Informe TPE

2011

Primer Cuatrimestre

**Grupo 10**

**Integrantes:**

Tomás Mehdi

Alan Pomerantz

Desarrollo y justificaciones del diseño e implementacion del juego

Cuando comenzamos a hablar sobre el diseño surgieron varias ideas, a pesar de las variadas ideas, las buenas opciones para un buen diseño orientado a objetos eran pocas.

Para comenzar se decidio hacer una jerarquia para los monstruos y los jugadores, estas dos clases heredan de “Character”; la cual es una clase abstracta. La justificacion a esta jerarquia es que monstruo y player tienen comportamientos parecidos y muchas variables en comun, pero no son lo mismo ya que el player puede realizar algunas cosas mas que un monstruo.

Se creo la interfase “Putable” lo que nos permite tratar a todas las cosas que pueden ponerse en un tablero de la misma manera. Esta interfaz solo tiene metodos, “allowMovement” y “standOver” los cuales permiten el movimiento sobre el tablero sin necesidad de saber que objeto halla en una posicion del tablero.

Otra jerarquia no tan importante es la de la clase “Cell”, de la cual heredan “Wall” y “Floor” y a la vez “BloodyFloor” Hereda de “Floor”. Es una jerarquia muy teorica que no aporta mucho a la implementacion del juego.

Se realizo un parseo con una gran orientacion a objetos, lo cual permitio utilizar esa clase para realizar la carga de un juego luego de ser guardado. No hay mucho que acotar a esta implementacion, hay partes imperativas, no hay forma de evitarlo, pero se realizo de una forma ordenada.

El front se basa en los metodos del “DungeonGameListener”, lo cuales son implementados en “DungeonGameFrame” en una clase inner privada “DungeonGameListenerImp”. “DungeonGameFrame” hereda de “GrameFrame”, una clase abstracta que podria utilizarse para crear el frame de distintos juegos. “DungeonGameFrame” contiene una instancia de un “DataPanel” y una de “DungeonPanel”. Las dos clases heredan de la clase “GamePanel” otorgada por la catedra. Cada una de ellas implementa una interfase distinta, pero que extiende de la interfase otorgada por la catedra. Esta jerarquia ademas de ser muy logica favorece mucho la implementacion, ya que se reutiliza mucho codigo y simplifica la creacion de la parte grafica con JavaSwing.

El topico de guardado y cargue del un juego al principio parecia que debia ser una cualidad del juego, “el juego sabe guardarse y cargarse”, pero luego descubrimos que si no era una “cualidad” del juego podria modularizarse para guardarse como cada uno quisiera sin tener que modificar la clase juego sino creando una propia clase para el guardado del juego. Lo cual nos guio a crear una interfase e implementar nuestra propia clase que la implementara.