Monday
8-6-2020

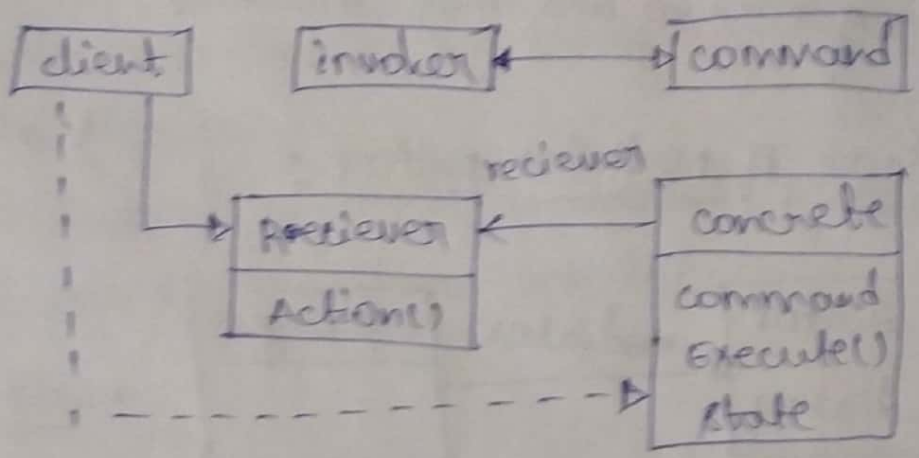
DP Assignment-4

UNIT-4

N.V.S.K. kalyani
17UN1A0584
3rd B.Tech CSE-C

2 marks

1) Structure of command pattern :-




2) Implementation of chain of responsibility pattern :-

* Implementing successor chain Define new links and use existing links.
* connecting successors
* Representing requests
* Automatic forwarding in small talk.

3) Interpreter pattern intent & applicability :-

Intent: Given a language, defines a responsibility for its grammar along with an interpreter that uses the representation to interpret sentences in the language.

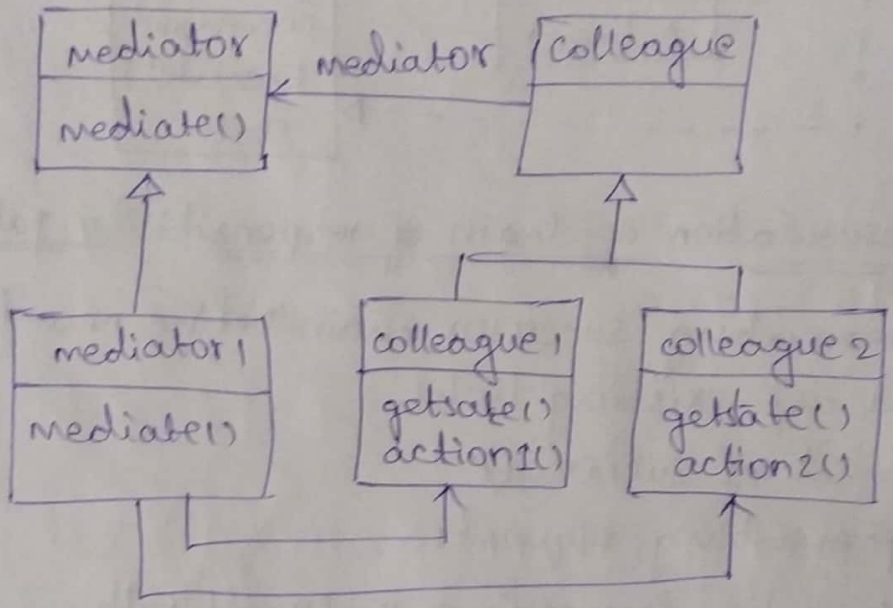
Applicability

* The grammar is simple
* Efficiency is not a critical expression.

4Ans Consequences of Memento pattern:-

* preserving encapsulation boundaries
* It simplifies originator
* Defining narrow and wide interface.

5Ans Structure of Mediator pattern:-




10 marks

1Ans ITERATOR PATTERN

Intent:- provide a way to access the elements of an aggregate object sequentially without exposing its underlying representation.

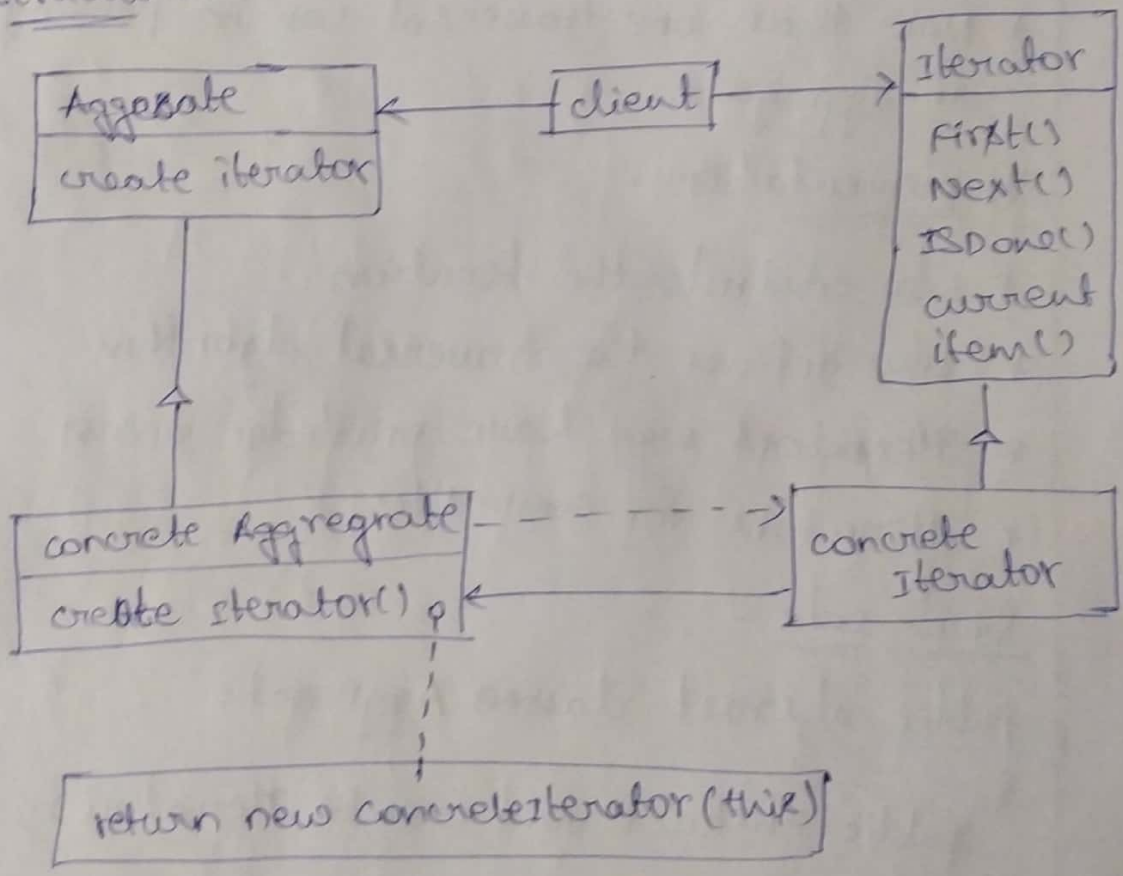
Also known as: cursor

Motivation:-

* The Iterator pattern provides a client with a way to access the elements of an aggregate object sequence without having the to known the underlying representation.

* Iterator also provides a way to define special Iterator classes that perform unique processing and return specific elements of the data collection
* It provides clients with common Interface so it doesn't need known any thing about the underling data structure.
* To support multiple traversals of aggregate objects
* To provide a uniform interface.

Structure :-




participants:-

* Iterator
* Concrete Iterator (ListIterator, SkipList, Iterator)
* Aggregate (Abstract List)
* Concrete Aggregate (List, SkipList)

Collaboration:-

* A concrete Iterator keeps track of the current object in the aggregate and can compute the succeeding object in the traversal.

Consequences:-

* It support variations in the traversal of an aggregate.
* Iterators simplify the Aggregate interface.
* More than one traversal can be pending on an aggregate.

Implementation:-

* who controls the iteration
* who defines the traversal algorithm
* Iterators may have priviledge access
* Iterators for composites

Sample code:-

```
public abstract class Aggregate
{
  public abstract void createIterator();
}
public class concrete Aggregate implements
                                  Aggregate
{
  public void concreteIterator();
}
public concrete command Extends command
{
  public interface Iterator
  {
```

```
    public void test();
    public void next();
    public void isDone();
}
public class concrete Iterator implements
                                    Iterator
{
    public void first()
    {
        //
    }
    public void next()
    {
        //
    }
}
```

Known uses:-

* Tree Traversal
* Spell checking

Related patterns

* composite
* Memento
* Factory method

R1

## OBSERVER PATTERN

Intent:- Define a one to many dependency between objects. So that when one object change state all its dependent are notified and update automatically

Also known as:- Dependent, publish Subscribe.

Motivation:-

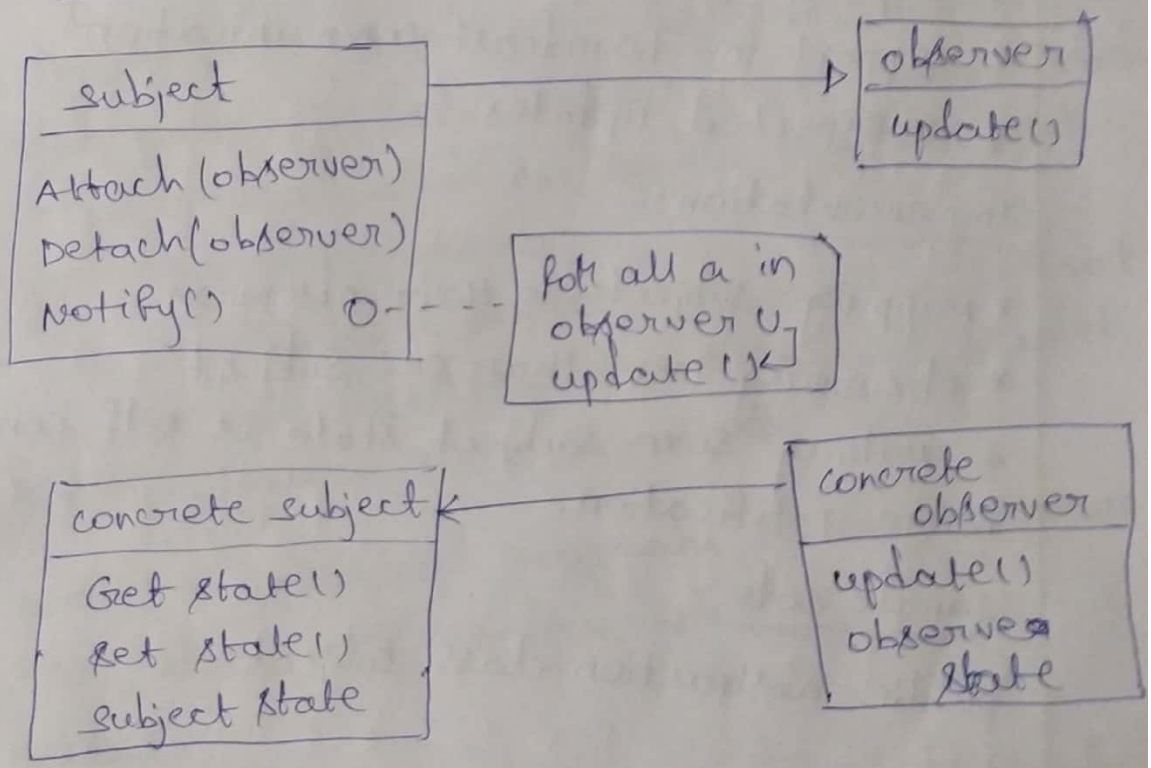
* partitioning a system into a collection of co-operating classes requires maintaining consistency b/w relating objects
* Two parts to the observers patterns.
  - subject
  - observer

* Relationship between subject and observer is one-to-many it needs to be de-coupled to make the subject and observer independently reusable.

Applicability:-

* when an change to an object requires changing others.
* Abstraction has two aspects one dependent on the other.

Structure:-




participants:-

1. subject:
   * knows it observer
   * can have any number of observers
2. observer
   * define an update interface for objects that should be notified

3. concrete subject
4. concrete observer

collaboration:-

* concrete subject notifies its observers whenev
a change occurs that could its observer's sta
inconsistent with its own.
* After this, a concrete may query the subject
for information and then change its internal stat

consequences:-

* coupling btw subject and observer
* support for boardcast communication
* unexpected updates

Implementation:-

* Mapping subjects to their observers.
* observing more than one subject
* Making sure subject state is self consiste
before notification.

sample code:-

```
public abstraction class subject
{
    observer o;
    public void attach();
    public void detach();
}
public class concrete subject Extens subject
{ public subject GetState()
  { //
  }
```

```
public void setstate()
{
  //
}
public abstract class observe
{
  public abstract void update();
}
public class concrete observer Extends observe
{
  public void update()
  {
    //
  }
```

known uses :-

* Broadcast communication
* Small talk mvc

related patterns

* Mediator
* Singleton