Tercera Práctica (P3)
Implementación completa del modelo

SESIÓN 3

Duración: 2 semanas

Examen: Semana del 27 de Noviembre (se retrasa una semana)

Descripción de la práctica

Vamos a desarrollar una interfaz de usuario textual para jugar a **Qytetet en Java y Ruby**. Todo lo que incluye el paquete **ModeloQytetet** (lo que hemos desarrollado hasta ahora) se corresponde con el modelo del juego. El modelo contiene las clases que implementan la funcionalidad del mismo (**Qytetet, Jugador, Dado, Tablero,** etc.) y no deben ser modificadas con aspectos de la interfaz. Por ello, vamos a crear un nuevo paquete denominado **InterfazTextualQytetet** que incluye una nueva clase denominada **ControladorQytetet** y otra llamada **VistaTextualQytetet**. La clase **ControladorQytetet** tiene como objetivo principal servir de nexo entre el modelo y la vista. La vista se encargará de la interacción de los usuarios(jugadores) con el juego y de mostrar el estado en el que se encuentra el juego en cada momento.

La dinámica del juego a implementar es la que se muestra en la siguiente figura:

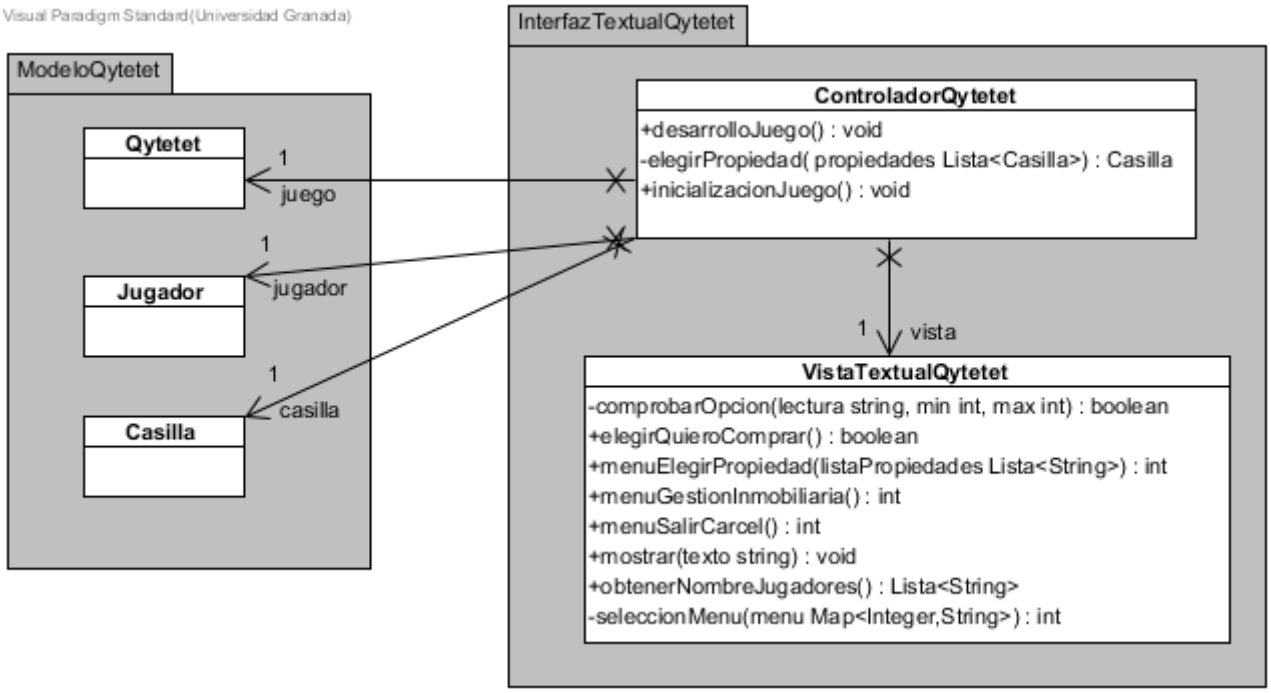
La clase **ControladorQytetet** tendrá tres variables de instancia: una que almacena la instancia del modelo (juego), otra que referencia al jugador actual (jugador), y otra a la casilla actual (casilla) (*). Tendrá tres métodos principales: **inicializacionJuego** (paso 1 de la figura), **desarrolloJuego** (pasos 2,3 y 4 de la figura) y un **main**. Dentro del main se llamará al método **inicializacionJuego** que inicializa el juego (paso 1 de la figura) y después al método **desarrolloJuego** que se encarga de gestionar todo el juego (pasos 2,3 y 4 de la figura).

La clase **ControladorQytetet** también incluirá los métodos privados que consideréis necesarios para simplificar la implementación de los anteriores. Uno de ellos será el método **elegirPropiedad**, que ofrecemos implementado para usarlo durante el desarrollo.

La clase denominada **VistaTextualQytetet**, también estará incluida dentro del paquete **InterfazTextualQytetet**.. Contiene los siguientes métodos: **menuGestionInmobiliaria, menuSalirCarcel, elegirQuieroComprar, menuElegirPropiedad, seleccionMenu obtenerNombreJugadores, comprobarOpcion, seleccionMenu, mostrar**. La mayoría los ofrecemos implementados, y debes completar los que faltan. (Ver ficheros aparte)

La clase **ControladorQytetet** incluirá otra variable de instancia que será una instancia de la clase **VistaTextualQytetet**, y será usada dentro de los métodos de inicialización y desarrollo del juego.

Todo esto se corresponde con el siguiente diagrama de clases:



A continuación, se describe en lenguaje natural cada uno de los pasos del juego (mostrados en la figura). No obstante, los métodos **inicializacionJuego** y **desarrolloJuego** están también descritos en los siguientes diagramas que se adjuntan a esta práctica:

- Diagrama de estados: que muestra los posibles estados en los que se puede encontrar un jugador y las condiciones que se tienen que dar para que se produzca un cambio en su estado.
- Diagrama de actividades: que muestra el flujo de control de las distintas acciones que se pueden desarrollar en el juego.

1) InicializacionJuego: Inicialización del modelo

Primero, se le da valor al atributo **qytetet** de la clase **ControladorQytetet**. Después, se solicitan los nombres de los jugadores por pantalla y se obtienen sus valores por teclado y se llama al método **inicializar** de **Qytetet** con los nombres de los jugadores. A continuación, se le da valor al atributo jugador y casilla de la clase **ControladorQytetet** como el jugador actual y su casilla actual respectivamente. Finalmente, se muestra por pantalla el estado completo del juego tras la inicialización: tablero, cartas sorpresa, jugador que comienza el turno, casilla actual, etc. Se recomienda introducir una pausa (por ejemplo, pidiendo al usuario que pulse una tecla para continuar), para dar tiempo a leer la información mostrada en pantalla.

2) DesarrolloJuego: Juego del jugador que tiene el turno

En cada turno de juego hay que mostrar por pantalla qué jugador va a jugar y cuál es su posición de partida, y después ponerlo a jugar.

El turno de un jugador tiene dos partes:

2.1) Movimiento

A) Si el jugador está en la cárcel:

Habrá que indicar que el jugador se encuentra en la cárcel y darle a elegir entre los dos métodos para salir (ver sección 4 de las reglas del juego): pagar la libertad o tirar el dado. En ambos casos habrá que mostrar si se consigue salir de la cárcel o no (añadir también pausa para leer). Si consigue liberarse puede jugar (B) y si no pasará de turno (ir a 3).

B) Si el jugador no está en la cárcel:

El jugador avanza desde la posición actual hasta una nueva casilla. Habrá que mostrar por pantalla la casilla dónde cae y también indicar qué acción ocurre en ella. La idea general es que se vaya mostrando el estado del jugador antes y después de cada acción y dar toda la información posible acerca de ello (de nuevo, añadir pausas para poder leer toda esa información en cada momento).

Por ejemplo:

· Si la casilla destino es de tipo sorpresa se indicará que se ha cogido una carta sorpresa y se mostrará la carta sorpresa que ha tocado. Después, se realizará la acción que corresponda (aplicarSorpresa), mostrando la nueva situación del jugador después de usar la carta.

· Si la casilla destino es una calle habrá que mostrar por pantalla el estado de la calle (casilla y título de propiedad). Puedes modificar los métodos *toString* que consideres necesarios para mostrar el nombre del propietario de la calle. Después, si la calle no tiene propietario se indicará al jugador que puede comprar su título de propiedad y darle opción para ello. Si la calle tiene propietario, se realizará el pago de alquiler. En ambos casos, se mostrará cómo queda el estado del jugador después de la operación que haya tenido lugar.

· ...

IMPORTANTE: Si el jugador cae en bancarrota (saldo menor o igual que 0) en algún momento durante este turno se acaba el juego (ir a 4).

2.2) Gestión inmobiliaria

Si el jugador no está en la cárcel, ni en bancarrota y tiene propiedades:

Después del movimiento del jugador, se facilitará un menú que permita: vender, hipotecar, cancelar una hipoteca, edificar un hotel, edificar una casa o pasar de turno (ir a 3). En estas acciones, el jugador necesita saber cuáles son sus propiedades y si están hipotecadas para poder elegir la casilla sobre la que va a realizar la acción.

Una vez elegida la opción, habrá que mostrar cómo queda el estado del jugador después de la acción (con pausas para leer).

3) DesarrolloJuego: Cambio de turno

Después de que juegue un jugador se pasa el turno al siguiente para que juegue (ir a 2).

4) DesarrolloJuego: Fin del juego

Si algún jugador ha caído en bancarrota se indica que ha acabado el juego y después se calcula y muestra un ranking con todos los jugadores.

(*) Recuerda que en Ruby, para usar una clase del modelo que está en otro módulo debes poner:

ModeloQytetet:: antes de la clase, ejemplos:

ModeloQytetet::Qytetet.instance y ModeloQytetet::TipoSorpresa::SalirCarcel

Diagrama de transición de estados

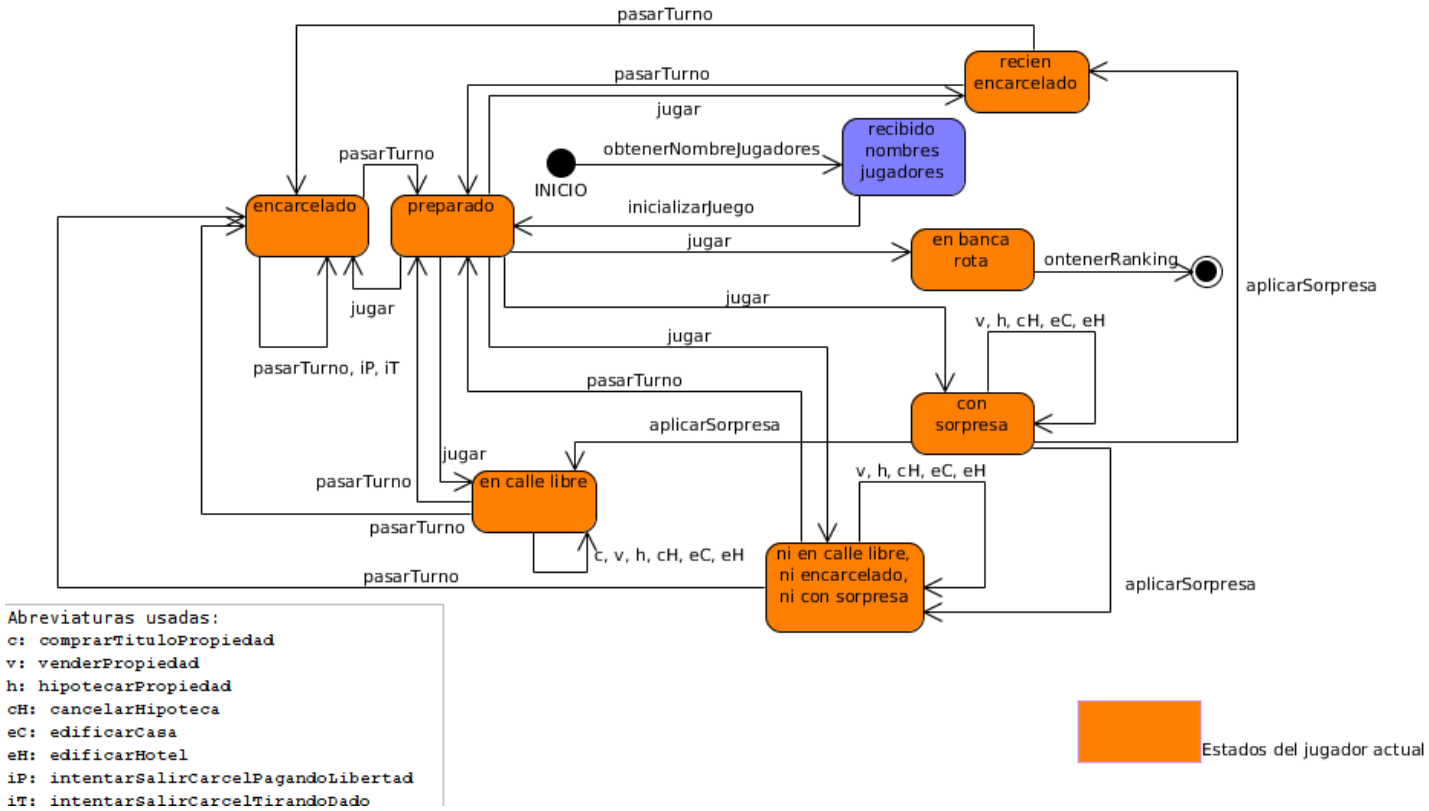


Diagrama de actividades

El diagrama de actividades se proporciona en un fichero aparte.