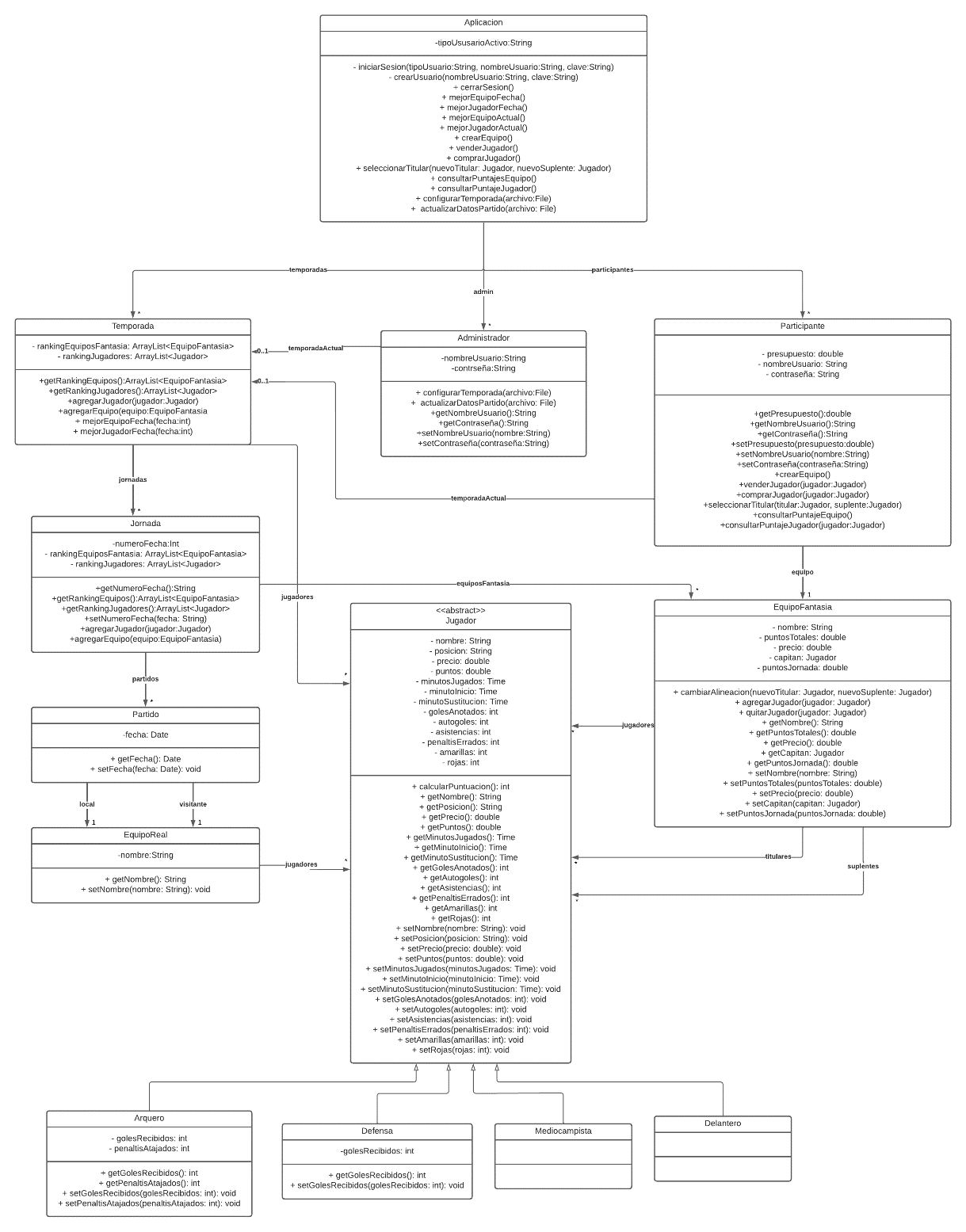
**Nicolas Lozano - 202021428**

**Juan Esteban López - 202021417**

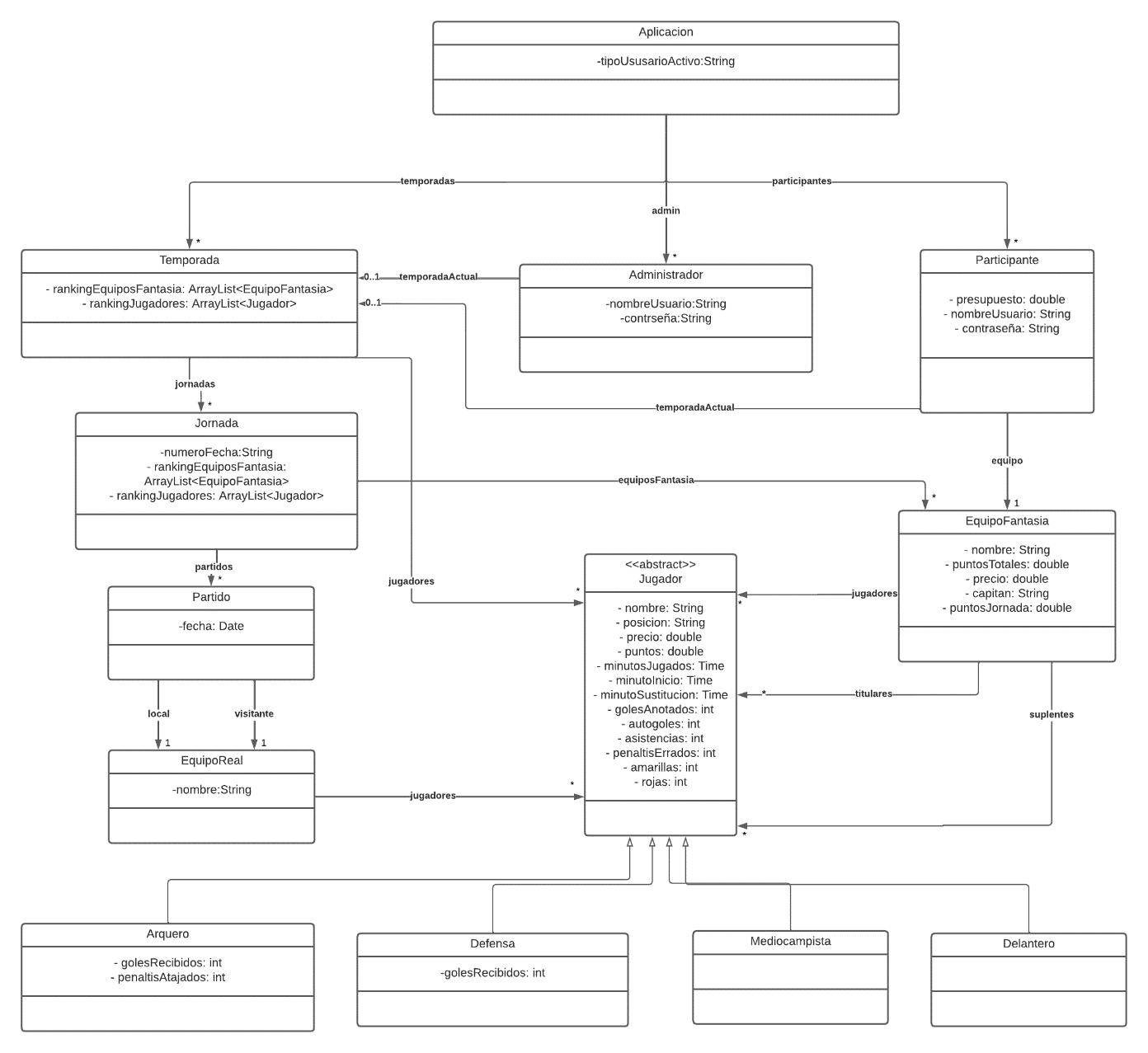
**Manuel Pérez - 202021826**

**ETAPA DE DISEÑO**

1. **Diagrama de Clases de Diseño**



1. **Diagrama de Clases de Alto Nivel**



1. **Diagrama de Secuencia**

Diagrama

Descripción generada automáticamente

En este diagrama se muestra la secuencia que seguirá la aplicación para crear un equipo. Acá se está contemplando solo el caso base, pero vale la pena aclarar que este va a ser un proceso recursivo donde, hasta no completar los 15 jugadores sin sobre pasar el presupuesto dado. Tampoco se tiene en cuenta la selección de jugadores titulares y suplentes ya que esta es una secuencia diferente.

1. **Justificaciones**

1. **Clase jugador:** La clase jugador se creó como una clase abstracta. Esto debido a que existen diferentes tipos de jugadores, dependiendo de la posición. Todos cuentan con atributos y funcionalidades en común, sin embargo, las formas de calcular los puntajes dependiendo de la posición varían (este sería el método abstracto).

2. **Atributo tipoUsuarioActivo:** En la clase de la aplicación se creó un atributo llamado tipoUsuarioActivo. Esto es para que la aplicación sepa si el usuario que la está usando es un participante o el administrador del torneo. Para facilitar el trabajo y ahorrar tiempo, dependiendo del tipo de usuario que inicie sesión, se muestra un menú con las funcionalidades disponible según el usuario, y se restringe el acceso a funcionalidades que no estén disponibles.

3. **Métodos:** En la clase aplicación existe un método para cada requerimiento funcional. Esto con el fin de que los usuarios y administradores puedan ejecutar el requerimiento que desean y esta llama a diferentes métodos de las clases correspondientes que ayudan a satisfacer las necesidades de los diferentes usuarios. Por esta misma razón existen métodos con el mismo nombre en diferentes clases esto para que el controlador pueda controlar las acciones de los otros objetos y llevar a cabo todos los requerimientos funcionales.

4. **Clase equipoReal:** Se decidió crear la clase equipoReal, para modelar los jugadores y equipos reales del torneo. Esto con el fin de agrupar fácilmente la información de las jornadas, y que sea más fácil actualizar la información de los equipos de fantasía cuando el administrador suba los resultados de cada fecha.

5. **Carga de archivos:** La información de la temporada y los resultados de cada fecha serán subidos en formato JSON a la aplicación. Aunque esto resulta en un desarrollo adicional para leer dichos documentos, es más el alivio que representa para el administrador al no tener que subir toda esta información por consola.

1. **Estereotipos**

**Temporada: Estructurador. N**o tendrá una lógica compleja y su rol será principalmente organizar las jornadas para que acceder a ellas sea sencillo, además de que contendrá todos los jugadores de una temporada para que las clases que lo necesiten puedan acceder a ellos.

**Jugador: Service Provider.** Realiza la tarea de calcular los puntajes de los jugadores al finalizar cada partido, este servicio es esencial para la aplicación, pues todos los requerimientos necesitan estas puntuaciones.

**Equipo de fantasía: Controller.** Concentra alguna de la lógica más importante del sistema, pues interactúa con la clase jugador para ejercer todos los requerimientos relacionados con crear y manejar un fantasy, por ende, sin la lógica de esta clase, la aplicación no funcionaría.

**Jornada: Information Holder.** Mantiene información de cada jornada disputada.

**Aplicación: Controller.** Concentra mucha de la lógica del sistema e interactúa con otras clases para lograr desarrollar sus métodos. Por ejemplo, da la opción de que nuevos usuarios se unan al juego, arroja estadísticas interesantes de la temporada y guarda los datos de las temporadas anteriores.

**Partido: Information Holder.** Mantiene información de cada partido disputado durante la temporada.

**EquipoReal: Structurer.** No tendrá una lógica compleja y su rol será principalmente organizar los jugadores para permitir guardar su información fácilmente en los partidos.

**Participante: Coordinador.** No tendrá una lógica complicada y delegará al equipo de fantasía la lógica y realización de tareas más complejas como comprar jugadores, venderlos o modificar la alineación para una jornada.

**Administrador: Interfacer.** Su rol principal será mediar en la interacción con un elemento externo, en este caso, la base de datos, tanto para configurar una temporada, como para actualizarla constantemente.