- Se nos complicó el emplear JUnit dado que no sabíamos en qué parte del código hacerlo, finalmente si lo hicimos, pero nos falta control en esta área.