Implementación relacional de la herencia en objetos:

Para el caso particular de los distintos tipos de jugadores (estandar,solidario,condicional) optamos por una entidad que represente

las distintas modalidades disponibles para ser anotado a un partido. Con esto conseguimos que el modelo sea escalable ya que tanto en el modelo de objetos,

como en el modelo de datos, existe la posibilidad de añadir nuevos tipos de jugadores.

Por tanto tenemos una entidad para cada tipo de jugador, cada una con sus atributos especificos.

Otra posibilidad analizada, pero finalmente descartada fue la de una unica tabla donde se compartan todas los atributos de cada tipo de jugador a que no

nos parece logico y dificulta el mantenimiento y escalabilidad del sistema.

Relaciones:

Se ha normalizado a tercera forma usando por tanto tabla intermedias para romper las relaciones muchos a muchos, y para mantener la integridad de datos

se ha utilizado campos con clave foranea. (Por ejemplo: Interesado hace referencia a Usuario, imposibilitando asi que un interesado sea creado sin estar

relacionado de manera directa a un usuario ya existente en el sistema.

Tipos de datos:

Hemos considerado utilizar el tipo de datos dinamico VARCHAR para los campos de tipo texto, para así optimizar el espacio fisico implicado en cada entidad,

versus la opcion de utilizar campos de texto de tamaño fijo. Para los identificadores hemos utilizado el tipo INT ya que además de ocupar menos espacio,

facilita y mejora la performance de las distintas consultas que se pueden realizar sobre la base de datos.

Implementación ORM:

No tuvimos que modificar practicamente ningun componente, solo implementar la persistencia y la genercion del DBML.

Consideramos que esto fue posible gracias al bajo nivel de acoplamiento entre clases que conseguimos ya que las diferentes clases realizan su trabajo de

manera particular si depender (o dependiendo lo minimo e indispensable) del correcto funcioamiento de alguna otra clase.