MODULE 5

8.00 # GRASP –

8.30 General Responsibility Assignment Software Patterns (Principles) (GRASP), consists of guidelines for assigning responsibility to classes and objects in object oriented design. It is a mental toolset, a learning aid to help in the design of object-oriented software.

9.00
9.30
10.00
10.30
11.00 → Patterns / Principles used in GRASP –
11.30
12.00 1) Controller : it assigns the responsibilities of dealing with system events to a non-UI class that represents the overall system or a use case scenario. It is responsible for recieving or handling a system event.
12.30
13.00
13.30
14.00 2) Creator : Creation of objects is one of the most common activities in an object-oriented system. Which class is responsible for creating objects is a fundamental property of the relationship b/w objects of particular class.
14.30
15.00
15.30
16.00
16.30 3) High Cohesion : it is an evaluative pattern that attempts to keep objects appropriately focused, manageable & understandable. It means that responsibilities of a given element are strongly related and highly focused.
17.00
17.30
18.00
Evening

4) Information Expert : it is a principle used to determine where to delegate responsibilities. Using this principle, a general approach to assigning responsibilities is to look at a given responsibility, determine the information needed to fulfill it, and then determine where the info. ~~needed~~ is stored.

Meetings | Things To Do | Important Calls

5) Low coupling - it is an evaluative pattern, which dictates how to assign responsibilities to support :
   - lower dependency b/w the classes
   - change in one class having lower impact on another class.
   - higher reuse potential.

6) Polymorphism - According to polymorphism, responsibility of defining the variation of behaviors based on type is assigned to the types for which this variation ~~occurs~~ happens.

7) Protected Variations - it protects elements from the variations on other elements by wrapping the focus of instability with an interface and using polymorphism to create various implementations of this interface.

8) Pure Fabrication - It is a class that does not represent a concept in problem domain, specially made up to achieve low coupling, high cohesion, and the reuse potential thereof derived.

20 SUNDAY

GoF Patterns -

The patterns are based on following principles of object oriented design -

i) Program to an interface not an implementation

ii) Favour object composition over inheritance.

To

Meetings ✓ | Things To Do ✓ | Important Calls ✓

→ Types:

8.00
8.30 1) Factory Pattern: it is one of the most used pattern in JAVA. In factory pattern, we create object without exposing the creation logic to the client and refer to newly created object using a common interface.

- Benefits -

i) Seperate the responsibilities of complex creation into cohesive helper objects

ii) can provide object caching.

9.00
9.30
10.00
10.30
11.00
11.30
12.00
12.30 2) Singleton Pattern: It involves a single class which is responsible to create own objects while making sure that only single object is created. It provides a way to access its only object which can be accessed directly without the need to instantiate the object of the class.

13.00
13.30
14.00
14.30
15.00 3) Adapter Pattern: It works as a bridge b/w two incompatible interfaces. It involves a single class which is responsible to join functionalities of independent or incompatible interfaces.

15.30
16.00
16.30 eg: Case of card reader which acts as adapter b/w memory card and a laptop. Memory card is plugged into a ~~card reader~~ laptop using a carel reader.

17.00
17.30
18.00
Evening 4) Observer Pattern: It is used when there is one to many relationship b/w objects such as if one object is modified, its dependent objects are to be notified automatically.

eg: Alarm clock is publisher of alarm events. Different types of objects can register as listners & react to same alarm event in their own ways.

Meetings | Things To Do | Important Calls

22 Week 12
March
Tuesday (082-284)

8.00
8.30
9.00
9.30
10.00
10.30
11.00
11.30
12.00
12.30
13.00
13.30
14.00
14.30
15.00
15.30
16.00
16.30
17.00
17.30
18.00
Evening

Operation Contracts -

Use cases or system features are the main ways in the UP to describe system behavior, and are usually sufficient. Sometimes a more detailed or precise description of system behavior has value. Operation contracts use a pre-condition and post-condition form to describe detailed changes to objects in a domain model, as the result of a system operation.

State Diagram —

It describes diff states of a component in a system. The state are specific to a component / object of a system. It defines diff states of an object and these states are controlled by external or internal events.

- Purpose -
  - to model dynamic aspect of system
  - to model the life time of a reactive system
  - to describe different states of an object during its lifetime.

Deployement Diagram -

They are used to visualize the topology of the physical components of the system, where the software components are deployed. They are used to describe the static deployement view of a system. It consists of nodes and their relationships.

- Purpose -
  - Visualize the hardware topology of system
  - Describe the runtime processing nodes.
  - Describe the hardware components used to deploy software components.

Meetings | Things To Do | Important Calls

April

Week	13	14	15	16	17
Monday		4	11	18	25
Tuesday		5	12	19	26
Wednesday		6	13	20	27
Thursday		7	14	21	28
Friday	1	8	15	22	29
Saturday	2	9	16	23	30
Sunday	3	10	17	24	

2016
March
(083-283) Wednesday 23

Holi (India, Nepal)

8.00 # Component Diagram-

8.30 They are used to model physical aspects of a system like executables, libraries, files, documents, etc. which reside in a node. They are used to visualize the organisation and relationship among componants in a system.

9.00

9.30

10.00

10.30

11.00 ° Purpose -

11.30 - Visualize the component of a system

12.00 - Construct executables by using forward & reverse engineering.

12.30 - Describe the organization & relationships of components.

13.00

13.30

14.00

14.30

15.00

15.30

16.00

16.30

17.00

17.30

18.00

Evening

APR MAY JUN

Meetings	✓	Things To Do	✓	Important Calls	✓
	☐		☐		☐
	☐		☐		☐
	☐		☐		☐