Examples

Object Orientated Analysis and Design

Benjamin Kenwright

Outline

- Revision Questions
- Group Project
  - Review Deliverables
- Example System Problem
  - Case Studey

Group Project

Case-Study Example

- Vision to Requirements
- Manage Complex System
- Develop Solution
- UML Diagrams/Visualize Problem/System

Example Develop Course Registration System

Course Registration System

- What are we trying to build?
- Customer’s problem?
- Expand upon the vision statement and associated high-level requirements and constraints
- Researching/evidencing
  - Facts

Simplification

- Develop simplified perspective of the problem
- Levels to manage the complexity

Ground-Work

- Engineering Steps
- Realization of a Successful System
- System Requirements
- Elements and Sub-Systems
- Allocation and Interconnection
  - ▷Meet the System Requirements

Vision Statement

- Provide an effective and affordable course registration service for students

Functional Requirements

- Provide course registration service
- Operate the course registration service
- Maintain the course registration service

Nonfunctional Requirements

- Level of reliability to ensure adequate service guarantees
- Sufficient accuracy to support current and future user needs

Constraints

- Compatibility with other systems/standards
- Maximal use of commercial-off-the-shelf (COTS) hardware and software

Basic Mechanics

Step 1

- Get the student registration number and his date of birth.
- The registration number should be verified with the existing database of valid registration numbers.

Step 2

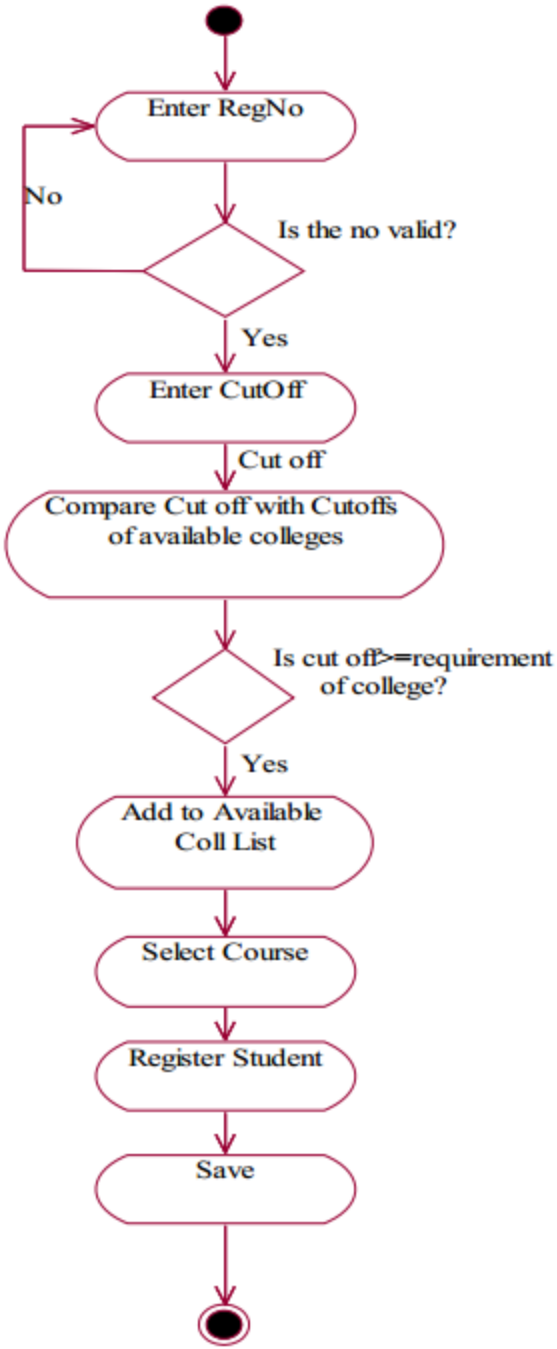
- Get the cutoff.
- Compare the cutoffs and record the cutoff along with the student's registration time.

Step 3

- Display the colleges, whose required cutoff is less than or equal to the student's total cutoff mark.
- The student must choose his preferred college.
- Display the available courses in the selected college.
- The student must now choose his preferred course.

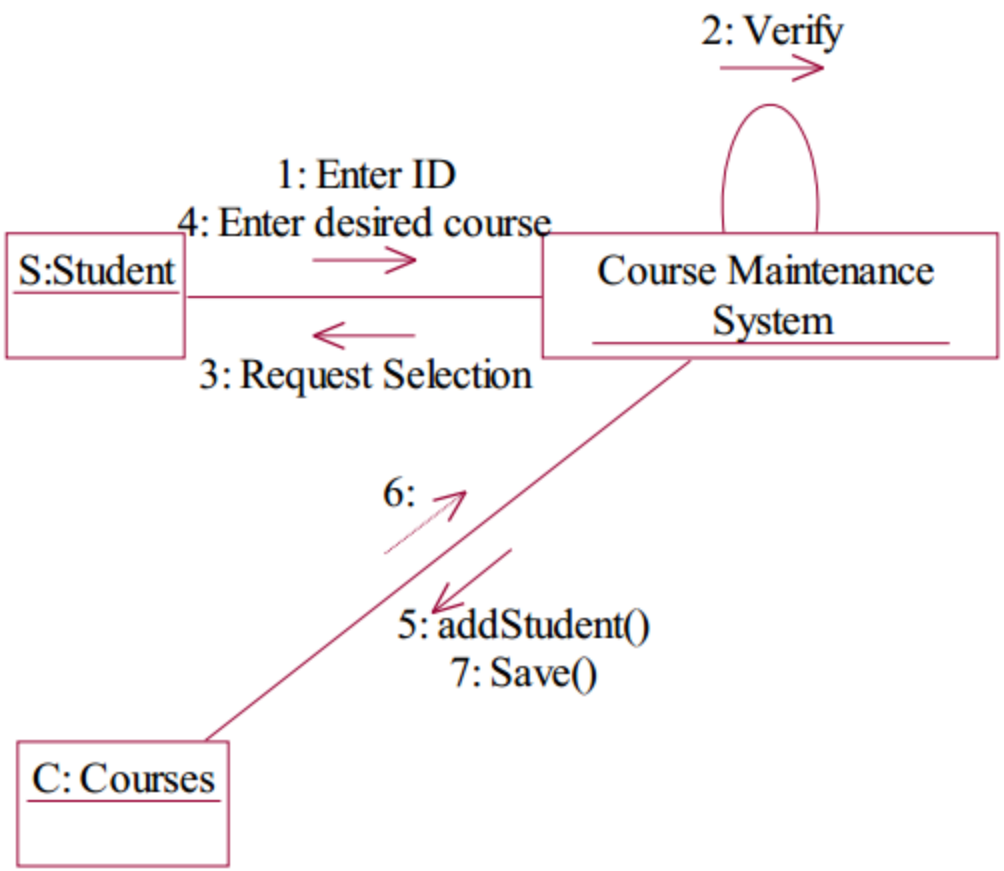
Step 4

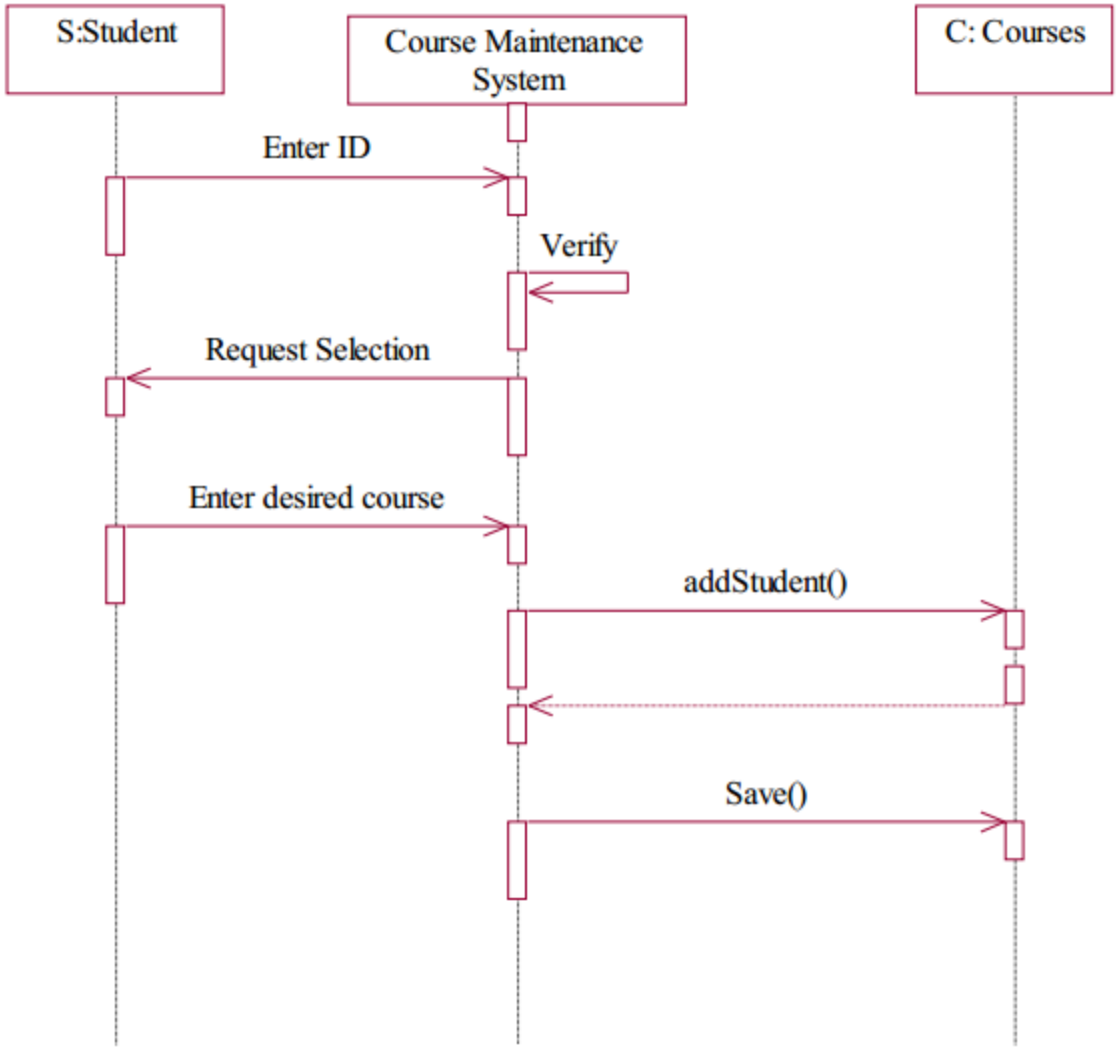
- The student must finally be informed that he has been registered successfully



Activity Diagram

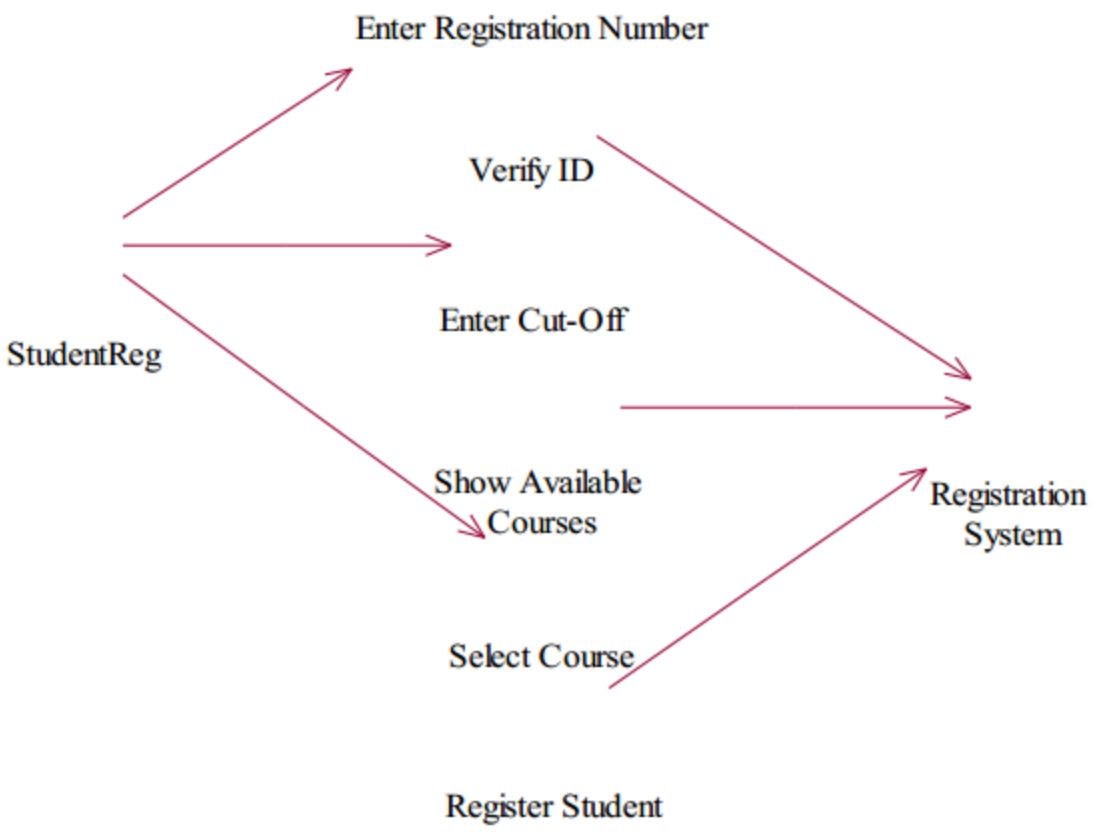
Collaboration Diagram



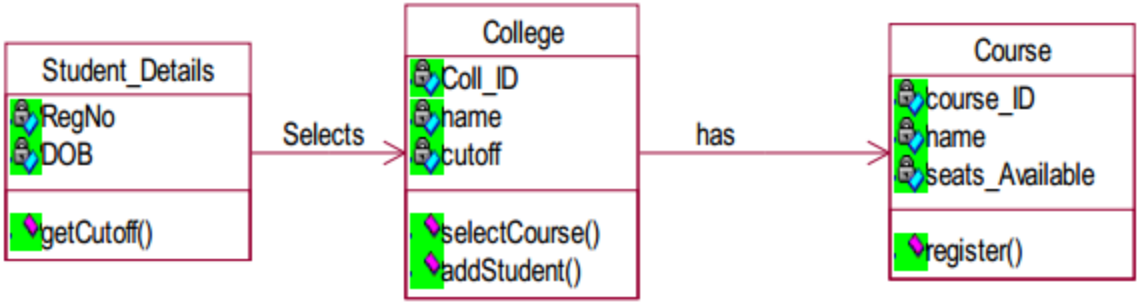


Sequence Diagram

Use Case Diagram

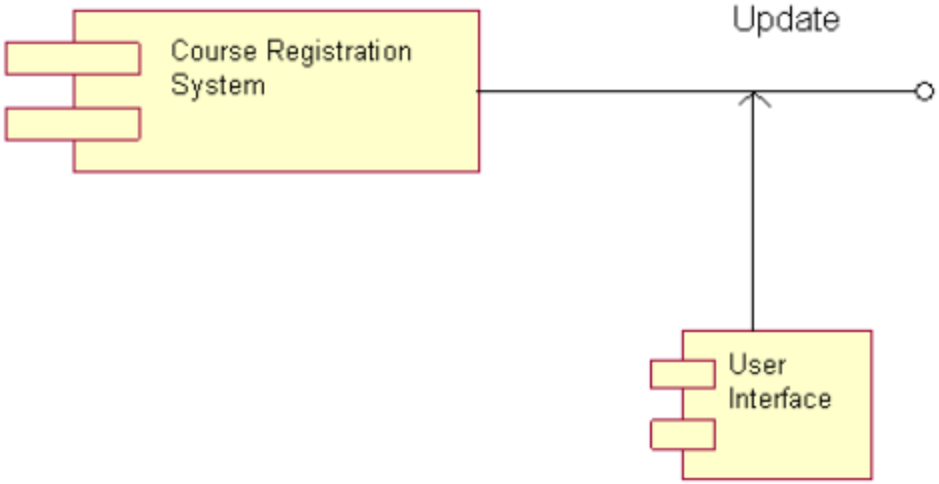


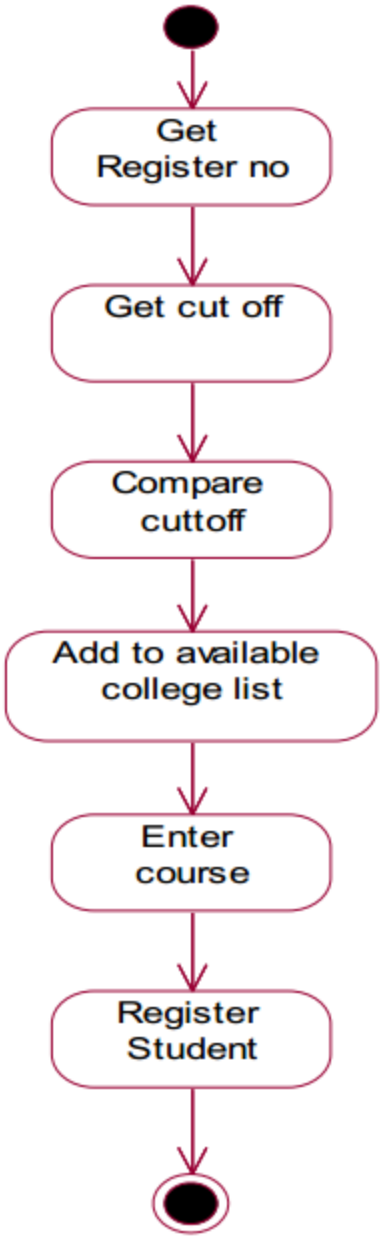
Class Diagram



Deployment Diagram

Component Diagram





State Chart Diagram

Design/Verification

- Testing/verification
- Compare against original design
- User feedback

Question

- Which of the following mechanisms is/are provided by Object Oriented Language to implement Object Oriented Model?

- A. Encapsulation
- B. Inheritance
- C. Polymorphism
- D. All of the mentioned

Answer

- D. All of the mentioned

Question

- Which of these is the *functionality* of ‘Encapsulation’?

- A. Binds together code and data
- B. Using single interface for general class of actions
- C. Reduce Complexity
- D. All of the mentioned

Answer

- A. Binds together code and data
- `Encapsulation' acts as protective wrapper that prevents code and data from being accessed by other code defined outside the wrapper

What is the output of this Program?

```
1. class Test {
2.     int a;
3.     public int b;
4.     private int c;
5. }
6. class AcessTest {
7.     public static void main(String args[])
8.     {
9.         Test ob = new Test();
10.         ob.a = 10;
11.         ob.b = 20;
12.         ob.c = 30;
13.         System.out.println(" Output :a, b, and c" + ob.a + " " + ob.b +
    " " + ob.c);
14.     }
15. }
```

- A. Compilation error
- B. Run time error
- C. Output : a, b and c 10 20 30
- D. None of the mentioned

Answer

- A. Compilation error

Explanation: Private members of a class cannot be accessed directly. In the above program, the variable c is a private member of class ‘Test’ and can only be accessed through its methods.

Question

- Object-oriented design methods cause security problems as they don’t use object-based and object-oriented programming languages

a) True
b) False

Answer

- b) False

Object-oriented design methods have evolved to help developers *exploit* the expressive power of *object-based* and object-oriented programming *languages*

Question

- What are the three important parts of Object-Oriented Programming (OOP)?

A. uses objects; each object is an instance of some class; classes may be related to one another via inheritance
B. use modules; hierarchical structure; structures must be related to one another via inheritance
C. hierarchical structure; collection of objects; objects must be related to one another via polymorphism

Answer

- A. uses objects; each object is an instance of some class; classes may be related to one another via inheritance

Question

- The three minor elements of the object model are:

a) Abstraction, Encapsulation, Inheritance

b) Modularity, Persistence, Concurrency

c) Concurrency, Persistence, Typing

d) Persistence, Abstraction, Concurrency

Answer

- c) Concurrency, Persistence, Typing

Question

- Abstraction and encapsulation are not complementary concepts

a) True
b) False

Answer

- b) False

Abstraction and encapsulation <u>are</u> complementary concepts

Question

- For abstraction to work, implementations must be encapsulated

a) True

b) False

Answer

- a) True

Question

- Why is modularity important?

a) Enables us to partitioning a program into individual components can reduce its complexity
b) Enables us to develop more optimised algorithms
c) Modularity causes issues with boundaries (or interfaces) within the program

Answer

- a) Enables us to partitioning a program into individual components can reduce its complexity

Question

- Encapsulation is avoided through information sharing, which is the process of showing all the secrets of an object that do not contribute to its essential characteristics

a) True
b) False

Answer

- a) False

Encapsulation is <u>achieved</u> through <u>information</u> hiding (not just data hiding), which is the process of <u>hiding</u> all the secrets of an object that do not contribute to its essential characteristics

Revision Question

- What is an advantage of polymorphism?

a) The same program logic can be used with objects of several related types.

b) Variables can be re-used in order to save memory.

c) Constructing new objects from old objects of a similar type saves time.

d) Polymorphism is a dangerous aspect of inheritance and should be avoided.

Answer

- a) The same program logic can be used with objects of several related types.

Question

- Classification is relative to the perspective of the observer doing the classification

a) True
b) False

Answer

- a) True

Question

- Which of the following supports the concept of hierarchical classification?

a) Polymorphism

b) Encapsulation

c) Abstraction

d) Inheritance

Answer

- d) Inheritance

Use of Hierarchical classification avoids defining the properties of object explicitly at each level which have acquired their properties from higher levels.

Revision Question

- Requirements analysis is critical to the success of a development project.

a) True

b) False

c) Depends upon the size of project

Answer

- Answer a)

Explanation: Requirements must be actionable, measurable, testable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design

Question

- Requirements should specify ‘what’ but not ‘how’.

a) True

b) False

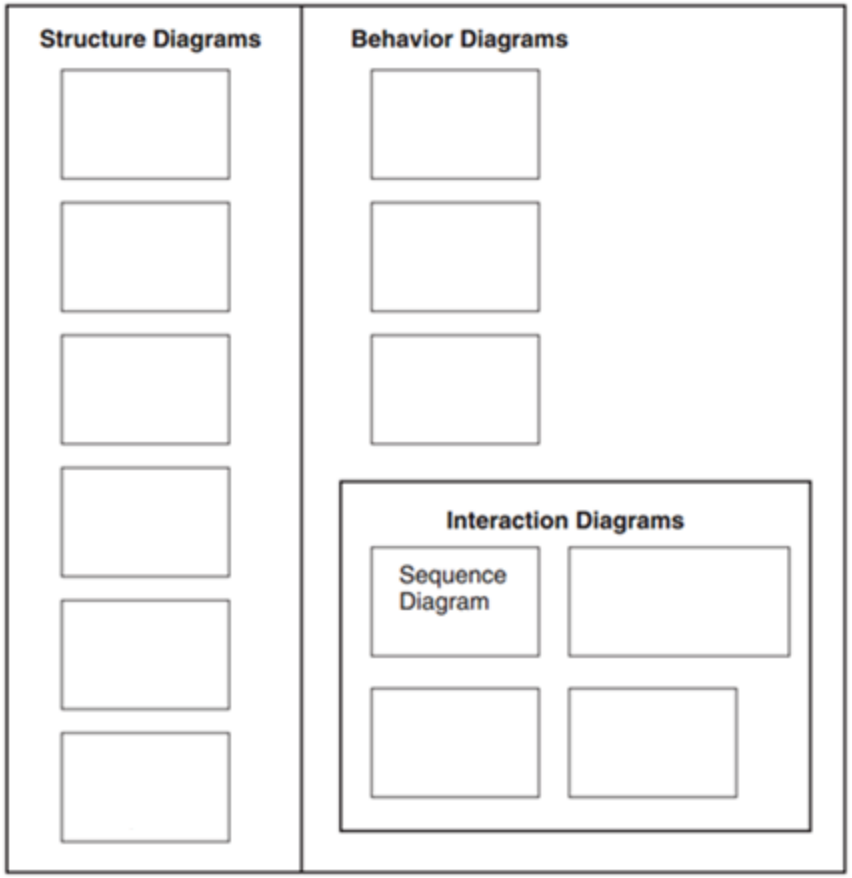
Answer

- Answer: a)

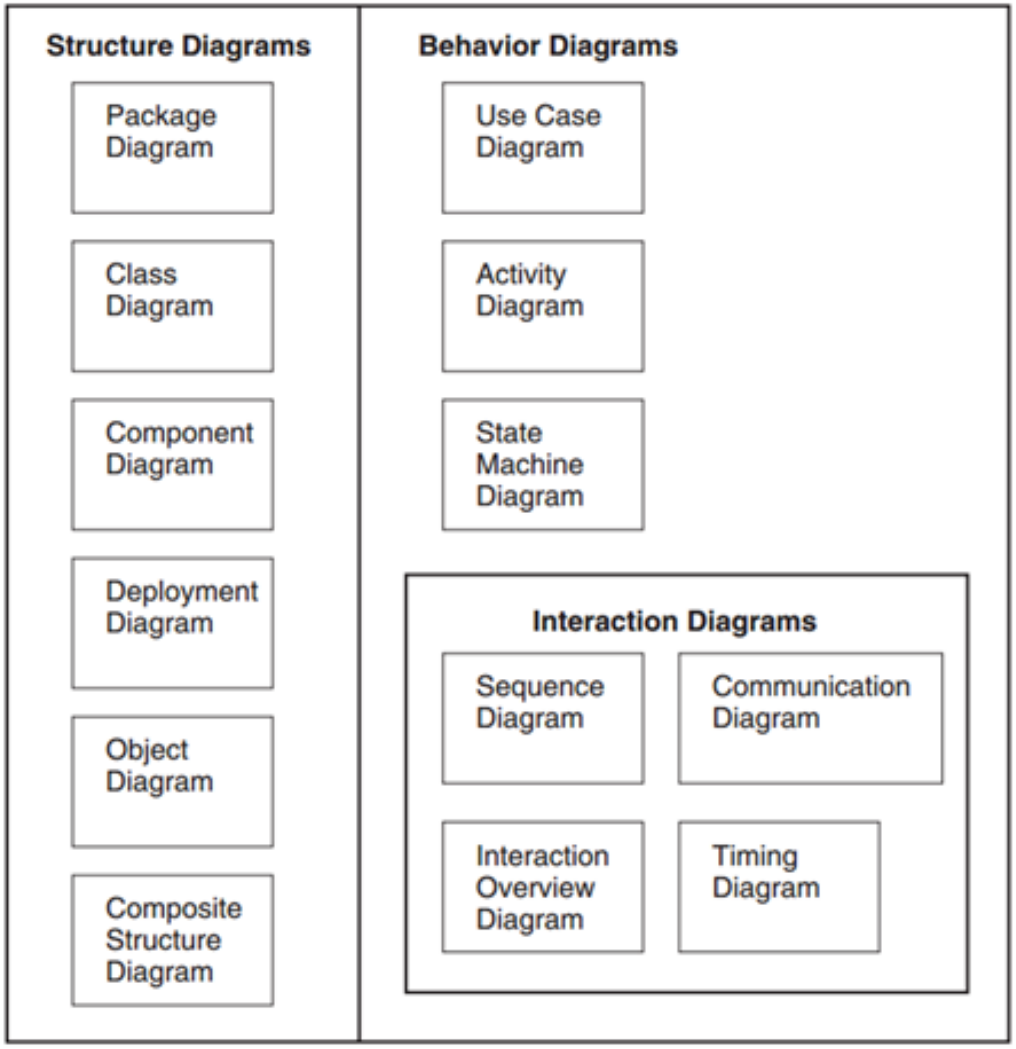
Explanation: ‘What’ refers to a system’s purpose, while ‘How’ refers to a system’s structure and behavior.

Revision Question

- List the various UML Diagram Types



Answer


Structure Diagrams
Package Diagram
Class Diagram
Component Diagram
Deployment Diagram
Object Diagram
Composite Structure Diagram
Behavior Diagrams
Use Case Diagram
Activity Diagram
State Machine Diagram
Interaction Diagrams
Sequence Diagram
Communication Diagram
Interaction Overview Diagram
Timing Diagram

Question

- Is a “Class Diagram” a static or dynamic system model view?

a) Static (or Structural)
b) Dynamic (or Behavioral**)**

Answer

a) Static (or Structural)

Question

- The three essential elements of a deployment diagram are:

a) nodes, connections and their elements

b) artifacts, nodes, and their connections

c) elements, relationships and connectors

d) inheritance, relationships and connectors

Answer

b) artifacts, nodes, and their connections

Revision Question

- Draw the notations for the different types of relationships

Dependency

Association

Direct Association

Inheritance

Realization

Aggregation

Answer

Dependency

Association

Direct Association

Inheritance

Realization

Aggregation

Question

- Briefly summarize the importance of using inheritance

Answer

- Inheritance is one of the most powerful features of object oriented programming. Most important advantages of inheritance are:
- Reusability
- Saves times and efforts
- Closeness with the real world
- Easy modification
- Transitive Nature of inheritance

Question

- What do you mean by overloading of a function? When do you use this concept? Give an example of function overloading?

Answer

- Function overloading is a technique where several function declarations are specified with a same name that can perform similar tasks, but on different data types (distinguished by their number and type of arguments)

- Example

```
int add (int a, int b);
int add (int a, int b, int c);
float add (float a, float b);
```

Hence, overloaded functions perform different activities depending upon the kind of data sent to them

Question

- List the difference between Polymorphism and Overloading?

Answer

Polymorphism

Polymorphism is an important concept of OOPS.

Polymorphism means ability of one object to take many different forms.

Two main types of polymorphism:

Runtime polymorphism

Compile time polymorphism

Overloading

Overloading is the mechanism to implement polymorphism.

Overloading is the mechanism to use the same thing for different purposes.

Question

- Which development approach is the waterfall model?

a) incremental development approach

b) iterative development approach

c) static development approach

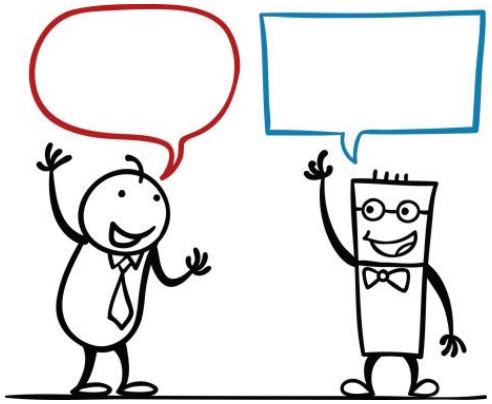
d) behavioral development approach

Answer

- a) incremental development approach

Question

- What are the four lifecycle phases for SCRUM?



Answer

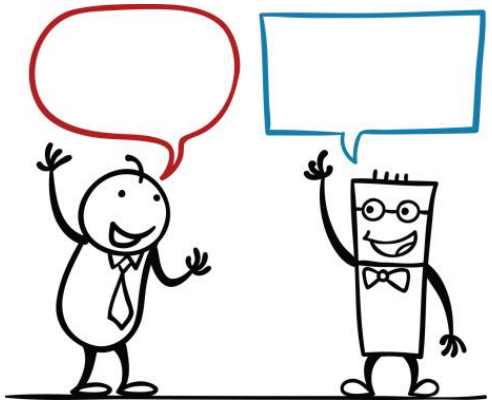
- SCRUM lifecycle includes four phases:

1. *Planning*
2. *Staging*
3. *Development*
4. *Release*

Question

- Write down the differences between Agile and Plan-Driven development

  (5 Minutes)



Answer

Agile

- Project is small
- Experienced teams with a wide range of abilities take part
- Teams are self-starters, independent leaders and others who are self-directing
- Project is an in-house project and the team co-located
- System is new with lots of unknowns
- Requirements must be discovered
- Requirements and environment are volatile with high change rates
- End-user environment is flexible
- Relationship with customer is close and collaborative
- Customer is readily available dedicated and co-located
- High trust environment exists within the development teams and customer
- Rapid value and high-responsiveness are required

Plan-Driven

- Project is large
- Teams include varied capabilities and skill sets
- Teams are geographically distributed and/or outsourced
- Project is of strategic importance
- System is well understood (scope and features set)
- Requirements are fairly stable
- System is large and complex (critical safety/high reliability requirements)
- Project stakeholders have a weak relationship with the development team
- External legal concerns
- Focus is on a strong, quantitative process improvement
- Definition and management of process are important
- Predictability and stability of process are important

Summary

- Example Problem/Solution
- Review Questions

This Week

- Review Slides
- Coursework
- Reviewing Quiz Questions
- Reviewing Associated Chapter

Questions/Discussion