Bridge Pattern

A Structural Design Pattern

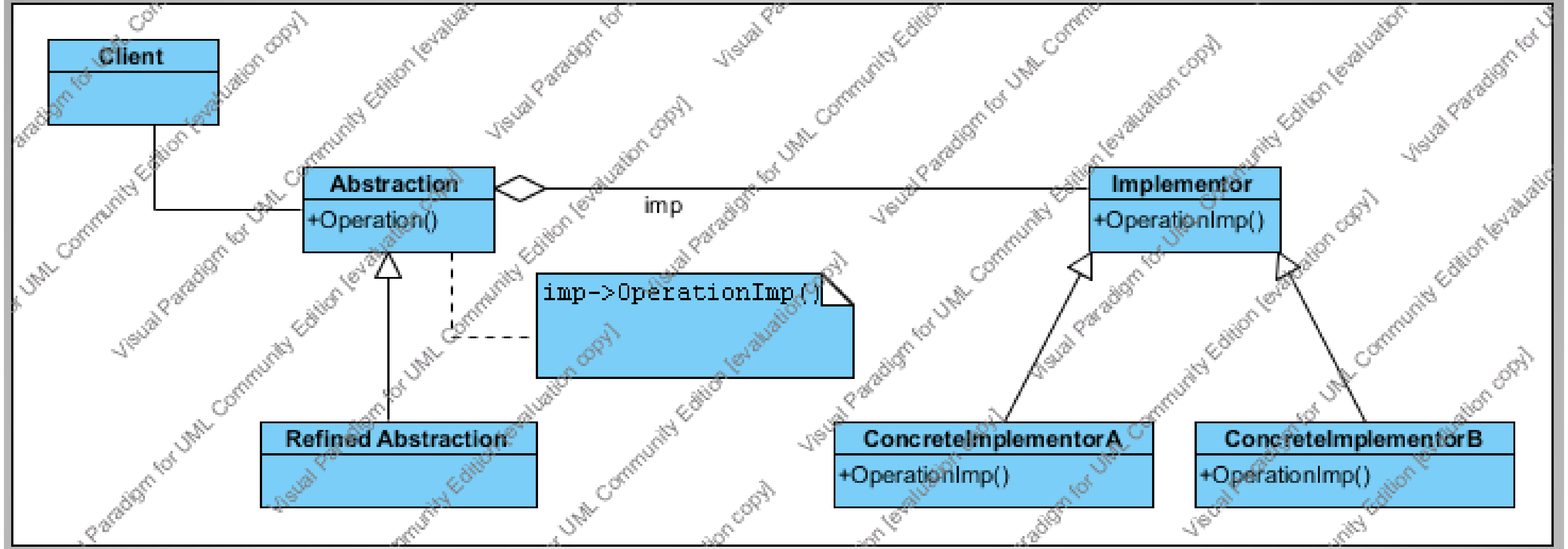
Context

- When an abstraction has several implementations, we normally do this through inheritance
- Not always flexible because inheritance binds an implementation to the abstraction permanently
- Makes difficult to modify, extend abstractions and implementations independently

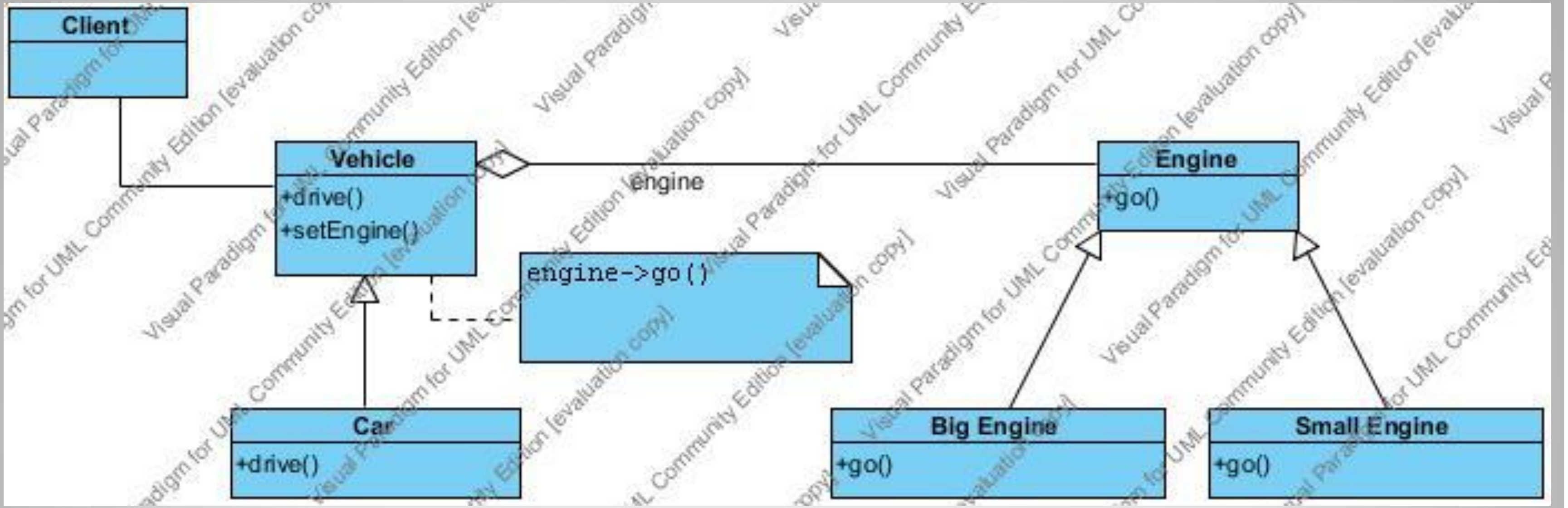
Intent

- Decouple an abstraction from its implementation so that the two can vary independently.
- Separate the abstraction and its implementation and have separate inheritance structure for both.
- Also known as “handle/body”

Structure


Client
Abstraction
+Operation()
imp
Implementor
+OperationImp()
imp->OperationImp()
Refined Abstraction
ConcreteImplementorA
+OperationImp()
ConcreteImplementorB
+OperationImp()

Example


Client
Vehicle
+drive()
+setEngine()
engine
Engine
+go()
engine->go()
Car
+drive()
Big Engine
+go()
Small Engine
+go()

Code

```
/* Abstraction */
Public abstract class Vehicle {
   Engine engine; // reference to the implementor
   Public abstract void drive();
   Public void setEngine(Engine engine) {
     This.engine = engine;
   }
}

/* Refined Abstraction */
Public class Car extends Vehicle {
    Car(Engine engine) {
    This.engine = engine;
    }

    Public void drive() {
    // Car implementation of the drive method
    Engine.go();
    }
}
```

```
/* Implementor */
Public interface Engine {
   Public void go();
}

/* Concrete Implementation1 */
Public class BigEngine implements Engine {
   Public void go() {
     System.out.println("Running big engine");
   }
}

/* Concrete Implementation2 */
Public class SmallEngine implements Engine {
   Public void go() {
     System.out.println("Running small engine");
   }
}

Public class Client {
   public static void main(String[] args) {
       Vehicle v = new Car(new SmallEngine());
        v.drive();
        v.setEngine(new BigEngine());
        v.drive();
   }
}
```

Consequences

- Decouples interface and implementation
- Change the object implementation at run-time
- Extend the abstraction and implementor hierarchies independently
- Hides implementation details from clients