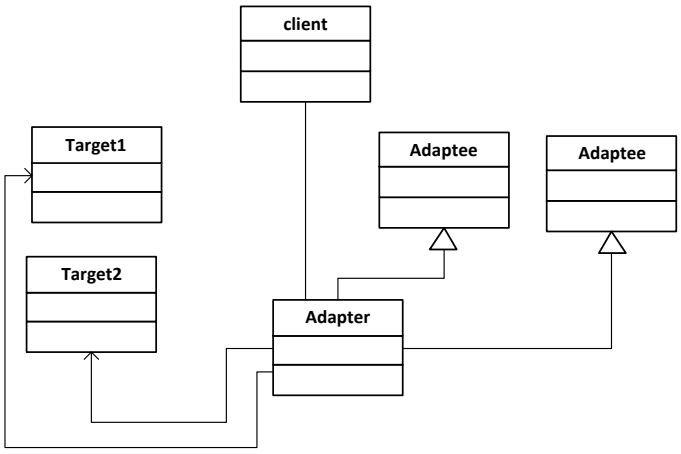
6 Adapter Pattern [Gamma et al]

Unlike adapters in the real world which provide a compatible interface between two separate objects, adapters in design patterns brings the same idea on board. There are situations when a certain change has a bigger impact on the existing code. A certain scenario may occur when there are different classes exposing different interfaces but need a same type of input object. So in order to work with both classes adapter classes came into existence that make two incompatible interfaces make work. [18] Defines an example of adapter pattern in a real time application. Adapters can be well known implemented as Class Adapters and Object Adapters. Class adapters involve multiple inheritances which are obsolete in modern languages but a way of implementation still exists in the form of multiple interface implementations. On the other hand object adapters involve composition technique as implement [1]. Following are the variants of Adapter Pattern:

6.1 Pluggable Adapters [54]

Dynamic adaptation of Adapter Class to one or many existing classes is termed as pluggable adapters. The term pluggable is related to run time loading of classes. Not every class can be adapted. There is a qualification criteria involved. I.e. Adapters must recognize which Class it should adapt to. The optimum way to know the Adaptee class and Target Class is to define certain methods with appropriate parameter types as Adaptee features. These features can be used to identify the classes which Adapter should adapt to. Reflection [4] provides a mechanism to read .Net Assembly metadata and locate methods with those feature types.

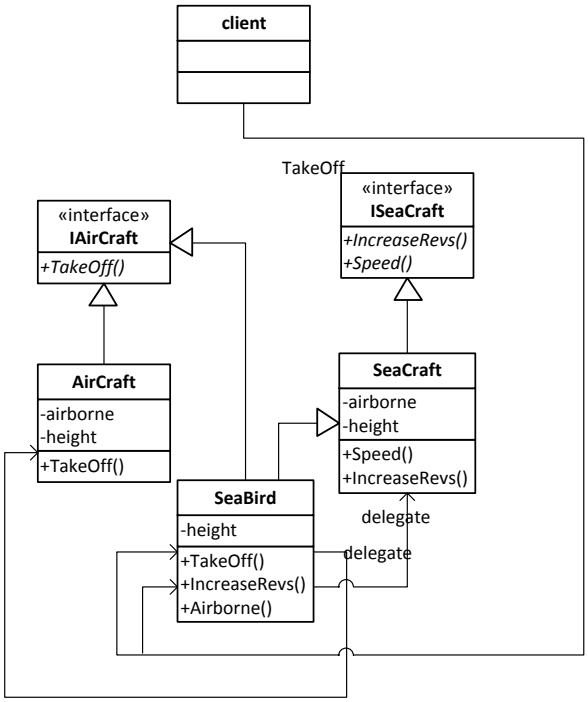


6.1 UML Diagram of Two Way Adapter

6.2 Two Way Adapters [54]

Two way adapters provide a transparency view to multiple Adaptee interfaces. They are used in situations for example when there is implementation of a certain process that involves some

third party vendor, as the implemented gets older a requirement pops in which illustrates the fact to embed the new vendor interface. In that case it is unrealistic to remove the old working functionality, so implementation of an interface that acts as a bidirectional i.e. an old and new interface at the same time.



6.2 UML Diagram of Two Way Adapter