**Informe TPE: Silverphere**

Alumno: Leandro Matias Rivas

Lejago: 51274

Con el objetivo de desarrollar Silverphere, se implementó una determinada jerarquía de clases que respete el paradigma orientado a objetos.

Dos son las clases, entre todas las creadas, que deben destacarse: *AbstractCell* y *Element.*

Ellas modelan el comportamiento de los objetos con los cuales se conforma el mapa:

1) **celda**: contiene un objeto ( de la clase *Element*). Conoce cómo debe realizar el pase a otra celda de cierto objeto y como guardarlo.Además, es capaz de mover al personaje cuando éste lo recibe.

Las subclases son las siguientes: *Cell*, *Water*, *Switch*, *Destiny*, *Tree*.

Las celdas no conocen su ubicación en el mapa, con excepcion de la clase Destiny. Este posee su posición para ser utilizado en el método *drawDestiny*, de la clase *GamePlayFrame*, quien pregunta si el destino es visible, en cuyo caso lo dibuja.

2) **elemento:** contiene un board y su posición. Conoce si puede moverse a cierta celda. En caso de ser posible, el mismo cambia su posición. Cada celda posee una referencia a un elemento (Box, IceBox). Por otra parte, la instancia de *Player* no es contenida en ningún celda, sino que está presente en la instancia de *Board*. Esto se hizo con la intención de tener mayor comodidad con los llamados a la instancia del jugador.

Las subclases son las siguientes: *Box*, *IceBox*, *Player*.

Un aspecto a resaltar, es el agregado de bordes,formado por árboles, a los mapas. Estos permiten despreciar ciertos inconvenientes que el juego pueda sufrir al momento que el personaje se encuentre hacia un lado del mapa o quisiera salir del mismo. Como los bordes no forman parte del mapa que se esté jugando, no son impresos en pantalla. El manejo de ellos está a cargo del backEnd.

Respecto a diseños alternativos, se pensó crear una clase *WaterBox* (celda con agua y una caja) que extendería de *Cell*, con la diferencia que el método *hasWater* retornaría false y la variable que contenía la caja se coloca en *null* para poder recibir a otra caja. Esto no se implementó dado que las cajas y las cajas en agua estarían contenidas en el mismo set ‘boxes’, de forma no ordenada. Esto provocaría que al momento de dibujarlos, se los haga de desordenadamente apareciendo, por ejemplo, una caja en agua (de color azul) encima de una caja ( de color marrón).

Para evitar este problema, se decidió que una celda *Water* se convierta en una *Cell* sin elemento y que la caja pase a un nuevo set ‘boxesInWater’. Luego, a la hora de mostrar en pantalla las cajas, se dibujan primero todas las cajas en agua y después las cajas que no lo están.

En tanto a problemas durante el desarrollo, estos se presentaron con la ejecución del archivo build.xml por falta de ciertos paquetes. Otro dificultad fue la ventana de juego. Esta poseía el tamaño adecuado en Windows, pero no en Ubuntu. Esto se resolvió adecuandolo para Ubuntu para así poder ejecutarlo apropiadamente tanto en mi computadora personal con en las del laboratorio. Para finalizar, el no funcionamiento de las PCs del laboratorio como la imposibilidad de realizar los commit, producto de los recientes cortes de luz, dificultan el curso del TPE en los últimos días anteriores a la entrega.