Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Programación Orientada a Objetos

Prof.: Mauricio Avilés

# Proyecto 2 - Modelado

## El Maharajá y los cipayos

Tomáš Apeltauer

Saúl Zamora

I Semestre

2014

# Tabla de contenido

# Introducción

El modelado es una parte central de todas las actividades que tienen como objetivo la producción de buen software. Se construyen modelos para:

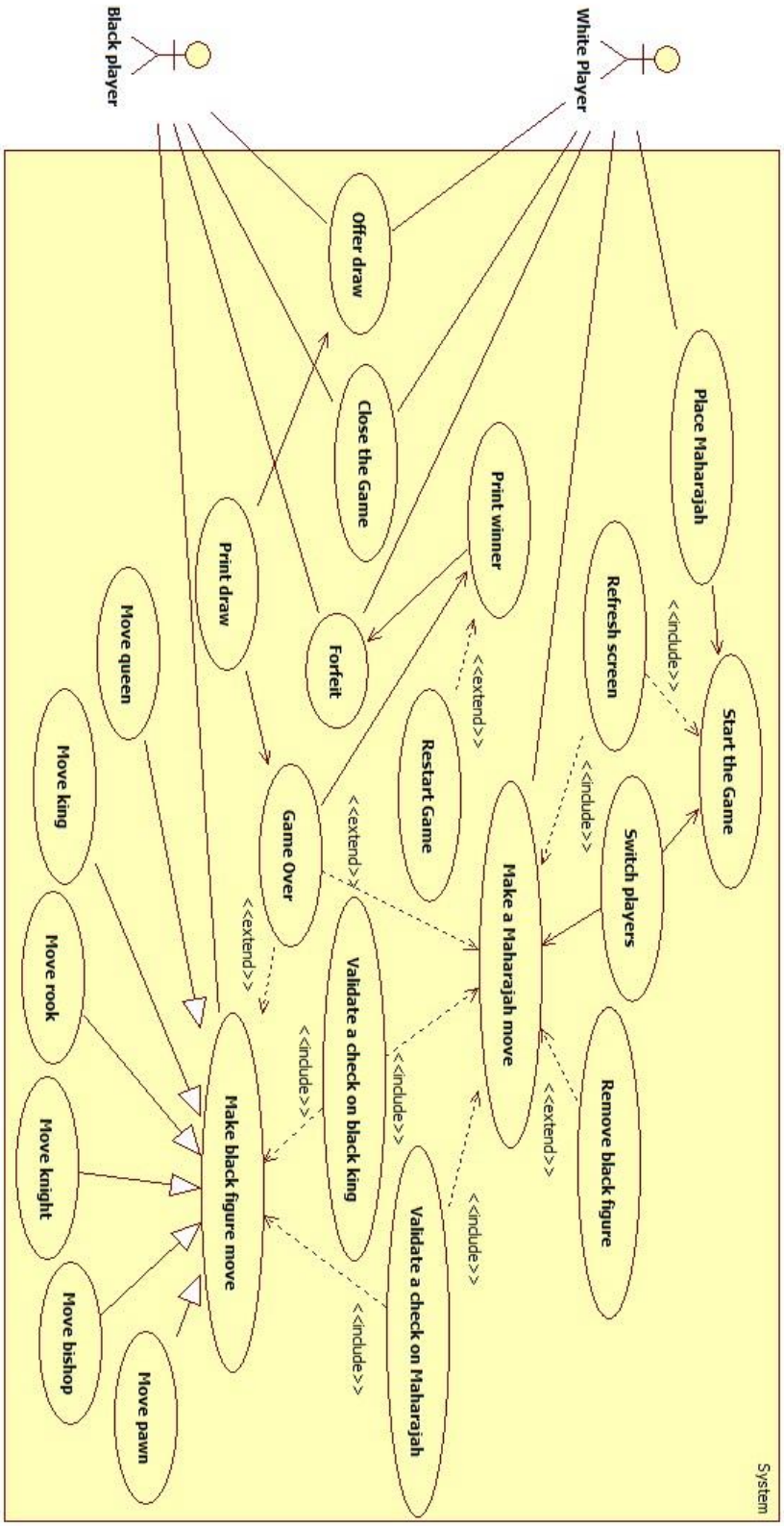
- Comunicar la estructura deseada y el comportamiento del sistema.
- Visualizar y controlar la arquitectura del sistema.
- Comprender lo que se está construyendo, muchas veces descubriendo oportunidades para la simplificación y reutilización.
- Controlar riesgos.

El modelado es común en los proyectos de software exitosos. Es una técnica de ingeniería probada y aceptada que nos ayuda a:

- Mostrar a los usuarios una visualización del producto final.
- Comprender mejor el sistema.
- Comunicar ideas a otros.

Con base en esas ideas, este proyecto tiene como objetivo fundamental, inculcar la importancia de un buen modelo de software como base para un proyecto, sin importar su tamaño o magnitud.

## Diagrama de casos de uso



# Descripción de casos de uso

## Ubicar Maharajá

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- El jugador blanco ubica su única pieza (el Maharajá) en la posición de su preferencia antes de iniciar el juego.

Precondiciones

- El juego no debe haber empezado todavía.

Condición de terminación exitosa

- El Maharajá es ubicado en una posición válida en el tablero.

Condición de terminación fallida

- La posición escogida por el jugador blanco NO es válida. El Maharajá no es ubicado.

Actores Primarios

- Jugador Blanco

Actores secundarios

- Sistema

Desencadenador

- El jugador solicita el inicio del juego.

Casos de uso incluidos

- Ninguno

Flujo principal

1. El jugador (cualquiera) solicita el inicio del juego.
2. El jugador blanco ubica al Maharajá en una posición válida.
3. El juego inicia.

Extensiones

- Ninguna

## Retirarse

Requerimientos relacionados

- Ninguno

Meta en el contexto

- Abandonar el juego.

Precondiciones

- Ninguna

Condición de terminación exitosa

- El jugador abandona el juego. Automáticamente, el otro jugador gana.

Condición de terminación fallida

- Ninguna.

Actores Primarios

- Jugador.

Actores secundarios

- Ninguno

Desencadenador

- El jugador solicita retirarse del juego.

Casos de uso incluidos

- Anunciar ganador

Flujo principal

1. El jugador (cualquiera) solicita retirarse del juego.
2. El jugador restante es proclamado ganador.
3. El juego termina.

Extensiones

- Ninguna

## Ofrecer empate

Requerimientos relacionados

- Ambos jugadores deben aceptar.

Meta en el contexto

- El jugador (cualquiera) ofrece al otro jugador la opción de finalizar el juego en un empate.

Precondiciones

- Ninguna

Condición de terminación exitosa

- El juego finaliza en empate.

Condición de terminación fallida

- La solicitud de empate es rechazada. El juego continúa.

Actores Primarios

- Jugadores

Actores secundarios

- Ninguno

Desencadenador

- Un jugador solicita el empate al otro.

Casos de uso incluidos

- Ninguno

Flujo principal

1. El jugador (cualquiera) solicita el empate.
2. Hay dos opciones:
    a. El otro jugador acepta el empate.
        i. El juego termina.
    b. El otro jugador rechaza el empata. El juego continúa.

Extensiones

- Anunciar empate.

## Cerrar el juego

Requerimientos relacionados

- Ninguno

Meta en el contexto

- El jugador (cualquiera) cierra la ventana del juego.

Precondiciones

- Ninguna

Condición de terminación exitosa

- El juego termina.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Jugador (cualquiera)

Actores secundarios

- Ninguno

Desencadenador

- Un jugador cierra la ventana del juego.

Casos de uso incluidos

- Ninguno

Flujo principal

1. El jugador (cualquiera) cierra la ventana del juego.
2. El juego termina.

Extensiones

- Ninguna

## Juego finalizado

Requerimientos relacionados

- Algún evento que provoque el final del juego.

Meta en el contexto

- El juego finaliza correctamente al presentarse alguna de las siguientes precondiciones.

Precondiciones

- *Checkmate* en el rey negro o
- El jugador negro elimina (*checkmate*) el Maharajá o
- Una oferta de empate es aceptada o
- Un jugador abandona el juego.

Condición de terminación exitosa

- El juego finaliza correctamente.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.
- Jugadores.

Actores secundarios

- Ninguno

Desencadenador

- Se cumple alguna de las precondiciones.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Se verifica alguna de las precondiciones.
2. Se anuncia el resultado del juego.

Extensiones

- Anunciar ganador.

## Anunciar ganador

Requerimientos relacionados

- El juego debe haber finalizado.

Meta en el contexto

- Anunciar el resultado del juego (ganador o empate).

Precondiciones

- El juego finalizó correctamente.

Condición de terminación exitosa

- Se anuncia al ganador o se anuncia el empate.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.

Actores secundarios

- Ninguno

Desencadenador

- El juego finalizó correctamente.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Se anuncia el resultado del juego.
2. El juego finaliza.

Extensiones

- Reiniciar juego.

## Reiniciar juego

Requerimientos relacionados

- El juego debe haber finalizado correctamente.

Meta en el contexto

- Reiniciar el juego.

Precondiciones

- Ninguna.

Condición de terminación exitosa

- El juego empieza de nuevo.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Jugador (cualquiera).

Actores secundarios

- Ninguno

Desencadenador

- Un jugador solicita reiniciar el juego.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Un jugador solicita reiniciar el juego.
2. Si el otro jugador acepta, el juego inicia de nuevo, en caso contrario el juego continúa.

Extensiones

- Ninguna.

## Anunciar empate

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Anuncia que la partida ha finalizado en empate.

Precondiciones

- Ninguna.

Condición de terminación exitosa

- Se anuncia el empate.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.

Actores secundarios

- Ninguno

Desencadenador

- La solicitud de empate es aceptada o el juego ha finalizado en empate.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Se anuncia el empate.
2. El juego termina.

Extensiones

- Ninguna.

## Iniciar juego

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Empezar a jugar.

Precondiciones

- El Maharajá ha sido ubicado en una posición válida.

Condición de terminación exitosa

- El juego inicia.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Jugador blanco.

Actores secundarios

- Ninguno

Desencadenador

- El jugador blanco inicia el juego luego de ubicar su única pieza (el Maharajá) en una posición válida.

Casos de uso incluidos

- Actualizar pantalla.

Flujo principal

1. El Maharajá es ubicado en una posición válida por el jugador blanco.
2. El jugador blanco inicia el juego.

Extensiones

- Ninguna.

## Actualizar pantalla

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Después de que algún movimiento sea realizado por cualquiera de los jugadores, se actualiza la pantalla con los cambios realizados.

Precondiciones

- Un cambio es realizado.

Condición de terminación exitosa

- Se actualiza la pantalla (interfaz gráfica) con los nuevos cambios.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.

Actores secundarios

- Ninguno

Desencadenador

- Un movimiento por parte de alguno de los jugadores.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Se realiza un cambio.
2. Se actualiza la pantalla.

Extensiones

- Ninguna.

## Cambiar jugador

Requerimientos relacionados

- El turno del jugador actual debe finalizar correctamente.

Meta en el contexto

- Cambia el jugador activo (de blanco al negro y viceversa) para el siguiente turno.

Precondiciones

- Ninguna.

Condición de terminación exitosa

- Se cambia el jugador activo para el siguiente turno.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.

Actores secundarios

- Ninguno

Desencadenador

- El turno anterior finalizó.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Acaba el turno del jugador anterior.
2. Se cambia el jugador activo para el siguiente turno.

Extensiones

- Ninguna.

## Mover Maharajá

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Hacer el Maharajá se mueva a una nueva posición válida.

Precondiciones

- Ninguna.

Condición de terminación exitosa

- El Maharajá se mueve.

Condición de terminación fallida

- El Maharajá no se mueve.

Actores Primarios

- Jugador blanco.

Actores secundarios

- Ninguno

Desencadenador

- El turno del jugador blanco inicia y este realiza un movimiento.

Casos de uso incluidos

- Actualizar pantalla.
- Validar *check* en el Maharajá.
- Validar *check* en el rey negro.

Flujo principal

1. El jugador blanco mueve el Maharajá de posición.
   include::Validar *check* en el Maharajá.
   a. Se revisa si la nueva posición del Maharajá es un *check* en sí mismo.
   b. El movimiento se rechaza.
   include::Validar *check* en el rey negro.
   a. Se revisa si la nueva posición crea un estado de *check* en el rey del jugador negro.
   b. El movimiento es aceptado.
   extend::Juego finalizado

a. Se revisa si la nueva posición del Maharajá crea un estado de *checkmate* en el rey negro.
b. De existir *checkmate*, el juego finaliza, en caso contrario, el juego continúa.

extend::Remover figura negra

a. Se revisa si la nueva posición está ocupada por una pieza del jugador negro.
b. Se remueve la pieza del jugador negro.
c. Se acepta el movimiento.

Extensiones

- Remover figura negra.
- Juego finalizado.

## Remover figura negra

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Remover una figura negra de la pantalla.

Precondiciones

- El Maharajá debe moverse a una casilla ocupada por una figura negra.

Condición de terminación exitosa

- La figura en cuestión es eliminada de la pantalla.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.

Actores secundarios

- Ninguno

Desencadenador

- El Maharajá es movido a una posición ocupada por una figura negra.

Casos de uso incluidos

- Ninguno

Flujo principal

3. El Maharajá es ubicado a una posición ocupada.
4. La figura negra es removida.

Extensiones

- Ninguna.

## Validar *check* en el Maharajá

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Revisar si luego de un movimiento de cualquiera de los jugadores, se crea un estado de *check* en el Maharajá.

Precondiciones

- El jugador actual debe realizar su movida.

Condición de terminación exitosa

- Se anuncia el estado de *check* en el Maharajá.
- Según el jugador activo:
    - Jugador blanco: se rechaza la nueva ubicación del Maharajá, se solicita mover de nuevo.
    - Jugador negro: el juego continúa.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.

Actores secundarios

- Ninguno

Desencadenador

- Se realiza un movimiento.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Se realiza un movimiento.
2. Se revisa si se crea un estado de *check* en el Maharajá.

Extensiones

- Ninguna.

## Validar *check* en el rey negro

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Revisar si se crea un estado de *check* en el rey negro luego de un movimiento.

Precondiciones

- El jugador actual debe realizar su movida.

Condición de terminación exitosa

- Se anuncia el estado de *check* sobre el rey negro.
- Según jugador activo:
    - Jugador blanco: el juego continúa.
    - Jugador negro: se rechaza el movimiento, se solicita un nuevo movimiento.

Condición de terminación fallida

- Ninguna

Actores Primarios

- Sistema.

Actores secundarios

- Ninguno

Desencadenador

- Alguno de los jugadores realiza un movimiento.

Casos de uso incluidos

- Ninguno

Flujo principal

1. Se realiza un movimiento.
2. Se revisa si se crea un estado de *check* en el rey negro.

Extensiones

- Ninguna.

## Mover figura negra (reina, rey, torre, alfil, caballo o peón).

Requerimientos relacionados

- Ninguno.

Meta en el contexto

- Realizar un movimiento por parte del jugador negro.

Precondiciones

- El jugador actual debe ser el jugador negro.

Condición de terminación exitosa

- La pieza seleccionada se mueve a una nueva posición.

Condición de terminación fallida

- El movimiento NO se realiza.

Actores Primarios

- Jugador negro.

Actores secundarios

- Ninguno

Desencadenador

- El jugador negro realiza un movimiento.

Casos de uso incluidos

- Validar *check* en el Maharajá.
- Validar *check* en el rey negro.

Flujo principal

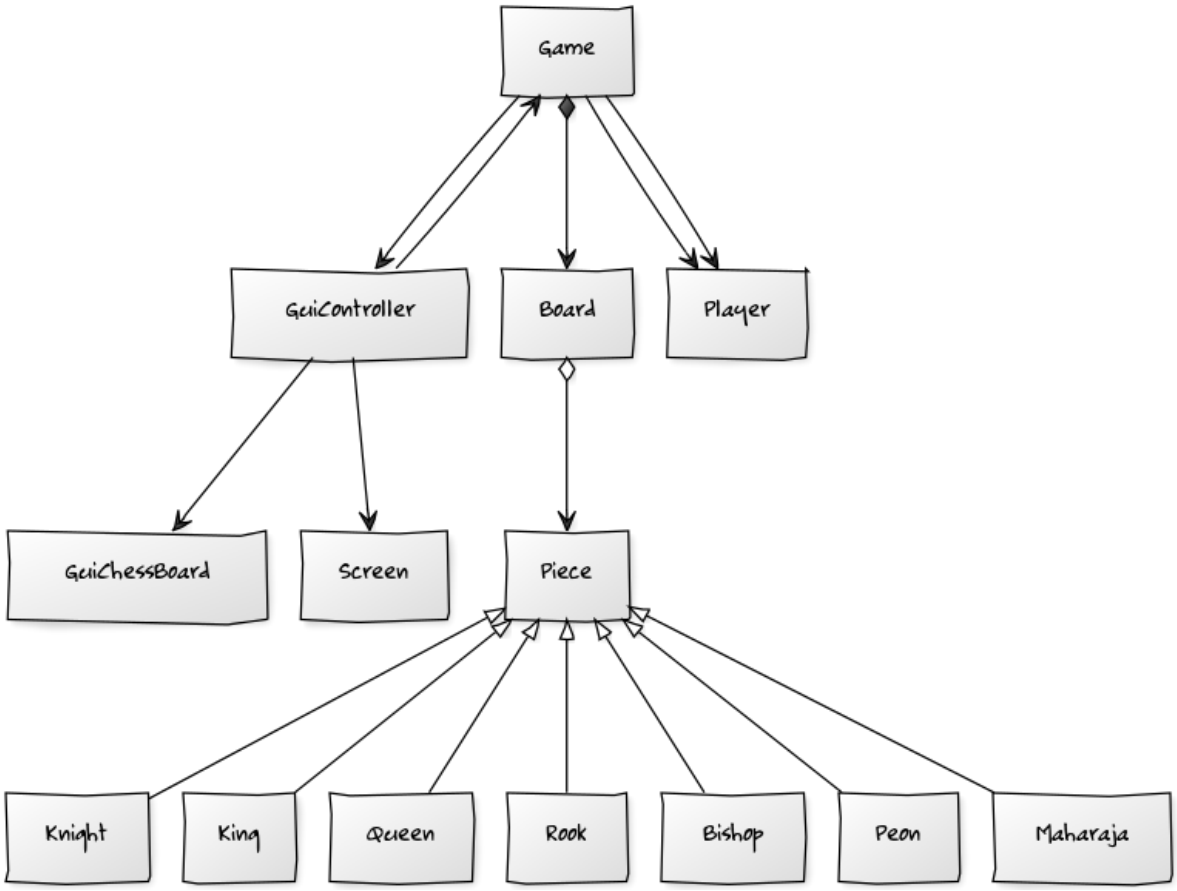
1. El jugador negro mueve una pieza.
   include::Validar *check* en el Maharajá.
   a. Se revisa si la nueva posición del Maharajá es un *check* en sí mismo.
   b. El movimiento es aceptado.
   include::Validar *check* en el rey negro.
   c. Se revisa si la nueva posición crea un estado de *check* en el rey del jugador negro.
   d. El movimiento se rechaza.
   extend::Juego finalizado
   c. Se revisa si la nueva ubicación de la pieza crea un estado de *checkmate* en el Maharajá.

d. De existir *checkmate*, el juego finaliza, en caso contrario, el juego continúa.

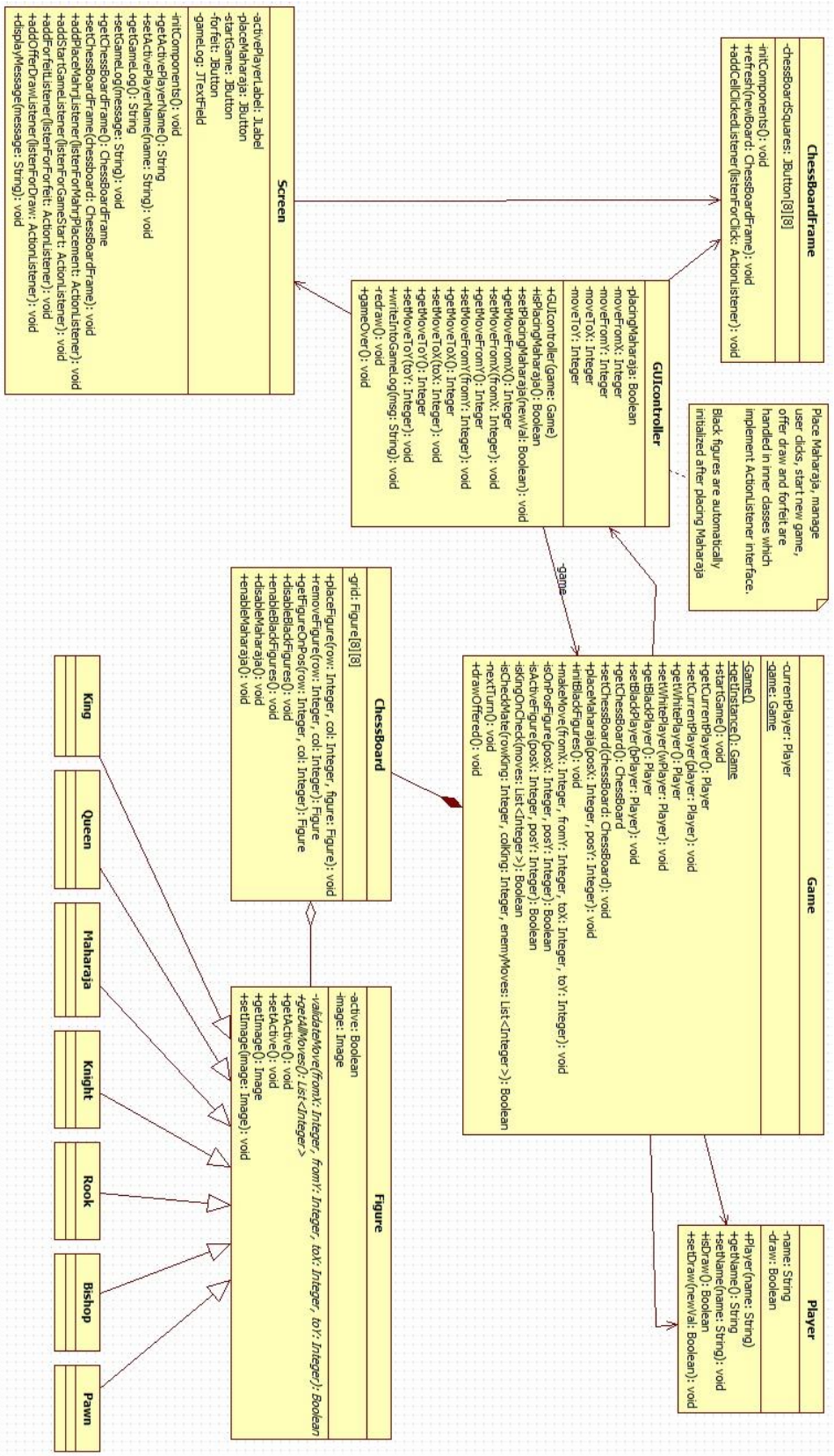
Extensiones

- Juego finalizado.

## Modelo conceptual

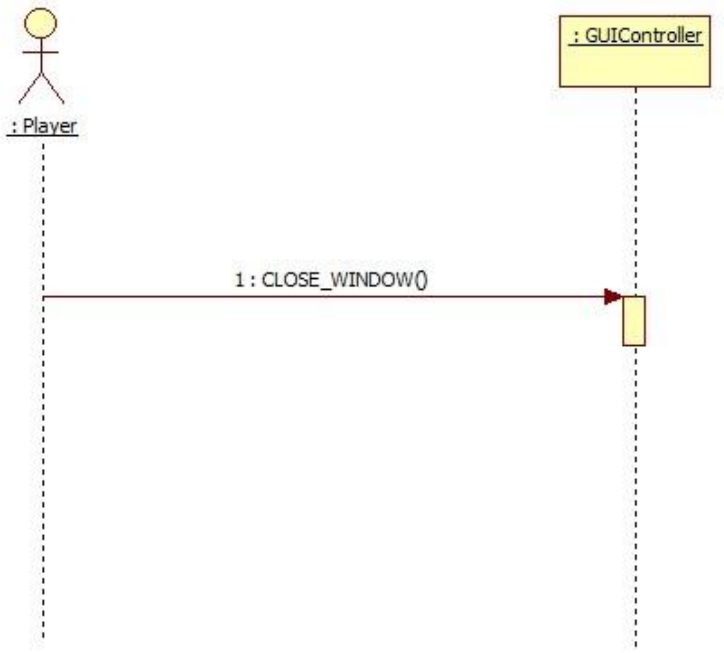


## Diagrama de clases detallado

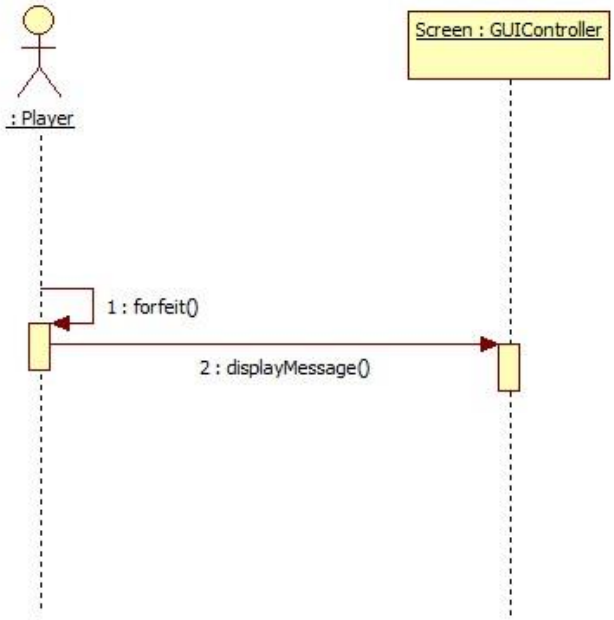


## Diagramas de secuencias

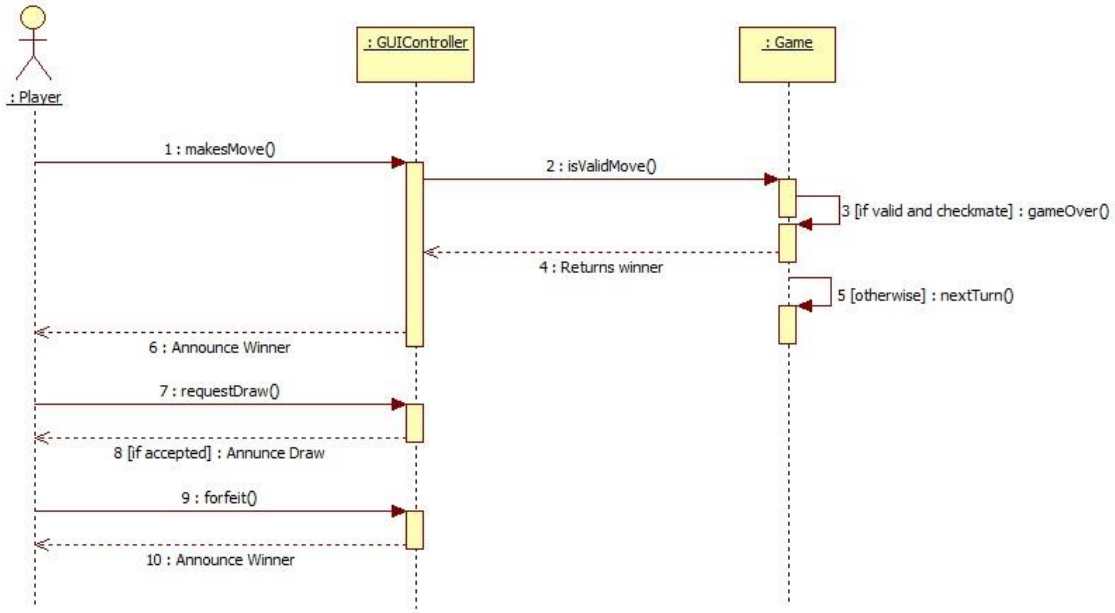
### Cerrar juego



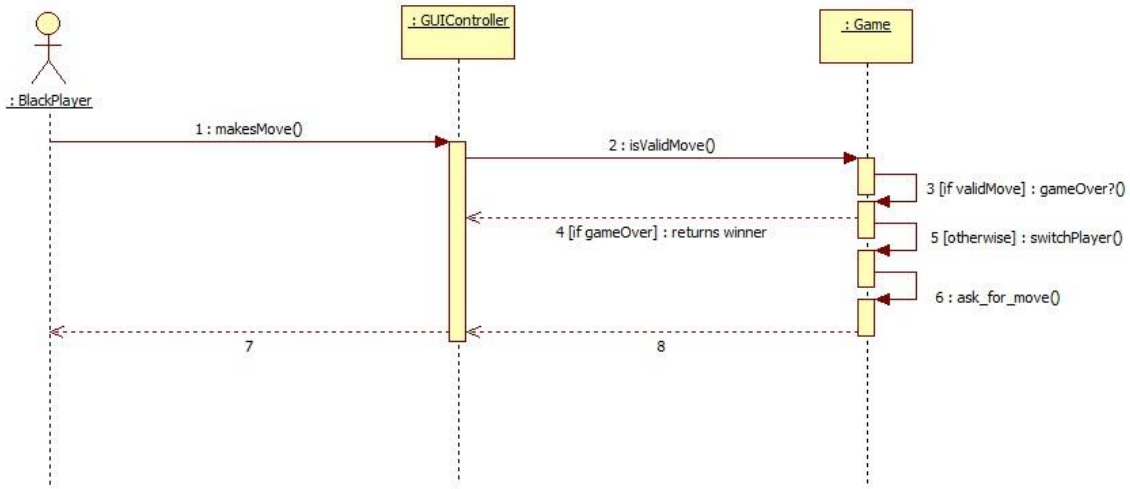

### Rendirse



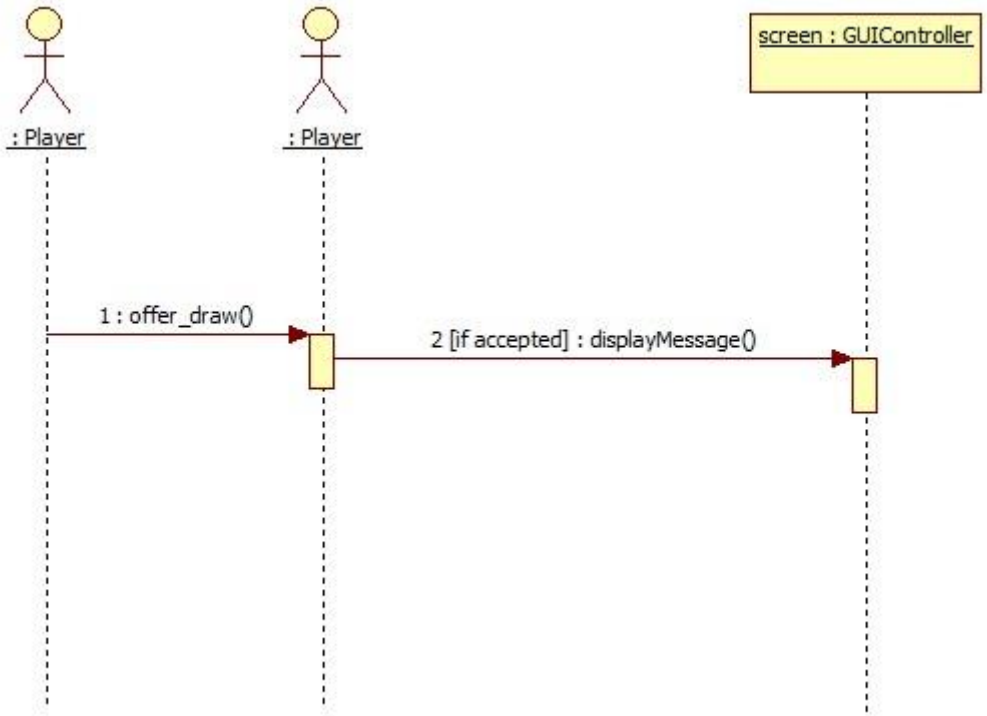
## Juego finalizado



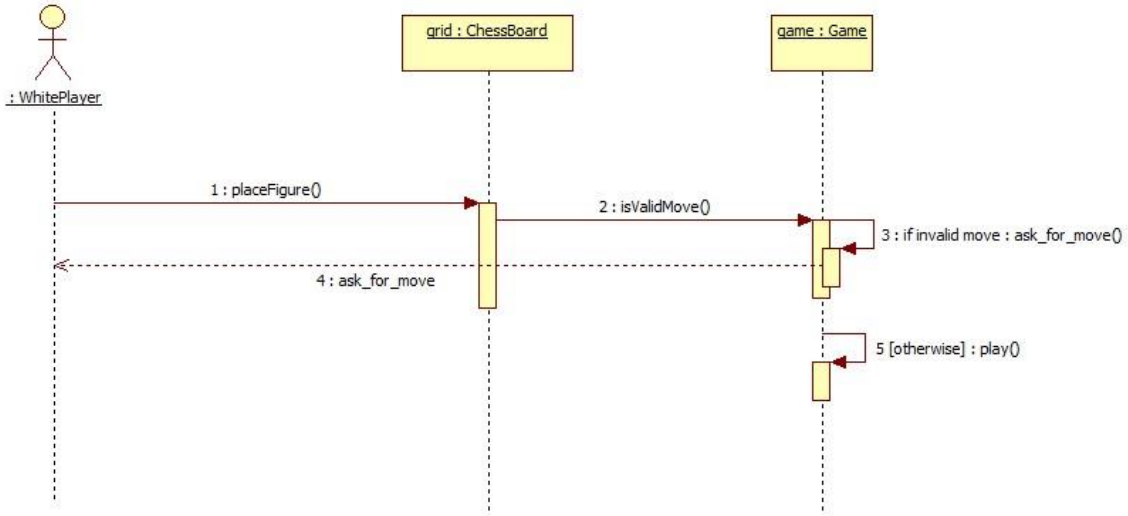

## Mover figura negra



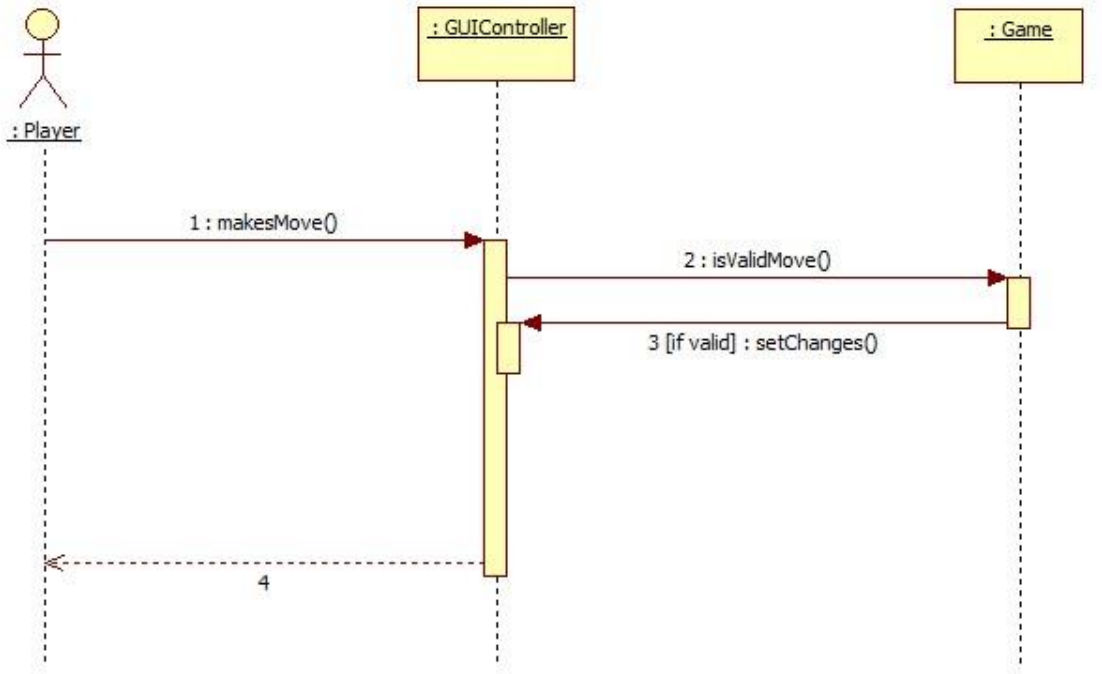
## Ofrecer empate



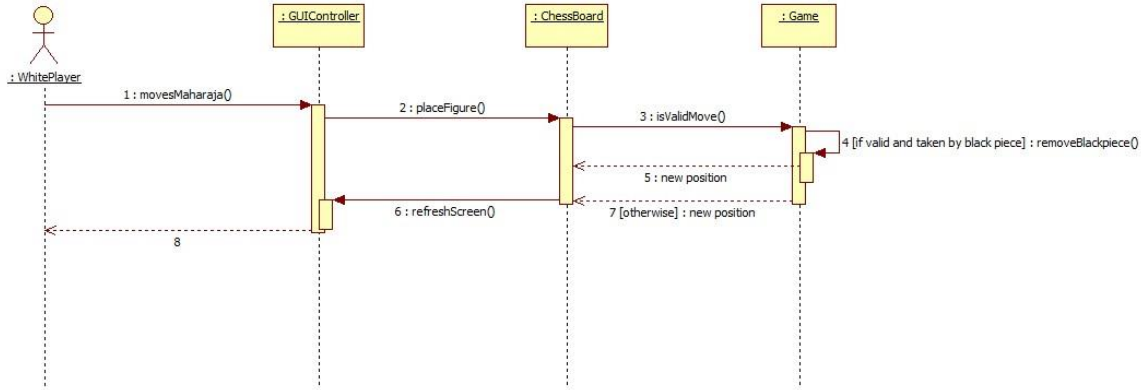

## Colocar Maharajá



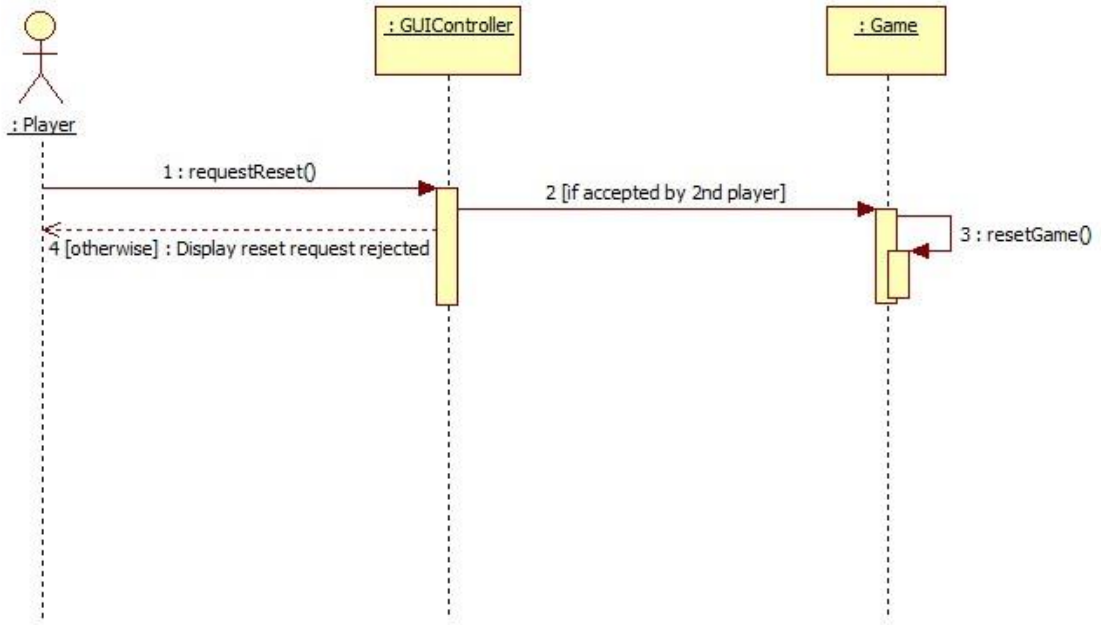
## Actualizar pantalla



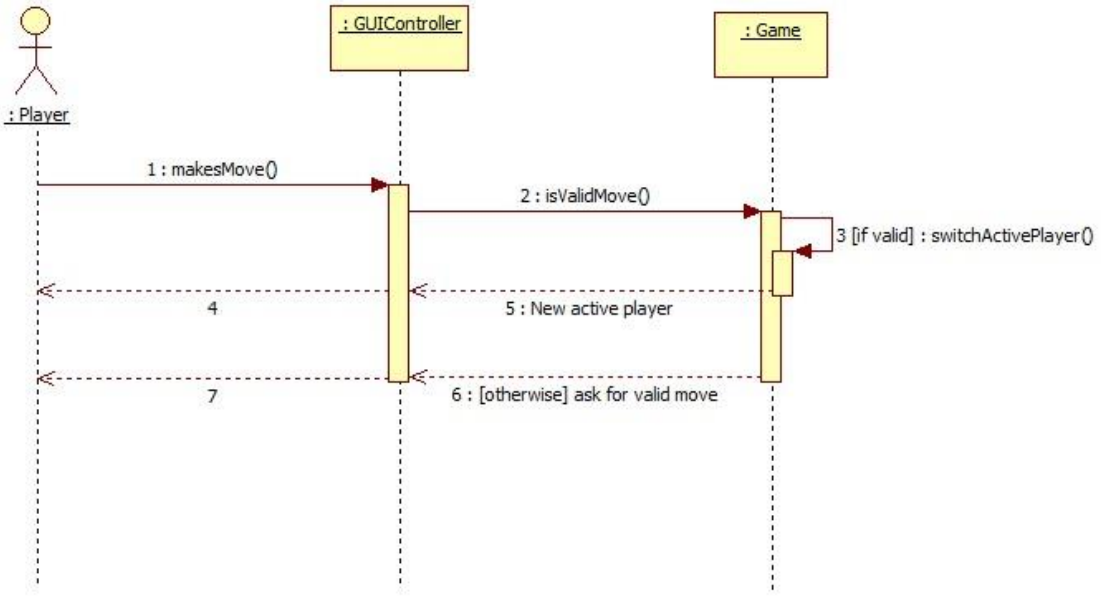

## Remover figura negra



## Reiniciar juego

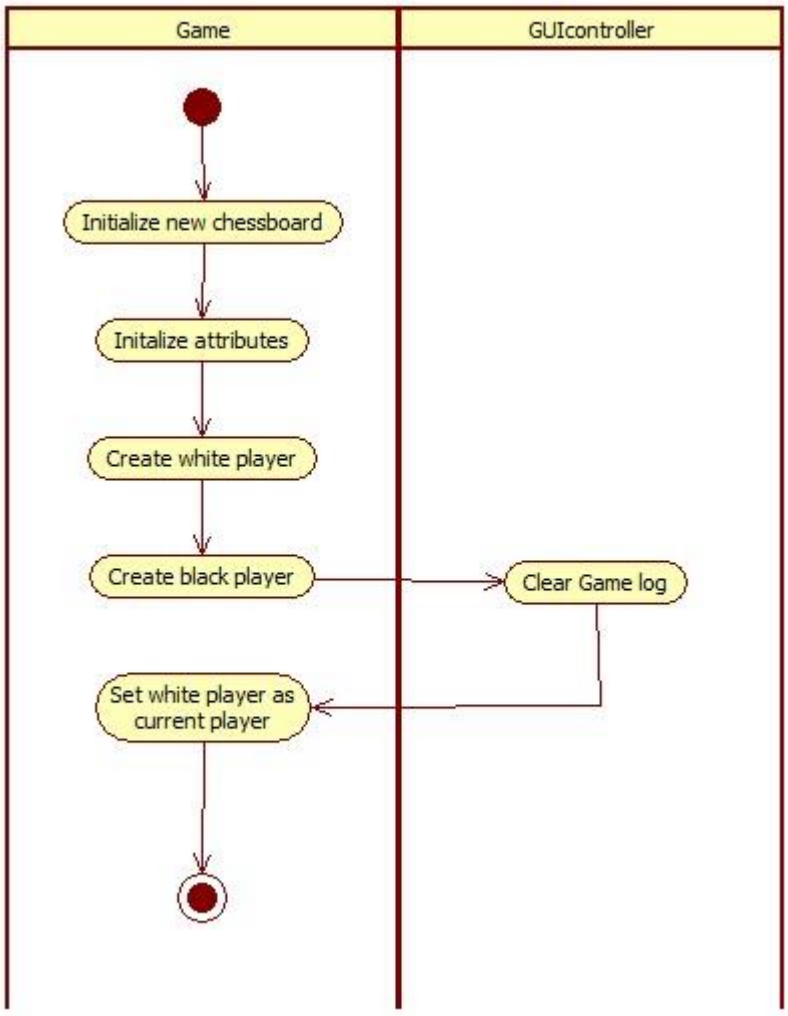



## Cambiar jugador

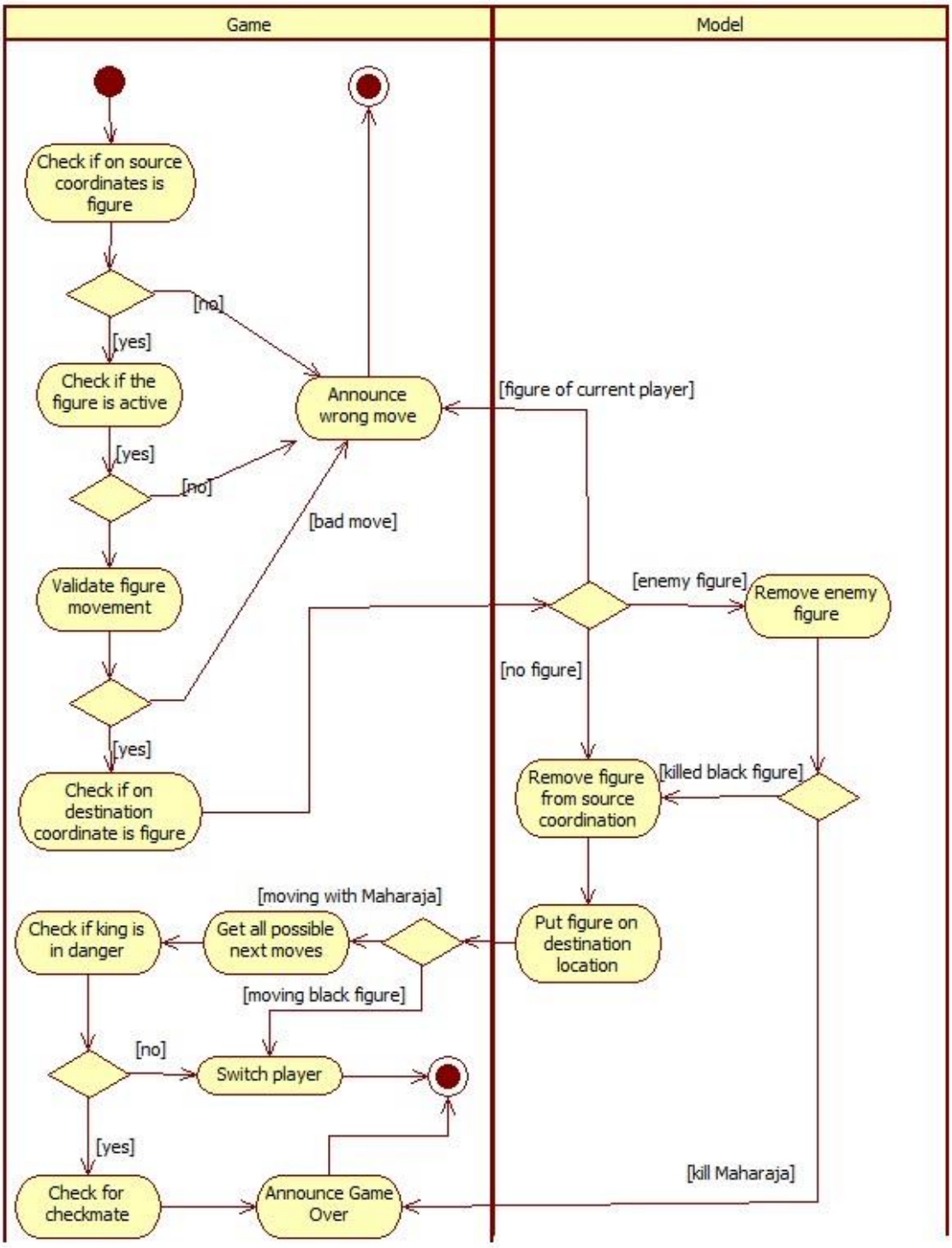


## Diagramas de actividades

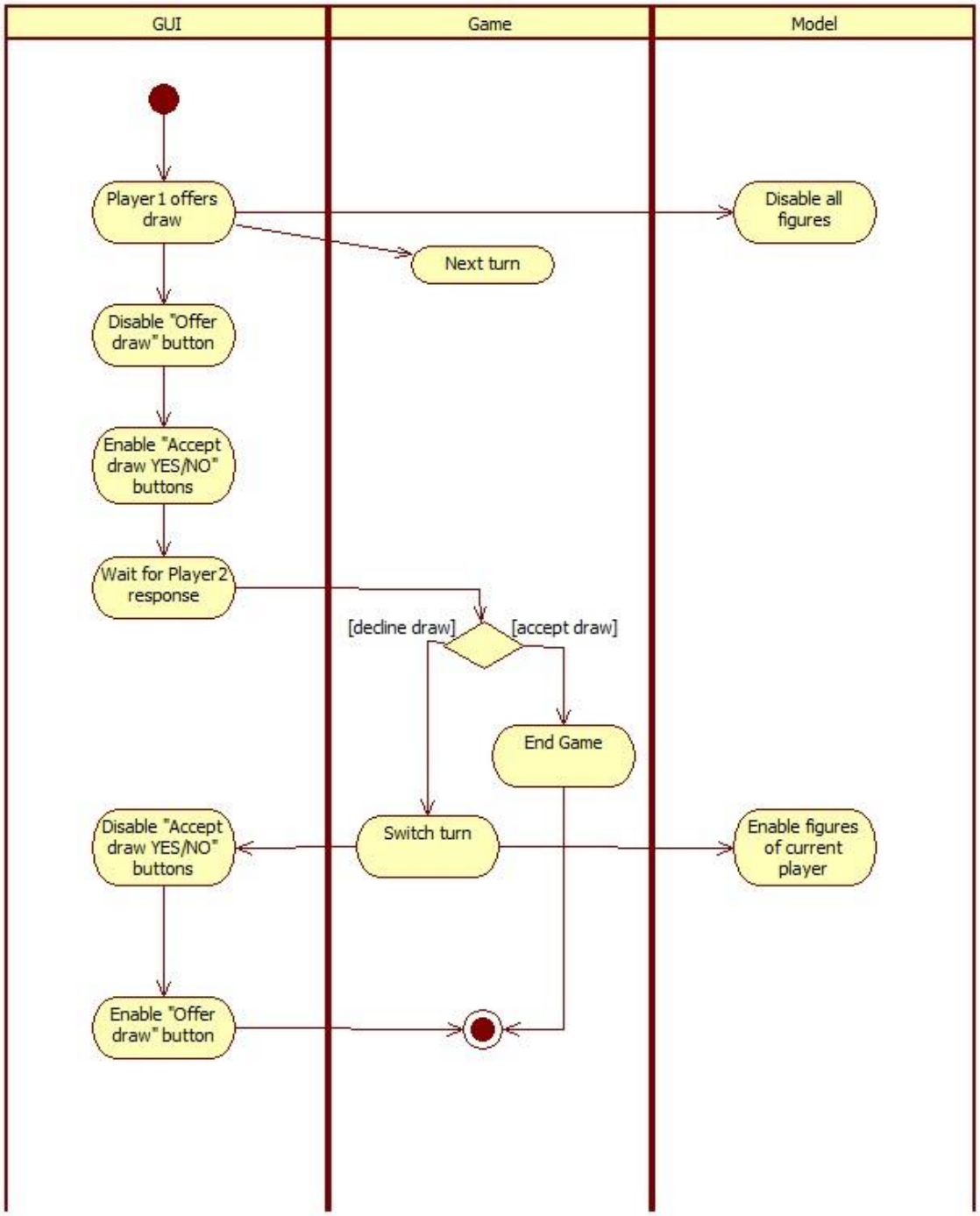
### Iniciar juego



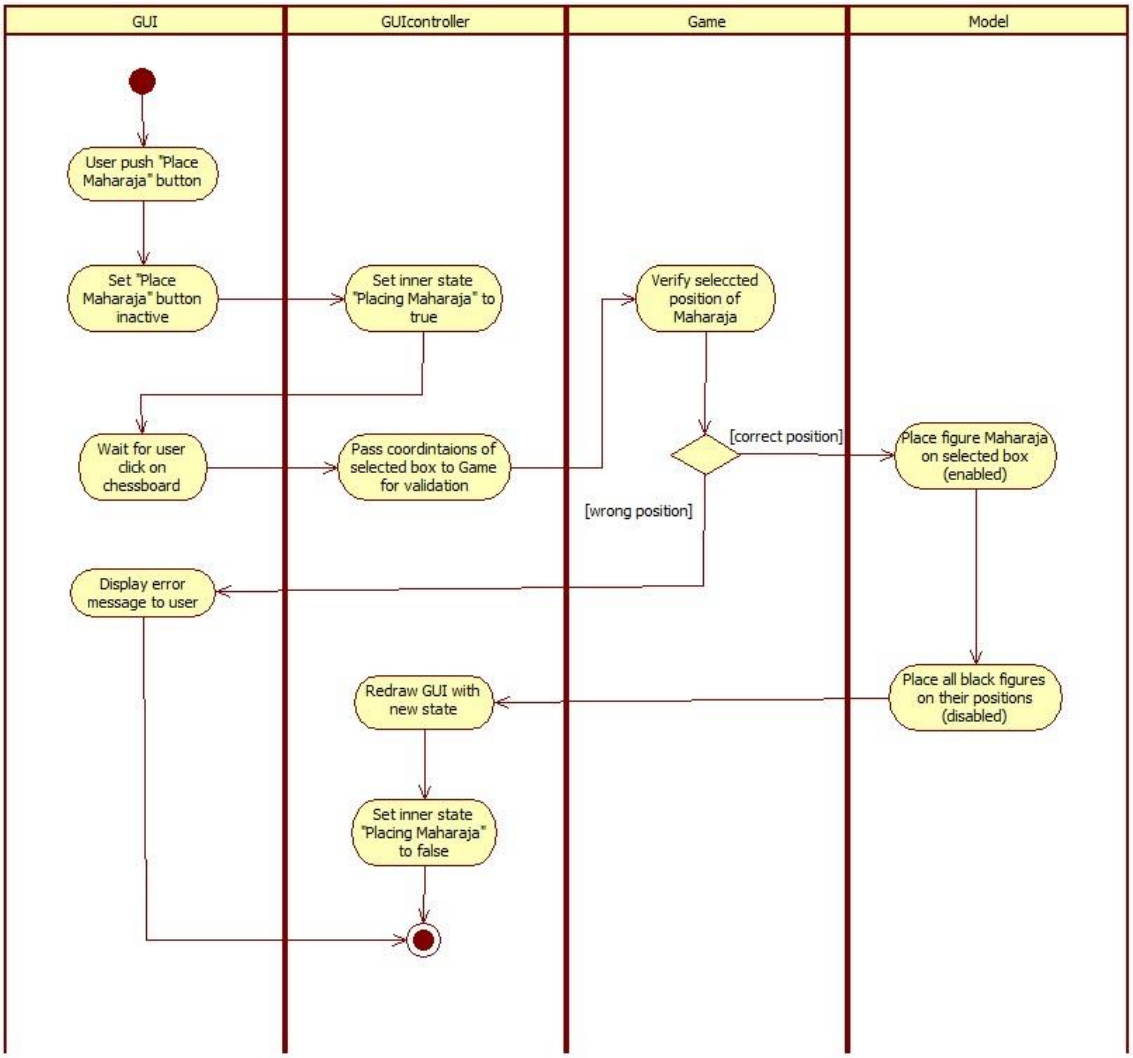
## Mover figura



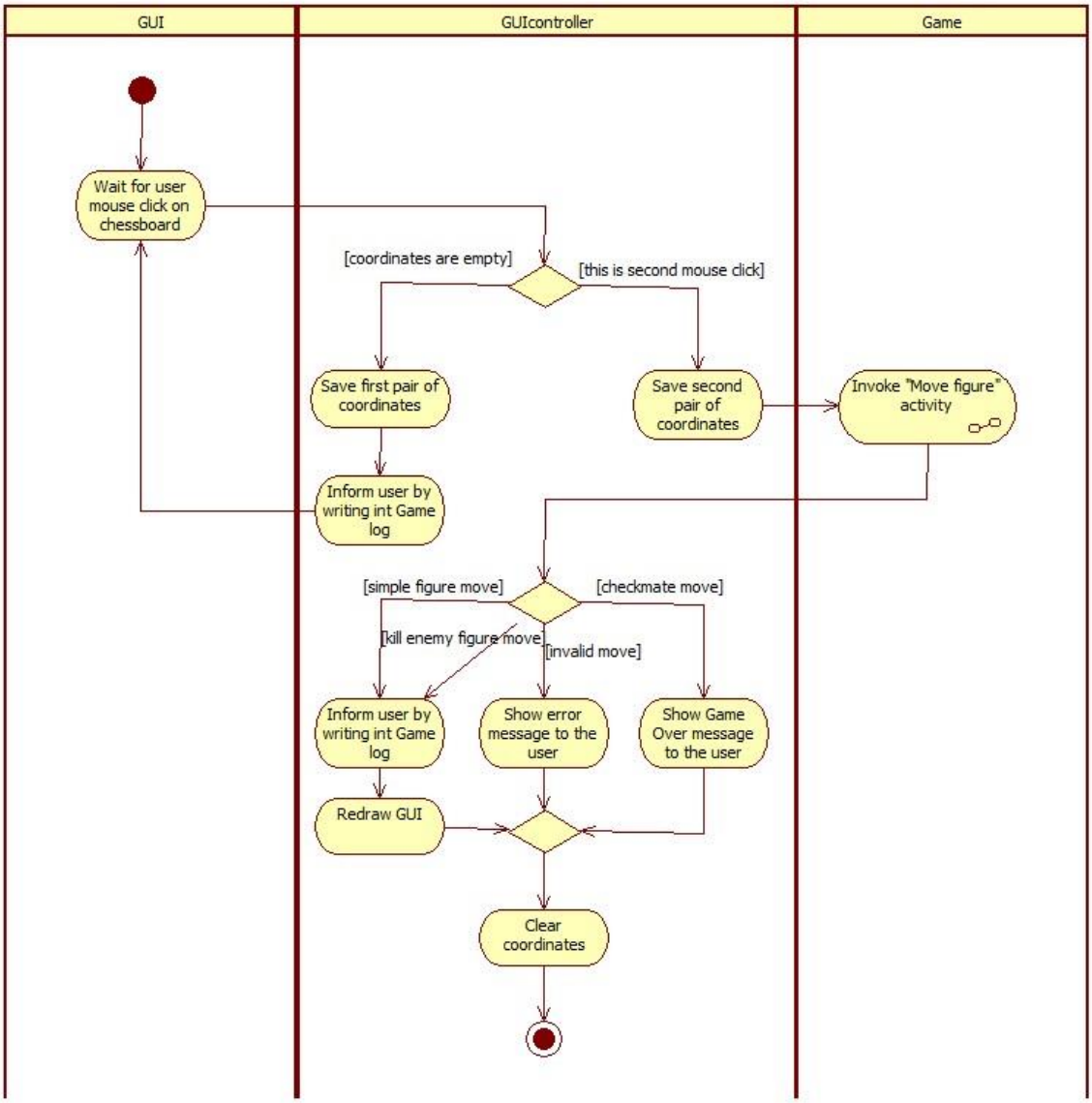
## Ofrecer empate



## Ubicar Maharajá



## Interactuar con el tablero



## Conclusiones

- La elección de un modelo y patrón de diseño influye directamente en cómo se afronte un problema. Es necesario seleccionar el modelo y patrón de diseño adecuado para cada situación; y dependiendo de dicha selección se obtienen distintos resultados.
- Cualquier modelo puede ser expresado con distintos niveles de precisión y detalle.
- Es importante crear un modelo que permita representar la realidad lo más claramente posible.

## Recomendaciones

- Buscar un software de mejor calidad para la elaboración de los diagramas UML.
- Elaborar los diagramas en el siguiente orden: diagrama de clases general, diagrama de casos de uso, diagrama de clases detallado, diagramas de secuencia y diagramas de actividades.
- Al crear el diagrama de clases detallado, es mejor pensar en código (enviar mensajes entre clases, métodos, etc.).
- Al crear diagramas de actividad es bueno crear líneas divisorias para diferenciar las partes del sistema.

## Bibliografía


- Modelado para desarrollo de software. (2013).
  - http://wiki.monagas.udo.edu.ve/index.php/Modelado_para_Desarrollo_de_Software
- Compile and Execute Java online (2014).
  - http://www.compileonline.com/compile_java_online.php
- Activity diagram – Wikipedia, the free encyclopedia. (2014).
  - http://en.wikipedia.org/wiki/Activity_diagram
- Sparx Systems – UML 2 Tutorial – Activity Diagram
  - http://www.sparxsystems.com.au/resources/uml2_tutorial/uml2_activitydiagram.html
- ArrayList of int array in java – Stack Overflow
  - http://stackoverflow.com/questions/10477628/arraylist-of-int-array-in-java
- Create UML diagrams online in seconds, no special tolos needed.
  - http://yuml.me/diagram/scruffy/class/draw