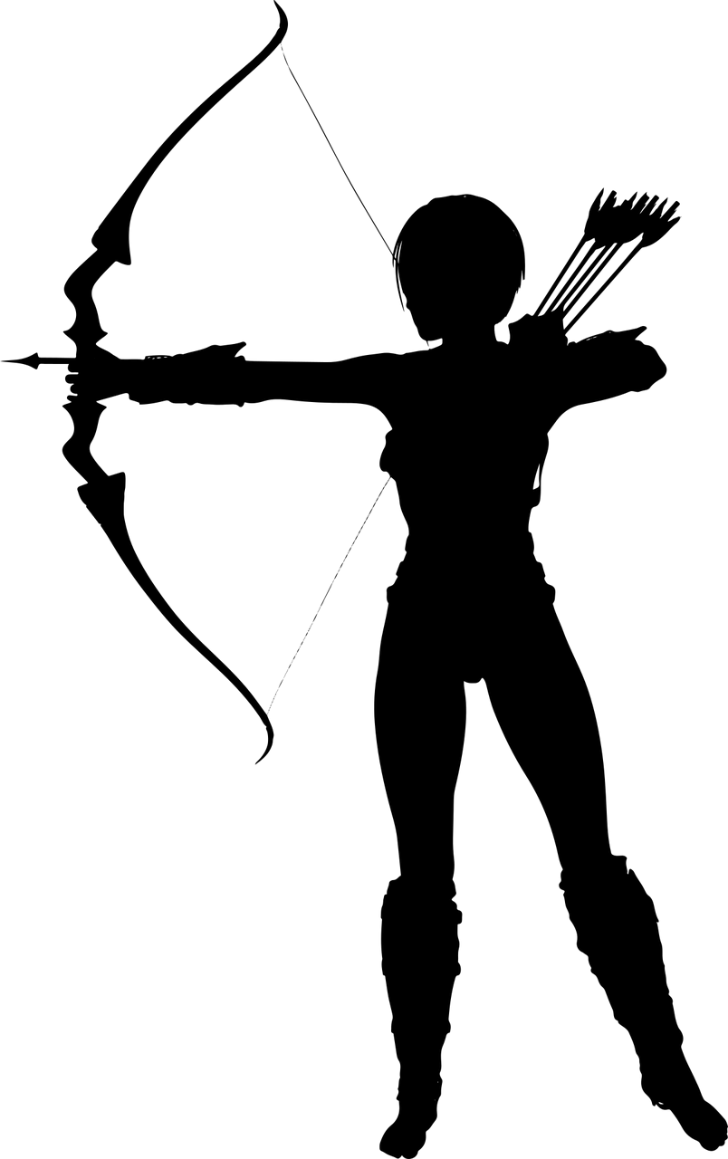
El Favor de las guerreras



Samuel Moreno Vincent

Contenido

[1. Introducción y Objetivos 2](#_Toc73995295)

[2. Reglas del juego 2](#_Toc73995296)

[2.1 Preparación 2](#_Toc73995297)

[2.2 Secuencia de Juego 2](#_Toc73995298)

[2.2.1 Fase 1: Reparto 2](#_Toc73995299)

[2.2.2 Fase 2: Acción 2](#_Toc73995300)

[2.2.3 Fase 3: Puntuar y Actualizar 3](#_Toc73995301)

[2.3 Finalización del Juego 4](#_Toc73995302)

[3. Uso e instalación 4](#_Toc73995303)

[3.1 Despliegue del entorno 4](#_Toc73995304)

[3.2 Ejecución y configuración 5](#_Toc73995305)

[4. Interfaces y alcance 6](#_Toc73995306)

[4.1 Interfaz grafica 6](#_Toc73995307)

[4.1.1 Acciones del adversario 7](#_Toc73995308)

[4.1.2 Guerreras, armas usadas y favor 7](#_Toc73995309)

[4.1.3 Acciones del jugador 8](#_Toc73995310)

[4.1.4 Mano del jugador 9](#_Toc73995311)

[4.1.5 Acción elegida 10](#_Toc73995312)

[4.1.6 Botón de aceptar 10](#_Toc73995313)

[4.1.7 Pantalla de información final 10](#_Toc73995314)

[5. Desarrollo 11](#_Toc73995315)

[5.1 Estrategia de solución 11](#_Toc73995316)

[5.1.1 Disposición del tablero y transformación de este a datos útiles 11](#_Toc73995317)

[5.2 Vista de bloques de construcción 13](#_Toc73995318)

[5.2.1 Nivel 1 14](#_Toc73995319)

[5.2.2 Nivel 2 14](#_Toc73995320)

[5.2.3 Nivel 3 15](#_Toc73995321)

[5.3 Vista de tiempo de ejecución 18](#_Toc73995322)

[5.3.1 Opciones 18](#_Toc73995323)

[5.3.2 Ciclo de una partida 18](#_Toc73995324)

[5.4 Bibliotecas y software externo 20](#_Toc73995325)

# Introducción y Objetivos

El trabajo consiste en una versión en digital del juego de mesa Hanamikoji, en el que el jugador se enfrenta a una inteligencia artificial entrenada usando una red neuronal que aprende a jugar usando aprendizaje profundo. Para ello se ha implementado la lógica del juego en Python y se le suministrará a la librería de Python TensorFlow, en la que se programará la red neuronal. De manera que llegue a tener un nivel suficiente como para llegar a ser un reto para el jugador.

# Reglas del juego

## Preparación

Coloca 7 Cartas de Geisha en una fila, en el siguiente orden de izquierda a derecha, entre los jugadores.

Coloca 1 Marcador de Victoria en el centro de cada Carta de Geisha.

Apila las Cartas de Objeto boca abajo en un mazo y colócalo a un lado.

Cada jugador coge 4 Marcadores de Acción del mismo color con el lado coloreado hacia arriba y los coloca frente a él.

El jugador más joven es el jugador inicial.

## Secuencia de Juego

El juego se desarrolla durante una o varias rondas. Cada ronda consiste en 3 fases en el siguiente orden:

Fase 1: Reparto

Fase 2: Acción

Fase 3: Puntuar y Actualizar

Si cualquier jugador consigue el objetivo de victoria en la fase de Puntuar, el juego finaliza inmediatamente. Si ningún jugador consigue el objetivo de victoria, el juego prosigue a la siguiente ronda. El juego continuará hasta que alguno de los jugadores gane.

### Fase 1: Reparto

El jugador inicial baraja las 21 Cartas de Objeto y las apila boca abajo, y al azar retira 1 carta del mazo y la devuelve a la caja del juego sin mirarla. Esta carta no se usará en esta ronda. Ningún jugador tiene permitido revisarla.

Reparte a cada jugador 6 Cartas de Objeto para conformar su mano, que se mantendrá oculta.

Apila el resto de Cartas de Objeto boca abajo como “Mazo de Objetos” y colócalo a un lado de la fila de Cartas de Geisha.

### Fase 2: Acción

Empezando por el jugador inicial, los jugadores se alternan en turnos (Jugador A -> Jugador B -> Jugador A -> Jugador B -> Etcétera) hasta que ambos jugadores hayan realizado 4 turnos.

En tu turno, debes coger una carta del Mazo de Objetos y realizar una acción.

Cuando realizas una acción, escoges 1 de tus Marcadores de Acción del lado colorado y realizas la correspondiente acción. Tras resolverla, coloca boca abajo el marcador. No puedes usar marcadores boca abajo (los marcadores de acción de cada jugador se podrán usar una sola vez por ronda.

Hay 4 acciones en Hanamikoji:

##### 1. Secreto

Escoge una carta de tu mano y colócala boca abajo debajo del Marcador de Acción usado (Secreto). Esta carta se revelará en la Fase de Puntuar y se puntuará.

Puedes revisar esta carta en cualquier momento.

##### 2. Renuncia

Escoge 2 cartas de tu mano y colócalas boca abajo frente a ti, debajo del Marcador de Acción usado (Renuncia). Estas cartas no puntuarán en esta ronda.

Puedes revisar estas cartas en cualquier momento.

##### 3. Regalo

Escoge 3 cartas de tu mano y colócalas boca arriba frente a ti.

Tu oponente escoge 1 de esas cartas y la coloca en su lado junto a la correspondiente Geisha. Tú colocas las otras 2 cartas en tu lado junto a la(s) correspondiente(s) Geisha(s). Estas cartas puntuarán.

##### 4. Competición

Escoge 4 cartas de tu mano y colócalas boca arriba frente a ti. Divídelas en dos grupos, cada uno con 2 cartas.

Tu oponente escoge 1 grupo y coloca las 2 cartas en su lado junto a la(s) correspondiente(s) Geisha(s). Tú colocas las otras 2 cartas en tu lado junto a la(s) correspondiente(s) Geisha(s). Estas cartas puntuarán.

### Fase 3: Puntuar y Actualizar

Después de que ambos jugadores hayan realizado 4 acciones, el juego prosigue a la Fase 3.

Los jugadores revelan la carta debajo del Marcador de Acción (Secreto) y la colocan en su lado junto a la correspondiente Geisha. Compara el número de Cartas de Objeto en ambos lados de cada Geisha.

- Un lado tiene más cartas que el otro: El lado con más Cartas de Objeto gana el Favor de la Geisha. Mueve el Marcador de Victoria al lado vencedor.

- Ambos lados empatan o no hay cartas: No se mueve el Marcador de Victoria.

Tras puntuar, los jugadores calculan el número de Geishas cuyo Favor han ganado y suman sus Puntos de Carisma. Si algún jugador alcanza el objetivo de victoria, el juego finaliza inmediatamente. (Ver Finalización del Juego).

Si ningún jugador consigue el objetivo de victoria, se procede a Actualizar:

- Recoge TODAS las Cartas de Objeto de la mesa y de la caja, apílalas boca abajo en un mazo y colócalo a un lado.

- Los Marcadores de Victoria permanecerán en su lugar. Nota: No los devuelvas al centro de cada Carta de Geisha.

- Los jugadores colocan boca arriba sus Marcadores de Acción.

- El segundo jugador se convierte en el jugador inicial.

- La siguiente ronda está lista para empezar.

## Finalización del Juego

Si algún jugador gana el Favor de 4 Geishas u 11 (o más) Puntos de Carisma, el juego finaliza.

Si un único jugador alcanza el objetivo de victoria, es el vencedor.

Si un jugador gana el Favor de 4 Geishas y el otro gana 11 (o más) Puntos de Carisma, éste último vence.

# Uso e instalación

## Despliegue del entorno

Descargar en instalar [Anaconda](https://www.anaconda.com/products/individual)

Una vez instalado Anaconda, desde la aplicación de promt de anaconda Anaconda Prompt crear un entorno nuevo con el comando:

*conda create --name <nombre del entorno>*

Para activar el entorno creado ejecutar el comando:

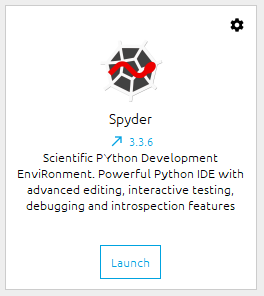
*conda activate <nombre del entorno>*

Abrir el navegador de Anaconda Anaconda Navigator, en Aplications on seleccionar el entorno creado. A continuación, abrir la sección Environments y buscar e instalar:

* Python 3.7.9
* Spyder 3.3.6
* Tensorflow 2.3.0
* ConfigParser 5.0.2

## Ejecución y configuración

Para la ejecución de la aplicación abrir el programa Spyder instalado dentro de Anaconda

[](https://github.com/samuelmorenov/El-Favor-de-las-Guerreras/blob/main/doc/images/spyder.png)

Desde Spyder abrir el archivo *main.py* que está en *\src\main\python* y pulsar F5 para ejecutar.

Para cambiar la configuración, abrir el archivo *param.properties* que está en \src\main\resources. Hay 3 campos configurables:

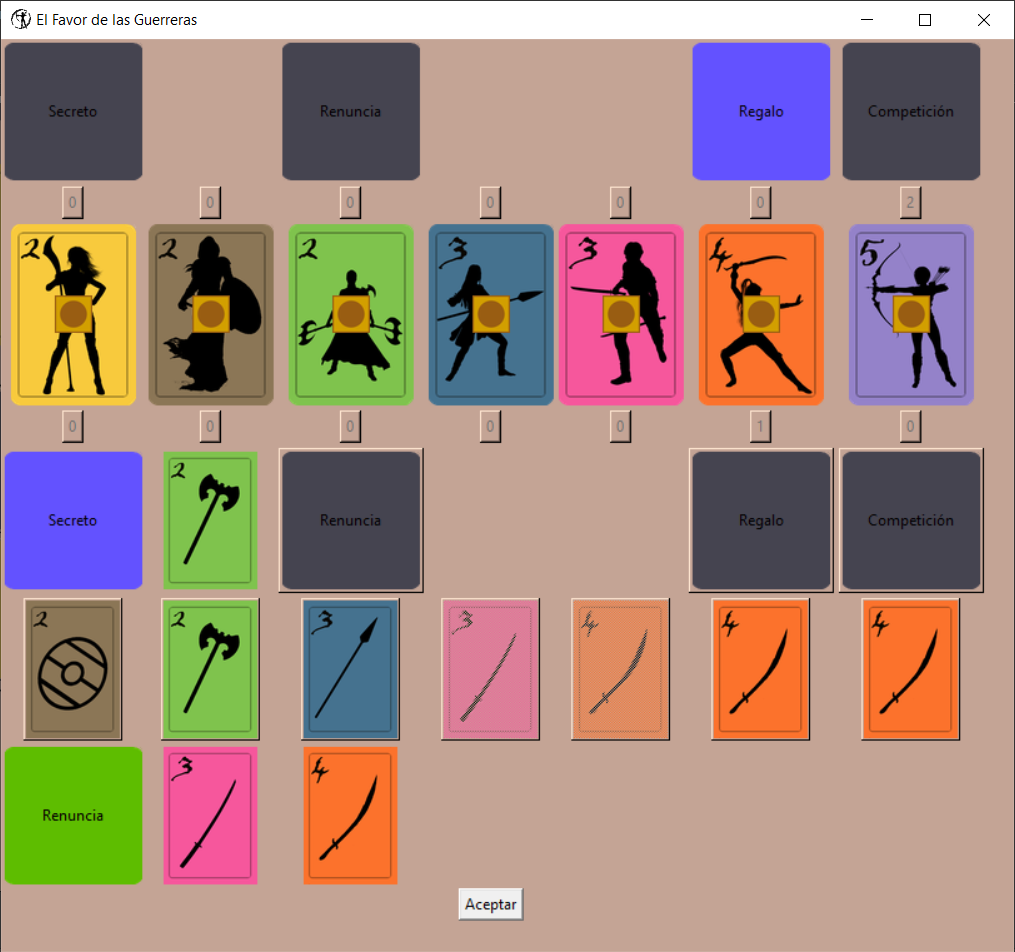
* MODO: Para elegir entre **Generar datos** (1), **Entrenar a la red neuronal** (2) o **Jugar** (3)
* NUM\_SIMULACIONES: En caso de haber elegido el modo de generar datos, este número indica el número de partidas que se van a simular
* DIFICULTAD: Para elegir entre **Bot de acciones aleatorias** (1) o **Red neuronal entrenada** (2)

# Interfaces y alcance

## Interfaz grafica

La interfaz gráfica se compone de una única pantalla en la que el usuario tiene una visión parcial de la situación actual del tablero.

Se compone de una matriz de 7x7 casillas donde cada fila corresponde a un conjunto diferente de datos que explicaremos a continuación.



*Imagen 4.1.a*

### Acciones del adversario



*Imagen 4.1.1.a*

Al inicio de la partida el adversario no habrá utilizado ninguna de las acciones, por lo que se mostraran todas en un color oscuro indicando esto tal y como se muestra en la Imagen 4.1.1.a.



*Imagen 4.1.1.b*

Cuando el adversario utilice alguna acción esta pasará a estar en un color azul mas claro.

Existe un espacio entre la acción de secreto y la acción de renuncia que representa la carta que el adversario haya usado para la acción de secreto. Si no se ha utilizado el hueco estará vacío, mientras que si se ha utilizado aparecerá una carta gris, ya que es una carta oculta y el jugador no puede saber que carta es.

También existen dos espacios entre la acción de renuncia y la de regalo que representan las dos cartas de renuncia del adversario en caso de haber usado ya esta acción al igual que con la carta de secreto.

### Guerreras, armas usadas y favor

**

*Imagen 4.1.2.a*

Estas 3 filas están orientadas, es decir, la parte superior corresponde al adversario y la parte inferior corresponde al jugador.

La fila del medio representa a las guerreras, y el marcador central representan su favor actual. Al inicio de la partida este favor estará siempre en el centro representando que son neutrales. Sin embargo, tras la primera ronda, si ningún jugador ha ganado el favor cambiará dependiendo de las acciones de la ronda anterior de manera que estará en la parte superior de la carta de la guerrera en caso de que esa guerrera este a favor del adversario (imagen 4.1.2.b) o en la parte inferior en caso de este a favor del jugador (imagen 4.1.2.c).

|  |  |
| --- | --- |
| *Imagen 4.1.2.b* | *Imagen 4.1.2.c* |
|  |  |

Los marcadores numéricos que aparecen encima y debajo de las cartas de las guerreras representan el numero de cartas que cada jugador ha jugador en favor de esa guerrera durante esta ronda. Los marcadores superiores corresponden al adversario y los inferiores al jugador. Hay que tener en cuenta, las acciones de secreto y renuncia guardan las cartas en la parte de acciones usadas y no se tienen en cuenta hasta el final de la ronda.

### Acciones del jugador



*Imagen 4.1.3.a*

Esta fila es muy similar a la de acciones del adversario al inicio, pero tiene dos diferencias fundamentales. Para empezar las acciones son interactivas, por lo que el usuario puede pulsar encima de ellas para seleccionarla. Cuando se selecciones una acción aparecerá marcada más abajo. Si se desea se puede cambiar de acción seleccionando una diferente, además seleccionar de nuevo una acción reinicia las cartas que hayan sido seleccionadas para esta acción.



*Imagen 4.1.3.b*

Al igual que con las acciones del adversario cuando ya han sido utilizadas en esta ronda se marcan en azul mas claro, pero a diferencia del adversario en esta ocasión si se podrán ver las cartas seleccionadas para las acciones de secreto y renuncia. Además, las acciones utilizadas dejan de ser interactiva, ya que el usuario no puede volver a utilizarlas.



*Imagen 4.1.3.b*

Cuando el jugador se encuentre con una acción de decisión en la que tenga que elegir cartas del adversario, las acciones del jugador quedarán bloqueadas, tornándose en un tono mas grisáceo, pero mostrando diferencia entre las acciones usadas y no usadas, así como las cartas usadas para las acciones de secreto y renuncia para poder tomar una decisión con toda la información del tablero.

### Mano del jugador



*Imagen 4.1.4.a*

En esta fila encontramos las cartas que tiene en la mano el jugador. El color y el símbolo representan a que guerrera corresponden. Estas cartas son interactivas, para poder seleccionarlas hay que haber elegido antes una acción, y dependiendo de la acción elegida se podrán seleccionar un numero distinto de cartas.



*Imagen 4.1.4.b*

Cuando se seleccione alguna carta esta quedará bloqueada, de manera que no se podrá volver a seleccionar. Si se desea cambiar de cartas se puede volver a seleccionar la acción borrando así las cartas seleccionadas.



*Imagen 4.1.4.c*

Cuando el jugador se encuentre con una acción de decisión en la que tenga que elegir cartas del adversario, las cartas del jugador quedarán bloqueadas al igual que las acciones.

### Acción elegida



*Imagen 4.1.5.a*

Esta fila aparecerá en un principio vacía, pero irá cambiando, dependiendo de la acción y las cartas que seleccione el jugador. No es interactiva, ya que el jugador no puede pulsar en ninguna casilla. En caso de que el jugador tenga que realizar una acción de decisión esta aparecerá en esta fila.

#### Acción por realizar

En la primera casilla se muestra la acción que ha seleccionado el jugador (ver sección 4.1.3). A continuación, se muestran las cartas que ha seleccionado el jugador. Es importante destacar que en para la acción de competición el orden en el que se seleccionen las cartas tiene relevancia, ya que las dos primeras formarán un grupo y las dos segundas el otro grupo. Para el resto de las acciones el orden es indiferente.

#### Acción de decisión



*Imagen 4.1.5.2.a*

Cuando el jugador tenga que realizar una acción de decisión en el lugar de la acción aparecerá una breve descripción de lo que debe hacer. En caso de ser una acción de regalo tendrá que seleccionar una carta de las 3, la cual aparecerá bloqueada después de ser seleccionada, pudiendo seleccionar cualquiera de las otras dos si el jugador cambia de opinión.

En caso de ser una acción de competición el jugador podrá elegir entre el grupo de las dos primeras o el grupo de las dos últimas. Al pulsar sobre una carta se quedan marcadas las dos cartas de ese grupo, pudiendo cambiar de grupo en caso de cambio de opinión.

### Botón de aceptar

En ultimo lugar tenemos el botón de aceptar, este botón aparece únicamente cuando el jugador ha seleccionado una acción y el correspondiente grupo de cargas o ha seleccionado las cartas que desea en la acción de decisión. En caso de que el jugador cambie de acción y las cartas seleccionadas se reinicien el botón desaparecerá hasta que el jugador vuelva a seleccionar el numero correcto de cartas.

Al pulsar este botón la acción queda registrada y se envía a la aplicación, para que el adversario realice su acción y le vuelva a tocar el turno al jugador.

### Pantalla de información final

Cuando se acaba la partida se cierra la pantalla principal y aparece una pequeña ventana anunciando el resultado de la partida.

# Desarrollo

## Estrategia de solución

### Disposición del tablero y transformación de este a datos útiles

#### Elementos del tablero

Como se describe en las reglas del juego, hay 7 cartas centrales que en este caso llamaremos guerreras con diferentes valores.



Estos valores vienen dados por el número de cartas de favor que hay para cada guerrera en el mazo de cartas.

##### Favor de las guerreras

Las guerreras tienen un marcador de favor, que empieza estando imparcial y al final de cada turno cambiará dependiendo del favor que haya conseguido cada jugador. Por tanto, tendremos que guardar un array de 7 campos, uno para cada guerrera, con valores de 0 si ningún jugador tiene su favor, 1 si el jugador 1 tiene su favor y 2 si lo tiene el jugador 2.

##### Cartas en la mano

Cada jugador tendrá cartas en su mano, llegando a tener como máximo 7 cartas. Por lo que habrá que guardar los datos de las cartas de cada jugador, para ello se utilizará un array para cada jugador de 7 campos con valor 0 si es un hueco y valores del 1 al 7 de manera que el 1 corresponde a una carta de favor de la guerrera 1 y así sucesivamente hasta el 7.

##### Acciones realizadas

Cada jugador tendrá la opción de realizar las 4 acciones por turno, por lo que hay que guardar los datos de acciones se han realizado y cales no para cada jugador. Además, en caso de la acción 1 hay que guardar la carta que se utilizó para el secreto, y en el de la acción 2 hay que guardar las 2 cartas que se descartaron para la renuncia. Así pues, se guardarán todos estos datos en un único array para cada jugador con 5 posiciones con la siguiente configuración:

* Posición 0: guarda el valor de la carta de secreto (0 en caso de no haber sido usado)
* Posición 1 y 2 valores de las cartas de renuncia (0 en caso de no haber sido usado)
* Posición 3 guarda si se ha usado el regalo (0 si no se ha usado, 1 si se ha usado)
* Posición 4 guarda si se ha usado la competición (0 si no se ha usado, 1 si se ha usado)

##### Armas usadas

Las armas (que son las cartas que se roban del mazo de armas y se usan para subir el favor de las guerreras) que cada jugador ha utilizado. Estas se guardarán en un array para cada jugador, en el que cada campo corresponde a una de las guerreras y el valor de este campo es el número de armas que ha usado en esa guerrera.

##### Acción de decisión

En caso de que en que un jugador haya jugado la acción 3 o la acción 4, el otro jugador deberá decidir entre las opciones que se le ofrecen. Para esto guardaremos las cartas que se ofrecen junto al tipo de acción en un array auxiliar, donde la primera posición será el valor de la acción y las siguientes las cartas a elegir. En el caso de que sea la acción 4 estarán en orden, la posición 1 y 2 corresponderán a un grupo y la posición 3 y 4 al otro grupo.

#### Matriz de información

Resumiendo, el punto anterior, tenemos 2 array para las manos de los jugadores, 2 array para las armas usadas, 2 array para las acciones realizadas, 1 array para el valor del favor actual y 1 array para la acción de decisión. Aunque cada array tiene un tamaño distinto, para poder conformar una matriz con toda la información tomaremos el valor máximo de todas ellas, que es 7. De esta manera tenemos una matriz de 7x8 con toda la información de un estado concreto del tablero de juego.

##### Transformación de la matriz

A la hora de enviar esta información a un jugador esta

##### Un ejemplo de matriz

0000023 -> Cartas en la mano del jugador 1 (2 y 3)

0000000 -> Mano vacía del jugador 2

4001100 -> Acciones realizadas por el jugador 1 (ha realizado las acciones 3, 4 y 1 (usando la carta 4))

1111100 -> Acciones realizadas por el jugador 2 (ha realizado todas sus acciones)

0001213 -> Armas utilizadas por el jugador 1 (1 en la guerrera 4, 2 en la guerrera 5, ...)

0110122 -> Armas utilizadas por el jugador 2 (1 en la guerrera 2, 1 en la guerrera 3, ...)

0201210 -> Favor de las guerreras (el jugador 1 gana en las 4 y 6 y el jugador 2 en las 2 y 5)

0000000 -> Acción de decisión, al estar vacía hay que realizar una acción normal

#### Acción decidida por el jugador

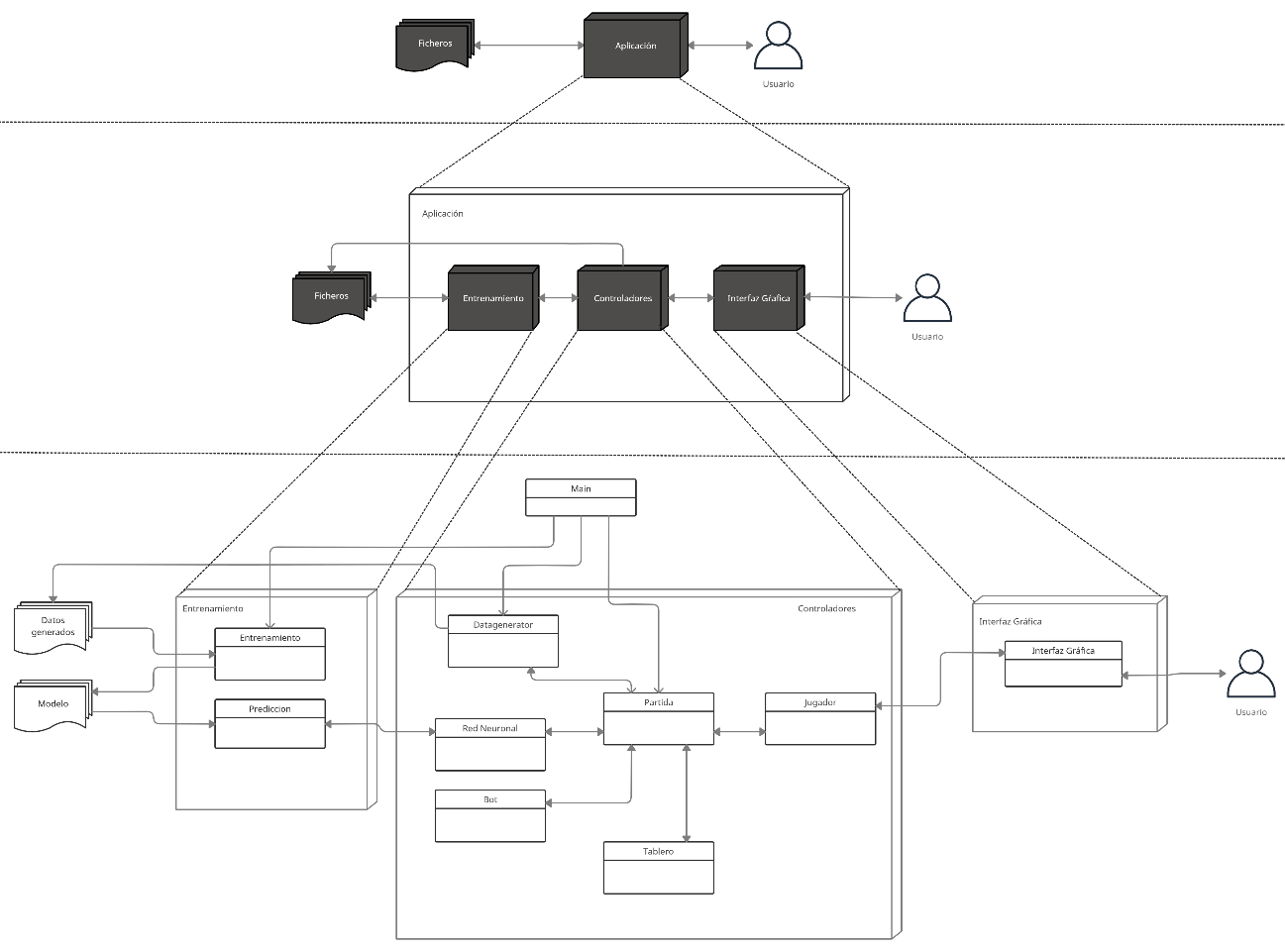
Ya sea un usuario mediante la interfaz gráfica, o la red neuronal, recibirá la matriz descrita anteriormente con la información de la situación actual del tablero, con esta información el jugador realizará una acción dentro de las 4 posibles o una acción de decisión dentro de las 2 posibles. Además, todas las acciones van acompañadas de un numero de cartas entre 1 y 4 dependiendo de la acción. Por lo tanto, la acción completa se enviará como un array de 5 campos, donde el primer campo es la acción por realizar (0, 1, 2 o 3 para las acciones normales, 4 para la acción de decisión de la acción de regalo y 5 para la acción de decisión de la acción 4) y el resto de los campos las cartas seleccionadas (0 en caso de que la acción no requiera todos los campos).

Un ejemplo de acción

13200 -> El jugador realiza la acción de renuncia con las cartas 3 y 2

## Vista de bloques de construcción

En la siguiente imagen (Imagen 5.2.a) podemos ver una vista general de los bloques de construcción que se describirán en los siguientes puntos. La leyenda de la imagen se comparte con la de todos los puntos de esta sección de la documentación.

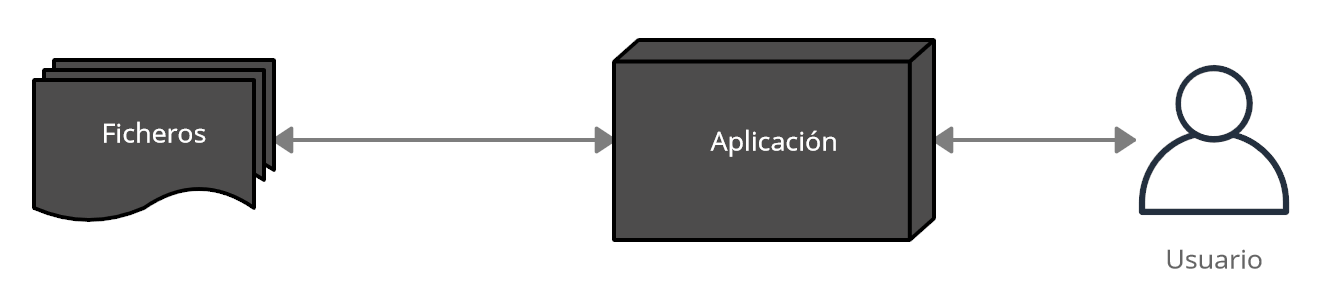


*Imagen 5.2.a*

Leyenda:

|  |  |
| --- | --- |
|  | Persona que interactua con la aplicación |
|  | Bloque de caja negra |
|  | Bloque de caja blanca |
|  | Ficheros de caja negra |
|  | Ficheros de caja blanca |
|  | Clase o archivo de funciones |
|  | Llamada a función sin paso de datos |
|  | Llamada a función con paso de datos |

### Nivel 1

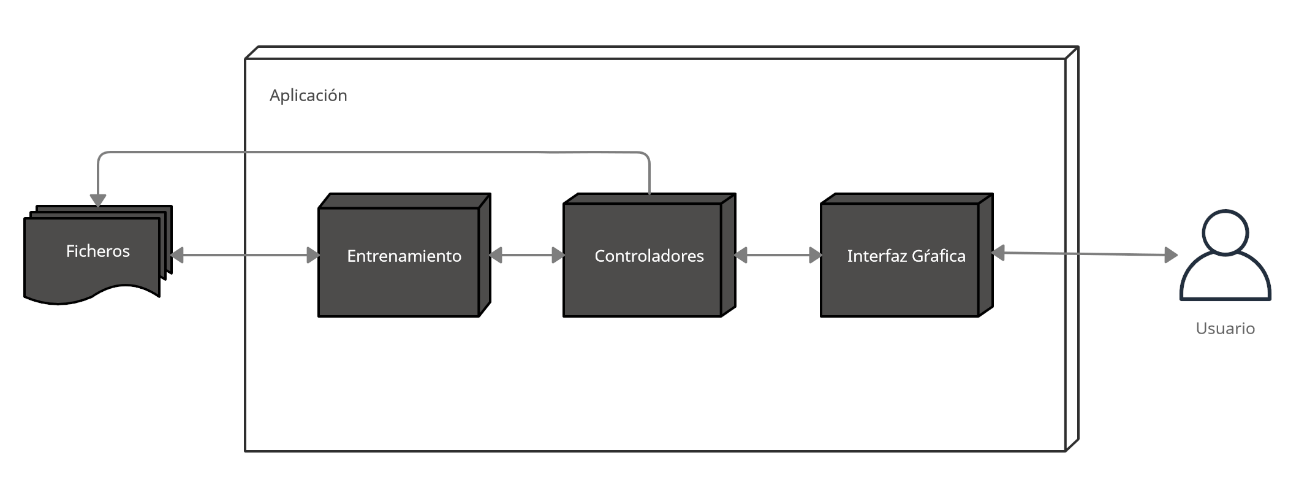


*Imagen 5.2.1.a*

En el nivel 1 podemos ver como la aplicación puede comunicar y recibir información del usuario que la esté ejecutando. Cabe destacar que en caso de que se ejecutando en modo generación de datos o entrenamiento de la red neuronal el usuario no interactúa con la aplicación, esta comunicación se produce únicamente cuando se ejecuta en modo jugar.

También podemos ver como la aplicación se comunica con ficheros externos. Esta comunicación si se realiza para los 3 modos de ejecución como veremos más adelante

### Nivel 2



*Imagen 5.2.2.a*

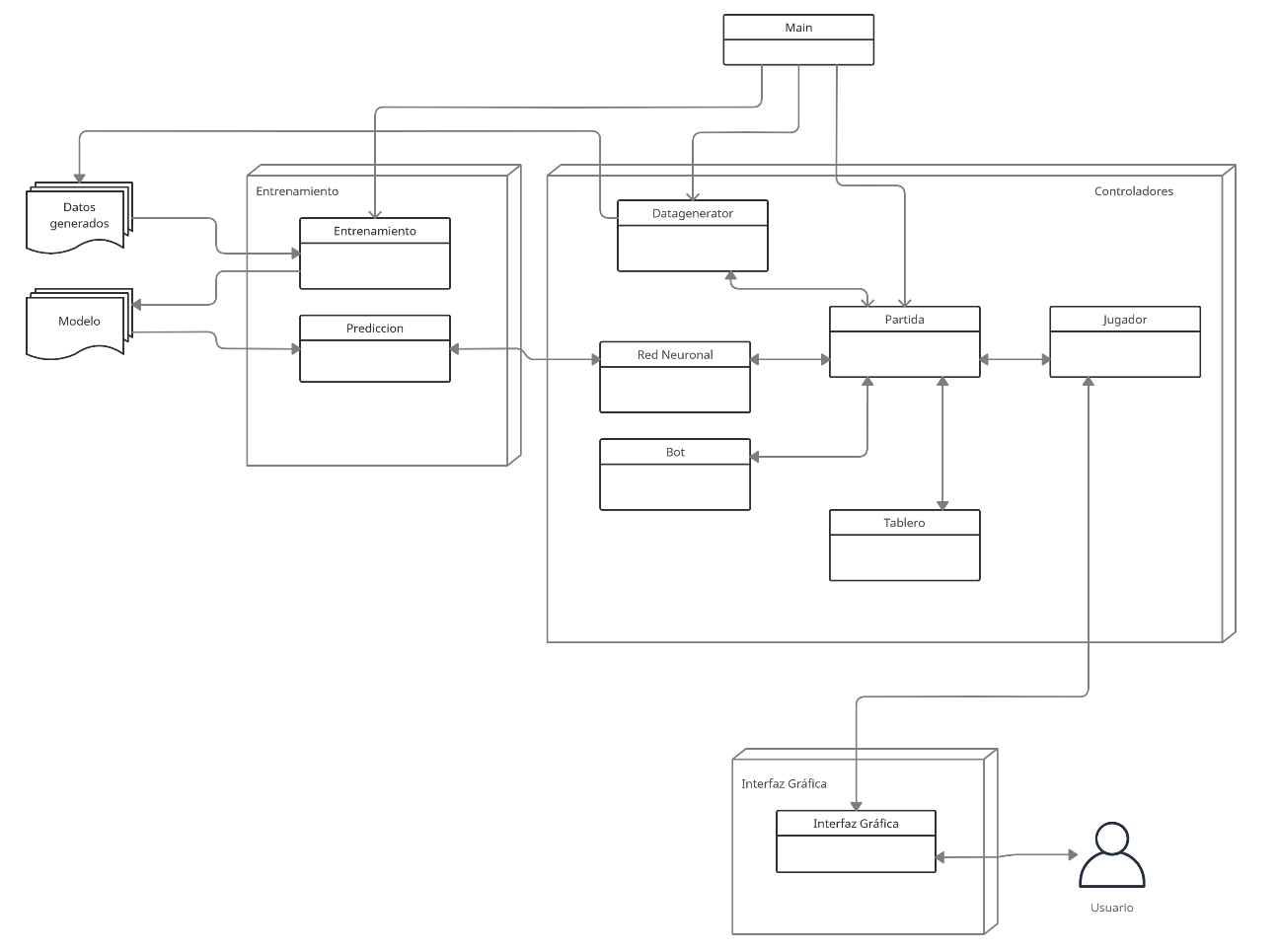
En el nivel 2 podemos ver como la aplicación tiene 3 paquetes principales.

El paquete de controladores está en el centro de ellos porque es el encargado de gestionar el resto, este se comunica con el paquete de entrenamiento de la red neuronal tanto para pedir datos como para recibirlos, así como con el de la interfaz gráfica. También puede generar información que se guarda en ficheros.

El paquete de entrenamiento es el encargado de gestionar la red neuronal. Hace uso y guarda archivos y se comunica con el paquete de controladores.

El paquete de interfaz gráfica es el encargado de comunicar el paquete de controladores con el del usuario.

### Nivel 3

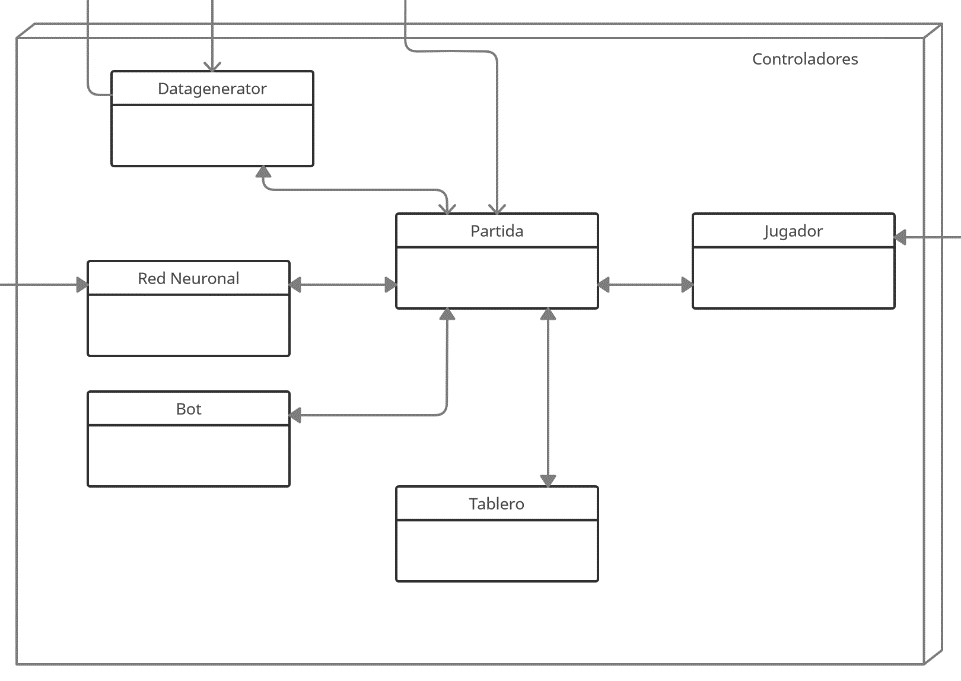


*Imagen 5.2.3.a*

En este nivel podemos ver todas las clases principales de la aplicación y la relación entre ellas.

Desde el main se puede llamar a 3 clases dependiendo del modo de ejecución en el que estemos. Para el modo de generación de datos se llama la clase DataGeneratorController, para el modo de entrenamiento de la red neuronal se llama a la clase Entrenamiento y para el modo jugar se llama a la clase PartidaController.

#### Controladores



*Imagen 5.2.3.1.a*

En el paquete de controladores tenemos 2 clases que se llaman desde el main.

La clase DataGeneratorController sirve para llamar a la clase PartidaController el numero de veces que se establezcan en la parametrización, de esta clase recibe los datos de los tableros y acciones que se han realizado en cada partida y se guardan en los ficheros de datos generados.

La clase PartidaController es el centro de la aplicación, ya que es la que se encarga de gestionar y comunicar entre sí gran parte del resto de clases. En el caso de que la aplicación se ejecute en modo de generación de datos, la PartidaController será llamada un numero parametrizado de veces, pero en todas ellas hará lo mismo que si fuera una ejecución en modo jugar, como podemos ver en el apartado 5.3.2 de la documentación. Recibe el tablero de la clase TableroController y se la envía a la clase Red Neuronal (NeuralNetworkController), JugadorController o BotTonto dependiendo de que jugadores estén jugando esa partida. De esas clases recibirá la acción realizada y se la pasará a clase TableroController.

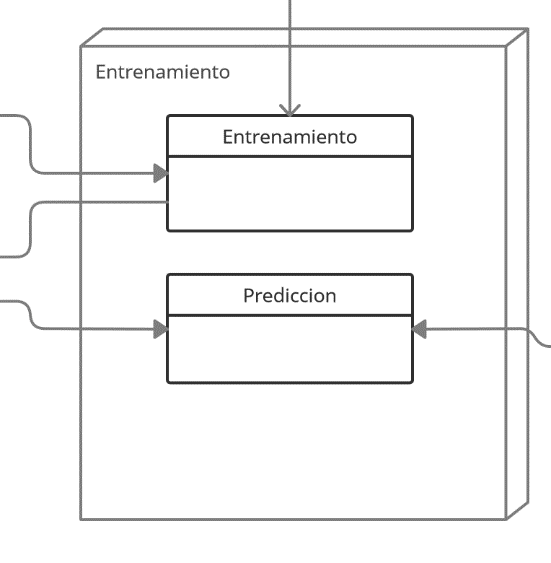
La clase JugadorController sirve de interfaz intermediaria entre la clase PartidaController y la interfaz de usuario.

La clase Red Neuronal (NeuralNetworkController) sirve de intermediaria entre la clase PartidaController y la clase de Predicción que forma parte del paquete de entrenamiento de la red neuronal.

La clase TableroController es la que se encarga de toda la gestión del tablero, lo inicializa al inicio de cada turno para enviárselo a la clase PartidaController, luego recibe las acciones que realicen los diferentes jugadores desde la clase PartidaController y guarda los cambios que realizan esas acciones en el tablero, devolviéndoselo de nuevo para la siguiente acción.

La clase del Bot sirve como jugador aleatorio. Esta clase recibe un tablero y realiza una acción completamente aleatoria dentro de las acciones posibles y con unas cartas aleatorias suficientes para realizar esa acción que estén disponibles en la mano de este jugador. Sirve como jugador inicial para el entrenamiento y como jugador de dificultad baja para el modo jugar si así se define en el archivo de parametrización.

#### Entrenamiento



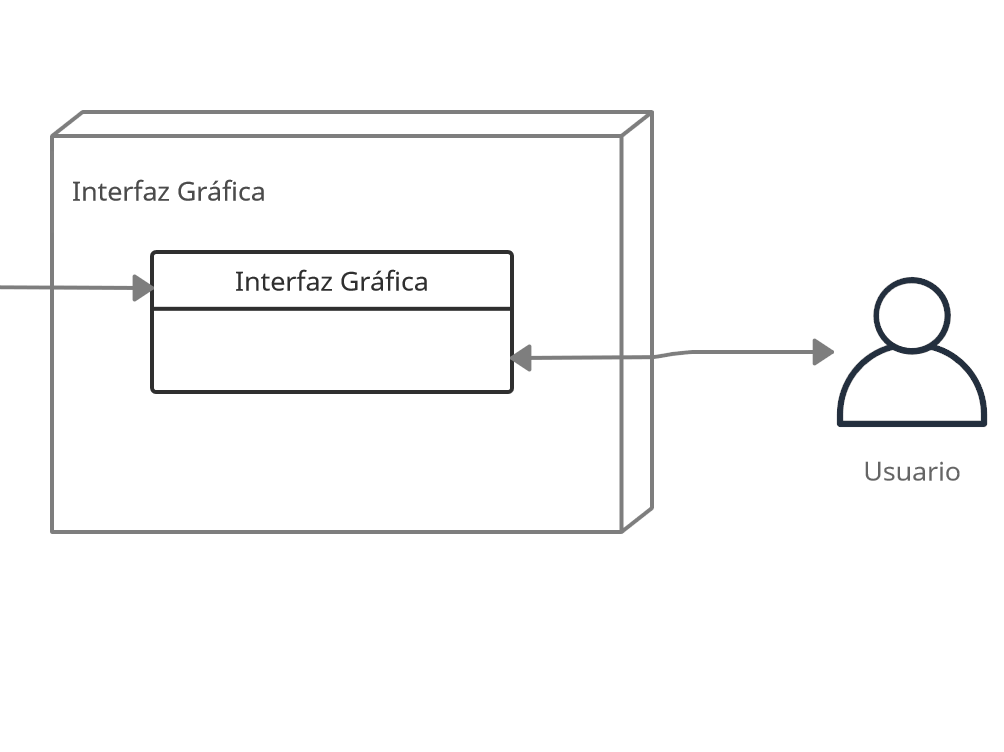
*Imagen 5.2.3.2.a*

En el paquete de entrenamiento hay únicamente 2 clases, la clase de entrenamiento y la de predicción.

La clase de entrenamiento se ejecuta directamente desde el main, lee los archivos de datos y genera los archivos del modelo.

La clase de predicción carga los archivos del modelo, recibe el tablero de la clase de red neuronal y le devuelve una predicción de una acción que será la que realice como jugador.

#### Interfaz grafica



*Imagen 5.2.3.2.a*

En la interfaz gráfica tenemos una única clase que utiliza Tkinter para hacer la ventana gráfica donde el usuario interactuará como jugador con la aplicación, de manera que esta clase pinta el tablero y ofrece al usuario las posibles acciones que puede realizar. Una vez seleccionada la acción esta vuelve a los controladores para seguir la partida.

En este paquete también existe una clase auxiliar que no aparece representada en el modelo ya que solo sirve para informar al usuario de la finalización del programa.

## Vista de tiempo de ejecución

### Opciones

Como hemos visto anteriormente existen 3 opciones parametrizables a la hora de ejecutar la aplicación. Estas opciones se arrancan desde el mismo main y llaman a su clase inicial correspondiente.

* Generar datos
* Entrenar a la red neuronal
* Jugar

La vista de tiempo de ejecución de entrenar a la red neuronal es la mas simple de las 3, ya que únicamente se establece el modelo de la red neuronal y se arranca, no hay mas que el uso de una sola clase.

Las vistas de generación de datos y jugar es prácticamente la misma, únicamente se diferencian en quienes son los jugadores de cada partida y el numero de partidas que se juegan.

Para la opción de generación de datos el main llaman a la clase DataGeneratorController, que se encarga de hacer las llamadas a la clase PartidaController y guardar los datos en un archivo.

Para la opción de Jugar se llama desde el main directamente a la clase PartidaController que es la encargada de gestionar el ciclo completo de ejecución de una partida.

### Ciclo de una partida

Como acabamos de ver la clase PartidaController se encarga de gestionar la partida con un bucle que realiza una ronda completa hasta que uno de los 2 jugadores gana.

Cada ronda se compone de a su vez de un bucle de 4 turnos tal y como se ve representado en la imagen 5.3.2.a.

Al inicio de cada ronda la clase PartidaController pide a la clase TableroController

En cada ciclo del turno cada jugador realiza una acción, siempre en el mismo orden de jugadores para cada turno y intercambiando el orden para cada ronda. El jugador inicial recibe el estado actual del tablero y decide que acción realizar. Si la acción es la acción de “secreto” o la acción de “renuncia” (es decir, es una acción simple) esta acción llega a la clase PartidaController que se encarga de enviarla a la clase TableroController para que haga los cambios correspondientes en el tablero y devolvérselo a la clase PartidaController. Si la acción es de “regalo” o “competición” (es decir, una acción compleja) después de que el jugador realice la acción y la clase TableroController devuelva el resultado en el tablero igual que en una acción simple, la clase PartidaController envía la acción de decisión en el tablero al jugador 2 para que elija las cartas, este una vez ha elegido las cartas le devuelve esa acción a la clase PartidaController y esta vuelve a enviársela a la clase TableroController para que guarde dicha decisión en el tablero.

Una vez terminada la acción del jugador 1 se realiza la acción del jugador 2.

A modo de resumen, cada partida consta de al menos 1 ronda. Cada ronda consta de 4 turnos en los que se realizan 2 acciones, una por cada jugador.

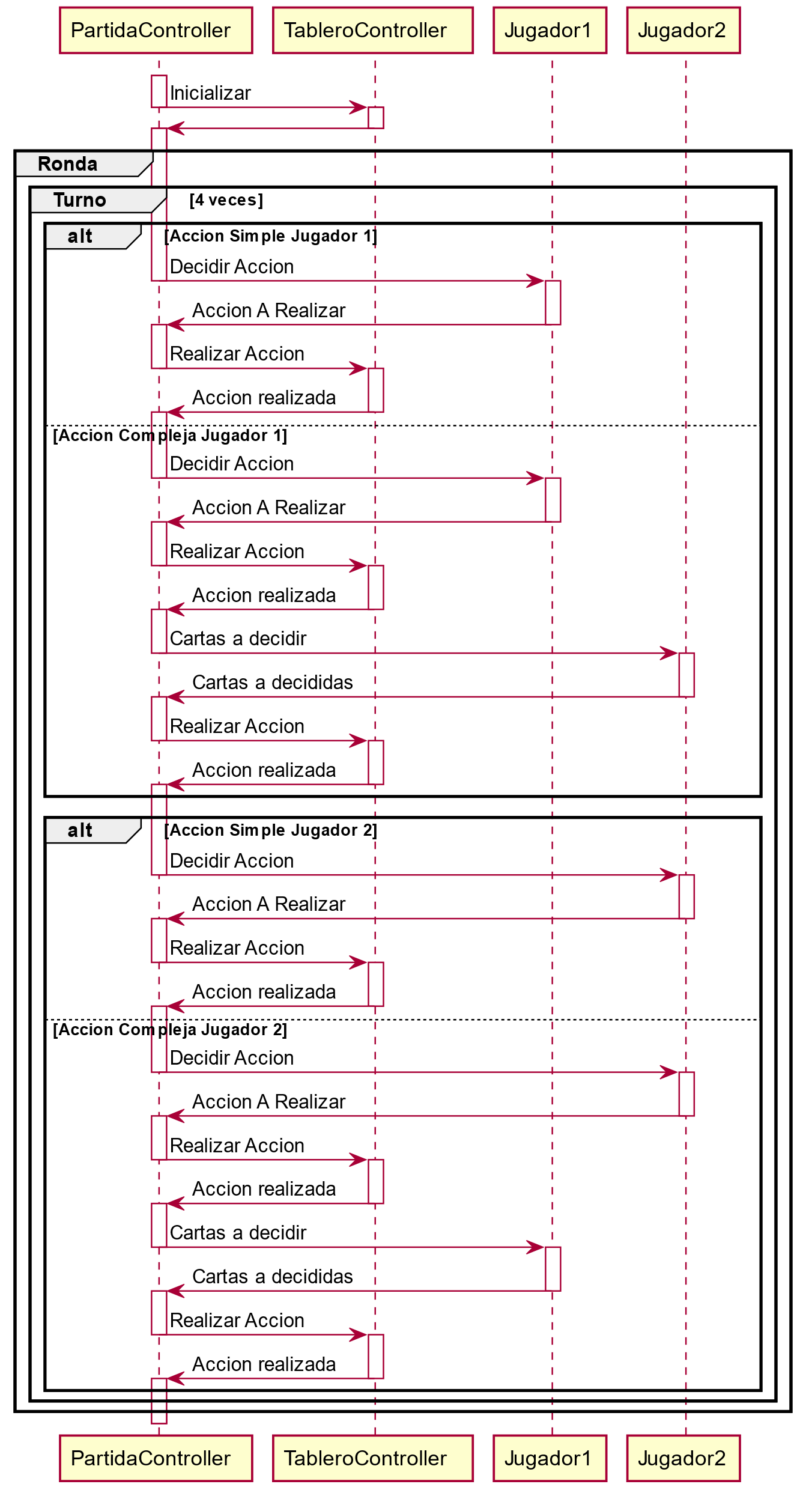


Imagen 5.3.2.a

## Bibliotecas y software externo

Work in progres…