

全国计算机技术与软件专业技术资格（水平）考试

2019 下半年 系统架构设计师 上午试卷解析

试题 (1)

参考答案: D

答案解析: (P_i, P_j) 表示前驱图的一个进程之间的逻辑关系, 该图一共有 11 个逻辑关系。记为:

$$\rightarrow = \{(P_1, P_2), (P_1, P_3), (P_2, P_3), (P_2, P_5), (P_3, P_6), (P_3, P_4), (P_4, P_7), (P_5, P_6), (P_6, P_7), (P_6, P_8), (P_7, P_8)\}.$$

试题 (2) (3)

参考答案: A B

答案解析: $4KB = 2^{12}B$, 也就是页面大小是 $2^{12}B$, 这需要有 12 位二进制数表示, 这 12 位二进制的数也是页内地址。逻辑地址为十六进制 5148H, 其中的 148 可以分别用 4 位二进制的数表示, 这里一共需要 12 位二进制的数。

与页内地址的 12 位正好对应。而逻辑地址是由页号+页内地址组成的, 既然 148 是页内地址, 剩下的 5 就表示页号了。页号与页帧号的对应关系查表可得。

淘汰页面首先考虑不在内存的页面(状态位), 如果都在内存则在看访问位为 0 的页面, 根据局部性原理选择不经常访问的页面。如果所有页面都会被访问到, 最后再看修改位为 0 的页面, 选择没有被修改过的页面淘汰, 避免数据不一致的问题。

试题 (4)

参考答案: B

答案解析:

在网络操作系统环境中, 若 UserA 的文件或文件夹被共享后, 则其安全性与未共享时相比将会有所下降, 这是因为访问 UserA 的计算机或网络的人可能会读取、复制或更改共享文件夹中的文件。

试题 (5)

参考答案: C

答案解析: 存储过程(Stored Procedure)是在大型数据库系统中, 一组为了完成特定功能的 SQL 语

句集，它存储在数据库中，一次编译后永久有效，用户通过指定存储过程的名字并给出参数（如果该存储过程带有参数）来执行它。

存储过程是数据库所提供的一种数据库对象，通过存储过程定义一段代码，提供给应用程序调用来执行。从安全性的角度考虑，更新数据时，通过提供存储过程让第三方调用，将需要更新的数据传入存储过程，而在存储过程内部用代码分别对需要的多个表进行更新，从而避免了向第三方提供系统的表结构，保证了系统的数据安全。

试题 (6) (7)

参考答案: D A

答案解析: 根据 Armstrong 公理的传递律，可以判断出 F 蕴涵 $A \rightarrow D$ 、 $A \rightarrow E$ 、 $B \rightarrow E$ ，故 F 存在传递依赖。 $U_1 \cap U_2 = B$ ， B 可以唯一确定 U_2 的 D 、 E 。因此分解 ρ 是无损连接。

U_1 蕴含 $A \rightarrow BC$ ， U_2 蕴含 $B \rightarrow D$ ， $D \rightarrow E$ 。因此该分解也保持了函数依赖。

试题 (8)

参考答案: D

答案解析: 在分布式数据库中，局部 DBMS 中的内模式与概念模式与集中数据库是完全一致的，不同之处在于新增的全局 DBMS，而整个全局 DBMS，可以看作是相对于局部概念模式的外模式。由于外模式部分有一系列的分布模式、分片模式、全局概念模式和全局外模式，以及多级映射使得用户在使用分布式数据库时，可以使用集中式数据库同样的方式。

(1) 全局外模式。全局外模式是全局应用的用户视图，是全局概念模式的子集，该层直接与用户（或应用程序）交互。

(2) 全局概念模式。全局概念模式定义分布式数据库中数据的整体逻辑结构，数据就如同根本没有分布一样，可用传统的集中式数据库中所采用的方法进行定义。

(3) 分片模式。在某些情况下，需要将一个关系模式分解成为几个数据片，分片模式正是用于完成此项工作的。

(4) 分布模式。分布式数据库的本质特性就是数据分布在不同的物理位置。分布模式的主要职责是定义数据片段（即分片模式的处理结果）的存放节点。

(5) 局部概念模式。局部概念模式是局部数据库的概念模式。

(6) 局部内模式。局部内模式是局部数据库的内模式。

试题 (9) (10)

参考答案: C A

答案解析: 生命攸关系统是指一个系统的失效或误动作会产生诸如: 人员重伤或死亡, 或者设备的严重毁损, 或者环境的危害。

通常在开发安全做关软件时, 需求分析阶段必须考虑安全性需求, 使其不会出现不可接受的违反系统安全的行为需求。

软件安全需求的获取是根据已知的系统信息, 如: 软件危害条件等以及其他一些类似的系统数据和通用惯例, 完成通用软件安全性需求的裁剪和特定软件安全性需求的获取工作。

试题 (11)

参考答案: C

答案解析: 在嵌入式系统的应用开发中, 采用嵌入式实时操作系统(简称 RTOS)能够支持多任务, 使得程序开发更加容易, 便于维护, 同时能够提高系统的稳定性和可靠性。

嵌入式实时操作系统存在多种调度算法:

(1) 优先级调度算法

系统为每个任务分配一个相对固定的优先顺序, 调度程序根据任务优先级的高低程度, 按时间顺序进行高优先级任务优先被调度。

(2) 抢占式优先级调度算法

在优先级调度算法的基础上, 允许高优先级任务抢占低优先级任务而运行。 (3) 最晚截止期调度算法是指系统按每个任务的最接近其截止期末端的时间进行调度, 系统根据当前任务截止期的情况, 选取最接近截止期的任务运行。

(4) 最早截止期调度算法

是指系统按每个任务的截止期时间, 选取最早到截止期的头端时间的任务进行调度。

试题 (12)

参考答案: B

答案解析: 混成系统在工业控制和国防等领域大量存在。同时, 现代计算机技术的高速发展和普及应用, 为系统的模型化、优化控制和决策问题提供了强有力的技术支持。

混成系统一般由离散分离组件和连续组件并行或串行组成, 组件之间的行为由计算模型进行控制。

试题 (13)

参考答案: D

答案解析: 网络中的计算机是通过 IP 地址来代表其身份的, 但是它只能表示某台特定的计算机, 而这台计算机上可能同时提供诸如 Web 服务、数据库服务等多种服务, 此时可以使用端口号来区分相同计算机上的不同服务。起到对应用层进程的寻址的作用。

试题 (14)

参考答案: C

答案解析: 一个 Web 页面访问的过程如下:

- (1) 浏览器首先查询本机的系统, 获取主机名对应的 IP 地址。
- (2) 若本机查询不到相应的 IP 地址, 则会发起 DNS 请求, 获取主机名对应的 IP 地址。(3) 使用查询到的 IP 地址, 直接访问目标服务器。
- (4) 浏览器发送 HTTP 请求。
- (5) 服务器从请求信息中获得客户机想访问的主机名、Web 应用、Web 资源。
- (6) 服务器读取相应的主机下的 Web 应用, Web 资源。并用读取到的 Web 资源数据, 创建并回送一个 HTTP 响应。
- (7) 客户机浏览器解析回送的资源, 并显示结果。

不难看出在浏览器发出 HTTP 请求报文之前是不可能发送请求信息, 获取将要访问的 Web 应用。

试题 (15)

参考答案: A

答案解析: DHCP (DynamicHostConfigurationProtocol, 动态主机配置协议) 通常被应用在大型的局域网络环境中, 主要作用是集中的管理、分配 IP 地址, 使网络环境中的主机动态的获得 IP 地址、网关地址、DNS 服务器地址等信息, 并能够提升地址的使用率。

DHCP 协议采用 UDP 作为传输协议, 客户端发送广播消息到服务器的 68 号端口, 服务器回应广播消息给客户端的 67 号端口。

DHCP 客户端从 DHCP 服务器获取 IP 地址, 主要通过: 发现、提供、选择、确认, 四个阶段进行。

如果网络中有多个 DHCP 服务器发送 OFFER 报文，客户端只根据第一个收到的 OFFER 报文，返回 REQUEST 报文。

在网络范围内可能存在多个 DHCP 服务器，各自负责不同的网段，也可能由同一个 DHCP 服务器，负责多个不同网段的地址分配。

试题（16）（17）

参考答案: A B

答案解析: 通常用户采用评价程序来评价系统的性能。评价程序一般有专门的测量程序、仿真程序等，而评测准确度最高的评价程序是真实程序。在计算机性能评估中，通常将评价程序中用得最多、最频繁的那部分核心程序作为评价计算机性能的标准程序，称其为基准测试程序。

试题（18）（19）

参考答案: C D

答案解析: 关键成功因素法是由 JohnRockart 提出的一种信息系统规划方法。该方法能够帮助企业找到影响系统成功的关键因素，通过分析来确定企业的信息需求，从而为管理部门控制信息技术及其处理过程提供实施指南。

关键成功因素法通过对关键成功因素的识别，找出实现目标所需要的关键信息集合，从而确定系统开发的优先次序。关键成功因素来源于组织的目标，通过组织的目标分解和关键成功因素识别、性能指标识别，一直到产生数据字典。

试题（20）（21）

参考答案: A D

答案解析: 应用集成是指两个或多个应用系统根据业务逻辑的需要而进行的功能之间的相互调用和互操作。应用集成需要在数据集成的基础上完成。应用集成在底层的网络集成和数据集成的基础上实现异构应用系统之间语义层次上的互操作。它们共同构成了实现企业集成化运行最顶层会聚集成所需要的，技术层次上的基础支持。

系统应用集成构建统一标准的基础平台，在各个应用系统的接口之间共享数据和功能，基本原则是保证应用程序的独立性。系统应用集成提供了 4 个不同层次的服务，最上层服务是流程控制服务。

试题 (22) (23)

参考答案: A A

答案解析: 结构化范型也称软件生命周期方法学, 属于传统方法学。把软件生命周期划分成若干个阶段, 每个阶段的任务相对独立, 而且比较简单, 便于不同人员分工协作, 从而降低了整个软件开发过程的困难程度。在传统的软件工程方法中, 软件的生存周期分为定义时期、开发时期、运行和维护时期这几个阶段。

可行性研究属于软件定义阶段的主要任务。

试题 (24) (25)

参考答案: D B

答案解析: 需求变更管理是需求管理的重要内容。需求变更管理的过程主要包括问题分析和变更描述、变更分析和成本计算、变更实现。具体来说, 需求变更是因为需求发生变化, 根据软件工程思想, 需求说明书一般要经过论证, 如果在需求说明书经过论证后, 需要在原有需求基础上追加和补充新的需求或对原有需求进行修改和削减, 均属于需求变更。因此, 需求变更必然会带来相应的问题, 绝不是百利而无一害的。

试题 (26) (27) (28)

参考答案: D A B

答案解析: 软件方法学是软件开发全过程的指导原则与方法体系。其另一种含义是以软件方法为研究对象的学科。从开发风范上看, 软件方法有自顶向下的开发方法、自底向上的开发方法。在实际软件开发中, 大都是自顶向下与自底向上两种方法的结合, 只不过是以何者为主而已。自顶向下是指将一个大问题分化成多个可以解决的小问题, 然后逐一进行解决。每个问题都会有一个模块去解决它, 且每个问题包括抽象步骤和具体步骤。形式化方法是指采用严格的数学方法, 使用形式化规约语言来精确定义软件系统。非形式化的开发方法是通过自然语言、图形或表格描述软件系统的行为和特性, 然后基于这些描述进行设计和开发, 而形式化开发则是基于数学的方式描述、开发和验证系统。

试题 (29) (30)

参考答案: D B

答案解析: 软件开发工具是指用于辅助软件开发过程活动的各种软件。其中, 软件建模工具是辅

助建立软件系统的抽象模型的。常见的软件建模工具包括 RationalRose、Together、WinA&D、QuickUML、EclipseUML 等。

试题 (31) (32)

参考答案: B A

答案解析: 从工程管理角度来看, 软件设计可分为概要设计和详细设计两个阶段。概要设计也称为高层设计或总体设计, 即将软件需求转化为数据结构和软件的系统结构; 详细设计也称为低层设计, 即对结构图进行细化, 得到详细的数据结构与算法。

试题 (33)

参考答案: D

答案解析: 软件结构化设计包括架构设计、接口设计、数据设计和过程设计等任务。它是一种面向数据流的设计方法, 是以结构化分析阶段所产生的成果为基础, 进一步自顶而下、逐步求精和模块化的过程。

试题 (34)

参考答案: B

答案解析: 模块化设计是将一个待开发的软件分解成为若干小的简单部分——模块。具体来说, 模块是指执行某一特定任务的数据结构和程序代码。通常将模块的结构和功能定义为其外部特性, 将模块的局部数据和实现该模块的程序代码称为内部特性。模块独立是指每个模块完成相对独立的特定子功能, 与其他模块之间的关系最简单。通常用内聚和耦合两个标准来衡量模块的独立性, 其设计原则是“高内聚、低耦合”。

试题 (35) (36) (37)

参考答案: A B D

答案解析: 基于构件的软件开发中, 已有的构件分类方法可以归纳为三大类:

(1) 关键字分类法

根据领域分析的结果将应用领域的概念按照从抽象到具体的顺序逐次分解为树形或有向无回路图结构。

(2) 刻面分类法

利用 Facet (刻面) 描述构件执行的功能、被操作的数据、构件应用的语境或任意其他特征。

(3) 超文本方法

基于全文检索技术，使得检索者在阅读文档过程中可以按照人类的联想思维方式任意跳转到包含相关概念或构件的文档。

试题 (38)

参考答案: C

答案解析: 本题考查构件组装的基础知识。

构件组装是将库中的构件经适当修改后相互连接，或者将它们与当前开发项目中的软件元素相连接，最终构成新的目标软件。构件组装技术大致可分为基于功能的组装技术、基于数据的组装技术和面向对象的组装技术。

试题 (39) (40)

参考答案: B C

答案解析: 逆向工程过程能够导出过程的设计模型（实现级）、程序和数据结构信息（结构级）、对象模型、数据和控制流模型（功能级）以及 UML 状态图和部署图（领域级）。其中，结构级包括反映程序各部分之间相关依赖关系的信息；功能级包括反映程序段功能及程序段之间关系的信息。

试题 (41)

参考答案: D

答案解析: 重组是指在同一抽象级别上转换系统描述形式；设计恢复是指借助工具从已有程序中抽象出有关数据设计、总体结构设计和过程设计等方面的信息；重构工程是指在逆向工程所获得信息的基础上，修改或重构已有的系统，产生系统的一个新版本。

试题 (42) (43)

参考答案: A D

答案解析: 软件性能测试类型包括负载测试、强度测试和容量测试等。其中：

- (1) 负载测试用于测试超负荷环境中程序是否能够承担；
- (2) 强度测试是在系统资源特别低的情况下考查软件系统极限运行情况；

(3) 容量测试可用于测试系统同时处理的在线最大用户数量。

试题 (44) (45)

参考答案: D B

答案解析: 软件系统需从不同的角度进行描述。其著名的 4+1 视角架构模型 (The "4+1" View Model of Software Architecture) 提出了一种用来描述软件系统体系架构的模型, 这种模型是基于使用者的多个不同视角出发。这种多视角能够解决多个“利益相关者”关心的问题。利益相关者包括最终用户、开发人员、系统工程师、项目经理等, 他们能够分别处理功能性和非功能性需求。4+1 视角架构模型的五个主要的视角为逻辑视图、开发视图、处理视图、物理视图和场景。其中逻辑视图通常包括类图, 对象图, 状态图和协作图。

试题 (46) (47) (48)

参考答案: D C A

答案解析: 数据流体系结构包括批处理体系结构风格和管道-过滤器体系结构风格。虚拟机体系结构风格包括解释器体系结构风格和规则系统体系结构风格。图中描述的为层次型体系结构风格。

试题 (49) (50)

参考答案: A C

答案解析: 面向服务的体系结构(Service-oriented Architecture, SOA)是一种软件系统设计方法, 通过已经发布的和可发现的接口为终端用户应用程序或其他服务提供服务。

企业服务总线(Enterprise Service Bus, ESB)是构建基于 SOA 解决方案时所使用基础架构的关键部分, 是由中间件技术实现并支持 SOA 的一组基础架构功能。ESB 支持异构环境中的服务、消息, 以及基干事件的交互, 并且具有适当的服务级别和可管理性。简而言之, ESB 提供了连接企业内部及跨企业间新的和现有软件应用程序的功能, 以一组丰富的功能启用管理和监控应用程序之间的交互。在 SOA 分层模型中, ESB 用于组件层以及服务层之间, 它能够通过多种通信协议连接并集成不同平台上的组件将其映射成服务层的服务。

试题 (51) (52) (53)

参考答案: A A C

答案解析: 基于架构的软件开发模型(Architecture-Based Software Design Model, ABSDM)把整个基

于架构的软件过程划分为架构需求、设计、文档化、复审、实现、演化等 6 个子过程。

绝大多数的架构都是抽象的，由一些概念上的构件组成。例如，层的概念在任何程序设计语言中都不存在。因此，要让系统分析师和程序员去实现架构，还必须得把架构进行文档化。文档是在系统演化的每一个阶段，系统设计与开发人员的通信媒介，是为验证架构设计和提炼或修改这些设计（必要时）所执行预先分析的基础。架构文档化过程的主要输出结果是架构需求规格说明和测试架构需求的质量设计说明书这两个文档。生成需求模型构件的精确的形式化的描述，作为用户和开发者之间的一个协约。

试题 (54) (55) (56) (57)

参考答案: A D C B

答案解析: 创建型模式是对对象实例化过程的抽象。例如 Singleton 模式确保一个类只有一个实例，并提供了全局访问入口；Prototype 模式允许对象在不了解要创建对象的确切类以及如何创建等细节的情况下创建自定义对象；Builder 模式将复杂对象的构建与其表示分离。

结构型模式主要用于如何组合已有的类和对象以获得更大的结构，一般借鉴封装、代理、继承等概念将一个或多个类或对象进行组合、封装，以提供统一的外部视图或新的功能。行为型模式主要用于对象之间的职责及其提供的服务的分配，它不仅描述对象或类的模式，还描述它们之间的通信模式，特别是描述一组对等的对象怎样相互协作以完成其中任一对象都无法单独完成的任务。

试题 (58) (59) (60) (61) (62) (63)

参考答案: B C A C C A

答案解析: 架构的基本需求主要是在满足功能属性的前提下，关注软件质量属性，结构设计则是为满足架构需求（质量属性）寻找适当的战术。

根据题干描述，其中“数据传递时延不大于 1s，并提供相应的优先级管理”主要与性能质量属性相关，性能的战术有资源需求、资源管理和资源仲裁，此需求通常可采用资源仲裁架构策略实现该属性；

“系统采用双机热备，主备机必须实时监测对方状态，以便完成系统的实时切换”主要与可用性质量属性相关，可用性的战术有错误检测、错误恢复和错误预防，此需求通常可采用错误检测中的心跳架构策略实现该属性；

“系统应能够防止 99% 的黑客攻击”主要与安全性质量属性相关，安全性相关的战术有抵抗攻击、检测攻击和从攻击中恢复，此需求通常可采用检测攻击架构策略实现该属性。

试题 (64)

参考答案: C

答案解析: SSL(Secure Sockets Layer, 安全套接层)及其继任者 TLS(Transport Layer Security, 传输层安全)是为网络通信提供安全及数据完整性的一种安全协议，在传输层对网络连接进行加密。在设置电子邮箱时使用 SSL 协议，会保障邮箱更安全。

HTTPS 协议是由 HTTP 加上 TLS/SSL 协议构建的可进行加密传输、身份认证的网络协议，主要通过数字证书、加密算法、非对称密钥等技术完成互联网数据传输加密，实现互联网传输安全保护。

MIME 是设定某种扩展名的文件用一种应用程序来打开的方式类型，当该扩展名文件被访问的时候，浏览器会自动使用指定应用程序来打开。它是一个互联网标准，扩展了电子邮件标准，使其能够支持：非 ASCII 字符文本；非文本格式附件（二进制、声音、图像等）；由多部分(Multiple Parts)组成的消息体；包含非 ASCII 字符的头信息(Header Information)。PGP 是一套用于消息加密、验证的应用程序，采用 IDEA 的散列算法作为加密与验证之用。PGP 加密由一系列散列、数据压缩、对称密钥加密，以及公钥加密的算法组合而成。每个公钥均绑定唯一的用户名和/或者 E-Mail 地址。

因此，上述选项中 MIME 是扩展了电子邮件标准，不能用于保障电子邮件安全。

试题 (65)

参考答案: B

答案解析: 网络冗余设计的目的就是避免网络组件单点失效造成应用失效；备用路径是在主路径失效时启用，其和主路径承担不同的网络负载；负载分担是网络冗余设计中的一种设计方式，其通过并行链路提供流量分担来提高性能；网络中存在备用链路时，可以考虑加入负载分担设计来减轻主路径负担。

试题 (66)

参考答案: D

答案解析: 发表权也称公开作品权，指作者对其尚未发表的作品享有决定是否公之于众的权利，发表权只能行使一次，且只能为作者享有。

著作权的发行权，主要是指著作权人许可他人向公众提供作品原件或者复制品。而发行权可

以行使多次，并且不仅仅为作者享有。

传播权指著作权人享有向公众传播其作品的权利，传播权包括表演权、播放权、发行权、出租权、展览权等内容。

署名权是作者表明其身份，在作品上署名的权利，它是作者最基本的人身权利。根据《中华人民共和国著作权法》的规定，作者的署名权、修改权、保护作品完整权的保护期不受限制。

试题 (67)

参考答案: A

答案解析: 著作权是知识产权中的例外，因为著作权的取得无须经过个别确认，这就是人们常说的“自动保护”原则。软件经过登记后，软件著作权人享有发表权、开发者身份权、使用权、使用许可权和获得报酬权。

软件著作权自软件开发完成之日起产生。自然人的软件著作权，保护期为自然人终生及其死亡后 50 年，截止于自然人死亡后第 50 年的 12 月 31 日；软件是合作开发的，截止于最后死亡的自然人死亡后第 50 年的 12 月 31 日。法人或者其他组织的软件著作权，保护期为 50 年，截止于软件首次发表后第 50 年的 12 月 31 日，但软件自开发完成之日起 50 年内未发表的不予保护。未经软件著作权人许可，修改、翻译、复制、发行著作人的软件的，属于侵权行为，应承担相应的民事、行政和刑事责任。

B 选项中的“软件盗版者”在法律条文中并无定义，有定义的是“侵权行为”。而根据《计算机软件保护条例》第十七条“为了学习和研究软件内含的设计思想和原理，通过安装、显示、传输或者存储软件等方式使用软件的，可以不经软件著作权人许可，不向其支付报酬。”因此 B 选项有误。

试题 (68)

参考答案: B

答案解析: 仅对源程序的标识符进行修改，如将数组名“arr”修改为“Arr”，程序其他部分不变属于侵犯软件著作权的行为。

试题 (69)

参考答案: C

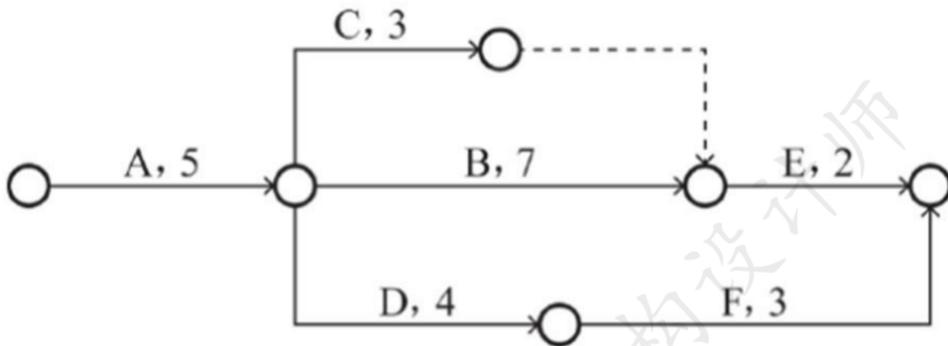
答案解析: 实际问题的数学模型往往都是近似的，常带有多个参数，而参数会随环境因素而变化。

根据数学模型求出最优解或满意解后，还需要进行灵敏性分析，对计算结果进行检验，分析计算结果对参数变化的反应程度。如果对于参数的微小变化引发计算结果的很大变化，那么这种计算结果并不可靠，并且不可信。

试题 (70)

参考答案: C

答案解析: 关键路径 (所需天数最多的路径) : ABE, 总工期=5 天+7 天+2 天=14 天。



作业 D、F 与作业 B、E 可并行实施，为不影响总工期，作业 D、F 可以在 7 天+2 天=9 天内完成，所以作业 D 最多可以延迟 2 天。

试题 (71) (72) (73) (74) (75)

参考答案: B C A A D

答案解析: 参考译文:

在系统分析阶段，必须决定如何组织、存储和管理数据。数据结构是组织、存储和管理数据的一个框架。每个文件或表都包含有关人物、地点、事物或事件的数据。文件处理环境中存在的潜在问题之一是数据冗余，这意味着两个或多个信息系统共用的数据存储在多个不同位置。

在数据库管理系统(DBMS)中，相互连接的表形成了一个统一的数据结构，极大地提高了数据质量和访问能力。实体关系图是显示系统实体之间的逻辑关系和交互的模型。它提供了系统的总体视图和用于创建物理数据结构的蓝图。规范化是通过为数据库中的每个表分配特定字段或属性来创建表设计的过程。表设计指定字段并标识特定表或文件中的主键。三种范式构成了一个渐进过程，其中第三范式代表了好的设计。大部分与业务相关的数据库都必须以这种形式设计。

2019 下半年 系统架构设计师 下午试卷 I 解析

试题一 (25 分)

本题主要考查考生对于软件架构风格的理解与掌握，以及对软件质量属性的理解、掌握和应用。在解答该问题时，应认真阅读题干中给出的场景与需求描述，分析需求与架构风格的对应关系，并需要理解每个需求描述了何种质量属性，根据质量属性描述对其归类。

【问题 1】

参考答案: 该系统更适合采用基于规则的虚拟机架构风格。

- (1) 根据用户级别建立用户级别-折扣规则矩阵，在系统启动时加载并支持运行过程中动态更新，灵活性好；
- (2) 加入新的用户级别和折扣规则时需要增加相应的类来扩展，可通过系统重启、动态反射或动态加载扩展，扩展性较差；
- (3) 可根据类型判断或策略模式直接获得用户级别对应的折扣规则对象实时计算，性能很好。

答案解析: 面向对象设计模式中的策略模式和虚拟机中的基于规则的架构风格是动态规则场景中两种常用的解决方案。从灵活性、可扩展性和性能方面综合比较来看，基于规则的虚拟机风格在灵活性和可扩展性两个方面均具备较大优势，而从性能方面会比面向对象处理速度差一些。

【问题 2】

参考答案: (1) 安全性 (2) 可修改性 (3) h (4) j (5) e (6) k

答案解析: 质量属性效用树时对质量属性进行分类、权衡、分析的架构分析工具，主要关注系统的性能、可用性、可修改性和安全性四个方面。根据对相关质量属性的定义和含义，题干中：(c) 在正常负载情况下，系统应在 0.5 秒内对用户的商品查询请求进行响应和(h)系统在展示商品的实时视频时，需要保证视频画面具有 1024x768 像素的分辨率，40 帧/秒的速率，描述的是系统的性能属性；(e)系统主站点断电后，应在 5 秒内将请求重定向到备用站点和(g)当系统发生网络失效后，需要在 15 秒内发现错误并启用备用网络，描述的是系统的可用性属性；(i)系统要扩容时，应保证在 10 人·月内完成所有的部署与测试工作和(k)更改系统的 Web 界面接口必须在 4 人·周内完成，描述的是系统的可修改性属性；(b)系统应该具备完善的安全防护措施，能够对黑客的攻击行为进行检测与防御和(j)系统应对用户信息数据库的所有操作都进行完整记录，描述的是系统的安全性属性。

试题二 (25 分)

本题考查系统过程建模的相关知识。

数据流图(DataFlowDiagram, DFD)从数据传递和加工角度,以图形方式来表达系统的逻辑功能、数据在系统内部的逻辑流向和逻辑变换过程,是结构化系统分析方法的主要表达工具及用于表示软件模型的一种图示方法。为了表达数据处理过程的数据加工情况,用一个数据流图往往是不够的。层次结构的数据流图按照系统的层次结构进行逐步分解,并以分层的数据流图反映这种结构关系,能清楚地表达和容易理解整个系统。

层次结构数据流图一般分为:顶层数据流图,中层数据流图和底层数据流图。中层数据流图中最高层次一般从 0 开始,最高层级的中层数据流图即是 0 层数据流图,0 层数据流图主要目的是将顶层流图的系统分解为若干子系统,并决定每个子系统间的数据接口和活动关系。

【问题 1】

参考答案: E1 客户, E2 厨房, E3 快餐店经理, E4 供应商

P1 订餐, P2 备餐, P3 生成报表, P4 采购食材

答案解析: 外部实体,是指系统之外的人员、事物或组织。外部实体指出系统的边界,即系统输入数据的发源地或系统所产生的数据的目标去处,如题目中的客户、厨房、供应商和快餐店经理均属于外部实体。

加工,又称处理或过程,描述输入数据流到输出数据之间的变换,也就是输入数据流经过什么处理过程后变成了输出数据。作用是把输入数据加工成所要的输出数据。每个加工都有一个名字和编号,加工的名字一般采用动宾结构短语,偶尔使用动词,如题目中提交订单、支付订单费用、加工餐品、采购食材、生成报表等均属于加工。

【问题 2】

参考答案:

- 1) E1-->P1 餐品订单
- 2) P1-->P2 餐品订单
- 3) D1-->P3 订单汇总
- 4) P3-->E3 统计报表

答案解析: 外部实体、加工、数据存储和数据流四要素在数据流程图中常见的错误:

外部实体作为数据来源和数据去处,常见错误主要集中在孤立的外部实体和实体命名方面。孤立外部实体既没有输入任何数据给系统,系统也没有为其提供任何数据,而实体命名一般采用代表人群、事物或组织名称的名词。

加工常见的错误集中在加工的输入和输出，常见的错误包括有输入而没有输出的加工，有输出而没有输入的加工，输入和输出不守恒的加工。

同样的数据存储常见的错误也集中在输入和输出，包括有输入无输出的存储，有输出无输入的存储和输入输出不守恒的存储。

而数据流的常见错误主要集中在命名方面，数据流名称应该是数据流中传递的数据内容，不要使用处理、传递、变换相关的内容来命名数据流。

【问题 3】

参考答案:

- (1) 数据流图中的处理过程可并行；系统流程图在某个时间点只能处于一个处理过程。
- (2) 数据流图展现系统的数据流；系统流程图展现系统的控制流。
- (3) 数据流图展现全局的处理过程，过程之间遵循不同的计时标准；系统流程图中处理过程遵循一致的计时标准。

答案解析：外部实体、加工、数据存储和数据流四要素在数据流程图中常见的错误：

外部实体作为数据来源和数据去处，常见错误主要集中在孤立的外部实体和实体命名方面。孤立外部实体既没有输入任何数据给系统，系统也没有为其提供任何数据，而实体命名一般采用代表人群、事物或组织名称的名词。

加工常见的错误集中在加工的输入和输出，常见的错误包括有输入而没有输出的加工，有输出而没有输入的加工，输入和输出不守恒的加工。

同样的数据存储常见的错误也集中在输入和输出，包括有输入无输出的存储，有输出无输入的存储和输入输出不守恒的存储。

而数据流的常见错误主要集中在命名方面，数据流名称应该是数据流中传递的数据内容，不要使用处理、传递、变换相关的内容来命名数据流。

试题三（25 分）

本题主要考查 CPS 系统的定义、架构层次结构和安全性质量属性。

信息物理系统 (CPS, Cyber-PhysicalSystems) 是一个综合计算、网络和物理环境的多维复杂系统，通过 3C (Computation、Communication、Control) 技术的有机融合与深度协作，实现大型工程系统的实时感知、动态控制和信息服务。CPS 实现计算、通信与物理系统的一体化设计，可使系统更加可靠、高效、实时协同，具有重要而广泛的应用前景。

【问题 1】

参考答案:

信息物理系统作为计算进程和物理进程的统一体，是集成计算、通信与控制于一体的下一代智能系统，通过人机交互接口实现和物理进程的交互，使用网络化空间以远程的、可靠的、实时的、安全的、协作的方式操控一个物理实体。

感知层主要是由传感器、控制器和采集器等设备组成，主要是通过传感器获取环境的信息数据，定时发送给服务器，并接收服务器处理结果数据后进行相应的控制器变化；网络层是连接信息世界和物理世界的桥梁，主要实现数据传输，为系统提供实时的网络服务，保证网络分组的实时可靠；控制层根据感知层的认知结果，根据物理设备传回来的数据进行相应的分析，将分析结果通过客户端以可视化的界面呈现给客户。

答案解析:

信息物理系统(CyberPhysicalSystems, CPS)作为计算进程和物理进程的统一体，是集成计算、通信与控制于一体的下一代智能系统。信息物理系统通过人机交互接口实现和物理进程的交互，使用网络化空间以远程的、可靠的、实时的、安全的、协作的方式操控一个物理实体。

信息物理系统包含了将来无处不在的环境感知、嵌入式计算、网络通信和网络控制等系统工程，使物理系统具有计算、通信、精确控制、远程协作和自治功能。它注重计算资源与物理资源的紧密结合与协调，主要用于一些智能系统上如设备互联，物联传感，智能家居，机器人，智能导航等。

信息物理系统主要分为 3 个部分，分别是感知层、网络层和控制层，感知层主要是由传感器、控制器和采集器等设备组成。感知层中的传感器作为信息物理系统中的末端设备，主要采集的是环境中的具体信息感知层主要是通过传感器获取环境的信息数据，并定时地发送给服务器，服务器接收到数据之后进行相应的处理，再返回给物理末端设备相应的信息，物理末端设备接收到数据之后要进行相应的变化；网络层主要是连接信息世界和物理世界的桥梁，主要实现的是数据传输，为系统提供实时的网络服务，保证网络分组的实时可靠；控制层主要是根据认知层的认知结果，根据物理设备传回来的数据进行相应的分析，将相应的结果返回给客户端以可视化的界面呈现给客户。

【问题 2】**参考答案:**

感知层: 2, 4, 7, 10

网络层: 8

控制层: 1, 3, 5, 6

非 CPS: 9

答案解析:

根据 CPS 感知层、网络层、控制层定义，2 步进电机控制和 4 发电机控制属于控制器，7 转速传感器和 10 火警信号探测属于传感器，因此 2, 4, 7, 10 归类为感知层设备；8 传感器总线属于数据传输通讯链路设备，归类为网络层设备；3 显控属于飞行器任务级功能，1 飞行传感器管理属于飞行器平台级功能，5 环控和 6 配电管理属于飞行器平台级下的功能组件，因此 1, 3, 5, 6 归类为控制层；而 9 飞行员属于系统使用者，不属于 CPS 系统。

【问题 3】

参考答案:

1. 感知层威胁

信息窃听：通过搭线或电磁泄漏造成数据隐私泄露；

感知破坏：未经授权对感知层信息篡改、增删或破坏。

2. 网络层威胁

拒绝服务：发送大量请求迫使服务器停止接受新请求；

选择性转发：恶意节点接收数据后有选择性的转发，破坏数据完整性。3. 控制层威胁

非授权访问：未经授权情况下不合理进入系统执行恶意操作；

恶意代码：注入对系统造成不良影响的恶意代码，对系统造成破坏。

答案解析:

感知执行层安全威胁：

感知执行层主要由各种物理传感器等组成，是整个物理信息系统中信息的来源。为了适应多变的环境，网络节点多布置在无人监管的环境中，因此易被攻击者攻击。常见的针对感知执行层的攻击方式有：

1) 感知数据破坏：攻击者未经授权，对感知层获取的信息进行篡改、增删或破坏等；

2) 信息窃听：攻击者通过搭线或利用传输过程中的电磁泄露获取信息，造成数据隐私泄露等问题；

3) 节点捕获：攻击者对部分网络节点进行控制，可能导致密钥泄露，危及整个系统的通信安全。

数据传输层安全威胁：

数据传输层一般要接入网络，而接入网络本身就会给整个物理信息系统带来威胁。一方面，作为链接感知层和控制层的数据传输的通道，其中传输的信息易成为攻击者的目标；另一方面，

由于接入网络，数据传输层易受到攻击。数据传输层的主要安全威胁如下：

- 1) 拒绝服务攻击：攻击者通过先向服务器发送大量请求，使得服务器缓冲区爆满而被迫停止接受新的请求，使系统崩溃从而影响合法用户的使用；
- 2) 选择性转发：恶意节点在接收到数据后，不全部转发所有信息，而是将部分或全部关键信息在转发过程中丢掉，破坏了数据的完整性；
- 3) 方向误导攻击：恶意节点在接收到数据包后，对其源地址和目的地址进行修改，使得数据包沿错误路径发送出去，造成数据丢失或网络混乱。

应用控制层安全威胁：

应用控制层中数据库中存放着大量用户的隐私数据，因此在这一层中一旦发生攻击就会出现大量隐私泄漏的问题。针对应用层的主要威胁有：

- 1) 用户隐私泄漏：用户的所有的数据都存储在应用控制层中的数据库中，其中包含用户的个人资料等隐私的数据都存放在数据库中，一旦数据库被攻陷，就会导致用户的隐私产生泄漏，造成很严重的影响；
- 2) 恶意代码：恶意代码是指在运行过程中会对系统造成不良影响的代码库，攻击者一般会将这些代码嵌入到注释中，脚本一旦在系统中运行，就会对系统造成严重的后果；
- 3) 非授权访问：对于一个系统来说，会有各种权限的管理者，比如超级管理员，对该系统有着最高的操作权限，一般管理员对该系统有部分的操作权限。非授权访问指的就是攻击者在未经授权的情况下不合理的访问本系统，攻击者欺骗系统，进入到本系统中对本系统执行一些恶意的操作就会对本系统产生严重的影响。

试题四（25 分）

本题主要考查分布式架构中数据库和缓存双写一致性的架构策略和方案。

Redis 是企业级系统高性能、高并发、高可用架构中非常重要的一个环节。Redis 主要解决了关系型数据库并发量低的问题，有助于缓解关系型数据库在高并发场景下的压力，提高系统的吞吐量。而在 Redis 的实际使用过程中，难免会遇到缓存与数据库双写时数据不一致的问题，这是架构策略必须要考虑的问题。

【问题 1】

参考答案：

在高并发条件下存在不同线程间网络延迟不同的情况，且缓存和数据库数据不同写入请求的速度也存在差异，并且缓存和数据库删除和写入均存在失败的可能性，这些会导致解决思路无法

解决数据同步和不一致性问题。

- (a)读数据库,
- (b)写缓存
- (c)数据库
- (d)更新缓存 key 值/删除缓存 key 值/使缓存 key 值失效

答案解析:

在缓存和数据库双写过程中:

如果是先修改数据库, 再删除缓存的方案, 当删除缓存失败时数据库中是新数据, 缓存中是旧数据, 出现数据不一致。

如果先删除缓存, 再修改数据库。缓存读不到数据, 查数据库更新缓存的时候就拿到了最新的库存数据。

如果删除缓存成功了, 而修改数据库失败了, 那么数据库中依旧是旧数据, 缓存中是空的, 那么数据不会不一致。

【问题 2】

参考答案:

存在问题: 空值缓存需要更多的键, 浪费内存空间

解决思路: 查询缓存前, 对 key 值进行过滤, 只允许系统中存在的 key 进行后续操作 (例如采用 key 的 bitmap 进行过滤)。

答案解析:

缓存穿透是指缓存和数据库中都没有的数据, 而用户或攻击者不断发起请求会导致缓存数据库压力过大。缓存穿透的常规策略有:

- 1.接口层增加校验, 如用户鉴权校验, 键校验;
- 2.缩短无效键有效时间, 以防止攻击用户的暴力攻击。

【问题 3】

参考答案:

1.搭建高可用 Redis 集群, 正式部署前进行数据预热, 在大并发访问前加载缓存键并尽量均匀分布缓存过期时间;

2.将热点数据设置为永不过期, 开启 Redis 的持久化功能, 当 Redis 启动时, 从磁盘恢复数据到缓存中。

答案解析:

缓存雪崩是指缓存中数据大批量到过期时间，而查询数据量巨大，引起数据库压力过大甚至宕机。缓存雪崩的常见策略有：

- 1.缓存数据的过期时间随机设置，防止同一时间大量数据过期现象发生；
- 2.在分布式缓存数据库部署条件下，将热点数据均匀分布在不同的缓存数据库中；
- 3.热点数据设置为永不过期。

试题五（25 分）

本题考查高性能、高并发、高可用的分布式系统架构设计实践相关知识。

在当前的技术环境下，高性能、高并发、高可用的三高架构设计是众多技术企业需要在日常工作中经常面对的常见架构需求。这些需求的常见架构策略有：分层、冗余、分隔、异步通信、分布式、安全、自动化、集群、缓存、微服务等。

【问题 1】

参考答案:

性能：(1)、(2)、(6)

安全性：(5)

可用性：(3)、(7)

易用性：(4)、(8)

答案解析:

质量属性归类，是需求分析和架构设计阶段的主要工作之一。根据对相关质量属性的定义和含义，题干中：(1)系统应支持大于 50 个终端设备的并发请求，(2)系统应能够实时识别车牌，识别时间应小于 1s，(6)独立事务操作响应时间应小于 3s，描述的是系统的性能属性；(5)可抵御常见 SQL 注入攻击，描述的是系统的安全性属性；(3)系统应 7X24 小时工作，(7)系统在故障情况下，应在 1 小时内恢复，描述的是系统的可用性属性；(4)具有友好的用户界面，(8)新用户学习使用系统的时间少于 1 小时，描述的是系统的易用性属性。

【问题 2】

参考答案: (1) d (2) e (3) i (4) h (5) g (6) f (7) a

答案解析: 三层架构就是为了符合“高内聚，低耦合”思想，把各个功能模块划分为表示层（UI）、业务逻辑层（BLL）和数据访问层（DAL）三层架构。

由于在高并发环境下，由于无法及时完成同步处理，请求往往会发生堵塞而直接导致无数的行锁表锁，甚至最后请求会堆积过多触发连接数超过上限的错误。通过使用消息队列可以异步处

理请求，从而缓解系统的压力。该案例中采用 Kafka 分布式消息队列。

缓存是分布式系统中的重要组件，主要解决高并发，大数据场景下，热点数据访问的性能问题。

提供高性能的数据快速访问。本案例中采用 Redis 键值数据库缓存。

【问题 3】

参考答案:

SOL 注入攻击，就是通过把 SOL 命令插入到 Web 表单提交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的 SQL 命令。

可以通过以下方式抵御 SQL 注入攻击：

- 1、使用 PreparedStatement
- 2、使用存储过程
- 3、验证输入/过滤输入
- 4、专业的安全产品

答案解析:

SQL 注入即是指应用程序对用户输入数据的合法性没有判断或过滤不严，攻击者可以在应用程序中事先定义好的查询语句的结尾上添加额外的 SQL 语句，在管理员不知情的情况下实现非法操作，以此来实现欺骗数据库服务器执行非授权的任意查询，从而进一步得到相应的数据信息。

SQL 注入的主要防范措施包括：

分级管理：对用户进行分级管理，严格控制用户的权限。

参数校验：编写 SQL 时禁止直接传值入 SQL，必须通过设置参数传递变量值。**参数过滤：**对参数值预过滤敏感内容，如 and, or, select, 分号，双引号等。**安全参数：**使用编程语言中提供的数据库操作安全参数的强制执行检查特性。**漏洞扫描：**利用 SQL 漏洞扫描工具及时扫描系统存在的相应漏洞。

被访数据输入必须经过严格的验证才能进入系统，拒绝未通过验证的输入直接和不：对数据库敏感信息进行加密，传统加密方法包括对称加密、非对称加密和不可逆加密。

2020 下半年 系统架构设计师 下午试卷 II 解析

写作要点

试题一 论软件设计方法及其应用

论文试题一 写作要点

1. 简要描述所参与设计的软件系统，并明确指出在系统设计过程种承担的主要工作。

2. 分析系统设计的主要方法，并详细阐述每种设计方法。

1) 净室方法

净室软件工程(净室方法)是软件开发的一种形式化方法，它可以生成高质量的软件。它使用盒结构规约进行分析和设计建模，并且强调将正确性验证(而不是测试)作为发现和消除错误的主要机制，使用统计的测试来获取认证被交付的软件的可靠性所必需的出错率信息。

净室方法从使用盒结构表示的分析和设计模型入手，一个“盒”在某特定的抽象层次上封装系统(或系统的某些方面)。通过逐步求精的过程，盒被精化为层次，其中每个盒具有引用透明性：每个盒规约的信息内容对定义其精华是足够的，不需要信赖于任何其他盒的实现。这使得分析人员能够层次地划分一个系统，从在顶层的本质表示转移向在底层的实现特定的细节。净室方法主要使用三种盒类型：黑盒、状态盒和清晰盒。

净室方法是一种严格的软件工程方法，它是一种强调正确性的数学验证和软件可靠性认证的软件过程模型，其目标和结果是非常低的出错率，这是使用非形式化方法难于或不可能达到的。

2) 结构化设计

结构化方法由结构化分析、结构化设计、结构化程序设计构成，它是一种面向数据流的开发方法。结构化分析是根据分解与抽象的原则，按照系统中数据处理的流程，用数据流图来建立系统的功能模型，从而完成需求分析工作。

①结构化设计原则

②结构化设计步骤

③结构化缺点

在结构化设计中，模块和模块之间的关系局限于信息流，限制了对模块之间众多关系的表达，也无法体现模块和模块之间其他的众多关系，包含各种各样的结构、行为、依赖、包含(在结构化设计中这种关系隐含在分层中)、继承、关联关系等等。结构化设计仅解决了模块在封装和信息隐藏方面的问题。

④结构化设计适合场景

3) 面向对象设计

面向对象的设计模型包含以包图表示的软件体系结构图、以交互图表示的用例实现图、完整精确的类图、针对复杂对象的状态图和用以描述流程化处理过程的活动图等。

①UML 与 4+1 视图

②设计原则

③设计模式

④面向对象设计适合场景

4) 原型法

结构化方法和面向对象方法有一个共同点：在系统开发初期必须明确系统的功能要求，确定系统边界。从工程学角度来看，这是十分自然的：解决问题之前必须明确要解决的问题是什么，然而对于信息系统建设而言，明确问题本身不是一件轻松的事情。

①原型分类

水平原型和垂直原型，抛弃原型、演进原型和递增原型

②原型类型的选择

③原型法适合场景

3. 结合项目实践。针对实际参与的软件设计过程，说明所采用的设计方法，并描述其具体实施过程和效果。

试题二 论软件系统架构评估及其应用

论文试题二 写作要点

1. 简要描述所参与架构评估的软件系统，并明确指出在评估过程中承担的主要工作。

2. 分析软件系统架构评估中所普遍关注的质量属性，并详细阐述每种质量属性的具体含义。

系统架构评估中普遍关注的质量属性包括：

1) 性能

性能是指系统的响应能力，即需要多长时间才能对某个事件做出响应，或者在某段事件内系统所能处理的事件个数。经常用单位事件内所处理事务的数量或系统完成某个事务处理所需的时间来对性能进行定量表示。

2) 可靠性

可靠性是软件系统在应用或者系统错误面前，在意外或者错误使用的情况下维持软件系统的

功能特性的基本能力。

3) 可用性

可用性是系统能够正常运行的时间比例。经常用两次故障之间的时间长度或在出现故障时系统能够恢复正常的速度来表示。

4) 安全性

安全性是指系统在向合法用户提供服务的同时能够阻止非授权用户使用的企图或拒绝服务的能力。

5) 可修改性

可修改性是指能够快速地以较高的性能价格比对系统进行变更的能力，包括可维护性、可扩展性、结构重构、可移植性。

6) 功能性

功能性是系统所能完成所期望的工作的能力。一项任务的完成需要系统中许多或大多数构件的相互协作。

7) 可变性

可变性是指体系结构经扩充或变更而成为新体系结构的能力。

8) 互操作性

互操作性是指作为系统组成部分的软件不是独立存在的，经常与其他系统或自身环境相互作用。如程序和用其他编程语言编写的软件系统的交互作用就是互操作性的问题。

3. 分析软件系统架构评估种现阶段主要评估方法

业界已开发出多种软件架构评估的方法，按基于的技术手段来看，可以分为三类：基于调查问卷或检查表的方式、基于场景的方式和基于度量的方式。

1) 基于调查问卷或检查表的方式：该方式的关键是要设计好问卷或检查表，它充分利用系统相关人员的经验和知识，获得对架构的评估。其缺点是在很大程度上依赖于评估人员的主观推断。

2) 基于场景的方式：基于场景的方式由 SEI 首先提出并应用在架构权衡分析法 (ArchitectureTradeoffAnalysisMethod，ATAM) 和 软 件 架 构 分 析 方 法 (SoftwareArchitectureAnalysisMethod, SAAM) 中。它是通过分析软件架构对场景(也就是对系统的使用或修改活动)的支持程度，从而判断该架构对这一场景所代表的质量需求的满足程度。

(1) 架构权衡分析方法 ATAM 是一种系统架构评估方法，主要在系统开发之前，针对性能、可用性、安全性和可修改性等质量属性进行评价和折中。ATAM 可以分为 4 个主要的活动阶段，包括需求收集、架构视图描述、属性模型构造和分析、架构决策与折中，整个评估过程强调以属

性作为架构评估的核心概念。

(2)SAAM 是最早形成文档并得到广泛应用的软件架构分析方法。SAAM 的主要输入是问题描述、需求说明和架构描述，其分析过程主要包括场景开发、架构描述、单个场景评估、场景交互和总体评估。

3) 基于度量的方式：制定一些定量值来度量架构，如代码行数等。要制定质量属性和度量结果之间的映射。

4. 针对实际参与的软件系统架构评估，说明所采用的评估方法，并描述其具体实施过程和效果。

试题三 论数据湖技术及其应用

论文试题三 写作要点

1. 简要叙述所参与的大数据或数据湖系统项目，并明确指出在其中承担的职务和主要工作。

2. 分析并阐述数据湖技术：

1) 大数据的由来和表现。

2) 大数据 5V 特性

3) 大数据与数据仓库

4) 数据仓库的 4 个特性

3. 分析并阐述数据湖技术：

1) 数据仓库的缺点

2) 数据湖的技术特点

3) 数据湖的概念

4) 数据库的特性

4. 分析并对比数据湖(DataLake, DL)与数据仓库(DataWarehouse, DW)

1) 数据来源

DW：从外部数据源获取

DL：既可从外部数据源获取，也可从外部应用或外部存储获取

2) 数据模式（Schema）转换时机

DW：采用 ETL

DL：既支持 ETL，也支持 ELT

3) 数据存储成本

DW: 通常采用 ODS-DWD-DW-DM-ST 五层架构, 存储成本高

DL: 通常采用原始数据和流计算, 存储成本低

4) 数据质量

DW: 依赖数据治理规范, ELT 过程损失数据准确度和精度, 质量较低

DL: 存储原始数据, 不损失数据准确度和精度, 质量高

5) 面对用户和主要支撑应用类型

DW: 联机分析(OLAP)和决策支持(DSS)

DL: 物联网、人工智能和以智能制造、智慧城市等为代表的智能赋能应用

- 结合自身参与项目的实际状况, 阐述数据湖技术在实际项目中的实施过程和应用效果。

试题四 论负载均衡技术在 Web 系统中的应用

论文试题四 写作要点

1. 简要叙述所参与管理和开发的 Web 系统项目, 并明确指出在其中承担的主要任务和开展的主要工作。

2. 分析并描述负载均衡在 Web 应用项目高并发、高性能、高可用三高架构中的意义和作用

3. 选择三种常见的通用负载均衡算法, 详细阐述负载均衡中的算法和每种算法的基本原理:

1) 轮询算法

2) 随机算法

3) 比率算法

4) 优先级算法

5) 最少连接数算法

6) 最快响应时间算法

4. 结合实际项目, 详细阐述实际参与项目中采用了哪个或哪几个负载均衡算法实现, 说明如何在项目中实践应用的负载均衡, 并阐述负载均衡技术在项目中的应用效果。

系统架构设计师学习 QQ 群: 231352210 软件设计师学习 QQ 群: 759713504

诸葛老师 QQ: 362842353

VIP 购买方式, 淘宝搜索: 诸葛老师系统架构设计师

全国计算机技术与软件专业技术资格（水平）考试

2020 下半年 系统架构设计师 上午试卷解析

试题 (1)

参考答案: B

答案解析: 按“先大后小”原则找出箭头表示的 11 对逻辑关系: P1P2, P1P3, P1P4, P2P3, P2P5, P3P4, P3P5, P4P5, P5P6, P5P7, P6P7, 经核对 B 项正确;

A 项中, P3P1、P4P1 等逻辑关系反了, 排除;

C 项中, 也存在逻辑关系反的情况如 P5P3、P7P5 等, 排除;

D 项中, P4P7 并不存在, 并且也有逻辑关系反的情况如 P6P5、P7P5 等, 排除。

试题 (2)

参考答案: C

答案解析: 同一进程创建的不同线程之间是相互隔离的, 因此只有 C 选项正确;

A 选项描述错误, 进程已打开的文件是能被 T1/T2/T3 共享的, 排除;

B 选项描述错误, 进程内不同的线程之间资源无法共享, 排除;

D 选项描述错误, 进程内不同的线程之间资源无法共享, 排除;

试题 (3)

参考答案: A

答案解析: $\because 300G = 300 * 1024M$

又 \because 物理块大小为 4M

$\therefore 300G$ 占用块数为: $300 * 1024 / 4 = 76800$

\therefore 32 位的计算机需要 $76800 / 32 = 2400$ 个字, 才能管理 300G 的磁盘。

试题 (4)

参考答案: D

答案解析: 实时操作系统 (Real Time Operating System, 简称 RTOS) 是指当外界事件或数据产生

时，能够接受并以足够快的速度予以处理，其处理的结果又能在规定的时间之内来控制生产过程或对处理系统做出快速响应，调度一切可利用的资源完成实时任务，并控制所有实时任务协调一致运行的操作系统。提供及时响应和高可靠性是其主要特点。

试题 (5)

参考答案: C

答案解析: 根据出生日期, 可以推出年龄, 因此“出生日期”与“年龄”是派生关系, 应予以合并。

试题 (6) (7)

参考答案: D A

答案解析: 设 X 是函数依赖集, 被 X 逻辑蕴涵的函数依赖全体构成的集合, 称为函数依赖集 X 的闭包。简言之, 闭包本质上就是这些属性能直接或间接推出的属性集的集合。

不难求出, (A) =AB, (B) =B, (AC) =ABC, (AD) =ABCDE, 因此 D 选项正确。AD 能推出 U 的所有属性, 因此 AD 即为 R 的候选关键字。

试题 (8)

参考答案: C

答案解析: 题干关键词“物理位置”推知, 只有 C 选项才符合, 其他选择均不符合。

试题 (9)

参考答案: D

答案解析: 从操作系统中去掉尽可能多的东西, 而只留下一个最小的核心, 称之为微内核。微内核技术的主要优点如下:

- .统一的接口, 在用户态和核心态之间无需进程识别。
- .可伸缩性好, 能适应硬件更新和应用变化。
- .可移植性好, 所有与具体机器特征相关的代码, 全部隔离在微内核中, 如果操作系统要移植到不同的硬件平台上, 只需修改微内核中极少代码即可。
- .实时性好, 微内核可以方便地支持实时处理。
- .安全可靠性高, 微内核将安全性作为系统内部特性来进行设计, 对外仅使用少量应用编程接口。

.支持分布式系统，支持多处理器的体系结构和高度并行的应用程序。

微内核的缺点：一是难以进行良好的整体优化，二是进程间互相通信的开销大、内核功能代码不能被相互调用而带来服务的效率低。但总体而言，微内核在效率上的损失小于其在结构上获得的收益。

试题 (10)

参考答案: A

答案解析: 分页内存管理的地址结构为: 页号+内页地址(偏移量)。

∴页面大小为 $4KB = 2^2 * 2^{10} = 2^{12}$, 可知页内地址(偏移量)为 12 位

∴虚拟地址 0010000000000100 的前 4 位为页号 (0010, 第 2 页), 后 12 位为偏移量。查表知, 第 2 页的物理地址为 110, 而页内地址不变。

∴物理地址为: 1100000000000100, 选项 A 正确。

试题 (11)

参考答案: C

答案解析: 段页式存储管理的基本思想是将段式存储管理与分页存储管理结合起来, 段页式存储管理将作业分成若干段, 每个段分成若干页, 每段赋予一个段名, 为了实现地址转换, 必须为每个作业配置一张段表和若干张页表。内存的分配与回收以页为单位进行。作业的逻辑地址是二维的, 包括段号和段内地址, 其中段内地址又包含页号和页内地址两部分。显然, 只有 C 选项符合。

试题 (12)

参考答案: B

答案解析: 此题实际上考察的是分层架构的缺点, 分层架构缺点主要有:

- 级联修改问题。当业务流程发生变化时, 所有层都需要修改。
- 性能问题。整个系统中层层传递, 势必造成性能的下降, 同时也加大的开发的复杂度。
- 层层依赖问题。基础层(底层)被其他层依赖, 它处于最核心的位置, 一旦出错, 必然影响上层的调用, 从而可能导致系统异常, 严重的甚至崩溃。

试题 (13)

参考答案: D

答案解析: 挖矿，是指利用电脑硬件计算出比特币的位置并获取的过程。简单的说，挖矿就是一个记账的过程，矿工是记账员，区块链就是账本。每隔一个时间点，通过哈希（Hash）算法，将过去一段时间内未经比特币全网认可的交易信息收集、整理，对全网所有节点进行广播，最终被所有节点承认并得到比特币的过程。比特币系统会在系统节点上生成一个随机代码，互联网中的所有计算机都可以去寻找此代码，谁找到此代码，就会产生一个区块，随即得到一个比特币，这个过程就是人们常说的挖矿。计算这个随机代码需要大量的 GPU 运算，于是矿工们采购海量显卡用以更快速的获得比特币获利。同时，挖矿是发行新币的唯一方式，对于发行新币、维系比特币支付功能、保证系统安全，有着举足轻重的影响。挖矿本身并不能防止双花（同一笔数字资产因不当操作被重复使用的情况）攻击。由于挖矿越来越难，耗电量也越来越大，我国已明文禁止。

试题 (14)

参考答案: C

答案解析: resolv.conf 是多种操作系统(如 Linux、Unix、Mac 等)的域名系统解析器 (DNSResolver) 的配置文件。

试题 (15)

参考答案: D

答案解析: 造成网络延迟的因素有很多，比如物理线路的带宽、网络之间的跳数、排队时延、服务器的处理性能（如队列、数据的读写、磁盘 IO 等），如果不考虑网络环境，服务器的延迟的主要因素是队列延迟和磁盘 IO 延迟。

对等网络 (peer-to-peer) 中，每一个节点的地位都是对等的，既当服务器，同时也享有其他节点提供的服务，因此 P2P 网络的延迟与终端数量是有关的，排除 A 选项；由于二层交换机按 mac 地址寻址，比三层路由器寻址少一层，因而同样速率的交换机比路由器速度更快，排除 B；Internet 中服务器越多，之间的跳数越多，延迟就越高，排除 C。

试题 (16) (17)

参考答案: A C

答案解析: Linux 中 ps(processstatus)命令用于显示当前进程的状态，last 命令用于显示用户最近登录信息。netstat、iptables、top 均为系统命令，只有 Perfmon 才是可视化工具。

试题 (18) (19) (20) (21)

参考答案: B A D C

答案解析: 按照政府、公众和企业三种主体之间的相互关系, 可以将电子政务分为 G2G、G2C、C2G、G2B、B2G 五种体系。

●G2G 是指政府与政府之间的电子政务, 即上下级政府、不同地方政府和不同政府部门之间实现的电子政务活动。

●G2C 是政府通过电子网络系统为公民提供各种服务。

●C2G 是居民对政府的活动除了包括个人应向政府缴纳的各种税款和费用, 按政府要求应该填报的各种信息和表格, 以及缴纳各种罚款等外, 更重要的是开辟居民参政、议政的渠道, 使政府的各项工作不断得以改进和完善。政府需要利用这个渠道来了解民意, 征求群众意见, 以便更好地为人民服务。此外, 报警服务 (盗贼、医疗、急救、火警等) 即在紧急情况下居民需要向政府报告并要求政府提供的服务, 也属于这个范围。

●G2B 指政府与企业之间的电子政务, 也是 G2G、G2B 和 G2C 电子政务模式的基础。G2B 电子政务主要是利用 Intranet 建立起有效的行政办公和企业管理体系, 为提高政府工作效率。

●B2G 指的是企业对政府的电子政务体系, 主要是企业从政府获得订单合同, 其实际应用为政府电子采购。

试题 (22) (23)

参考答案: A C

答案解析: 软件系统的文档可以分为用户文档和系统文档两类。用户文档主要描述系统功能和使用方法, 并不关心这些功能是怎样实现的; 系统文档描述系统设计、实现和测试等方面的内容。

试题 (24) (25)

参考答案: A C

答案解析: 软件需求开发的最终文档经过评审批准后, 则定义了开发工作的需求基线 (baseline)。这个基线在客户和开发者之间构筑了计划产品功能需求和非功能需求的一个约定 (agreement)。需求约定是需求开发和需求管理之间的桥梁。

试题 (26) (27) (28)**参考答案:** D C C

答案解析: 软件过程是制作软件产品的一组活动以及结果，这些活动主要由软件人员来完成，软件活动主要包括如下。

- 软件描述。必须定义软件功能以及使用的限制。
- 软件开发。也就是软件的设计和实现，软件工程人员制作出能满足描述的软件。
- 软件有效性验证。软件必须经过严格的验证，以保证能够满足客户的需求。
- 软件进化。软件随着客户需求的变化不断改进。

试题 (29) (30)**参考答案:** A B

答案解析: 对应软件开发过程的各种活动，软件开发工具有需求分析工具、设计工具、编码与排错工具、测试工具等。

按描述需求定义的方法可将需求分析工具分为基于自然语言或图形描述的工具和基于形式化需求定义语言的工具。

试题 (31) (32)**参考答案:** B D

答案解析: 软件设计包括四个既独立又相互联系的活动，即数据设计、软件结构设计、人机界面设计和过程设计，这四个活动完成以后就得到了全面的软件设计模型。

- 结构设计：定义软件系统各主要部件之间的关系。
- 数据设计：将模型转换成数据结构的定义。好的数据设计将改善程序结构和模块划分，降低过程复杂性。
- 人机界面设计：软件内部，软件和操作系统间以及软件和人之间如何通信。
- 过程设计：系统结构部件转换成软件的过程描述。

试题 (33) (34)

参考答案: A C

答案解析: 信息隐蔽是开发整体程序结构时使用的法则, 即将每个程序的成分隐蔽或封装在一个单一的设计模块中, 定义每一个模块时尽可能少地显露其内部的处理。在设计时首先列出一些可能发生变化的因素, 在划分模块时将一个可能发生变化的因素隐蔽在某个模块的内部, 使其他模块与这个因素无关。在这个因素发生变化时, 只需修改含有这个因素的模块, 而与其他模块无关。信息隐蔽原则对提高软件的可修改性、可测试性和可移植性都有重要的作用。

试题 (35)

参考答案: B

答案解析: 如果把软件系统看成构件的集合, 那么从构件的外部形态来看, 构成一个系统的构件可分为 5 类:

- 独立而成熟的构件。独立而成熟的构件得到了实际运行环境的多次检验, 该类构件隐藏了所有接口, 用户只需用规定好的命令进行使用。例如, 数据库管理系统和操作系统等。
- 有限制的构件。有限制的构件提供了接口, 指出了使用的条件和前提, 这种构件在装配时, 会产生资源冲突、覆盖等影响, 在使用时需要加以测试。例如, 各种面向对象程序设计语言中的基础类库等。
- 适应性构件。适应性构件进行了包装或使用了接口技术, 处理了不兼容性、资源冲突等, 可以直接使用。这种构件可以不加修改地使用在各种环境中, 例如 ActiveX 等。
- 装配的构件。装配的构件在安装时, 已经装配在操作系统、数据库管理系统或信息系统不同层次上, 使用胶水代码 (gluecode) 就可以进行连接使用。目前一些软件商提供的大多数软件产品都属这一类。
- 可修改的构件。可修改的构件可以进行版本替换。如果对原构件修改错误、增加新功能, 可以利用重新“包装”或写接口来实现构件的替换。这种构件在应用系统开发中使用得比较多。

试题 (36) (37) (38)

参考答案: A B D

答案解析: 中间件是一种独立的系统软件或服务程序, 中间件位于操作系统之上, 管理计算资源

和网络通信，实现应用之间的互操作。具体来说，中间件的基本功能包括以下 6 个方面：

- 负责客户机与服务器之间的连接和通信，客户机与应用层之间的高效率通信机制。
- 提供应用层不同服务之间的互操作机制，应用层与数据库之间的连接和控制机制。
- 提供一个多层次架构的应用开发和运行的平台，一个应用开发框架，支持模块化的应用开发。
- 屏蔽硬件、操作系统、网络和数据库的差异。
- 提供应用的负载均衡和高可用性、安全机制与管理功能，以及交易管理机制，保证交易的一致性。
- 提供一组通用的服务去执行不同的功能，避免重复的工作和使应用之间可以协作。

试题（39）（40）

参考答案: B C

答案解析: 常见的软件开发模型有瀑布模型、演化模型、原型模型、螺旋模型、喷泉模型、V 模型和基于可重用构件的模型、RAD 模型、统一过程、敏捷方法等。螺旋模型将整个开发流程分为目标设定、风险分析、开发和有效性验证、评审四个过程；构件是可以独立部署的单元，通过复用构件可以提高软件的可靠性和易维护性。

试题（41）

参考答案: B

答案解析: 敏捷方法的核心思想主要有下面三点：

- 敏捷方法是适应型，而非可预测型。与传统方法不同，敏捷方法拥抱变化，也可以说它的初衷就是适应变化的需求，利用变化来发展、改变、完善自己。
- 敏捷方法是以人为本，而非以过程为本。传统方法以过程为本，强调充分发挥人的特性，不去限制它，强调面对面沟通。并且软件开发在无过程控制和过于严格繁琐的过程控制中取得一种平衡，以保证软件的质量。
- 迭代增量式的开发过程。敏捷方法以原型开发思想为基础，采用迭代增量式开发，发行版本小型化。它根据客户需求的优先级和开发风险，制定版本发行计划，每一发行版都是在前一成功发行版的基础上进行功能需求扩充，最后满足客户的所有功能需求。

试题 (42) (43)

参考答案: A C

答案解析: 当前流行的自动化测试工具主要使用脚本技术来生成测试用例。脚本是一组测试工具执行的指令集合，其作用是通过回放的方式来模拟手工测试所执行的操作，生成的脚本必须是可读、可编辑的，并且应提供控制指令的支持，使工具能够复用所编写的脚本。好的脚本应该编写注释、功能独立，结构清晰、可读，文档完整。

脚本的基本结构主要有以下 5 种：

- 线性脚本。线性脚本是录制手工测试的测试用例时得到的脚本，这些脚本是未做修改的。
- 结构化脚本。结构化脚本具有各种逻辑结构，包括选择型结构、分支结构、循环迭代结构，而且具有函数调用功能。结构化脚本具有很好的可用性和灵活性，易于维护。
- 共享脚本。共享脚本是指一个脚本可以被多个测试用例使用，即脚本语言允许一个脚本调用另一个脚本。
- 数据驱动脚本。数据驱动脚本是指将测试输入存储在独立的数据文件中，而不是脚本中。这样，脚本可以针对不同的数据输入实现多个测试用例。
- 关键字驱动脚本。关键字驱动脚本是数据驱动脚本的逻辑扩展，它用测试文件描述测试用例，它说明测试用例做什么，而不是如何做。关键字驱动脚本允许使用描述性的方法，只需要提供测试用例的描述，即可生成测试用例。

试题 (44) (45) (46) (47)

参考答案: A B B A

答案解析: 考虑体系结构时，重要的是从不同的视角 (perspective) 来检查，这促使软件设计师考虑体系结构的不同属性。例如，展示功能组织的静态视角能判断质量特性，展示并发行为的动态视角能判断系统行为特性。选择的特定视角或视图也就是逻辑视图、进程视图、实现视图和配置视图。使用逻辑视图来记录设计元素的功能和概念接口，设计元素的功能定义了它本身在系统中的角色，这些角色包括功能性能等。

试题 (48) (49) (50)

参考答案: B A B

答案解析: 敏感点 (sensitivitypoint) 和权衡点 (tradeoffpoint)。敏感点和权衡点是关键的体系结构决策。敏感点是一个或多个构件 (和/或构件之间的关系) 的特性。研究敏感点可使设计人员或分析员明确在搞清楚如何实现质量目标时应注意什么。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。例如, 改变加密级别可能会对安全性和性能产生非常重要的影响。提高加密级别可以提高安全性, 但可能要耗费更多的处理时间, 影响系统性能。如果某个机密消息的处理有严格的时间延迟要求, 则加密级别可能就会成为一个权衡点。

试题 (51) (52) (53)

参考答案: A B A

答案解析: 针对二层 C/S 体系结构的缺点, 三层 C/S 体系结构应运而生。在三层 C/S 体系结构中, 增加了一个应用服务器。三层 C/S 体系结构是将应用功能分成表示层、功能层和数据层三个部分。表示层是应用的用户接口部分担负与应用逻辑间的对话功能。它用于用户从工作站输入的数据, 并显示应用输出的数据。为使用户能直观地进行操作, 一般要使用图形用户界面 (GraphicUserInterface, GUI), 在变更用户界面时, 只需改写显示控制和数据检查程序, 而不影响业务逻辑。

- 功能层是应用的本体, 它负责具体的业务处理逻辑, 例如在制作订购合同时要计算合同金额。表示层和功能层之间的数据互交要尽可能简洁。例如, 用户检索数据时, 要将有关检索要求的信息一次性地传送给功能层, 检索结果数据也由功能层一次性地传送给表示层。
- 数据层通常是数据库管理系统, 负责管理对数据库数据的读写。数据库系统必须能迅速执行大量数据的更新和检索。
- 三层 C/S 的解决方案对这三层进行明确分割, 不同层构件相互独立, 层间的接口简洁, 适合复杂事务处理。

试题 (54) (55)

参考答案: A C

答案解析: 设计模式主要用于得到简洁灵活的系统设计, GoF 的书中共有 23 个设计模式, 这些模式可以按两个准则来分类; 一是按设计模式的目的划分, 可分为创建型、结构型和行为型三种模式; 二是按设计模式的范围划分, 即根据设计模式是作用于类还是作用于对象来划分, 可以把

设计模式分为类设计模式和对象设计模式。

试题 (56) (57) (58)

参考答案: B C A

答案解析: AbstractFactory 模式在不指定具体类的情况下, 这种模式为创建一系列相关或相互依赖的对象提供了一个接口。根据给定的相关抽象类, AbstractFactory 模式提供了从一个相匹配的具体子类集创建这些抽象类的实例的方法。

Builder 模式将复杂对象的构建与其表示相分离, 这样相同的构造过程可以创建不同的对象。通过只指定对象的类型和内容, Builder 模式允许客户端对象构建一个复杂对象。客户端可以不受该对象构造的细节的影响。这样通过定义一个能够构建其他类实例的类, 就可以简化复杂对象的创建过程。

Prototype 模式允许对象在不了解要创建对象的确切类以及如何创建等细节的情况下创建自定义对象。使用 Prototype 实例, 便指定了要创建的对象类型, 而通过复制这个 Prototype, 就可以创建新的对象。

试题 (59) (60) (61) (62) (63)

参考答案: A D B C D

答案解析: 软件的质量属性反映软件产品某一方面质量的特征或特性。常见的质量属性有:

- .性能(Performance)
- .安全性(Security)
- .可用性 (Availability)
- .可扩展性 (Scalability)
- .易用性(Usability)
- .可靠性(Reliability)
- .可移植性(Portability)
- .可修改性(Modifiability)
- .可维护性(Maintainability)

本题中“网站在并发用户数量 10 万的负载情况下, 用户请求的平均响应时间应小于 3 秒”属于性能, 通常可采用提高计算效率、减少计算开销、引入并发控制、控制资源使用、资源调度、负载均衡等架构策略实现该属性; “主站宕机后, 系统能够在 10 秒内自动切换至备用站点并恢复

正常运行”属于可用性，通常可采用 Ping / Echo、心跳、异常检测、主动冗余、被动冗余、检查点等架构策略实现该属性；“系统完成上线后，少量的外围业务功能和界面的调整与修改不超过 10 人·月”属于可修改性。

试题 (64)

参考答案: A

答案解析: SYNFlooding 攻击利用了 TCP 协议实现上的一个缺陷，通过向网络服务所在端口发送大量的伪造源地址的攻击报文，就可能造成目标服务器中的半连接队列被占满，从而阻止其他合法用户进行访问。这种攻击早在 1996 年就被发现，但至今仍然显示出强大的生命力。

很多操作系统，甚至防火墙、路由器都无法有效地防御这种攻击，而且由于它可以方便地伪造源地址，追查起来非常困难。

试题 (65)

参考答案: B

答案解析: Kerberos 是一个用于鉴定身份 (authentication) 的协议，它采取对称密钥加密 (symmetric-keycryptography)，这意味着密钥不会在网络上传输。在 Kerberos 中，未加密的密码 (unencryptedpassword) 不会在网络传播，因此攻击者无法通过嗅探网络来偷取用户的密码。Kerberos 协议中有三个角色，分别是：

- 客户端 (client)：发送请求的一方
- 服务端 (Server)：接收请求的一方
- 密钥分发中心 (KeyDistributionCenter, KDC)，而密钥分发中心一般又分为两部分，分别是：

①AS (AuthenticationServer)：认证服务器，专门用来认证客户端的身份并发放客户用于访问 TGS 的 TGT (票据授予票据)

②TGS (TicketGrantingTicket)：票据授予服务器，用来发放整个认证过程以及客户端访问服务端时所需的服务授予票据 (Ticket)

Kerberos 认证流程可简化为两大步：1) 客户端向 KDC 请求获取想要访问的目标服务的服务授予票据 (Ticket)；2) 客户端拿着从 KDC 获取的服务授予票据 (Ticket) 访问相应的网络服务。

试题 (66) (67)

参考答案: D D

答案解析: 本题中, 研发人员参与开发的该软件是职务作品, 该软件著作权属于公司。

由于该研发人员公开发表了算法和部分程序代码, 导致公司丧失了该软件的商业秘密。

试题 (68)

参考答案: B

答案解析: 按照《中华人民共和国著作权法》第二十二条作者的署名权、修改权、保护作品完整权的保护期不受限制。

试题 (69)

参考答案: D

答案解析: 选项 D 本质上是用蒙特卡洛方法计算三维体积。

在计算机上利用伪随机数, 很容易取得正立方体(x, y, z)在区间[-1,1]内均匀分布的随机点, 也很容易判断该点是否位于 V 内。对大量的随机点, 很容易统计在该正立方体中的随机点位于 V 中的比例。该比例值的 8 倍就近似地等于 V 的体积。

试题 (70)

参考答案: A

答案解析: ①根据题意计算出每台电视机的可变成本:

$$4000 * 10000 / 25000 = 1600 \text{ (元)}$$

②设盈亏平衡点为 X 台, 则有:

$$\text{收益} = \text{单价} * \text{数量} * (100\% - \text{税率}) = 2500 * X * (100\% - 16\%)$$

$$\text{成本} = \text{固定成本} + \text{可变成本} = 250 * 10000 + 1600 * X$$

收益=成本时, 盈亏平衡, 因此 $2500 * X * (100\% - 16\%) = 250 * 10000 + 1600 * X$ ③解上述方程得, $X=5000$ 。

试题 (71) (72) (73) (74) (75)

参考答案: D A B A B

答案解析: 参考译文: 系统设计的目的是确定一种应用架构, 该架构定义了用于构建所提议的信息系统的技术。该任务通过分析最初在需求分析期间创建的数据模型和流程模型来完成。数据流

图用于在网络上建立物理流程和数据存储。为了完成这个活动，分析人员可能包含许多系统设计人员和系统用户，他们参与其中，帮助处理业务数据、流程和定位问题。此任务的关键输入是从各种来源获取的事实、建议和意见，以及决策分析阶段批准的系统建议。

- (71) A. 物理模型 B. 原型系统 C. 数据库模式 D. 应用架构
- (72) A. 需求分析 B. 问题分析 C. 因果分析 D. 决策分析
- (73) A. 实体关系图 B. 物理 DFD C. DFD D. 数据库物理模型
- (74) A. 系统用户 B. 系统分析员 C. 系统所有者 D. 项目经理
- (75) A. 系统架构 B. 系统建议 C. 技术模型 D. 业务过程

2020 下半年 系统架构设计师 下午试卷 I 解析

试题一 (25 分)

【问题 1】

参考答案: 根据该系统的功能需求，采用仓库架构风格更合适。

- (1) 数据统一保存在中央数据仓库，数据处理流程相对独立，支持交互式处理。
- (2) 管道过滤器风格容易添加新的过滤器，可扩展性好。
- (3) 仓库风格不支持并行，效率低。
- (4) 仓库风格容错性和健壮性好。

答案解析: 管道过滤器架构风格由一系列处理单元（过滤器）组成，每个单元的输出是下一个单元的输入。过滤器负责处理数据，管道负责数据传输。如 linux/unix 中的 shell、传统的编译器就是管道过滤器风格。优点：①高内聚，过滤器是执行特定功能的处理服务，具有较强的内聚性；②低耦合，过滤器之间仅通过管道通信；③可重用，支持过滤器的重用；④能简单地实现并发或顺序系统；⑤可扩展性，容易添加新的过滤器；⑥灵活性，过滤器功能可重新定义，管道线路可改变。缺点：①管道中数据传输需要通用的标准；②难以支持基于事件的交互。

仓库风格是一种数据共享风格，主要有两类，一是传统的数据库，另一是黑板。数据库系统是由输入流中的事件来驱动信息处理，把执行结构存储到中央数据单元。黑板则由中央数据单元的当前状态来驱动系统运行，用来解决状态冲突并处理可能存在的不确定性知识源。黑板常用信号处理，如语音识别、模式识别、机器翻译、句法分析等。优点：①便于多客户共享大量数据，不必关心数据的产生、有谁提供、如何提供等；②便于将构件作为知识源添加到系统中来；③解

解决问题的多方法性; ④具有可修改性和可维护性; ⑤有可重用的知识源; ⑥支持容错性和健壮性。

缺点: ①对共享数据结构, 不同知识源要达成一致; ②需要同步机制、加锁机制来保障数据的完整性和一致性, 增大了系统设计的复杂度; ③测试困难; ④缺少对并行机的支持, 效率低; ⑤开发成本高。

题干的系统功能包括代码编辑、语法高亮显示、代码编译、系统调试、代码仓库管理等。这些功能相对独立, 代码通过中央数据单元存储, 各个功能彼此之间不依赖, 采用仓库风格更合适。

【问题 2】

参考答案: (1) 安全性 (2) 可修改性 (3) g (4) i (5) f (6) j

答案解析: 架构评估是软件开发过程的重要环节, 在架构评估中的质量属性有: 性能、可用性、可修改性、安全性、可靠性、易用性。质量属性效用树 (UtilityTree) 是对质量属性进行分类、权衡、分析的架构分析工具, 它主要关注系统的性能、可用性、可修改性和安全性这四个方面的质量属性。本题中, (b)(g)属于性能; (c)(i)属于安全性; (d)(f)属于可用性; (h)(j)属于可修改性。

试题二 (25 分)

【问题 1】

参考答案/答案解析: 逻辑模型设计也称为逻辑结构设计, 其任务是将概念模型转化为某个特定的 DBMS 上的逻辑模型。设计逻辑结构时, 首先为概念模型选定一个合适的逻辑模型 (例如, 关系模型、网状模型或层次模型), 然后将其转化为由特定 DBMS 支持的逻辑模型, 最后对逻辑模型进行优化。逻辑设计的目的是将概念设计阶段设计好的 E-R 图转换为与选用的具体机器上的 DBMS 所支持的数据模型相符合的逻辑结构。

该包裹单包含三类实体: 收件人 (主键: 电话)、寄件人 (主键: 电话)、包裹单 (主键: 包裹单编号)。

【问题 2】

参考答案/答案解析: 某个实体类型中所有实体同时也是另一个实体类型中的实体, 此时称前一实体为子类实体, 后一实体为超类实体。

本题的超类实体为: 包裹单 (包裹单编号、收件人、寄件人)

【问题 3】

参考答案/答案解析: 实体的某个属性可以从其他属性或其他数据推导出来, 那么这个属性就是派生属性。本题中, 资费总计=资费+挂号费+保价费+回执费, 因此资费总计是一个派生属性。

试题三 (25 分)

【问题 1】

参考答案/答案解析: 存在的难点有:

①描述语言方面, 需求由于更加贴近用户, 其所用的描述语言可能会不够正规和标准, 可能存在语义含糊不清、甚至歧义等情况; 而软件架构所用的描述语言需要正规和标准, 不允许存在含糊、歧义等情况。

②非功能性需求描述方面, 需求阶段可能不太关注, 或者描述的比较笼统; 而架构阶段, 则需要明确量化。

③需求和架构的一致性方面, 需求与架构无法做到完全的一致和一一映射, 例如某个需求在架构上是需要通过多种技术来实现, 简言之, 架构提供的能力是大于需求的。

【问题 2】

参考答案/答案解析: FACE 架构的五个部分及其含义是:

①操作系统(OSS): 基础服务由操作系统提供。其余四个部分依赖于 OSS;

②I/O 服务(IOSS): 该段将接口标准化为 I/O 设备;

③平台服务(PSSS): 它提供平台服务, 例如数据服务、日志记录、健康管理(带有 GPU 接口);

④传输服务(TSS): 该段提供通信服务;

⑤可移植组件(PCS): 这提供了功能或业务逻辑, 如公共服务、可移植的组件等;

【问题 3】

参考答案/答案解析: 紧耦合主要表现为, 一个模块高度依赖于另一个模块, 当修改前者时, 必须同时需要修改后者。解决方案为: 为了提高系统的修改性和可移植性, 应避免紧耦合, 而应采用松耦合方式。

封装主要表现为: 把业务逻辑封装在本模块中, 不允许外部模块直接访问本模块的内部信息。

解决方案为: 模块设计尽量采用封装, 外部模块只能通过该模块提供的接口予以调用。

试题四 (25 分)

【问题 1】

参考答案: (1) a (2) b, g (3) c, d (4) f (5) e

答案解析:

redis 支持丰富的数据类型如下, 其中, 前 5 种常见, 后 4 种为新版本增加。

●string: 最基本的数据类型, 二进制安全的字符串, 可以包含任何数据(图片或序列化对象), 最大 512M。应用场景: 缓存、计数器、共享用户 session、分布式锁、分布式系统的全局序列号等;

●list: 按照添加顺序保持顺序的字符串列表, 应用场景: 栈、队列、阻塞队列、最新列表等;

●set: 无序的字符串集合, 不存在重复的元素, 应用场景: 用户标签、好友/关注/粉丝/感兴趣的人集合、随机展示、黑/白名单、抽奖小程序等;

●sortedset: 已排序的字符串集合, 不允许重复, 应用场景: 标签、共同好友/喜好、统计网站的独立 IP、统计点赞/取消点赞、排行榜;

●hash: key-value 对的一种集合, 特别适合用于存储对象, 应用场景: 存储对象、电商购物车等;

●bitmap: 更细化的一种操作, 以 bit 为单位;

●hyperloglog: 基于概率的数据结构。为 V2.8.9 新增;

●geo: 地理位置信息储存起来, 并对这些信息进行操作, 为#V3.2 新增;

●stream: 流, 相当于消息队列的 topic, 为 V5.0 新增

【问题 2】

参考答案: RDB 与 AOF 相比较, 具备下述特点:

- (1) 磁盘更新频率: RDB 更新频率比 AOF 低。
- (2) 数据安全: AOF 可以保证数据不丢失, 比 RDB 更安全。
- (3) 数据一致性: RDB 间隔一段时间存储, 可能发生数据丢失和不一致; AOF 通过 append 模式写文件, 即使发生服务器宕机, 也可通过 redis-check-aof 工具解决数据一致性问题。
- (4) 重启性能: RDB 比 AOF 更高。
- (5) 数据文件大小: RDB 文件比 AOF 小。

由于 RDB 数据恢复更快, 能在最短时间内重新建立服务, 因此团队最终选择了 RDB。

答案解析: RDB 持久化是指在指定的时间间隔内将内存中的数据集快照写入磁盘。RDB 的优点:

●体积更小: RDB 文件紧凑, 全量备份, 非常适合用于进行备份和灾难恢复。

●性能更高: 生成 RDB 文件的时候, redis 主进程会 fork()一个子进程来处理所有保存工作,

主进程不需要进行任何磁盘 IO 操作。

●恢复更快: RDB 在恢复大数据集时的速度比 AOF 的恢复速度要快。RDB 的缺点: ①可能丢失数据; ②耐久性差。

AOF 是 redis 将每一个写命令都通过 write 函数追加到文件中, 通俗的理解就是日志记录。

AOF 优点:

●AOF 可以更好的保护数据不丢失, 一般 AOF 会每隔 1 秒, 通过一个后台线程执行一次 fsync 操作, 最多丢失 1 秒钟的数据。

●AOF 日志文件没有任何磁盘寻址的开销, 写入性能非常高, 文件不容易破损。

●AOF 日志文件即使过大的时候, 出现后台重写操作, 也不会影响客户端的读写。

●日志文件的命令通过非常可读的方式进行记录, 这个特性非常适合做灾难性的误删除的紧急恢复。比如某人不小心用 flushall 命令清空了所有数据, 只要这个时候后台

rewrite 还没有发生, 那么就可以立即拷贝 AOF 文件, 将最后一条 flushall 命令给删了, 然后再将该 AOF 文件放回去, 就可以通过恢复机制, 自动恢复所有数据。

AOF 缺点: ①AOF 日志文件比 RDB 快照文件更大; ②性能 AOF 比 RDB 低; ③数据恢复时无法一模一样地恢复出来。

【问题 3】

参考答案/答案解析: 有两种场景: ①不管定期删除还是惰性删除, 都是一种不完全精确的删除策略, 始终还是会存在已经过期的 key 无法被删除的场景; ②内存被挤满了;

redis 内存淘汰机制:

●volatile-lru, 针对设置了过期时间的 key, 使用 lru (LeastRecentlyUsed, 最近很少使用) 算法进行淘汰。

●allkeys-lru, 针对所有 key 使用 lru 算法进行淘汰。

●volatile-lfu, 针对设置了过期时间的 key, 使用 lfu (LeastFrequentlyUsed, 最不经常使用) 算法进行淘汰。

●allkeys-lfu, 针对所有 key 使用 lfu 算法进行淘汰。

●volatile-random, 从所有设置了过期时间的 key 中使用随机淘汰的方式进行淘汰。

- allkeys-random, 针对所有的 key 使用随机淘汰机制进行淘汰。
- volatile-ttl, 针对设置了过期时间的 key, 越早过期的越先被淘汰。
- noeviction, 不会淘汰任何数据, 当使用的内存空间超过 maxmemory 值时, 再有写请求来时返回错误。

试题五 (25 分)

【问题 1】

参考答案: (1) a、b (2) d、f (3) c、e

答案解析: 非功能性需求一般关注可用性、可修改性、性能、安全性、可测试性、易用性等。本题中 (a) (b) 属于性能; (c) (e) 属于可用性; (d) (f) 属于安全性。

【问题 2】

参考答案: (1) a (2) c (3) d (4) k (5) j (6) h (7) i

答案解析: SSM (Spring+SpringMVC+MyBatis) 框架集由 Spring、MyBatis 两个开源框架整合而成 (SpringMVC 是 Spring 中的部分内容), 常作为数据源较简单的 web 项目的框架。

【问题 3】

参考答案/答案解析: 标准的数据访问机制提供了不同标准间的数据访问机制, 它屏蔽了不同通信协议之间的差异, 为上层应用程序提供统一的访问接口, 可以容易地实现应用程序对不同总线协议设备的互操作, 使得从低层的开发中脱离出来。它独立于平台, 确保来自多个厂商的设备之间信息的无缝传输, 具有语言无关性、代码重用性、易于集成性等优点。当硬件升级或修改时只需改动硬件接口部分即可, 不会影响上层应用程序。例如, 工业自动化领域常用 OPC。

2020 下半年 系统架构设计师 下午试卷 II 解析

一、论文写作要点

1、摘要

摘要是对论文全文内容的浓缩和精华, 通过摘要, 可让读者迅速总览全文内容。一般地, 摘要至少包含三部分内容。一是项目背景简介, 如时间、项目名称、项目主要内容; 二是作者角色及工作内容; 三是项目技术简介, 如采用的主要方法、结论和成果。摘要字数为 300~300 为宜。

2、正文

正文由项目背景、理论部分、实践部分、结论四个部分组成。

1) 项目背景, 400~600 字

①项目由来/缘起/定位/目标, 主要介绍项目的前提和诞生的背景等;

②项目主要内容: 简要介绍, 不是摘要中简单的重复, 应比摘要稍详细;

具体可以参考下述 5w2h 框架

●when: 何时, 近三年项目 (体现技术先进性);

●where: 何地, 某省/某市, 注意不能实名;

●who: 甲乙方、作者, 甲方名称不能实名, 我方称“我司/我单位/我厂/我公司”;

●why: 为何立项, 项目建设目的;

●what: 项目名、项目内容、作者的工作内容;

●howmuch: 项目预算, 不宜太小, 不能太大;

●how: 作者采用的技术/方法;

注意 5W2h 内容及先后次序可根据具体情况而定, 没有的内容可以不要写。

2) 理论部分, 500~600 字

①紧扣题干要求: 一般是对题干的应答, 注意要写成一段;

②逐点回答: 分论点的基本概念、基本原理、应用场景、简单举例即可;

③惜字如金: 注意控制字数, 不要挤占实践部分;

3) 实践部分, 1000~1200 字

①结构上: 与理论部分相呼应对, 最好保持一致

②分论点标题: 最好拟一个小标题, 注意小标题的行文格式

③分论点内容: 先识别问题, 阐述 why; 然后分析/设计/解决问题, 阐述 how; 再阐述效果

4) 结尾, 300~400 字

①项目效果: 呼应论点、上线、稳定运行、获得好评、下一步计划等;

②存在的问题: 阐述小问题, 且已解决的;

注意各部分之间要有过渡, 不能太生硬。

二、试题一 知识补充

企业集成架构有三类，分别是数据集成、应用集成、企业集成。

数据集成，是为了解决不同应用和系统间的数据共享和交换需求，具体包括共享信息管理、共享模型管理和数据操作管理三个部分。共享信息管理通过定义统一的集成服务模型和共享信息访问机制，完成对集成平台运行过程中产生数据信息的共享、分发和存储管理；共享模型管理则提供数据资源配置管理、集成资源关系管理、资源运行生命周期管理及相应的业务数据协同监控管理等功能；数据操作管理则为集成平台用户提供数据操作服务，包括多通道的异构模型之间的数据转换、数据映射、数据传递和数据操作等功能服务。数据集成的模式包括：数据联邦、数据复制模式、基于结构的数据集成模式。

●应用集成，是指两个或多个应用系统根据业务逻辑的需要而进行的功能之间的互相调用和互操作。应用集成需要在数据集成的基础上完成。应用集成在底层的网络集成和数据集成的基础上实现异构应用系统之间应用层次上的互操作。它们共同构成了实现企业集成化运行最顶层集成所需要的技术层次上的支持。应用集成的模式包括：集成适配器模式、集成信使模式、集成面板模式和集成代理模式。

●企业集成，应用软件系统从功能逻辑上可以分为表示、业务逻辑和数据三个层次。其中表示层负责完成系统与用户交互的接口定义；业务逻辑层主要根据具体业务规则完成相应业务数据的处理；数据层负责存储由业务逻辑层处理所产生的业务数据，它是系统中相对稳定的部分。支持企业间应用集成和交互的集成平台通常采用多层结构，其目的是在最大程度上提高系统的柔性。在集成平台的具体设计开发中，还需要按照功能的通用程度对系统实现模块进行分层。企业集成的模式包括：前端集成模式、后端集成模式和混合集成模式。

三、试题二 知识补充

1、软件缺陷定义

软件缺陷（Defect），常常又被叫做 Bug。所谓软件缺陷，即为计算机软件或程序中存在的某种破坏正常运行能力的问题、错误，或者隐藏的功能缺陷。缺陷的存在会导致软件产品在某种程度上不能满足用户的需要。IEEE729-1983 对缺陷有一个标准的定义：从产品内部看，缺陷是软件产品开发或维护过程中存在的错误、毛病等各种问题；从产品外部看，缺陷是系统所需要实现的某种功能的失效或违背。

2、产生原因

在软件开发的过程中，软件缺陷的产生是不可避免的。那么造成软件缺陷的主要原因有哪些？从软件本身、团队工作和技术问题等角度分析，就可以了解造成软件缺陷的主要因素。软件缺陷的产生主要是由软件产品的特点和开发过程决定的。

2.1 软件本身

- 需求不清晰，导致设计目标偏离客户的需求，从而引起功能或产品特征上的缺陷。
- 系统结构非常复杂，而又无法设计成一个很好的层次结构或组件结构，结果导致意想不到的问题或系统维护、扩充上的困难；即使设计成良好的面向对象的系统，由于对象、类太多，很难完成对各种对象、类相互作用的组合测试，而隐藏着一些参数传递、方法调用、对象状态变化等方面问题。
- 对程序逻辑路径或数据范围的边界考虑不够周全，漏掉某些边界条件，造成容量或边界错误。
- 对一些实时应用，要进行精心设计和技术处理，保证精确的时间同步，否则容易引起时间上不协调，不一致性带来的问题。
- 没有考虑系统崩溃后的自我恢复或数据的异地备份、灾难性恢复等问题，从而存在系统安全性、可靠性的隐患。
- 系统运行环境的复杂，不仅用户使用的计算机环境千变万化，包括用户的各种操作方式或各种不同的输入数据，容易引起一些特定用户环境下的问题；在系统实际应用中，数据量很大。从而会引起强度或负载问题。
- 由于通信端口多、存取和加密手段的矛盾性等，会造成系统的安全性或适用性等问题。
- .新技术的采用，可能涉及技术或系统兼容的问题，事先没有考虑到。

2.2 团队工作

- 系统需求分析时对客户的需求理解不清楚，或者和用户的沟通存在一些困难。
- 不同阶段的开发人员相互理解不一致。例如，软件设计人员对需求分析的理解有偏差，编程人员对系统设计规格说明书某些内容重视不够，或存在误解。
- 对于设计或编程上的一些假定或依赖性，相关人员没有充分沟通。
- 项目组成员技术水平参差不齐，新员工较多，或培训不够等原因也容易引起问题。

2.3 技术问题

- 算法错误：在给定条件下没能给出正确或准确的结果。
- 语法错误：对于编译性语言程序，编译器可以发现这类问题；但对于解释性语言程序，只能在测试运行时发现。
- 计算和精度问题：计算的结果没有满足所需要的精度。
- 系统结构不合理、算法选择不科学，造成系统性能低下。
- 接口参数传递不匹配，导致模块集成出现问题。

2.4 项目管理的问题

- 缺乏质量文化，不重视质量计划，对质量、资源、任务、成本等的平衡性把握不好，容易挤掉需求分析、评审、测试、等时间，遗留的缺陷会比较多。
- 系统分析时对客户的需求不是十分清楚，或者和用户的沟通存在一些困难。
- 开发周期短，需求分析、设计、编程、测试等各项工作不能完全按照定义好的流程来进行，工作不够充分，结果也就不完整、不准确，错误较多；周期短，还给各类开发人员造成太大的压力，引起一些人为的错误。
- 开发流程不够完善，存在太多的随机性和缺乏严谨的内审或评审机制，容易产生问题。
- 文档不完善，风险估计不足等。

四、试题三 知识补充

云原生架构设计原则有 7 条。

1、服务化原则

通过服务化架构拆分不同生命周期的业务单元，实现业务单元的独立迭代，从而加快整体的迭代速度，保证迭代的稳定性。同时，服务化架构采用的是面向接口编程方式，增加了软件的复用程度，增强了水平扩展的能力。服务化设计原则还强调在架构层面抽象化业务模块之间的关系，从而帮助业务模块实现基于服务流量（而非网络流量）的策略控制和治理，而无须关注这些服务是基于何种编程语言开发的。通过微服务，需要将单体应用进一步拆分，按业务边界重新划分成分布式应用，使应用与应用之间不再直接共享数据，而是通过约定好的契约进行通信，以提高扩展

性.业务化垂直扩展(ScaleUp), 并将微服务数据水平扩展(ScaleOut)。

2、弹性原则

系统部署规模可以随着业务量变化自动调整大小, 而无须根据事先的容量规划准备固定的硬件和软件资源。优秀的弹性能力不仅能够改变企业的 IT 成本模式, 使得企业不用再考虑额外的软硬件资源成本支出(闲置成本), 也能更好地支持业务规模的爆发式扩张, 不再因为软硬件资源储备不足而留下遗憾。简言之, 弹性原则是指系统部署规模可以随着业务量变化自动调整大小, 而无须根据事先的容量规划准备固定的硬件和软件资源。

3、可观测性原则

强调主动性, 在云计算这样的分布式系统中, 主动通过日志、链路跟踪和度量等手段, 让一次 App 点击所产生的多次服务调用耗时、返回值和参数都清晰可见, 甚至可以下钻到每次第三方软件调用、SQL 请求、节点拓扑、网络响应等信息中。运维、开发和业务人员通过这样的观测能力可以实时掌握软件的运行情况, 并获得前所未有的关联分析能力, 以便不断优化业务的健康度和用户体验。简言之, 可观测性更强调主动性, 在云计算这样的分布式系统中, 主动通过日志、链路跟踪和度量等手段, 让一次 App 点击所产生的多次服务调用耗时, 返回值, 参数都可见。

4、韧性原则

是指当软件所依赖的软硬件组件出现异常时, 软件所表现出来的抵御能力。这些异常通常包括硬件故障、硬件资源瓶颈(如 CPU 或网卡带宽耗尽)、业务流量超出软件设计承受能力、影响机房正常工作的故障或灾难、所依赖软件发生故障等可能造成业务不可用的潜在影响因素。业务上线之后, 在运行期的大部分时间里, 可能还会遇到各种不确定性输入和不稳定依赖的情况。当这些非正常场景出现时, 业务需要尽可能地保证服务质量, 满足当前以联网服务为代表的“永远在线”的要求。因此, 韧性能力的核心设计理念是面向失败设计, 即考虑如何在各种依赖不正常的情况下, 减小异常对系统及服务质量的影响并尽快恢复正常。简言之, 韧性是指当软件所依赖的软硬件组件出现异常时, 软件所表现出来的抵御能力。韧性原则的实践与常见架构主要包括: 服务异步化能力, 服务治理能力(重试/限流/降级/熔断/反压)、主从模式、集群模式、多 AZ(AvailabilityZone, 可用区)的高可用、单元化、跨区域(Region)容灾、异地多活容灾等。

5、自动化原则

通过 IaC、GitOps、OAM、Operator 和大量自动化交付工具在 CI/CD(持续集成/持续交付)流水线中的实践, 企业可以标准化企业内部的软件交付过程, 也可以在标准化的基础上实现自动化, 即通过配置数据自描述和面向终态的交付过程, 实现整个软件交付和运维的自动。

6、零信任原则

传统安全架构认为防火墙内的一切都是安全的，而零信任模型假设防火墙边界已经被攻破，且每个请求都来自于不可信网络，因此每个请求都需要经过验证。第一不能基于 IP 配置安全策略；第二身份应该成为基础设施；第三标准的发布流水线。

7、架构持续演进原则

云原生架构本身也应该且必须具备持续演进的能力，而不是一个封闭式的，被设计后一成不变的架构。特别是在业务高速迭代，更应该考虑如何保证架构演进与业务发展之间的平衡。演进式架构是指软件开发的初始阶段，就可以通过可拓展和松耦合设计，让后续可能发生的变更更加容易，升级性重构的成本更低，并且能够发生在开发实践，发布实践和整体敏捷度等软件生命周期中的任何阶段。

五、试题四 知识补充

数据分片技术作为目前架构设计中处理大数据的一种常规手段，当前被广泛用于缓存、数据库、消息队列等中间件的开发与使用当中。数据分片概念就是按照一定的规则，将数据集划分成相对独立的数据子集，然后将数据子集分布到不同的节点上，这个节点可以是逻辑上节点，也可以是物理上的节点。数据分片需要按照一定的规则，不同的分布式场景需要设计不同的规则，但基本都遵循同样的原则：按照最主要、最频繁使用的访问方式来分片。一般有以下三种方式对数据进行分片：hash 方式、一致性 hash、按照数据范围。

1、hash 方式

通过对数据（一般为 Key 值）先进行 hash 计算再取模的方式是一种简单且使用频繁的分片方式，也就是 $\text{Hash}(\text{Key}) \% N$ ，这里的 N 大部分情况下就是我们的结点个数，这种方式相对简单实用，一般场景下能够满足我们的要求。但 Hash 取模方式主要的问题是节点扩容或缩减的时候，会产生大量的数据迁移，比如从 N 台设备扩容到 N+1 台，绝大部分的数据都要在设备间进行迁移。该种方式代码实现较为简单，既可以采用 jdk 自带的 hash 方式也可以采用其他 hash 算法，大家可以自行搜索具体实现。

2、一致性 hash

一致性 hash 是将数据按照特征值映射到一个首尾相接的 hash 环上，同时也将节点映射到这个环上。对于数据，从数据在环上的位置开始，顺时针找到的第一个节点即为数据的存储节点。这种模式的优点在于节点一旦需要扩容或缩减的时候只会影响到 hash 环上相邻的节点，不会发生大规模的数据迁移。

常规的一致性 hash 分片模式也有缺点，一致性 hash 方式在增加节点的时候，只能分摊一个

已存在节点的压力，在其中一个节点挂掉的时候，该节点的压力也会被全部转移到下一个节点。理想的目标是当节点动态发生变化时，已存在的所有节点都能参与进来，达到新的均衡状态。因此在实际开发中一般会引入虚拟节点（virtualnode）的概念，即不是将物理节点映射在 hash 环上，而是将虚拟节点映射到 hash 环上。虚拟节点的数目远大于物理节点，因此一个物理节点需要负责多个虚拟节点的真实存储。操作数据的时候，先通过 hash 环找到对应的虚拟节点，再通过虚拟节点与物理节点的映射关系找到对应的物理节点。

引入虚拟节点后的一致性 hash 需要维护的元数据也会增加：第一，虚拟节点在 hash 环上的问题，且虚拟节点的数目又比较多；第二，虚拟节点与物理节点的映射关系。但带来的好处是明显的，当一个物理节点失效时，hash 环上多个虚拟节点失效，对应的压力也就会发散到多个其余的虚拟节点，事实上也就是多个其余的物理节点。在增加物理节点的时候同样如此。除此之外，可以根据物理节点的性能来调整每一个物理节点对于虚拟节点的数量，充分、合理利用资源。

3、按数据范围（rangebased）

按数据范围分片其实也就是基于数据的业务属性进行分片，如唯一编码、时间戳、使用频率等，比如在数据库层面按 ID 范围、按时间进行分库、分表、分片，按数据被访问频率分为热点库与历史库等方法，都是按数据范围方式的具体应用。基于数据范围的分片

模式需要贴合项目实际场景，使用中需要注意以下几点：

- 分片与扩展实现比较简单，结合 ID 范围、时间结合业务自行实现即可；
- 较为依赖备份机制，否则某个节点发生异常无法迅速恢复，可用性较难保证；
- 对数据规模要有前瞻性的评估，例如按时间分片，需要考虑单位时间片内数据分布是否均匀；
- 注意各分片数据之间的性能平衡，因为在常规场景下，无论采用哪种基于数据范围的分片模式，都是距离当前时间点较近的数据被访问和操作的几率较大，所以要特别注意随着数据规模与时间的推移，历史数据规模不断膨胀导致的整体性能下降。

每种分片方式是否适用，一方面需要结合项目的实际情况与规模，另一方面也要从几个常规的维度去评估：

- 数据分片策略，也就是具体的分片方式；
- 数据分片节点的动态扩展，随着数据量的逐步增长，是否能够通过增加节点来动态扩展适应；

●数据分片节点的负载均衡，结合分片策略能否保证数据均匀的分布在各个节点上以及各个节点的负载压力是否均衡；

●数据分片的可用性，当其中一个节点产生异常，能否将该节点的数据转移到其他节点上。

系统架构设计师学习 QQ 群: 231352210 软件设计师学习 QQ 群: 759713504

诸葛老师 QQ: 362842353

VIP 购买方式，淘宝搜索：诸葛老师系统架构设计师

全国计算机技术与软件专业技术资格（水平）考试

2021 下半年 系统架构设计师 上午试卷解析

试题 (1)

参考答案: C

答案解析: 按数字“先小后大”原则找出箭头表示的 12 对逻辑关系, 记为 $\rightarrow=\{(P1, P2), (P1, P3), (P2, P3), (P2, P5), (P3, P4), (P3, P5), (P4, P6), (P5, P6), (P5, P7), (P5, P8), (P6, P8), (P7, P8)\}$, 经核对只有 C 正确。

选项 A、B、D 中均含有不存在的逻辑关系 (P6, P7), 排除。

试题 (2)

参考答案: C

答案解析: 页面大小 $4K=2^2 \times 1024=2^2 \times 2^{10}=2^{12}$, 页内地址有 12 位, 由于十六进制转换为二进制为“一分四”, 因此 1B1AH 的后三位 B1AH 即为页内地址, 逻辑地址中的 1 就是页号, 通过查表可知对应的物理块号是 6, 物理地址为 6B1AH。

试题 (3)

参考答案: C

答案解析: 直接地址索引有 5 个地址项, 对应逻辑块号 0~4。

一级间接索引有 2 个地址项, 每个地址项对应 $1KB/4B=1024B/4B=256$ 个物理块, 对应逻辑块号范围是 5~516。二级间接索引有一个地址项, 对应 $256 \times 256=65536$ 个物理块, 对应逻辑块号范围是 517 以上。

试题 (4)

参考答案: B

答案解析: R 的可用数初始为 25, 目前已经给 4 个进程分配出去 $6+4+7+6=23$ 个资源, R 剩余 2 个资源只能分配给尚需资源数最少的 P3, 使之可以顺利运行, 分配后系统也是安全的。

试题 (5) (6)

参考答案: B C

答案解析: “同一实体在不同 E-R 图中属性个数和排列次序不同” 属于结构冲突, 可以在员工实体中加入职称属性, 删除培训师实体。

试题 (7) (8)

参考答案: C C

答案解析: 自然连接是在等值连接的基础上去掉重复的属性列, 而等值连接又要求 R 和 S 有相等的属性才能连接。自然连接的结果是 6615、6151、6154。

R × S 的六个属性为 R.A、R.B、R.C、R.D、S.B 和 S.C, “ $1,5(\sigma 2=5(R \times S))$ 表示为 R 与 S 笛卡尔积之后, 先做 R.B 等于 S.B 的选择运算, 后做第一列、第五列的投影运算。

试题 (9)

参考答案: B

答案解析: SoC (System on Chip) 的定义多种多样, 由于其内涵丰富、应用范围广, 很难给出准确定义。一般说来, SoC 称为系统级芯片, 也有称片上系统, 意指它是一个产品, 是一个有专用目标的集成电路, 其中包含完整系统并有嵌入软件的全部内容。同时它又是一种技术, 用以实现从确定系统功能开始, 到软/硬件划分, 并完成设计的整个过程。A 选项描述正确。

从狭义角度讲, 它是信息系统核心的芯片集成, 是将系统关键部件集成在一块芯片上。C 选项正确。

从广义角度讲, SoC 是一个微小型系统, 如果说中央处理器(CPU)是大脑, 那么 SoC 就是包括大脑、心脏、眼睛和手的系统。国内外学术界一般倾向将 SoC 定义为将微处理器、模拟 IP (Intellectual Property) 核、数字 IP 核和存储器(或片外存储控制接口)集成在单一芯片上, 它通常是客户定制的, 或是面向特定用途的标准产品。D 选项正确。

试题 (10)

参考答案: C

答案解析: 嵌入式实时操作系统 (Embedded Real-time Operation System, RTOS)。嵌入式系统是“用于控制、监视或者辅助操作机器和设备的装置”, 兼具嵌入式 OS 和实时 OS 的特点,

不是通用 OS。

试题 (11)

参考答案: C

答案解析: NDB 需要支持可剪裁性、代码可重用。

试题 (12)

参考答案: B

答案解析: AI 芯片的关键特点有: 新型计算范式、训练和推断、大数据处理能力、数据精度、可重构能力、软件工具。A、D 选项有“低精度设计”，排除；C 选项的“专用开发工具”，排除。

试题 (13)

参考答案: A

答案解析: MAC 地址表是交换机的一个核心组成部分，交换机主要是根据 MAC 地址表来进行帧的转发的。交换机对帧的转发操作行为一共有 3 种：泛洪(Flooding)、转发(Forwarding)和丢弃 (Discarding)。

在不涉及 VLAN 的情况下，交换机的转发原理可以概括地描述为：

(1)如果进入交换机的是一个单播帧,则交换机会去 MAC 地址表中查找这个帧的目的 MAC 地址,如果查不到这个 MAC 地址,则交换机将对该帧执行泛洪操作;如果查到了这个 MAC 地址,则比较这个 MAC 地址在 MAC 地址表中对应的端口是不是这个帧进入交换机的那个端口,如果不是,则交换机将对该帧执行转发操作,如果是,则交换机将对该帧执行丢弃操作。

(2)如果进入交换机的是一个广播帧,则交换机不会去查 MAC 地址表,而是直接对该帧执行泛洪操作。

(3)如果进入交换机的是一个组播帧,则交换机的处理行为比较复杂。

B 错误, 因为交换机接收到数据帧后, 如果没有相应的表项, 交换机会采用 ARP 泛洪操作, 即广播方式进行转发。C 错误, 因为交换机通过读取输入帧中的源地址添加相应的 MAC 地址表项。D 选项错误, 交换机的 MAC 地址表项是动态增长的。所以选 A。

试题 (14)

参考答案: A

答案解析: Internet 网络核心协议是 TCP/IP 协议, 而 TCP/IP 协议是一种分组交换的协议, 分组交换也叫包 (Packet) 交换, 它将所接收的分组先进行存储再进行转发。

分组交换技术的优点:

- . 无需建立连接。
- . 线路利用率较高。
- . 简化了存储管理 (相较于报文交换)
- . 加速传输。
- . 减少了出错概率和重发数据量。 分组交换的缺点如下:
 - . 存在传输时延, 但比报文交换的传输时延要少。
 - . 需要传输额外的信息量。
 - . 当分组交换需要数据报服务时, 可能会出现失序、丢失、重复分组问题。

试题 (15)**参考答案: A**

答案解析: SDN (Software Defined Network) 软件定义网络, 本质是将网络软件化, 这样就可以像 升级、安装软件一样对网络进行修改, 方便更多的应用程序能快速部署到网络上。SDN 网络架构有三层, 分别是:

- (1) 应用层。对应用户不同的业务和应用。
- (2) 控制层。主要负责处理数据平面资源的编排, 维护网络拓扑、状态信息等。
- (3) 数据转发层。负责用户数据的转发。

试题 (16) (17)**参考答案: A D**

答案解析: Web 服务器主要性能指标有最大并发连接数、响应延迟、吞吐量、QPS (Queries Per Second)、TPS (Transactions Per Second)、RPS (Requests Per Second) 等。常见的 Web 服务器性能评测方法有基准性能测试、压力测试和可靠性测试。

试题 (18)**参考答案: A**

答案解析: 企业数字化转型的五个发展阶段依次是：初始级发展阶段（主要任务是数码化 Digitization）、单元级发展阶段（主要任务为数量化 Datafication）、流程级发展阶段（任务为数字化 Digitalization）、网络级发展段（主要任务为数模化 Digifax）、生态级发展阶段（主要任务为数用化 DigiMarketing）。

试题 (19)

参考答案: D

答案解析: 信息化的主体是全体社会成员，包括政府、企业、事业、团体和个人等。

试题 (20)

参考答案: B

答案解析: 信息化需求包含三个层次，即战略需求、运作需求和技术需求。

(1) 战略需求。组织信息化的目标是提升组织的竞争能力，为组织的可持续发展提供一个支持环境。

(2) 运作需求。组织信息化的运作需求是组织信息化需求非常重要且关键的一环，它包含三方面的内容：一是实现信息化战略目标的需要；二是运作策略的需要^三是人才培养的需要。

(3) 技术需求。由于系统开发时间过长等问题在信息技术层面上对系统的完善、升级、集成和整合提出了需求。

试题 (21)

参考答案: C

答案解析: 根据《软件产品管理办法》第四条 软件产品的开发、生产、销售、进出口等活动应遵守我国有关法律、法规和标准规范。任何单位和个人不得开发、生产、销售、进出口含有以下内容的软件产品：

- (一) 侵犯他人知识产权的；
- (二) 含有计算机病毒的；
- (三) 可能危害计算机系统安全的；
- (四) 含有国家规定禁止传播的内容的；
- (五) 不符合我国软件标准规范的。

国家准许软件的进出口；但是，法律、行政法规另有规定的除外。属于禁止出口的技术，不得出

口。无介质的软件属于技术进出口，实体进出口则直接报关，技术进出口没有“进出关境”的实体行为，不需要报关，但是为了统计和收付汇需要则需办理合同备案。

试题 (22)

参考答案: B

答案解析: CMMI 评估分为五个等级，自低向高分别为一、二、三、四、五级，分别对应初始级、管理级、定义级、定量管理级和优化级。

初始级 (Initial)：软件过程是无序的，有时甚至是混乱的，对过程几乎没有定义，成功取决于个人努力。管理是反应式的。

管理级 (Managed)：建立了基本的项目管理过程来跟踪费用、进度和功能特性。制定了必要的过程纪律，能重复早先类似应用项目取得的成功经验。

定义级 (Defined)：已将软件管理和工程两方面的过程文档化、标准化，并综合成该组织的标准软件过程。所有项目均使用经批准、剪裁的标准软件过程来开发和维护软件，软件产品的生产在整个软件过程是可见的。

定量管理级 (Q-Managed)：分析对软件过程和产品质量的详细度量数据，对软件过程和产品都有定量的理解与控制。管理有一个作出结论的客观依据，管理能够在定量的范围内预测性能。

优化级 (Optimizing)：过程的量化反馈和先进的新思想、新技术促使过程持续不断改进。在 CMMI 的已定义级中，开发过程，包括技术工作和管理工作，均已实现标准化、文档化。建立了完善的培训制度和专家评审制度，全部技术活动和管理活动均可控制，对项目进行中的过程、岗位和职责均有共同的理解。

试题 (23)

参考答案: D

答案解析: 产品配置是指一个产品在其生命周期各个阶段所产生的各种形式（机器可读或人工可读）和各种版本的文档、计算机程序、部件及数据的集合。该集合中的每一个元素称为该产品配置中的一个配置项 (Configuration Item, CI)，配置项主要有以下两大类。

- 属于产品组成部分的工作成果，如需求文档、设计文档、源代码和测试用例等。
- 属于项目管理和机构支撑过程域产生的文档，如工作计划、项目质量报告、项目跟踪报告等。这些文档虽然不是产品的组成部分，但是值得保存。

每个配置项的主要属性有名称、标识符、文件状态、版本、作者和日期等。所有配置项都被

保存在配置库里，确保不会混淆、丢失。配置项及其历史记录反映了项目产品的演化过程。置于配置管理之下的工作产品包括将交付给顾客的产品、指定的内部工作产品、采办的产品、工具和其他用于创建和描述这些工作产品的实体。

试题 (24)

参考答案: A

答案解析: 需求管理包括变更控制、版本控制、需求跟踪、需求状态跟踪。

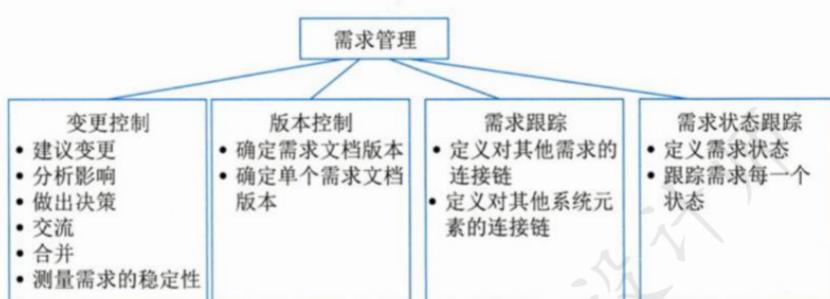


图 5-5 需求管理的主要活动

试题 (25)

参考答案: D

答案解析: 需求跟踪包括编制每个需求与系统元素之间的联系文档，这些元素包括其它需求、体系结构、设计部件、源代码模块、测试、帮助文件和文档等。

试题 (26)

参考答案: B

答案解析: 传统的软件生命期 (software life cycle) 是指软件产品从形成概念 (构思) 开始，经过定义、开发、使用和维护，直到最后被废弃（不能再使用）为止的全过程。按照传统的软件生命周期方法学，可以把软件生命周期划分为软件定义、软件开发、软件运行与维护三个阶段。

试题 (27)

参考答案: B

答案解析: 敏捷方法的核心思想主要有下面三点：

- 敏捷方法是适应型，而非可预测型。与传统方法不同，敏捷方法拥抱变化，也可以说它的

初衷就是适应变化的需求，利用变化来发展、改变、完善自己。

●敏捷方法是以人为本，而非以过程为本。传统方法以过程为本，强调充分发挥人的特性，不去限制它，强调面对面沟通。并且软件开发在无过程控制和过于严格繁琐的过程控制中取得一种平衡，以保证软件的质量。

●迭代增量式的开发过程。敏捷方法以原型开发思想为基础，采用迭代增量式开发，发行版本小型化。它根据客户需求的优先级和开发风险，制定版本发行计划，每一发行版都是在前一成功发行版的基础上进行功能需求扩充，最后满足客户的所有功能需求。

试题 (28)

参考答案: C

答案解析: RUP (Rational Unified Process) 软件开发生命周期是一个二维的软件开发模型，RUP 的 9 个核心工作流有：

●业务建模：理解待开发系统所在的机构及其商业运作，确保所有参与人员对待开发系统所在的机构有共同的认识，评估待开发系统对所在机构的影响。

●需求：定义系统功能及用户界面，使客户知道系统的功能，使开发人员理解系统的需求，为项目预算及计划提供基础。

●分析与设计：把需求分析的结果转化为分析与设计模型。

●实现：把设计模型转换为实现结果，对开发的代码做单元测试，将不同实现人员开发的模块集成为可执行系统。

●测试：检查各子系统的交互与集成，验证所有需求是否均被正确实现，对发现的软件质量上的缺陷进行归档，对软件质量提出改进建议。

●部署：打包、分发、安装软件，升级旧系统;培训用户及销售人员，并提供技术支持。

●配置与变更管理：跟踪并维护系统开发过程中产生的所有制品的完整性和一致性。

●项目管理：为软件开发项目提供计划、人员分配、执行、监控等方面的指导，为风险管理提供框架。

●环境：为软件开发机构提供软件开发环境，即提供过程管理和工具的支持。

试题 (29)

参考答案: **B**

答案解析: 在软件开发和维护过程中一个软件会有多个版本, 版本控制工具用来存储、更新、恢复和管理一个软件的多个版本。

试题 (30)

参考答案: **C**

答案解析: N-S 图, 也被称为盒图或 NS 图 (Nassi Shneiderman 图)。是结构化设计中的一种可视化建模。

HIPO 图(Hierarchy plus Input-Process-Output,HIPO)是 IBM 公司于 20 世纪 70 年代中期在层次结构图的基础上推出的一种描述系统结构和模块内部处理功能的工具。HIPO 图由层次结构图和 IPO 图两部分构成, 前者描述整个系统的设计结构以及各类模块之间的关系, 后者描述某个特定模块内部的处理过程和输入/输出关系。

顺序图是一种 UML 图, 主要用于面向对象设计。

试题 (31) (32)

参考答案: **B C**

答案解析: 耦合表示模块之间联系的程度。紧密耦合表示模块之间联系非常强, 松散耦合表示模块之间联系比较弱, 非耦合则表示模块之间无任何联系, 是完全独立的。模块的耦合类型通常分为 7 种, 根据耦合度从高到低依次排序为: 内容耦合、公共耦合、外部耦合、控制耦合、特征/标记耦合、数据耦合、非直接耦合。

内容耦合耦合性最强, 模块的独立性最弱, 因此不应该使用内容耦合。根据题干信息, 数据耦合在这里耦合性最弱, 尽量使用数据耦合。

内聚表示模块内部各成分之间的联系程度, 是从功能角度来度量模块内的联系, 一个好的内聚模块应当恰好做目标单一的一件事情。模块的内聚类型通常也可以分为 7 种, 根据内聚度从高到低依次排序为: 功能内聚、顺序内聚、通信内聚、过程内聚、瞬时/时间内聚、逻辑内聚、偶然/巧合内聚。

功能内聚内聚性最强, 模块独立性也最强。

试题 (33)

参考答案: D

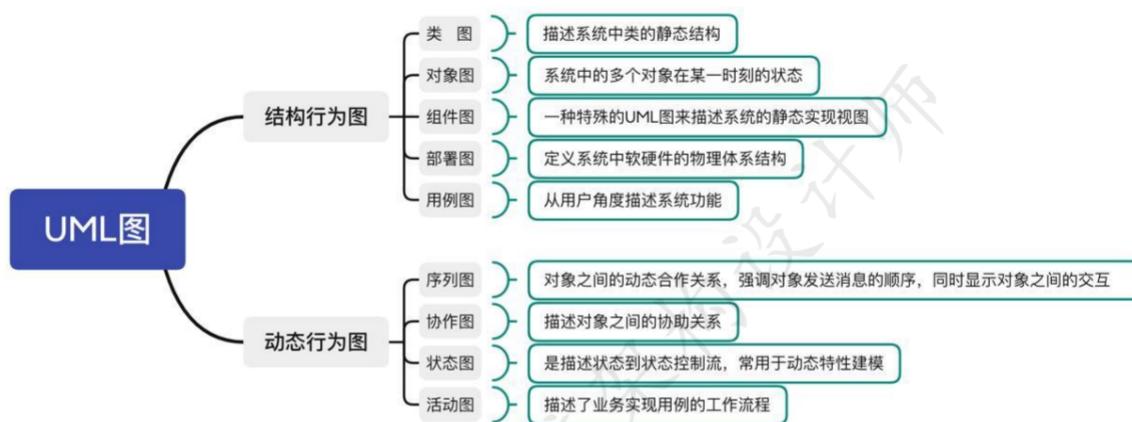
答案解析: UML 图分为结构图和行为图。

- 结构图分为类图、轮廓图、组件图、组合结构图、对象图、部署图、包图。

- 行为图又分活动图、用例图、状态机图和交互图。

交互图又分为序列图、时序图、通讯图、交互概览图。

9 种常见的 UML 图, 如下图所示:



PAD 图用于结构化设计的工具。

试题 (34)

参考答案: B

答案解析: McCabe 度量法是通过定义环路复杂度, 建立程序复杂度的度量, 它是基于一个程序模块的程序图中环路的个数。

计算 G 的环路复杂型有两种方法:

- ① $V(G)=m-n+2$ (m 值是有向弧数, 也就是箭头的个数, n 指的是结点个数);
- ② 就是闭合区域的个数+1。

本题中, 采用第二种方法相对比较简单, 即: 环路复杂度等于闭环个数+1=4。

试题 (35)

参考答案: C

答案解析: 构件的特性如下。

●独立部署单元。一个构件是独立可部署的，意味着它必须能跟它所在的环境及其他构件完全分离。因此，构件必须封装自己的全部内部特征。并且，构件作为一个部署单元，具有原子性，是不可拆分的，排除 A、B。

●作为第三方的组装单元。如果第三方厂商能将一个构件和其他构件组装在一起，那么这个构件不但必须具备足够好的内聚性，还必须将自己的依赖条件和所提供的服务说明清楚。换句话说，构件必须封装它的实现，并且只通过良好定义的接口与外部环境进行交互。

●没有（外部的）可见状态。这要求构件不能与自己的拷贝有所区别，给定一个进程（或者其他语境），至多会存在一个特定构件的拷贝，排除 D，选项 C 描述错误。
在一个特定进程中只能存在一个特定构件的拷贝。

试题 (36)

参考答案: A

答案解析: 当若干个进（线）程竞争使用资源时，如果每个进（线）程都占有了一定的资源，又申请使用已被另一个进（线）程占用、且不能抢占的资源，则所有这些进（线）程都纷纷进入阻塞状态，不能继续运行，即系统中两个或两个以上的进（线）程无限期地等待永远不会发生的条件，系统处于一种停滞状态，这种现象就称为死锁。

死锁的产生的原因之一有系统资源竞争，而多线程会争夺系统资源。

试题 (37) (38)

参考答案: B D

答案解析: 为实现对象重用，COM 支持两种形式的对象组装：包含（Containment）和聚集（Aggregation）。

包含就是一种简单的对象组装技术，其含义是一个对象拥有指向另一个对象的唯一引用。

聚集是直接把内部对象的接口引用传给外部对象的客户，而不再转发请求。

试题 (39) (40) (41)

参考答案: B A D

答案解析: 信息的完整性是指信息在传输、交换、存储和处理过程中，保持信息不被破坏或修改、不丢失和信息未经授权不能改变的特性，也是最基本的安全特征。

拒绝服务攻击由攻击者向服务器发送大量垃圾信息或干扰信息的方式，导致服务器无法向正常用户提供服务。

陷阱门是编程人员在进行程序设计时建立的，在程序正常运行时，通过某些按键和参数可以绕过正常的安全检查和错误跟踪检查。

旁路攻击会绕过对加密算法的繁琐分析，利用密码算法的硬件实现的运算中泄露的信息，如执行时间、功耗、电磁辐射等，结合统计理论快速的破解密码系统。

欺诈者采用某种手段欺骗合法的用户或系统自愿地放弃敏感信息等。

试题 (42) (43)

参考答案: A C

答案解析: 静态测试是指被测试程序不在机器上运行，而采用人工检测和计算机辅助静态分析的手段对程序进行检测。

动态测试是通过运行程序发现错误。动态测试可以采用黑盒测试和白盒测试。

白盒测试也称为结构测试，要根据程序内部逻辑结构设计测试用例，检测程序中的主要执行通路是否都能按预定要求正确工作。主要用于软件单元测试阶段。

黑盒测试也称为功能测试，不考虑程序的内部结构和处理算法，只检查软件功能是否能按照要求正常使用。主要用于集成测试、确认测试和系统测试阶段。

试题 (44) (45)

参考答案: A C

答案解析: ABSD 方法是一个自顶向下，递归细化的方法，软件系统的体系结构通过该方法得到细化，直到能产生软件构件和类。

试题 (46) (47)

参考答案: C B

答案解析: 逻辑视图：也称为设计视图，它表示了设计模型中在架构方面具有重要意义的部分，即类、子系统、包和用例实现的子集。用于建模系统的组成部分以及各组成部分之间的交互方式。通常包括类图，对象图，状态图和协作图。

进程（过程）视图：是可执行线程和进程作为活动类的建模，它是逻辑视图的一次执行实例，描述了并发与同步结构。描述了系统中的进程，当可视化系统中一定会发生的事情时，此视

图特别有用。

实现（开发）视图：对组成基于系统的物理代码的文件和构件进行建模。描述系统的各部分如何被组织为模块和组件即开发环境中软件的静态组织结构。该视图通常包含包图和组件图。

部署视图：把构件部署到一组物理节点上，表示软件到硬件的映射和分布结构。展示了抽象部分如何映射到最终部署的系统中。

用例视图：用例视图是最基本的需求分析模型。

试题 (48)

参考答案: A

答案解析: 物联网属于层次型架构，分为：

- 1) 感知层：负责信息采集和物物之间的信息传输。
- 2) 网络层：利用无线和有线网络对采集的数据进行编码、认证和传输。
- 3) 应用层：实现应用。

试题 (49) (50)

参考答案: A B

答案解析: 特定领域的架构可以分为：

- (1) 垂直域：定义了一个特定的系统族，包含整个系统族内的多个系统，结果是在该领域中可作为系统的可行解决方案的一个通用软件体系结构。
- (2) 水平域：定义了在多个系统和多个系统族中功能区域的共有部分。在子系统级上涵盖多个系统族的特定部分功能。

试题 (51) (52)

参考答案: D A

答案解析: 社保金的计算方式不是固定的，可以采用 Rule-based system（基于规则的系统）实现。基于规则的系统中的规则引擎负责对规则进行解释实现，能够将业务规则跟应用程序分离开来，减少了运维的工作。

试题 (53) (54)

参考答案: B A

答案解析: 安全性是系统向合法用户提供服务并阻止非法用户的能力。

性能是指系统的响应能力，即要经过多长时间才能对某个事件做出响应，或者在某段时间内系统所能处理的事件的个数。

可用性是指系统能够正常运行的时间比例。经常用两次故障之间的时间长度或在出现故障时系统能够恢复正常的速度来表示。

可靠性是软件系统在应用或系统错误面前，在意外或错误使用的情况下维持软件系统的功能特性的基本能力。

试题 (55)

参考答案: B

答案解析: 可用性是系统能够正常运行的时间比例。经常用两次故障之间的时间长度或在出现故障时系统能够恢复正常的速度来表示。“在系统修复后，连接于 30 秒之内恢复 … …” 这属于可用性的特征。

试题 (56) (57)

参考答案: B B

答案解析: 1、保密性。也称机密性，是不将有用信息泄漏给非授权用户的特性。可以通过信息加密、身份认证、访问控制、安全通信协议等技术实现，信息加密是防止信息非法泄露的最基本手段，主要强调有用信息只被授权对象使用的特征。

2、完整性。是指信息在传输、交换、存储和处理过程中，保持信息不被破坏或修改、不丢失和信息未经授权不能改变的特性，也是最基本的安全特征。

3、可用性。也称有效性，指信息资源可被授权实体按要求访问、正常使用或在非正常情况下能恢复使用的特性（系统面向用户服务的安全特性）。在系统运行时正确存取所需信息，当系统遭受意外攻击或破坏时，可以迅速恢复并能投入使用。是衡量网络信息系统面向用户的一种安全性能，以保障为用户提供服务。

4、可控性。指网络系统和信息在传输范围和存放空间内的可控程度。是对网络系统和信息传输的控制能力特性。

5、不可否认性。又称拒绝否认性、抗抵赖性，指网络通信双方在信息交互过程中，确信参与者本身和所提供的信息真实同一性，即所有参与者不可否认或抵赖本人的真实身份，以及提供信息的原样性和完成的操作与承诺。

试题 (58) (59)

参考答案: C C

答案解析: 在进行体系结构 (架构) 评估时, 一般首先要精确地得出具体的质量目标, 并以之作为判定该体系结构优劣的标准。我们把为得出这些目标而采用的机制叫做场景。

场景是从风险承担者的角度对与系统的交互的简短描述。在体系结构评估中, 一般采用刺激、环境和响应三方面来对场景进行描述。

刺激是场景中解释或描述风险承担者怎样引发与系统的交互部分。例如, 用户可能会激发某个功能, 维护可能会做某个更改, 测试人员可能会执行某种测试这些都属于对场景的刺激。

环境描述的是刺激发生时的情况。例如, 当前系统处于什么状态? 有什么特殊的约束条件? 系统的负载是否很大? 某个网络通道是否出现了阻塞等。

响应是指系统是如何通过体系结构对刺激作出反应的例如, 用户所要求的功能是否得到满足? 维护人员的修改是否成功? 测试人员的测试是否成功等。

试题 (60) (61)

参考答案: A C

答案解析: 敏感点是一个或多个构件 (和 / 或构件之间的关系) 的特性。改变加密级别的设计决策属于权衡点, 因为它可能会对安全性和性能产生非常重要的影响。

试题 (62) (63)

参考答案: A C

答案解析: 三层 C/S 架构将应用系统分成表示层、功能层和数据层三个部分:

(1) 表示层。表示层是系统的用户接口部分, 担负着用户与系统之间的对话功能。它用于检查用户从键盘等输入的数据, 显示输出的数据。

(2) 功能层。功能层也称为业务逻辑层, 是将具体的业务处理逻辑编入程序中。

(3) 数据层。数据层相当于二层 C/S 架构中的服务器, 负责对 DBMS 的管理和控制。

试题 (64)

参考答案: B

答案解析: 此行为侵犯了软件的著作权, 如果是在复制品上申请了相同或相似的商标则属于侵

犯商标权。

试题 (65)

参考答案: A

答案解析: 根据《中华人民共和国著作权法》第十二条规定, 改编、翻译、注释、整理已有作品而产生的作品, 其著作权由改编、翻译、注释、整理人享有, 但行使著作权时不得侵犯原作品的著作权。

一般职务作品, 作者享有著作权, 单位享有优先使用作品的权利; 特殊职务作品的著作权归单位, 作者享有署名权, 排除 B。

委托作品的著作权通过合同约定, 合同未明确或未签订的著作权属于受托人, 排除 C。

合作作品的著作权归属于所有参与人不含组织创作的人, 排除 D。

试题 (66)

参考答案: B

答案解析: 根据《中华人民共和国著作权法》第十九条 受委托创作的作品, 著作权的归属由委托人和受托人通过合同约定。合同未作明确规定或者没有订立合同的, 著作权属于受托人。

试题 (67) (68)

参考答案: A D

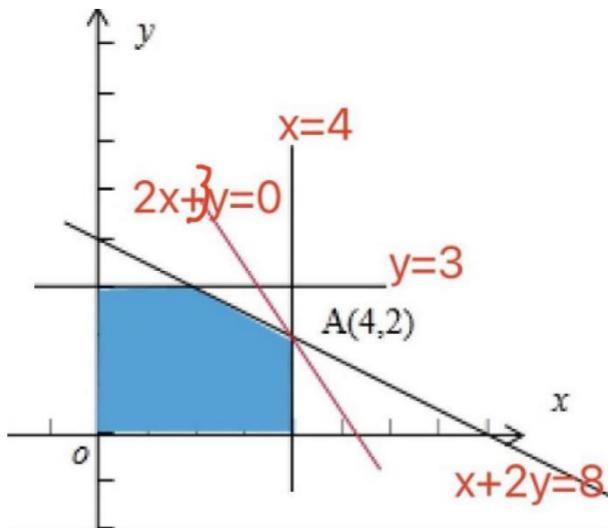
答案解析: CA 证书是一种数字证书, 既可以验证网站的身份, 又可以对敏感信息进行加密, 这样任何密码、地址或信用卡号都不会被预期接收者以外的任何人拦截或读取。部署 CA 证书后, 可以通过验证 HTTPS 中的 CA 证书信息, 确认网站的真实身份, 增强用户识别正确网站信息, 避免用户点击了假冒网站而上当受骗。通过 CA 证书加密层, 也可以对传输的数据进行加密和解密, 确保数据在传输过程中的安全, 保障数据的完整性。

用户可以使用 CA 的公钥对证书上的签名进行验证, 一旦验证通过, 该证书就被认为是有效的, 从而该验证网站的真伪, 并不能验证用户的真伪。

试题 (69)

参考答案: B

答案解析: 本题为线性规划问题。



根据题意，画出可行域，如上图阴影部分。

目标函数 $z=2x+3y$ ，变形为 $y=-2x/3+z/3$ ，则该直线过 A 点时，纵截距最大，

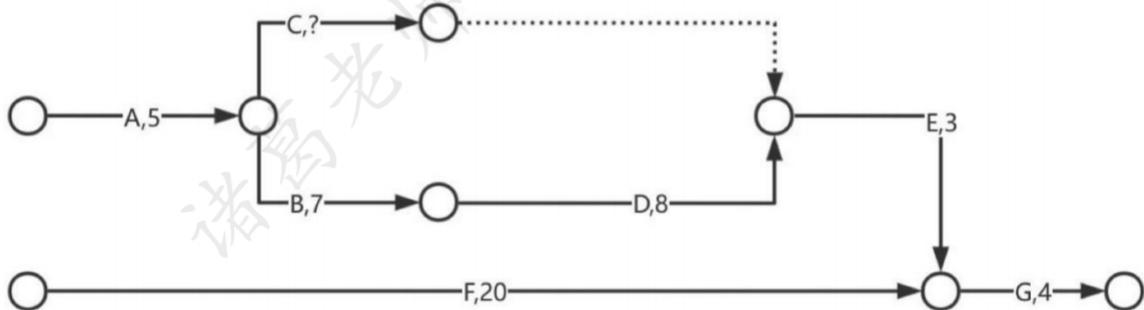
此时， $x+2y=8$ ，且 $x=4$ ，联立得 $x=4, y=2$

\therefore 目标函数 $2x+3y$ 的最大值= $2*4+3*2=14$ 。

试题 (70)

参考答案: D

答案解析: 绘制项目网络图如下所示:



作业 C 的期望时间= (乐观时间+4 × 最可能时间+保守时间) ÷ 6= 13。

项目的总工期，即关键路径为 ABDEG=5+7+8+3+4=27 (天)，活动 C 所在的路径 ACEG=5+13 (用期望值 +3+4=25 (天))，因此活动 C 比期望时间最多可拖延 27-25=2 (天)。

试题 (71) (72) (73) (74) (75)

参考答案: A D C B C

答案解析: 参考译文: 当前主流的分布式计算模型是所谓的客户/服务器计算。客户/服务器系统

是一种解决方案，其中表示层、表示逻辑层、应用逻辑、数据操作、数据层分布在客户的 PC 端与一个或多个服务器端之间。瘦客户机是一种个人计算机，它在处理器速度和内存方面不需要非常强大，因为它只向用户呈现界面。胖客户端是指在处理器速度、内存和存储容量方面通常更强大的个人计算机、笔记本电脑或工作站。数据库服务器承载一个或多个共享数据库，同时也为信息系统执行所有数据库命令和服务。web 服务器托管 Internet 或 intranet 网站，它通过向客户端返回文档和数据与客户端进行通信。

- | | | | | |
|------|-----------|-----------|------------|----------|
| (71) | A. CS 系统 | B. 客户端 | C. 服务端 | D. 数据库 |
| (72) | A. 服务端 | B. 浏览器 | C. 胖客户 | D. 瘦客户 |
| (73) | A. 云平台 | B. 集群系统 | C. 胖客户 | D. 瘦客户 |
| (74) | A. 交易服务器 | B. 数据库服务器 | C. 应用服务器 | D. 消息服务器 |
| (75) | A. 数据库服务器 | B. 消息服务器 | C. web 服务器 | D. 应用服务器 |

2021 下半年 系统架构设计师 下午试卷 I 答案解析

试题一 (25 分)

【问题 1】

参考答案: (1) 性能 (2) 可修改性 (3) e (4) j (5) h (6) i

答案解析: 架构评估是软件开发过程的重要环节，在架构评估中的质量属性有：性能、可用性、可修改性、安全性、可靠性、易用性。质量属性效用树 (Utility Tree) 是对质量属性进行分类、权衡、分析的架构分析工具，它主要关注系统的性能、可用性、可修改性和安全性这四个方面的质量属性。本题中，(b) (h) 属于安全性；(f) (g) 属于性能；(c) (e) 属于可用性；(g) (i) 属于可修改性；

【问题 2】

答案/答案解析: 管道-过滤器风格具备高内聚低耦合、支持软件重用、扩展性好、支持并发等优点，但它有编写复杂、不适合处理交互应用等缺点；

隐式调用基于事件触发思想，具备支持软件重用、改进系统方便等优点，但它有构件放弃了对系统计算的控制、事件传递中的数据交换存在问题、语义依赖于被触发事件的上下文约束等缺

点。

解释器通常包括解释引擎、代码存储区、记录解释引擎当前工作状态的数据结构、记录源代码被解释执行进度的数据结构。它含有一个虚拟机，可以仿真硬件的执行过程和一些关键应用，其缺点是执行效率比较低。优点：语法由很多类（每个规则对应一个类）表示，容易改变及扩展；缺点：如果语法规则数量太多，会增加系统复杂度，性能下降。

本题中，由于需要交互操作，显然管道-过滤器风格不合适；基于事件触发的隐式调用风格也不合适；只有解释器风格通过灵活自定义规则，具备较强的灵活性和可扩展性，适合本题中的机器学习应用。

试题二（25 分）

【问题 1】

参考答案：

- (1) 系统操作员（系统管理员）
- (2) 预约人（患者）
- (3) (a) 注册登录
- (4) - (8) (c)、(f)、(h)、(i)、(j) [注意：与次序无关]
- (9) - (12) (b)、(d)、(e)、(g) [注意：与次序无关]

答案解析：分析图 2-1，该系统主要有两类使用者，一类是医院内部的操作人员，另一类是预约人员或患者，再结合题干描述的功能介绍，不难得出两类人员所使用的功能模块。

左边是系统操作者，使用功能包括：(a)注册登录、(c)账号管理、(f)号源管理、(h)预约管理、(j)信用管理。从字面上看，含有“**管理”的功能应该属于系统操作者。

右边是预约人员（患者），使用功能包括：(a)注册登录、(b)信息浏览、(d)预约挂号、(e)查询与取消预约、(g)报告查询。

其中，(a)注册登录属于两者都需要的功能。

【问题 2】

答案/答案解析： (1) 预约人员（患者） (2) 发起预约挂号请求 (3) 显示医生出诊时段 (4) 显示是否预约成功

序列图强调交互的消息时间顺序。

协作图强调接受和发送消息的对象的结构组织，强调通信的方式。

【问题 3】

答案/答案解析: 对象模型描述系统中对象的静态结构、对象之间的关系、属性和操作，主要用对象图来实现。

动态模型描述与时间和操作顺序有关的系统特征，例如，激发事件、事件序列、确定事件先后关系的状态等，主要用状态图来实现。

功能模型描述一个计算如何从输入值得到输出值，它不考虑计算的次序，主要用 DFD 来实现。

功能模型指发生了什么，动态模型确定什么时候发生，而对象模型确定发生的客体。上述模型均可用于需求分析。

试题四 (25 分)

【问题 1】

答案/答案解析: 常见的反规范技术有增加冗余列、增加派生列、重新组表和分割表。

(1) 增加冗余列：增加冗余列是指在多个表中具有相同的列，它常用来在查询时避免连接操作。

(2) 增加派生列指增加的列可以通过表中其他数据计算生成。它的作用是在查询时减少计算量，从而加快查询速度。

(3) 重新组表指如果许多用户需要查看两个表连接出来的结果数据，则把这两个表重新组成一个表来减少连接而提高性能。

(4) 表分割有水平分割、垂直分割两种。

水平分割表，按记录进行分割，把数据放到多个独立的表中，主要用于表数据规模很大、表中数据相对独立或数据需要存放到多个介质上时使用。

垂直分割表，对表进行分割，将主键与部分列放到一个表中，主键与其它列放到另一个表中，在查询时减少 I/O 次数。

本题中用到了重新组表的方式。

【问题 2】

答案/答案解析: 解决数据不一致性问题的三种常见方法，有批处理维护、应用逻辑和触发器。

(1) 批处理维护：通过定期运行一批处理作业或存储过程对数据库进行修改，适用于对实

时性要求不高的情况。

(2) 应用逻辑: 在同一事务中对所有涉及的表进行增、删、改操作。同一逻辑必须在所有的应用中使用和维护, 容易遗漏, 特别是在需求变化时, 不易于维护。

(3) 触发器: 对数据的任何修改立即触发对数据库某些列的相应修改。触发器实时性好, 也易于维护。

【问题 3】

答案/答案解析:

(1) ZSet

Redis 数据类型:

String: 基本类型。可用于缓存层或计数器, 如视频播放量、文章浏览量等。

Hash: 代替 string 类型, 节省空间。描述用户信息较为方便。

Set: 无序集合, 每个值不能重复。可用于去重、抽奖、初始化用户池等。

List: 双向链表结构, 可以模拟栈、队列等形式。可用于回复评论、点赞。

ZSet: 有序集合、每个元素有一个分数。如首页推荐 10 个最热门的帖子。

(2) Redis 和 MySQL 数据实时同步方案有下面三种:

①引用 Mysql 的事务, 因为事务有一致性保证, 事务提交成功后再更新缓存;

②在缓存里面引用一些访问控制位, 数据库数据变化后, 同步变更对应的访问控制位, 然后从缓存查询时, 率先判断该访问控制位, 有变化就从数据库查, 无变化直接从缓存返回数据;

③三是通过数据库中间件产品保证缓存和数据库数据时时同步。

试题五 (25 分)

【问题 1】

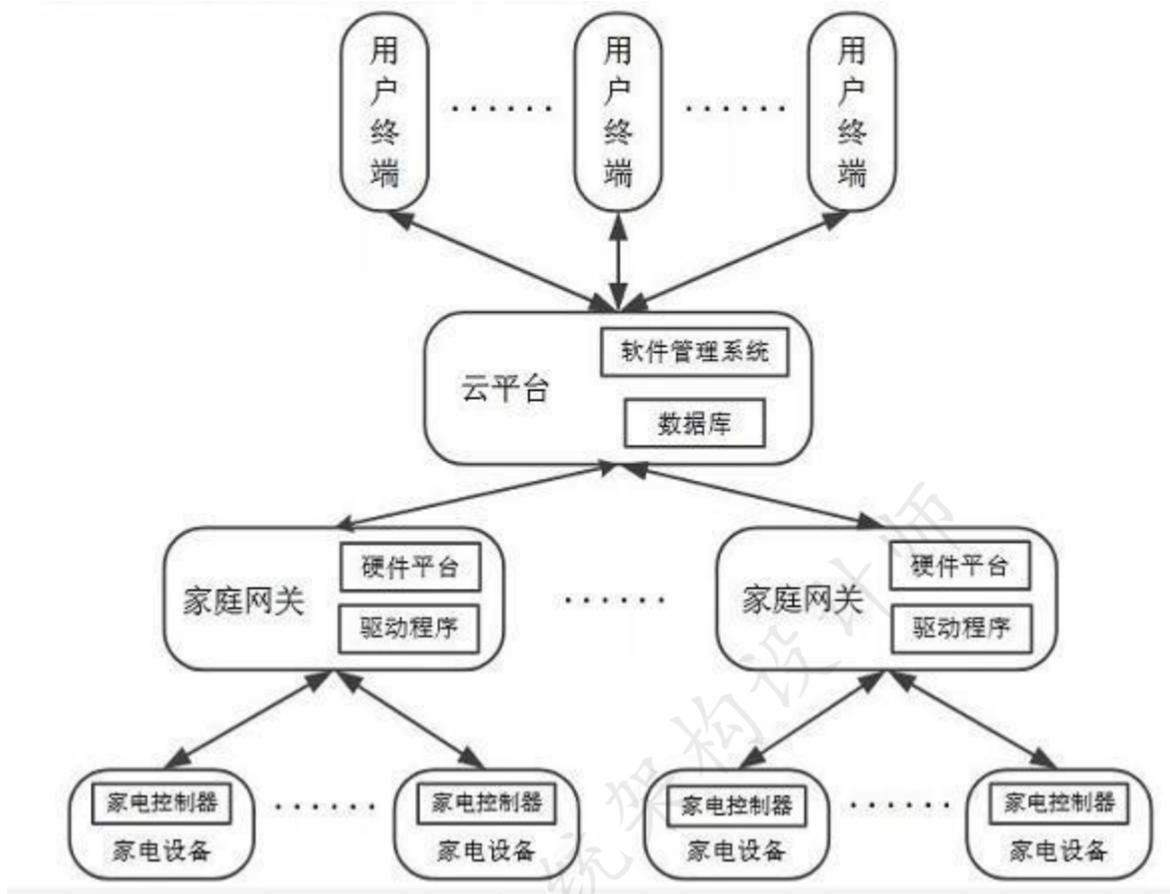
答案/答案解析: 在网关管理方面, 基于云平台的智能家居管理系统可以将分散的智能家居网关数据集中起来, 实现对智能家居网关的远程高效管理。

在数据处理方面, 云端服务器对智能家居网数据进行备份存储, 当家庭网关由于故障等原因导致数据丢失时, 可以通过云端管理系统对网关数据进行恢复, 从而提高数据的容灾性。

在系统性能方面, 基于云服务平台的智能家居管理系统将数据信息存储在云端, 减少了数据请求时间, 提高了通信效率。

【问题 2】

参考答案: (1) h (2) i (3) f (4) d (5) e (6) c



答案解析: 分析图 5-1 可知, 系统为典型的分层架构:

- (1) 层为用户交互层; (3) 层为平台层; (5) 层为通信网关层。
- (1) 层包含多种移动 app 操作系统, 推知 (2) 为鸿蒙 OS。
- (3) 层为云平台层, 包含各种软件系统、数据库等。
- (5) 层为网关层, 包含硬件系统及其驱动或通信协议。

【问题 3】

答案/答案解析: TCP 在 IP 协议提供的不可靠数据服务的基础上, 采用了重发技术, 为应用程序提供了一个可靠的、面向连接的、全双工的数据传输服务。TCP 协议一般用于传输数据量比较少, 且对可靠性要求高的场合。

UDP 是一种不可靠的、无连接的协议, 可以保证应用程序进程间的通信, 与 TCP 相比, UDP 是一种无连接的协议, 它的错误检测功能要弱得多。

该系统应采用 TCP 协议。

诸葛老师 QQ: 362842353

VIP 购买方式, 淘宝搜索: 软考诸葛老师

2021 下半年 系统架构设计师 下午试卷 II 答案解析

写作要点

试题一 论面向方面的编程技术及其应用 (AOP)

试题一 写作要点

面向方向编程 (AOP) 包括以下三个主要的步骤:

(1) 方面分解。分解需求提取出横切关注点和核心关注点。把核心模块级关注点和系统级的横切关注点进行分离。例如, 对于一个信用卡系统, 可以分解出三个关注点: 核心的信用卡处理、日志和验证。

(2) 关注点实现。各自独立地实现这些关注点, 用 OOP (面向对象的程序设计) 实现核心关注点, 用 AOP 实现横切关注点。例如, 可以用 OOP 实现信用卡处理单元, 而用 AOP 实现日志单元和验证单元。

(3) 方面的重新组合。方面集成器通过创建一个模块单元 (方面) 来制定重组的规则, 重组过程也称为编织。

AOP (Aspect-Oriented Programming, 面向方面编程), 可以说是 OOP(Object-Oriented Programming, 面向对象编程)的补充和完善。OOP 引入封装、继承和多态性等概念来建立一种对象层次结构, 用以模拟公共行为的一个集合。当我们需要为分散的对象引入公共行为的时候, OOP 则显得无能为力。也就是说, OOP 允许你定义从上到下的关系, 但并不适合定义从左到右的关系。例如日志功能, 日志代码往往水平地散布在所有对象层次中, 而与它所散布到的对象的核心功能毫无关系。对于其他类型的代码, 如安全性、异常处理和透明的持续性也是如此。这种散布在各处的无关的代码被称为横切(cross-cutting)代码, 在 OOP 设计中, 它导致了大量代码的重复, 而不利于各个模块的重用。

而 AOP 技术则恰恰相反, 它利用一种称为“横切”的技术, 剖解开封装的对象内部, 并将那些影响了多个类的公共行为封装到一个可重用模块, 并将其命名为"Aspect", 即方面。所谓“方面”, 简单地说, 就是将那些与业务无关, 却为业务模块所共同调用的逻辑或责任封装起来, 以

减少系统的重复代码，降低模块间的耦合度，并有利于未来的可操作性和可维护性。AOP 代表的是一个横向的关系，如果说“对象”是一个空心的圆柱体，其中封装的是对象的属性和行为；那么面向方面编程的方法，就仿佛一把利刃，将这些空心圆柱体剖开，以获得其内部的消息。而剖开的切面，也就是所谓的“方面”了。然后它又以巧夺天工的妙手将这些剖开的切面复原，不留痕迹。

使用“横切”技术，AOP 把软件系统分为两个部分：核心关注点和横切关注点。业务处理的主要流程是核心关注点，与之关系不大的部分是横切关注点。横切关注点的一个特点是，他们经常发生在核心关注点的多处，而各处都基本相似，比如权限认证、日志、事务处理。AOP 的作用在于分离系统中的各种关注点，将核心关注点和横切关注点分离开来。

AOP 应用程序包括以下三个主要的开发步骤：

1. 将系统需求进行功能性分解，区分出普通关注点以及横切关注点，确定哪些功能是组件语言必须实现的，哪些功能可以以 aspect 的形式动态加入到系统组件中。

2. 单独完成每一个关注点的编码和实现，构造系统组件和系统 aspect。这里的系统组件，是实现该系统的基本模块，对于 OOP 语言，这些组件可以是类，对于过程化程序设计语言，这些组件可以是各种函数和 API。系统 aspect 是指用 AOP 语言实现的将横切关注点封装成的独立的模块单元。

3. 用联接器指定的重组规则，将组件代码和 aspect 代码进行组合，形成最终系统。为达到此目的，应用程序需要利用或创造一种专门指定规则的语言，用它来组合不同应用程序片断。这种用来指定联结规则的语言可以是一种已有编程语言的扩展，也可以是一种完全不同的全新语言。

试题二 论系统安全架构设计及其应用

试题二 写作要点

鉴别（Authentication）的基本目的，就是防止其他实体占用和独立操作被鉴别实体的身份。鉴别提供了实体声称其身份的保证，只有在主体和验证者的关系背景下，鉴别才是有意义的。鉴别有两种重要的关系背景：一是实体由申请者来代表，申请者与验证者之间存在着特定的通信关系（如实体鉴别）；二是实体为验证者提供数据项来源。

鉴别的方式主要基于以下 5 种。

- (1) 已知的，如一个秘密的口令。
- (2) 拥有的，如 1C 卡、令牌等。
- (3) 不改变的特性，如生物特征。

(4) 相信可靠的第三方建立的鉴别 (递推)。

(5) 环境 (如主机地址等)。

鉴别服务分为以下阶段: 安装阶段; 修改鉴别信息阶段; 分发阶段; 获取阶段; 传送阶段; 验证阶段; 停活阶段; 重新激活阶段; 取消安装阶段。

在安装阶段, 定义申请 AI 和验证 AI。修改鉴别信息阶段, 实体或管理者申请 AI 和验证 AI 变更 (如修改口令)。在分发阶段, 为了验证交换 AI, 把验证 AI 分发到各实体 (如申请者或验证者) 以供使用。在获取阶段, 申请者或验证者可得到为鉴别实例生成特定交换 AI 所需的信息, 通过与可信第三方进行交互或鉴别实体间的信息交换可得到交换 AI。例如, 当使用联机密钥分配中心时, 申请者或验证者可从密钥分配中心得到一些信息, 如鉴别证书。在传送阶段, 在申请者与验证者之间传送交换 AI。在验证阶段, 用验证 AI 核对交换 AI。在停活阶段, 将建立一种状态, 使得以前能被鉴别的实体暂时不能被鉴别。在重新激活阶段, 使在停活阶段建立的状态将被终止。在取消安装阶段, 实体从实体集合中被拆除。

访问控制 (AccessControl) 决定开放系统环境中允许使用哪些资源、在什么地方适合阻止未授权访问的过程。在访问控制实例中, 访问可以是对一个系统 (即对一个系统通信部分的一个实体) 或对一个系统内部进行的。

ACI(访问控制信息) 是用于访问控制目的的任何信息, 其中包括上下文信息。ADI(访问控制判决信息) 是在做出一个特定的访问控制判决时可供 ADF 使用的部分 (或全部) ACI。ADF(访问控制判决功能) 是一种特定功能, 它通过对访问请求、ADI 以及该访问请求的上下文使用访问控制策略规则而做出访问控制判决。AEF(访问控制实施功能) 确保只有对目标允许的访问才由发起者执行。

涉及访问控制的有发起者、AEF、ADF 和目标。发起者代表访问或试图访问目标的人和基于计算机的实体。目标代表被试图访问或由发起者访问的, 基于计算机或通信的实体。例如, 目标可能是 OSI 实体、文件或者系统。访问请求代表构成试图访问部分的操作和操作数。

当发起者请求对目标进行特殊访问时, AEF 就通知 ADF 需要一个判决来做出决定。为了作出判决, 给 ADF 提供了访问请求 (作为判决请求的一部分) 和下列几种访问控制判决信息 (ADI)。

试题三 论企业集成平台的理解与应用

试题三 写作要点

集成平台是支持企业集成的支撑环境, 包括硬件、软件、软件工具和系统, 通过集成各种企

业应用软件形成企业集成系统。由于硬件环境和应用软件的多样性，企业信息系统的功能和环境都非常复杂，因此，为了能够较好地满足企业的应用需求，作为企业集成系统支持环境的集成平台，其基本功能要如下。

(1) 通信服务

提供分布环境下透明的同步/异步通信服务功能，使用户和应用程序无需关心具体的操作系统和应用程序所处的网络物理位置，而以透明的函数调用或对象服务方式完成它们所需的通信服务要求。

(2) 信息集成服务

为应用提供透明的信息访问服务，通过实现异种数据库系统之间数据的交换、互操作、分布数据管理和共享信息模型定义（或共享信息数据库的建立），使集成平台上运行的应用、服务或用户端能够以一致的语义和接口实现对数据（数据库、数据文件、应用交互信息）的访问与控制。

(3) 应用集成服务

通过高层应用编程接口来实现对相应应用程序的访问，这些高层应用编程接口包含在不同的适配器或代理中，被用来连接不同的应用程序。这些接口以函数或对象服务的方式向平台的组件模型提供信息，使用户在无需对原有系统进行修改（不会影响原有系统的功能）的情况下，只要在原有系统的基础上加上相应的访问接口就可以将现有的、用不同的技术实现的系统互联起来，通过为应用提供数据交换和访问操作，使各种不同的系统能够相互协作。

(4) 二次开发工具

是集成平台提供的一组帮助用户开发特定应用程序（如实现数据转换的适配器或应用封装服务等）的支持工具，其目的是简化用户在企业集成平台实施过程中（特定应用程序接口）的开发工作。

(5) 平台运行管理工具

是企业集成平台的运行管理和控制模块，负责企业集成平台系统的静态和动态配置、集成平台应用运行管理和维护、事件管理和出错管理等。通过命名服务、目录服务、平台的动态静态配置，以及其中的关键数据的定期备份等功能来维护整个服务平台的系统配置及稳定运行。

试题四 论微服务架构及其应用

试题四 写作要点

微服务好处：高异构性，高性能，高弹性，高扩展，易部署，可组合性，可替代性

微服务优点：

- 通过应用“分而治之”的原则，持续交付和部署大型，复杂的应用程序
- 通过更易于理解，开发和测试系统来提高模块化
- 通过每个微服务具有较小的代码库来降低复杂性
- 允许更新功能，而对系统的其余部分没有影响或影响极小
- 使架构变得高度可扩展
- 大大减少了破坏系统无关部分的机会
- 可以独立交付和部署服务，而不必等待整个系统发布
- 允许部署到多个云和本地基础设施环境
- 在持续发展现有系统的同时持续融入和利用最新的技术
- 使同一时间在同一系统上工作的一组开发人员间的协作更可控

基于微服务的系统设计实现：

设计原则

- 围绕业务概念建模
- 实现自动化
- 隐藏内部实现细节
- 一切去中心化
- 独立部署
- 隔离失败
- 高度可观察

微服务 RESTful API：业务服务及通用服务

服务网关 API Gateway：客户端到微服务通信

服务注册 Service Registry：微服务注册，发现中心

事件总线 Event Bus：微服务到微服务通信

安全保护 Auth Provider：认证授权提供服务

系统架构设计师学习 QQ 群: 231352210

软件设计师学习 QQ 群: 759713504

诸葛老师 QQ: 362842353

VIP 购买方式，淘宝搜索：诸葛老师系统架构设计师

全国计算机技术与软件专业技术资格（水平）考试

2022 下半年 系统架构设计师 上午试卷解析

试题 (1)

参考答案: B

答案解析: PaaS (Platform as a Service) , 平台即服务; SaaS (Software as a Service) , 软件 (应用) 即服务; IaaS (Infrastructure as a Service) , 基础设施即服务。

试题 (2)

参考答案: B

答案解析: 按数字先小后大原则找出箭头表示的 12 对逻辑关系: { (P1, P2), (P1, P3), (P1, P4), (P2, P3), (P2, P5), (P3, P4), (P3, P6), (P4, P7), (P5, P6), (P5, P8), (P6, P7), (P7, P8) }, 经核对只有 B 为正确选项。

A、C 选项均有 (P3, P5), 而图中无此逻辑, 显然不对, 排除; D 选项缺 (P1, P4), 排除。

试题 (3)

参考答案: A

答案解析: 一般情况下, 系统不会去回写用户程序、用户数据, 即便系统去写了这两部分也不会对系统本身造成较大的影响。只有系统对系统本身的目录或关键文件回写异常, 才有可能导致较大的影响, 严重的会导致系统崩溃。举个例子, linux 系统下 root 账号回写/etc/resolv.conf 异常时, 可能会导致系统 DNS 服务异常。目录其实就是文件控制块的集合, 文件控制块记录了文件各种信息, 如果出现问题影响很大。

试题 (4)

参考答案: C

答案解析: 移臂就是找柱面, 旋转则找扇区, 它们均按找最近原则调度。

首先，根据题干，距离当前移臂位置 20 号柱面最近的显然是 21 号柱面，对应的请求序列为④、⑥，排除 A、B 选项；其次，对于 16 号柱面，序列②⑧③访问扇区 3、4、6 的路基最优；因此只有 C 选项才符合。

试题 (5)

参考答案: C

答案解析: 内模式定义的是存储记录的类型、存储域的表示、存储记录的物理顺序，指引元、索引和存储路径等数据的存储组织。

试题 (6)

参考答案: D

答案解析: 动态备份是指备份期间允许对数据库进行存取或修改，即备份和用户事务可以并发执行。

试题 (7)

参考答案: C

答案解析: Armstrong 公理系统设关系模式 $R<U, F>$ ，其中 U 为属性集， F 是 U 上的一组函数依赖，那么有如下推理规则：

- ① A1 自反律：若 $Y \subseteq X \subseteq U$ ，则 $X \rightarrow Y$ 为 F 所蕴含；
- ② A2 增广律：若 $X \rightarrow Y$ 为 F 所蕴含，且 $Z \subseteq U$ ，则 $XZ \rightarrow YZ$ 为 F 所蕴含；
- ③ A3 传递律：若 $X \rightarrow Y$ ， $Y \rightarrow Z$ 为 F 所蕴含，则 $X \rightarrow Z$ 为 F 所蕴含。

根据上面三条推理规则，又可推出下面三条推理规则：

- ④ 合并规则：若 $X \rightarrow Y$ ， $X \rightarrow Z$ ，则 $X \rightarrow YZ$ 为 F 所蕴含；
- ⑤ 伪传递规则：若 $X \rightarrow Y$ ， $WY \rightarrow Z$ ，则 $XW \rightarrow Z$ 为 F 所蕴含；
- ⑥ 分解规则：若 $X \rightarrow Y$ ， $Z \subseteq Y$ ，则 $X \rightarrow Z$ 为 F 所蕴含。

试题 (8)

参考答案: B

答案解析: 首先， R 与 S 中未提及具体值'7'，因此排除 C、D 选项；其次，A 选项未考虑自然连接，排除 A；最后，只有 B 选项中 $R \times S$ 做选择时，第一列与第五列 ($R.A=S.A$)、第三

列与第六列 (R.C=S.C)，才考虑了自然连接，B 选择符合。

试题 (9)

参考答案: B

答案解析: HarmonyOS 是基于微内核的全场景分布式 OS。

试题 (10)

参考答案: A

答案解析: 最新的英伟达 GeForce RTX 4090 GPU 达到 100 TFLOPs 的算力。

试题 (11)

参考答案: A

答案解析: 常用的 AI 芯片的技术架构包括 GPU, FPGA, ASIC 和 TPU。

试题 (12)

参考答案: C

答案解析: 消息中间件，用于在两个应用程序之间，或分布式系统中发送消息，进行异步通信。

消息队列 (Message queue, 即 MQ) 是在消息的传输过程中保存消息的容器。MQ 负责两个系统之间传递消息，这两个系统可以是异构的，处于不同硬件、不同操作系统、用不同语言编写，只需要简单配置、以及简单的调用几个 MQ 的 API，就可以互相通讯，不必考虑底层系统和网络的复杂性，MQ 能够应付多种异常情况。消息中间件支持点对点模式、订阅发布模式、推拉模式。

试题 (13)

参考答案: D

答案解析: HTTP 协议是一种使用明文数据传输的网络协议，缺省端口 80。

HTTPS 协议是在 HTTP 的基础上增加了 ssl 数据加密，缺省端口 443。

试题 (14) (15)

参考答案: B B

答案解析: POP3, 邮件接收协议, 缺省端口 110; SMTP, 邮件发送协议, 缺省端口 25; HTTP, 超文本传输协议, 缺省端口 80; IMAP, 交互式邮件存取协议, 缺省端口 143。

试题 (16) (17)

参考答案: A D

答案解析: 可靠性 (Reliability) 是指产品在规定的条件下和规定的时间内完成规定功能的能力。常用的度量指标主要有故障率 (或失效率)、平均失效等待时间、平均失效间隔时间和可靠度等。

可靠度就是软件系统在规定的条件下、规定的时间内不发生失效的概率。

试题 (18)

参考答案: D

答案解析: 数据资产的特性包括: 可控制、可量化、可变现、虚拟性、共享性、时效性、安全性、交换性和规模性。

试题 (19)

参考答案: A

答案解析: DCMM 评估内容包括数据战略、数据治理、数据架构、数据应用、数据安全、数据质量、数据标准和数据生存周期。

试题 (20) (21)

参考答案: B C

答案解析: 一个完整的信息安全系统至少包含三类措施: 技术方面的安全措施, 管理方面的安全措施和相应的政策法律。信息安全的技术措施主要有: 信息加密、数字签名、数据完整性保护、身份鉴别、访问控制、数据备份和灾难恢复、网络控制技术、反病毒技术、安全审计、业务填充、路由控制机制、公证机制等。

试题 (22)

参考答案: C

答案解析: 增量式开发相比于瀑布模型的一些重要优点: 降低了适应用户需求变更的成本。重新分析和修改文档的工作量较之瀑布模型要少很多。在开发过程中更容易得到用户对于已做的开

发工作的反馈意见。用户可以评价软件的现实版本，并可以看到已经实现了多少。这比让用户从软件设计文档中判断工程进度要好很多。使更快地交付和部署有用的软件到客户方变成了可能，虽然不是所有的功能都已经包含在内。相比于瀑布模型，用户可以更早地使用软件并创造商业价值。

试题 (23)

参考答案: D

答案解析: 软件过程构架结构由四个层次组成: 方针、过程、规程和第四层的标准、规范、指南、模板、Checklist 等组成。

- 1、方针为第一层文件，它是组织标准软件的高层次的抽象描述，它反映在公司的过程改进总体方针、政策中，由公司主管副总裁批准执行。
- 2、过程为第二层文件，主要规定在项目开发中执行该过程时应当执行的各项活动及适用标准。过程定义文件及其相关文件制定必须符合方针的要求。
- 3、规程为第三层文件，是对过程某些复杂活动的具体描述。
- 4、标准、规范、指南、模板、Checklist、范例库等是对上级过程或规程提供细致的步骤、活动及说明的支持性文档，第四层的文件从属于上级过程。

试题 (24)

参考答案: B

答案解析: 业务流程图 (TFD) 是分析和描述现有系统的传统工具，是业务流程调查结果的图形化表示。实体联系图 (E-R 图) 它是描述概念世界，建立概念模型的实用工具。

数据流图 (DFD) 是表达系统内数据的流动并通过数据流描述系统功能的一种方法。

数据字典 (DD) 是在 DFD 的基础上，对 DFD 中出现的所有命名元素都加以定义，使得每个图形元素的名字都有一个确切的解释。

试题 (25)

参考答案: C

答案解析: 需求抽取和分析的过程包含：发现需求、需求分类和组织、需求优先级划分和协商、需求规格说明。

试题 (26)

参考答案: A

答案解析: 按抽象到具体的次序, 软件建模的三个层面:

- (1) 计算无关模型 (CIM) : Computational Independent Model
- (2) 平台无关模型 (PIM) : Platform Independent Model
- (3) 平台相关模型 (PSM) : Platform Dependent Model, 又称平台特定模型。

试题 (27)

参考答案: B

答案解析: 中间件是独立的系统级软件, 连接操作系统层和应用程序层, 将不同操作系统提供应用的接口标准化, 协议统一化, 屏蔽具体操作的细节, 中间件一般提供如下功能:

- 通信支持。中间件为其所支持的应用软件提供平台化的运行环境, 该环境屏蔽底层通信之间的接口差异, 实现互操作, 所以通信支持是中间件一个最基本的功能。
- 应用支持。中间件的目的就是服务上层应用, 提供应用层不同服务之间的互操作机制。
- 公共服务。公共服务是对应用软件中共性功能或约束的提取。将这些共性的功能或者约束分类实现, 并支持复用, 作为公共服务, 提供给应用程序使用。

试题 (28)

参考答案: C

答案解析: 活动图可以用于描述系统的工作流程和并发行为, 活动图其实可看作状态图的特殊形式, 活动图中一个活动结束后将立即进入下一个活动。

BPMN (Business Process Modeling Notation) 指业务流程建模与标注, 包括这些图元如何 组合成一个业务流程图 (Business Process Diagram)。Petri-Net 是所有流程定义语言之母。

试题 (29)

参考答案: D

答案解析: 软件配置管理工具是指支持完成配置项标识、版本控制、变化控制、审计和状态统计等任务的工具, 主要有下述功能:

- (1)配置支持。配置是一组有共同目的的中间软件产品, 其中每一个中间软件产品称为一个配

置项。软件配置管理支持用户建立配置项之间的各种关系，并对这些关系加以维护，维护这些关系有助于完成某些特定任务(例如 Build)和标识某一变化对整个系统开发的影响。

(2)版本控制。版本控制是软件配置管理的基本要求，它可以保证在任何时刻恢复任何一个版本、版本控制还记录每个配置项的发展历史，这样就保证了版本之间的可追踪性，也为查找错误提供了帮助，版本控制也是支持并行开发的基础。

(3)变更控制。变更控制是指在整个软件生存周期中对软件变更的控制。变更控制系统 记录每次变更的相关信息(变更的原因、变更的实施者以及变更的内容等)。这些信息有助于 追踪出现的各种问题。

(4)构造支持。软件系统往往由许多配置项构成，建立整个系统是个复杂和费时的过程，软件配置管理工具可以记录和追踪每个配置项信息，帮助用户自动和快速地建立系统，和版本控制结合在一起，可以有效地支持同时开发系统的多个版本。

(5)过程支持。过程详细描述了各种人员在整个软件生存周期中如何使用整个系统，过程控制可以保证每一步都按照正确的顺序由合适的人员实施。过程控制本来是软件开发环境中一个独立的部分，软件配置管理也开始提供这部分功能。软件配置管理工具对过程的支持还很不够，而且支持方式差别也很大，许多管理只是提供一个预先定义好的生存周期模型，并保证开发的每一步都按照这个模型规定进行。

(6)团队支持。团队支持是指多个开发人员同时开发一个软件系统。大多数软件系统都需要多个开发人员参与，有效的团队支持对开发人员是很有用的。团队支持主要包括工作区管理、并行开发管理和远程开发管理(某些软件配置管理工具还包括对开发人员支持)。

试题 (30)

参考答案: C

答案解析: OMG 在发布 2.0 修订信息需求之后， 广泛听取了来自建模工具提供商、 用户、 学术团体、 咨询机构以及其他标准化组织的 26 个响应者的建议，并于 2000 年年初发布了 UML 2.0 的 4 个组成部分的提案需求(RFP)，分别是：基础结构(Infrastructure)、上层结构(Superstructure)、对象约束语言(OCL)和图交换(Diagram Interchange)的需求。其中基础结构和上层结构构成了 UML 2.0 提案需求的主体部分。

UML 2.0 基础结构的设计目标是： 定义一个元语言的核心 InfrastructureLibrary， 通过对此核心的复用， 除了可以定义一个自展的 UML 元模型， 也可以定义其他元模型， 包括 MOF 和 CWM(Common Warehouse Model， 公共仓库模型)。

UML 2.0 上层结构的设计目标是：严格地复用基础结构 InfrastructureLibrary 包中的构造物；提高对基于构件开发和 MDA(Model Driven Architecture, 模型驱动体系结构)的支持；优化构架规约的能力；增强行为图的可伸缩性、精确性、集成性等。

试题 (31)

参考答案: B

答案解析: 2004 年 Eric Evans 发表 Domain-Driven Design – Tackling Complexity in the Heart of Software (领域驱动设计)，简称 Evans DDD。领域驱动设计分为两个阶段：

- ① 以一种领域专家、设计人员、开发人员都能理解的通用语言作为相互交流的工具，在交流的过程中发现领域概念，然后将这些概念设计成一个领域模型；
- ② 由领域模型驱动软件设计，用代码来实现该领域模型；由此可见，领域驱动设计的核心是建立正确的领域模型。

试题 (32)

参考答案: C

答案解析: SOA 还是中心化集成老架构，ESB 是 SOA 架构的中心；而微服务架构倡导去中心化的服务管理和治理，尽量不设置中心化的管理服务。

试题 (33) (34)

参考答案: D D

答案解析: UML2.0 添加了交互框(Interaction Frame)，交互框指途中的一块区域(Region)或片段(Fragment)，包含一个操作符(或称为标签)，并包含一个警戒。UML2.0 的消息类型有：简单消息、同步消息、异步消息、对象创建消息、对象销毁消息、无触发对象和无接收对象的消息。

试题 (35)

参考答案: D

答案解析: 构件作为可部署单元，具有原子性，是不可拆分的。

试题 (36)

参考答案: A

答案解析: 构件的接口是一种服务提供者对消费者的承诺, 即契约 (contract), 用契约确保一组构件之间的行为组合。

试题 (37)

参考答案: A

答案解析: EJB (Enterprise Java Beans) 是运行在独立服务器上的组件。

COM+ 是 Microsoft 组件对象模型 (COM) 和 Microsoft Transaction Server (MTS) 的演变。COM+ 可用于开发适用于 Windows 的企业范围、任务关键型分布式应用程序。Servlet (Server Applet) 是 Java Servlet 的简称, 称为小服务程序或服务连接器。ASP 即 Active Server Pages, 是 Microsoft 公司开发的服务器端脚本环境, 可用来创建动态交互式网页并建立强大的 web 应用程序。

试题 (38)

参考答案: D

答案解析: 此题采用排除法, ABC 显然都是正确的。另外, 安装新版本构件时, 有两种方式, 一种是全量构建, 另一种是增量构建, 后一种不需要停止所有已有版本构件的运行只要升级增量部分即可。

试题 (39)

参考答案: B

答案解析: 软件源码复杂度度量方法主要有三种: 代码行、Helstead 方法、McCabe 方法 (环形复杂度)。

● 代码行方法: 是一种最简单的方法, 该方法认为, 代码行越多, 软件越容易产生漏洞。

● Helstead 方法: 思路是根据程序中可执行代码行的操作符和操作数的数量来计算程序的复杂性。操作符和操作数的量越大, 程序结构就越复杂。

● McCabe 方法: 环路复杂度用来定量度量程序的逻辑复杂度。

试题 (40)

参考答案: D

答案解析: 路径覆盖实际上考虑了程序中各种判定结果的所有可能组合，是一种较强的覆盖标准。

试题 (41)

参考答案: C

答案解析: 判定表最适合描述在多个逻辑条件取值的组合所构成的复杂情况下，分别要执行哪些不同的动作。

试题 (42)

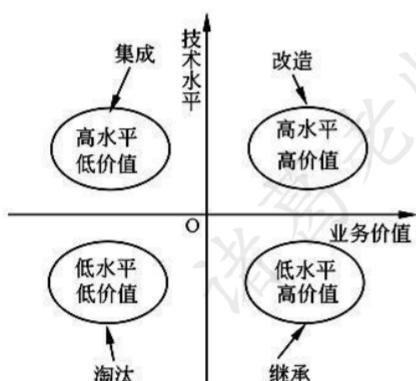
参考答案: D

答案解析: 回归测试的目的是测试软件变更之后，变更部分的正确性和对变更需求的符合性，以及软件原有的、正确的功能、性能和其他规定的要求的不损害性。

试题 (43)

参考答案: C

答案解析: 遗留系统的演化策略如下图所示。



试题 (44) (45)

参考答案: A A

答案解析: Kruchten 在 1995 年提出了一个“4+1” 的视图模型。“4+1” 视图模型从 5 个不同的视角来 描述软件架构，每个视图只关心系统的一个侧面，5 个视图结合在一起才能反映软件架构的 全部内容，其思想即为关注点分离。其中，“4”为逻辑视图、开发视图、进程视图、物理 视图；“1”为场景。

试题 (46) (47)

参考答案: B B

答案解析: ABSD 方法是体系结构驱动, 即指构成体系结构的商业、质量和功能需求的组合驱动的。ABSD 方法是一个自顶向下, 递归细化的方法, 软件系统的体系结构通过该方法得到细化, 直到能产生软件构件和类。

试题 (48) (49)

参考答案: B D

答案解析: 构件为一系列固定顺序的计算单元, 构件之间只通过数据传递交互。每个处理步骤是一个独立的程序, 每一步必须在其前一步结束后才能开始, 数据必须是完整的, 以整体的方式传递。基于规则的系统包括规则集、规则解释器、规则/数据选择器和工作内存, 一般用在人工智能领域和 DSS 中。

试题 (50) (51)

参考答案: C B

答案解析: 根据题干关键词“可复用的资产”, 推知 (50) 为“资产复用”; 根据题干关键词“规划”, 推知 (51) 为“计划复用”。

试题 (52)

参考答案: B

答案解析: 软件复用过程包含: 创建、复用、支持、管理 4 个过程。

- . 创建过程: 界定和提供可复用资产, 以满足复用者的需要;
- . 复用过程: 利用可复用资产来生产应用软件产品;
- . 支持过程: 全面支持可复用资产的获取、管理和维护工作;
- . 管理过程: 执行计划、启动、资源、跟踪, 并协调其他各个过程。

试题 (53)

参考答案: C

答案解析: 领域分析的主要目标是获得领域模型。

试题 (54) (55)

参考答案: C C

答案解析: 软件质量特性包括功能性、可靠性、易用性、效率、可维护性、可移植性等 6 个方面, 每个方面都包含若干个子特性。

- . 功能性: 适合性、准确性、互操作性、依从性、安全性;
- . 可靠性: 成熟性、容错性、易恢复性;
- . 易用性: 易理解性、易学性、易操作性;
- . 效率: 时间特性、资源特性;
- . 可维护性: 易分析性、易改变性、稳定性、易测试性;
- . 可移植性: 适应性、易安装性、遵循性、易替换性。

试题 (56) (57)

参考答案: A C

答案解析: 最常见的质量属性分别是: 可用性 (Availability)、可修改性 (Modifiability)、性能 (Performance)、安全性 (Security)、可测试性 (Testability)、易用性 (Usability)。刻画质量属性的手段由六部分组成: 刺激源、刺激、环境、制品、响应、响应度量。

试题 (58)

参考答案: C

答案解析: 敏感点是一个或多个构件的特性。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。由题干知, 改变加密级别会影响安全性和性能两个质量属性, 因此属于权衡点。

试题 (59)

参考答案: B

答案解析: 质量属性效用树结构为: 根——质量属性——属性求精 (细分) ——场景 (叶)。

试题 (60)

参考答案: D

答案解析: 在失效率为常数和修复时间很短的情况下, MTTF 和 MTBF 几乎相等。

试题 (61) (62)

参考答案: D D

答案解析: 信息安全包括的要素有:

- 1、机密性: 确保信息不暴露给未授权的实体或进程。
- 2、完整性: 只有得到允许的人才能修改数据, 并且能够判别出数据是否已被篡改。
- 3、可用性: 得到授权的实体在需要时可以访问数据, 即攻击者不能占用所有的资源而阻碍授权者的工作。
- 4、可控性: 可以控制授权范围内的信息流向及行为方式。
- 5、可审查性: 对出现的网络安全问题提供调查的依据和手段。

试题 (63)

参考答案: D

答案解析: 场景 (scenarios) 在进行体系结构评估时, 一般首先要精确地得出具体的质量目标, 并以之作为判定该体系结构优劣的标准。为得出这些目标而采用的机制做场景。场景是从风险承担者的角度对与系统的交互的简短描述。在体系结构评估中, 一般采用刺激 (stimulus)、环境 (environment) 和响应 (response) 三方面来对场景进行描述。

试题 (64)

参考答案: A

答案解析: 5G 网络的切片技术是将 5G 网络分割成多张虚拟网络, 从而支持更多的应用。就是将一个物理网络切割成多个虚拟的端到端的网络, 每个虚拟网络之间, 包括网络内的设备、接入、传输和核心网, 是逻辑独立的, 任何一个虚拟网络发生故障都不会影响到其它虚拟网络。在一个网络切片中, 至少可分为无线网子切片、承载网子切片和核心网子切片三部分。

试题 (65)

参考答案: C

答案解析: wifi 加密技术常见的有: WEP、WPA/WPA2、WPA-PSK/WPA2-PSK, 安全性依次递增。

- 1、WEP 安全加密方式

WEP(有线等效保密), 一种数据加密算法, 用于提供等同于有线局域网的保护能力。它的安全技术源自于名为 RC4 的 RSA 数据加密技术, 是无线局域网 WLAN 的必要的安全防护层。目前常见的是 64 位 WEP 加密和 128 位 WEP 加密, WEP 基本已被弃用。

2、WPA 安全加密方式

WEP 之后, 人们将期望转向了其升级后的 WPA, 与之前 WEP 的静态密钥不同, WPA 需要不断的转换密钥。WPA 采用有效的密钥分发机制, 可以跨越不同厂商的无线网卡实现应用, 其作为 WEP 的升级版, 在安全的防护上比 WEP 更为周密, 主要体现在身份认证、加密机制和数据包检查等方面, 而且它还提升了无线网络的管理能力。

3、WPA2

WPA2 是 IEEE 802.11i 标准的认证形式, WPA2 实现了 802.11i 的强制性元素, 特别是 Michael 算法被公认彻底安全的 CCMP(计数器模式密码块链消息完整码协议)讯息认证码所取代、而 RC4 加密算法也被 AES 所取代。简言之, WPA2 是 WPA 的增强版, 安全性更高。

试题 (66)

参考答案: A

答案解析: 甲完成该软件的开发后就拥有了该软件的著作权, 乙将该程序稍加修改就发布, 侵犯了甲的 软件著作权。

试题 (67)

参考答案: C

答案解析: 《中华人民共和国计算机软件保护条例》第十四条软件著作权自软件开发完成之日起产生。

试题 (68)

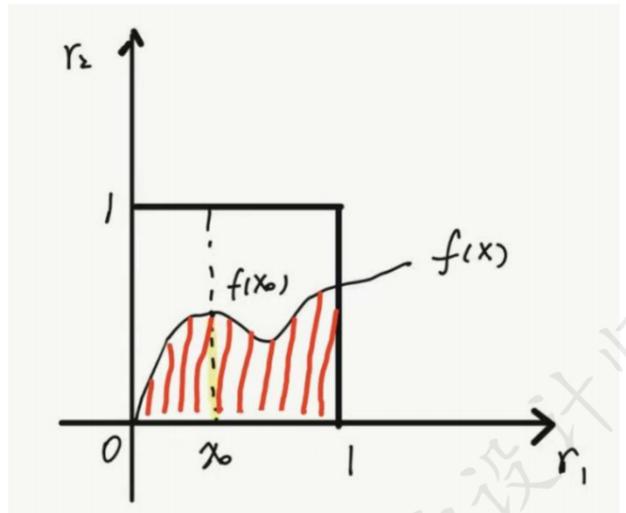
参考答案: B

答案解析: M 公司的软件产品开发完成时就自动享有软件著作权, 注册了商标就享有了商标权, 对员工签了保密协议就享有了商业秘密权。因此只有专利权未提及。

试题 (69)

参考答案: D

答案解析: 此题为蒙特卡罗方法（也称统计模拟法、统计试验法）的应用。蒙特卡罗是著名的赌城，该法通过构造一个与系统相近似的概率模型，然后在计算机上进行随机试验，通过对模型或过程的观察或抽样试验来计算这些参数或数字特征，最后求出近似值。它是一种用来模拟随机现象的数学方法，适用于对离散系统进行计算仿真试验。

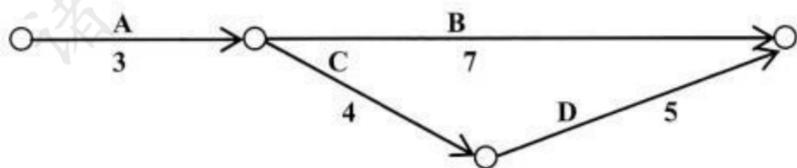


\because 随机数在 $(0, 1)$ 区间内均匀分布， M 可以视为正方形的面积又 $\because r_2 \leq f(r_1)$ ， N 可以视为函数 $f(x)$ 下方与正方形相交的面积（上图中阴影部分），而这恰好是函数 $f(x)$ 在定积分的几何含义。

试题 (70)

参考答案: D

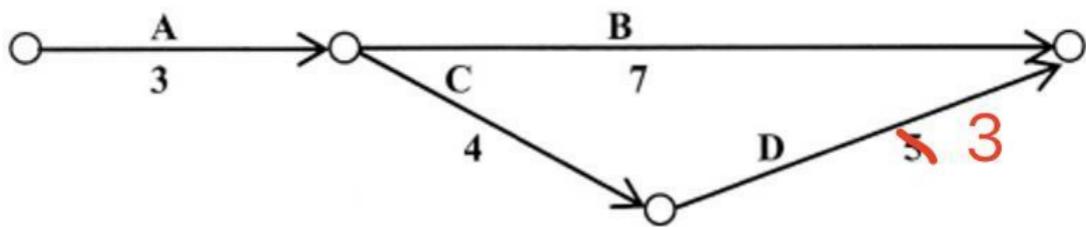
答案解析: 绘制正常进度计划网络图如下：



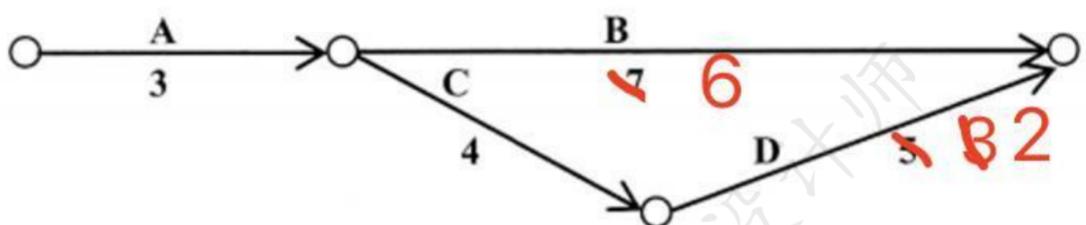
由上图可知，正常进度的关键路径为 A-C-D，总工期为 $3+4+5=12$ 天，总费用为 $12*4.5+10+15+12+8=99$ 。要想缩短总工期，必须先缩短关键路径上的作业时间。

① 由于 A-B 路径需要 10 天，先尝试对 D 缩短 2 天，总工期就可以缩短 2 天。

此时，总费用为 $10*4.5+10+15+12+14=96$ ，总工期为 10 天，关键路径有两条：A-B 和 A-C-D。

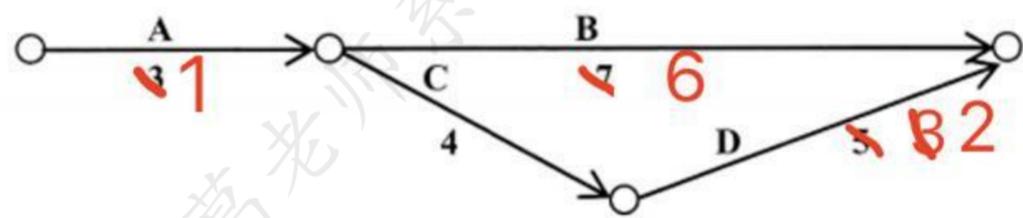


② 然后尝试对作业 B 和作业 D 各缩短 1 天。关键路径不变，总工期为 9 天。此时，总费用为 $9*4.5+10+19+12+14=95.5$ 。

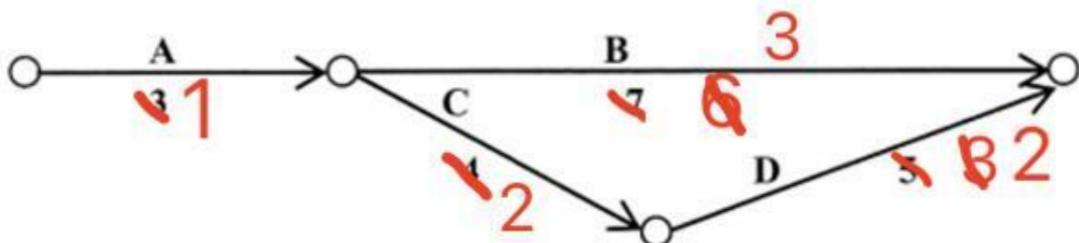


③ 再尝试对作业 A 缩短 2 天，关键路径不变，总工期为 7 天。

此时，总费用为 $7*4.5+18+19+12+14=94.5$ 。



④ B 还能缩短 3 天,C 还能缩短 2 天,总工期为 5 天。此时,总费用 $5*4.5+18+19+20+14=93.5$



至此，A、B、C、D 均已赶工到最大限度，总工期为 5 天，总费用为 93.5 万元，为最低成

本。

试题 (71) (72) (73) (74) (75)

参考答案: A B B D D

答案解析: 参考译文: 微服务是一种软件开发技术, 它主张将单个应用程序划分为一组小服务, 相互协调合作, 为用户提供终极价值。微服务架构 (71) 有许多重要的好处。首先, 它解决了业务复杂性的问题。它将原来庞大的单个应用程序分解为一组服务, 尽管功能的总数保持不变, 但应用程序已被分解为可管理的服务。单个服务的开发速度要快得多, 而且更容易理解和维护 (72); 其次, 这种架构允许每个服务由一个团队独立开发 (73), 开发人员可以自由选择任何适当的技术; 再次, 微服务架构模式允许各个服务独立部署 (74), 开发人员永远不需要协调对其服务的本地更改的部署, 这些类型的更改可以在测试后立即部署; 最后, 微服务架构允许每个服务独立扩展 (75)。

- | | | | |
|------------|-------|-------|-------|
| (71) A. 架构 | B. 软件 | C. 应用 | D. 技术 |
| (72) A. 开发 | B. 维护 | C. 使用 | D. 部署 |
| (73) A. 计划 | B. 开发 | C. 使用 | D. 部署 |
| (74) A. 使用 | B. 开发 | C. 测试 | D. 部署 |
| (75) A. 分析 | B. 使用 | C. 设计 | D. 扩展 |

2022 下半年 系统架构设计师 下午试卷 I 答案解析

试题一(共 25 分)

【问题 1】

参考答案: (1) 安全性 (2) 可修改性 (3) e (4) j (5) h (6) k

答案解析: 架构评估是软件开发过程的重要环节, 在架构评估中的质量属性有: 性能、可用性、可修改性、安全性、可靠性、易用性。质量属性效用树 (Utility Tree) 是对质量属性进行分类、权衡、分析的架构分析工具, 它主要关注系统的性能、可用性、可修改性和安全性这四个方面的质量属性。本题中, (c) (e) 属于性能; (b) (j) 属于安全性; (f) (h) 属于可用性; (g) (k) 属于可修改性。

【问题 2】

答案/答案解析: 面向对象风格通过编写新的规则实现代码，并通过应用重启或热加载添加规则，可修改性稍差；解释器风格通过编写新的规则文件，并通过导入资源文件或外部配置添加规则，可修改性较好。面向对象风格通过策略模式定义规则对象，规则以程序逻辑实现，灵活性较差，解释器风格可灵活定义规则计算表达式，灵活性更好。面向对象风格以编译后代码运算规则，性能好；而虚拟机风格需要加载规则，解析规则，规则运算，再得出结果，性能较差。

面向对象风格：效率高质量高易维护，可修改性高，灵活性稍差，性能好；解释器风格：可修改性高，个性化和灵活性强，性能较差；

由于本项目的目标是提升灵活性，并且规模不大，因此建议采用解释器风格。

试题二(共 25 分)

【问题 1】

参考答案: (1)f (2)g (3)h (4)d (5)b (6)e

层间平衡：数据流个数一致，方向一致；

图内平衡：有输入无输出的黑洞，有输出无输入的奇迹，输入不足的灰洞。

答案解析: 观察图 2-1，(1) 输入的是指标数据，输出的包含项目指标数据表，显然该功能是 f；(2) 由安全副经理操作并输入审核信息，由此可知该功能是 g；同理，(3) 的功能为 h；再来看(5)，由于输出为事故及影响参数表，可知(5) 为 b；(6) 输出有指标预警分析表，可知该项功能为 e；最后看(4)，输入是预警分析的结果，推知因为 d。

数据平衡原则有两个，一是父图与子图之间数据要平衡；二是每张子图内数据要平衡，避免奇迹（无入有出）、黑洞（有入无出）、灰洞（无法出）。

【问题 2】

参考答案: (1)项目管理员 (2)项目经理 (3)项目指标 (4)项目信息 (5)影响因素 (6)关联事故。

答案解析: 结合问题 2，安全员填报的是项目指标表，因此(3)应该就是项目指标；又因为项目指标表由项目经理确认，因而(2)为项目经理；项目管理员需要维护三类信息，即项目信息、关联事故、影响因素参数，推知(1)为项目管理员，(4)、(5)、(6)为项目信息、关联事故、影响因素参数，注意此三者次序无关。

【问题 3】

答案/答案解析: 在分析阶段，数据流图用于界定系统上下文范围和建立业务流程的加工说明，自顶向下对系统进行功能分解；指明数据在系统内移动变换；描述功能及加工规约。数据字典用

于建立业务概念有组织的集合，是模型核心库，有组织的系统相关数据元素列表，使涉众对模型中元素有共同的理解。

在设计阶段，结构化设计根据不同的数据流图类别分别做变换和事务映射来初始化系统结构图；根据数据字典中的数据存储描述来建立数据库存储设计。

试题三(共 25 分)

【问题 1】

答案/答案解析：顾名思义，心跳检测技术是节点以固定频率向其他节点发送心跳信息，表示自己存活。如果一段时间之后仍然没有收到来自此节点的心跳，就认定该节点已失效，其资源和服务就会被接管。优点是可以快速反应，缺点是容易产生误判。

超时探测技术是节点主动向被探测节点发出 PING 信号，被探测节点则在收到 PING 信号后回复一个 ECHO 信号，表示自己的健康状态良好，还可以附加一些状态信息。如果在预定的时间之后仍然收不到 ECHO 信号，则判定被探测节点失效。优点是可以获得更详细的探测结果，缺点是判断的周期较长。

【问题 2】

参考答案： (1)a (2)b (3)f (4)c (5)e (6)h (7)d (8)g

答案解析：

- (a)应用程序除零，属于核状态
- (b)看门狗是定期查看芯片内部的情况，一旦发送错误就发出重启信号，属于核状态
- (c)任务超时，属于模块状态
- (d)网络诊断故障，属于系统状态
- (e)BIT 检测故障，BIT（Build-in Test）内装测试属于模块状态
- (f)分区堆栈溢出，属于分区状态
- (g)操作系统异常，属于系统状态
- (h)模块掉电，属于模块状态

【问题 3】

答案/答案解析：数据驱动方法是一种问题求解方法。从初始的数据或观测值出发，运用启发式规则，寻找和建立内部特征之间的关系，从而发现一些定理或定律。通常也指基于大规模统计数据的自然语言处理方法。

在本题中，由于是分布式环境，需要综合多种故障信息和系统状态，依据智能决策数据库的决策策略判定，如果采用预先定制的解析模型，这个模型可能会非常复杂。因此采用数据驱动方法能通过已有的数据去训练模型，可以达到逐渐精细化，并兼容未来的变化。

试题四(共 25 分)

【问题 1】

答案/答案解析: 李工同步方案思路，更新数据时在同一事务内依次完成删除缓存，更新数据库，再写入缓存。张工异步准实时方案思路，更新数据时在同一事务内首先通过消息队列发布待更新数据的消息给缓存更新服务，再更新数据库；缓存更新服务订阅消息队列，待收到更新事件执行缓存更新。

本项目数据量极大，且性能要求高，较适合采用张工提出的异步准实时方案较好。

【问题 2】

答案/答案解析: 哈希算法是把任意长的输入通过某种哈希算法转换成固定长度的一个哈希值，简言之就是按照某种算法散列得到一个值。由于不同的输入可能会有相同的哈希值，所以不能从哈希值来确定唯一的输入值。

一致性哈希算法是一种特殊的哈希算法，它将整个哈希值空间映射成一个按顺时针方向组织的虚拟圆环，使用哈希算法算出数据哈希值，然后根据哈希值的位置沿圆环顺时针查找，将数据分配到第一个遇到的集群节点进行缓存。一致性哈希能尽可能小地改变已存在的服务请求与处理请求服务器之间的映射关系，解决了简单哈希算法在分布式哈希表中存在的动态伸缩问题。

一致性哈希算法有两大优点，1)可扩展性。一致性哈希算法保证了增加或减少服务器时，数据存储的改变最少，相比传统哈希算法大大节省了数据移动的开销；2)更好地适应数据的快速增长。

【问题 3】

答案/答案解析: 布隆过滤器本质上是一种数据结构，比较巧妙的概率型数据结构，特点是高效地插入和查询，告诉你“某个元素一定不存在或者可能存在”。布隆过滤器的原理是当一个元素被加入集合时，通过 K 个散列函数将这个元素映射成一个位数组中的 K 个点，把它们置为 1。检索时，只要看看这些点是不是都是 1 就大概知道集合中有没有它了；如果这些点有任何一个 0，则被检元素一定不在；如果都是 1，则被检元素很可能在。

优点:

- . 占用内存小
- . 增加和查询元素的时间复杂度为: $O(K)$, (K 为哈希函数的个数, 一般比较小), 与数据量大小无关
- . 哈希函数相互之间没有关系, 方便硬件并行运算
- . 布隆过滤器不需要存储元素本身, 在某些对保密要求比较严格的场合有很大优势
- . 数据量很大时, 布隆过滤器可以表示全集, 其他数据结构不能
- . 使用同一组散列函数的布隆过滤器可以进行交、并、差运算缺点:
- . 有误判率, 即存在假阳性(False Position), 不能准确判断元素是否在集合中(概率较小)
- . 不能获取元素本身
- . 一般情况下不能从布隆过滤器中删除元素
- . 如果采用计数方式删除, 可能会存在计数回绕问题

试题五(共 25 分)

【问题 1】

答案/答案解析: MQTT 是一个物联网传输协议, 它被设计用于轻量级的发布/订阅式消息传输, 旨在为低带宽和不稳定的网络环境中的物联网设备提供可靠的网络服务。MQTT 是专门针对物联网开发的轻量级传输协议。MQTT 协议针对低带宽网络, 低计算能力的设备, 做了特殊的优化, 使得其能适应各种物联网应用场景。

【问题 2】

参考答案: (1)HTTP (2)HTTP (3)MQTT (4)HTTP (5)HTTP (6)HTTP (7)端侧识别模块 (8)模型训练模块 (9)设备调度平台模块 (10)访客注册模块

答案解析: 根据 MQTT 的协议特征可知, 只有 (3) 才适合, 其他均为 HTTP。

【问题 3】

答案/答案解析: 传统云计算模型中引入边缘计算模型的优势如下。

- (1) 速度: 如果使用边缘计算, 则物联网设备将在边缘数据中心或本地处理数据。因此, 数据无需传输回中央服务器, 速度优势明显;
- (2) 安全: 边缘计算将在不同的数据中心和设备之间分配数据处理工作。黑客无法通过攻击一

台设备来影响整个网络；

(3) 可扩展性：通过购买具有足够计算能力的设备来扩展边缘网络。企业无需为其数据需求建立自己的私有或集中式数据中心；

(4) 可靠性：所有的边缘数据中心和物联网设备都位于用户附近。因此，网络中断的可能性非常小。

2022 下半年 系统架构设计师 下午试卷 II 答案解析

写作要点

一、论文写作要点

1、摘要

摘要是对论文全文内容的浓缩和精华，通过摘要，可让读者迅速总览全文内容。一般地，摘要至少包含三部分内容。一是项目背景简介，如时间、项目名称、项目主要内容；二是作者角色及工作内容；三是项目技术简介，如采用的主要方法、结论和成果。摘要字数为 300~300 为宜。

2、正文

正文由项目背景、理论部分、实践部分、结论四个部分组成。

1) 项目背景, 400~600 字

① 项目由来/缘起/定位/目标，主要介绍项目的前提和诞生的背景等；

② 项目主要内容：简要介绍，不是摘要中简单的重复，应比摘要稍详细；具体可以参考下述 5w2h 框架：

●when：何时，近三年项目（体现技术先进性）；

●where：何地，某省/某市，注意不能实名；

●who：甲乙方、作者，甲方名称不能实名，我方称“我司/我单位/我厂/我公司”；

●why：为何立项，项目建设目的；

●what：项目名、项目内容、作者的工作内容；

●how much: 项目预算，不宜太小，不能太大；

●how: 作者采用的技术/方法；

注意 5W2h 内容及先后次序可根据具体情况而定，没有的内容可以不要写。

2) 理论部分, 500~600 字

- ① 紧扣题干要求：一般是对题干的应答，注意要写成一段；
- ② 逐点回答：分论点的基本概念、基本原理、应用场景、简单举例即可；③惜字如金：注意控制字数，不要挤占实践部分；

3) 实践部分, 1000~1200 字

- ① 结构上：与理论部分相呼应用对，最好保持一致
- ② 分论点标题：最好拟一个小标题，注意小标题的行文格式
- ③ 分论点内容：先识别问题，阐述 why；然后分析/设计/解决问题，阐述 how；再阐述效果

4) 结尾, 300~400 字

- ① 项目效果：呼应论点、上线、稳定运行、获得好评、下一步计划等；
- ② 存在的问题：阐述小问题，且已解决的；

注意各部分之间要有过渡，不能太生硬。

二、试题一 知识补充

CBSD 方法使得软件开发不再一切从头开发，开发的过程就是构件组装的过程，维护的过程就是构件升级、替换和扩充的过程，其优点是提高了软件开发的效率；构件可由一方定义其规格说明，被另一方实现，然后供给第三方使用，CBSD 允许多个项目同时开发，降低了费用，提高了可维护性，可实现分步提交软件产品。

CBSD 方法由软件的需求分析和定义、架构设计、构件库的建立、应用软件构建、测试和发布五个阶段组成。

(1) 需求分析和定义：在这阶段需要重点说明系统跟曾经开发过的其他系统类似，具有大量可复用的成熟构件。

(2) 架构设计：结合实际项目，根据上一阶段获得的需求和定义提出架构模型。

(3) 构件库的建立：这是本论文主题的重点。构件的获得有四个途径：

- 从现有构件中获得符合要求的构件，直接使用或作适应性修改，得到可复用的构件。

●通过遗留工程（Legacy Engineering），将具有潜在复用价值的构件提取出来，得到可复用的构件。

●从市场上购买现成的商业构件，即 COTS（Commercial Off-The-Shelf）构件。

●开发新的符合要求的构件。

而构件库的检索方法有 3 种：基于关键字的检索、刻面检索法、超文本检索法。

(4) 应用软件构建：构建过程主要是构件的组装过程，而大致有三种技术：

●基于功能的组装技术。基于功能的组装技术采用子程序调用和参数传递的方式将构件组装起来。它要求库中的构件以子程序/过程/函数的形式出现，并且接口说明必须清晰。当使用这种组装技术进行软件开发时，开发人员首先要对新系统进行功能分解，将系统分解为强内聚、松耦合的功能模块；然后根据各模块的功能需求提取构件，进行适应性修改后，再挂接到上述功能分解框架中。

●基于数据的组装技术。基于数据的组装技术首先根据当前软件问题的核心数据结构设计出一个框架，然后根据框架中各结点的需求提取构件并进行适应性修改，再将构件逐个分配至框架中的适当位置。此后，构件的组装方式仍然是传统的子程序调用与参数传递。这种组装技术也要求库中构件以子程序形式出现，但它所依赖的软件设计方法不再是功能分解，而是面向数据的设计方法，例如，Jackson 系统开发方法。

●面向对象的组装技术。由于封装和继承特征，面向对象方法比其他软件开发方法更适合支持软件复用。在面向对象的软件开发方法中，如果从类库中检索出来的基类能够完全满足新系统的需求，则可以直接应用，否则，必须以基类为父类，生成相应的子类，以满足新系统的需求。

(5) 测试和发布：可以是一个增量和迭代的过程。

三、试题二 知识补充

影响软件的可维护性有以下 7 个因素。

1、可理解性。一个可维护的软件必然是可理解的。软件的可理解性是指通过阅读源代码和相关文档，了解软件的功能和如何运行的容易程度。软件的可理解性可以使用“90-10 测试”的方法来衡量，即如果一个有经验的程序员阅读一份源代码清单 10 分钟，可以写出该程序的 90%，则认为这个程序具有可理解性。

2、可测试性。一个可维护的软件必然是可测试的。软件的可测试性是指验证软件程序正确

的难易程度。可测试性好的软件，通常意味着软件设计简单，复杂性低。因为软件的复杂性越大，测试的难度也就越大。

3、可修改性。一个可维护的软件必然是可修改的。软件的可修改性是指修改软件的难易程度。软件的可修改性可以通过进行几个简单的修改练习来评价。假设软件的平均复杂性是 C，要修改的模块的复杂性是 A，那么修改的难度可由下面公式计算： $D=A/C$ 。

4、可靠性。一个软件的可靠性越高，需要维护的概率就会越低。软件的可靠性是指软件在满足用户需求的前提下，在给定的时间段内正确运行的概率。软件可靠性的度量有以下两种方法：根据软件的错误统计进行可靠性预测。如度量软件的平均失效间隔时间（MTTF）。根据软件的复杂性进行可靠性预测。

5、可移植性。软件运行环境的变化是软件维护的一种常见情形，可移植性好的软件会降低维护的概率。软件的可移植性是指将软件从一个环境移植到新的的环境下正确运行的难易程度。一个可移植的软件应具有良好的结构，使用独立于机器的高级语言编写。

6、可使用性。软件易于使用通常意味着软件设计简单，易于理解。软件的可使用性是指用户使用软件的难易程度。软件的可使用性可以通过测试用户首次使用软件掌握常用功能的时间来衡量。

7、效率。效率是指软件既能很好地完成用户期望的功能、性能，又不浪费机器资源的程度。软件设计不能一味地追求效率，盲目地追求效率会使得软件的其它质量特性受到影响，比如降低软件的可维护性。

四、试题三 知识补充

区块链从本质上来看就是一个数据库，在其中存储的数据具备了“不可伪造，全程留痕，公开可追溯”等特性。 区块链的四大核心技术如下：

1、分布式账本。指的是交易记账由分布在不同地方的多个节点共同完成，而且每一个节点记录的是完整的账目，因此它们都可以参与监督交易合法性，同时也可以共同为其作证。跟传统的分布式存储有所不同，区块链的分布式存储的独特性主要体现在两个方面：一是区块链每个节点都按照块链式结构存储完整的数据，传统分布式存储一般是将数据按照一定的规则分成多份进行存储。二是区块链每个节点存储都是独立的、地位等同的，依靠共识机制保证存储的一致性，而传统分布式存储一般是通过中心节点往其他备份节点同步数据。没有任何一个节点可以单独记录账本数据，从而避免了单一记账人被控制或者被贿赂而记假账的可能性。也由记账节点足够多，理论上讲除非所有的节点被破坏，否则账目就不会丢失，从而保证了账目数据的安全性。

2、非对称加密。存储在区块链上的交易信息是公开的，但是账户身份信息是高度加密的，只有在数据拥有者授权的情况下才能访问到，从而保证了数据的安全和个人的隐私。

3、共识机制。就是所有记账节点之间怎么达成共识，去认定一个记录的有效性，这既是认定的手段，也是防止篡改的手段。区块链提出了四种不同的共识机制，适用于不同的应用场景，在效率和安全性之间取得平衡。区块链的共识机制具备“少数服从多数”以及“人人平等”的特点，其中“少数服从多数”并不完全指节点个数，也可以是计算能力、股权数或者其他计算机可以比较的特征量。“人人平等”是当节点满足条件时，所有节点都有权优先提出共识结果、直接被其他节点认同后并最后有可能成为最终共识结果。以比特币为例，采用的是工作量证明，只有在控制了全网超过 51% 的记账节点的情况下，才有可能伪造出一条不存在的记录。当加入区块链的节点足够多的时候，这基本上不可能，从而杜绝了造假的可能。

4、智能合约。是基于这些可信的不可篡改的数据，可以自动化的执行一些预先定义好的规则和条款。以保险为例，如果说每个人的信息（包括医疗信息和风险发生的信息）都是真实可信的，那就很容易的在一些标准化的保险产品中，去进行自动化的理赔。在保险公司的日常业务中，虽然交易不像银行和证券行业那样频繁，但是对可信数据的依赖是有增无减。

因此，笔者认为利用区块链技术，从数据管理的角度切入，能够有效地帮助保险公司提高风险管理能力。具体来讲主要分投保人风险管理及保险公司的风险监督。

五、试题四 知识补充

湖仓一体是一种新型的开放式架构，打通了数据仓库和数据湖，将数据仓库的高性能及管理能力与数据湖的灵活性融合了起来，底层支持多种数据类型并存，能实现数据间的相互共享，上层可以通过统一封装的接口进行访问，可同时支持实时查询和分析，为企业进行数据治理带来了更多的便利性。湖仓一体可在数据入湖后原地进行数据处理与分析，能有效避免数据冗余及流动导致的算力、网络及成本开销，可以作为超大型 ODS 存储贴源数据，实现全量数据的实时处理。

湖仓一体架构在数据管理中主要具有以下几大关键特征：

1、支持分析多种类型数据。湖仓一体架构可为多应用程序提供数据的入库、转换、分析和访问。数据类型包括结构化与非结构化类型，如文本、图像、视频、音频等，以及半结构化数据，如 JSON 等。

2、数据可治理，避免产生数据沼泽。湖仓一体架构可以支持各类数据模型的实现和转变，支持 DW 模式架构，例如星型模型、雪花模型等，可保证数据的完整性，同时具有健全的治理和审计机制，能够避免数据沼泽现象的出现。

3、事务支持。在企业中，数据库往往要为业务系统提供并发的数据读取和写入。湖仓一体架构对事务 ACID 的支持，可确保并发访问，尤其是 SQL 访问模式下的数据一致性、正确性。

4、BI 支持。湖仓一体支持直接在源数据上使用 BI 工具，这样可以提高分析效率，降低数据延时。另外，相比于在数据湖和数据仓库中分别操作两个副本的方式，湖仓一体更具成本优势。

5、存算分离。湖仓一体采用存算分离架构，可使系统能够扩展到更大规模的并发能力和数据容量，能满足新时代对于分布式数据架构的要求。

6、开放性。湖仓一体采用开放、标准化的存储格式（例如行存、列存、块存），能提供丰富的 API 支持。因此，各种工具和引擎（包括机器学习和 Python/R 库）可以高效地对数据进行直接访问。

从落地性来看，湖仓一体技术架构落地目前有三种方式：

1、基于 Hadoop 体系的数据湖向数据仓库能力扩展，湖中建仓，从数据湖进化到湖仓一体。湖仓一体结合了数据湖和数据仓库特点，直接在用于数据湖的低成本存储上实现与数据仓库中类似的数据结构和数据管理功能。目前主要有 Netflix 等开源企业在探索此技术路线。

2、基于自身云平台或第三方对象存储（如 OSS、S3、Ceph 等），基于 Hadoop 或自研技术进行湖仓一体能力的搭建。探索此技术路线的通常是各大云厂商，如 AWS、阿里云、华为云等。

3、以数据库技术为基础，自研分布式平台，从调度、计算到存储不依赖第三方平台，形成可以灵活在公有云、私有云、裸金属等场景独立部署使用的能力。技术方向上更注重于实时高并发场景及非结构化数据数据治理，并逐步向更广泛的分析场景发展，主要厂商以 Snowflakes、Databricks、巨杉数据库等为代表。

系统架构设计师学习 QQ 群: 231352210 软件设计师学习 QQ 群: 759713504

诸葛老师 QQ: 362842353

VIP 购买方式，淘宝搜索：软考诸葛老师

全国计算机技术与软件专业技术资格（水平）考试

2023 下半年 系统架构设计师 综合知识 答案解析

试题（1）

参考答案: A

答案解析: 真题缺失。

试题（2）

参考答案: A

答案解析: 第一范式 (1NF) : 确保每列都是原子的, 即不可再分。根据描述, 此模型显然满足 1NF。

第二范式 (2NF) : 要求在 1NF 的基础上, 所有非主属性完全依赖于整个主键, 而不是主键的一部分。在这个模型中, “工资金额”依赖于“工资级别”, 但题目说明“工资级别”是通过“员工姓名”这个主键间接决定的, 因此“工资金额”实际上依赖于整个主键“员工姓名”, 所以模型满足 2NF。

第三范式 (3NF) : 要求在 2NF 的基础上, 所有非主属性不传递依赖于主键。这里的问题在于“工资金额”对“员工姓名”的依赖是通过“工资级别”传递的, 这违反了 3NF 的要求, 除非“工资级别”本身也是一个独立的属性, 直接由“员工姓名”决定, 而不是作为决定“工资金额”的中间变量。然而, 题目并未明确说明这一点, 仅提到“工资级别决定工资金额”, 因此按照严格定义, 此模型可能不直接满足 3NF。

试题（3）

参考答案: A

答案解析: (教材 P47) 嵌入式系统是以应用为中心、以计算机技术为基础, 并将可配置与可裁减的软、硬件集成于一体的专用计算机系统, 需要满足应用对功能、可靠性、成本、体积和功耗等方面的要求。

试题 (4)

参考答案: C

答案解析: (教材 P259) 软件体系结构风格是描述某一特定应用领域中系统组织方式的惯用模式。体系结构风格定义一个系统家族, 即一个体系结构定义一个词汇表和一组约束。

试题 (5)

参考答案: D

答案解析: (教材 P254) ABSD 方法有 3 个基础。第 1 个基础是功能的分解。在功能分解中, ABSD 方法使用已有的基于模块的内聚和耦合技术。第 2 个基础是通过选择体系结构风格来实现质量和商业需求。第 3 个基础是软件模板的使用, 软件模板利用了一些软件系统的结构。

基于架构的软件设计 (Architecture-Based Software Design, ABSD) 是一个自顶向下, 递归细化的软件开发方法, 它以系统功能的分解为基础, 通过选择架构风格实现质量和业务需求, 并强调在架构设计过程中使用软件架构模板因此, 正确答案是: D。

试题 (6)

参考答案: D

答案解析: (教材 P254) ABSD 方法中使用的设计元素如图 7-1 所示。在最顶层, 系统被分解为若干概念子系统和一个或若干个软件模板。在第 2 层, 概念子系统又被分解成概念构件和一个或若干个附加软件模板。

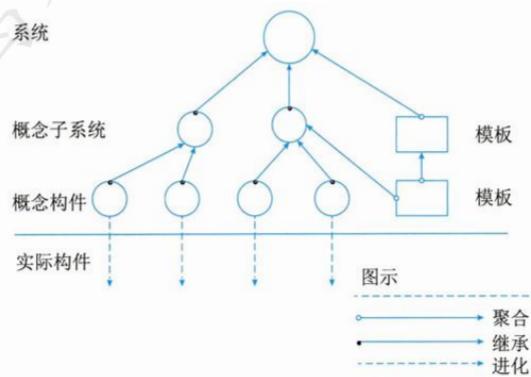


图 7-1 ABSD 方法过程

试题 (7)

参考答案: A

答案解析: (教材 P255) 体系结构需求一般来自 3 个方面, 分别是系统的质量目标、系统的商

业目标和系统开发人员的商业目标。软件体系结构需求获取过程主要是定义开发人员必须实现的软件功能，使得用户能完成他们的任务，从而满足业务上的功能需求。与此同时，还要获得软件质量属性，满足一些非功能需求。

2023 下半年 系统架构设计师 案例分析 答案解析

试题一(共 25 分)

参考答案:

【问题 1】

(1)2 (2)1 (3)高 (4)低 (5)必要时进行全量计算，计算开销相对较小 (6)满足实时性 (7)大 (8)弱

【问题 2】

(1)d (2)e (3)k (4)g (5)h (6)I (7)j (8)f (9)n

【问题 3】

(1)批处理层 (2)加速层 (3)服务层

该系统的大数据架构是基于 Lambda 架构搭建的大数据平台处理奥运会大规模视频网络观看数据。

试题二(共 25 分)

参考答案:

【问题 1】

需求图(REQ 或 req):用于表述文字化的需求、需求间的关系，以及与之存在满足、验证等关系的其他模型元素。

用例图：从用户的角度提供系统或业务流程功能的概述。

这两者的主要区别在于，需求图更关注系统内部的功能和流程，以及这些功能和流程如何满足用户的需求；而用例图更关注用户如何使用系统，以及系统与用户的交互。简单来说，需求图更偏向于系统的功能和流程，而用例图更偏向于用户的使用体验。

【问题 2】

(1)-(5):BCDEF

1)复合关系：复合需求可以包含需求层次结构中的子需求。复合需求可以声明系统应执行 A 和 B，可以将其分解为系统应执行 A 和系统应进行 B 的子需求

2)派生关系：派生的需求通常对应于系统层次结构下一级的需求。一个简单的例子是车辆加

速需求，该需求被分析以导出发动机动力等方面的需求

3)细化关系：细化关系可用于描述如何使用模型元素或元素集进一步细化需求。例如，可以使用用例或活动图来细化基于文本的功能需求

4)满足关系：满足关系描述了设计或实现模型如何满足一个或多个需求。然后，系统建模者可以指定旨在满足要求的系统设计元素

5)验证关系：验证关系定义了测试用例或其他模型元素如何验证需求。在 SysML 中，测试用例或其他元素可以用作表示任何标准检验方法的通用机制，分析，演示或测试。

6)复制关系：真正需要跨产品系列和项目重用需求。典型的方案是适用于产品和/或产品系列中重复使用的项目和要求的法定法规或合同要求。

7)追溯关系：追溯关系提供了需求和任何其他模型元素之间的通用关系。追溯关系对于将需求与源文档相关联或在规范树中的规范之间建立关系可能很有用。

试题一-试题五

见真题即可。

2023 下半年 系统架构设计师 论文写作 答案解析

见真题即可。

全国计算机技术与软件专业技术资格（水平）考试

2024 上半年 系统架构设计师 综合知识 答案解析

试题（1）

参考答案: A

答案解析：多道程序设计技术是指在内存同时放若干道程序，使它们在系统中并发执行，共享系统中的各种资源。当一道程序暂停执行时，CPU 立即转去执行另一道程序。简单来说就是有多个程序同时进入主存并行运行，在操作系统中采用多道程序设计方式能提高 CPU 和外部设备的利用效率。一般来说，为了实现多道程序设计，计算机需要有更大的内存。

试题 (2)

参考答案: C

答案解析: 高响应比优先: 基本思想是把 CPU 分配给就绪队列中响应比最高的进程。高响应比优先调度算法既考虑作业的执行时间也考虑作业的等待时间, 综合了先来先服务和最短作业优先两种算法的特点。

时间片轮转: 是一种常见的进程调度算法, 其目的是确保所有进程都能公平地获得 CPU 时间片。每个进程被分配一个固定大小的时间片, 当一个进程用尽其时间片后, 调度器将其移到队列的末尾, 然后选择下一个进程执行。

先来先服务: 按照进程到达的先后顺序进行调度, 不考虑执行时间的长短。

短作业优先: 选择执行时间最短的进程来先执行。这可能导致长作业等待时间过长, 也无法保证所有进程都能公平地获得 CPU 时间。

试题 (3)

参考答案: D

答案解析: 事务是数据库系统中不可分割的逻辑工作单位, 包括四个特性: 原子性 A、一致性 C、隔离性 I、持久性 D。

试题 (4)

参考答案: D

答案解析: 笛卡尔积是指两个集合之间的每个元素对的组合。关系 R 有 m 个元组, 关系 S 有 n 个元组, 那么 R 和 S 的笛卡尔积将包含每个 R 中的元组与 S 中的每个元组的组合, 即每一个 R 中的元组都会与 S 中的 n 个元组进行组合。对于 R 中的每个元组, 都需要与 S 中的 n 个元组进行组合, 所以笛卡尔积将会有 $m \times n$ 个元组。因此答案选择 D 选项。

试题 (5)

参考答案: C

答案解析: (教材 P550) 大多数嵌入式系统都具备实时特征, 其典型架构可以概括为层次化模式架构和递归模式架构。

试题 (6)

参考答案: C

答案解析: (教材 P181) 敏捷方法是适应型, 非可预测型。

试题 (7)

参考答案: B

答案解析: (教材 P208-210) 净室(Cleaning Room)软件工程是一种应用数学与统计学理论以经济的方式生产高质量软件的工程技术, 力图通过严格的工程化的软件过程达到开发中的零缺陷或接近零缺陷。净室方法不是先制作一个产品, 再去消除缺陷, 而是要求在规约和设计中消除错误, 然后以“净”的方式制作, 可以降低软件开发中的风险, 以合理的成本开发出高质量的软件。

1、净室软件工程的理论基础主要是函数理论和抽样理论。

2、净室软件工程中应用的技术手段主要有以下 4 种。

①统计过程控制下的增量式开发

②基于函数的规范与设计

③正确性验证(正确性验证被认为是 CSE 的核心, 正是由于采用了这一技术, 净室项目的软件质量才有了极大的提高。)

④统计测试和软件认证

3、净室软件工程在使用的过程中, 也显示出一些缺点。

1) CSE 太理论化, 需要更多的数学知识。其正确性验证的步骤比较困难且比较耗时。CSE 要求采用增量式开发、盒子结构、统计测试方法, 普通工程师必须经过加强训练才能掌握, 开发软件的成本比较高昂。

2) CSE 开发小组不进行传统的模块测试, 这是不现实的。工程师可能对编程语言和开发环境还不熟悉, 而且编译器或操作系统的 bug 也可能导致未预期的错误。

3) CSE 毕竟脱胎于传统软件工程, 不可避免地带有传统软件工程的一些弊端。

试题 (8)

参考答案: C

答案解析: (教材 P323-324) 软件测试不可以证明被测对象的正确性。软件测试的目标是发现软件中的错误和缺陷, 而不是证明软件是正确的。测试只能提供有限的覆盖范围和可靠性。即便通过了所有测试用例, 软件也可能存在其他未被发现的错误。因此, 测试只是尽可能地减少风险,

而不能完全保证软件的正确性。C 选项错误。答案选 C。

在软件测试中，优先测试那些最重要或最频繁使用的功能，80% 软件错误都可以在大概 20% 模块中找到根源。A 选项正确。

设计测试用例就是针对特定功能或组合功能设计测试方案，并编写成文档。测试用例的选择既要有一般情况，也应有极限情况以及最大和最小的边界值情况。因为测试的目的是暴露应用软件中隐藏的缺陷，所以在设计选取测试用例和数据时要考虑那些易于发现缺陷的测试用例和数据，结合复杂的运行环境，在所有可能的输入条件和输出条件下确定测试数据，来检查应用软件是否都能产生正确的输出。B 选项正确。

一个典型的测试用例应该包括下列组成部分。

- (1) 测试用例标识。
- (2) 被测对象。
- (3) 测试环境及条件。
- (4) 测试输入。
- (5) 操作步骤。
- (6) 预期输出。
- (7) 判断输出结果是否符合标准。
- (8) 测试对象的特殊需求。D 选项正确。

试题 (9)

参考答案: B

答案解析: 系统测试的依据是软件需求规格说明书。系统测试采用黑盒测试，以此来检查该系统是否符合软件需求。

试题 (10) (11)

参考答案: D、D

答案解析: (教材 P213) 构件组装是指构件相互直接集成或是用“胶水代码”将其整合在一起创造一个系统或另一个构件的过程。常见的方式包括：顺序组装、层次组装、叠加组装。同时，构件组装中经常会面临接口不兼容的问题，如果一个构件的提供接口是另一个构件请求接口的一个子集，则属于操作不完备的情况。

试题 (12)

参考答案: A

答案解析: 关键路径是项目中的最长路径, 决定着项目的总工期(最短工期), 关键路径上的活动称为关键活动。这里从图中可以计算出, 关键路径为 ADIL, 工期为 $8+3+7+8=26$ 天。

试题 (13)

参考答案: B

答案解析: (教材 P83) 用例图展现了一组用例、参与者以及它们之间的关系。用例图通常包括用例和参与者以及它们之间的关系。用例之间有扩展关系和包含关系, 参与者和用例之间有关联关系, 用例与用例、参与者与参与者之间有泛化关系。

试题 (14)

参考答案: D

答案解析: (教材 P170) 《计算机信息系统安全保护等级划分准则》把计算机信息安全划分为了 5 个等级, 安全性从低到高依次为第 1 级: 用户自主保护级 (对应 TCSEC 的 C1 级)、第 2 级: 系统审计保护级 (对应 TCSEC 的 C2 级)、第 3 级: 安全标记保护级 (对应 TCSEC 的 B1 级)、第 4 级: 结构化保护级 (对应 TCSEC 的 B2 级) 和第 5 级: 访问验证保护级 (对应 TCSEC 的 B3 级)。

试题 (15)

参考答案: D

答案解析: 企业应用集成(Enterprise Application Integration, EAI)是伴随着企业信息系统的发展而产生和演变的。EAI 技术是将进程、软件、标准和硬件联合起来, 在两个或更多的企业信息系统之间实现无缝集成, 使它们就像一个整体一样。EAI 提供 4 个层次的服务, 从下至上依次为通讯服务、信息传递与转化服务、应用连接服务、流程控制服务, 最上层是流程控制服务。

试题 (16)

参考答案: B

答案解析: (教材 P524) URI 和资源是一对一关系。

试题 (17)

参考答案: A

答案解析: (教材 P267) 复用的基本过程主要包括 3 个阶段: 首先构造/获取可复用的软件资产, 其次管理这些资产, 最后针对特定的需求, 从这些资产中选择可复用的部分, 以开发满足需求的应用系统。

1. 复用的前提: 获取可复用的软件资产: 首先需要构造恰当的、可复用的资产, 并且这些资产必须是可靠的、可被广泛使用的、易于理解和修改的。

2. 管理可复用资产: 该阶段最重要的是: 构件库(Component Library), 由于对可复用构件进行存储和管理, 它是支持软件复用的必要设施。构件库中必须有足量的可复用构件才有意义, 构件库应提供的主要功能包括构件的存储、管理、检索以及库的浏览与维护等, 以及支持使用者有效地、准确地发现所需的可复用构件。在这个过程中, 存在两个关键问题:

一是构件分类, 构件分类是指将数量众多的构件按照 某种特定方式组织起来; 二是构件检索, 构件检索是指给定几个查询需求, 能够快速准确地找到相关构件。

3. 使用可复用资产: 在最后阶段, 通过获取需求, 检索复用资产库, 获取可复用资产, 并定制这些可复用资产: 修改、扩展、配置等, 最后将它们组装与集成, 形成最终系统。

试题 (18)

参考答案: D

答案解析: (教材 P268) DSSA 的基本活动:

1. 领域分析

这个阶段的主要目标是获得领域模型。领域模型描述领域中系统之间的共同需求, 即领域模型所描述的需求为领域需求。

2. 领域设计

这个阶段的主要目标是获得 DSSA。DSSA 描述在领域模型中表示的需求的解决方案, 它不是单个系统的表示, 而是能够适应领域中多个系统的需求的一个高层次的设计。

3. 领域实现

这个阶段的主要目标是依据领域模型和 DSSA 开发和组织可重用信息。这些可重用信息可能是从现有系统中提取得到, 也可能需要通过新的开发得到。它们依据领域模型和 DSSA 进行组织, 也就是领域模型和 DSSA 定义了这些可重用信息的重用时机, 从而支持了系统化的软件重用。

试题 (19)

参考答案: D

答案解析: (教材 P475) 物联网可以分为三个层次, 底层是用来感知数据的感知层, 即利用传感器、二维码、RFID 等设备随时随地获取物体的信息。第二层是数据传输处理的网络层, 即通过各种传感网络与互联网的融合, 将对象当前的信息实时准确地传递出去。第三层则是与行业需求结合的应用层, 即通过智能计算、云计算等将对象进行智能化控制。

试题 (20) (21)

参考答案: D、C

答案解析: (教材 P273) 安全性(security)是指系统在向合法用户提供服务的同时能够阻止非授权用户使用的企图或拒绝服务的能力。安全性可根据系统可能受到的安全威胁类型来分类。其中, 机密性保证信息不泄露给未授权的用户、实体或过程; 不可否认性是指信息交换的双方不能否认其在交换过程中发送信息或接收信息的行为。

试题 (22)

参考答案: B

答案解析: (教材 P271) 软件系统质量属性(Quality Attribute)是一个系统的可测量或者可测试的属性, 用来描述系统满足利益相关者(Stakeholders)需求的程度。基于软件系统的生命周期, 可以将软件系统的质量属性分为开发期质量属性和运行期质量属性 2 个部分。开发期质量属性主要指在软件开发阶段所关注的质量属性, 主要包含 6 个方面: 易理解性、可扩展性、可重用性、可测试性、可维护性、可移植性。

运行期质量属性主要指在软件运行阶段所关注的质量属性, 主要包含 7 个方面: 性能、安全性、可伸缩性、互操作性、可靠性、可用性、鲁棒性。

试题 (23) (24)

参考答案: A、C

答案解析: (教材 P272) (1)性能: 性能是指软件系统及时提供相应服务的能力, 如速度、吞吐量和容量等的要求。

(2)安全性: 指软件系统同时兼顾向合法用户提供服务, 以及阻止非授权使用的能力。

(3)可伸缩性: 指当用户数和数据量增加时, 软件系统维持高服务质量的能力。例如, 通过增

加服务器来提高能力。

(4)互操作性：指本软件系统与其他系统交换数据和相互调用服务的难易程度。

(5)可靠性：软件系统在一定的时间内持续无故障运行的能力。

(6)可用性：指系统在一定时间内正常工作的时间所占的比例。可用性会受到系统错误，恶意攻击，高负载等问题的影响。

试题 (25) (26)

参考答案: A、C

答案解析: (教材 P274) 为了精确描述软件系统的质量属性，通常采用质量属性场景作为描述质量属性的手段。其中，响应描述在激励到达后所采取的行动。

试题 (27)

参考答案: B

答案解析: (教材 P280) 架构权衡分析方法(Architecture Trade off Analysis Method, ATAM)是在 SAAM 的基础上发展起来的，主要针对性能、实用性、安全性和可修改性，在系统开发之前，对这些质量属性进行评价和折中。(其中实用性就是可用性)

试题 (28)

参考答案: C

答案解析: (教材 P317) 本题考查软件可靠性管理相关概念。需求分析的任务包括：

- ①确定软件的可靠性目标
- ②分析可能影响可靠性的因素
- ③确定可靠性的验收标准
- ④制定可靠性管理框架
- ⑤制定可靠性文档编写规范
- ⑥制定可靠性活动初步计划
- ⑦确定可靠性数据收集规范

可靠性建模是测试阶段的任务，本题选择 C 选项。

试题 (29)

参考答案: D

答案解析: 数字孪生生态系统由基础支撑层、数据互动层、模型构建与仿真分析层、共性应用层和行业应用层组成。其中基础支撑层由具体的设备组成，包括工业设备、城市建筑设备、交通工具、医疗设备组成。数据互动层包括数据采集、数据传输和数据处理等内容。模型构建与仿真分析层包括数据建模、数据仿真和控制。共性应用层包括描述、诊断、预测、决策四个方面。行业应用层则包括智能制造、智慧城市在内的多方面应用。

试题 (30)**参考答案: D**

答案解析: 知识产权法属于民法范畴，是民法中的特别法。《中华人民共和国民法典》第一百二十三条民事主体依法享有知识产权。知识产权是权利人依法就下列客体享有的专有的权利：(一)作品；(二)发明、实用新型、外观设计；(三)商标；(四)地理标志；(五)商业秘密；(六)集成电路布图设计；(七)植物新品种；(八)法律规定的其他客体。

试题 (31)**参考答案: B**

答案解析: 有定义：若 X, Y 是某线性规划问题的最优解，则 $Z = \lambda X + (1 - \lambda)Y (0 \leq \lambda \leq 1)$ 也是该问题的最优解。所以 $\mu = 1 - \lambda$ ，因此应满足 $\lambda + \mu = 1$ ，又因为 $0 \leq \lambda \leq 1$ ，所以 $0 \leq 1 - \lambda \leq 1$ ，所以 $\lambda, \mu \geq 0$ 。

2024 上半年 系统架构设计师 案例分析 答案解析

见案例真题即可。

2024 上半年 系统架构设计师 论文写作 答案解析

试题一 写作要点

撰写关于 Lambda 架构的软考论文时，一个清晰且结构化的大纲是成功的关键。以下是一个简单的论文大纲示例，旨在覆盖 Lambda 架构的核心概念、设计原则、优缺点、实际应用案例以及对比其他架构(如 Kappa 架构)的分析：

大纲

- 简要介绍 Lambda 架构的基本概念及其在大数据处理领域的地位。
- 概述论文的主要研究内容、目的及预期贡献。
- 背景介绍: 阐述大数据处理的挑战, 特别是在实时性和历史数据一致性方面。
- 研究意义: 解释研究 Lambda 架构的重要性, 尤其是在现代数据驱动型企业中的应用。
- 论文结构概述。

结合项目可以写的一些内容

论述 1: Lambda 架构概述

- 定义: 明确 Lambda 架构的定义及提出背景。
- 架构层次: 详细介绍批处理层、加速层和服务层的功能与作用。
- 设计原则: 概括 Lambda 架构遵循的核心设计理念。

论述 2: Lambda 架构的核心组件与工作原理

- 批处理层(Batch Layer): 描述其在数据摄入、处理和建立不可变数据视图的作用。
- 加速层(Speed Layer): 分析实时处理和补充最近数据的功能。
- 服务层(Serving Layer): 说明如何合并实时与批处理数据, 提供统一查询接口。
- 数据流与一致性: 探讨数据在不同层间流动的机制及确保查询结果一致性的策略。

论述 3: Lambda 架构的优势与局限性

- 优势: 即时查询能力、容错与可扩展性、兼容历史数据处理。
- 局限性: 数据重复处理、维护复杂度高、系统设计复杂。

论述 4: Lambda 架构的实际应用案例分析

- 选取一至两个行业或具体项目, 分析 Lambda 架构的应用场景、实施效果与面临的挑战。
- 例对比: 在不同规模和需求下的适用性讨论。

论述 5: Lambda 架构与其他架构的对比(可选, 具体看论文题目有没有)

- Kappa 架构介绍: 概述 Kappa 架构的设计理念和工作流程。
- 对比分析: 从数据处理模式、实时性、维护成本等方面对比 Lambda 与 Kappa。
- 选择考量: 根据应用场景指导何时选择 Lambda 或 Kappa。

论文结尾:

- 总结 Lambda 架构在项目实践过程中的使用效果, 以及对于项目的核心价值和存在的问题。
- 强调其在大数据处理领域的持续影响力, 并指出研究的现实意义。

试题二 写作要点

领域驱动设计(Domain-Driven Design, DDD)与模型驱动软件开发方法(Model-Driven Software Development, MDSD)是两种注重软件开发质量和效率的方法论。下面分别概述两者的核心概念，并探讨它们如何相互作用以提升软件项目的成功概率。

(1) 领域驱动设计(DDD)

核心要素：

- 1.核心领域：识别并聚焦于业务中最关键、最具价值的部分。
- 2.界限上下文：划分不同的领域边界，确保领域内的模型一致性。
- 3.领域模型：通过实体、值对象、聚合根等元素精确表达业务逻辑。
- 4.通用语言：建立业务和技术团队间的共同语言，减少误解。
- 5.分层架构：常见的有战略设计、战术设计分层，确保领域逻辑的纯净性。

(2) 模型驱动软件开发方法(MDSD)

基本概念：

- 1.抽象模型：从不同的视角(如业务、技术)建立软件的抽象表示。
- 2.模型转换：使用转换引擎或代码生成工具将高级模型转换为可执行代码或配置。
- 3.元模型与元数据：定义模型的结构和规则，支持模型的标准化和复用。
- 4.工具支持：依赖于建模工具、代码生成器等自动化工具链来提高开发效率。
- 5.迭代与反馈：模型不是静态的，需在开发过程中不断迭代优化，以反映真实需求。

DDD 与 MDSD 的协同

结合点：

- 领域模型作为起点：DDD 的领域模型可作为 MDSD 中高层次抽象模型的基础，确保模型直接反映业务本质。
- 模型驱动的领域实现：利用 MDSD 工具将 DDD 中定义的领域模型转换为具体的实现代码，提高开发效率，减少错误。
- 复杂性管理：DDD 帮助识别和管理软件的复杂性，而 MDSD 通过模型的自动化转换减少实现层面的复杂性。
- 迭代与进化：DDD 强调通过迭代深化对领域的理解，而 MDSD 支持模型的快速迭代和演化，二者结合促进了软件的持续优化。
- 标准化与复用：MDSD 的元模型和元数据管理有助于 DDD 中核心领域模型的标准化表述和跨项目复用。

结论

结合领域驱动设计与模型驱动软件开发方法，可以构建更加贴合业务需求、易于维护和扩展的软件系统。DDD 确保软件设计深度契合业务领域，而 MDSD 则通过自动化工具链提升开发效率和质量，两者相辅相成，共同促进软件工程实践的现代化和高效性。在实践中，需要根据项目特点灵活运用这两种方法，平衡业务理解的深度与技术实现的效率。

试题三 写作要点

软件单元测试、静态测试和动态测试，白盒测试的覆盖标准以及回归测试如何实施。

仅从单元测试来写一篇论文，内容较为狭隘，可能从单元测试的以下方面来写：

(1) 测试的内容

模块接口测试、局部数据结构测试、边界条件测试、路径覆盖测试、错误处理与异常测试。

(2) 测试的执行场景

描述为什么要进行这些测试，以及测试的必要性。

撰写关于“软件单元测试的架构设计师”这一主题的论文，可以从介绍单元测试的基本概念入手，逐步深入到架构设计层面如何有效指导和实施单元测试策略，最终提出创新的设计方法或最佳实践案。

下面是简化的论文大纲及部分内容示例，仅供参考：

关键词：单元测试，架构设计，测试策略，软件质量，开发效率

1. 单元测试基础理论

2. 架构设计与测试的关联性

3. 现有单元测试方法及局限性

4. 研究缺口分析

基于架构设计的单元测试策略

架构视角下的单元测试原则

- 模块独立性
- 测试可达性
- 依赖管理

设计阶段的单元测试考虑

- 接口设计与测试先行
- 模块边界清晰化

实施框架与工具选择

- 自动化测试框架
- 持续集成/持续部署(CI/CD)集成

实现步骤:

项目背景与目标系统概述，为什么需要进行单元测试

测试架构设计与实施步骤

识别核心模块与接口

设计测试用例与桩/模拟对象

- 集成自动化测试流程

结果分析与评估

- 测试覆盖率提升
- 缺陷检测率与修复周期
- 对开发流程的影响

结论:

- 使用了单元测试给项目和系统带来哪些好处，收到了.....好评；
- 总结与建议。

试题四 写作要点

云原生 DevOps、云上运维是新技术。以下是云原生 DevOps 简介。

1.核心要素:

- 容器化：利用 Docker 等技术，实现应用的轻量级打包与标准化部署。
- 微服务架构：将应用拆分成小型、独立的服务，便于独立开发、部署和扩展。
- 持续集成/持续部署(CI/CD)：自动化代码集成、测试与部署，加速软件交付流程。
- 基础设施即代码(IaC)：使用代码(如 Terraform、CloudFormation)管理基础设施配置。
- 服务网格：如 Istio，管理服务间的交互，提供安全、监控与故障恢复等功能。

2.云上运维特点:

(1) 优势

- 弹性与可扩展性：根据需求自动调整资源，应对流量高峰。
- 成本效益：按需付费模型，减少闲置资源成本。
- 自动化运维：利用云服务商工具(如 AWS CloudFormation、Azure Automation)实现运维任

务自动化。

- 全球部署：轻松实现应用的全球分布与低延迟访问。
- 安全性与合规：利用云平台提供的安全服务和最佳实践，保障数据安全与合规性。

(2) 实施策略

- 1.拥抱容器化与 Kubernetes：作为云原生基础，实现应用的高效部署与管理。
- 2.建立 CI/CD 管道：集成如 Jenkins、GitLab CI/CD，确保代码变更快速、可靠地交付到生产环境。
- 3.采用微服务设计：分解大型应用，提高开发效率与系统的可维护性。
- 4.实施全面监控与日志管理：利用 ELK Stack、Prometheus+Grafana 等工具，实现系统状态的实时监控与问题快速定位。
- 5.强化安全性：集成云原生安全解决方案，如密钥管理、网络策略、安全扫描等。
- 6.文化和组织调整：推动 DevOps 文化，促进开发、运维及其它团队之间的紧密合作与知识共享。

(3) 协同工作

云原生 DevOps 与云上运维的协同，关键在于实现开发与运维流程的高度自动化与协同，确保从代码提交到生产部署的每一个环节都能快速、安全、可靠。通过云原生技术栈的运用，结合云平台的强大功能，可以极大地提升软件交付的速度与质量，同时降低运维复杂度和成本。此外，持续的学习与反馈循环也是保持体系高效运行的重要组成部分。

系统架构设计师学习 QQ 群: 231352210

软件设计师学习 QQ 群: 759713504

诸葛老师 QQ: 362842353

VIP 购买方式，淘宝搜索: 诸葛老师系统架构设计师