SDA Lab 9

Somphon Rueangsri

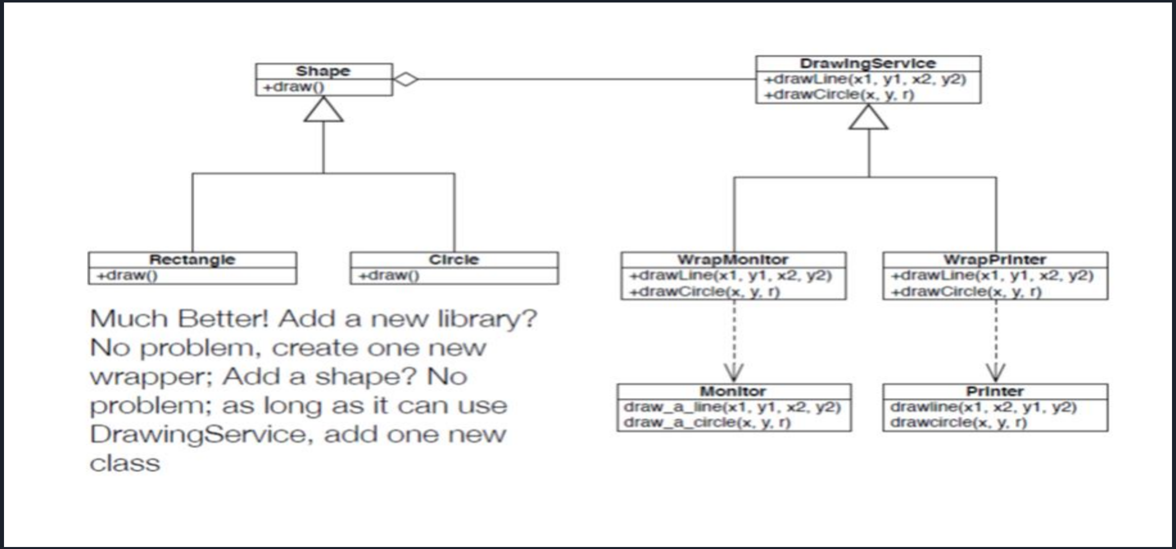
Task 1 : bridge

Task 1 :“Bridge” design pattern and explain the design problem that can be solved by this pattern.

This is a design mechanism that encapsulates an implementation class inside of an interface class.

- The bridge pattern allows the Abstraction and the Implementation to be developed independently and the client code can access only the Abstraction part without being concerned about the Implementation part.
- The abstraction is an interface or abstract class and the implementor is also an interface or abstract class.
- The abstraction contains a reference to the implementor. Children of the abstraction are referred to as refined abstractions, and children of the implementer are concrete implementers. Since we can change the reference to the implementer in the abstraction, we are able to change the abstraction implementer at run-time. Changes to the implementer do not affect client code.
- It increases the loose coupling between class abstraction and its implementation.

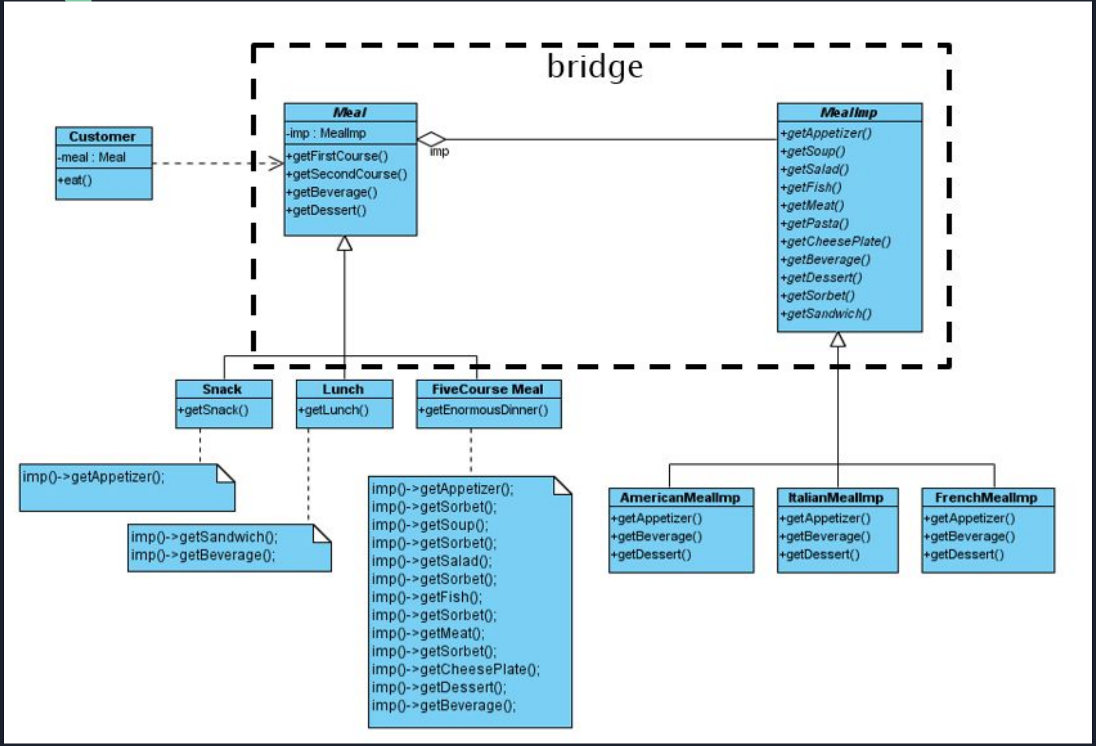
Task 2 :The Bridge Pattern allows us to “decouple an abstraction from its implementation so that the two can vary independently”. Explain this with Shape example and Meal Example.



Much Better! Add a new library? No problem, create one new wrapper; Add a shape? No problem; as long as it can use DrawingService, add one new class

In this you can add function to drawing Service class and Shape class is decouple from drawingService when you want to add other function it can be done easily as it independent on each other.

Meal example



In this you can add function in mealImp and it will not affect the meal class similar to the shape function.

Task 2 : mediator

Task 1 :1“Mediator” design pattern and explain the design problem that can be solved by this pattern.

The mediator design pattern defines an object that encapsulates how a set of objects interact.

We are used to see programs that are made made up of a large number of classes. The mediator makes the communication between objects encapsulated with a mediator object. Objects don’t communicate directly with each other, but instead, they communicate through the mediator.

A great real world example of mediator pattern is traffic control room at airports. If all flights will have to interact with each other for finding which flight is going to land next, it will create a big mess.

Task 2 :1. The Mediator Pattern allows us to “Encapsulate what varies?”. Explain what this pattern encapsulates.

Mediator pattern defines an object that encapsulates how a set of objects interact. Mediator promotes loose coupling by keeping objects from referring to each other explicitly, and it lets us vary their interaction independently.

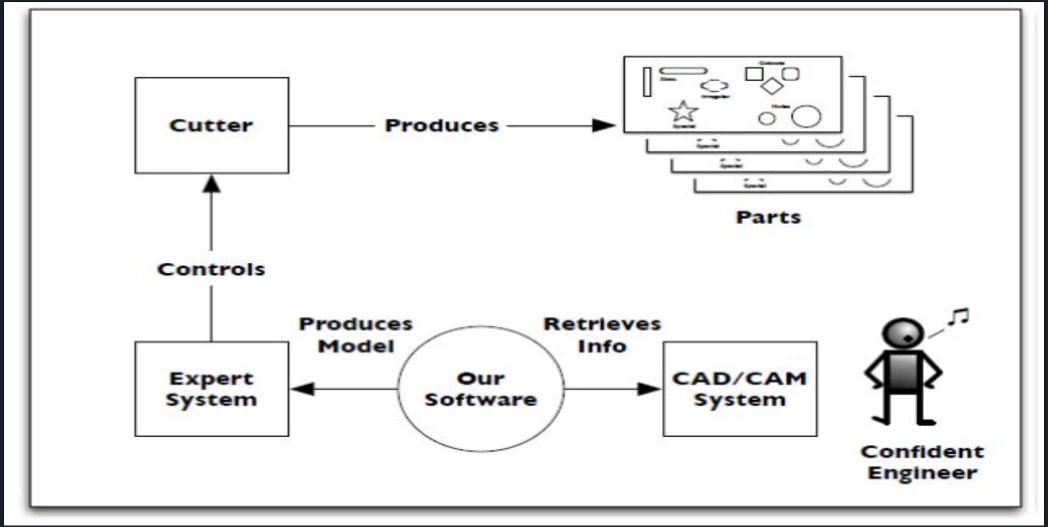
This pattern defines a separate (mediator) object that encapsulates the interaction between a set of objects and the objects delegate their interaction to a mediator object instead of interacting with each other directly.

Task 3 :Different between pattern focus design and class focus design

In class focus design we build by fitting things together or build from pieces , We build up from small piece.

In pattern focus design we build by putting together thing and is often through synthesis. It is a process of combination.

Example

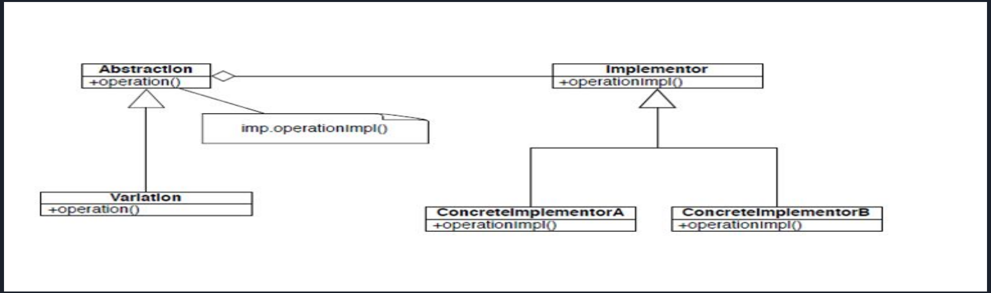


This is class focus design pattern
We build single system and fit them together.

Example

In pattern focus design it divide into 3 steps

1. Design pattern going to use for example Abstract factory, Adapter
2. Apply pattern



For example bride pattern

3. Add details

In pattern focus design is easy to maintained and can be reimplement easily.