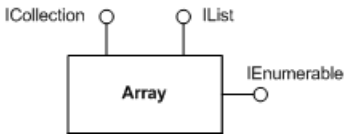
Ball And Socket

3 February 2005

Martin Fowler

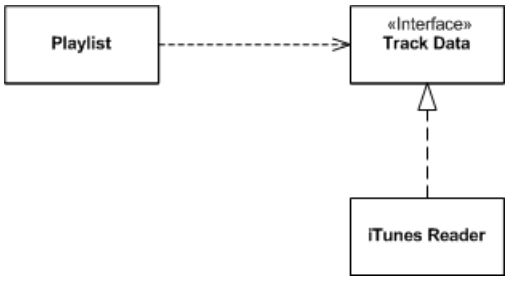
UML

One of the new notations that appeared in UML 2 was the socket notation to show interfaces required by a class. The origins of this was the 'lollipop' notation that was popularized by Microsoft to show a class implementing multiple interfaces. So I can show that the Array class implements multiple interfaces like this.



This is a useful notation for showing this kind of thing. Using the realization arrow would result in a much more messy diagram.

Classes don't just implement interfaces, you can also think of them requiring interfaces. Let's imagine I write a class that can provide various information about my digital music playlists, such as the total length of the playlist. In order to get information about individual music files I need to get the data from somewhere. Since I'm a reasonably hip dude I can get it from iTunes - but since I'm aware that there are other possibilities (such as reading directly from an mp3) I use an interface so that I can easily substitute another implementation.

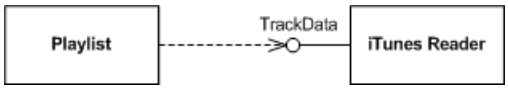


The required interface notation allows me to show this required interface with a compact socket notation.

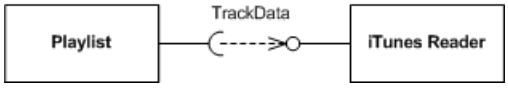
With the ball and socket so close to each other, it only seems natural that they should mate. So the UML specification that I saw when writing UML Distilled let them do so.

Now although this was allowed in those UML specifications, Bran Selic kindly let me know that the UML committee has decided this was an error and you shouldn't be able to use the ball and socket notation in this way. Ball and socket notation is still in the UML, but it can only be used for connectors within Composite Structure Diagrams - which is a separate topic area.

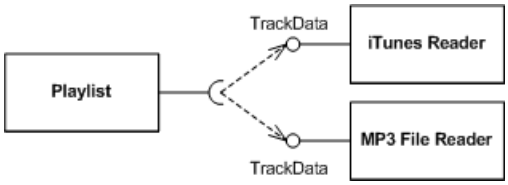
This leaves the question of how you should show this link between required and provided interfaces. The way we did it in UML 1 was with a dependency.



Jim Rumbaugh's reference manual shows another way.



For an example like this, I think I prefer the UML 1 style, since there's less notation to convey the same meaning. However the Rumbaugh style shines when you want to show something more complicated, such as multiple classes implementing a single required interface.



Despite the technical invalidity of the mated ball and socket notation, I rather like it as a way to show how interfaces and implementations connect, so am again inclined to break strict UML rules to use it. I may use the dependency notation between the ball and socket in more complicated cases, but when that happens I prefer explicit class boxes for the interfaces.