


Unit-wise questions - Software Engineering

[Syllabus](#)[Notes](#)[Old Questions & solutions](#)[Yearly Questions](#)[Text & reference books](#)[Software Engineering](#)[Compiler Design and Construction](#)[E-Governance](#)[NET Centric Computing](#)[Technical Writing](#)[Applied Logic](#)[E-commerce](#)[Automation and Robotics](#)[Neural Networks](#)[Computer Hardware Design](#)[Cognitive Science](#)[Real Time System \(old-course\)](#)[Unit: 1 Introduction](#)[24 questions](#)[Unit: 2 Software Processes](#)[25 questions](#)[Unit: 3 Agile Software Development](#)[questions](#)[Unit: 4 Requirements Engineering](#)[24 questions](#)[Unit: 5 System Modeling](#)[4 questions](#)[Unit: 6 Architectural Design](#)[12 questions](#)[Unit: 7 Design and Implementation](#)[10 questions](#)

6. What are the main advantages of using an object-oriented design approach over a function-oriented approach? Explain.


 **hide solution**
asked in 2068(II)

Solution

The main advantages of using an object-oriented design approach over a function-oriented approach are

- 1. Simplicity:** Software objects model real world objects, so the complexity is reduced and the program structure is very clearly.
- 2. Code reuse and recycling:** Object created for one program can be easily reused in other program.
- 3. Faster development:** Reuse enables faster development.
- 4. Lower cost of development:** The reuse of software also lowers the cost of development.
- 5. High-quality software:** Faster development of software and lower cost of development allows more time and resource to be used in verification of the software.
- 6. Maintainable:** OOP methods make code more maintainable. Objects can be maintained separately, making locating and fixing problems easier.
- 7. Scalable:** As an object's interface provides for reusing the object in new software, it also provides all the information needed to replace the object without affecting other code. This makes it easy to replace old and aging code with faster algorithms and newer technology.
- 8. Highly secure:** Data is hidden and cannot be accessed by external functions.

7. Explain the sequence diagram with example.

 **hide solution**
asked in 2069

Solution

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

Some basic sequence diagram notation are:

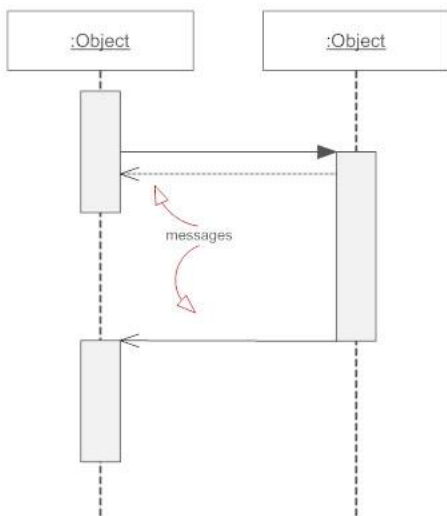
- 1. Class Roles or Participants:** Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.



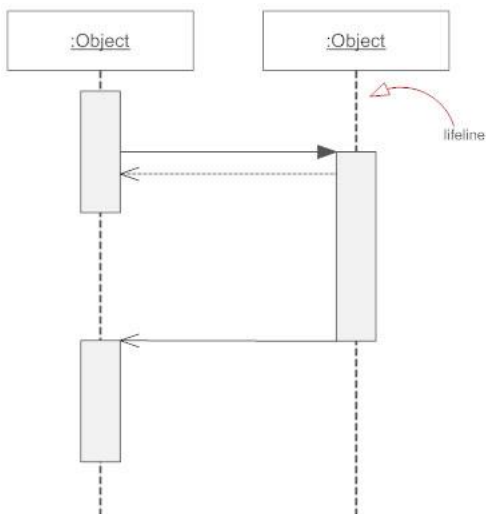
2. Activation or Execution Occurrence: Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.



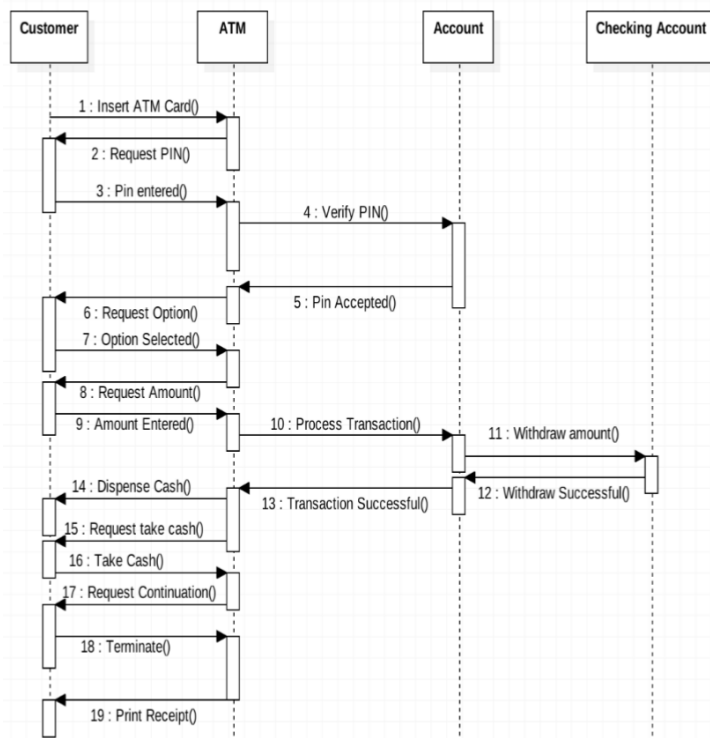
3. Messages: Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.



4. Lifelines: Lifelines are vertical dashed lines that indicate the object's presence over time.

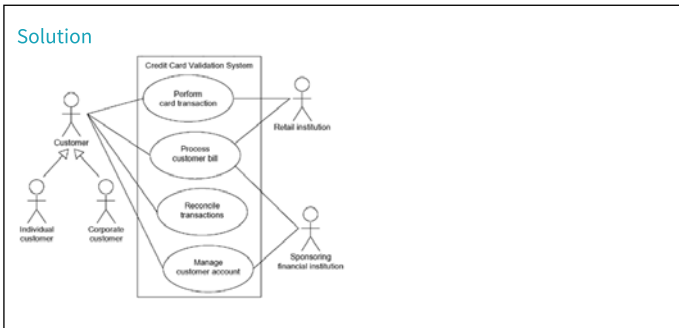


Example: Sequence diagram for ATM system



8. Draw use case diagram of credit card validation system.[Use your own assumptions].

hide solution
asked in 2074



9. Discuss sequence diagram with suitable example.

hide solution
asked in 2071

Solution

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

Some basic sequence diagram notation are:

1. Class Roles or Participants: Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.

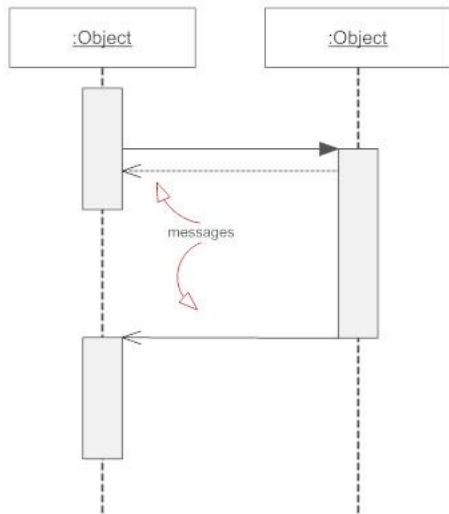


2. Activation or Execution Occurrence: Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.

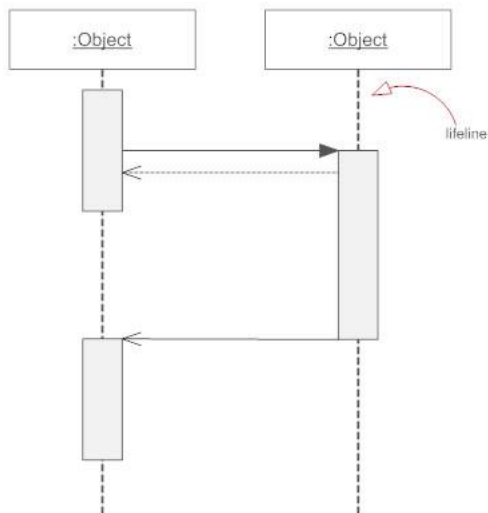


Activation or Execution Occurrence

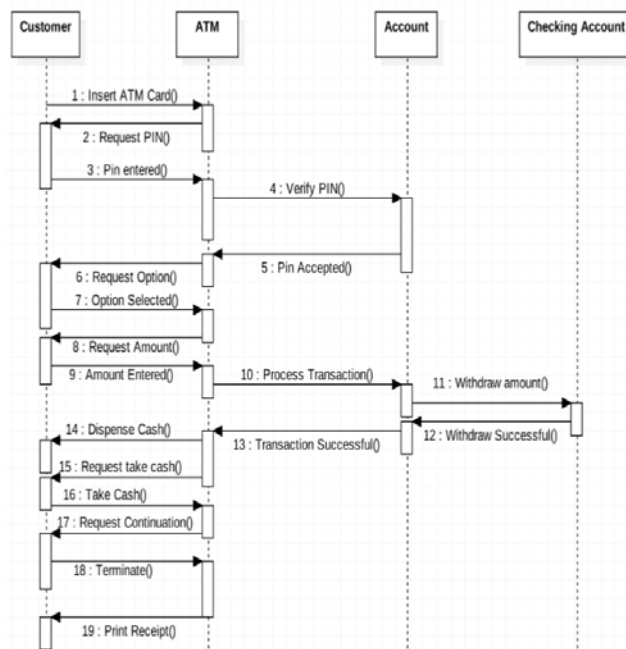
3. Messages: Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.



4. Lifelines: Lifelines are vertical dashed lines that indicate the object's presence over time.



Example: Sequence diagram for ATM system

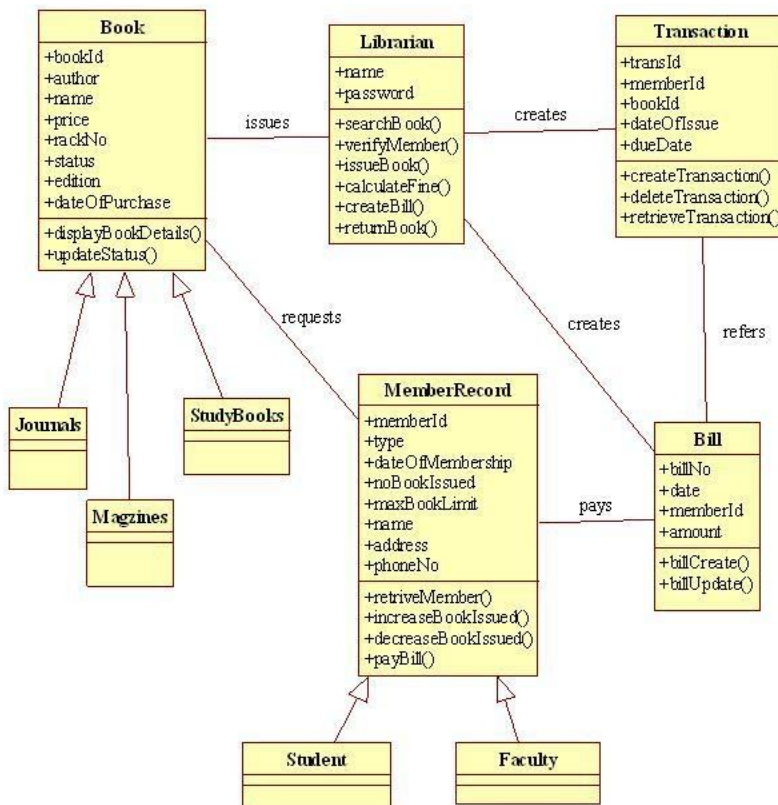


8. Draw class diagram of Library Management System [Use your own assumptions].

[hide solution](#)

asked in 2075

[Solution](#)



9. Why do we use Use-Case diagram in object-oriented development? Draw a Use-Case diagram for an online course registration system.

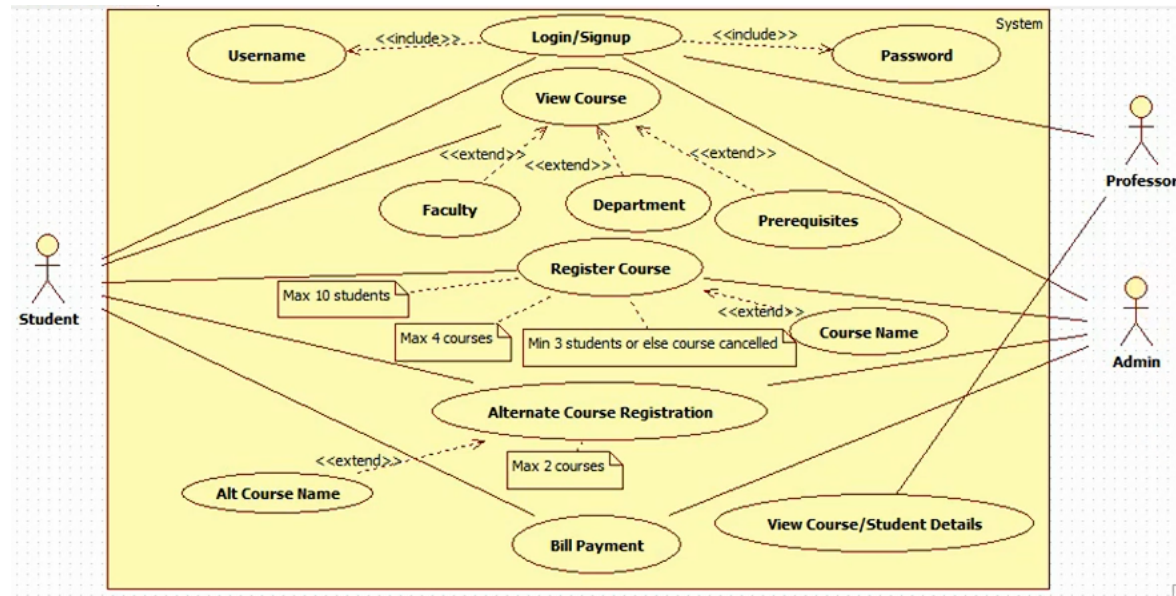
[hide solution](#)

asked in 2072

[Solution](#)

A Use Case consists of use cases, persons, or various things that are invoking the features called as actors and the elements that are responsible for implementing the use cases. The purpose of **use case diagram** is to capture the dynamic aspect of a system.

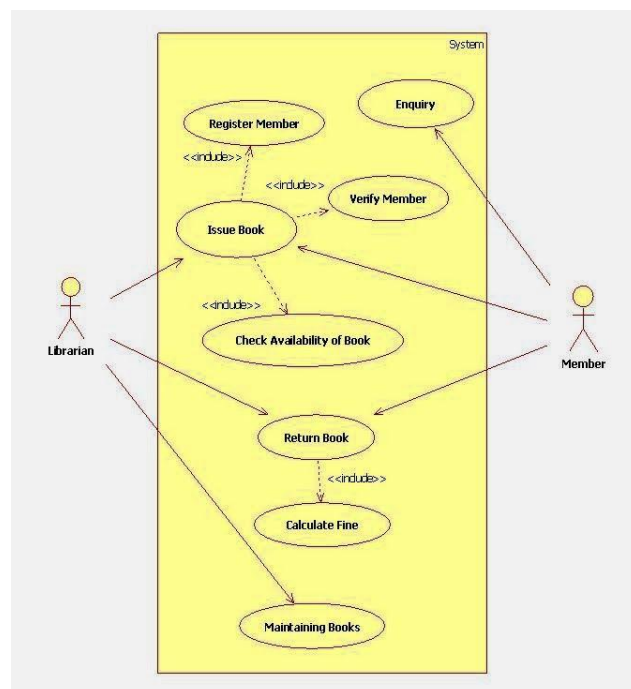
Use-Case diagram for an online course registration system:





 hide solution
asked in 2076

A Use Case consists of use cases, persons, or various things that are invoking the features called as actors and the elements that are responsible for implementing the use cases. The purpose of **use case diagram** is to capture the dynamic aspect of a system.

Use case diagram for library system



 **show solution**
asked in 2068

 **hide solution**
asked in 2068(II)

Solution

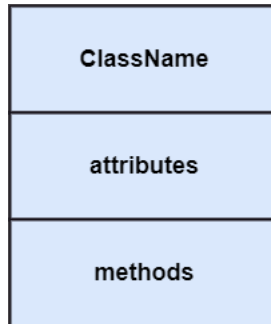
A class diagram is a **type of static structure diagram** that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

A UML class diagram is made up of:

- A set of classes and
- A set of relationships between classes

Class

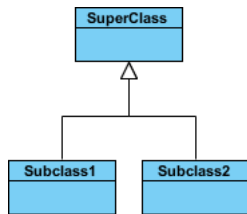
A class is a description of a group of objects all with similar roles in the system. A class notation consists of three parts:



Relationships

A class may be involved in one or more relationships with other classes. A relationship can be one of the following types:

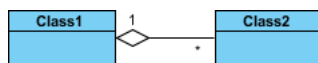
1) Generalization: A generalization is a relationship between a parent class (superclass) and a child class (subclass). In this, the child class is inherited from the parent class.



2) Association: This kind of relationship represents static relationships between two classes.

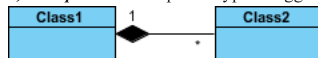


3) Aggregation: Aggregation is a special type of association that models a whole- part relationship between aggregate and its parts.



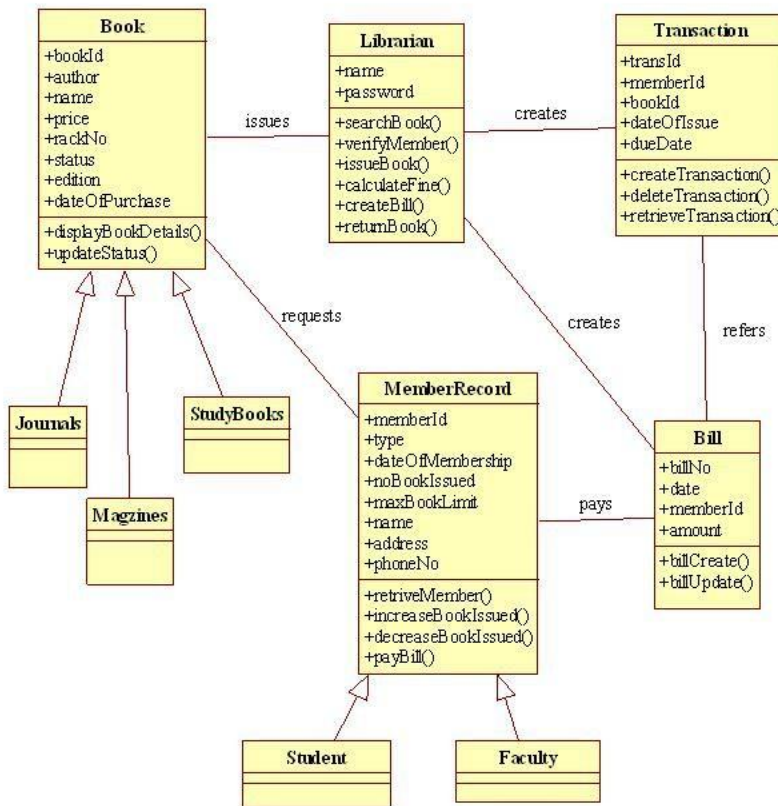
Here, Class2 is part of Class1. Many instances (denoted by the *) of Class2 can be associated with Class1.

4) Composition: A special type of aggregation where parts are destroyed when the whole is destroyed.



Here, Objects of Class2 live and die with Class1. Class2 cannot stand by itself.

Example: Class diagram for library management system



11. What is USE CASE diagram? Explain with example.

Hide solution
asked in 2069

Solution

A Use Case consists of use cases, persons, or various things that are invoking the features called as actors and the elements that are responsible for implementing the use cases. The purpose of **use case diagram** is to capture the dynamic aspect of a system.

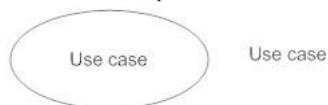
Use Case Diagram captures the system's functionality and requirements by using actors and use cases. Use Cases model the services, tasks, function that a system needs to perform. Use cases represent high-level functionalities and how a user will handle the system.

Use case diagram consists of 4 objects:

1. **System:** System is used to define scope of the use case and drawn as a rectangle.



2. **Use Case:** It represents the function or an action within the system. It is drawn as an oval and name with the function.

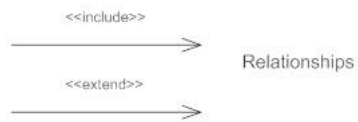


3. **Actors:** Actors are the users of a system. They interact with the system being designed in order to obtain some value from that interaction.

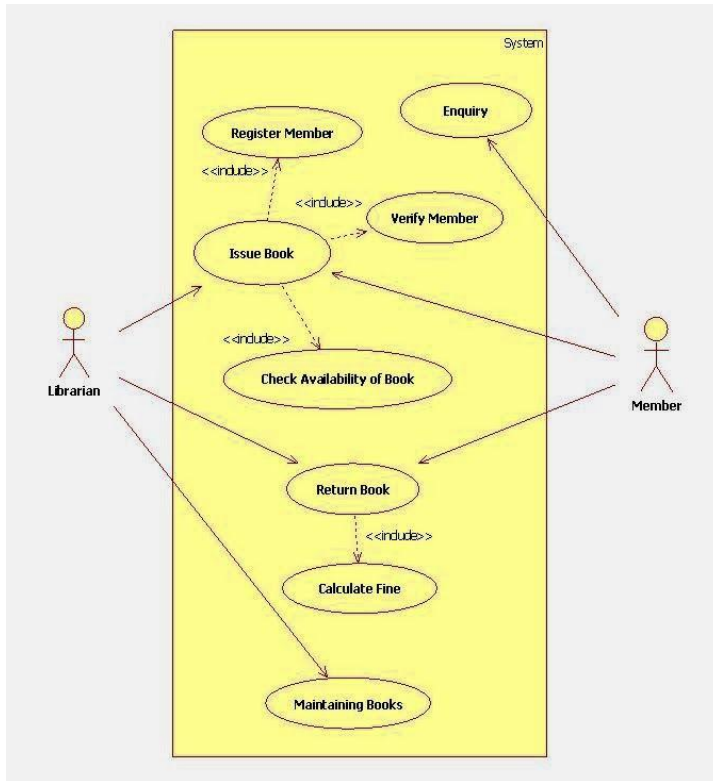


Relationships: Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.

Relationship indicates alternative options under a certain use case.



Example: Use case diagram for library system



Unit: 8 Software Testing **26 questions**

Unit: 9 Software Evolution **4 questions**

Unit: 10 Software Management **14 questions**

Suggest Us

Please give us feedback and suggestions to improve collegenote.

collegenoteofficial@gmail.com

Help

[About Us](#)

[Term & conditions](#)

[Privacy Policy](#)

Links and Resources

[CSIT](#)

[BIT](#)

[BCA](#)

[Jobs and Internships](#)

[Blog](#)

[Contact](#)

Find us @

