## 服务架构

2层C/S架构(表现层，数据层)：前端直连数据库

3层C/S架构(表现层，应用服务，数据层)：业务逻辑提取到服务器中(如java)

**架构风格：反应结构和语义特征**

数据流风格：

批处理：对多个任务批量处理,可以同时执行

管道-过滤器：执行有先后顺序的步骤，数据流驱动

调用返回风格：调用其他服务，等返回后再继续，单线程，会阻塞

主/子程序：统一入口，不同功能对应不同子程序执行后，再执行主程序

面向对象：java中的调用返回

层次结构：降低性能

独立构件风格

进程通信: 两个独立的进程，RPC/接口调用来进行消息传递

事件驱动：通过MQ驱动其他消费者处理，解耦

虚拟机风格

解释器：前端输入伪sql,后端进行解析，并去数据库查询

基于规则：根据规则(温度传感器，高于26度)出发动作(开启制冷)，

仓库风格:数据存在中心，各组件访问数据仓库进行协作，避免相互传递

黑板系统：通过中央数据结构(黑板, cpu缓存中)来协作，处理复杂问题

数据库系统: 数据存在数据库中

超文本系统：只有链接展示静态资源

C/S：服务器提供服务和数据，客户端和用户交互

C2:将一组构件绑定在一起

RESTful

GET:查询用

POST:对数据整体更新

PUT:新建数据

PATCH:对数据部分更新

工业大模型

基础设施层：提供算例，网络保障

基座层：数据分析，形成知识图谱

模型层：模型训练，调整

交互层：人机协同

应用层：场景化处理

数据中台：风险控制、数据治理、用户画像

SOA(面向服务架构)：将软件分成各个微服务

ESB:一条总线(消息+服务+路由)将所有应用，服务连起来

服务治理：负载均衡，超时充实，限流

服务注册与发现

WSDL网页服务描述语言： 服务的作用，如何访问，位于何处

EJB:企业级java构件，用于分布式，面向对象的系统

**中间件**：提供通信；保证交易一致性；提供负载均衡和高可用

## 管理架构

ERP资源流：物流，资金流，信息流

信息系统监管过程

四控：质量、进度、投资、变更控制

三管：合同、信息、安全管理

一协调：有关单位和人员的关系

RUP：用例驱动，迭代增量的开发过程

步骤：建模、需求、分析设计、实现、测试、部署、配置和变更管理、项目管、环境

软件开发阶段

初始：需求分析，可行性

细化：设计体系，制定工作计划，识别风险

构建：开发，实现

交付：部署，交付

开放系统的安全服务：

认证：登录

访问控制：不同角色能访问的数据等级不同

数据保密性：避免数据泄露

数据完整性：避免数据被篡改

不可否认：避免否认数据发送/接收过的事实

ATAM:质量属性效用树(基于SAAM)

树根-质量属性-属性分类-属性场景

质量属性的优先级是通过场景评估后才确定的

最开始关注的属性：性能、安全性、可修改性、可用性

会对质量属性进行权衡和折中

SAAM软件架构分析方法:

关注单个场景的单个质量属性

输入：问题描述，需求说明，架构(体系结构)描述

架构风格:实现质量属性与商业需求、采用软件模板设计软件结构

架构复审：标识潜在的风险，及早发现架构设计中的缺陷和错误，用户和领域专家决定

架构演化：活动针对用户的需求变化，修改应用架构，满足新的需求

架构文档化输出是架构规格说明书和架构质量说明书

ABSD基于体系结构的软件设计(自顶向下)

需求:来自系统质量目标，系统商业目标，开发者的商业目标

阶段：需求、设计、文档化、复审、实现、演化

用例和质量场景描述需求

视角和视图描述架构

DSSA特定领域软件体系结构

并发的，递归的，迭代的

领域分析阶段：获得领域模型

领域设计阶段：将模型转为设计方案，获得领域架构

领域实现阶段：实现可复用构建(架构)

体系结构演化：需求归类,制定演化计划，修改构建，更新构建的相互作用，组装，技术评审

体系结构：一个词汇表(构件)和约束

企业集成:网络集成，数据集成，会聚集成，服务集成，接口集成

软件架构复用：构造/获取可复用资产， 管理..， 使用..

机会复用：开发中发现有能复用的

系统复用：开发前明确哪些能复用

安全审计的4要素：控制目标，安全漏洞，控制措施，控制测试

信息化需求：战略需求、运作需求、技术需求

CMMI(能力成熟度模型):

初始级：过程不可预测

管理级(重复)：项目级管理，类似项目有迹可循

定义级：组织级管理，标准过程化，文档化

量化管理级：数据驱动决策，过程可预测，可度量

优化级：持续改进，创新导向

专家系统：以知识库和推理机为中心

嵌入式系统架构(可裁剪)： 层级模式，代理模式(组件间解耦)

EAI企业应用集成：通讯、消息、应用、流程服务

物联网分层：感知层，网络层，应用层

数字孪生

感知控制层：传感器

数据互动层：数据清洗

模型构建层：构建模型

仿真分析层：提供云端-边缘分布式计算支撑服务

共性应用层：描述、诊断、预测、决策

## 计算机网络

无干扰下最大传输速率(奈奎斯特公式 信号状态数V)=2\*带宽B\*log2(V) bps

有干扰下最大传输速率(香农公式 信噪比-S/N)=带宽\*log2(1+S/N) bps

|  |  |  |  |
| --- | --- | --- | --- |
| **端口** | **服务** | **端口** | **服务** |
| 20 | ftp(数据) | 80 | http |
| 21 | ftp(控制) | 110 | POP3(邮箱接收) |
| 23 | Telnet终端仿真协议 | 443 | https |
| 25 | SMTP简单邮件发送协议 | 143 | IMAP电子邮件 |
|  | MIME不安全 S/MIME是安全 | 161 | SNMP-snmp |
|  |  | 162 | SNMP-trap -snmp |

## 计算机组成原理

存储器

ROM：出场自带，不可编程

PROM：可编程一次

EPROM： 可多测编程

EEPROM：可多次编程

调制解调：模拟信号转数字信号，再将数字信号转模拟信号

## 操作系统

调度算法

FIFO先进先出

LLF:开始时间-结束时间=剩余执行时间，最先的先执行(开始时间不同)

EDF(DeadLine):截止时间最早的先运行

最晚截止期：越接近截止期的先运行

页淘汰(访问位，修改位)：先淘汰没访问过的，再淘汰没修改的

用户态用访管中断来执行特权指令进行系统调用

分段存储：段大小可动态变化

IO方式

程序轮询：cpu一直循环判断外设是否准备好

程序中断：外设准备好后通知cpu去处理

DMA方式：交给DMA控制器，过程不用cpu处理(开始和结束需要cpu)

通道技术：数据传输不依赖cpu,传输完成后向cpu发出中断信号

Spooling假脱机技术：共享打印机，用户以为独占设备，提高并行效率(复用)

磁盘最短吊臂算法：先柱面最近，再扇区最近，再磁头

临界资源：同一时刻只能被一个进程访问(打印机)

页：页号(最大页数)+页内地址(最大页大小)

## 软件工程

内聚类型(低到高)

偶然：全放一块

逻辑：逻辑内聚(根据不同入参，执行增删改)

时间：相同时刻的步骤放一块(init中操作)

过程：按顺序执行步骤(数据无关)

通信：对同一个数据执行多个步骤(数据有关)

顺序：按顺序执行步骤(步骤间数据有关联)

功能：包括步骤涉及的其他数据联动操作

耦合类型

无：通过mq,中间件通信，不指定对方存在

数据：模块通过参数传递数据

标记：模块通过参数传整个对象，但只用个别字段

控制：模块A向B传递flag标志，B根据flag执行逻辑

外部：模块依赖外部系统(不同环境结果不同)

公共：多个模块共享全局变量

内容：模块A之间修改B中的数据

4+1视图：描述架构

进程(过程)视图：多进程间怎么并发交互(性能)

物理(部署)视图：服务器部署

逻辑视图：提供什么服务

开发(实现)视图：代码怎么实现，组织架构

统一场景(用例)视图：活动图，序列图，使用场景，参与者-用例的关系

**开发模型**

瀑布模型：软件需求明确，前一阶段的输出是后一阶段的输入

原型(演化)模型:需求不明确，容易变动，规模不太大

螺旋模型:原型模型(基础)+瀑布模型，是快速原型模型的改进，分为目标设定、风险分析、开发和有效性验证、评审

增量模型：尽早得到用户反馈

构件模型:重用提高软件可靠性和易维护性

喷泉模型：用户需求为动力，面向对象开发

快速应用开发：模块化程度高，基于构件

W模型:开发和测试同时进行

形式化方法：基于数学

敏捷开发:是适应性的，以人为中心，原型模型为基础，增量式开发

极限编程：交流，朴素，反馈，勇气

水晶法：以人为中心

并列争球法：增量迭代，按需求优先级实现

特征驱动：人，过程，技术

**软件结构化设计**：架构设计、接口设计、数据设计和过程设计。面向数据流，自顶而下、模块化的过程。

净室软件：基于函数和抽样理论，开发前解决缺陷，一次成功

需求跟踪

正向跟踪:从需求上找根因

逆向跟踪：从代码推理需求

用例图

参与者：

关联关系：参与者A和用例B关联

泛化(继承)关系:管理员角色继承用户所有权限

用例间：

包含关系：A包含B的行为

扩展关系：A比B在行为上有扩展(异常时)

泛化(继承)关系：A是B的特例(子类)

可维护性：修改缺陷的难度

可伸缩性：用户增加时，维持服务质量的能力

互操作性：和其他系统交互的难易成都

可用性：正常工作的时间占比

安全性

机密性：数据不给未授权的用户

完整性：方式数据被篡改

不可否认：不能否认之前调用/接收接口

可控性：信息访问的控制

安全审计：各种操作的记录

软件质量6个维度

功能性：适合性、准确性、互操作性、安全性

可靠性：容错性，易恢复性、成熟性

易用性：易学性、易理解性、易操作性

效率：资源、时间

维护性：可测试性、可修改性、稳定性、易分析性

可移植性：适应性、易安装性、一致性、可替换性

时期的维度：开发期质量属性，运行期质量属性

架构评估中的质量属性：

性能：给定时间内处理任务的效率

可靠性：能连续提供服正常服务(多久坏一次)

可用性：(允许宕机多久，多久内修复)

安全性：审计，非法修改数据，破坏系统

可修改性：改变系统的成本(时间，人力，资金)

功能性：满足用户的需求

可变性：通过配置即可满足不同用户

互操作性:和其他的系统/服务的接口进行交互

易用性：

设计模式

创建型：单例，原型(在不明确细节的情况下创建对象)，工厂方法，抽象工厂，建造者(构建与表示分离)

结构型：桥接(抽象和实现部分分离)，适配器，组合，装饰，外观，享元，代理

行为型：剩下的

权衡点：影响多个质量属性并进行权衡

风险点：存在潜在隐患

敏感点：1/多个质量属性的特性

产品配置项：输出物经评审后

**软件工具**分为软件开发工具、软件维护工具、软件管理和软件支持工具

**软件活动：**包括软件描述、软件开发、软件有效性验证和软件演化

**软件设计：**包括数据设计、软件结构设计、人机界面设计和过程设计

软件评审：保证软件质量

**质量属性场景描述**

刺激源：用户(谁)

刺激：修改(动作)

环境：设计/构建/运行(什么时候)

制品：平台/系统/地点(在哪操作)

响应：对修改内容进行查找/测试/部署

响应度量：造成的金额/时间/人力上造成的影响

数据流图

描述数据流向，不表示加工顺序

包含加工、外部实体、数据流、数据存储部分

构件

独立部署单元，外部不可见状态

可适应不同系统(适应性，装配性)

已编译好的二进制形式

操作不完备：确实部分操作，无法满足调用放

需求过程

**需求开发**：需求获取，需求分析，需求定义，需求验证

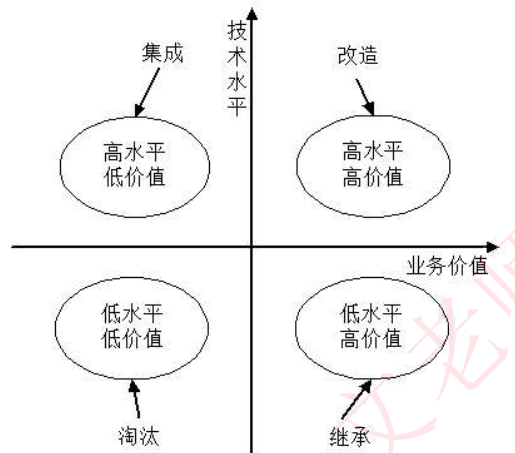
**需求管理**：变更控制、版本控制、需求跟踪和需求状态跟踪

**需求变更管理：**变更描述、变更分析和成本计算、变更实现。变更审批由CCB负责

**变更控制委员会CCB**:对项目产品的变更都可以做出决定

**需求跟踪**：需求和其他元素建关系，

**需求抽取**：迭代管理



## 测试

单元测试：根据详细设计

集成测试：根据概要设计

确认/系统/验收测试： 根据需求文档

AB测试：随机分组访问不同版本

基准性能测试：

压力测试：能有多少并发

可靠性测试：能无故障跑多久

灰盒测试:黑+白，输入/出和内部逻辑都要关注

黑盒测试：不关注代码逻辑；边界值分析，等价类划分，因果图，功能测试

白盒测试：根据代码逻辑测试

语句覆盖：每个语句

判定覆盖：if中整体true/false各一次 (true|true) (false|false)

条件覆盖：if中每个条件都true/false一次(true|false) (false|true)

判定-条件覆盖:满足判定和条件(true|true)(true|false)(false|true)

条件组合覆盖：if中所有组合都一次(true|true)(false|false)(true|false)(false|true)

路径覆盖：所有路径都走边(整体上)

MTTF: 平均两次故障间运行的时间

MTTR:平均修复时间

MTBF:平均故障间隔(MTTF+MTTR)

环路复杂度=边数-节点数+2 或 环数+1

## 法律

国家机密保密期：20年

信息系统安全保护等级：自主保护，审计保护，安全标记保护，结构化保护，访问验证保护

知识产权属于民法

软件著作中的修改权，署名权，保护作品完整权受到永久保护

著作权的保护期限为作者终身+死后50年

发表权只能使用一次

## 数据库

模式

内模式(机器): 分区，页，块，底层结构，索引

模式/概念模式(开发)：约束,表结构

外模式(用户)：视图，数据

实体完整性：主键必填

参照完整性：外建必填

自定义完整：自定义约束(只能是男/女)

自然连接：结果中会去除重复的字段

笛卡尔积

数据库事务特性：原子性，一致性，隔离性，持久性

范式

第一范式：每个字段都不能再分

第二范式：非主键字段必须完全依赖主键(学号key-课程号key, 成绩，课程名称； 要拆成2个表)

第三范式：非主键字段之间不能相互依赖(学号key-院系-院地址； 要拆成两个表)

第四范式：消除非主属性对码的全部依赖

## 杂

SOAP简单对象访问协议，定义客户端和服务端之间的数据规范,接口http调用

双生命周期

领域工程:识别共性、构建资产核心(架构\构建)、

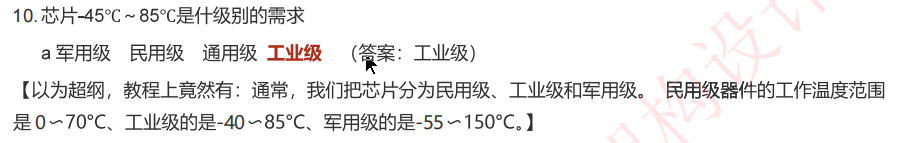
应用工程:通过配置、参数、低码快速生成产品

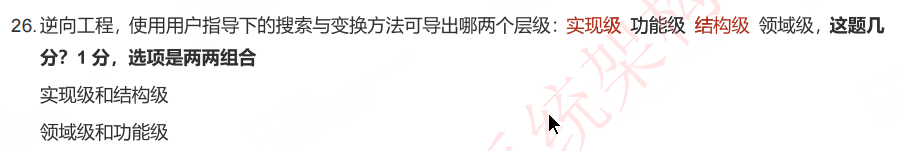
数据集成方式：

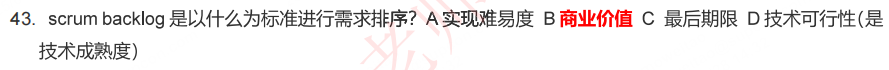
数据联邦：通过虚拟层(中间件)，访问不同数据库中的数据，看起来数据都在一起

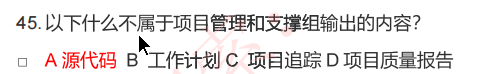
数据复制：将数据复制到数据仓库/湖，有延迟，有冗余

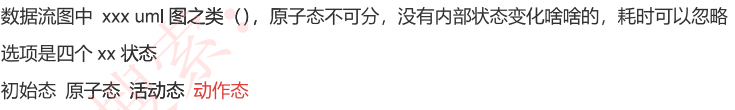
基于接口的数据集成：通过mq/接口将有改动的数据进行更新

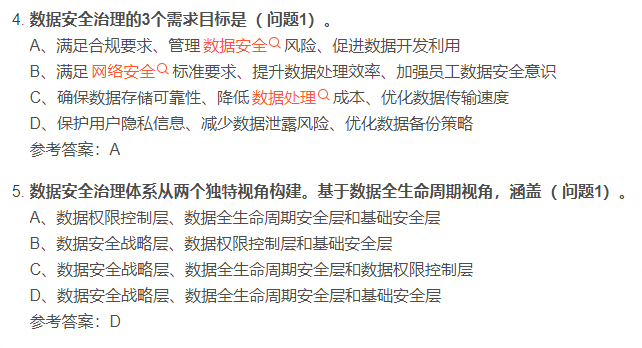


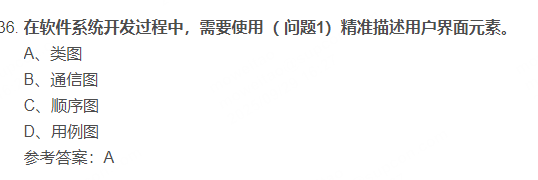


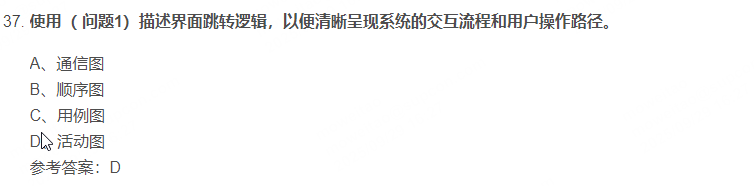


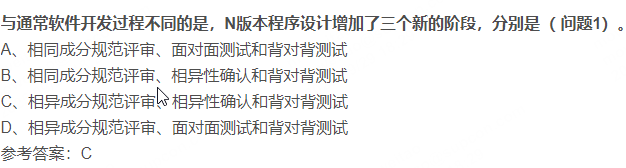


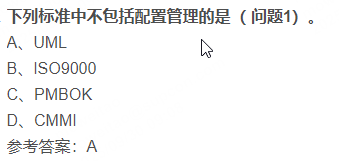


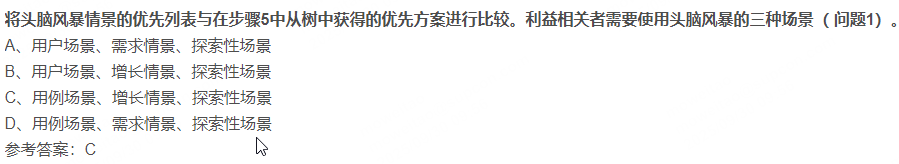


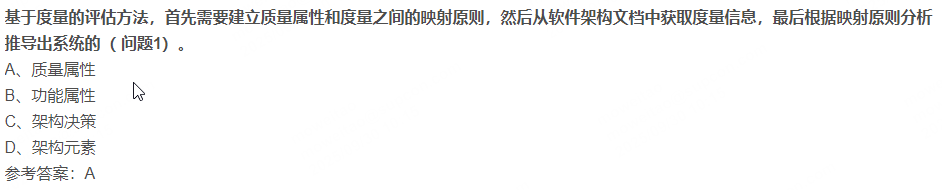




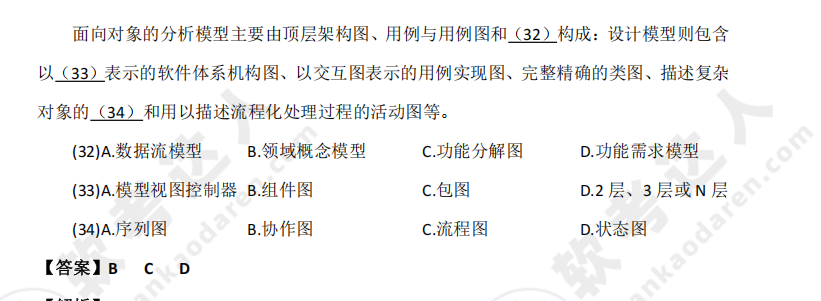


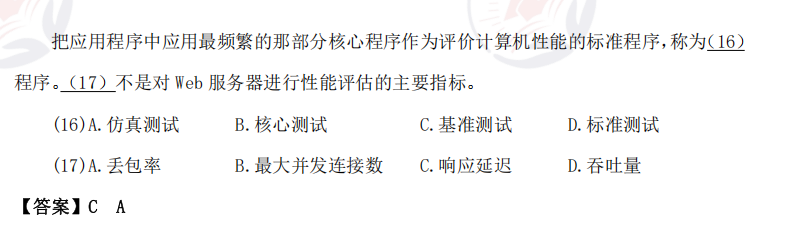


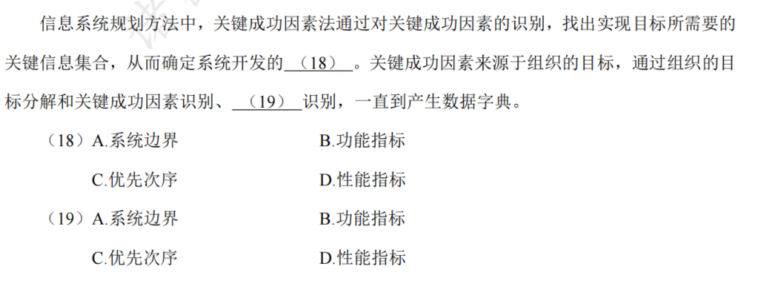




在软件可靠性管理过程中，以下工作不属于需求分析阶段应完成的是（ 𝐶 ）。  
A. 分析可能影响可靠性的因素  
B. 确定软件的可靠性目标  
C. 可靠性建模  
D. 确定可靠性的验收标准





、

CD