**Introducción**

***Sobre este documento***

Aquí vamos a mostrar los diagramas de clase, primero separados en los tres subsistemas principales: Controlador, Modelo y Vista; y luego mostrando la relación entre las principales clases de cada uno de estos.

**Especificación de los subsistemas**

***Subsistema controlador***

Diagrama de clases



Creación y Unirse



Aquí podemos ver como es la interacción necesaria para la creación y para unirse a partidas.

***Subsistema vista***

Diagrama de clases



Interacción entre el cliente y las primeras etapas del juego



En este diagrama se especifica cómo son las interacciones entre el cliente y las primeras pantallas del juego, hasta llegar a la etapa de la sopa de letras. Además se puede observar como es el proceso de conexión y el paso de mensajes entre el cliente y el servidor, y como a partir de estos mensajes el cliente va cambiando las pantallas que muestra.Interacción entre el cliente y y el juego en la etapa de la ciudad



En este diagrama se especifica cómo son las interacciones entre el cliente y las ultimas pantallas del juego.

***Subsistema modelo***

Diagrama de clases



***Relaciones principales entre subsistemas***

Diagrama de clases



Como podemos observar, utilizamos el patrón Observer en este diagrama, entre la clase ControladorCambios (del subsistema controlador) y la clase Observer (de interfaz de juego), con el fin de mantener actualizado el estado del juego en cada cliente (en cualquiera de sus etapas y con todos sus elementos). Para que, de esta forma, pueda ser visualizado en la pantalla de todos los usuarios que estén jugando le juego con la información mas actualizada.

Utilizamos este patrón ya que veíamos que en nuestro sistema existía la posibilidad de que el servidor tuviera que actualizar el mapa de la ciudad a varios clientes, y el patrón Observer facilita esta tarea. Simplemente se suscribe al usuario correspondiente a la lista que tiene ControladorCambios. El método notify de este se encarga de ejecutar el método update correspondiente a cada objeto de la clase Observer (que tiene cada cliente). En caso que cierto cliente haya terminado la partida, se lo retira de la lista y deja de recibir actualizaciones del mapa.