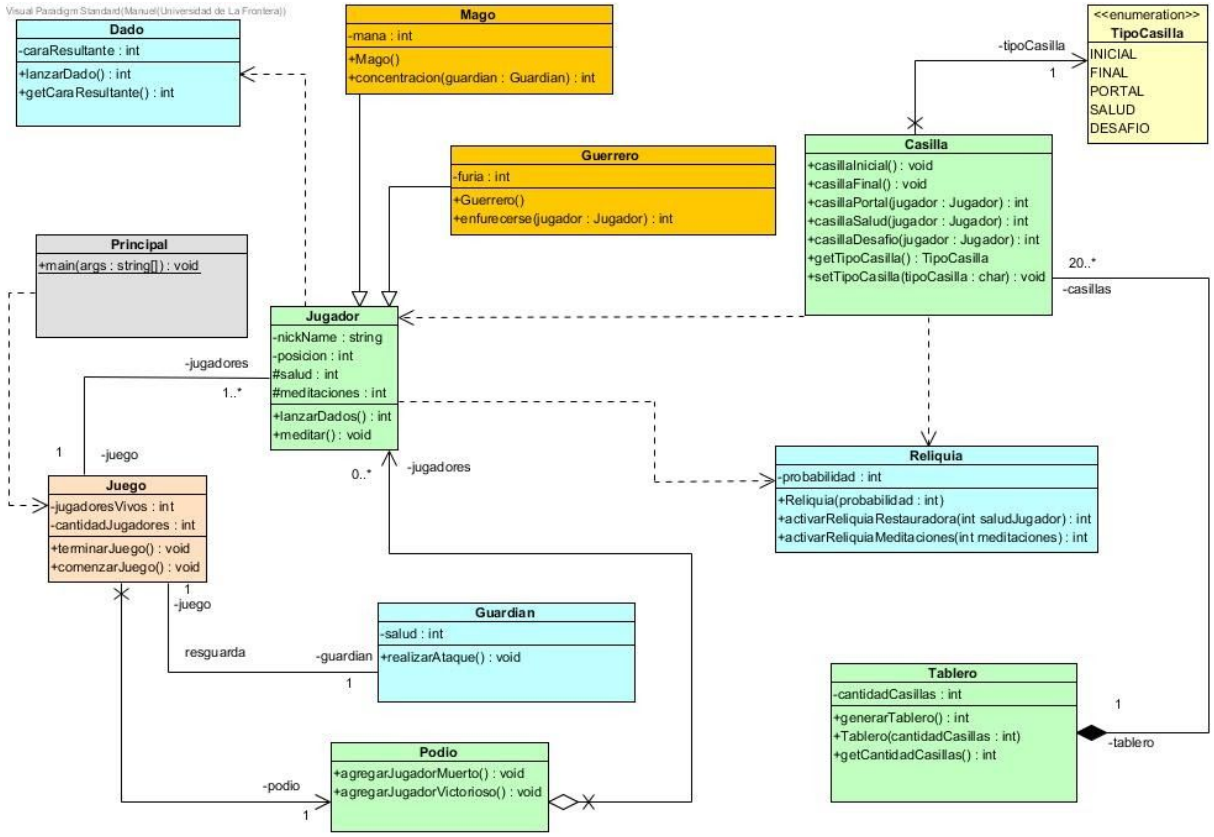
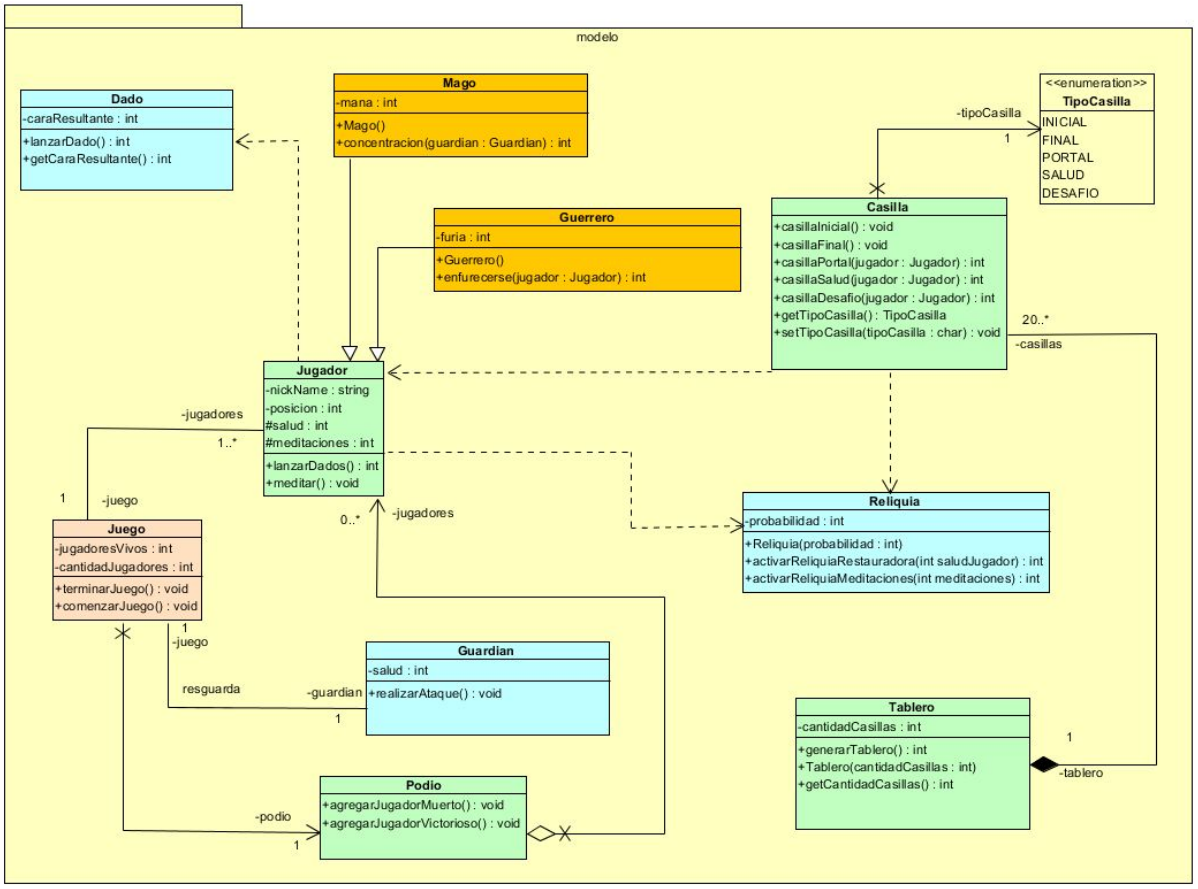
Pauta tarea 9.

En la tarea anterior se llegó al siguiente diagrama:



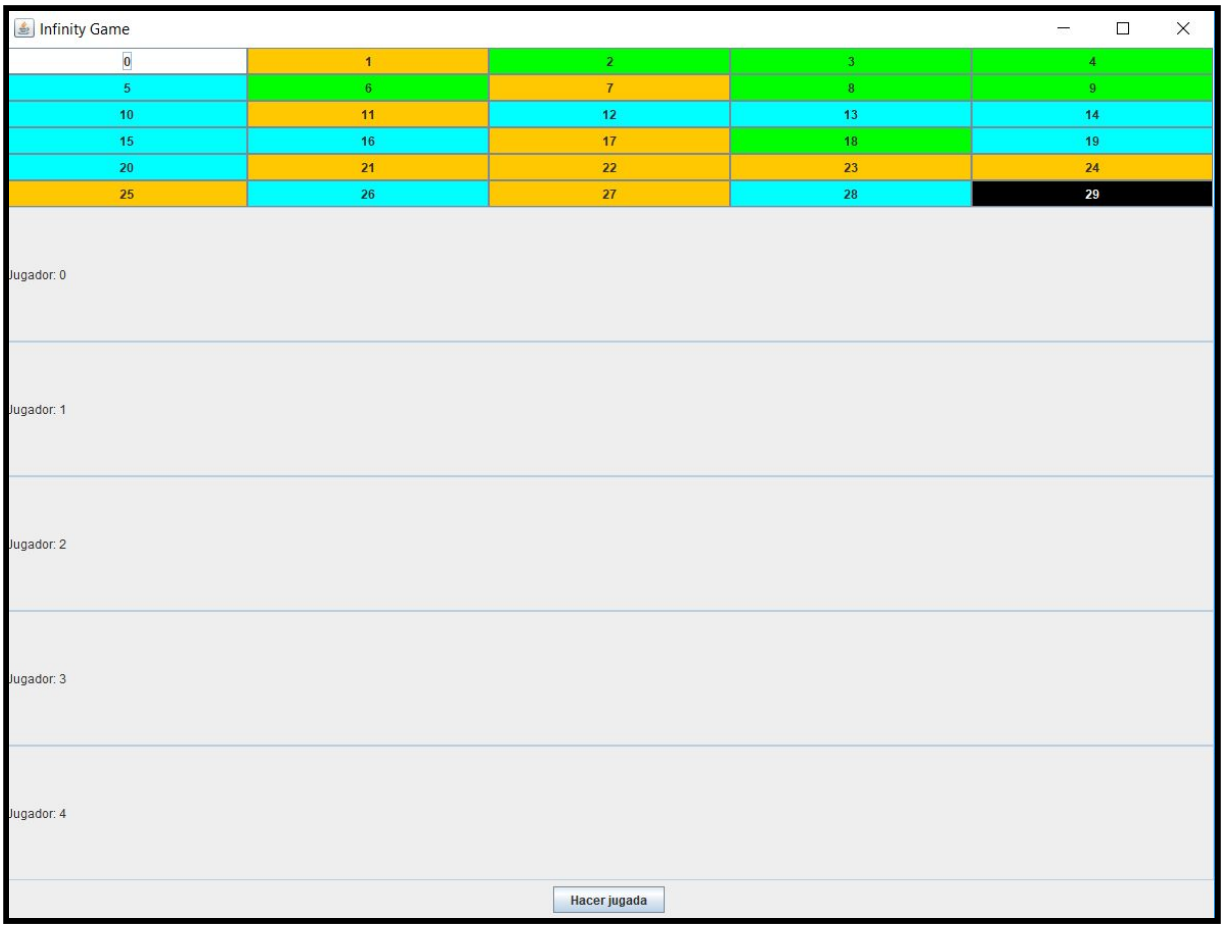
Capa de modelo.

Esto representa la lógica que tendrá el programa, es por aquello que el diagrama que se tiene hasta el momento corresponde a la capa de modelo (con la salvedad que se debe adaptar a la interacción con la GUI).



Capa de vista.

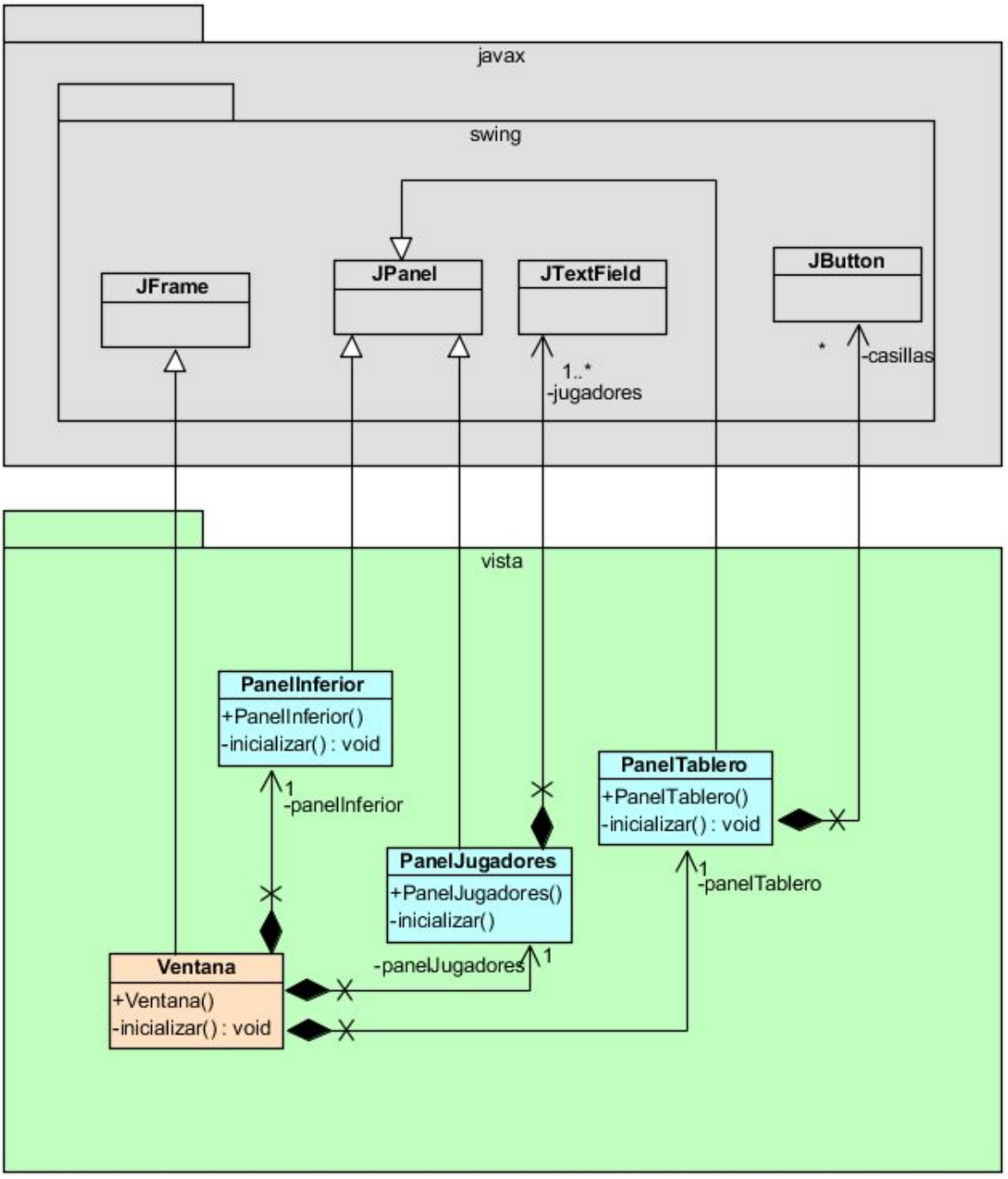
En este caso en particular, se busca un diseño de ventana como se muestra a continuación (ojo, el diseño depende de cada uno):



En este caso, la ventana consta de 3 paneles:

- Panel superior: es el que se encarga de la visualización del tablero. Cada casilla se ve representada por un botón, los cuales varían de color según el tipo de casilla (eso lo debe especificar cada uno).
- Panel central: se encarga de visualizar los jugadores, aquí debería mostrarse la información de cada jugador, como la salud actual, meditaciones, etc. En este caso, solamente se hizo en base a campos de texto (JTextField).
- Panel inferior: contiene la interacción con el usuario respecto a las jugadas. En este caso se incluye sólo un botón, el cual sería el encargado del sistema de turnos.

El diagrama para mostrar lo anterior, sería de la siguiente manera (ignorando los eventos, para ahorrar espacio en el diagrama):


javax
swing
JFrame
JPanel
JTextField
JButton
1..*
-jugadores
*
-casillas
vista
PanelInferior
+PanelInferior()
-inicializar() : void
1
-panelInferior
PanelTablero
+PanelTablero()
-inicializar() : void
PanelJugadores
+PanelJugadores()
-inicializar()
1
-panelTablero
Ventana
+Ventana()
-inicializar() : void
-panelJugadores
1

Código.

Ventana.

```
public class Ventana extends JFrame {

    private PanelJugadores panelInferior;
    private PanelTablero panelTablero;
    private PanelInferior panelIzqueirda;

    public Ventana() {
        inicializar();
    }

    private void inicializar() {

        this.panelIzqueirda = new PanelInferior();
        this.add(this.panelIzqueirda);
        this.add(this.panelIzqueirda, BorderLayout.SOUTH);

        this.panelTablero = new PanelTablero(30);
        this.add(this.panelTablero);
        this.add(this.panelTablero, BorderLayout.NORTH);

        this.panelInferior = new PanelJugadores(5);
        this.add(this.panelInferior);
        this.add(this.panelInferior, BorderLayout.CENTER);

        this.setTitle("Infinity Game");
        this.setSize(1200, 900);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setVisible(true);
        this.setLocation(400, 80);

    }
}
```

Paneles.

```java
public class PanelTablero extends JPanel {

    private ArrayList<JButton> casillas;

    public PanelTablero(int cantidadCasillas) {
        inicializar(cantidadCasillas);
    }

    private void inicializar(int cantidadCasillas) {
        int alto = 6;
        int ancho = cantidadCasillas / alto;

        this.setLayout(new GridLayout(alto, ancho));
        casillas = new ArrayList<>();
        for (int i = 0; i < cantidadCasillas; i++) {
            JButton casilla = null;
            if (i != 0 && i != cantidadCasillas - 1) {
                casilla = asignarColor(i);
            } else if (i == 0) {
                casilla = new JButton("" + i);
                casilla.setBackground(Color.WHITE);
            } else if (i == cantidadCasillas - 1) {
                casilla = new JButton("" + i);
                casilla.setBackground(Color.BLACK);
                casilla.setForeground(Color.WHITE);
            }

            casillas.add(casilla);
            this.add(this.casillas.get(i));
        }
    }

    private JButton asignarColor(int i) {
        JButton casilla = new JButton("" + i);
        Color[] colores = {Color.GREEN, Color.ORANGE, Color.CYAN};
        int indice = (int) (Math.random() * colores.length);
        Color selected = colores[indice];
        casilla.setBackground(selected);
        return casilla;
    }
}
```

```java
public class PanelJugadores extends JPanel {

    private ArrayList<JTextField> jugadores;

    public PanelJugadores(int cantidadJugadores) {
        inicializar(cantidadJugadores);
    }

    private void inicializar(int cantidadJugadores) {
        int alto = cantidadJugadores;
        int ancho = 1;

        this.setLayout(new GridLayout(alto, ancho));

        this.jugadores = new ArrayList<>();
        for (int i = 0; i < cantidadJugadores; i++) {
            JTextField jugador = new JTextField("Jugador: "+i);
            jugador.setEditable(false);
            this.jugadores.add(jugador);
            this.add(this.jugadores.get(i));
        }
    }
}
```

```java
public class PanelInferior extends JPanel {

    private JButton comenzarTurno;

    public PanelInferior(){
        inicializar();
    }

    private void inicializar(){
        this.comenzarTurno = new JButton("Hacer jugada");
        this.add(this.comenzarTurno);
    }
}
```

Ejemplos de diálogos iniciales con el usuario.

```java
private int elegirCantidadCasillas() {
    int cantidadCasillas = Integer.parseInt(JOptionPane.showInputDialog("Ingrese la"
            + " cantidad de casillas"));

    return cantidadCasillas;
}

private String ingresarNombreJugador() {
    String nombreJugador = JOptionPane.showInputDialog("Ingrese el"
            + " nombre del jugador");
    return nombreJugador;
}

private String elegirClaseJugador() {
    Object clase = JOptionPane.showInputDialog(null, "Seleccione la clase",
            "Clases", JOptionPane.QUESTION_MESSAGE, null,
            new Object[]{"Guerrero", "Mago"}, "Seleccione");
    return clase.toString();
}
```

```java
private ArrayList<Jugador> ingresoJugadores() {
    ArrayList<Jugador> jugadores = new ArrayList<>();

    while (true) {
        Jugador jugador;
        String nombre = ingresarNombreJugador();
        String clase = elegirClaseJugador();

        if(clase.equals("Mago")){
            jugador = new Mago();
            //inicializar mago
            jugadores.add(jugador);
        }else if(clase.equals("Guerrero")){
            jugador = new Guerrero();
            //inicializar guerrero
            jugadores.add(jugador);
        }

        int continuar = JOptionPane.showConfirmDialog(null, "¿Desea agregar más jugadores?", "Alerta",
                JOptionPane.YES_NO_OPTION, JOptionPane.ERROR_MESSAGE);

        if (continuar == 1) {
            return jugadores;
        }

    }
}
```