SA卷子预测+自测

历史

选择

1. **A system should be able to continue working when Internet attacks arrive. Which quality attribute is this? ———— Security**

   给出一个简单的例子，问符合哪个QA

2. **What are the advantage of pipe & filters? ———— D**
   A. Filters communicate with each other by transferring data.
   B. It's easy to reuse filters and add new filters.
   C. Interactive applications are encouraged by the style
   D. a,b
   E. b,c 给出一个style，选哪些是属于它的优缺点

判断

1. Performance can be measured with throughput, latency or deadline.

✅ **答案：正确 (True)**
解释：
这句话是对的，性能（Performance）是一个质量属性，在课件中明确指出其常见度量方式包括：

- **Throughput（吞吐量）**：单位时间内系统可以处理多少请求或事务。
- **Latency（延迟）**：从输入到输出所需的时间。
- **Deadline（截止时间）**：系统能否在指定的时间内完成任务。
  这些指标共同用于评估系统响应能力与处理效率。

---

2. Many systems adopt graphical user interface to meet scalability requirements.

❌ **答案：错误 (False)**
解释：
图形用户界面（GUI）是为了提升用户体验（usability）而设计的，**与系统可扩展性（scalability）无直接关系。**

- Scalability 是指系统在面对更大负载或更多用户时的扩展能力，常通过分布式架构、负载均衡、缓存等技术实现。
- GUI 不影响系统后端的可扩展能力，因此这道题错误。

---

3. A software system must be modelled with exact "4+1" views.

❌ **答案：错误 (False)**
解释：
"4+1视图模型"是一种推荐的建模方法，帮助架构师从多个角度描述系统，包括：

1. Logical View
2. Development View（或 Implementation View）
3. Process View
4. Physical View
5. +1 是 **Use Case View（用例视图）**

虽然很常用，但它**不是强制的**，根据实际项目需求，也可以采用其它方法建模，所以不是“must be”。

---

4. A software architecture can use more than one architectural patterns.

✅ **答案：正确 (True)**
解释：
现实中的软件系统往往结合使用多种架构风格（architectural styles），例如：

- 使用 **Layered Architecture** 进行模块分层；
- 同时使用 **Client-Server** 处理前后端通信；
- 使用 **Repository** 管理数据共享。

这些组合让系统更加灵活和强大，因此**一个架构使用多个架构模式是完全合理且常见的**。

---

5. If a system has low coupling and high cohesion, the modifiability of the system is good.

✅ **答案：正确 (True)**
解释：
Modifiability（可修改性）是质量属性之一。

- **Low coupling（低耦合）**：模块之间依赖性低，更易于独立修改。
- **High cohesion（高内聚）**：每个模块内部功能集中，逻辑清晰。

这两个特性有助于快速、低成本地修改系统而不引发连锁反应，因此**能显著提升系统的可维护性和演化能力**。

---

6. 4+1 view model consists of logical view, development view, process view, implementation view and physical view.

答案：错误 (False)
解释：
“4+1视图模型”的正确组成是：

- Logical View（逻辑视图）
- Development View（开发视图）
- Process View（进程视图）
- Physical View（物理视图）
- **Use Case View（用例视图）** ← 这才是 +1

题目中将 **Implementation View** 作为额外视图列出，其实它只是 Development View 的别称，且**遗漏了关键的 Use Case View**，因此说法错误。

简答

1. **What is the definition of Software Architecture? Why need we do architecture design?**
   Answer:
   The software architecture of a program or computing system is the structure or structures of the system, which comprise **software elements**, the externally **visible properties** of those elements, and the **relationships** among them.
   An architecture include: component, connecter and constraints and it is the result of a set of busiess and technical decisions
   Why: 选3条即可

原文	中文翻译
Architecture is the **vehicle for stakeholder communication.**	架构是利益相关者沟通的媒介。
Architecture **manifests the earliest set of design decisions**.	架构体现了最早期的一组设计决策。
The Architecture **Defines Constraints on Implementation**	架构定义了实现的约束。
The Architecture Dictates **Organizational Structure**	架构决定了组织结构。
The Architecture Inhibits or Enables a System’s Quality Attributes	架构决定系统质量特性是否可实现。
The Architecture Makes It Easier to **Reason about and Manage Change**	架构使变更管理和推理更容易。
The Architectur**e Helps in Evolutionary Prototyping**	架构支持原型迭代。
The Architecture **Enables More Accurate Cost and Schedule Estimates**	架构帮助更准确地估算成本与进度。

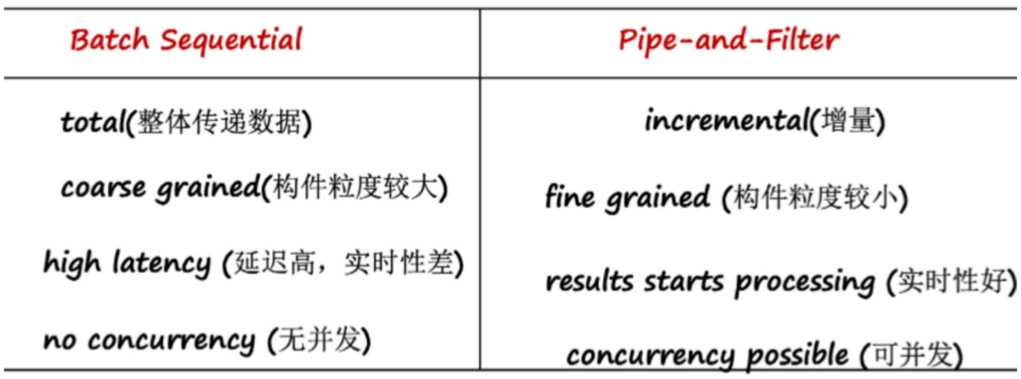
2. **Explain the meaning of Usability and briefly provide at least THREE solutions to improve Usability.**
   Answer: Usability means to reduce the difficulty for users to use the software.
   Solutions:
   Runtime Tactics
   - System Anticipates User Tasks
   - System Provides Appropriate Feedback to Users
   - System Provides Consistent Experience to Users

   Design-time Tactics
   - Support Undo Operations
   - Isolate the User Interface from Other Parts of the System

3. What is monolithic architecture? What is microservice architecture? Compare them and write their advantages and disadvantages. (没有讲过，但是课件中强调过Batch和PF的对比，这里答案是Batch和PF的答案)
   Answer:
   Batch: Computation is performed in separate steps; the output of one step is used as input for the next.

   Advantages: - Clear modular separation of processing steps. - Easy to audit and debug using intermediate files.
   Disadvantages: - No support for interactive or real-time processing. - High latency between steps.

   FP: A pipe-and-filter architecture consists of components called filters and connectors called pipes. Each filter processes its input data and generates output that is passed via pipes to subsequent filters.
   Advantages:
   High cohesion
   Low coupling.
   Reusability
   Simple to implement Extendibility and flexibility
   DisAdvantages:
   Not suitable for interactive applications.
   High overhead from parsing/format conversion.
   Low efficiency when large amounts of shared data are needed.
   Campare:



Batch Sequential	Pipe-and-Filter
total(整体传递数据)	incremental(增量)
coarse grained(构件粒度较大)	fine grained (构件粒度较小)
high latency (延迟高，实时性差)	results starts processing (实时性好)
no concurrency (无并发)	concurrency possible (可并发)

4. What are the benefits to produce software by software product line?
   A software product line is a set of software-intensive systems that share a common, managed feature set satisfying a particular market segment's specific needs or mission and that are developed from a common set of core assets in a prescribed way.

应用

Nowadays there are many online stores such as JingDong, Taobao, Amazon and so on. They bring a customer great convenience to do shopping. The following is simplified requirements for an online store.
Requirements:
 A customer can browse different goods.
 A customer can search the online store by keywords.
 A customer can filter goods by price, manufacturer, categories and so on.
 A customer can add goods to shopping cart. The user can view and modify the shopping list.

 A customer can pay a bill by a credit card, ali-pay, wechat-pay, or paypal. A customer should feel confident that their personal information will be protected and the payment is secure.
 The store provides special offers like order discounts, coupon codes, and free shipping. The offers can be combined. The special offers can encourage customers to take advantage of a "good deal".
Tasks:
i. (5 points) what **architectural style/pattern** can be used to design the system? Tell the reason for you to select the architecture style.
ii. (10 points) draw the architecture diagram for the system. The architecture diagram should **embody modules and the relationships between modules.**
iii. (10 points) Based on the requirements, what **quality attribute** should be considered? Tell the solutions to meet the quality attribute.

i. Architectural Style / Pattern (5 points)

Recommended Style:
Layered Architecture + Client-Server Style

Reasons:

- **Layered Style** allows clear separation of concerns between UI, business logic, and data management. This improves **modifiability**, **usability**, and **reusability**.
- **Client-Server Style** is appropriate for online shopping platforms, where the **client (browser/app)** communicates with a **centralized server** to access and manage services (e.g., browsing, payment, order).
- Combining both patterns helps structure the system for **scalability** and **security**, as required in the prompt.

---

ii. Architecture Diagram (10 points) （这个图有点不知道画什么）

```
text



CopyEdit
+----------------------+      +----------------------------+
|   Web/Mobile Client  | <--> |  Web Controller / REST API|
+----------------------+      +------------+---------------+
                                           |
                                           v
                              +-----------+-----------+
                              |     Business Logic     |
                              |  (Cart, Order, Search) |
                              +-----------+-----------+
                                           |
                                           v
                              +-----------+-----------+
                              |    Special Offer Mgr   |
                              |   Payment Processing   |
                              | Security / Auth Module |
                              +-----------+-----------+
                         |
                         v
              +-----------+-----------+
              |      Data Access       |
              |  (Products, Users, etc)|
              +-----------+-----------+
                         |
                         v
                 +------------------+
                 |     Database      |
                 +------------------+
```

Relationships:

- Top-down invocation between layers
- API exposed to clients (RESTful or GraphQL)
- Modules such as `SpecialOfferMgr` and `SecurityModule` encapsulate cross-cutting concerns

---

iii. Quality Attributes and Solutions (10 points)

Quality Attribute	Rationale (from Requirements)	Solution / Tactic
Security	Protect personal/payment info	- Authentication & Authorization (OAuth) - Secure Communication (HTTPS/SSL) - Input validation & firewall
Usability	Browsing, filtering, UI feedback	- UI Layer separation - Provide responsive design - Offer clear feedback (e.g., shopping cart status)
Performance	Efficient browsing & payment	- Caching frequently viewed items - Asynchronous processing for payment - Load balancing
Modifiability	Add new payment methods or offers	- Encapsulate features into independent modules (e.g., Strategy Pattern for promotions) - Follow layering to isolate changes
Availability	Ensure the store is always up	- Use replication for databases - Employ failover servers and monitoring
Testability	Ensuring quality across modules	- Separate concerns into layers (UI, business, data) - Use dependency injection to replace real components with mocks - Apply logging and monitoring mechanisms

一、选择题

Q1. What best describes the purpose of a software architecture?

A. To write code faster
B. To define how requirements are implemented line by line
C. To **define the structure of the system and guide high-level decisions**
D. To replace testing altogether
✅ **Answer: C**
📌 **考点**：软件架构的定义与作用（Lesson 12）

Q2. What is the main goal of the "Layered Architecture" style?

A. To allow every module to access each other freely
B. To support dynamic service registration
C. **To organize system responsibilities in hierarchical levels**
D. To speed up front-end development only
✅ **Answer: C**
📌 **考点**：架构风格：分层架构（Lesson 21）

Q3. Which architecture style emphasizes components communicating via asynchronous messages through a central medium?

A. Client-server
B. Pipe-and-filter
C. **Event-based architecture**
D. Layered
✅ **Answer: C**
📌 **考点**：事件驱动架构（Lesson 31）

Q4. What is the key benefit of using architectural patterns?

A. They guarantee code correctness
B. They eliminate the need for documentation
C. **They promote reuse and accelerate development**
D. They reduce runtime memory usage
✅ **Answer: C**
📌 **考点**：架构模式的优势（Lesson 12、51）

Q5. In the “4+1” view model, which view describes how the system behaves during execution?

A. Logical view
B. **Process view**
C. Development view
D. Physical view
✅ **Answer: B**

📌 **考点**：“4+1”架构视图模型（Lesson 71）

Q6. What does "separation of concerns" mean in software architecture?

A. Avoiding using design patterns
B. Splitting a system into independent modules with distinct responsibilities
C. Making modules duplicate each other's logic
D. Using a single module for all functionalities
✅ **Answer: B**
📌 **考点**：软件架构原则之一：关注点分离（Lesson 12）

Q7. Which of the following UML diagrams is used to represent concurrency and workflows?

A. Class diagram
B. Activity diagram
C. Use case diagram
D. Deployment diagram
✅ **Answer: B**
📌 **考点**：UML 图与架构建模用途（Lesson 71）

Q8. Which is NOT a tactic to improve testability?

A. Internal monitoring
B. Record/replay
C. Adding redundancy
D. Providing specific test paths
✅ **Answer: C**
📌 **考点**：可测试性架构策略（Lesson 61）

Q9. What is a key tactic to improve system usability at runtime?

A. Provide feedback (e.g., progress bars)
B. Eliminate undo functionality
C. Merge interface with implementation
D. Remove all user interaction logs
✅ **Answer: A**
📌 **考点**：可用性运行时策略（Lesson 86）

Q10. What is the function of the "Agenda" in a rule-based system like Drools?

A. To monitor memory usage
B. To resolve hardware conflicts
C. To manage the execution order of activated rules
D. To display user messages
✅ **Answer: C**
📌 **考点**：规则系统构件结构——Agenda（Lesson Drools 特讲）

Q11. What is a major disadvantage of using an interpreter-style architecture?

A. It's limited to only GUI applications
B. It prevents data sharing
C. It is significantly slower than compiled or hardware-based execution
D. It requires distributed systems
✅ **Answer: C**
📌 **考点**：解释器风格的劣势（Lesson 41）

Q12. Which of the following is a common use for deployment diagrams in UML?

A. Describe user interactions
B. Represent class hierarchies
C. Show how software components are mapped to hardware nodes
D. Model inheritance relationships
✅ **Answer: C**
📌 **考点**：UML 中部署图的作用（Lesson 71）

Q13. What does the "Stimulus" element in a quality attribute scenario refer to?

A. The system requirement
B. The user interface
C. The event that triggers a response from the system
D. The performance measurement tool
✅ **Answer: C**
📌 **考点**：质量属性场景六要素之"刺激"（Lesson 51）

Q14. What is the main focus of maintainability in quality attributes?

A. Reducing power consumption
B. Supporting changes and updates with low effort
C. Preventing user interaction
D. Increasing CPU usage
✅ **Answer: B**
📌 **考点**：可维护性定义与策略（Lesson 51）

Q15. Which of the following best defines the goal of software architecture documentation?

A. To replace system code
B. To serve as an internal debugging tool only
C. To communicate architecture decisions and support stakeholder needs
D. To generate runtime logs
✅ **Answer: C**
📌 **考点**：架构描述文档的作用（Lesson 71）

Q16. Which of the following is a key reason to use architecture evaluation early in the software lifecycle?

A. It improves CSS styling
B. It guarantees GUI success
C. It identifies risks before high implementation cost
D. It replaces runtime testing
✅ **Answer: C**
📌 **考点**：架构评估目的与重要性（Lesson 87）

Q17. In the ATAM method, what is the purpose of "scenarios"?

A. To write code templates
B. To list available frameworks
C. To describe possible situations to test quality attributes
D. To simulate database transactions
✅ **Answer: C**
📌 **考点**：ATAM 方法中的场景使用（Lesson 87）

Q18. What is the role of the Blackboard in the Blackboard architecture style?

A. It stores class diagrams
B. It handles low-level threading
C. It is a global data store accessed by independent knowledge sources
D. It monitors event logs
✅ **Answer: C**
📌 **考点**：黑板架构风格（Lesson 21）

Q19. What does the Rule Engine in a rule-based system mainly do?

A. Compiles source code
B. Manages memory allocation
C. Matches facts and rules and triggers actions accordingly
D. Logs user input
✅ **Answer: C**
📌 **考点**：Drools 规则引擎功能（Lesson Drools）

Q20. Which of the following is a valid tactic for improving performance?

A. Increase component coupling
B. Use interpreted languages
C. Control resource demand
D. Add more GUIs
✅ **Answer: C**
📌 **考点**：性能相关策略（Lesson 61）

Q21. Which of the following is **not** a typical element in an architecture quality scenario?

A. Source
B. Stimulus
C. Module name
D. Environment
✅ **Answer: C**
📌 **考点**：质量属性场景六要素（Lesson 51）

Q22. What is a key characteristic of Pipe-and-Filter architecture?

A. Shared global memory
B. Synchronous request-response
C. Sequential data flow between independent filters
D. Direct pointer manipulation
✅ **Answer: C**
📌 **考点**：Pipe-and-Filter 架构（Lesson 21）

Q23. Which of the following is NOT a benefit of using architectural styles?

A. Supports reuse
B. Enables standardization
C. Guarantees system correctness
D. Facilitates communication
✅ **Answer: C**
📌 **考点**：架构风格的优点（Lesson 12）

Q24. In a Client-Server architecture, the server is mainly responsible for:

A. Displaying all UI elements
B. Making low-level memory calls
C. Providing centralized services and managing resources
D. Running on mobile devices
✅ **Answer: C**

📌 **考点**：Client-Server 模式（Lesson 21）

Q25. What is the focus of modifiability as a quality attribute?

A. User satisfaction
B. Database partitioning
C. Ease of making changes after deployment
D. Adding security algorithms
✅ **Answer: C**
📌 **考点**：Modifiability 的定义与策略（Lesson 51）

Q26. In the context of software architecture, what is a "sensitivity point"?

A. A UML class attribute
B. A runtime error
C. A decision that has a large impact on quality attributes
D. A GUI input element
✅ **Answer: C**
📌 **考点**：ATAM 分析中敏感点定义（Lesson 87）

Q27. Which of the following statements about UML is TRUE?

A. UML is a programming language
B. UML diagrams cannot represent concurrency
C. UML provides a set of modeling diagrams to describe systems from multiple views
D. UML is only used in database modeling
✅ **Answer: C**
📌 **考点**：UML 建模用途（Lesson 71）

Q28. What does “loose coupling” promote in software architecture?

A. Faster memory allocation
B. Tight integration between components
C. Independence between components
D. Merging modules into one file
✅ **Answer: C**
📌 **考点**：松耦合原则（Lesson 12）

Q29. Which of the following is a typical application area of the Rule-Based style?

A. Image rendering
B. Compiler optimization
C. Expert systems and business rule processing
D. Network routing protocols
✅ **Answer: C**

📌 **考点**：规则驱动架构应用（Lesson Drools）

Q30. What is the role of “working memory” in a rule engine like Drools?

A. To execute Java code directly
B. To track the number of database queries
C. To hold facts and intermediate results for matching rules
D. To store log messages
✅ **Answer: C**
📌 **考点**：Drools 工作内存定义（Lesson Drools）

二、判断题

No.	题目	正确答案	考点说明
1	Software architecture focuses only on implementation-level code.	❌ False	软件架构不只关注实现细节，还包括系统级结构与设计决策（Lesson 12）
2	The 'Pipe and Filter' architecture allows for concurrent execution of filters.	✅ True	管道-过滤器架构支持并发处理（Lesson 21）
3	In client-server architecture, the client can directly modify server code.	❌ False	客户端不可直接修改服务器代码（Lesson 21）
4	In event-based architecture, components are tightly coupled to ensure synchronization.	❌ False	事件驱动架构的组件是松耦合的（Lesson 31）
5	In the call-and-return style, subroutines can call other subroutines and return control to the caller.	✅ True	调用-返回风格特点（Lesson 41）
6	An interpreter-style architecture executes source code directly without converting it to machine code.	✅ True	解释器风格直接执行源代码（Lesson 41）
7	Agenda in Drools is used to determine the firing order of rules.	✅ True	Drools中Agenda的作用（Drools课件）
8	Rule-based systems cannot handle forward chaining inference.	❌ False	基于规则的系统可以处理前向推理（Drools课件）
9	Quality attribute scenarios contain exactly five elements.	❌ False	应为六个元素：stimulus, source, environment, artifact, response, and response measure（Lesson 51）
10	Reliability can be enhanced by adding redundancy and recovery mechanisms.	✅ True	提高可靠性的方法之一（Lesson 51）

11	Testability tactics include limiting nondeterminism and making internal states observable.	✅ True	可测试性策略（Lesson 61）
12	Modifiability refers to the ease with which a system can be adapted to changes.	✅ True	可修改性定义（Lesson 51）
13	Performance can only be improved by reducing memory usage.	❌ False	性能也可以通过资源管理、并发等方式优化（Lesson 51）
14	Maintainability and modifiability are unrelated concepts in architecture.	❌ False	二者密切相关（Lesson 51）
15	The '4+1' view model includes physical, logical, process, development and use case views.	✅ True	4+1视图模型（Lesson 71）
16	The logical view in '4+1' model describes deployment of software onto hardware.	❌ False	物理视图才描述部署信息（Lesson 71）
17	Activity diagrams are mainly used to describe object relationships.	❌ False	活动图用于描述流程和并发（Lesson 71）
18	Sequence diagrams represent interactions over time between components.	✅ True	时序图用途（Lesson 71）
19	Usability is only concerned with graphical interface aesthetics.	❌ False	可用性还包括响应、帮助、可理解性等（Lesson 86）
20	Runtime tactics for usability include user feedback and system state visibility.	✅ True	可用性运行时策略（Lesson 86）
21	Evaluation methods such as ATAM are used to assess architectural decisions.	✅ True	架构评估方法ATAM（Lesson 87）
22	SAAM focuses on quality attribute scenarios and trade-off analysis.	✅ True	SAAM方法的主要内容（Lesson 87）
23	Layered architecture does not support hierarchical abstraction.	❌ False	分层架构支持分级抽象（Lesson 21）
24	In the blackboard architecture, all components read from and write to a common knowledge base.	✅ True	黑板架构定义（Lesson 31）
25	Component-and-connector views show system structure in terms of execution units and communication paths.	✅ True	C&C视图定义（Lesson 71）
26	Modularity decreases the understandability of the system.	❌ False	模块化提升系统可理解性（Lesson 12）

27	A rule engine like Drools does not support condition-action logic.	❌ False	Drools支持基于规则的条件-动作逻辑（Drools课件）
28	Scalability refers to the ability of a system to handle growth in load without compromising performance.	✅ True	可扩展性定义（Lesson 51）
29	A quality attribute like availability can be measured by Mean Time To Failure (MTTF).	✅ True	可用性常用MTTF衡量（Lesson 51）
30	Architecture styles and design patterns are the same concepts.	❌ False	架构风格与设计模式不同（Lesson 12）

三、简答题

1. What is the definition of software architecture?

Answer: Software architecture is the fundamental organization of a system, embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution. *(Lesson 12)*

2. What are the advantages of using architectural styles?

Answer: Architectural styles promote reuse, simplify communication among stakeholders, support analysis and design reasoning, and improve development efficiency. *(Lesson 12)*

3. Explain the main characteristics of the Layered Architecture style.

Answer: Responsibilities are organized in hierarchical layers; each layer only interacts with the adjacent layers; promotes modifiability and separation of concerns. *(Lesson 21)*

4. Describe the main components and interactions in the Pipe-and-Filter architecture.

Answer: Components (filters) process data and pass results via connectors (pipes); supports concurrent processing and transformable data streams. *(Lesson 21)*

5. What is the key benefit of using the Blackboard architecture style?

Answer: It allows multiple components to collaboratively solve a problem by sharing a common knowledge base, useful in AI and heuristic search. *(Lesson 31)*

6. What distinguishes Event-based Architecture from other styles?

Answer: Loose coupling between components, asynchronous message communication, and a central event bus or publish-subscribe system. *(Lesson 31)*

7. What is the purpose of an interpreter in software architecture?

Answer: To execute programs in a source language via step-by-step interpretation, often used when the target machine or language is unavailable. *(Lesson 41)*

8. List and explain the six parts of a quality attribute scenario.

Answer: Stimulus, Source, Environment, Artifact, Response, and Response Measure. *(Lesson 51)*

9. What tactics can improve system availability?

Answer: Introduce redundancy, enable fault detection, fast recovery mechanisms, and failover systems. *(Lesson 51)*

10. What are the goals of testability tactics?

Answer: To make system behavior predictable and observable, reduce nondeterminism, and control internal states. *(Lesson 61)*

11. Explain the role of source code annotations in improving modifiability.

Answer: Annotations localize assumptions, highlight variation points, and help isolate modules for targeted change. *(Lesson 61)*

12. Describe two runtime tactics that enhance usability.

Answer: Provide system feedback and make system state visible to users. *(Lesson 86)*

13. What is the role of the Agenda in Drools rule-based systems?

Answer: Agenda manages the rule execution order based on conflict resolution and rule priorities. *(Drools)*

14. What is the difference between forward and backward chaining in rule engines?

Answer: Forward chaining starts from facts to infer conclusions; backward chaining starts from goals to determine necessary conditions. *(Drools)*

---

15. What are the five views in the 4+1 architectural view model?

Answer: Logical view, Process view, Development view, Physical view, and Use-case view. *(Lesson 71)*

---

16. How does ATAM support architectural evaluation?

Answer: ATAM identifies quality attributes, scenarios, architectural approaches, risks, non-risks, and trade-offs through stakeholder participation. *(Lesson 87)*

---

17. What does a component-and-connector view represent?

Answer: It shows the system structure in terms of components (units of computation) and connectors (communication mechanisms). *(Lesson 71)*

---

18. What is the significance of scenario-based evaluation methods like SAAM?

Answer: They assess how well architecture supports future changes by analyzing real or hypothetical usage scenarios. *(Lesson 87)*

---

19. Why is it important to document architecture decisions?

Answer: Documentation improves communication, enables traceability, supports maintenance, and provides rationale for decisions. *(Lesson 71)*

---

20. Explain the concept of domain-specific reference architecture.

Answer: It captures common structures, patterns, and constraints for systems in a particular domain to promote reuse and consistency. *(Lesson 12)*

四、应用题

Application Question 1: Smart Home Control System

Background:
You are designing a smart home controller that supports temperature sensors, remote switches, device collaboration, and scheduling. Future plans include integration with third-party assistants like Alexa.

Questions & Answers:

1. **What architectural style is most appropriate for this system and why?**
   Answer: Event-Based Architecture, because it enables loose coupling, asynchronous communication, and flexible integration of heterogeneous devices.
2. **List three critical quality attributes and how to support them.**
   Answer:
   - **Scalability**: Use a publish-subscribe messaging bus.
   - **Modifiability**: Isolate device logic in separate modules.
   - **Availability**: Deploy components with redundancy and failure detection.
3. **How would you design a plugin-based module communication mechanism?**
   Answer: Use service discovery and registration patterns, such as a plugin registry and dynamically loaded interfaces.
4. **If using event-driven architecture, describe the basic event flow.**
   Answer: Event source → Event bus → Event listeners (handlers) → Response/actions triggered asynchronously.

Application Question 2: University Library Expansion

Background:
An existing system for student borrowing now needs to support faculty and external users. Rules vary across user types, and future versions should support multiple campuses.

Questions & Answers:

1. **What architectural style was likely used and what are its scalability concerns?**
   Answer: Likely a Layered Architecture; the concern is tight coupling between business logic and data access layers, which hinders scaling for new rules.
2. **Propose an architecture refactoring strategy to increase variability support.**
   Answer: Introduce a Rule-Based System (e.g., Drools) to encapsulate variable borrowing policies outside the main logic.
3. **Identify two reusable modules and explain how to share them in a product line.**
   Answer:
   - **User Management**: Shared across campuses.
   - **Notification Engine**: Can be reused for overdue alerts and reservation confirmations.
4. **Design a SAAM scenario to evaluate changeability.**
   Answer:
   Scenario: “Add new borrowing policies for research scholars with unique limits and periods.” Evaluate affected modules and required changes.

Application Question 3: Medical Appointment System

Background:
The system must support patients, doctors, administrators, appointment scheduling, payments, and reports while ensuring security and high availability.

Questions & Answers:

1. **Which architectural styles would you use and why?**
   Answer: Layered + Service-Oriented Architecture (SOA). Layers manage separation of concerns; SOA supports interoperability across roles.
2. **Propose one tactic for availability and one for consistency.**
   Answer:
   - **Availability**: Failover mechanism with hot standby replicas.
   - **Consistency**: Use transactional boundaries with ACID guarantees for data-sensitive modules.
3. **How would you apply the ATAM method to evaluate this system?**
   Answer:
   - Identify stakeholders (doctors, admins), scenarios (e.g., emergency access), architectural approaches (replication, encryption), risks (downtime), and trade-offs (latency vs. security).
4. **How to support offline access to patient data for doctors?**
   Answer:
   Design a local cache with controlled synchronization and encryption.

Application Question 4: Online Video Platform

Background:
You are building a YouTube-like system that allows users to upload, stream, and comment on videos, and receive personalized recommendations.

Questions & Answers:

1. **Propose a dataflow architecture for core modules.**
   Answer:
   Upload → Storage → Transcoding → CDN → Playback; each module is connected via a pipe passing transformed video data.
2. **Describe the upload and playback process using C&C view.**
   Answer:
   - **Components**: UploadManager, StorageService, PlayerModule
   - **Connectors**: REST API, file I/O, streaming protocol
   - Player connects to CDN and retrieves the processed file.
3. **Design two performance-related tactics.**
   Answer:
   - Use asynchronous pre-fetching for fast startup.

    - Apply caching and edge servers for high throughput.

4. **How would you adapt the architecture for live streaming?**
   Answer:
   Integrate a real-time ingestion pipeline with low-latency protocols (e.g., RTMP/WebRTC) and live transcoding services.

Application Question 5: Promotion Rule Engine in E-Commerce

Background:
An e-commerce platform requires a flexible promotion rule engine supporting discount types like percentage off, bundles, tiered pricing, and member-specific offers.

Questions & Answers:

1. **Which architectural style is best for the rule system and why?**
   Answer: Rule-Based Architecture using a Production Rule System like Drools. It separates logic from application code and supports complex reasoning.
2. **Explain the role of facts, rules, working memory, and agenda in Drools.**
   Answer:
    - **Facts**: Business data (cart, user, time).
    - **Rules**: Logic to apply promotions.
    - **Working memory**: Stores current facts.
    - **Agenda**: Manages rule execution order.
3. **How to support modularity and versioning of rules?**
   Answer:
   Organize rules by category (discounts, gifts, etc.) and apply version tags with a rule repository system for management.
4. **Describe the execution of forward chaining in this context.**
   Answer:
   Facts inserted → Rules matched → Matching rules fired → New facts asserted (e.g., “discount applied”) → Repeat until no more matches.
