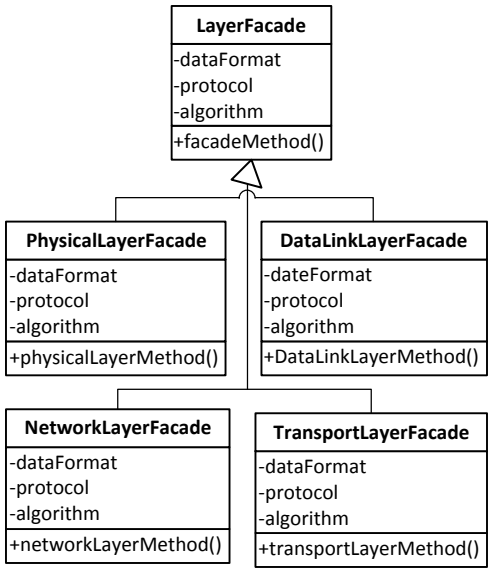
11 Façade Pattern [Gamma et al]

A façade pattern is based on the principle that all the complexities are to be hidden behind a simple interface. They encapsulate Class data and implementation and provide abstract level services. It decouples the layers so that to avoid dependency between them. Façade object acts as a gatekeeper of all method calls for the encapsulated implementation which in turn delegates the required task to perform. Following are the implementation variants of Façade pattern:

11.1 Encapsulating layered Façade [18]

This variant is based on the idea of multi network layers. Network layers present a hierarchy that encapsulates its own functionality. Façade pattern is used to define simplified interface based on some common operations that provide an access to the original functionality of the underlying protocols. This is a very useful variant for creating stubs when the original developments of the sub systems are under process. These stubs are independent of each other which can be replaced by original component facades when the subsystems are complete.

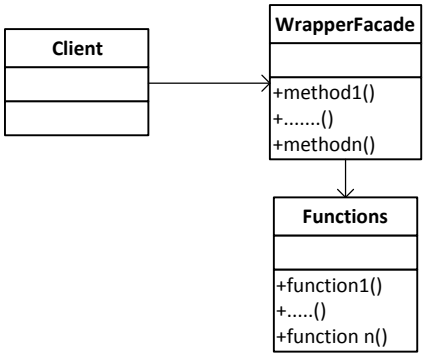


11.1 UML Diagram of layered facade [18].

11.2 Wrapper Façade [49]

This variant defines an access to the low level functions and data structures of a component. Dealing with low level functions of sockets and other OS natives is generally not a good practice.

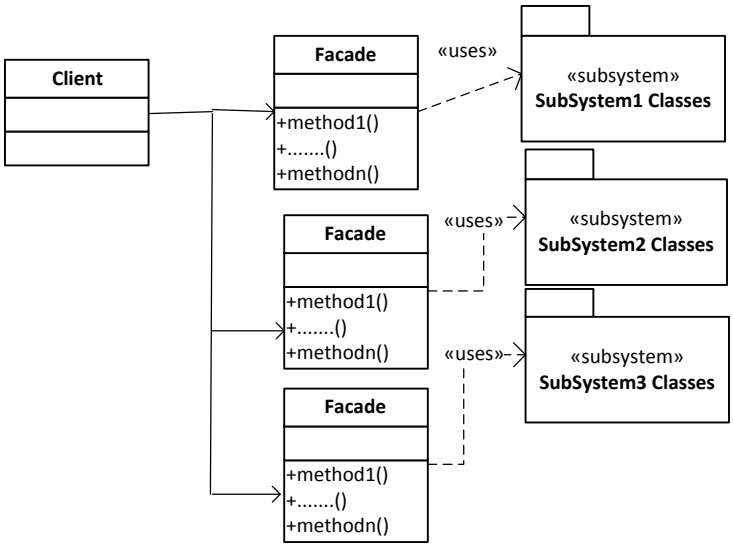
So for this purpose wrapper façade is used to redirect the low level access. Such variant is of great use while dealing with multi-platform threading issues (like Linux, Solaris and windows), sockets facades etc.



11.2 UML Diagram of Wrapper facade [49].

11.3 Sub System Façade [50]

Sub system façade is yet another important variant which provides a separate interface and encapsulate the entire subsystem inside it. This variant is very useful when subsystems functionality is integrated with other systems that may or may not lie on a current domain. The greatest advantage of using this decouples the subsystems from any external use and any change in the internal design does not affect the interface. It can be referred as well defined abstraction (gateway) of a system which is modeled as a class. [50] Suggested the sub system façade to be composed of client side proxy and server side façade with a middle tier agent such as CORBA or OLE to minimize the calls between server and client.



11.3 UML Diagram of sub system facade [50].