Department of Computer Science and Engineering

Subject Name: Agile Software Development

Subject Code:22CSE661

Question Bank

Unit 1

1. Explain Waterfall Development.
2. Explain Agile development and its characteristics and practices in detail.
3. What are the key roles in a Scrum team and what are their responsibilities? What is the purpose of splitting work into sprints or iterations in Scrum? What is a product backlog, and how is it managed in Scrum?
4. What questions are typically asked in a daily Scrum meeting? How is the number of user stories per iteration determined in Scrum planning?
5. You have recently joined a software company that follows the Scrum framework for its development process. Your team is currently building a mobile banking application. The Product Owner has prepared a list of features including user login, view balance, money transfer, and transaction history. Each feature is described in the form of user stories in the product backlog. The team follows 3-week sprints. Your Scrum Master conducts daily stand-up meetings and ensures that the Scrum principles are followed. The Scrum team has estimated that they can handle about 12 user stories per sprint based on past velocity.
   Tasks:
   a) Identify and explain three key roles in the Scrum framework and their responsibilities in this project.
   b) Describe the purpose of a sprint planning meeting and how the team should decide which user stories to include in the sprint.
   c) What is the significance of the daily Scrum (stand-up) meeting in this scenario? Provide examples of what team members might share during this meeting.
   d) If during Sprint 2, the "money transfer" feature is not working as expected, what Scrum activity can help the team reflect and improve in the next sprint? Explain briefly.
6. How Scrum fall under Agile Umbrella Model? Explain in detail.
7. What is Extreme Programming (XP) and illustrate the planning process and key practices of XP in the areas of management, team, and programming?
8. Illustrate the project led to development of Extreme Programming (XP) and list some advantages and disadvantages of using XP in software development?
9. What are the five main steps in the FDD process model?
10. How does FDD ensure customer involvement and maintain code quality? What is a 'feature' in FDD and what are its essential components?
11. What are the seven principles of Lean software development and explain each.
12. What types of waste are targeted for elimination in Lean thinking? Why is late decision-making considered beneficial in Lean development?
13. A software development team is applying Lean Software Development principles to improve their workflow and reduce waste. The table below shows the current breakdown of activities in their delivery cycle:

Activity Type	Time Spent (in days)
Actual development work	8
Waiting time	6
Context switching & delays	3
Rework	3
Total Lead Time	20

The team applies lean principles like limiting work-in-progress, shortening feedback loops, and removing delays. As a result, the non-value-adding time is reduced by 40%.
Tasks:

a) Calculate the total non-value-adding time before applying Lean practices.
b) Determine the number of days reduced from non-value-adding time.
c) Compute the new total lead time after applying Lean practices.
d) What is the percentage improvement in lead time?*)*
e) Mention any two Lean principles that help achieve this improvement.

14. What is the purpose of Test-Driven Development (TDD)? How do Acceptance TDD and Developer TDD differ in Agile development?
15. What are user stories and what role do they play in Agile development?
16. What are the key benefits of using user stories over traditional requirements documents?
17. What does the acronym INVEST stand for in writing good user stories?
18. What are the six lifecycle stages of a user story?
19. How are epics, user stories, and tasks related in Agile backlog management?

Unit 2

1. List the principles of Agile. Describe the Applicability and problems of agile methods.
2. Explain Extreme Programming(XP) practices and Illustrate XP Release Cycle process.
3. What is a user story and how is it structured? What are the three C's of a user story as described by Ron Jeffries?
4. List and explain the tests in User Story in Agile.
5. A product team is designing a Grocery Delivery App that should support user login, product search, adding items to a cart, real-time delivery tracking, and sending alerts when groceries are delivered or delayed.
   Task: Write four well-defined user stories in Agile format that reflect the features expected in the grocery delivery app. Then, explain how these stories contribute to creating a user-centric product backlog.
6. Describe Effort Estimation and the purpose of planning poker in Agile estimation.
7. A startup is building a fitness-tracking mobile application. During the release planning meeting, the product owner outlines several high-priority features: user registration, activity logging, integration with wearable devices, push notifications for workout reminders, and viewing past fitness stats. The development team has assigned story points based on complexity and effort. The current team velocity is 18 story points per sprint, based on the last three sprints.
   Tasks:
   a) If the total estimated effort for these features is 37 story points, how many sprints would the team need to deliver them? Show your calculation.
   b) Suppose the team adopts pair programming and automated testing, which increases their velocity to 24. How does this affect the number of sprints required?
   c) Mention two challenges a team might face when estimating user stories accurately in the early stages of a project.
8. What are the three types of risks identified in Agile projects?
9. How is risk level for user stories assessed?
10. Contrast between Waterfall & Agile.
11. What roles do the customer and developers play in release planning? What are the three phases of the planning game?
12. How does Scrum differ from Kanban in terms of roles and iteration planning? What is the purpose of a burn-down chart in Scrum?
13. What are the roles of the driver and navigator in pair programming?
14. Compare the three pair programming styles: Driver-Navigator, Strong Style, and Ping-Pong.
15. What are some anti-patterns in pair programming?
16. Illustrate and Explain 5 stages of TDD lifecycle.
17. Describe the advantages, disadvantages & Best practices of TDD.
18. What is the difference between stubs, fakes, and mocks? When should a developer prefer using mocks?
19. List TDD antipatterns.
20. List the types of Testing and explain them in detail.

21. Contrast between black-box and white-box testing.
22. Define CI and Describe a typical CI workflow and how Merge Conflicts occur.
23. What is the difference between Continuous Integration, Continuous Delivery, and Continuous Deployment?
24. List the advantages, Disadvantages & Best practices of CI.

Unit 3

1. What is the main design approach followed in Agile methods to maintain clean code and List indicators of bad software design and explain why they are problematic.
2. What are the five principles that contribute to good software design under the SOLID acronym?
3. Why is it important for code to remain flexible in Agile development?
4. What defines a "responsibility" according to the Single Responsibility Principle? In the Phone class example, what are the two responsibilities that violate SRP? How does introducing separate interfaces for connection and data transmission help uphold SRP?
5. Explain Open-Closed Principle (OCP) with Violation, Compliant, Compliant Design and its advantages.
6. A software company is developing a notification system that sends alerts to users via email. Initially, the NotificationSender class is implemented as follows:

```
public class NotificationSender {
    public void sendEmail(String message) {
        // Logic to send email
        System.out.println("Sending Email: " + message);
    }
}
```

Later, the client requests support for SMS notifications as well. The developer adds an sendSMS() method to the same class:

```
public class NotificationSender {
    public void sendEmail(String message) {
        System.out.println("Sending Email: " + message);
    }
    public void sendSMS(String message) {
        System.out.println("Sending SMS: " + message);
    }
}
```

The development team realizes this approach violates a key design principle.
Tasks:
a) Identify the design principle being violated and explain why this is a problem.
b) Describe how this scenario can be refactored to follow the Open-Closed Principle.
c) Redesign the class structure using abstraction or interfaces, showing how new notification types (e.g., Push Notification) can be added without modifying the existing code.
d) Briefly explain how applying the Open-Closed Principle supports maintainability in Agile development.

7. Explain Liskov Substitution Principle (LSP). What conditions must a subclass meet to satisfy the Liskov Substitution Principle?
8. A software development team is designing a Shape library for a drawing application. The base class Shape has the following specification:

Method	Preconditions	Postconditions
setWidth(w)	w > 0	width == w and height == old.height
setHeight(h)	h > 0	height == h and width == old.width

A new class Square is introduced that inherits from Rectangle. It overrides setWidth() and setHeight() as follows:

- setWidth(w): sets both width and height to w
- setHeight(h): sets both height and width to h

Tasks:

a) Based on the table, identify whether Square satisfies the postconditions defined by Rectangle. Justify your answer.
b) If a client method assumes Rectangle's postconditions and calls r.setWidth(5); r.setHeight(4);, what would be the result when Square is substituted for Rectangle?
c) Calculate the area the client expects and the actual area returned when Square is used instead of Rectangle.
d) Based on your findings, explain how violating LSP can introduce bugs in object-oriented systems.

9. A development team is working on a Smart Building Automation System. They have created a common interface called Device with the following methods:

```
public interface Device {
   void turnOn();
   void turnOff();
   void setTemperature(int degrees);
   void playMusic(String track);
}
```

This interface is implemented by different devices such as SmartBulb, SmartThermostat, and SmartSpeaker.
During development, the team notices that:

- SmartBulb only uses turnOn() and turnOff().
- SmartThermostat uses turnOn(), turnOff(), and setTemperature().
- SmartSpeaker uses turnOn(), turnOff(), and playMusic().

Tasks:
a) Identify the design flaw in the current Device interface with respect to the Interface Segregation Principle.
b) Suggest how the interface can be redesigned to comply with ISP. Provide names for new interfaces.
c) Redraw or describe a revised class structure where each device implements only the methods it actually needs.
d) Briefly explain how applying ISP benefits system scalability and maintainability in Agile environments.

10. In a library management system, the LibraryManager class is responsible for issuing and returning books. Initially, it directly creates and uses instances of classes like DatabaseLogger and FileLogger to log transactions. Later, the team realizes that every time a new type of logger is added (e.g., CloudLogger), the LibraryManager class must be modified, which violates a core design principle.
Tasks:
a) Identify the violated design principle and explain the issue in terms of dependency structure.
b) Define the Dependency Inversion Principle (DIP) and explain its two core rules.
c) Suggest how the LibraryManager can be refactored to follow DIP using abstraction.
d) Describe how applying DIP improves flexibility and reduces maintenance in the above scenario.
e) Illustrate the new class structure using an interface or abstract class approach.

11. What is the main design approach followed in Agile methods to maintain clean code and List indicators of bad software design and explain why they are problematic.
12. What are the five principles that contribute to good software design under the SOLID acronym?
13. Why is it important for code to remain flexible in Agile development?
14. What defines a "responsibility" according to the Single Responsibility Principle? In the Phone class example, what are the two responsibilities that violate SRP? How does introducing separate interfaces for connection and data transmission help uphold SRP?
15. Explain Open-Closed Principle (OCP) with Violation, Compliant, Compliant Design and its advantages.
16. A software company is developing a notification system that sends alerts to users via email. Initially, the NotificationSender class is implemented as follows:

```
public class NotificationSender {
    public void sendEmail(String message) {
        // Logic to send email
        System.out.println("Sending Email: " + message);
    }
}
```

Later, the client requests support for SMS notifications as well. The developer adds an sendSMS() method to the same class:

```
public class NotificationSender {
    public void sendEmail(String message) {
        System.out.println("Sending Email: " + message);
    }
    public void sendSMS(String message) {
        System.out.println("Sending SMS: " + message);
    }
}
```

The development team realizes this approach violates a key design principle.
Tasks:
a) Identify the design principle being violated and explain why this is a problem.
b) Describe how this scenario can be refactored to follow the Open-Closed Principle.
c) Redesign the class structure using abstraction or interfaces, showing how new notification types (e.g., Push Notification) can be added without modifying the existing code.
d) Briefly explain how applying the Open-Closed Principle supports maintainability in Agile development.

17. Explain Liskov Substitution Principle (LSP). What conditions must a subclass meet to satisfy the Liskov Substitution Principle?
18. A software development team is designing a Shape library for a drawing application. The base class Shape has the following specification:

Method	Preconditions	Postconditions
setWidth(w)	w > 0	width == w and height == old.height
setHeight(h)	h > 0	height == h and width == old.width

A new class Square is introduced that inherits from Rectangle. It overrides setWidth() and setHeight() as follows:

- setWidth(w): sets both width and height to w
- setHeight(h): sets both height and width to h

Tasks:

a) Based on the table, identify whether Square satisfies the postconditions defined by Rectangle. Justify your answer.
b) If a client method assumes Rectangle's postconditions and calls r.setWidth(5); r.setHeight(4);, what would be the result when Square is substituted for Rectangle?
c) Calculate the area the client expects and the actual area returned when Square is used instead of Rectangle.
d) Based on your findings, explain how violating LSP can introduce bugs in object-oriented systems.

19. A development team is working on a Smart Building Automation System. They have created a common interface called Device with the following methods:

```
public interface Device {
   void turnOn();
   void turnOff();
   void setTemperature(int degrees);
   void playMusic(String track);
}
```

This interface is implemented by different devices such as SmartBulb, SmartThermostat, and SmartSpeaker.
During development, the team notices that:

- SmartBulb only uses turnOn() and turnOff().
- SmartThermostat uses turnOn(), turnOff(), and setTemperature().
- SmartSpeaker uses turnOn(), turnOff(), and playMusic().

Tasks:
a) Identify the design flaw in the current Device interface with respect to the Interface Segregation Principle.
b) Suggest how the interface can be redesigned to comply with ISP. Provide names for new interfaces.
c) Redraw or describe a revised class structure where each device implements only the methods it actually needs.
d) Briefly explain how applying ISP benefits system scalability and maintainability in Agile environments.

20. In a library management system, the LibraryManager class is responsible for issuing and returning books. Initially, it directly creates and uses instances of classes like DatabaseLogger and FileLogger to log transactions. Later, the team realizes that every time a new type of logger is added (e.g., CloudLogger), the LibraryManager class must be modified, which violates a core design principle.
Tasks:
a) Identify the violated design principle and explain the issue in terms of dependency structure.
b) Define the Dependency Inversion Principle (DIP) and explain its two core rules.
c) Suggest how the LibraryManager can be refactored to follow DIP using abstraction.
d) Describe how applying DIP improves flexibility and reduces maintenance in the above scenario.
e) Illustrate the new class structure using an interface or abstract class approach.

21. List the three major categories of design patterns and give example for each.
22. Differentiate between creational and structural design patterns.
23. Describe Common behavioural pattern with simple explanation
24. How can UML class diagrams be used to represent the structure of design patterns? Draw a UML class diagram for the Factory Method pattern.
25. What is the purpose of the Factory Method pattern, and how does it support object creation in Agile design? In which scenario would using a Factory Method be preferable over a direct constructor call?
26. Explain how the Singleton pattern works and give one real-world use case in software systems. Why is Singleton often used for shared resources like logging and configuration?
27. Describe the Prototype and Adapter.

28. Illustrate Bridge solution to the modern problem and How does the Bridge pattern promote abstraction and implementation independence? In what situation would the Bridge pattern be better than simple inheritance?
29. Describe how the Composite pattern supports hierarchical structures. Mention an example.
30. How does the Decorator pattern, Facade pattern, Flyweight pattern, Proxy pattern add responsibilities and simplify complex subsystems and optimize memory usage.
31. Provide a code sketch showing the Command pattern for an Undo/Redo functionality in a text editor. And Explain how the Interpreter pattern is used to build a basic mathematical expression evaluator.
32. Show how the Iterator pattern can be applied to traverse a collection of books in a library system and Illustrate the Mediator pattern to manage interactions between UI components in a chat application.
33. Create a simple example showing how the Observer pattern updates multiple displays when weather data changes.
34. Write a conceptual model where the State pattern is used to manage the states of an Order (New, Packed, Shipped, Delivered). Show how the Strategy pattern can allow a navigation app to switch between different route calculation algorithms (fastest, shortest, scenic).
35. Explain why Refactoring is important. List and briefly explain two techniques: Extract Method and Extract Class in code refactoring.
36. Describe the cycle of Continuous Integration (CI) and explain its significance in Agile teams.
37. Explain Key agile architecture and Design consideration.

Unit 4

1. Explain why Agile teams prioritize "Individuals and Interactions over Processes and Tools." How does this philosophy influence communication methods?
2. Your Agile team has shifted to remote work. How would you ensure adaptive and flexible communication among team members to handle changing project needs?
3. During a sprint, misunderstandings arose because important updates were shared informally and not documented. How would you redesign the communication approach to balance informal chats with transparency?
4. List and briefly explain any three principles of Agile communication that promote faster project alignment.
5. A Scrum team holds a Sprint Planning meeting for a 3-week sprint. If the recommended duration is 2 hours per week of sprint, how long should the Sprint Planning meeting last? Justify how that time can be divided during the planning.
6. Describe the purpose and structure of the Sprint Review ceremony in Scrum.
7. During daily stand-ups, one developer starts solving technical problems instead of giving updates. What guidelines or best practices can the Scrum Master reinforce to keep stand-ups effective?
8. Two team members disagree strongly about which design pattern to use. Which conflict resolution techniques could you apply to ensure constructive discussions while preserving team relationships?
9. Imagine you are a new Scrum Master leading a self-organizing team. What actions would you take to encourage experimentation and build psychological safety?
10. Define Servant Leadership. How is it different from traditional command-and-control leadership models?
11. List three challenges faced by distributed Agile teams and suggest a practical solution for each.
12. A distributed team has members in 3 different time zones with only 2 overlapping hours daily. If each sprint requires 15 hours of joint collaboration, how many days are needed to complete the joint work?
13. Name two visual collaboration tools and explain how they assist Agile teams during sprint planning and retrospectives.
14. Your Agile team has constant interruptions during focused work periods because of unstructured chats. How can implementing Working Agreements solve this problem?

15. What strategies can Agile teams use to prevent communication overload and reduce information fatigue during remote work?
16. After a demo, a Product Owner misinterprets a key feature because they weren't actively listening during discussions. How would you improve feedback loops and listening practices?
17. An Agile team of 6 people has 7 effective working hours per day. Planning for a 2-week sprint (10 working days) and applying a focus factor of 85%, what is the sprint capacity in hours?
18. During sprint reviews, stakeholders often raise new feature requests rather than giving feedback on completed work. How should the Scrum Master handle this situation to maintain sprint review focus?
19. Explain two ways cultural differences can impact communication in globally distributed Agile teams. How can teams adapt to mitigate these issues?
20. Based on the Spotify case study, how did video calls and blocker-focused daily stand-ups improve remote team engagement? How can you apply similar practices in a startup environment?

Unit 5

1. Explain the three core characteristics of Agile Project Management: Iterative & Incremental, Collaborative, and Adaptive.
2. Your client requests major requirement changes in the middle of a sprint. As an Agile Project Manager, how should you handle this situation without disrupting the sprint goals?
3. Describe the three Scrum roles and list the main Scrum ceremonies and artifacts.
4. How does Kanban manage workflow differently compared to Scrum? Mention the importance of WIP (Work-in-Progress) limits.
5. List and briefly explain the four levels of SAFe: Portfolio, Large Solution, Program, and Team.
6. Your organization has multiple Agile teams working independently but facing integration challenges. Which scaling framework would you recommend and why?
7. Identify and explain two major challenges in scaling Agile across large organizations.
8. Compare SAFe, LeSS, Nexus, and Spotify models for scaling Agile. Highlight one strength for each.
9. A team completed 150 story points in 5 sprints. Calculate the average team velocity per sprint.
10. If the average cycle time of a user story is 6 days, how many stories can be completed in a 30-day project window?
11. Explain how Agile governance incorporates Quality Assurance and Risk Management into the development lifecycle.
12. If a key stakeholder continuously changes requirements during the sprint, how should the Agile team and Product Owner manage stakeholder expectations?
13. List three major shifts that Agile encourages compared to traditional command-and-control project management.
14. Your organization is starting an Agile transformation. What three steps would you prioritize during the Pilot Implementation phase?
15. Explain the purpose of Sprint Planning and Sprint Retrospective in Scrum.
16. Your product backlog has grown very large and unorganized. What steps would you take to refine and manage the backlog effectively?
17. What are the key steps to implementing an effective Kanban system? Name at least three practices.
18. During a PI Planning session, two teams identify conflicting dependencies. As a Release Train Engineer (RTE), how would you help them resolve it?
19. Compare Planning Poker and T-Shirt Sizing as Agile estimation techniques. In which situations would you prefer one over the other?
20. In a 2-week sprint, a burn-down chart shows that only 40% of tasks were completed by the end of week 1. How much work (in percentage) should ideally remain at the end of week 1 in a perfectly balanced sprint?