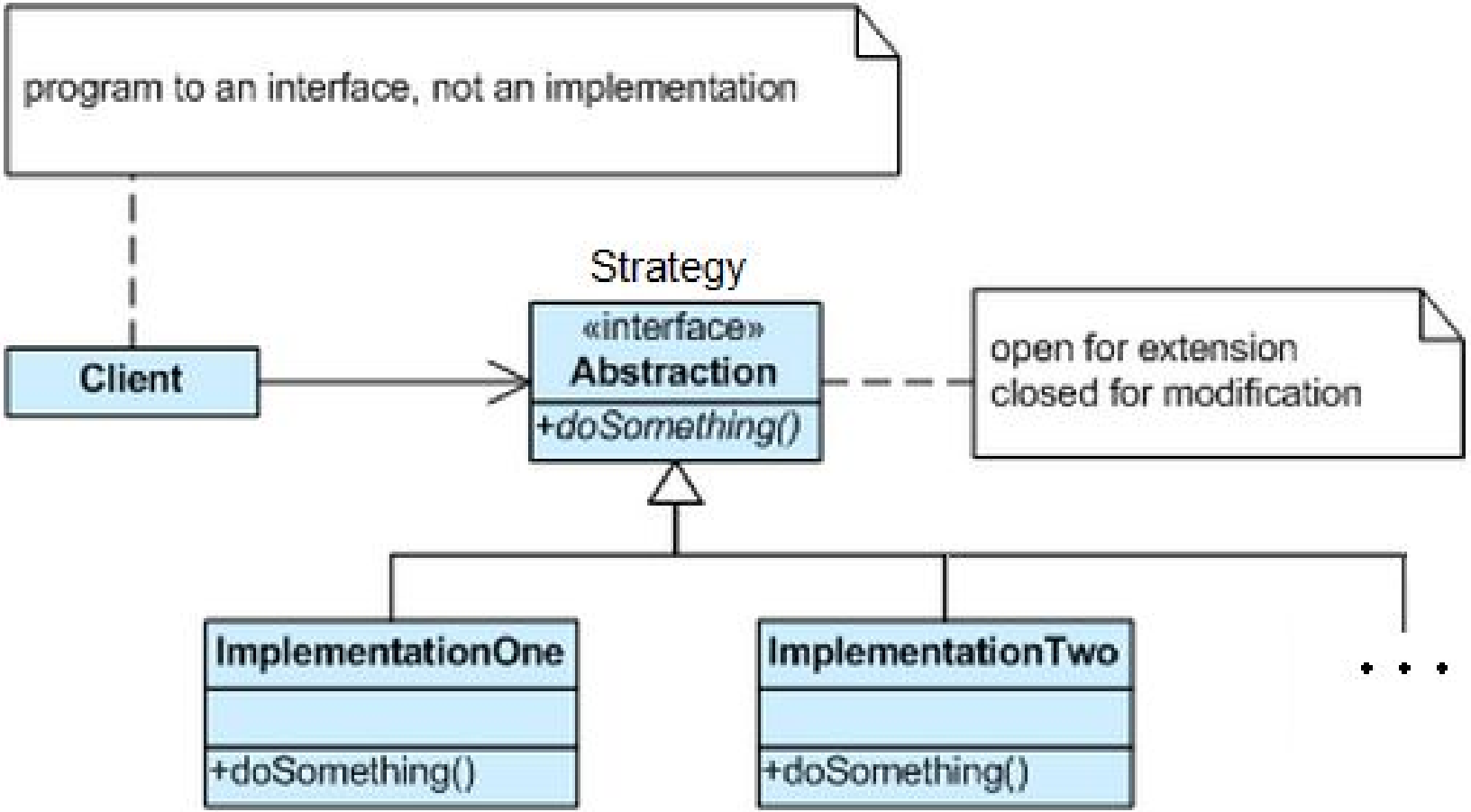
DESIGN PATTERNS

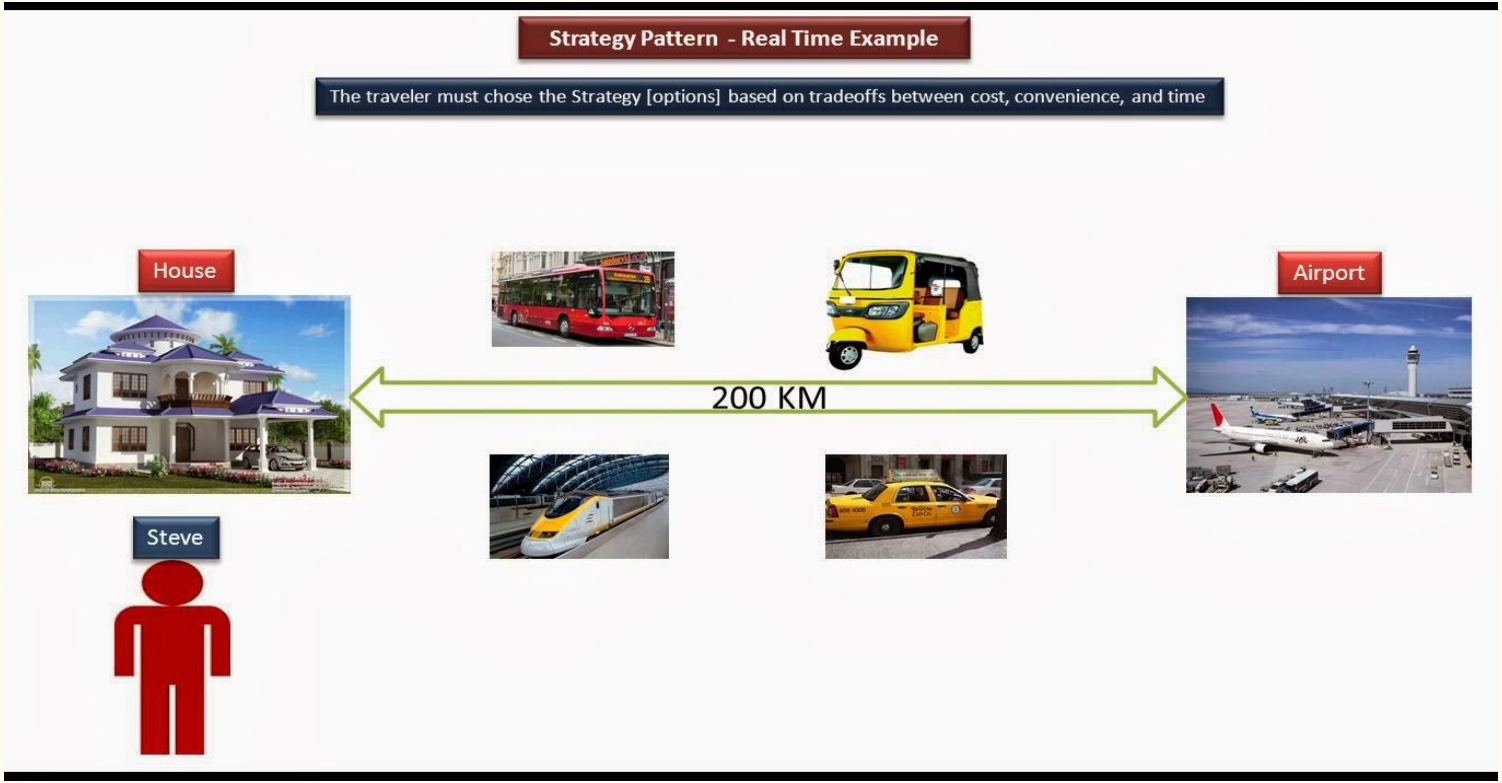
STRATEGY

Introduction

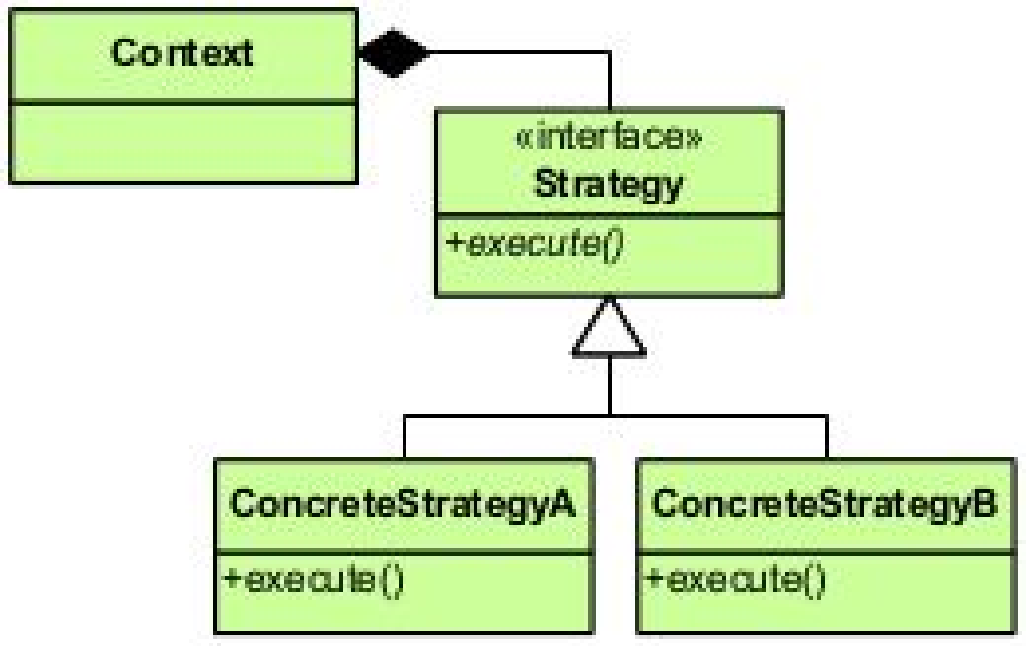
- Strategy Pattern is a behavioural pattern.
- GOF : “It is a set of encapsulated algorithms that can be swapped to carry out a specific behaviour.”
- Capture the abstraction in an interface, bury implementation details in derived classes.

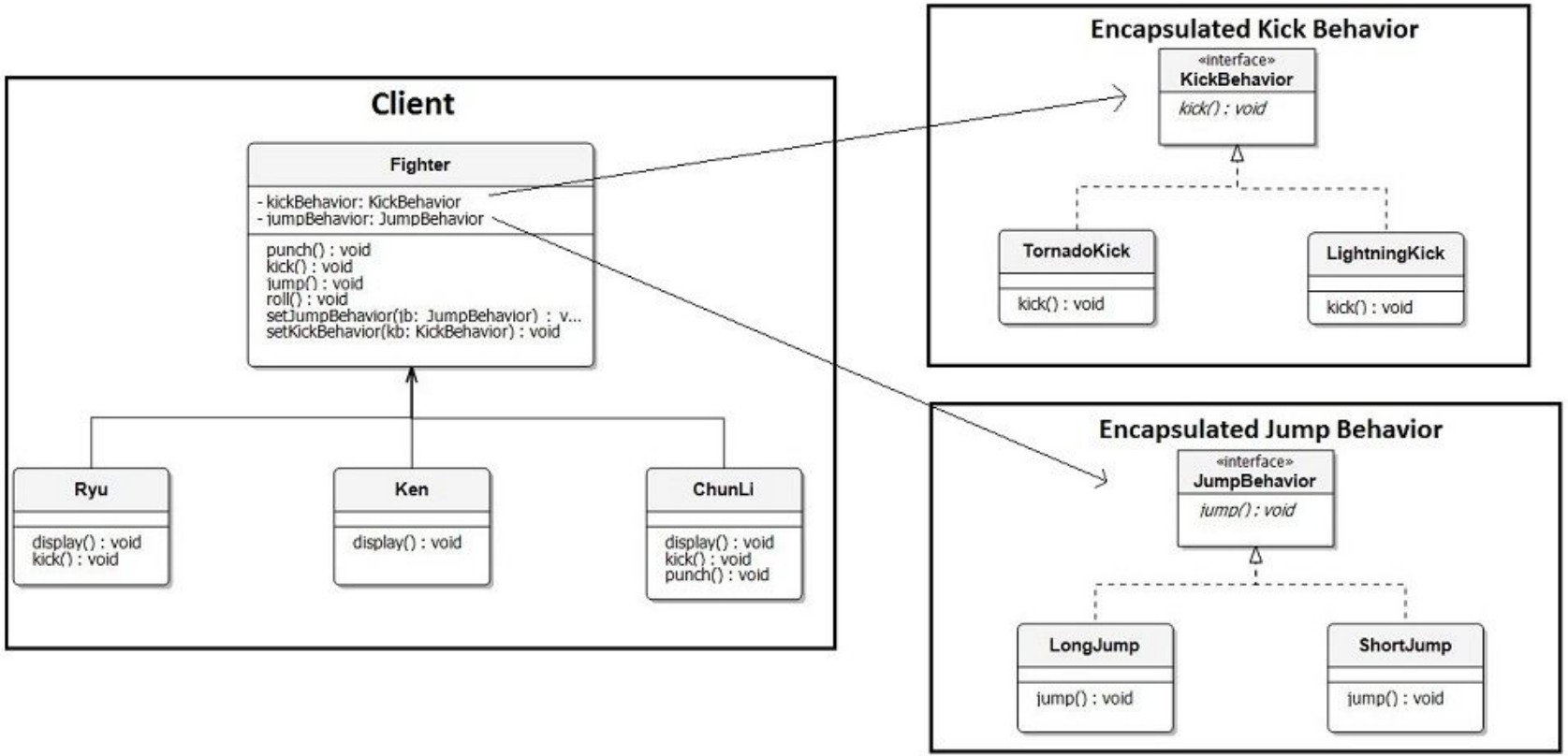

program to an interface, not an implementation
Strategy
«interface»
Abstraction
+doSomething()
Client
open for extension
closed for modification
ImplementationOne
+doSomething()
ImplementationTwo
+doSomething()

Real World Example



Structure


Context
«interface»
Strategy
+execute()
ConcreteStrategyA
+execute()
ConcreteStrategyB
+execute()


Client
Fighter
- kickBehavior: KickBehavior
- jumpBehavior: JumpBehavior
punch() : void
kick() : void
jump() : void
roll() : void
setJumpBehavior(jb: JumpBehavior) : v...
setKickBehavior(kb: KickBehavior) : void
Ryu
display() : void
kick() : void
Ken
display() : void
ChunLi
display() : void
kick() : void
punch() : void
Encapsulated Kick Behavior
«interface»
KickBehavior
kick() : void
TornadoKick
kick() : void
LightningKick
kick() : void
Encapsulated Jump Behavior
«interface»
JumpBehavior
jump() : void
LongJump
jump() : void
ShortJump
jump() : void

Let’s look at an example

Advantages

- By encapsulating the algorithm separately, new algorithms complying with the same interface can be easily introduced.
- The application can switch strategies at run-time.
- Strategy enables the clients to choose the required algorithm, without using a “switch” statement or a series of “if-else” statements.
- The implementation of an algorithm can be changed without affecting the Context class.

Disadvantages

- The application must be aware of all the strategies to select the right one for the right situation.
- Strategy base class must expose interface for all the required behaviours, which some concrete Strategy classes might not implement.
- In most cases, the application configures the Context with the required Strategy object. Therefore, the application needs to create and maintain two objects in place of one.

Thank You!