Documento de diseño - Proyecto 2

Diseño y Programación Orientada a Objetos

Integrantes grupo 5:

1. Diego Rubio (201816492)

2. Juan David Salguero (201923136)

3. Alonso Hernández Tavera (202012367)

1. Introducción

El presente documento tiene el propósito de presentar documentación escrita y gráfica sobre el proceso de diseño realizado para el proyecto 2 del curso Diseño y Programación Orientada a Objetos. El proyecto 2 consiste en construir una GUI para la aplicación de Futbol de Fantasía. Para esto, se utilizará la programación orientada a objetos implementada con el lenguaje Java, el framework Swing y awt. Se tendrá en cuenta el análisis y diseño previo realizados y documentados en este documento. En este se puede encontrar información y descripciones correspondientes a justificaciones de diseño, restricciones del diseño del problema (reglas de dominio), contratos establecidos para cada método necesario en la solución del problema, diagramas de secuencia que explican gráficamente el procedimiento que se lleva a cabo para la solución de requerimientos funcionales clave y un glosario de los conceptos más relevantes en el mundo del problema y la solución propuesta. Los diagramas UML de dominio que fueron realizados para modelar el problema y su solución se encuentran adjuntos en la entrega. Estos son:

-Diagrama de clases simplificado con todas las clases del sistema (interfaz y lógica)
-Diagrama de clases de alto nivel con todas las clases del sistema
-Diagrama de clases de alto nivel de la interfaz

2. Consideraciones para el proyecto

Consideraciones anteriores:

Se ha definido como restricción que cada usuario participante (no administrador) de la plataforma pueda tener uno y solo un equipo de fantasía. Los usuarios administradores ya están registrados en la plataforma (base de datos de la aplicación), por tanto, no existe la opción de crear una cuenta de Administrador. Cuando se hace mención de los “participantes” se refiere a aquellos usuarios que se registraron con su nombre de usuario y contraseña, tienen un equipo de fantasía y se encuentran jugando al Futbol de Fantasía.

Nuevas consideraciones:

Consideraciones para la carga de datos:

1. Cuando el administrador quiere crear una nueva temporada el programa le pide 3 archivos en el siguiente orden: 1) archivo de equipos 2) archivo de jugadores 3) archivos de fechas. Sin importar el nombre debe escribirse con la extensión csv ej. "equipospremier.csv"
2. Cuando el administrador desea ingresar el resultado de un partido el programa le pedirá 4 entradas en el siguiente orden 1) nombre archivo de resultado del partido (siguiendo el lineamiento del literal anterior) 2) numero de la fecha a la que pertenece el partido (la entrada es un numero) 3) seleccionar el equipo que jugó de local 4) seleccionar el equipo que jugó de visitante

3. Justificación de decisiones de diseño

Para implementar la interfaz se tomó la decisión de restructurar la aplicación y aplicar el modelo MVC. Así se va a poder distinguir de manera fácil que corresponde a la lógica y a la interfaz de la aplicación. Del mismo modo, se pueden realizar cambios en la vista sin tener que afectar el modelo y viceversa.

3.1 Justificación de decisiones de diseño para la lógica:

1. Se creó la clase abstracta Equipo que implementa la clase Serializable para facilitar la persistencia de datos.

2. Clase EquipoFantasia: Hereda de la clase Equipo y se ACTUALIZARON sus atributos de modo que ahora se puede identificar jugadores titulares y suplentes por cada posición.

3. Clase EquipoReal: Hereda de la clase Equipo y se ACTUALIZARON sus métodos de modo que ahora se puede agregar un jugador real, obtener jugadores por posición y buscar un jugador en el equipo. Esto va a facilitar la creación de datos reales.

4. Clase Fecha: se ACTUALIZARON sus métodos de modo que ahora se puede agregar y buscar un partido en el cronograma de fechas. Esto va a facilitar la creación de fechas para una temporada real.
5. Se creó la clase abstracta Jugador que implementa la clase Serializable para facilitar la persistencia de datos.

6. Clase JugadorFantasia: Hereda de la clase Jugador y se ACTUALIZARON sus atributos de modo que ahora se pueden identificar las estadísticas (goles, asistencias, etc) de cada jugador. Esto nos facilitará ver el rendimiento de los jugadores de nuestro equipo de fantasía.

7-8. Clase JugadorFantasiaArquero y Clase JugadorFantasiaDefensivo: Se añadieron estas dos nuevas clases ya que hay otras formas adicionales para darles puntos a estos tipos de jugadores.

9. Clase JugadorReal: : Ahora hereda de la clase Jugador.

10. Clase MarcadorPartidoReal: Se añadió el atributo de ganador, así como los métodos para agregar goles a cada equipos y también su rendimiento. Esta clase se va a relacionar con la clase RendimientoJugador. Esto nos va a permitir ver el rendimiento de todos los jugadores del equipo local y visitante para que luego el admin pueda agregar dicha info.

11. Clase Participante: Se añadió la relación con EquipoFantasía. Del mismo modo se añadieron los métodos para comprar jugadores dependiendo su posición y las verificaciones y actualizaciones de presupuesto.

12. Clase RendimientoJugador: Se añadió la relación con un jugador Real, de esta forma se podrán visualizar las estadísticas de un jugador luego de un partido real.

13. Se creó la clase abstracta Temporada que implementa la clase Serializable para facilitar la persistencia de datos.

14. Clase TemporadaFantasia: Se agregaron métodos para agregar un equipo de fantasía a la temporada, así como actualizar los puntos de los jugadores.

15. ClaseTemporadaReal: Se agregaron métodos para agregar un equipo real a la temporada y también para buscar un fecha. De este modo podemos ver los resultados de los equipos reales en una temporada.

16. Se implementaron las clases creadorObjetos, LectorArchivos y Persistencia. Estas fueron creadas para ayudar en el manejo y creación de datos.

3.2 Justificación decisiones de diseño interfaz:

1. Clase AdminAppFrame: Esta será la ventana que se le va a mostrar al Administrador. A partir de esta se le van a mostrar a este usuario las opciones que puede ejecutar.
2. Clase AdminAppPanel: En esta clase aparecen los botones para crear una nueva temporada, subir una temporada real,etc.
3. Clase AdminLoginPanel: Este panel se encarga de el inicio de sesión del Admin por lo que se va a relacionar con toda la lógica de verificación de un Admin.
4. Clase PlayerAppFrame: Esta será la ventana que se le va a mostrar al Participante. A partir de esta se le van a mostrar a este usuario las opciones que puede ejecutar.
5. Clase PlayerAppPanel: En esta clase aparecen los botones para crear un equipo de fantasía, comprar y vender jugadores, etc.
6. Clase PlayerLoginPanel: Este panel se encarga de el inicio de sesión de un Participante por lo que se va a relacionar con toda la lógica de verificación de un Participante.
7. Clase PlayerRegisterPanel: Este panel de registrar un nuevo Participante por lo que se va a relacionar con toda la lógica de creación de un Participante.

4. Diagramas de secuencia

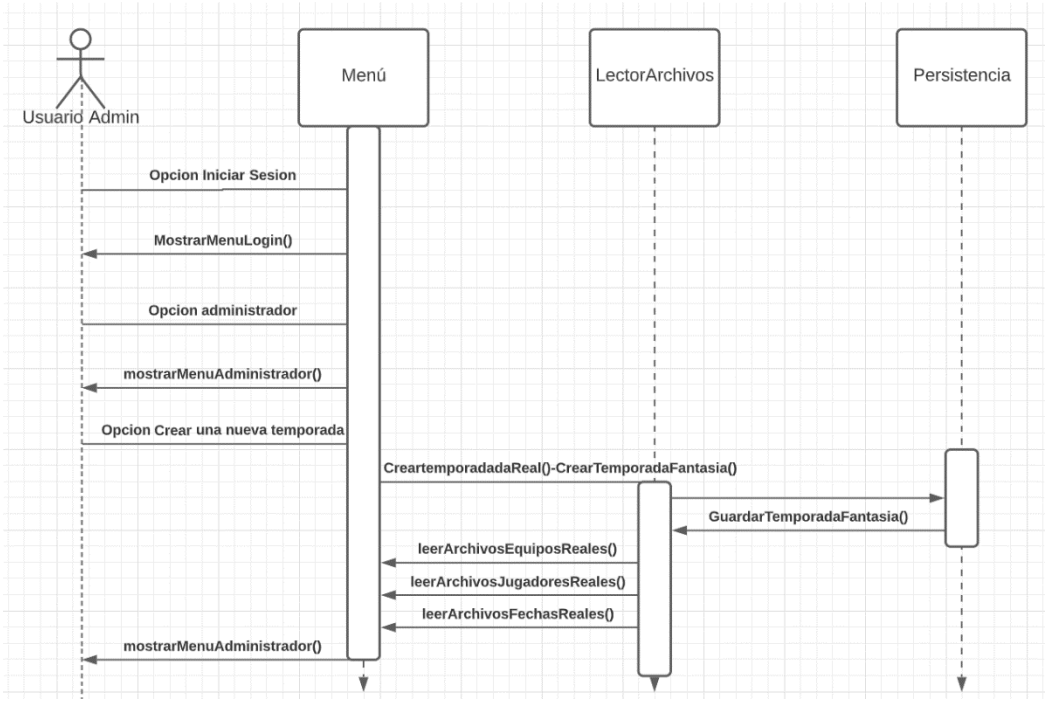

Figura 1 Diagrama carga de archivos

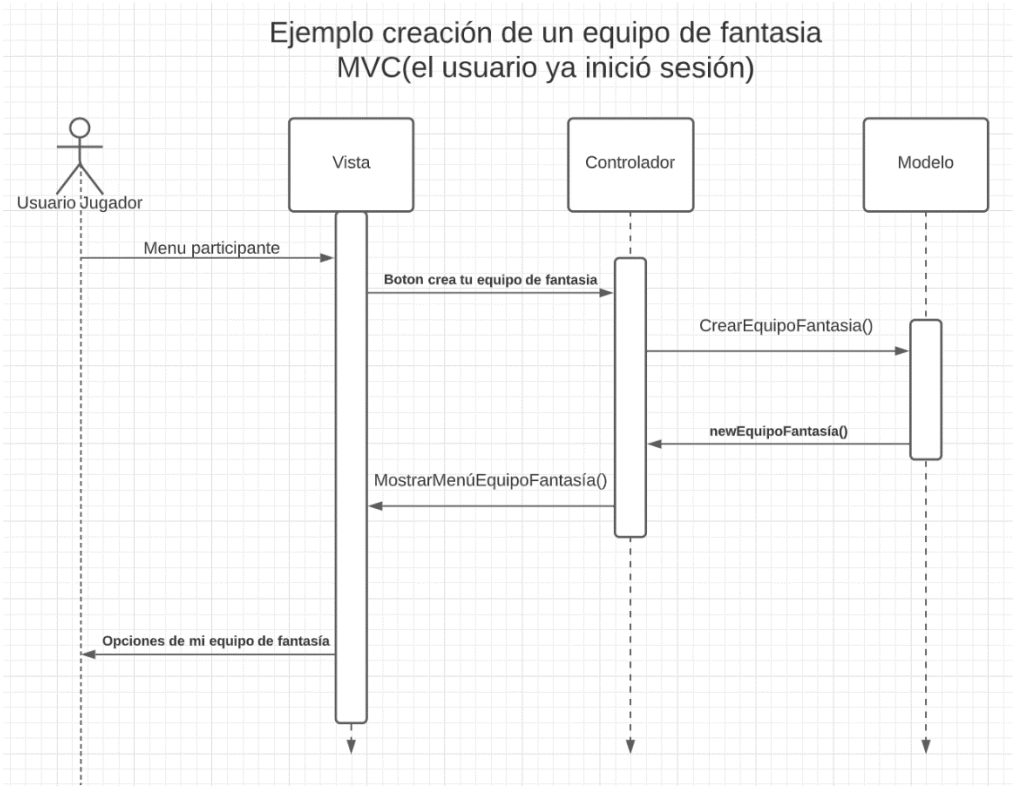

Figura 2 Diagrama interacción MVC

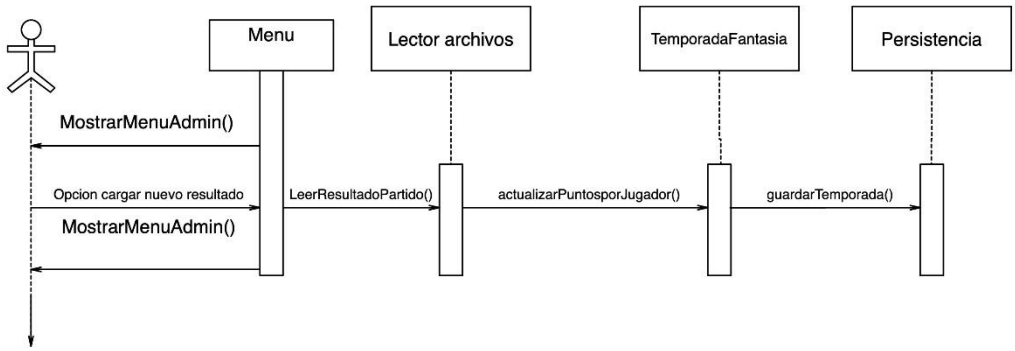

Figura 3 Diagrama carga de resultados partido real