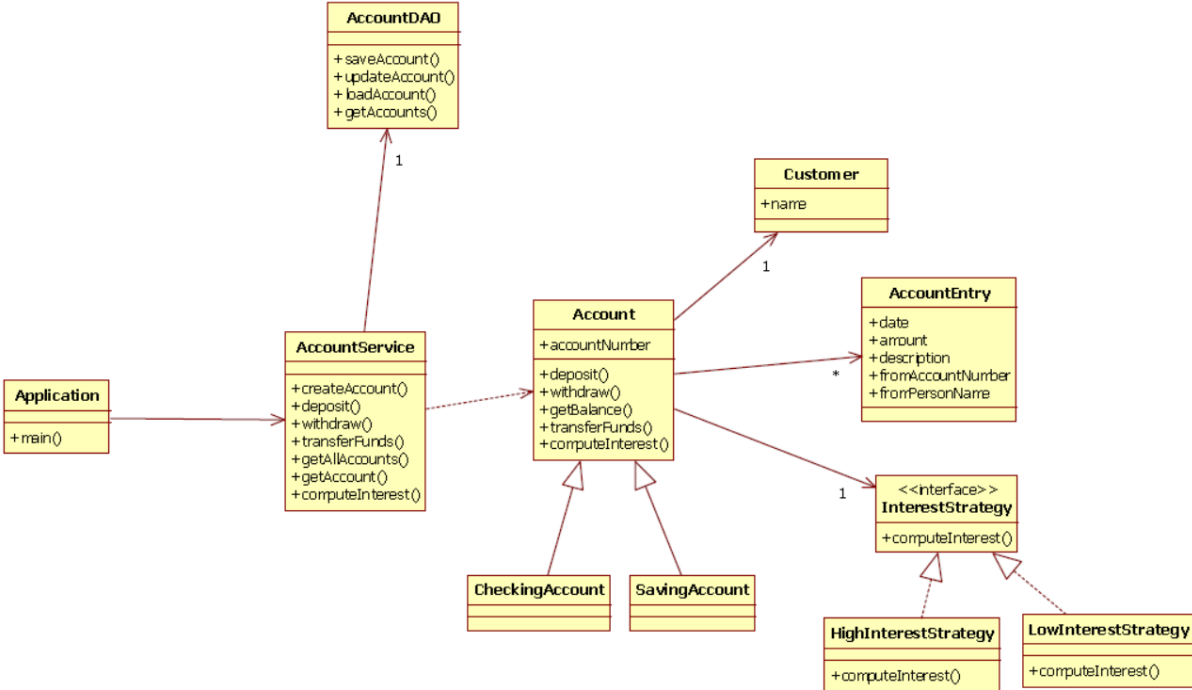
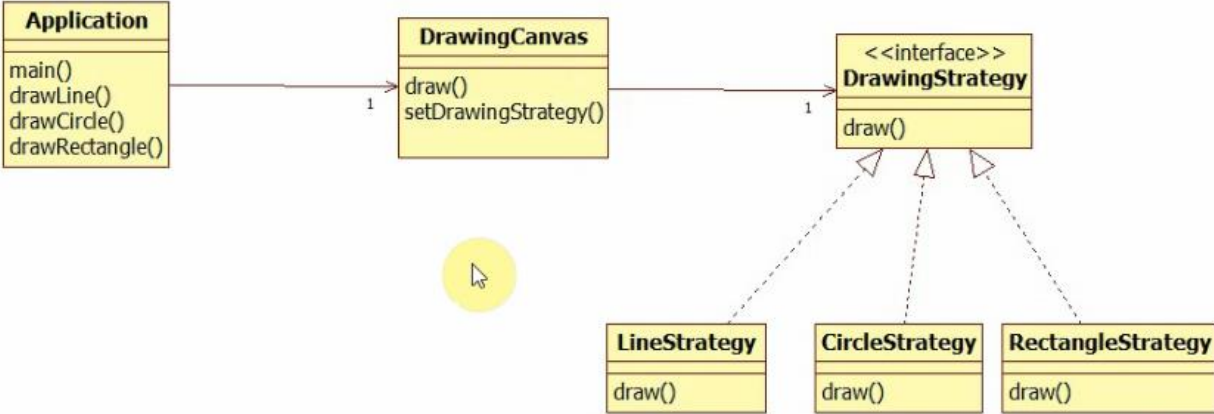
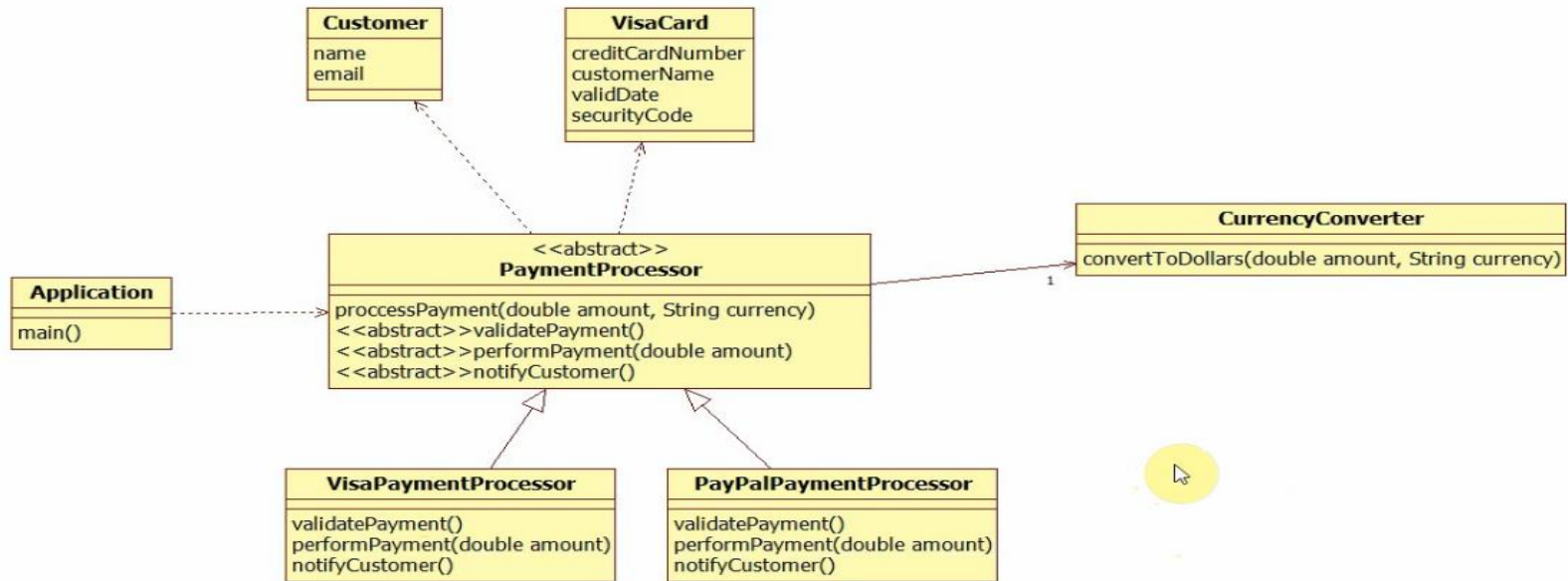


Usage & Benefits	Main Classes	Inheritance
<p>1 Facade</p> <p>Unified interface to a complex set of classes. It hides the complexity from the client(s)</p>	<p>Service Class calls all other classes.</p> 	<p>Pure Consciousness provides a unified interface to all aspects of creation, and the daily experience of Pure Consciousness makes life much more enjoyable</p>
<p>2 Strategy,</p> <ul style="list-style-type: none"> • We can easily add new algorithms without • changing the context class • The strategies are better reusable • define a family of algorithms, encapsulate each one as an object, and make them interchangeable. • Whenever you want to choose the algorithm to use at runtime. 	<p>Abstract calls or interface</p> 	<ul style="list-style-type: none"> • With the strategy pattern, different algorithms are extracted from its context and encapsulated as strategy classes • The unified field is the field of all possibilities.

3 Template

Common algorithm with different steps that is used in different situations, then define this algorithm in one place (parent class) and let child classes implement the concrete steps.

Abstract class:



- The template method defines an algorithm in the parent class and the different steps can be implemented in the child class(es)
- Every human being has free will to decide how to live your life within boundaries of the laws of nature.

4 Observer

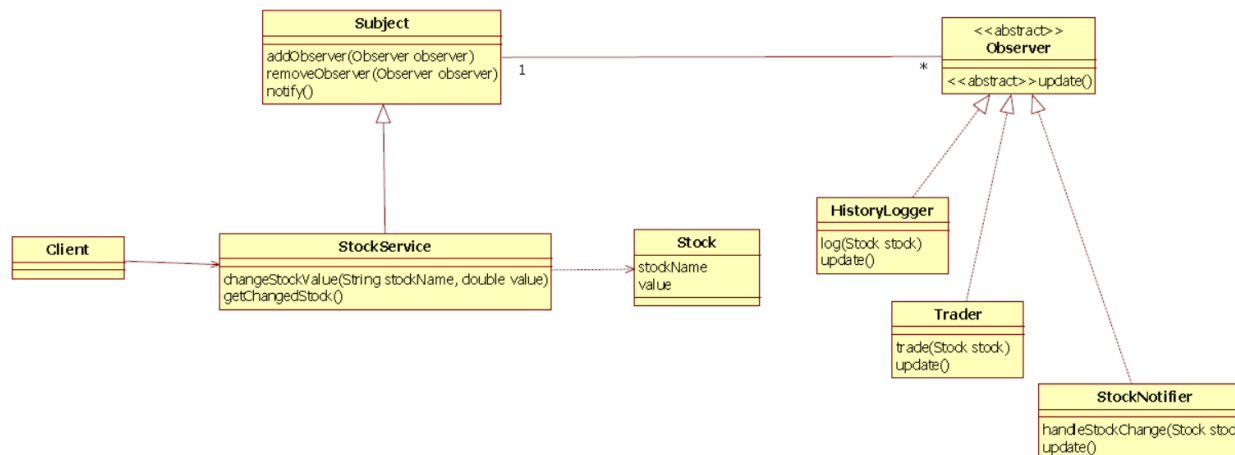
The Observer design pattern lets several observer objects be notified when a subject is changed in some way

Subject :

- addObserver(Observer observer)
- removeObserver(Observer observer)
- notify()

Abstract class: if pull → <<abstract>>update()

- Interface : → update(int id) if I use push, as I don't need to have method in abstract, I will get the data from the subject



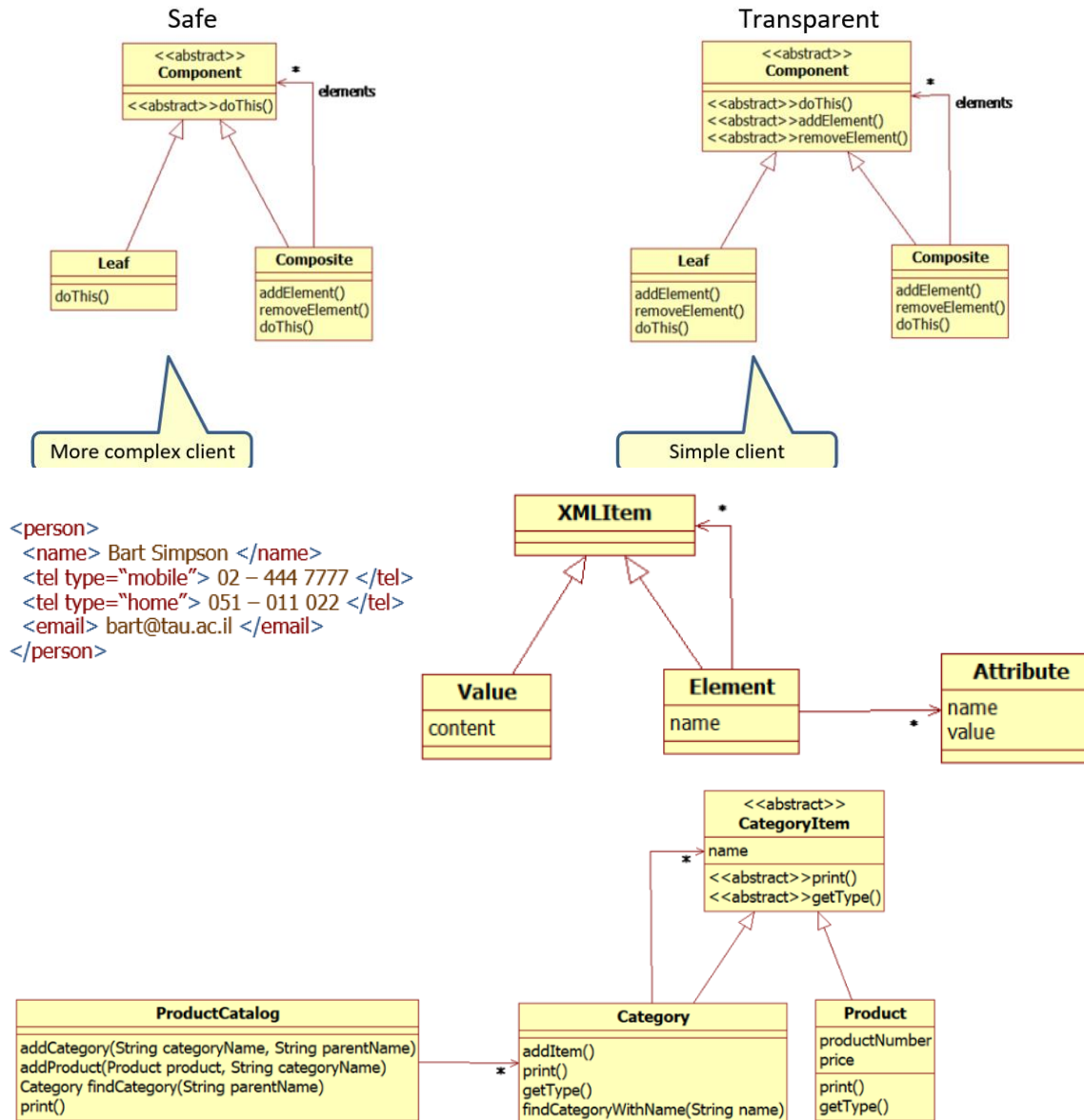
- The observer pattern makes observables (publishers) independent of observers (subscribers)
- All human beings have the ability to observe and live the intelligence of nature

5 Composite

- Menus with sub-menus
- The intent of this pattern is to compose objects into tree structures to represent partwhole hierarchies.

Composite lets clients treat individual objects and compositions of objects uniforml

XML structure, Filesystem, Product catalog, UI framework



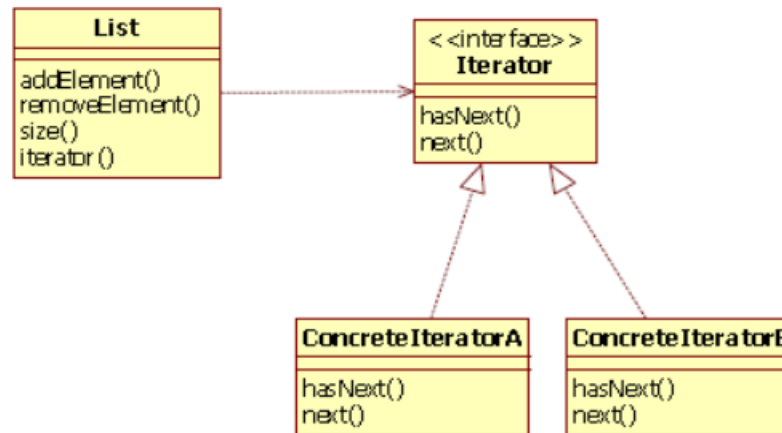
- The composite pattern can be used to represent tree structures and we want to handle the tree elements in an uniform manner.
- The structure of the universe emerges from pure consciousness which is the basis of all life.

6 Iterator

- Used separate the iteration functionality from the aggregates.

Use the same class for each iterators.

- External**
Iterator: while or for(String letter: alphabet){
System.out.println(letter.toUpperCase());
- Internal**
iterator:
alphabet.forEach (l -> System.out.println(l.toUpperCase()));

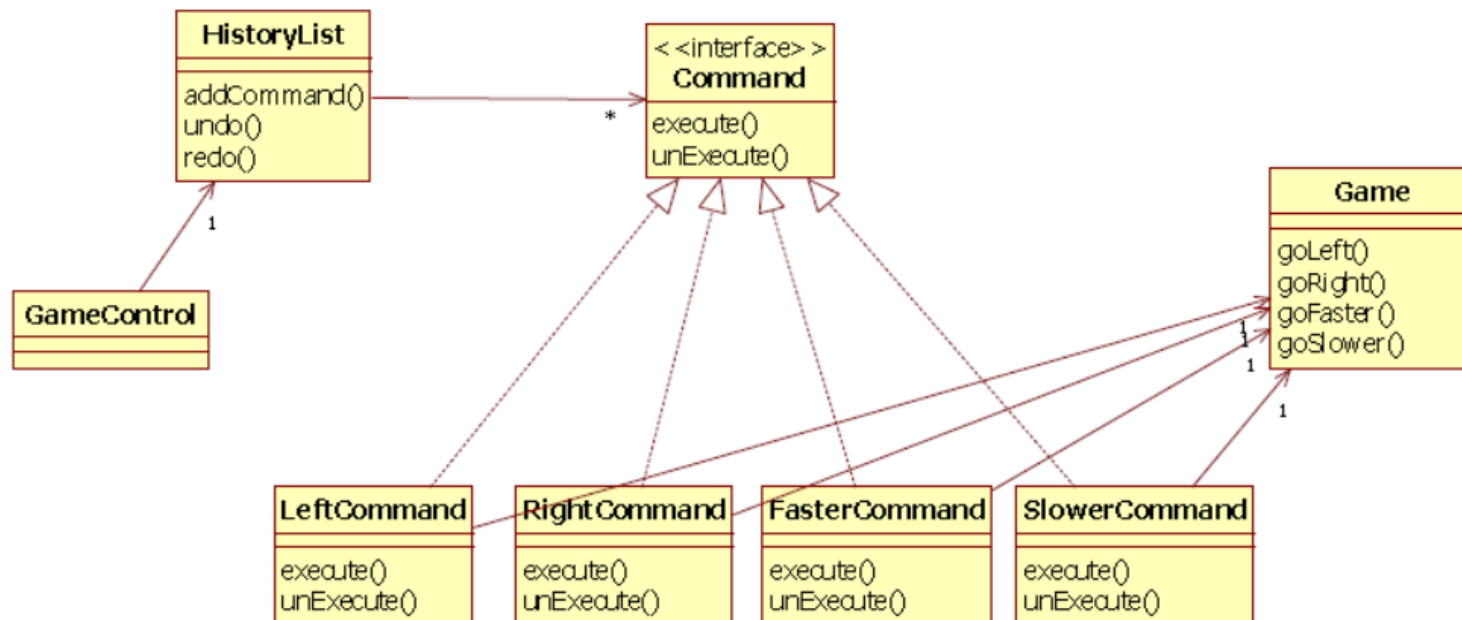


- The iterator pattern separates the iteration functionality from the collection so that the client is unaware of the structure of the collection.

When one grows in consciousness, one spontaneously starts to live in harmony with all elements in creation without knowing all the details.

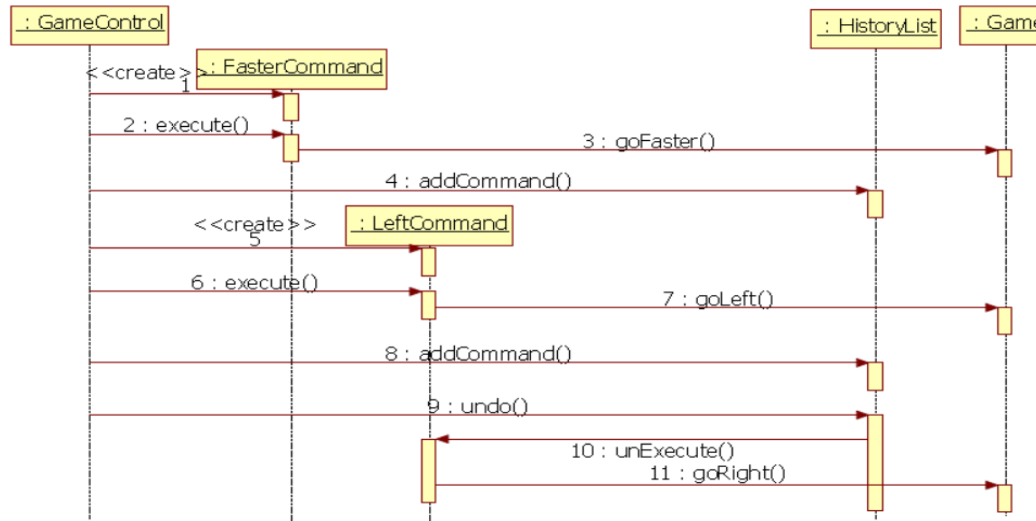
7 Command

- Encapsulate a request into a single object
- Advantages:
 - Command objects can be logged
 - Command objects can be used for undo/redo
 - Functionalit Command objects can be replayed
- Whenever you need to know the actions taken by a user,
- undo/redo functionality.



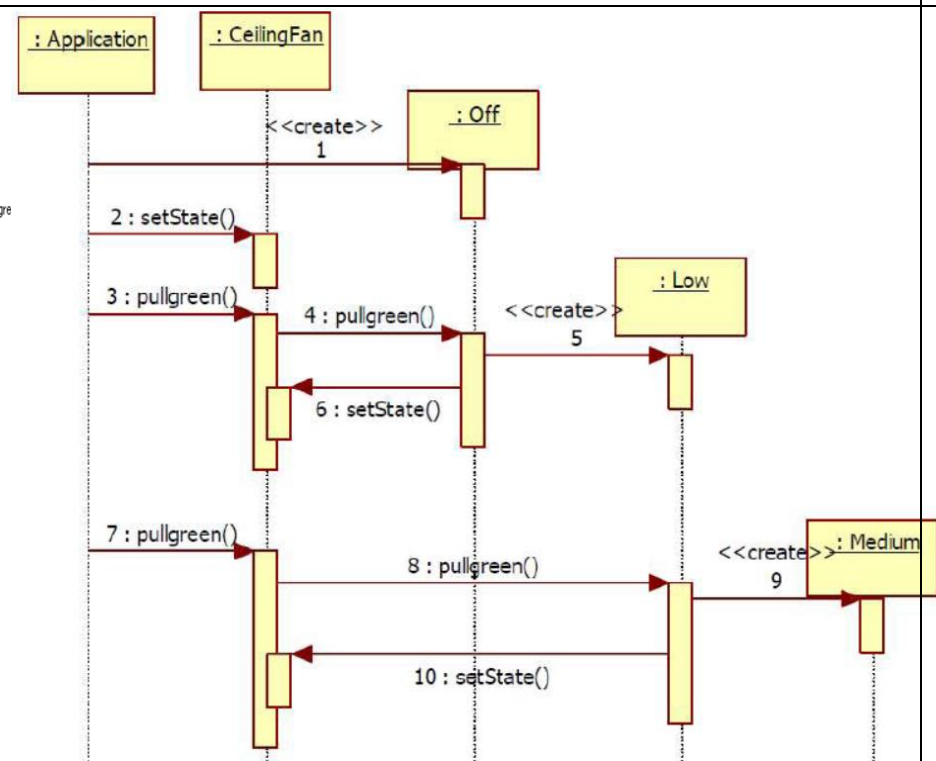
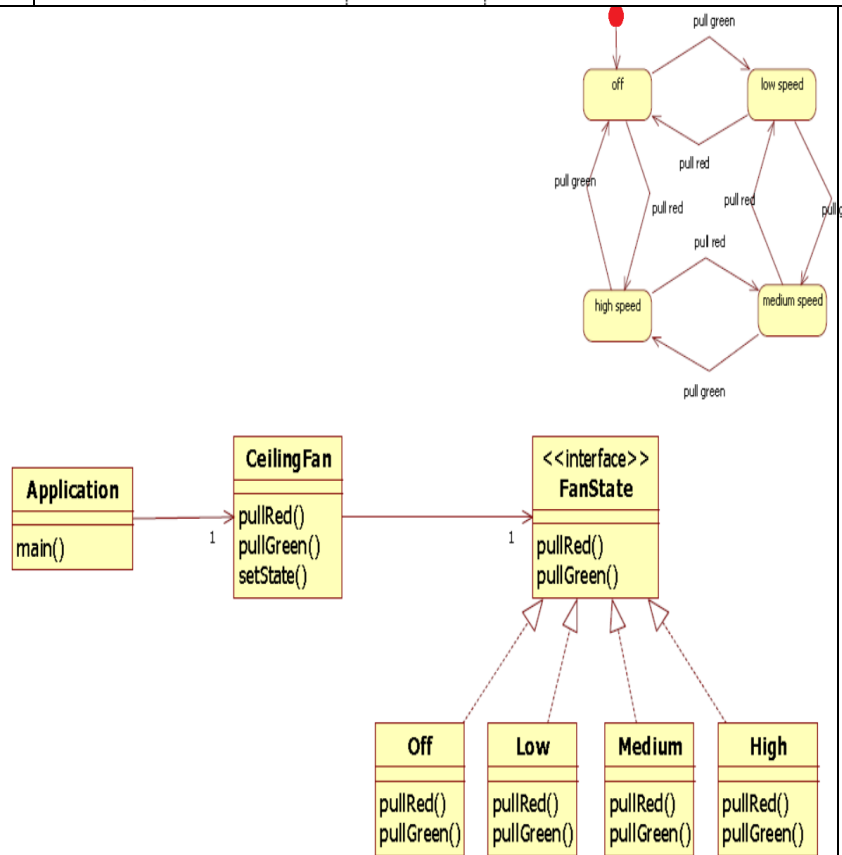
- The command pattern encapsulates a request as an object.
- Undo/redo functionality can be implemented by recording a HistoryList of Command objects.
- Transcendental consciousness is the source off all activity.
Wholeness moving within

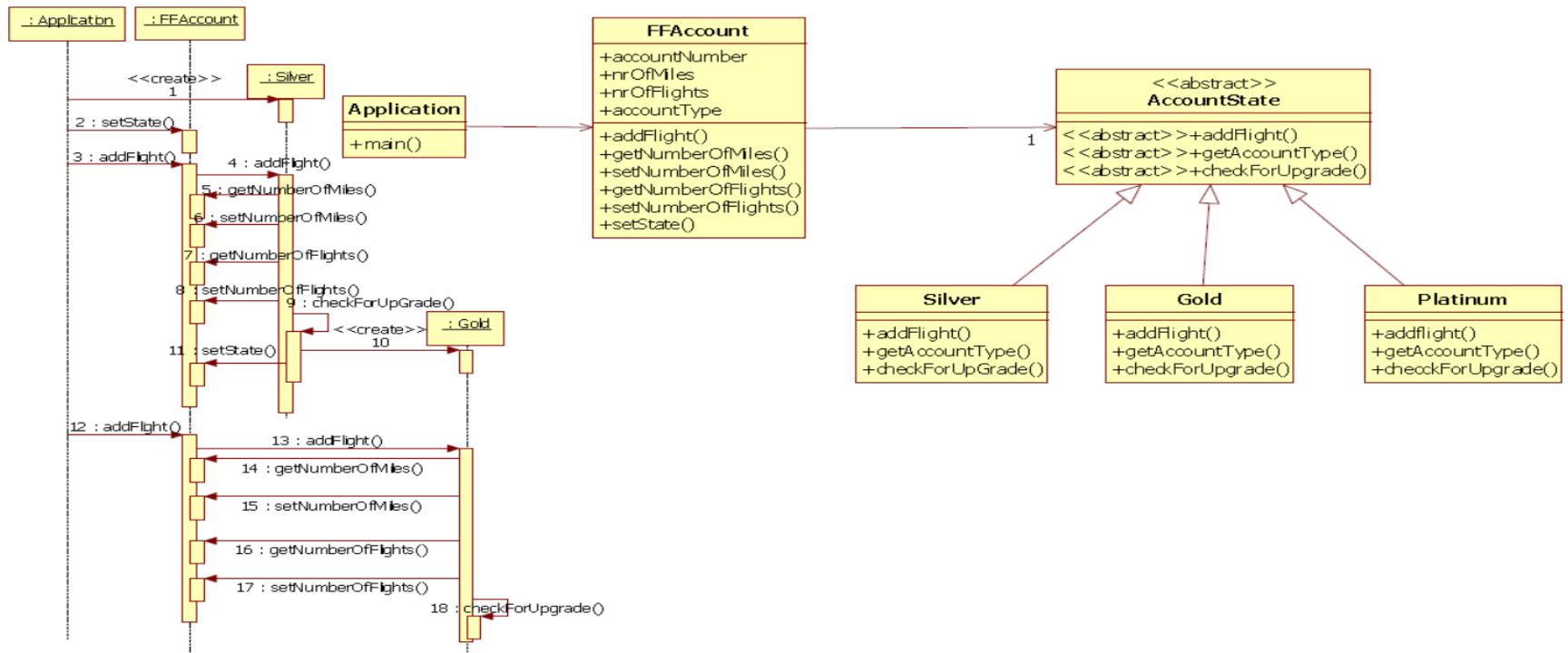
- support for macros (recording and playback of macros).



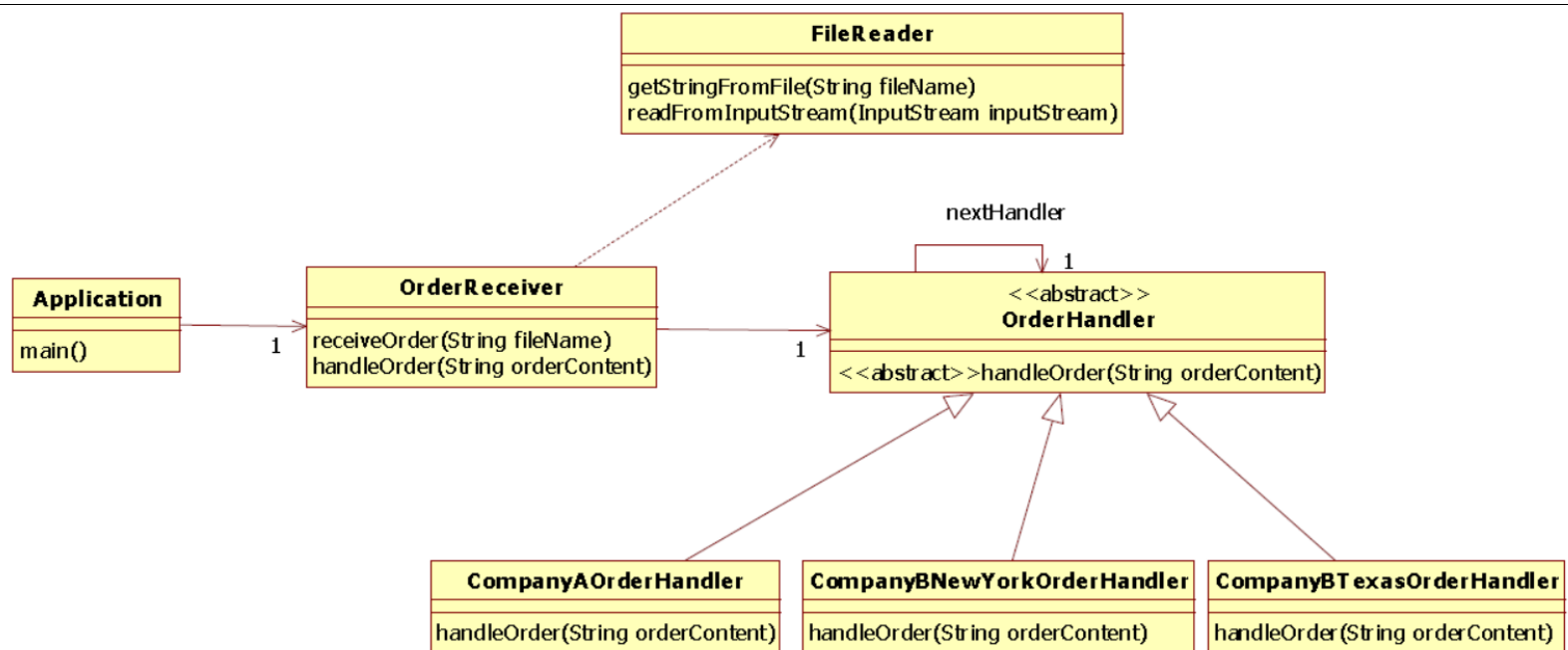
8 State

The state pattern is a design pattern that allows an object to completely change its behavior depending upon its current internal state.

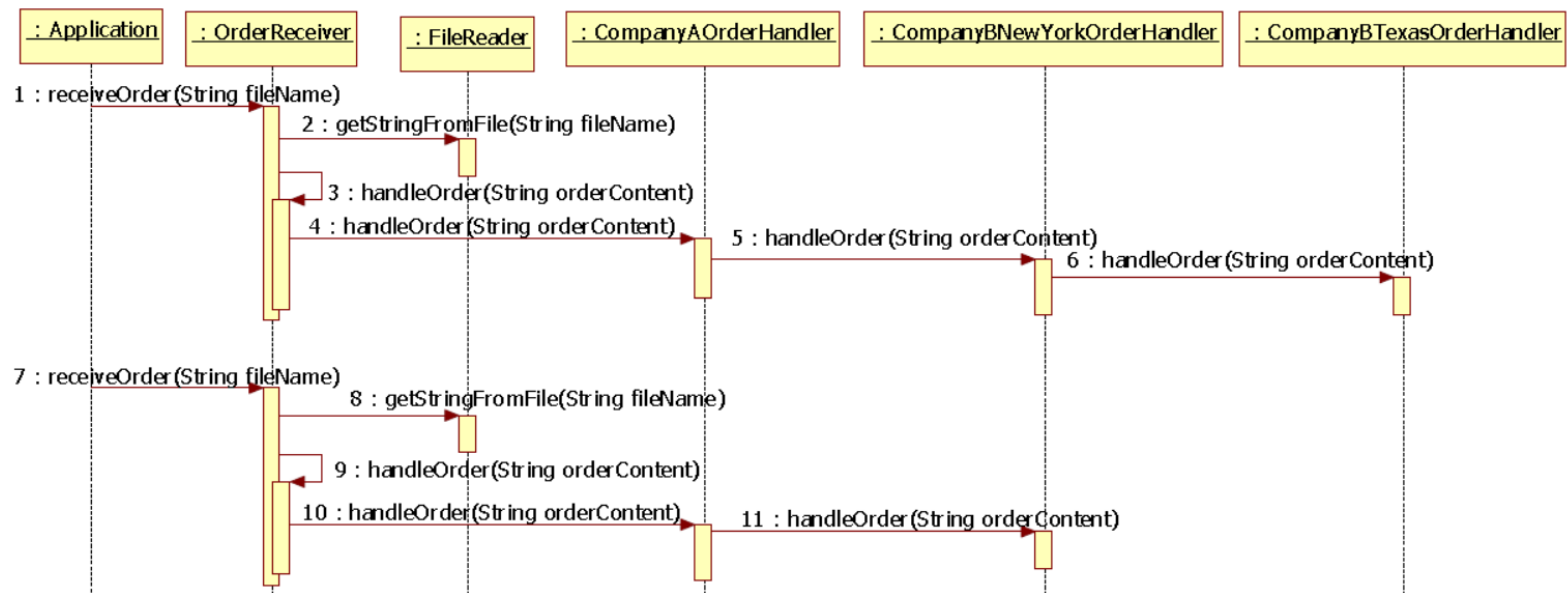




- 9 **Chain Of Responsibility pattern**
The chain of responsibility pattern transforms complex if-then-else logic into many simpler if-then-else structures.



1. With the chain of responsibility pattern we chain different handlers together.
2. The chain of responsibility pattern transforms complex if-then-else logic into



many simpler if-then-else structures.

3. Transcendental consciousness is the source off all activity.

4. Wholeness moving within itself: In Unity Consciousness, one realizes the unity between everything around you.

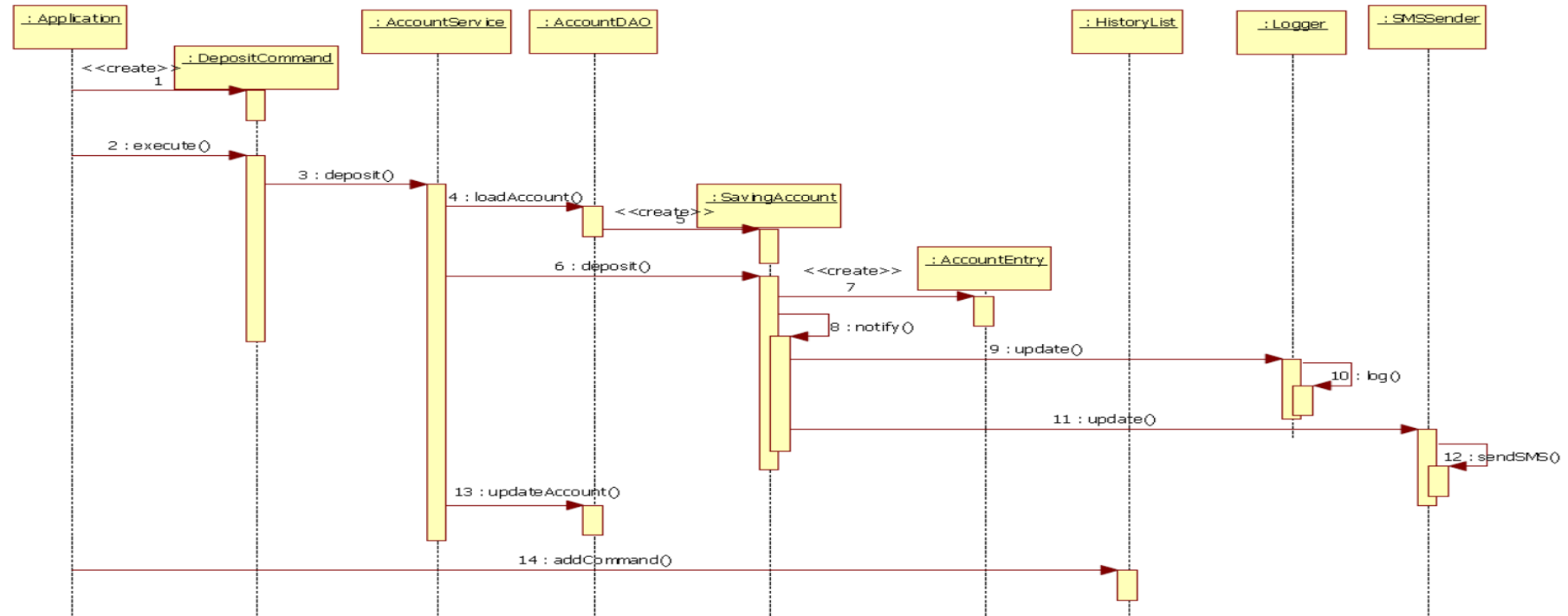
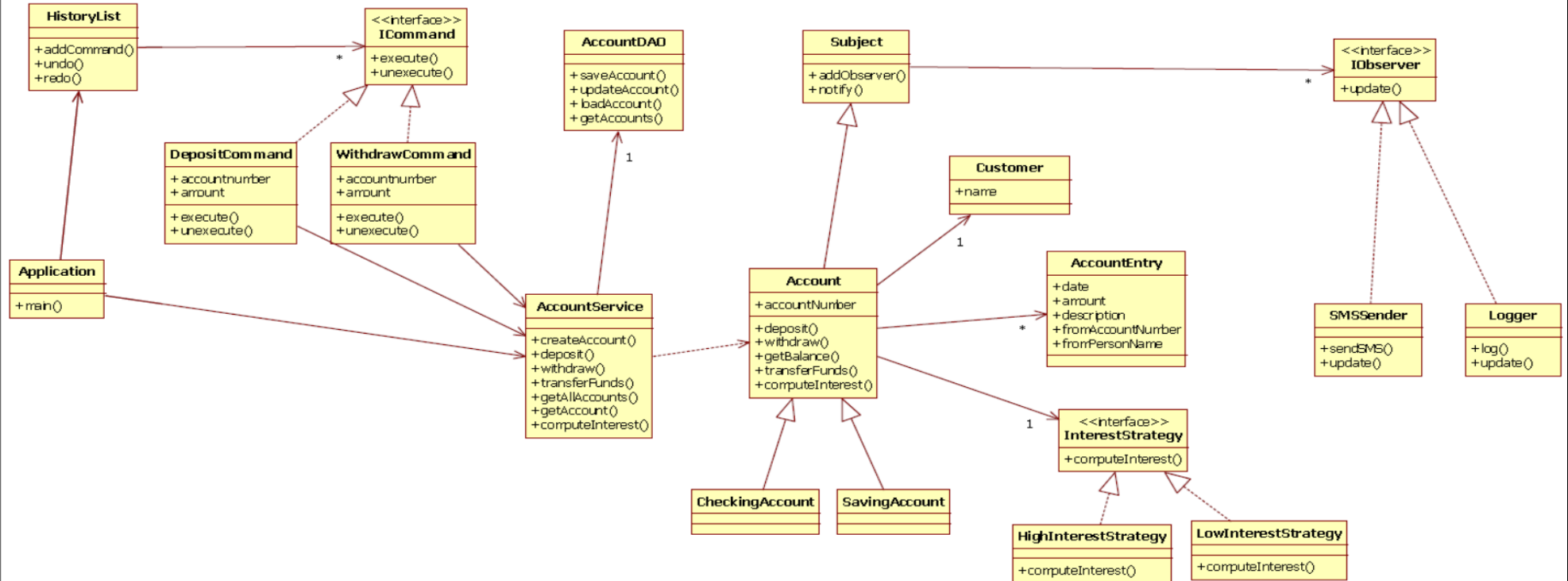
The state pattern can be applied whenever we have complex state logic.

2. The state pattern transforms complex if-then-else logic into many simpler if-then-else structures.

3. Transcendental consciousness is the source off all relative states.

4. Wholeness moving within itself: In Unity Consciousness, one experiences the unity between yourself and all of creation.

Strategy + observer + command



Strategy + State + command + Observer

