# CS616 - Software Engineering II - Patterns Exercise

You are designing the entire Football (Soccer) game engine, and have created an initial conceptual class diagram of the game.



You find that there may be many design problems with this design. The one you will consider here is has to do with the ball. In particular, when the position of a ball changes, all the players and the referee should be notified immediately. The solution to this problem is the use of the *Observer Pattern.*

*Observer Pattern*: Define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically. Its class diagram is given below.



The participants of the pattern are detailed below.

**Subject**

This class provides an interface for attaching and detaching observers. Subject class also holds a private list of observers. Functions in Subject class are

* Attach - To add a new observer to the list of observers observing the subject
* Detach - To remove an observer from the list of observers observing the subject
* Notify- To notify each observer by calling the Update() function in the observer, when a change occurs.

**ConcreteSubject**

This class provides the state of interest to observers. It also sends a notification to all observers, by calling the Notify function in its super class (i.e, in the Subject class). Functions in ConcreteSubject class are

* GetState - Returns the state of the subject

**Observer**

This class defines an updating interface for all observers, to receive update notification from the subject. The Observer class is used as an abstract class to implement concrete observers

* Update - This function is an abstract function, and concrete observers will over ride this function

**ConcreteObserver**

This class maintains a reference with the subject, to receive the state of the subject when a notification is received.

* Update - This is the overridden function in the concrete class. When this function is called by the subject, the ConcreteObserver calls the GetState() function of the subject to update the information it have about the subject's state.

**Exercise:**

Adapt the observer pattern to solve the ball problem *when the position of a ball changes, all the players and the referee should be notified immediately.*

# Solution

