体系结构往年考题整理：

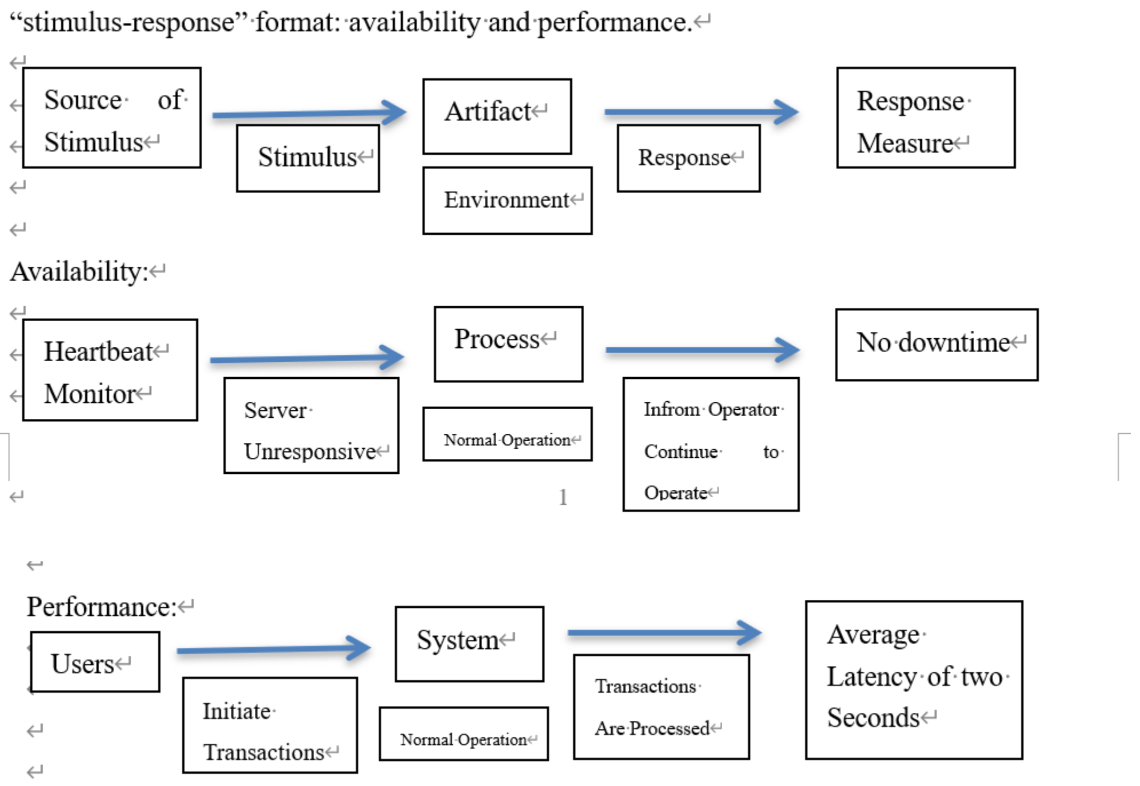
#！ 只保留了和研究生讲过的相关的

15年考题(包括多个版本

1. **Where do software architecture come from? List five possible sources of software architecture.**

   1. Requirements 2.System stakeholders 3.Development organization 4.Architects 5.Technical environment

2. **How to model quality attribute scenarios? Graphically model two quality attributes in "stimulus–response" format: availability and performance.**



“stimulus-response” format: availability and performance.

3. **Describe relationships between architecture patterns and tactics. List four tactics names and describe their usage.**
   a. Tactics are simpler than patterns, they use a single structure or mechanism to address a single architectural force
   b. Patterns typically combine multiple design decisions into a package
   c. Patterns and tactics together constitute the software architect’s primary tools

d. Tactics are building blocks of design from which architectural patterns are created

e. Most patterns consist of several different tactics

4. Briefly describe the general activities involved in a software architecture process.

a. Specify the ASRs, architects have to identify ASRs, usually after doing a significant bit of work to uncover candidate ASRs.

b. Architecture design, to locate architecturally significant requirements, capture quality attribute requirements and choose, generate, tailor and analyze design decisions for achieving those requirements

c. Documenting, creating an architecture isn’t enough. It has to be communicated in a way to let its stakeholders use it properly to do their jobs. Documenting software architectures is necessary.

d. Architecture evaluation, Architecture is such an important contributor to the success of a system and software engineering project that it makes sense to pause and make sure that the architecture that you’ve designed will be able to provide all that’s expected of it.

还有一个问题是每个环节的输入输出，后面提到了，先写上来

1. Specify the ASRs, architects have to identify ASRs, usually after doing a significant bit of work to uncover candidate ASRs.

Inputs: stakeholders,

outputs: prioritized quality attribute scenarios

2. Architecture design, to locate architecturally significant requirements, capture quality attribute requirements and choose, generate, tailor and analyze design decisions for achieving those requirements.

Inputs: prioritized quality attribute scenarios, requirements, constraints, patterns and tactics;

outputs: sketches of candidate views

3. Documenting, creating an architecture isn’t enough. It has to be communicated in a way to let its stakeholders use it properly to do their jobs. Documenting software architectures is necessary.

Inputs: sketches of candidate views, stakeholders;

outputs: chosen, combined views plus doc

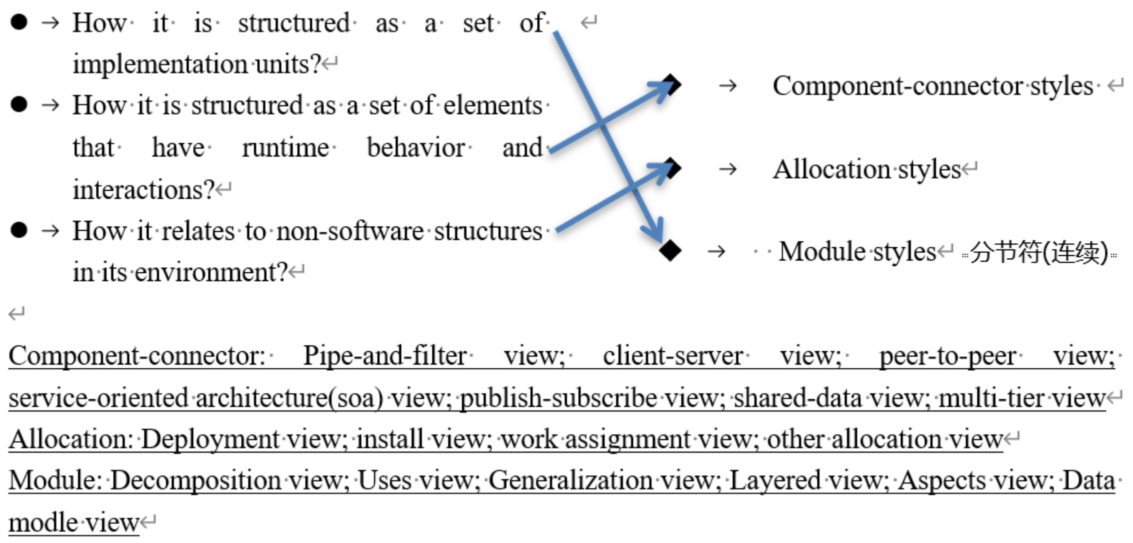
4. Architecture evaluation, Architecture is such an important contributor to the success of a system and software engineering project that it makes sense to pause and make sure that the architecture that you’ve designed will be able to provide all that’s expected of it.

Inputs: stakeholders, prioritized quality attribute scenarios;

outputs: chosen, combined views plus doc

5. Explain the context, benefits and limitations of Broker Architecture Pattern.
6. Briefly describe the fundamental principles of SOA and discuss the impact of SOA on quality attributes like interoperability, scalability and security.

7. Map each of the following questions (on the left) with the architectural style/view (on the right) that addresses the question. List four views of each category of style.



● → How it is structured as a set of implementation units?
● → How it is structured as a set of elements that have runtime behavior and interactions?
● → How it relates to non-software structures in its environment?
→ Component-connector styles
→ Allocation styles
→ Module styles 分节符(连续)

Component-connector: Pipe-and-filter view; client-server view; peer-to-peer view; service-oriented architecture(soa) view; publish-subscribe view; shared-data view; multi-tier view

Allocation: Deployment view; install view; work assignment view; other allocation view

Module: Decomposition view; Uses view; Generalization view; Layered view; Aspects view; Data modle view

8. What is difference between architecture and design? What is difference between architecture and structure?

All architecture is software design, but not all design is software architecture, architecting is a part of the design process. Architecture provides an abstract view of a design. Architecture is high–level designs and a set of design decisions; 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. Architecture is high–level structure. Architecture defines structure. Properties of structures are induced by the design of the architecture.

9. Explain the context, benefits and limitations of Broker Architecture Pattern.

The broker pattern defines a runtime component, called a broker, that mediates the communication between a number of clients and servers.

Benefits:

The pattern has a high extensibility and reusability, the server can be edited dynamically.

Weakness:

Brokers add a layer of indirection, and hence latency, between clients and servers, and that layer may be a communication bottleneck; the broker can be a single point of failure; a broker adds up-front complexity; a broker may be a target for security attacks; a broker may be difficult to test

10. Describe the difference and relationship between software requirements, quality attributes, and architecturally significant requirements.

Non-functional requirements or architectural requirements are alternative terms used for quliaty attributes. Quality attributes are over and above of system's functionality. An architecturally significant requirements is a requirement that will have a profound effect on the architecture. The more difficult and important the QA requirement, the more likely it is to significantly affect the architecture and hence to be an ASR. If a requirement affects the making of a critical architectural design decision, it is by definition an ASR.

11. What is the nature of component-connector style (patterns)? Describe the model-view-controller pattern as an example.

The MVC pattern breaks functionality into three components: a model, a view and a controller that mediates between the model and the view. The model is a representation of the application data or state. The view is a user interface component, and the controller manages the interaction between the model and the view, translating user actions into changes to the model or changes to the view. The notifies relation connects instances of model, view and controller.

12. Briefly describe the Attribute-Driven Design (ADD) process.

a. Confirm there is sufficient requirements information

b. Choose an element of the system to decompose

c. Identify the ASRs for the chosen element

d. Choose a design concept that satisfies the ASRs

4.1 Identify design concerns

4.2 List alternative patterns/tactics for subordinate concerns

4.3 Select patterns/tactics from the list

4.4 Determine relationship between patterns/tactics and ASRs

4.5 Capture preliminary architectural views

4.6 Evaluate and resolve inconsistencies

e. Instantiate architectural elements and allocate responsibilities

f. Define interfaces for instantiated elements

g. Verify and refine requirements and make them constraints for instantiated elements

h. Repeat until all the ASRs have been satisfied

1. **Explain the context, benefits and limitations of Layered Architecture Style.**

The layered pattern defines layers(groupings of modules that offer a cohesive set of services) and a unidirectional allowed–to–use relation among the layers. The pattern is usually shown graphically by stacking boxes representing layers on top of each other.

Benefits:

It can divide a complex problem into several parts; it has a high extensibility and reusability; connections among components can be shown by adjacency, and where "above" and "below" matter.

Weakness:

The addition of layers adds up–front cost and complexity to a system; layers contribute a performance penalty; if the layering is not designed correctly, it may actually get in the way, by not providing the lower–level abstractions; if many instances of layer bridging occur, the system may not meet its portability and modifiability goals that strict layering helps to achieve;

16年：包括2版

1. What is difference between architecture and design? What is difference between architecture and structure?重复
2. **What are the main kinds of requirements that are considered to influence architecture design decisions and how these requirements are usually specified?**

Functional requirement, quality requirements, constraints.

Functional requirements state what the system must do and address how the system provides value to the stakeholders.

Quality requirements are desirable characteristics of the overall system that system should provide.

Constraints are pre-specified design decisions that have been already made. Constraints are satisfied by accepting the design decision and reconciling it with other affected design decision.

3. Explain the context, benefits and limitations of Layered Architecture Style.重复
4. Briefly describe the general activities involved in a software architecture process, and the major inputs and outputs at each activity. 重复
5. **Briefly describe the fundamental principles of Service Oriented Architecture (SOA) and discuss the impact of SOA on quality attributes like performance, scalability and security.**
   a. Boundaries are explicit; services are autonomous; share schemas and contracts, not implementations; service compatibility is based on policy.
   b. Interoperability: The interfaces are defined in the indifferent way so that they are independent of the platform. Services in different systems can be interacted in the same way.
   Scalability: SOA is low-coupling. The interfaces are not bind to the specific implements. When the implements need to be changed, the interfaces can still remain.
   Security: It adopts the Web service security standard to ensure the security. It includes the exchange of certification, completeness of information and secrecy of information.
6. **Describe the difference and relationship between software requirements, quality attributes, and architecturally significant requirements.**

Non-functional requirements or architectural requirements are alternative terms used for quliaty attributes. Quality attributes are over and above of system's functionality. An architecturally significant requirements is a requirement that will have a profound effect on the architecture. The more difficult and important the QA requirement, the more likely it is to significantly affect the architecture and hence to be an ASR. If a requirement affects the making of a critical architectural design decision, it is by definition an ASR.

7. **What is the nature of component-connector style (patterns)? Describe the model-view-controller pattern as an example.**

The MVC pattern breaks functionality into three components: a model, a view and a controller that mediates between the model and the view. The model is a representation of the application data or state. The view is a user interface component, and the controller manages the interaction between the model and the view, translating user actions into changes to the model or changes to the view. The notifies relation connects instances of model, view and controller.

8. Briefly describe the Attribute-Driven Design (ADD) process 重复
9. **What distinguishes an architecture for a software product line from an architecture for a single product? 重复太多了，记一下**

   A software product line is a set of software intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way. SPL is usually multiple simultaneous variants instead of releases and versions of a single product. SPL uses systematic variation. And SPL architecture is not just a reconfigurable architecture.

10 Design a simple architecture for an automated teller machine (ATM) application in the three-tier style. Explain what responsibilities each component in the architecture has. Document your design using the "Views and Beyond" approach.

17年考题

1. **Briefly describe the general activities in a software architecture process, and the major inputs and outputs at each activity.**
2. What are generic design strategies applied in designing software? Give a concise working example with software architecture for each strategy.
3. **How to model quality attribute scenarios? Graphically model two quality attributes in "stimulus-response" format: availability and modifiability.**
4. What are ASR? List four sources and methods for extracting and identifying ASRs.重复
5. Describe 4+1 view 重复
6. Layered pattern 和 Multi-tier pattern 的区别
7. 描述 ADD 过程 重复
8. **为什么软件架构需要用不同的视图描述？举出四种视图的例子（列出名称和目的）**

19 年

1. 如何对质量属性场景建模？画出 **Interoperability 和 modifiability** 的刺激-响应图
2. What are ASR? List four sources and methods for extracting and identifying ASRs.重复
3. 4+1 视图介绍:

   Logical view: describes architecturally significant elements of the architecture and the relationships between them

Process view: describes the concurrency and communications elements of an architecture

Physical view: depicts how the major processes and components are mapped on to the applications hardware

Development view: captures the internal organization of the software components as held in e.g. a configuration management tool

4. What are generic design strategies applied in designing software? Give a concise working example with software architecture for each strategy. (和17年一样的)
5. **Map, and list four views of each category of style.（每年必考题）（其实没有的但是考太多了留下来了）**
6. Explain the context, benefits and limitations of Broker Architecture Pattern.
7. 微服务 和 SOA 的区别，相同点（按理说这是企业老师部分，但是可以参考，ppt有）

2018

1.软件架构的关注点有哪些?利益相关方有哪些?

关注点：组件接口 组件通信和依赖项 组件的责任

利益相关方：架构团队、涉众

2.Software requirements,quality attributes,ASRs的区别和联系

软件需求是用戶为了解决问题或达到某些目标所需要的条件或能力。

质量属性是整个系统的可取特征。系统应该在其功能需求之上提供。

ASR是对架构有深远影响的需求。

首先质量属性属于非功能需求，非功能需求包含于软件需求中，ASRs包含于软件需求中，并且多为质量 属性。

3.What is the nature of component connector style?以MVC pattern举例

MVC模式将系统功能分为三个组件：模型、视图和在模型和视图之间进行中介的控制器

模型是应用程序数据或状态的表示，它包含（或提供）应用程序逻辑。

视图是生成表示的用戶界面组件 或者允许某种形式的用戶输入，或者控制器管理模型和视图之间的交互。

4.如何对质量属性场景建模?画出 availability 和 modifiability 的刺激 – 响应图

使用刺激响应图为质量属性场景建模 availability: modifiablity

5.risks, sensitivity points, trade-off points 是什么?各举一个例子

系统架构风险是指架构设计中潜在的、存在问题的架构决策所带来的隐患。

敏感点是指为了实现某种特定的质量属性，一个或多个构件所具有的特性。

权衡点是影响多个质量属性的特性，是多个质量属性的敏感点。 例子没找到

6.连线,并对每种 style 列出四种视图 无

7.Layered pattern 和 Multi-tier pattern 的区别

分层模式与Multi-tier 模式： 分层模式定义了层（提供内聚服务集的模块组）和层之间允许使用的单向 关系。至少有两层。较低的层不能使用上面的层。添加层会增加系统的前期成本和复杂性。层会导致性 能损失。

Multi-tier模式： 许多系统的执行结构被组织为一组组件的逻辑分组。每个分组称为一层。将组件分组到层可以基于各种标准，例如组件的类型、共享相同的执行环境或具有相同的运行时目的。

8.描述 ADD 过程

ADD是将模块分解过程建立在软件必须满足的质量属性之上，再把功能作为模块的实例化来把功能附加 在模块上。即先满足质量，在对功能进行分配。即以质量为主要矛盾，功能为次要矛盾来进行体系结构 设计。 描述ADD过程： Step1:确定有足够的需求信息 Step2:选择系统中一个元素进行解耦 Step3:识别 所选元素的ASRs Step4：选择满足ASRs的设计

9.为什么软件架构需要用不同的视图描述?举 出四种视图的例子(列出名称和目的)

不同的视图支持不同的目标和用戶，突出不同的系统元素和关系。 不同的视图在不同程度上暴露了不同的质量属性。

逻辑视图：描述体系结构中重要的元素以及它们之间的关系。

流程视图：描述体系结构的并发性和通信元素。

物理视图：描述如何将主要流程和组件映射到应用程序硬件。

开发视图：捕获软件组件的内部组织，如配置管理工具

10.软件产品线架构如何实现可变性?描述可变 性机制的工作方式 19年有原题

11设计一个飞行模拟软件,要求能模拟多种飞 机的特性。为了在将来支持更多飞机种类,要 求使用策 略模式。画出架构图和类

2019软件系统设计试卷回忆

1. Where do software architecture come from? List five possible sources of software architecture.
2. How to model quality attribute scenarios? Graphically model one quality attributes in "stimulus–response" format: interoperability.
3. Why should a software architecture be documented using different views? Give the name and purposes of 4 example views.（重复）
4. architecture、structure和design的区别?
5. risks, sensitivity points, trade–off points 是什么？各举一个例子 （）