Trabajo Práctico
Organizador de Futbol 5

Entrega Nº 3

Materia: Diseño de Sistemas
Profesor: Nicolas Passerini
Ayudante: Gisela Decuzzi
Alumnos:

- **Acosta Naiara**
- **Brandoni Agustin**
- **Coiro Tomas**
- **Leder Brian**
- **Luis Ostiglia**

2014

Punto 3

Para empezar cuando hablamos de bajo acoplamiento, la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización de código y disminuyendo la dependencia entre las clases, todo esto ayuda a que nuestro sistema sea lo más flexible posible.
Para lograr un bajo acoplamiento lo primero que tenemos que hacer es eliminar las relaciones innecesarias y tratar de debilitar las relaciones necesarias, sin embargo debemos cuidar no bajar de forma excesiva la cohesividad produciendo los llamados "god object".

Luego aplicando algunos conceptos específicos de diseño podemos disminuir el acoplamiento por ejemplo utilizamos delegación, encapsulamiento (estas dos conceptos te ayudan a separar mejor las responsabilidades de cada objeto).

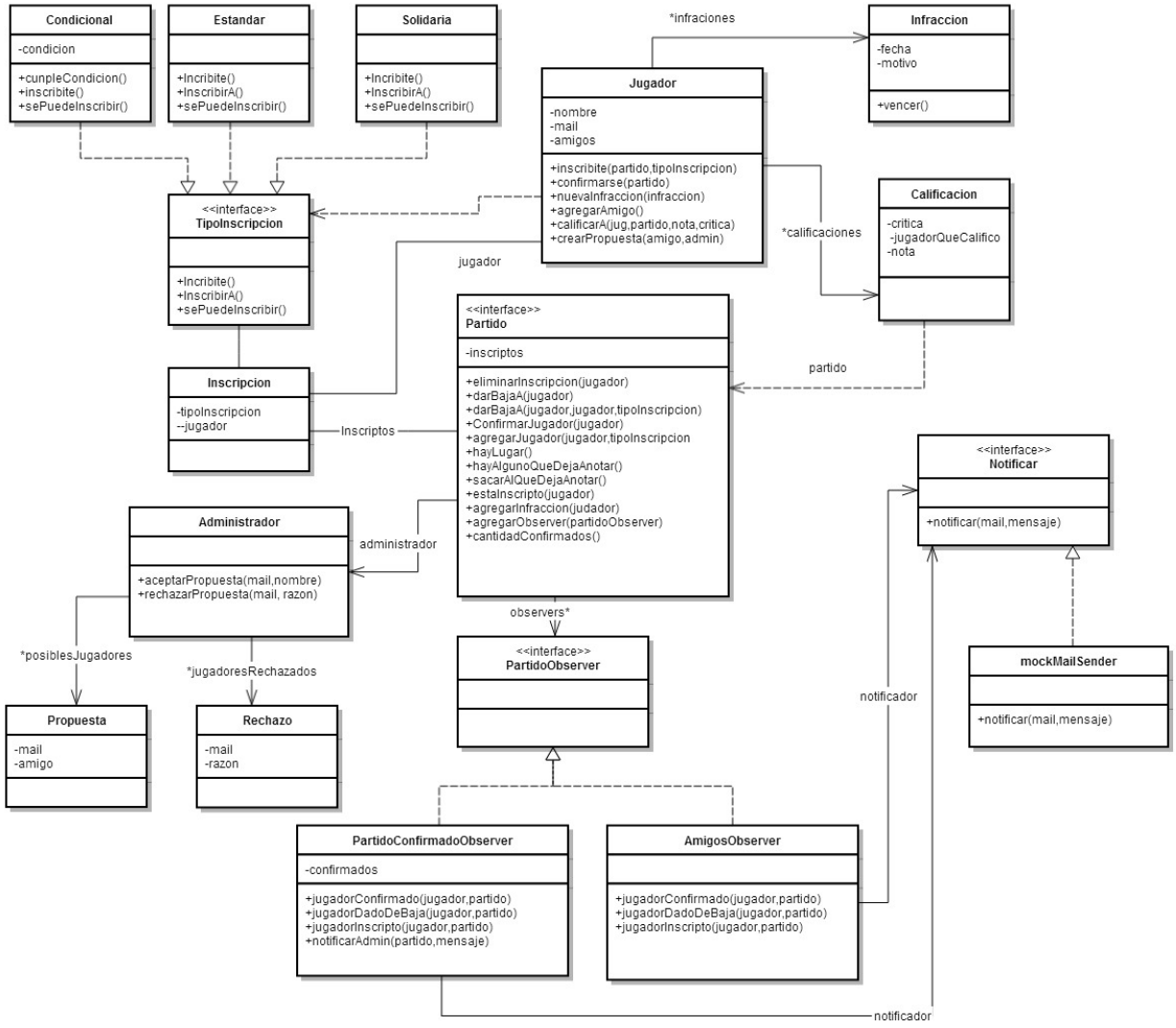
Por ejemplo el método calificarA(jugador, partido, nota, crítica) en la clase jugador aplica el conceptos de delegación, en el método crearCalificacion(), éste delega esta responsabilidad al jugador que se calificó con el método agregateCalificacion(), y éste último setea la calificación en el jugador.

El encapsulamiento protege a los datos asociados de un objeto contra su modificación por quienes no tengan derecho a acceder a ellos, delegando correctamente se encapsula a los objetos. Por ejemplo la clase Jugador posee un método específico que agrega una nueva calificacion, esta función que le es propia al objeto jugador, encapsula su comportamiento.
A su ves la clase Administrador posee metodos como nuevaPropuesta(), removerPropuesta() que representan comportamiento propio(no es heredado).

El acoplamiento disminuyó también ya que se definieron correctamente las responsabilidades de cada objeto. Las colecciones de Rechazados y Propuestas son conocidas solo por el administrador y no por el partido, por ejemplo, lo cual hubiera llevado un aumento del acoplamiento entre la clase Administrador y Partido.
Además al crear una colección de propuestas, estas son las pendientes, lo cual se pudo deshacer de la responsabilidad del Administrador de aceptar o rechazar en el momento en que llegan.

PUNTO 4

Diagrama de clases



Para poder satisfacer las nuevas funcionalidades de esta entrega se crearon nuevas clases. Para el requerimiento del caso de uso "Nuevo jugador" se crearon las clases "Administrador", "Rechazo" y "Propuesta". Las relaciones de estas clases se resumen en que un partido tiene un administrador y éste último tiene dos colecciones (posiblesjugadores y jugadoresRechazados) que son instancia de las clases "Rechazo" y "Propuesta" respectivamente.
Para satisfacer los casos de uso y "calificar jugadores del partido"
se creó la clase "calificacion", la cual se relaciona con un jugador, éste tiene una colección de calificaciones y las calificaciones conocen a un partido.

Las calificaciones las posee el jugador, ya que al haber nuevos requerimientos, es el jugador el que podría llegar a utilizarlas para pedirle un promedio de notas, revisar lo que les dijeron los jugadores, en que partido obtuvo su nota más alta etc.
En el caso de que fuera responsabilidad del partido, contribuiría a aumentar la cantidad de conocimiento que posee, lo cual es malo porque podría convertirse en un god object.
Además sería difícil de leer y plantear relaciones innecesarias entre Jugador y Partido, ya que jugador deberá cada vez que quiera su calificación, pedírsela a cada partido del cual participó. Como se mencionó anteriormente, tenemos que destacar que este punto fue fundamental para disminuir el acoplamiento.

La idea de reificar la calificación genera una solución más extensible ya que permite agregarle nuevos atributos y/o comportamiento. Y permite agrupar un conjunto de atributos que le son de importancia a cada jugador particular.