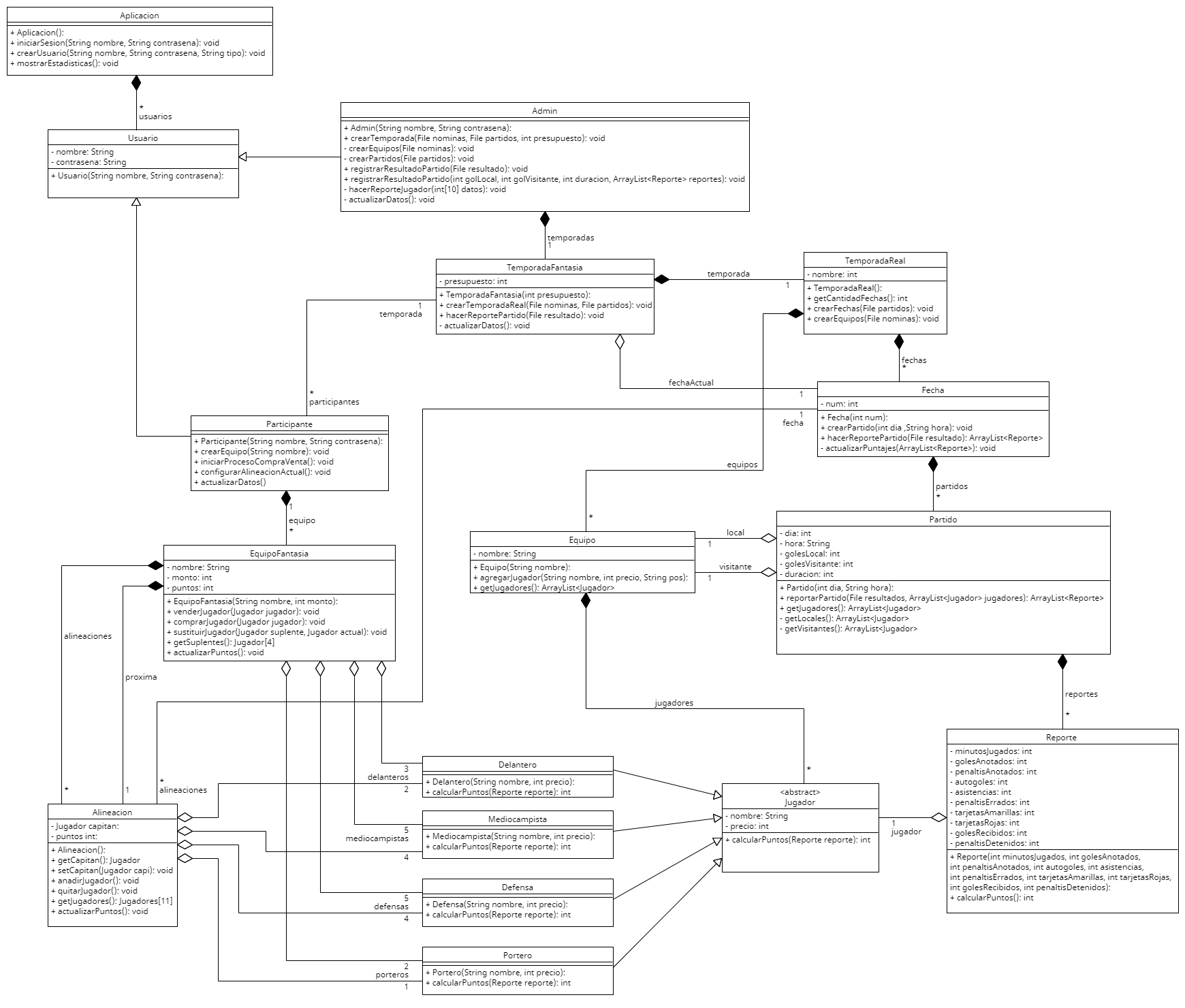
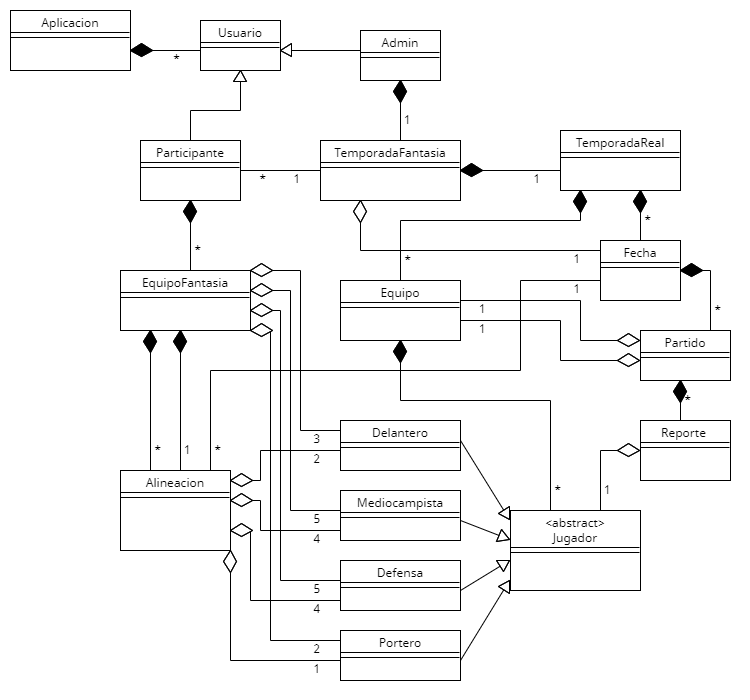
Diseño Proyecto 1

1. Diagrama de clases



1. Diagrama de alto nivel



1. Justificaciones

Para el manejo de usuarios en el sistema decidimos hacer una clase padre Usuario de la que salieran los participantes de la liga de fantasía y el administrador. Esto lo hicimos porque, aunque tengan funciones distintas, el administrador y el participante, su acceso se en ambas hace con usuario y contraseña.

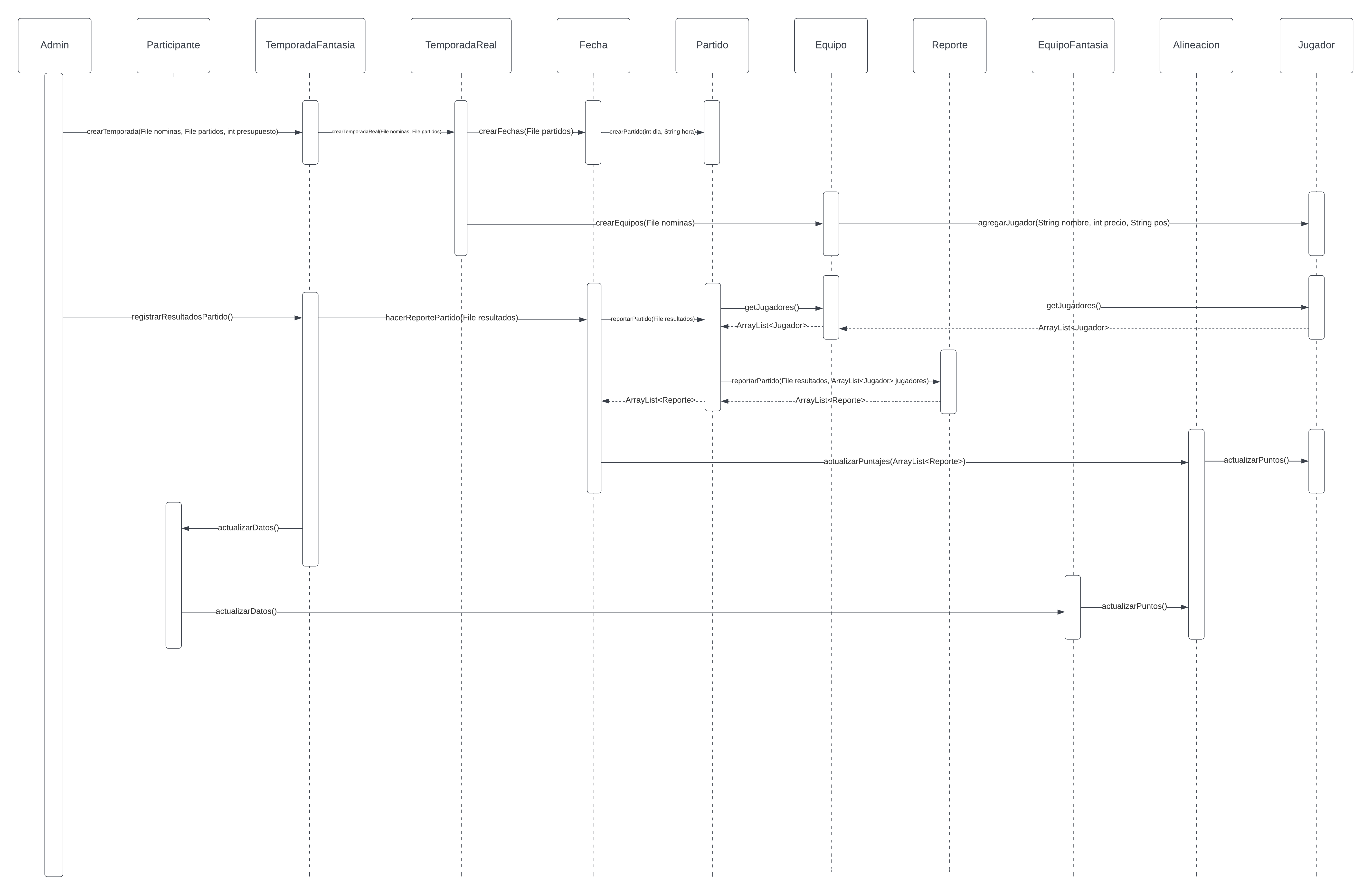
Para permitir que el administrador pudiera cumplir con sus funciones de: crear la temporada, ingresar las nóminas de los equipos, ingresar el presupuesto de los equipos de fantasía, ingresar los partidos y reportar los resultados de los partidos, decidimos que era esencial que la aplicación se centrara en el administrador. De este modo el administrador tiene acceso a todos los valores de una temporada de fantasía y puede así hacer las modificaciones pertinentes. Además, decidimos que la relación del administrador con la temporada de fantasía sería de composición, ya que, si a una temporada se le quita el administrador, nadie sería capaz de ingresar los reportes de los partidos, por lo que no tiene sentido que una temporada siga existiendo sin su administrador.

Para la representación de los jugadores decidimos aplicar herencia para representar las diferentes posiciones que puede tener un jugador. Esto lo hicimos en base a una clase abstracta ya que, a la hora de calcular los puntos recibidos después de un partido, el método es distinto para cada posición. De este modo podemos llamar a un solo método y calcular los puntos recibidos, y esto se hará con su función respectiva dependiendo de la posición. Para los jugadores también, decidimos que estos componen un equipo ya que, si se elimina el equipo, los jugadores ya no jugarían, y por consiguiente no serían relevantes para la aplicación.

Para elaborar los reportes, decidimos que los reportes iban a tener una relación de composición frente al partido, ya que estos estarían ligados al desempeño de cada jugador en este. Cada reporte estaría asociado además a un jugador, lo que luego permitiría que el jugador calcule sus puntos obtenidos en el partido. Con esto después estos puntos serían contados por la alineación y entregados al equipo de fantasía.

Para el manejo de la alineación decidimos manejarlo por dos partes: una instancia para la alineación actual y una lista de alineaciones guardando el histórico del equipo de fantasía. De este modo, antes de cada fecha, el usuario puede editar la instancia individual del equipo, y cuando entra la nueva fecha esta alineación se guarda en el historial de alineaciones, y se abre para que se pueda editar la alineación de la siguiente fecha. Por esta razón también una alineación está asociada con una fecha, ya que solo los puntos de esta serán contados para cada alineación.

1. Diagrama de secuencia



1. Responsabilidades

**Roles:**

* R1: Administrador
* R2: Usuario
* R3: Sistema
* R4: Jugador
* R5: Equipo Fantasía
* R6: Reporte
* R7: Partido real
* R8: Equipo real
* R9: Fecha
* R10: Alineación

**Responsabilidades:**

* RP1: Configurar temporada
* RP2: Registrar resultado partido
* RP3: Registrar desempeño jugador
* RP4: Registrar nóminas equipos reales
* RP5: Mostrar estadísticas temporada
* RP6: Crear equipo fantasía
* RP7: Comprar jugador
* RP8: Configurar alineación
* RP9: Vender jugador

**Colaboraciones:**

* C1: [R2, R4]
* C2: [R2, R5]
* C3: [R4, R5]
* C4: [R1, R6]
* C5: [R4, R6]
* C6: [R6, R7]
* C7: [R4, R8]
* C8: [R7, R8]
* C9: [R7, R9]
* C10: [R4, R10]

1. Diagrama de casos de uso

