第一章 系统工程与信息系统基础

1 考情分析

根据对历年的考试真题进行分析，本章要求考生掌握以下几个方面的知识：

(1) 了解信息与信息化的概念。

(2) 熟悉信息系统战略规划。

(3) 了解信息系统的生命周期。

(4) 了解政府信息化和与电子政务。

(5) 熟悉企业信息化与电子商务。

(6) 了解企业门户。

(7) 掌握企业应用集成。

1.1 本章概要

序号	知识领域	知识点详情
1	系统工程（占）	系统工程
1	信息系统的生命周期（占）	信息系统的生命周期
1	信息系统开发方法（占占）	信息系统开发方法
1	信息系统的分类（占占占）	信息系统的分类
1	政府信息化与电子政务（占占）	政府信息化与电子政务
1	企业信息化与电子商务（占占）	企业信息化与电子商务
1	数字化转型与智能制造（占占）	数字化转型与智能制造

2 考点精讲

2.1 系统工程（占）

2.1.1 系统工程概念

【系统工程】是一种组织管理技术。

【系统工程】是为了最好的实现系统的目的，对系统的组成要素、组织结构、信息流、控制机构进行分析研究的科学方法。

【系统工程】从整体出发、从系统观念出发，以求**【整体最优】**。

【系统工程】利用计算机作为工具，对系统的结构、元素、信息和反馈等进行分析，以达到最优规划、最优设计、最优管理和最优控制的目的。

【系统工程方法】是一种现代的科学决策方法。

2.1.2 系统工程方法

系统工程方法	关键点
霍尔三维结构 “硬科学”方法论	**逻辑维**：逻辑维即解决问题的**逻辑过程**。 **时间维**：时间维即**工作进程**。 **知识维**：知识维即**专业科学知识**。 应用场景：组织和管理大型工程建设项目。
切克兰德方法 “软科学”方法论	核心不是“最优化”，而是**“比较”和“探寻”**。 7 步骤：**认识问题、根底定义、建立概念模型、比较及探寻、选择、设计与实施、评估与反馈**
并行工程方法	**“制造过程”与“支持过程”并行。** 强调三个方面：产品设计开发期间，最快速度按质完成；各项工作问题协调解决；适当的信息系统工具。
综合集成法	钱学森命名，**【简单系统】和【巨系统】** 四原则：整体论原则、相互联系原则、有序性原则、动态原则
WSR 系统方法	实践准则：**【懂物理】-【明事理】-【通人理】**

霍尔三维结构内容：

逻辑维

1、明确问题
2、确定目标
建立价值体系或评价体系
3、系统综合
4、系统分析
5、优化
系统方案的优化选择
6、系统决策
7、实施计划

时间维

1、规划阶段
调研，谋求活动的规划与战略
2、拟定方案
提出具体的计划方案
3、研制阶段
完成研制方案及生产计划
4、生产阶段
生产零部件及提出安装计划
5、安装阶段
安装完毕，完成系统的运行计划
6、运行阶段
系统按照预期的用途开展服务
7、更新阶段
改进原有系统、或消亡原有系统

知识维

工程、医药
建筑、商业
法律、管理

2.2 信息系统的生命周期（占）

2.2.1 系统工程生命周期阶段

探索性研究→概念阶段→开发阶段→生产阶段→使用阶段→保障阶段→退役阶段

2.2.2 系统工程生命周期方法

计划驱动方法：需求→设计→构建→测试→部署

渐进迭代式开发：提供连续交付以达到期望的系统。

精益开发：起源于丰田，是一个动态的、知识驱动的，以客户为中心的过程。

敏捷开发：更好的灵活性。

2.2.3 信息系统的生命周期

2.2.4 信息系统建设原则

1.高层管理人员介入原则，如：CIO 介入

2.用户参与开发原则，用户确定范围、核心用户全程参与、用户深度参与。

3.自顶向下规划原则，以此减少信息不一致的现象。

4.工程化原则，引入【软件工程】

5.其它原则：创新性原则、整体性原则、发展性原则、经济性原则

2.3 软件开发方法（占占）

（1）结构化开发方法

自顶向下，逐步分解（求解），严格区分工作阶段，每阶段有任务与成果，强调系统开发过程的整体性和全局性，开发目标清晰化，工作阶段程式化，开发文档规范化，设计方法结构化。

优点：

理论基础严密，它的指导思想是用户需求在系统建立之前就能被充分了解和理解。由此可见，结构化方法注重开发过程的整体性和全局性。

缺点：

开发周期长；文档、设计说明繁琐，工作效率低；

要求在开发之初全面认识系统的信息需求，充分预料各种可能发生的变化，但这并不十分现实；若用户参与系统开发的积极性没有充分调动，就会造成系统交接过程不平稳，使系统运行与维护管理难度加大。

阶段固化，应变能力差，适用于需求明确的开发场景。

（2）原型法开发方法

适用于**需求不明确的开发**，按**功能**分为水平原型（界面）、垂直原型（复杂算法）；按**最终结果**分为抛弃式原型、演化式原型。原型法的特点在于原型法对用户的需求是动态响应、逐步纳入的，系统分析、设计与实现都是随着对一个工作模型的不断修改而同时完成的，相互之间并无明显界限，也没有明确分工。系统开发计划就是一个反复修改的过程。适于用户需求开始时定义不清、管理决策方法结构化程度不高的系统开发，开发方法更易被用户接受；但如果用户配合不好，盲目修改，就会拖延开发过程。

抛弃型原型（Throw-It-Away Prototype），此类原型在系统真正实现以后就放弃不用了。

演化型原型（Evolutionary Prototype），此类原型的构造从目标系统的一个或几个基本需求出发，通过修改和追加功能的过程逐渐丰富，演化成最终系统。

（3）面向对象方法

最早来源于仿真领域，其特点是系统的描述及信息模型的表示与客观实体相对应，符合人们的思维习惯，有利于系统开发过程中用户与开发人员的交流和沟通，缩短开发周期，提供系统开发的准确性和效率。具有更好的复用性，关键在于建立一个全面、合理、统一的模型，分析、设计、实现三个阶段界限不明确。

OMT 是面向对象建模技术（UML 前身）。用 OMT 方法开发软件，通常需要建立三种形式的模型：对象模型（描述系统数据结构）、动态模型（描述系统控制结构）、功能模型（描述系统功能）。

对象模型：

对象模型表示静态的、结构化的系统的“数据”性质。对象模型是对模拟客观世界实体的对象以及对象彼此间的关系的映射，描述了系统的静态结构。

动态模型：

动态模型表示瞬时的、行为化的系统的“控制”性质，规定了对象模型中对象的合法变化序列。即对象的动态行为。用状态图来描绘对象的状态、触发状态转换的事件、以及对象的行为（对事件的响应）。每个类的动态行为用一张状态图来描绘，各个类的状态图通过共享事件合并起来，从而构成系统的动态模型。

功能模型：

功能模型表示变化的系统的“功能”性质，它指明了系统应该“做什么”，故更直接反映了用户对目标系统的需求。功能模型也通常由一组数据流图表示。在面向对象方法中，数据流程图没有在结构化分析中重要，有时可以省略。

（4）面向服务的方法

以粗粒度、松散耦合的系统功能为核心，强调系统功能的标准化和构件化，加强了系统的灵活性、可复用性和可演化性。

OO 的应用构建在类和对象之上，随后发展起来的建模技术将相关对象按照业务功能进行分组，就形成了构件（Component）的概念。对于跨构件的功能调用，则采用接口的形式暴露出来。进一步将接口的定义与实现进行解耦，则催生了服务和面向服务（Service-Oriented，SO）的开发方法。

SO 方法有三个主要的抽象级别，分别是操作、服务和业务流程。位于最低层的操作代表单个逻辑单元的事物，执行操作通常会导致读、写或修改一个或多个持久性数据。服务的操作类似于对象的方法，它们都有特定的结构化接口，并且返回结构化的响应；位于第二层的服务代表操作的逻辑分组；最高层的业务流程则是为了实现特定业务目标而执行的一组长期运行的动作或活动，包括依据一组业务规则按照有序序列执行的一系列操作。其中操作的排序、选择和执行成为服务或流程的编排，典型的情况是调用已编排的服务来响应业务事件。

（5）其他开发方法

形式化方法（净室软件工程【受控污染级别的环境】；**数学模型化**：所有东西均可证明/验证，而不是测试）、统一过程方法、敏捷方法、基于架构的开发方法（ABSD）。

2.4 信息系统的分类（占占占）

信息系统的分类	关键点
业务处理系统【TPS】	早期最初级的信息系统【20 世纪 50-60 年代】。 功能：数据输入、**数据处理【批处理、OLTP】**、数据库维护、文件报表产生。
管理信息系统【MIS】	高度集成化的人机信息系统。 金字塔结构：分多个层级。
决策支持系统【DSS】	由**语言系统、知识系统和问题处理**系统组成。 用于辅助决策、支持决策。
专家系统【ES】	知识 + 推理 = 专家系统。人工智能的一个重要分支。
办公自动化系统【OAS】	由计算机设备、办公设备、数据通信及网络设备、软件系统组成。

企业资源计划【ERP】	打通供应链，集成，整合。

（1）业务处理系统【TPS】

【业务处理系统（Transaction Processing System, TPS）】又可称为**电子数据处理系统**（Electronic Data Processing System, EDPS），是计算机在管理方面早期应用的最初级形式的信息系统。TPS 是服务于组织管理层次中**最低层、最基础的信息系统**。

（2）管理信息系统【MIS】

【管理信息系统（Manage Information System,MIS）】是由业务处理系统发展而成的，是在 TPS 基础上**引进大量管理方法对企业整体信息进行处理**，并利用信息进行预测、控制、计划、辅助企业全面管理的信息系统。

MIS 系统四大部件：**信息源、信息处理器、信息用户和信息管理者**。

闭环结构在决定过程中，不断收集信息，不断发送给决策者。

批处理系统属于【开环系统】、计算机实时处理系统属于【闭环系统】

（3）决策支持系统【DSS】

【决策支持系统（Decision Support System, DSS）】是一个由**语言系统、知识系统和问题处理系统**3个互相关联的部分组成的，基于计算机的系统。

DSS 应具有的特征：

1.数据和模型是 DSS 的主要资源。

2.DSS 用来支援用户做决策而不是代替用户做决策。

3.DSS 主要用于**解决半结构化及非结构化问题**。

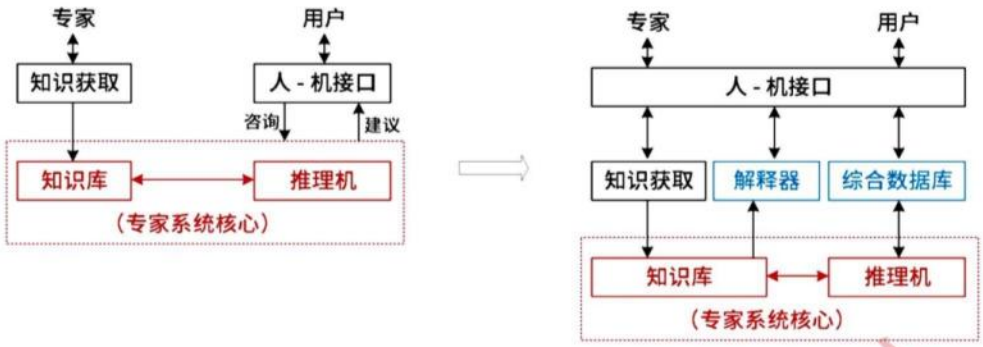
4.DSS 的作用在于提高决策的有效性而不是提高决策的效率。

（4）专家系统【ES】

【专家系统（Expert System, ES）】是一个智能计算机程序系统，其内部含有某个领域具有**专家水平的大量知识与经验**，能够利用人类专家的知识和解决问题的方法来处理该领域的问题。

系统	专家系统	一般计算机系统
功能	解决问题、解释结果、**进行判断与决策**	解决问题
处理能力	处理数字与符号	处理数字
处理问题种类	多属准结构性或非结构性，**可处理不确定的知识，适用于特定的领域**	多属结构性，处理确定的知识

组成：



知识库：存储求解实际问题的领域知识。

综合数据库：存储问题的状态描述、中间结果、求解过程的记录等信息。

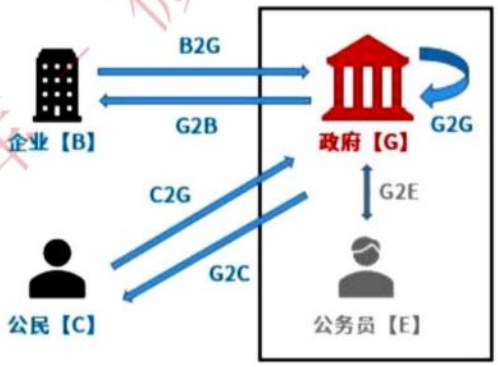
推理机：实质是【规则解释器】。

知识获取：两方面功能——知识的编辑求精及知识自学习。

解释程序：面向用户服务的。

2.5 政府信息化与电子政务（占占）

电子政务主要有3类角色：**政府（Government）、企（事）业单位（Business）及公民（Citizen）。**如果有第4类就是公务员（Employee）。



类型	应用
G2G	基础信息的采集、处理和利用，如：人口信息、地理信息。及政府间：计划管理、财务管理、通信系统、各级政府决策支持。
G2B	**政府给企业单位颁发【各种营业执照、许可证、合格证、质量认证】。**
B2G	企业向政府**缴税；** 企业向政府供应各种商品和服务【含竞/投标】；

	企业向政府**提建议，申诉。**
G2C	社区公安和水、火、天灾等与公共安全有关的信息； **户口、各种证件和牌照的管理。**
C2G	个人应向政府缴纳的各种税款和费用； 个人向政府反馈民意**【征求群众意见】；** 报警服务（盗贼、医疗、急救、火警等）。
G2E	政府内部管理系统。

2.6 企业信息化与电子商务（占占）

2.6.1 信息化概念

信息化是指在国家宏观信息政策指导下，通过**信息技术开发**、信息产业的发展、信息人才的配置，最大限度地利用信息资源以满足全社会的信息需求，从而加速社会各个领域的共同发展以**推进信息社会的过程**。

信息化的主体是全体社会成员（**政府、企业、团体和个人**），时域是一个**长期过程**，空域是经济和社会的一切领域，手段是**先进社会生产工具**。

2.6.2 企业信息化目的

企业信息化的具体目标是优化企业业务活动使之更加有效，它的**根本目的在于提高企业竞争能力**，使得企业具有平稳和有效的运作能力，对紧急情况和机会做出快速反应，为企业内外部用户提供有价值的信息。

2.6.3企业信息化涉及三类创新

【技术创新】在生产工艺设计、产品设计中使用计算机辅助设计系统，**并通过互联网及时了解和掌握创新的技术信息，加快从技术向生产的转化。**还有，生产技术与信息技术相结合，能够大幅度地提高技术水平和产品的竞争力。

【管理创新】按照市场发展的要求，要对企业现有的管理流程重新整合，从作为**管理核心的财务、资金管理，转向技术、物资、人力资源的管理，并延伸到企业技术创新**、工艺设计、产品设计、生产制造过程的管理，进而扩展到客户关系管理、供应链的管理乃至发展到电子商务。

【制度创新】那些不适应企业信息化的管理体制、管理机制和管理制度必须得到创新。

2.6.4 信息化需求的3个层次

组织对信息化的需求是**【组织信息化的原动力】**。

一是**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。从某种意义上来说，信息化对组织不仅仅是服务的手段和实现现有战略的辅助工具；信息化可以把组织战略提升到一个新的水平，为组织带来新的发展契机。特别是对于企业，信息化战略是企业竞争的基础。

二是**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。

三是**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**。也有的组织，原来基本上没有大型的信息系统项目，有的也只是一些单机应用，这样的组织的信息化需求，一般是从头开发新的系统。

组织的三个层次的需求并不是相互孤立的，而是有着内在的联系。信息化需求的获取是一个自上而下的过程，需要对这些需求进行综合分析，才能把握组织对信息化建设的方向。

2.6.5 企业信息化方法

业务流程重构方法：“彻底的、根本性的”**重新设计流程**。

核心业务应用方法：围绕**核心业务**推动信息化。

信息系统建设方法：**建设信息系统**作为企业信息化的重点和关键。

主题数据库方法：建立**面向企业的核心业务的数据库**，消除“信息孤岛”。

资源管理方法：切入点是为**企业资源管理**提供强大的能力。如：ERP、SCM。

人力资本投资方法：人力资本理论【注意不是人力资源管理】**把一部分企业的优秀员工看作是一种资本**，能够取得投资收益。

2.6.6 信息系统战略规划（ISSP）（占占占占）

信息系统战略规划（Information System Strategic Planning，ISSP）是从企业战略出发，构建企业基本的信息架构，对企业内、外信息资源进行统一规划、管理与应用，利用信息控制企业行为，辅助企业进行决策，帮助企业实现战略目标。

ISSP 方法经历了三个主要阶段，各个阶段所使用的方法也不一样。

2.6.6.1 第一个阶段（以数据处理为核心）

主要以数据处理为核心，围绕职能部门需求的信息系统规划，主要的方法包括：

（1）企业系统规划法（**BSP**）--CU 矩阵：自上而下识别系统目标，自下而上设计信息系统，对组织机构的变动具有适应性。

（2）关键成功因素法（**CSF**）：找出实现目标的关键信息集合，从而确定开发优先次序。

（3）战略集合转化法（**SST**）：把战略目标看成“信息集合”，把战略目标转变成信息系统的战略目标。

（4）其它方法包括：投资回收法、征费法、零线预算法、阶石法。

2.6.6.2 第二个阶段（以企业内部管理信息系统为核心）

主要以企业内部管理信息系统为核心，围绕企业整体需求进行的信息系统规划，主要的方法包括战略数据规划法（SDP）、信息工程法（IE）和战略栅格法（SG）；

2.6.6.3 第三个阶段（以集成为核心）

在综合考虑企业内外环境的情况下，以集成为核心，围绕企业战略需求进行的信息系统规划，主要的方法包括价值链分析法（VCA）和战略一致性模型（SAM）。

2.6.7 企业资源计划（ERP）

（1）发展过程

（2）ERP 结构

ERP 是将企业所有资源（企业三大流：物流、资金链、信息流）进行集成整合，全面一体化管理的管理信息系统。

包括三方面：生产控制（计划、制造）、物流管理（分销、采购、库存管理）和财务管理（会计核算、财务管理）。这三个系统本身就是一个集成体，它们相互之间有相应的接口，能够很好地整合在一起。

2.6.8 客户管理 CRM

CRM（Customer Relationship Management）理念：将客户看作资产；客户关怀是中心，**目的是提高收入；核心思想是以客户为中心。**

CRM 的主要模块：

销售自动化；

营销自动化；

客户服务与支持；

商业智能。

CRM 的价值：

提高工作效率，节省开支；

提高客户满意度；

提高客户的忠诚度。

2.6.9 供应链管理 SCM

SCM 理念：**强强联合**，整合与优化“三流”，**打通企业间“信息孤岛”**，严格的数据交换标准。将制造商、供应商、分销商、零售商，在计划（策略性）、采购、制造、配送、退货等各方面联系起来。

信息化的三流

1.信息流【核心】

需求信息流（需方到供方）：如客户订单、生产计划、采购合同等。

供应信息流（供方到需方）：如入库单、完工报告单、库存记录、可供销售量、提货发运单等。

2.资金流【辅助】

3.物流【辅助】

2.6.10 商业智能 BI

（1）过程

需求分析→数据仓库建模→数据抽取→建立 BI 分析报表→用户培训和数据模拟测试→系统改进和完善

（2）相关技术：数据仓库+数据挖掘+OLAP

（3）**用途：决策分析【分析历史数据预判未来】**

（4）数据仓库

数据库	数据仓库【特点】
面向应用：应用组织数据	**面向主题**：主题组织数据
零散的：一个应用对应一个数据库	**集成的**：整个企业对应一个数据仓库
CRUD：增删改查是常态	**相对稳定的（非易失的）**： 查询为主、基本无修改与删除
解决当下应用问题	**反映历史变化（时变的）**： 各个阶段信息都有，并可预测未来趋势

（5）数据挖掘：分类：

关联分析：挖掘出隐藏在数据间的相互关系。

序列模式分析：侧重点是分析数据间的前后关系（因果关系）。

分类分析：为每一个记录赋予一个标记再按标记分类。

聚类分析：分类分析法的逆过程。

（6）**数据湖**

概念：数据湖是一个存储企业的各种各样原始数据的大型仓库，其中的数据可供存取、处理、分析及传输。

特点：数据湖从企业的多个数据源获取原始数据，并且针对不同的目的，同一份原始数据还可能有多种满足特定内部模型格式的数据副本。因此，数据湖中被处理的数据可能是任意类型的信息，从结构化数据到完全非结构化数据。

区别：数据仓库仅支持分析处理，数据湖既支持分析处理，也支持事务处理。

比较：

维度	数据仓库	数据湖
数据	清洗过的数据 结构化的数据	原始数据 结构化，半结构化数据
模式	数据存储之前定义数据模式 数据集成之前完成大量工作 数据的价值提前明确	数据存储之后定义数据模式 提供敏捷，简单的数据集成 数据的价值尚未明确
存取方法	标准 SQL 接口	应用程序，类 SQL 的程序
优势	多数据源集成 干净，安全的数据 转换一次，多次使用	无限扩展性 并行执行 支持编程框架 数据经济

2.6.11 业务流程重组 BPR

BPR 是对企业的业务流程进行根本性的再思考和彻底性的再设计，从而获得可以用诸如成本、质量、服务和速度等方面的业绩来衡量的显著性的成就。

2.6.12 业务流程管理 BPM

BPM 是一种以规范化的构造端到端的卓越业务流程为中心，以持续的提高组织业务绩效为目的的系统化方法。

BPM 与 BPR 区别：

BPM 与 BPR 管理思想最根本的不同就在于流程管理并不要求对所有的流程进行再造。构造卓越的业务流程并不是流程再造，而是根据现有流程的具体情况，对流程进行规范化的设计。

2.6.13 企业应用集成

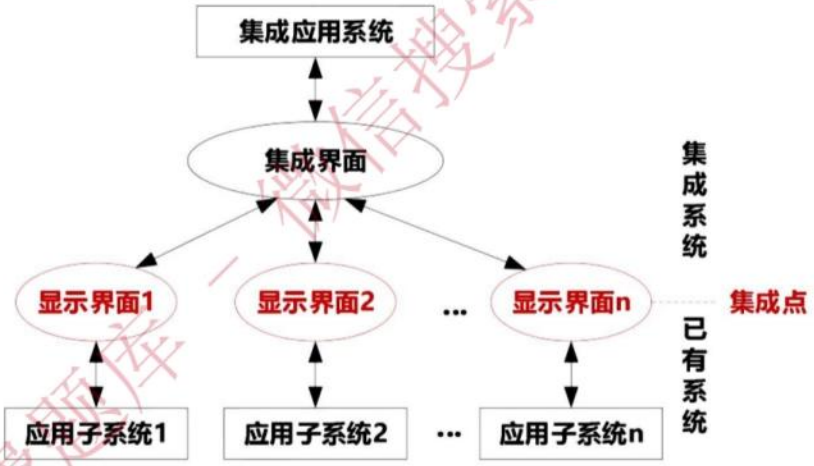
2.6.13.1企业集成分类

按集成点分：

	集成点	效果	解题关键点
界面集成	界面	统一入口，产生“整体”感觉	“整体”感觉 最小代价实现一体化操作
数据集成	数据	不同来源的数据逻辑或物理上“集中”	其他集成方法的基础
控制集成	应用逻辑	调用其他系统已有方法，达到集成效果	
业务流程集成（过程集成）	应用逻辑	跨企业，或优化流程而非直接调用	企业之间的信息共享能力
门户集成		将内部系统对接到互联网上	发布到互联网上

（1）表示集成（界面集成）

把各应用系统的界面集成起来，统一入口，产生“整体”感觉。



（2）数据集成

数据集成是应用集成和业务过程集成的基础。把不同来源、格式、特点性质的数据在逻辑上或物理上有机地集中，从而为企业提供全面的数据共享。ETL、数据仓库、联邦数据库都可视为数据集成。

（3）控制集成（功能集成、应用集成、API 集成）

业务逻辑层次集成，可以借助于远程过程调用或远程方法调用、面向消息的中间件等技术。

（4）业务流程集成（过程集成，B2B）

进行业务流程集成时，企业必须对各种业务信息的交换进行定义、授权和管理，以便改进操作、减少成本、提高响应速度。

按传输方式分：

	特点
消息集成	**数据量小**，交互频繁，立即地，**异步**
共享数据库	交互频繁，立即地，**同步**
文件传输	**数据量大**，交互频度小，即时性要求低（月末，年末）

EAI 提供4个层次的服务（从高层到低层）：

EAI 提供 4 个层次的服务	功能
流程控制服务	解决人工参与的长期的工作流程控制问题
应用连接服务	应用连接适配器将应用接口连接至 EAI 平台
信息传递与转化服务	负责传递消息和转化消息
通讯服务	通过通讯中间件进行消息的路由

企业集成技术的架构层次（从高层到低层）：

企业集成技术的架构层次	说明
会聚集成	集成化运行
应用集成	语用互操作 模式：集成适配器、集成信使、集成面板和集成代理4种
数据集成	语义互通 模式：数据联邦、数据复制、基于接口的数据集成
网络集成	语法互联

2.6.13.2 企业门户

企业**信息门户**（EIP，Enterprise Information Portal）：使员工/合作伙伴/客户/供应商都能够访问企业内部网络和因特网存储的各种自己所需的信息。**【统一访问入口】**

企业**知识门户**（EKP，Enterprise Knowledge Portal）：企业网站的基础上增加知识性内容。**【企业知识库】**

企业**应用门户**（EAP，Enterprise Application Portal）：以商业流程和企业应用为核心，把商业流程中功能不同的应用模块通过门户技术集成在一起。**【企业信息系统的网上集成界面】**

垂直门户：为某一特定的行业服务的，传送的信息只属于人们感兴趣的领域。

2.6.14 电子商务分类

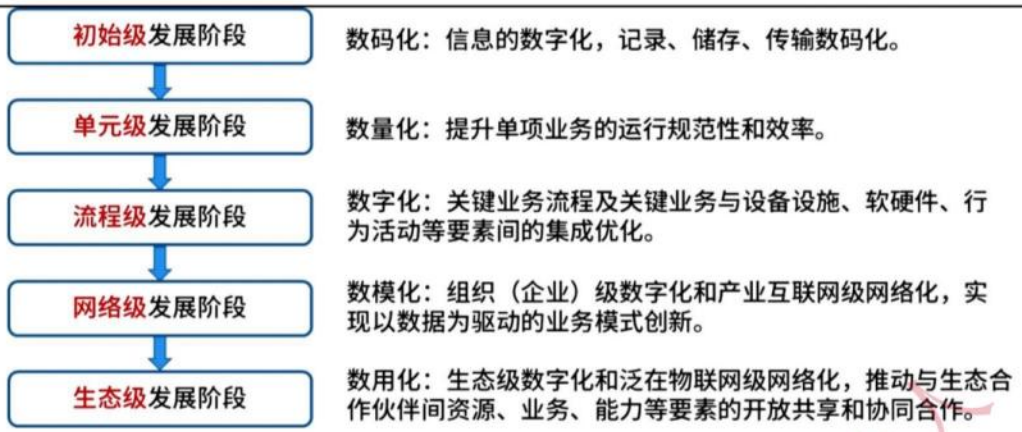
电子商务主要有2类角色：**企业（Business）及个人（Customer）**

类型	应用
B2B	1688
C2C	闲鱼
B2C	京东，天猫
C2B	个人给企业提供咨询服务
O2O（Online To Offline） 【线上对线下】	团购

2.7 数字化转型与智能制造（占占）

【数字化】是新一代信息技术真正的实现推动整个【商业模式的变革】，推动产业链的重构，推动改进企业与消费者之间的关系，以及企业与合作伙伴之间的关系。

企业数字化转型的五个发展阶段分别为：初始级发展阶段、单元级发展阶段、流程级发展阶段、网络级发展阶段、生态级发展阶段。



智能制造体系架构中，系统层级是指与企业生产活动相关的组织结构的层级划分，包括设备层、单元层、车间层、企业层和协同层。

（1）设备层是指企业利用传感器、仪器仪表、机器、 装置等，实现实际物理流程并感知和操控物理流程的层级；

（2）单元层是指用于企业内处理信息、实现监测和控制物理流程的层级；

（3）车间层是实现面向工厂或车间的生产管理的层级；

（4）企业层是实现面向企业经营管理的层级；

（5）协同层是企业实现其内部和外部信息互联和共享， 实现跨企业间业务协同的层级。

3 章节问答

（1）信息化战略和信息系统战略，这两者区别在哪里？

答：信息化战略会包括软件硬件网络的整体规划，而信息系统主要是指的软件系统这一块。可以理解为信息系统战略支撑着信息化战略，信息化战略全局大于信息系统战略。

（2）政府采集居民信息，是政府对居民还是政府对政府的电子政务活动？

答：国家和地方人口信息的采集、处理和利用，属于政府对政府的电子政务活动。关键是看国家和地方人口信息这个系统是给谁服务的。如果从人口信息的处理和利用这个角度来看就很容易了。

（3）面向信息集成、面向过程集成、面向服务集成与界面集成、数据集成、应用集成、过程集成有什么区别和联系？

答：企业内的集成按照集成的层次进行分类可分为：界面集成、平台集成、数据集成、应用集成、过程集成等。

按照集成技术可分为：面向信息的集成、面向过程的集成和面向服务的集成。

对于二者的联系：一般而言，界面集成、平台集成、数据集成、应用集成可通过面向信息的集成技术实现;过程集成可通过面向过程的集成技术来实现,但其中又常包含接口集成等面向信息的集成技术。当然，应用集成和过程集成也可以通过面向服务的集成技术来实现。

(4) 企业信息门户和企业应用门户的区别是什么?

答：企业应用门户强调的是对企业业务流程的集成。它是以业务流程和企业应用为核心的，把业务流程中功能不同的应用模块通过门户技术集成在一起的。

(5) 为什么说事务型数据库容易形成数据孤岛，而主题数据库就不容易?

答：信息孤岛就是信息像孤岛一样被隔离开了，丧失了信息的联动性，也就丧失了信息的意义和价值。主题数据库，顾名思义，这种数据库是面向主题的，根据不同的业务主题来进行组织和存储。这种设计使得主题数据库的数据能够更广泛的被各种不同的应用程序使用和共享,因此它比事务型数据库更不容易形成数据孤岛。

第二章 软件工程

1 考情分析

根据对历年的考试真题进行分析，本章要求考生掌握以下几个方面的知识：

(1) 了解软件开发方法，掌握软件开发模型。

(2) 了解需求工程概念，熟悉面向对象分析，掌握结构化分析，掌握需求管理过程。

(3) 了解系统建模过程。

(4) 了解系统设计的过程，了解人机界面设计和结构化设计，熟悉面向对象设计。

(5) 了解软件测试的概念和目标，了解软件测试的原则，熟悉基本的软件测试活动，掌握软件测试的阶段和分类，了解面向对象测试。

(6) 了解软件调试方法，区分软件调试与软件测试。

(7) 掌握系统转换计划，了解系统维护的概念。

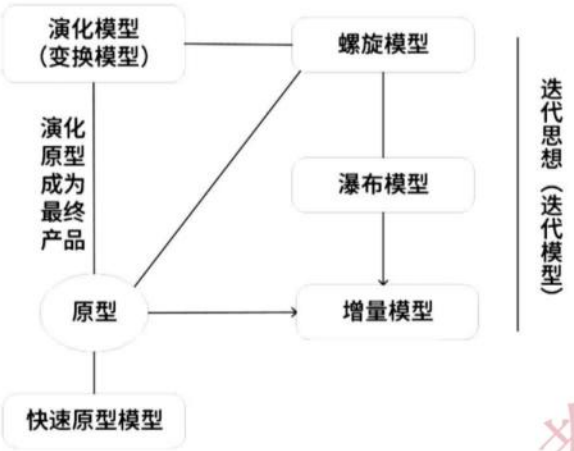
(8) 了解系统工程建模相关知识。

1.1 本章概要

序号	知识领域	知识点详情
1	软件过程模型（占占占占）	软件过程模型
1	基于构件的软件工程（占占）	基于构件的软件工程
2	逆向工程（占）	逆向工程
3	净室软件工程（占）	净室软件工程
1	需求工程（占占）	需求工程
1	系统建模过程（占）	系统建模过程
1	系统设计（占占）	人机界面设计
2		结构化设计
3		面向对象设计
1	软件测试（占占）	软件测试
1	系统运行与软件维护（占）	系统转换计划
2		系统维护
1	系统工程建模（占占）	系统工程建模

2 考点精讲

2.1 软件过程模型（占占占占）



（1）原型模型

典型的原型开发方法模型。适用于**需求不明确**的场景，可以帮助用户明确需求。可以分为**【抛弃型原型】与【演化型原型】**

原型模型两个阶段：

1、原型开发阶段；

2、目标软件开发阶段。

（2）**瀑布模型**

瀑布模型是将软件生存周期中的各个活动规定为线性顺序连接的若干阶段的模型，包括需求分析、软件设计、程序设计、编码实现、单元测试、集成测试、系统测试、运行维护。

瀑布模型的特点是严格区分阶段，**每个阶段因果关系紧密相连**，只适合**需求明确**的项目。

缺点：软件需求完整性、正确性难确定；

严格串行化，很长时间才能看到结果；

瀑布模型**要求每个阶段一次性完全解决该阶段工作**，这不现实。

（3）增量模型

融合了瀑布模型的基本成分和原型实现的迭代特征，可以有多个可用版本的发布，核心功能往往最先完成，在此基础上，每轮迭代会有新的增量发布，核心功能可以得到充分测试。强调每一个增量均发布一个可操作的产品。

（4）螺旋模型

以**快速原型为基础**+瀑布模型，典型特点是引入了**风险分析**。它是由制定计划、风险分析、实施工程、客户评估这一循环组成的，它最初从概念项目开始第一个螺旋。

（5）V 模型和 W 模型

V 模型强调测试贯穿项目始终，而不是集中在测试阶段。是一种测试的开发模型。

W 模型强调**测试和开发【并行进行】**。

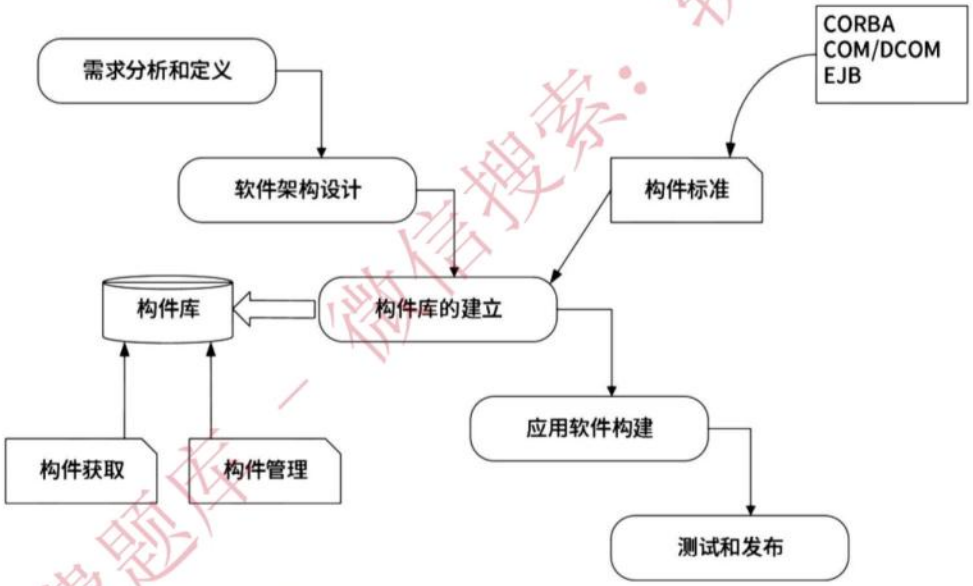
（6）快速应用开发 RAD

概念：RAD 是瀑布模型的一个高速变种，适用比传统生命周期快得多的开发方法，它强调极短的开发周期，通常适用基于构件的开发方法获得快速开发。

过程：业务建模→数据建模→过程建模→应用生成→测试与交付

适用性：RAD 对模块化要求比较高，如果某项功能不能被模块化，则其构件就会出问题；如果高性能是一个指标，且必须通过调整结构使其适应系统构件才能获得，则 RAD 也有可能不能奏效；RAD 要求开发者和客户必须在很短的时间完成一系列的需求分析，任何一方配合不当都会导致失败；RAD 只能用于管理信息系统的开发，不适合技术风险很高的情况。

（7）构件组装模型



【优点】易扩展、易重用、降低成本、安排任务更灵活。

【缺点】构件设计要求经验丰富的架构师、设计不好的构件难重用、强调重用可能牺牲其它指标（如性能）、第三方构件质量难控制。

（8）统一过程（在软考中 UP、RUP 都指统一过程）

典型特点是**用例驱动、以架构为中心、迭代和增量**。

统一过程把一个项目分为四个不同的阶段：

构思阶段（初始/初启阶段）：定义最终产品视图和业务模型、确定系统范围。

细化阶段（精化阶段）：**设计及确定系统架构**、制定工作计划及资源要求。

构造阶段：开发剩余构件和应用程序功能，把这些构件集成为产品，并进行详细测试。

移交阶段：确保软件对最终用户是可用的，进行β测试，制作产品发布版本。

9个核心工作流：业务建模、**需求、分析与设计、实现、测试、部署**、配置与变更管理、项目管理、环境。

（9）敏捷开发

敏捷开发是一种以人为核心、迭代、循序渐进的开发方法，适用于小团队和小项目，具有小步快跑的思想。常见的敏捷开发方法有极限编程法、水晶法、并列争球法和自适应软件开发方法。

极限编程（XP）：一些对费用控制严格的公司中的使用，非常有效，近螺旋式的开发方法。**四大价值观（沟通【加强面对面沟通】、简单【不过度设计】、反馈【及时反馈】、勇气【接受变更的勇气】）**，十二大最佳实践（简单设计、测试驱动、代码重构、结对编程、持续集成、现场客户、发行版本小型化、系统隐喻、代码集体所有制、规划策略、规范代码、40小时工作机制）。

水晶方法：提倡“机动性”的方法，拥有对不同类型项目非常有效的敏捷过程。

开放式源码：程序开发人员在地域上分布很广【其他方法强调集中办公】。

SCRUM：明确定义了可重复的方法过程。

特征驱动开发方法（FDD）：认为有效的软件开发需要3要素**【人、过程、技术】**。定义了6种关键的项目角色：**项目经理、首席架构设计师、开发经理、主程序员、程序员和领域专家。**

ASD 方法：其核心是三个非线性的、重叠的开发阶段：猜测、合作与学习。

动态系统开发方法（DSDM）：倡导以业务为核心。

敏捷宣言：

个体和交互胜过过程和工具；

可工作的软件胜过大量的文档；

客户合作胜过合同谈判；

响应变化胜过遵循计划。

2.2 基于构件的软件工程（CBSE）（占占）

CBSE 体现了“购买而不是重新构造”的哲学。

CBSE 的构件应该具备的特征：

1、**可组装性：**所有**外部交互**必须通过**公开定义的接口**进行。

2、**可部署性：**构件总是二进制形式的，能作为一个**独立实体在平台上运行。**

3、**文档化：**用户根据文档来判断构件是否满足需求。

4、**独立性：**可以在无其他特殊构件的情况下进行组装和部署。

5、**标准化：符合某种标准化的构件模型。**

【构件的组装】（一般都要借助胶水代码）

顺序组装：按顺序调用已经存在的构件，可以用两个已经存在的构件来创造一个新的构件。

层次组装：被调用构件的“提供”接口必须和调用构件的“请求”**接口兼容**。

叠加组装：多个构件合并形成新构件，新构件整合原构件的功能，**对外提供新的接口**。

组装可能出现3种不兼容：**参数不兼容、操作不兼容、操作不完备**。

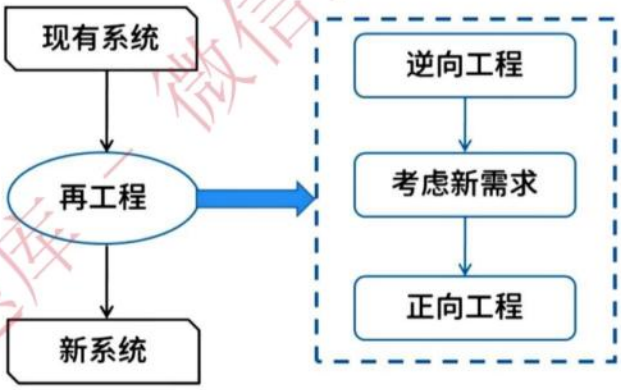
构件模型要素：

【接口】构件通过构件接口来定义，构件模型规定应如何定义构件接口以及在接口定义中应该包含的要素，如操作名、参数以及异常等。

【使用信息】为使构件远程分布和访问，必须给构件一个特定的、全局唯一的名字或句柄。**构件元数据是构件本身相关的数据**，比如构件的接口和属性信息。用户可以通过元数据找到构件提供的服务。构件模型的实现通常包括访问构件的元数据的特定方法。**构件是通用实体，在部署的时候，必须对构件进行配置来适应应用系统。**

【部署】构件模型包括一个规格说明，指出应该如何打包构件使其部署成为一个独立的 可执行实体。**部署信息中包含有关包中内容的信息和它的二进制构成的信息。**

2.3 逆向工程（占）



【逆向工程是设计的恢复过程】

实现级：包括程序的**抽象语法树、符号表、过程**的设计表示。

结构级：包括反映**程序分量之间相互依赖关系**的信息，例如调用图、结构图、程序和数据结构。

功能级：包括反映**程序段功能及程序段**之间关系的信息，例如数据和控制流模型。

领域级：包括反映程序分量或程序诸实体与**应用领域概念之间对应关系**的信息，例如实体关系模型。

与逆向工程相关的概念有重构、设计恢复、再工程和正向工程。

（1）重构/重组（restructuring）。重构是指在**【同一抽象级别】**上转换系统描述形式。

（2）设计恢复（design recovery）。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。

（3）逆向工程（reverse engineering）：逆向工程是分析程序，力图在比源代码更高抽象层次上建立程序的表示过程，**逆向工程是设计的恢复过程**。

（4）正向工程（forward engineering）。正向工程是指不仅从现有系统中恢复设计信息，**而且使用该信息去改变或重构现有系统，以改善其整体质量**。

（5）再工程/重构工程（Re-engineering）。再工程是对现有系统的重新开发过程，**包括逆向工程、新需求的考虑过程和正向工程**三个步骤。

2.4 净室软件工程（占）

净室即无尘室、洁净室。也就是一个**受控污染级别的环境**。

使用盒结构规约（或形式化方法）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。

【技术手段】：

1.统计过程控制下的增量式开发：**控制迭代**

2.基于函数的规范和设计：盒子结构

定义3种抽象层次：行为视图（**黑盒**）→有限状态机视图（**状态盒**）→过程视图（**明盒**）

3.正确性验证：净室工程的核心

4.统计测试和软件认证：使用统计学原理，总体太大时必须采用**抽样方法**。

【缺点】：

太理论化，正确性验证的步骤比较困难且耗时。

开发小组**不进行传统的模块测试，这是不现实的。**

脱胎于传统软件工程，不可避免**带有传统软件工程的一些弊端。**

2.5 需求工程（占占）

2.5.1 需求工程阶段划分

软件需求是指**用户对系统**在功能、行为、性能、设计约束等方面的**期望**。

【需求工程主要活动的阶段划分】

2.5.2 需求开发

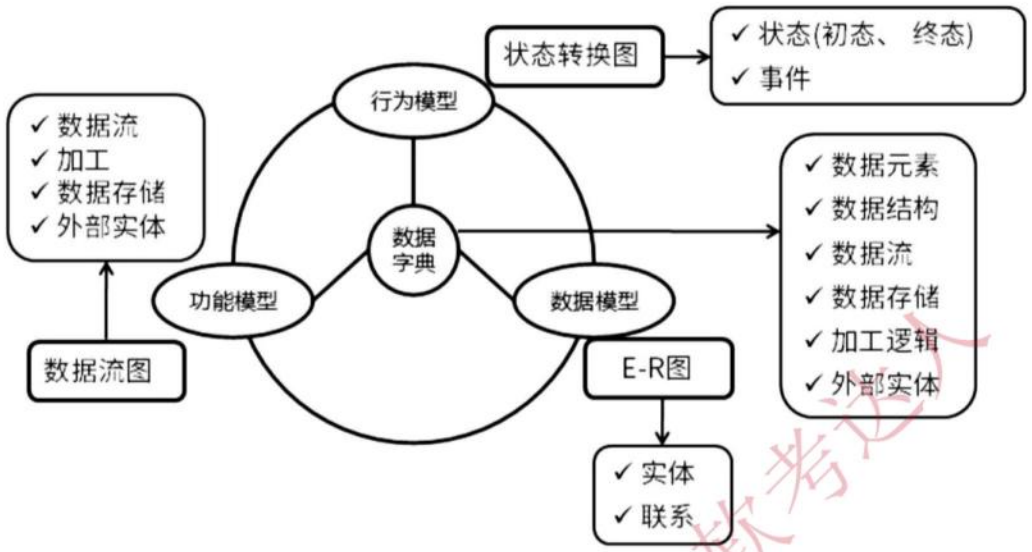
2.5.2.1 需求获取

方法	特点
用户面谈	1 对 1-3，有代表性的用户，了解主观想法，交互好。**成本高，要有领域知识支撑。**
联合需求计划（JRP）	**高度组织的群体会议，各方参与，了解想法，消除分歧，交互好，成本高。**
问卷调查	**用户多**，无法一一访谈，**成本低**。
现场观察	针对较为**复杂的流程**和操作。
原型化方法	通过简易系统方式**解决早期需求不确定问题。**
头脑风暴法	一群人围绕新业务，**发散思维**，不断产生新的观点。

2.5.2.2 需求分析

结构化需求分析（SA）

（1）结构化分析过程



(2) 结构化分析工具-数据流图 DFD

元素	说明	图元
数据流	由一组固定成分的数据组成，表示数据的流向。每个数据流通常有一个合适的名词，反映数据流的含义	
加工	加工描述了输入数据流到输出数据流之间的变换，也就是输入数据流做了什么处理后变成了输出数据流	
数据存储（文件）	用来表示暂时存储的数据，每个文件都有名字。流向文件的数据流表示写文件，流出的表示读文件	
外部实体	指存在于软件系统外的人员或组织	

(3) 数据流图题答题技巧

详细分析试题说明

利用数据平衡原则【父图与子图之间的平衡；子图内平衡】

黑洞：一个加工只有输入数据流而无输出数据流。

奇迹：一个加工只有输出数据流而无输入数据流。

灰洞：一个加工的输入数据流无法通过加工产生输出流。

【补充实体】

人物角色：如客户、管理员、主管、经理、老师、学生

组织机构：如银行、供应商、企业服务产总线（ESB）构

外部系统：如银行系统、工资系统、后台数据库（当要开发的是中间件时）

【补充存储】存储的文字方面特征：“**文件”“**表”“**库”“**清单”“**档案”

【补充数据流】

数据平衡原则：顶层图与0层图对比，是否有顶层图有，但0层图无的数据流，或反之。检查图中每个加工，是否存在只有入没有出，或只有出没有入，或根据输入的数据无法产生对应的输出的情况。

按题目说明与图进行匹配：说明中的每一句话，都能与图中有对应关系，当把说明中的实体与数据流标识出来之后，容易缩小对应范围，找出纰漏。

【补充加工名】

加工是用于处理数据流的，所以要补充加工名，可以把该加工涉及的数据流，在说明中标识出来，再在数据流名称所在的句子中，找“动词+名词”的结构，分析是否可作为加工。

“动词+名词”如：生成报告，发出通知，批改作业，记录分数，当然这只是普遍情况，也有例外，如物流跟踪、用户管理

面向对象需求分析

（1）面向对象基本概念

对象：属性（数据）+方法（操作）+对象 ID

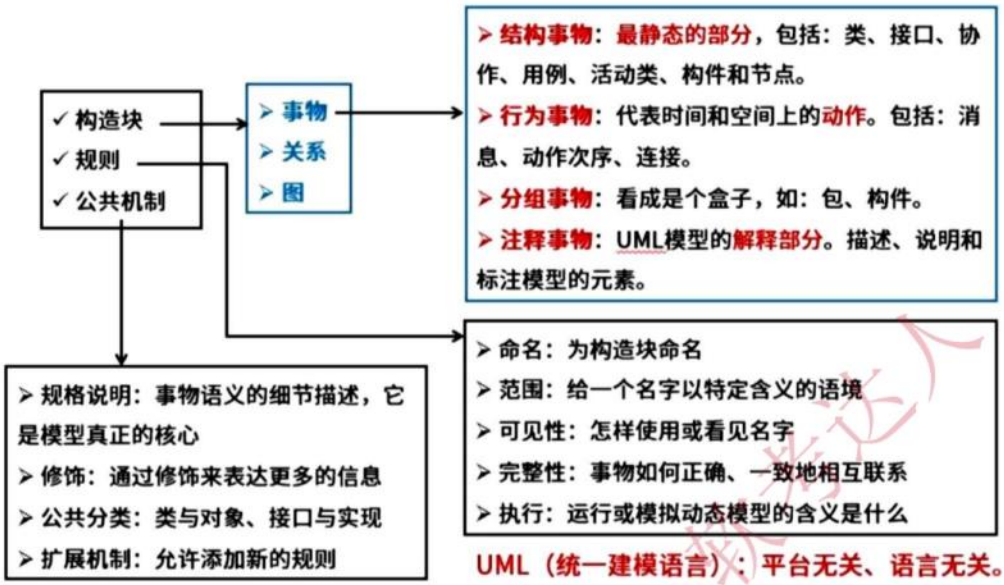
类（实体类/控制类/边界类）

实体类映射需求中的每个实体，实体类保存需要存储在永久存储体中的信息，例如，在线教育平台系统可以提取出学员类和课程类，它们都属于实体类。

控制类是用于控制用例工作的类，一般是由动宾结构的短语（“动词+名词”或“名词+动词”）转化来的名词，例如，用例“身份验证”可以对应于一个控制类“身份验证器”，它提供了与身份验证相关的所有操作。

边界类用于封装在用例内、外流动的信息或数据流。边界类位于系统与外界的交接处，包括所有窗体、报表、打印机和扫描仪等硬件的接口，以及与其他系统的接口。

（2）UML 图概念



UML 包括两组公共分类，分别是类与对象（类表示概念，而对象表示具体的实体）、接口与实现（接口用来定义契约，而实现就是具体的内容）

结构事物：结构事物在模型中属于最静态的部分，代表概念上或物理上的元素。UML 有七种结构事物，分别是类、接口、协作、用例、活动类、构件和节点。类是描述具有相同属性、方法、关系和语义的对象的集合，一个类实现一个或多个接口；接口是指类或构件提供特定服务的一组操作的集合，接口描述了类或构件的对外的可见的动作；协作定义了交互的操作，是一些角色和其它事物一起工作，提供一些合作的动作，这些动作比事物的总和要大；用例是描述一系列的动作，产生有价值的结果。在模型中用例通常用来组织行为事物。用例是通过协作来实现的；活动类的对象有一个或多个进程或线程。活动类和类很相似，只是它的对象代表的事物的行为和其他事物是同时存在的；构件是物理上或可替换的系统部分，它实现了一个接口集合；节点是一个物理元素，它在运行时存在，代表一个可计算的资源，通常占用一些内存和具有处理能力。一个构件集合一般来说位于一个节点，但有可能从一个节点转到另一个节点。

行为事物：行为事物是 UML 模型中的动态部分，代表时间和空间上的**动作**。UML 有两种主要的行为事物。第一种是交互（内部活动），交互是由一组对象之间在特定上下文中，为达到特定目的而进行的一系列消息交换而组成的动作。交互中组成动作的对象的每个操作都要详细列出，包括消息、动作次序（消息产生的动作）、连接（对象之间的连接）；第二种是状态机，状态机由一系列对象的状态组成。

分组事物：分组事物是 UML 模型中组织的部分，可以把它们看成是个盒子，模型可以在其中进行分解。UML 有两种分组事物，包括包和构件。包是一种将有组织的元素分组的机制。与构件不同的是，包纯粹是一种概念上的事物，只存在于开发阶段，而构件可以存在于系统运行阶段。

注释事物：注释事物是 UML 模型的**解释部分**。

（3）UML 图分类

类图（Class diagram）:类图描述一组类、接口、协作和它们之间的关系。

对象图（Object diagram）:对象图描述一组对象及它们之间的关系。对象图描述了在类图中所建立的事物实例的静态快照。

构件图（Component diagram）。构件图描述一个封装的类和它的接口、端口，以及由内嵌的构件和连接件构成的内部结构。构件图用于表示系统的静态设计实现视图。对于由小的部件构建大的系统来说，构件图是很重要的。构件图是类图的变体。一个封装的类和它的接口

部署图（Deployment diagram）。部署图描述对运行时的处理节点及在其中生存的构件的配置。部署图给出了架构的静态部署视图，通常一个节点包含一个或多个部署图。软硬件之间映射。

制品图：系统的物理结构

包图：由模型本身分解而成的组织单元，以及它们之间的依赖关系，包的图标像是一个带标签的文件夹，包的基本思想是把共同工作的元素放到一个文件夹中。例：多个类或构件组成了一个子系统，就可以将它们放到一个包中。

组合结构图

用例图：系统与外部参与者的交互

顺序图（Sequence diagram，序列图）：顺序图是一种交互图（Interaction diagram），它强调对象之间消息发送的顺序，同时显示对象之间的交互。强调按时间顺序。顺序图的组合片段：Loop【循环】：如果满足“循环条件”，则重复执行本框中的内容。Alt【条件分支】：满足条件1，则执行条件1对应的内容，满足条件2则执行条件2对应的内容。Opt【可选分支】：如果条件满足，则执行框中内容，否则跳过不执行。

通信图（Communication diagram）。协作图。通信图也是一种交互图，它强调收发消息的对象或参与者的结构组织。顺序图和通信图表达了类似的基本概念，但它们所强调的概念不同，顺序图强调的是时序，通信图强调的是对象之间的组织结构（关系）。

定时图：强调实际时间

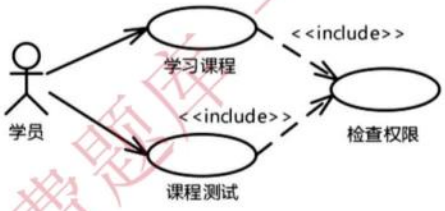
交互概览图

状态图（State diagram）。状态图描述一个状态机，它由状态、转移、事件和活动组成。状态图给出了对象的动态视图。它对于接口、类或协作的行为建模尤为重要，而且它强调事件导致的对象行为，这非常有助于对反应式系统建模。状态转换变迁。

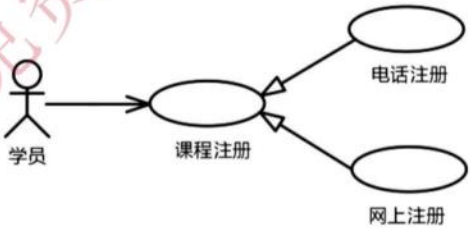
活动图（Activity diagram）。活动图将进程或其他计算结构展示为计算内部一步步的控制流和数据流。活动图专注于系统的动态视图。它对系统的功能建模和业务流程建模特别重要，并强调对象间的控制流程。类似程序流程图，并行行为。

（4）UML 图关系

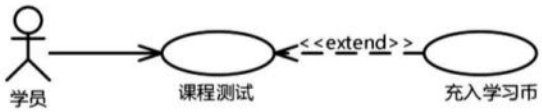
用例关系包括：包含关系、扩展关系、泛化关系。



包含关系



泛化关系



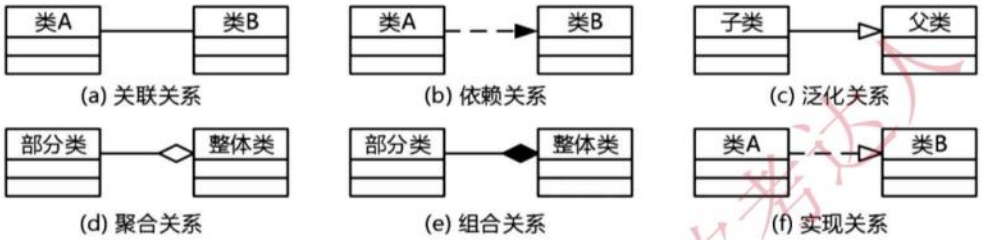
扩展关系

包含关系：其中这个提取出来的**公共用例称为抽象用例**，而把**原始用例称为基本用例**或基础用例，当可以从两个或两个以上的用例中提取公共行为时，应该使用包含关系来表示它们。

扩展关系：如果一个用例明显地混合了两种或两种以上的不同场景，即根据情况可能发生多种分支，则可以将这个用例分为**一个基本用例和一个或多个扩展用例**，这样使描述可能更加清晰。

泛化关系：当多个用例共同拥有一种类似的结构和行为的时候，可以将它们的共性抽象成为父用例，其他的用例作为泛化关系中的子用例。在用例的泛化关系中，子用例是父用例的一种特殊形式，子用例继承了父用例所有的结构、行为和关系。

类图/对象图关系：



依赖关系：一个事物发生变化影响另一个事物。

泛化关系：特殊/一般关系

关联关系：描述了一组链，链是对象之间的连接。

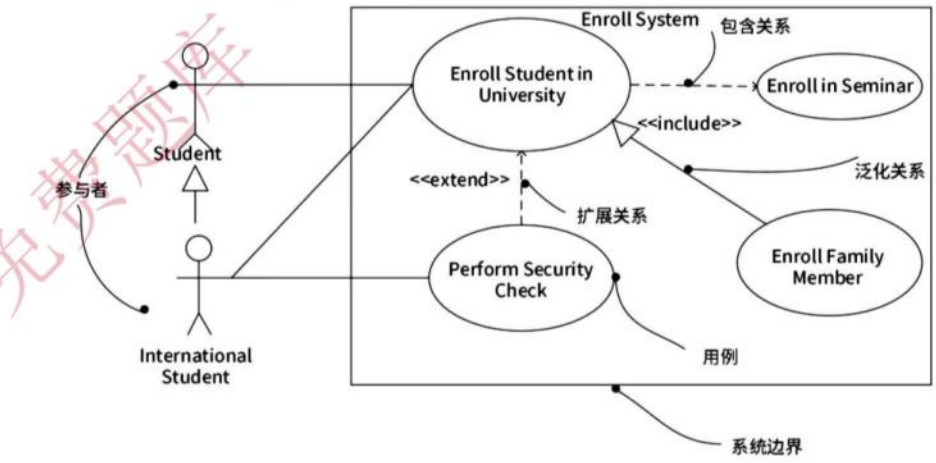
聚合关系：整体与部分生命周期**不同**。

组合关系：整体与部分生命周期**相同**。

实现关系：接口与类之间的关系

(5) 常见 UML 图考查

用例图常见考查形式：



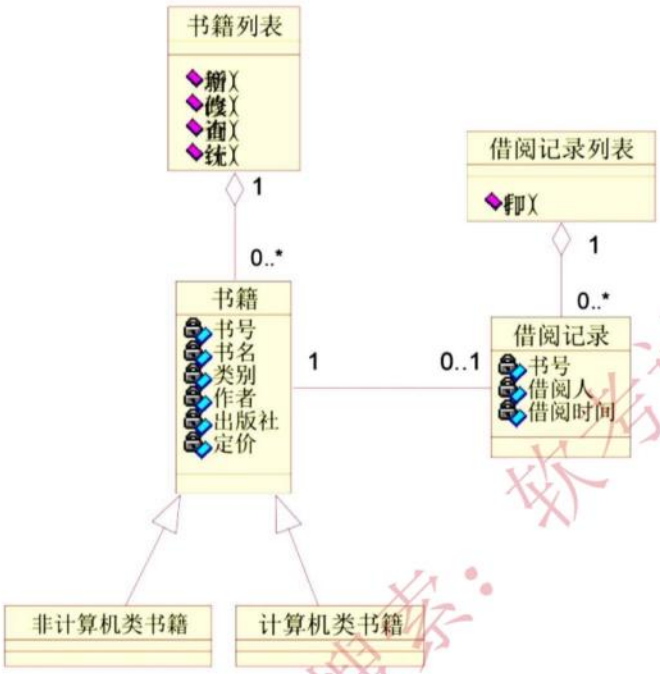
补充参与者：参与者一般为外部实体，可以是人，也可以是外部系统。

补充用例：根据题干描述补充缺失的用例，在补充过程中需要参照用例图中用例间的关系。（用例名从题干中直接获取或提炼）

分析用例间的关系

其他：可能会出现其他题型，比如添加新的用例等。

类图/对象图常见考查形式（常与其他 UML 图结合考查）



补充类名/对象名：一般为名词，来源于题干描述。

补充多重度：多重度体现类与类或对象与对象之间的对应关系，有0…1，1..1，1…*/n，*…*/m…n。

1：表示一个集合中的一个对象对应另一个集合中一个对象。

0..*：表示一个集合中的一个对象对应另一个集合中的0个或多个对象。

（可以不对应）

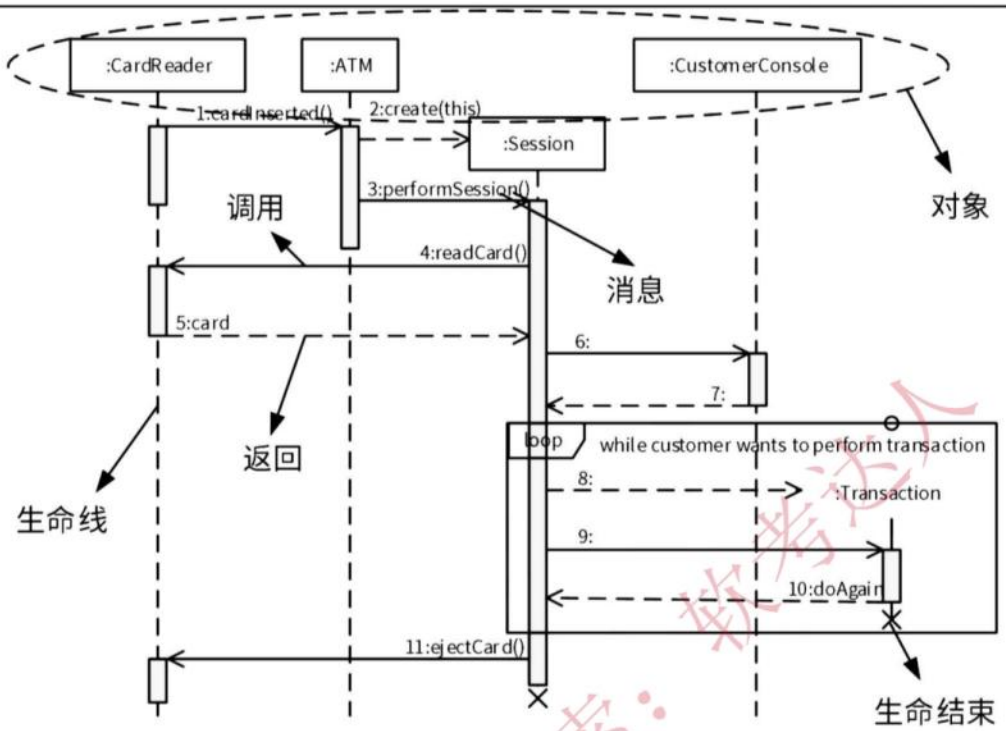
1..*：表示一个集合中的一个对象对应另一个集合中的一个或多个对象。

（至少对应一个）

*：表示一个集合中的一个对象对应另一个集合中的多个的对象。

分析类与类或对象与对象之间的关系。

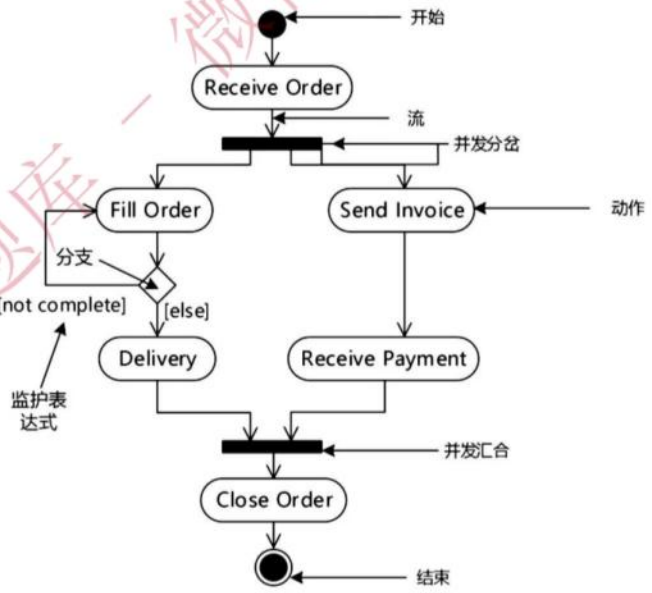
顺序图常见考查形式：



补充类名/对象名

补充消息

活动图常见考查形式



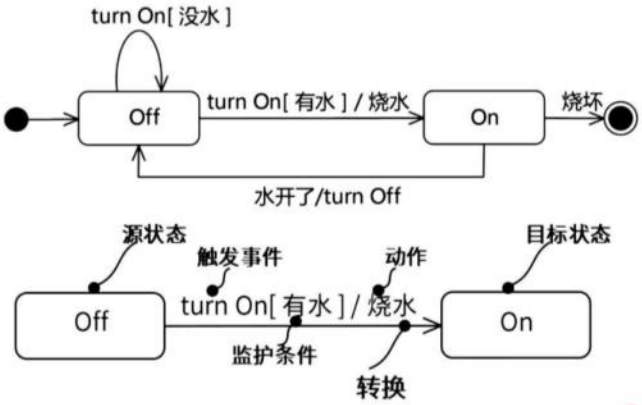
补充动作名称；

补充监护表达式；

分析并发关系；

活动图与状态图对比，活动图与进程图关系。

状态图常见考查形式：

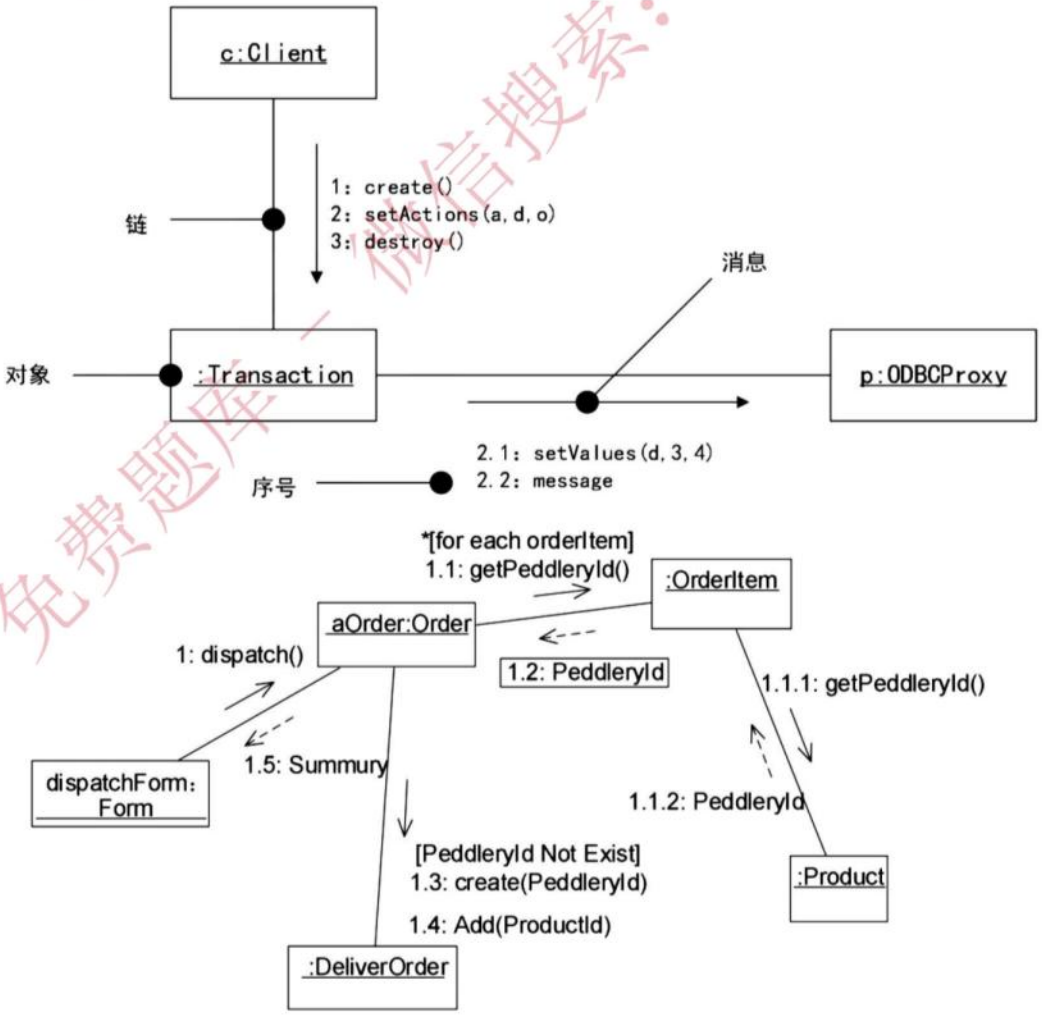


补充状态；

补充触发事件、监护条件、动作；

与状态模式结合考查。

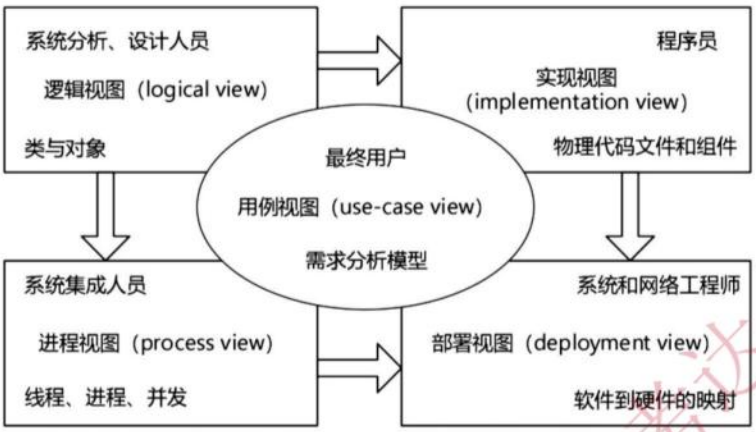
通信图常见考查形式



补充对象名；

补充消息。

(6) “4+1”视图



UML 采用4+1视图来描述软件和软件开发过程：

逻辑视图：以问题域的语汇组成的**类和对象**集合。

进程视图：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，描绘了所设计的**并发**与同步结构。

实现视图：对组成基于系统的物理代码的文件和组件进行建模。

部署视图：把构件部署到一组物理的、可计算的节点上，表示**软件到硬件的映射**及分布结构。

用例视图：最基本的需求分析模型。

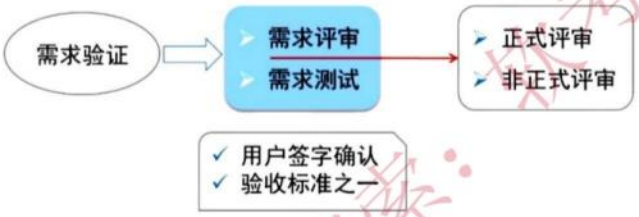
(7) OOA 需求建模

2.5.2.3 需求定义（形成需求规格）

- 所有需求都能够被预先定义
- 开发人员于用户之间能够准确而清晰地交流
- 采用图形/文字可以充分体现最终系统

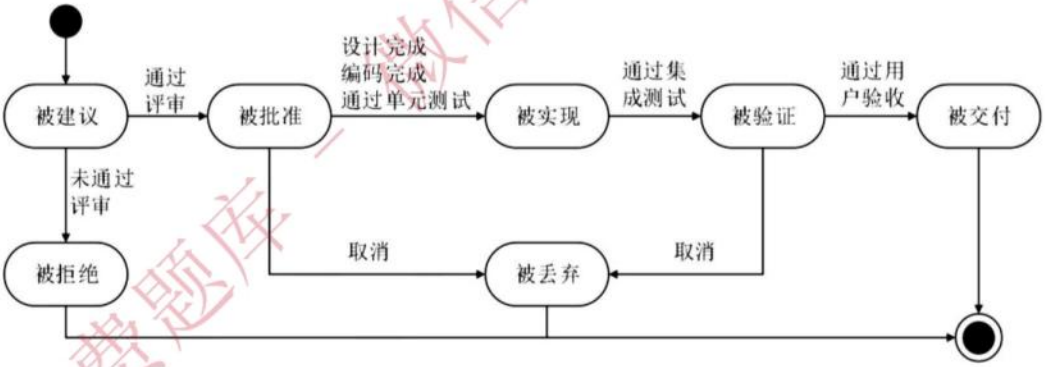
- 并非所有的需求都能在开发前被准确的说明
- 项目参加者之间通常都存在交流上的困难
- 需要实际的、可供用户参与的系统模型
- 有合适的系统开发环境
- 反复是完全需要和值得提倡的，需求一旦确定，就应遵从严格的方法

2.5.2.4 需求确认与验证



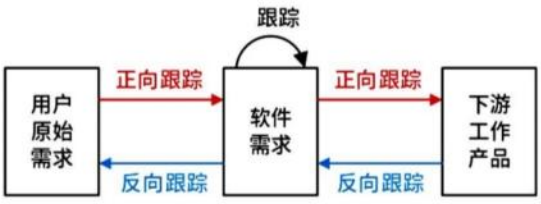
2.5.3 需求管理

(1) 定义需求基线



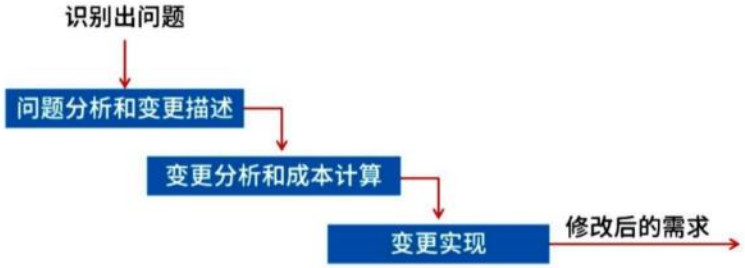
需求的状态

(2) 需求跟踪



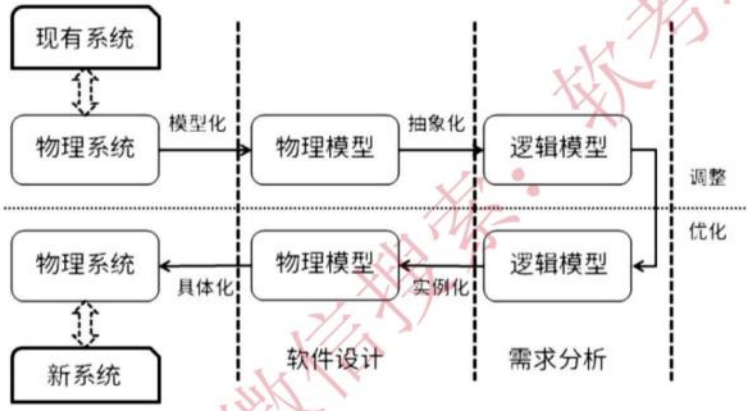
(3) 变更控制

带有风险的做法：无足够用户参与，忽略了用户分类，用户需求的不断增加，模棱两可的需求，不必要的特性，过于精简的 SRS，不准确的估算。



需求变更控制流程的十大步骤：1、明确问题；2、书面申请；3、判断变更需求类别；4、评估变更影响；5、判断变更的紧急级别；6、沟通确认；7、明确解决方案；8、审批管理；9、执行变更；10、版本控制。

2.6 系统建模过程（占）



（1）**结构化建模方法**：结构化建模方法是以过程为中心的技术，可用于分析一个现有的系统以及定义新系统的业务需求。结构化建模方法所绘制的模型称为数据流图（DFD）。对于流程较为稳定的系统可考虑结构化建模方法。

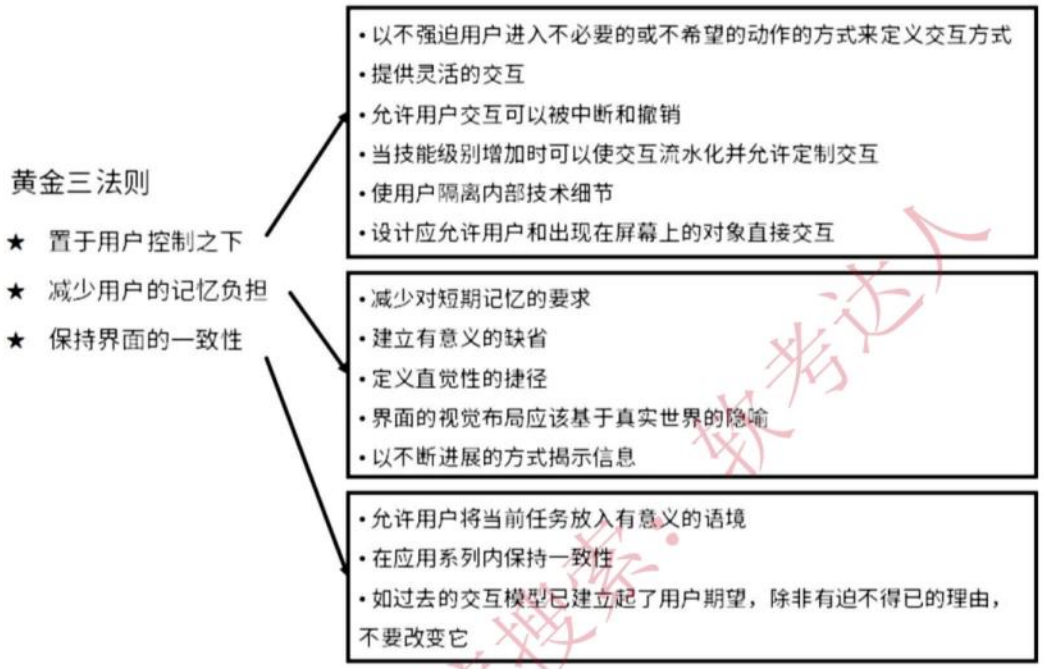
（2）**信息工程建模方法（或数据库建模方法）**：信息工程建模方法是一种以数据为中心，但过程敏感的技术，它强调在分析和研究过程需求之前，首先研究和分析数据需求。信息工程建模方法所创建的模型被称为实体-联系图（ERD）。主要用于数据建模。

（3）**面向对象建模方法**：面向对象建模方法将“数据”和“过程”集成到被称为“对象”的结构中，消除了数据和过程的人为分离现象。面向对象建模方法所创建的模型被称为对象模型。随着面向对象技术的不断发展和应用，形成了面向对象的建模标准，即 UML（统一建模语言）。UML 定义了几种不同类型的模型图，这些模型图以对象的形式共建一个信息系统或应用系统。目前比较常用的建模方法。

2.7 系统设计 （占占）

2.7.1 界面设计

用户界面设计是指用户与系统之间架起一座桥梁，主要内容包括：定义界面形式、定义基本的交互控制形成、定义图形和符号、定义通用的功能键和组合键的含义及其操作内容、定义帮助策略等。



2.7.2 结构化设计

（1）**概要设计【外部设计】**：功能需求分配给软件模块，确定每个模块的功能和调用关系，形成**模块结构图。**

（2）**详细设计【内部设计】**：为每个**具体任务**选择适当的技术手段和**处理方法。**

（3）结构化设计原则：

模块独立性原则（**高内聚、低耦合**）；

保持模块的大小适中；

多扇入，少扇出；

深度和宽度均不宜过高。

（4）模块四要素：

输入和输出：模块的输入来源和输出去向都是同一个调用者，即一个模块从调用者那儿取得输入，进行加工后再把输出返回调用者。

处理功能：指模块把输入转换成输出所做的工作。

内部数据：指仅供该模块本身引用的数据。

程序代码：指用来实现模块功能的程序。

（5）模块独立性的度量

1.聚合：衡量模块内部各元素结合的紧密程度

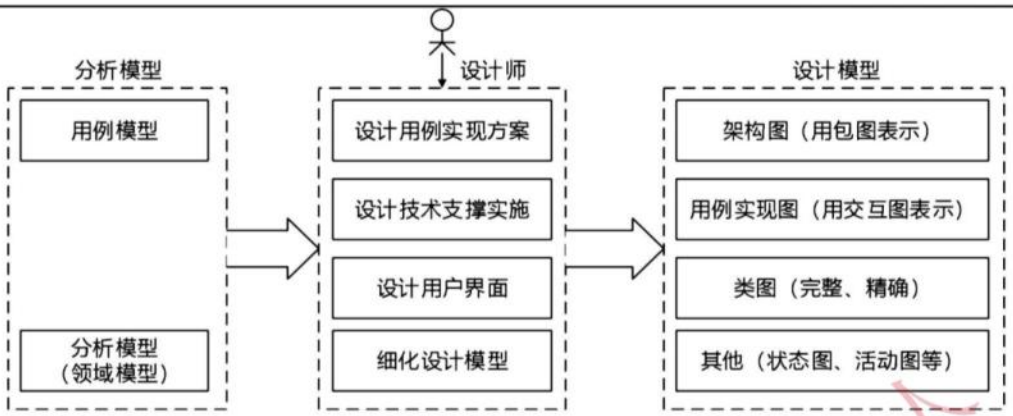
内聚类型	描述
功能内聚	完成一个**单一功能**，各个部分协同工作，缺一不可。
顺序内聚	处理元素相关，而且**必须顺序执行。**
通信内聚	所有处理元素集中在一个数据结构的区域上。
过程内聚	处理元素相关，而且必须按特定的次序执行。
时间内聚 （瞬时内聚）	所包含的任务必须在**同一时间间隔内**执行。
逻辑内聚	完成逻辑上相关的一组任务。
偶然内聚 （巧合内聚）	完成一组没有关系或松散关系的任务。

2.耦合：度量不同模块间互相依赖的程度

耦合类型	描述
非直接耦合	两个模块之间没有直接关系，它们之间的联系完全是通过主模块的控制和调用来实现的。
数据耦合	一组模块借助参数表传递**简单数据。**
标记耦合	一组模块通过参数表传递记录信息（**数据结构**）。
控制耦合	模块之间传递的信息中包含用于**控制模块内部逻辑的信息。**
外部耦合	一组模块都访问同一**全局简单变量**，而且不是通过参数表传递该全局变量的信息。
公共耦合	多个模块都访问同一个公共数据环境。
内容耦合	一个模块直接访问另一个模块的内部数据；一个模块不通过正常入口转到另一个模块的内部；两个模块有一部分程序代码重叠；一个模块有多个入口。

2.7.3 面向对象设计

（1）过程



（2）设计原则

单一职责原则：设计目的单一的类。

开放-封闭原则：对扩展开放，对修改封闭。

李氏（Liskov）替换原则：子类可以替换父类。

依赖倒置原则：要依赖于抽象，而不是具体实现；针对接口编程，不要针对实现编程。

接口隔离原则：使用多个专门的接口比使用单一的总接口要好。

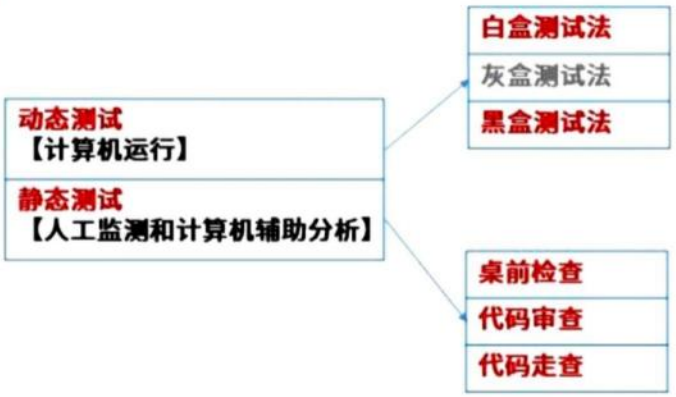
组合重用原则：要尽量使用组合，而不是继承关系达到重用目的。

迪米特（Demeter）原则（最少知识法则）：一个对象应当对其他对象有尽可能少的了解。

2.8 软件测试（占占）

软件测试方法的分类有很多种，以测试过程中程序执行状态为依据可分为静态测试和动态测试；以具体实现算法细节和系统内部结构的相关情况为根据可分黑盒测试、白盒测试和灰盒测试；从程序执行的方式来分类，可分为人工测试和自动化测试。

（1）动态测试与静态测试



静态测试都是做的静态分析，其中静态分析包括：

控制流分析：是否存在没有使用的语句/**无法达到的语句**/调用并不存在的子程序。

数据流分析：引用未定义的变量、对以前未使用的变量再次赋值。

接口分析：模块之间**接口的一致性**、子程序和函数之间的接口一致性、函数形参与实参的数量、顺序、类型的一致性。

表达式分析：括号不配对、数组引用越界、除数为零。

（2）黑盒测试、白盒测试与灰盒测试

白盒测试【结构测试】：关注内部结构与逻辑。

控制流分析、数据流分析、路径分析、程序变异【错误驱动测试】
【路径覆盖】 【逻辑覆盖】 修改条件/判定>条件组合>条件/判定覆盖>条件覆盖>判定覆盖>语句覆盖

黑盒测试【功能测试】：关注输入输出及功能。

等价类划分：不同等价类，揭示不同问题；有效等价类/无效等价类。
边界值分析：1<=x<=10，可取 x 的值为 0、1、10 和 11 作为测试数据。
错误推测：依靠测试人员的**经验和直觉**。
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。
因果图：根据**输入条件与输出结果之间的因果关系**来设计测试用例。

灰盒测试。**灰盒测试介于黑盒与白盒测试之间**。灰盒测试除了重视输出相对于输入的正确性，也看重其内部的程序逻辑。但是，它不可能像白盒测试那样详细和完整。它只是简单地靠一些象征性的现象或标志来判断其内部的运行情况，因此在内部结果出现错误，但输出结果正确的情况下可以采取灰盒测试方法。

（3）自动化测试。自动化测试就是软件测试的自动化，即在预先设定的条件下自动运行被测程序，并分析运行结果。总的来说，这种测试方法就是将以人驱动的测试行为转化为机器执行的一种过程。不适合场景：项目周期短，需求变动频繁。常见自动化测试：单元自动化测试、接口自动化测试、UI 自动化测试。

（4）软件测试阶段

单元测试：依据【**详细设计**】，模块测试，模块功能、性能、接口等。

集成测试：依据【**概要设计**】，模块间的接口。

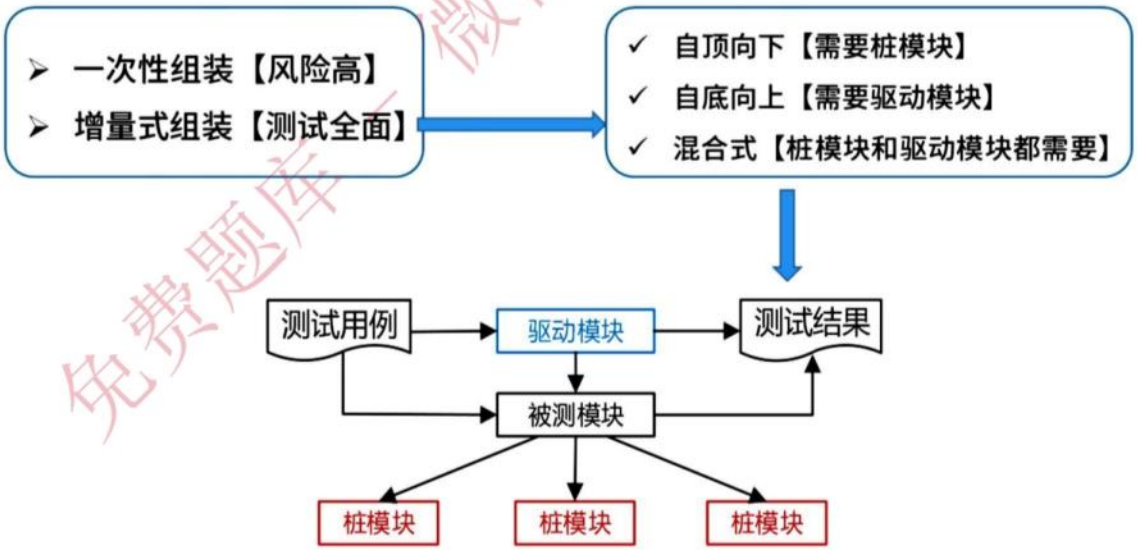
系统测试：依据【**需求文档**】，包括功能测试、性能测试、验收测试、压力测试等。

确认测试：依据【**需求文档**】，验证软件与需求的一致性。内部确认测试、Alpha 测试、Beta 测试、验收测试。

其他测试：

其他测试	描述
AB 测试	**多版本同时使用**，利于收集各版本的用户反馈，评估出最好版本。故也算是一种【**网页优化方法**】。
Web 测试	Web 系统测试与其他系统测试**测试内容基本相同**，只是**测试重点不同**。 Web 代码测试包括：源代码规则分析、链接测试、框架测试、表格测试、图形测试等方面。
链接测试	链接测试可分为 3 个方面： 1、测试所有链接是否按指示的那样**确实链接到了该链接的页面**。 2、测试所链接的**页面是否存在**。 3、保证 Web 应用系统上没有**孤立的页面**。
表单测试	验证服务器是否能**正确保存这些数据**，后台运行的程序能否正确解释和使用这些信息。 测试提交操作的**完整性**。
回归测试	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。

（5）集成测试策略



（6）系统测试

系统测试

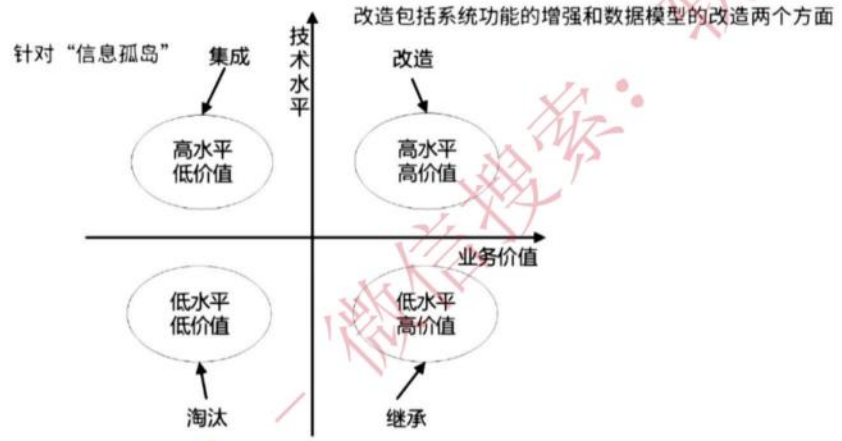
- 功能测试
- 性能测试
- 健壮性测试
- 用户界面测试
- 安全性测试
- 安装与反安装测试

- ✓ 负载测试：各种工作负载下系统的性能
- ✓ 压力测试：系统的瓶颈或不能接受的性能点
- ✓ 强度测试：系统资源特别低的情况下运行
- ✓ 容量测试【并发测试】：同时在线的最大用户数
- ✓ 可靠性测试：MTTF之类的参数

2.9 系统运行与软件维护（占）

2.9.1 系统转换计划

（1）遗留系统演化策略



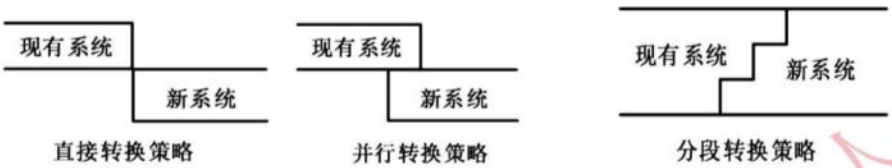
淘汰策略：遗留系统的技术含量较低，且具有较低的业务价值。对遗留系统的完全淘汰是企业资源的根本浪费，我们应该善于“变废为宝”，通过对遗留系统功能的理解和借鉴，可以帮助新系统的设计，降低新系统开发的风险。

继承策略：遗留系统的技术含量较低，已经满足企业运作的功能或性能要求，且具有较高的商业价值，目前企业的业务尚紧密依赖该系统。对这种遗留系统的演化策略为继承。**在开发新系统时，需要完全兼容遗留系统的功能模型和数据模型。**为了保证业务的连续性，新老系统必须并行运行一段时间，再逐渐切换到新系统上运行。

改造策略：遗留系统具有较高的业务价值，基本上能够满足企业业务运作和决策支持的需要。这种系统可能建成的时间还很短，对这种遗留系统的演化策略为改造。改造包括系统功能的增强和数据模型的改造两个方面。系统功能的增强是指在原有系统的基础上增加新的应用要求，对遗留系统本身不做改变；数据模型的改造是指将遗留系统的旧的数据模型向新的数据模型的转化。

集成策略：遗留系统的技术含量较高，但其业务价值较低，可能只能完成某个部门（或子公司）的业务管理。这种系统在各自的局部领域里工作良好，但对于整个企业来说，存在多个这样的系统，不同的系统基于不同的平台、不同的数据模型，形成了一个个**信息孤岛**，对这种遗留系统的演化策略为集成。

（2）新旧系统转换策略



直接转换：直接转换是在原有系统停止运行的某一时刻，新系统立即投入运行，中间没有过渡阶段。采用这种方式时，人力和费用最省，适用于系统不太复杂或现有系统完全不能使用的场合。但是这种方式风险高。

并行转换：并行转换就是新系统和旧系统并行工作一段时间，经过这段时间的试运行后，再用新系统正式替换下现有系统。那么这种方式，它的好处就是风险很小。在转换期间还可以同时比较新旧两套系统的性能，而且能够让操作人员得到全面的培训，所以对于一些比较大的信息系统，或者处理过程比较复杂，数据比较重要的系统。并行转换是一种最常用的转换方式。那么这种转换方式也有缺点，缺点就在于两套系统并行期间。要有两组人员，或者两套处理方式同时并存，在人力和费用上消耗比较大，转换的周期比较长，而且难以控制新旧系统当中数据的变化。所以这就要求要做好转换计划，并且要加强管理。

分段转换：这是直接转换和并行转换的结合，也就是分期分批、逐步转换。一般比较大的系统采用这种方式比较合适，它能够保证软件平稳运行，费用也不太高，就是将大的系统分成多个子系统，成熟一个子系统就切换一个子系统，主要是分期分批。这种新旧转换，震动比较小，用户比较容易接受。但是由于采取的是渐进的方式，会导致新旧系统的转换周期比较长。

（3）数据转换与迁移

2.9.2 系统维护

（1）影响软件可维护性的因素

【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。

【可修改性】是指修改软件的难易程度。

【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。

【可靠性】一个软件的可靠性越高，需要维护的概率就会越低。

【可移植性】是指将软件从一个环境移植到新的环境下正确运行的难易程度。

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。

（2）软件维护类型

正确性维护【修 BUG】：识别和纠正软件错误/缺陷，测试不可能发现所有错误。

适应性维护【应变】：指使应用软件**适应环境变化**【外部环境、数据环境】而进行的修改。

完善性维护【新需求】：为**扩充功能和改善性能**而进行的修改。

预防性维护【针对未来】：为了**适应未来**的软硬件环境的变化，应主动增加预防性的新的功能，以使用系统适应各类变化而不被淘汰。经典实例：**【专用】改【通用】**。

2.10 系统工程建模（占占）

2.10.1 MBSE（Model Based Systems Engineering，基于模型的系统工程）

【MBSE的三大支柱】 【软件工程】

【建模语言】沟通标准化，MBSE的核心。典型代表：SysML，OPM。 ---------> UML

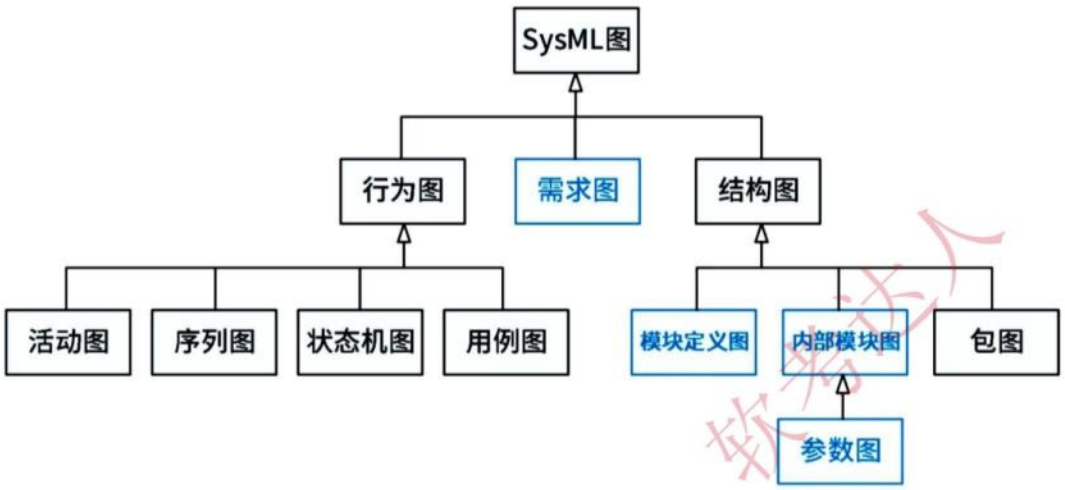
【建模思路】类似于路线图，如：INCONSE面向对象系统工程法。 ---------> 面向对象开发

【建模工具】如：Agilian，Modelio。 ---------> Visio、Rose

MBSE 是建模方法的形式化应用，以使建模方法支持系统需求、分析、设计、验证和确认等活动，这些活动从概念性设计阶段开始，持续贯穿到设计开发以及后来的所有生命周期阶段。

2.10.2 SysML【系统建模语言】

对象管理组织 OMG 在对 UML 2.0 的子集进行重用和扩展的基础上，提出了一种新的系统建模语言 SysML（Systems Modeling Language），作为系统工程的标准建模语言。SysML 的目的是统一系统工程中使用的建模语言。



SysML 中的需求关系：**包含、跟踪、继承需求、改善、满足、验证**和复制。

包含（Include）：需求可以且只能包含其他需求。

跟踪（Trace）：对提供方元素（位于箭头端）的修改可能会导致对客户端元素（位于尾端）修改的需要。

继承（deriveReqt）：一个需求可以继承另一个需求的属性。

改善（refine）：表示一个需求改进了另一个需求的满足程度。

满足（satisfy）：一般是模块满足某种需求。

验证（verify）：表示一个需求验证了另一个需求的正确性。

复制（Copy）：表示一个需求复制了另一个需求的特性。

3 章节问答

（1）演化模型跟螺旋模型，增量模型，原型法开发的关系？

答：首先是原型化，原型化又可以分为两种：抛弃式原型和演化式原型。其中最终演化成一个产品的就演变成为演化模型。螺旋模型是通过演化模型和瀑布模型的结合所产生的，并且螺旋模型强调了风险管理。增量模型是原型模型和瀑布模型所结合而产生的。螺旋模型和增量模型的区别：螺旋模型它是一直旋下去的，旋到最后才是一个产品，而增量模型每一个增量都发布了一个可操作的产品。

（2）面向对象方法中的分析模型、设计模型是什么？

答：面向对象的分析模型主要由顶层架构图、用例与用例图和领域概念模型构成；设计模型则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。

（3）原型可以分为哪些？分别适合什么场景？

答：从原型是否实现功能来分，可以分为水平原型和垂直原型：

水平原型也称为行为原型，用来探索预期系统的一些特定行为，并达到细化需求的目的。水平原型通常只是功能的导航，但未真实实现功能。水平原型主要用在界面上。

垂直原型也称为结构化原型，实现了一部分功能。垂直原型主要用在复杂的算法实现上。

从原型的最终结果来分，可分为抛弃式原型和演化式原型：

抛弃式原型也称为探索式原型，是指达到预期目的后，原型本身被抛弃。抛弃式原型主要用在解决需求不确定性、二义性、不完整性、含糊性等。

演化式原型为开发增量式产品提供基础，逐步将原型演化成最终系统，主要用在必须易于升级和优化的场合，适合于 Web 项目。

（4）构件模型在使用上的优点和缺点有哪些？

答：优点：大大提高了软件开发的效率，而且构件模型允许多个项目同时开发，降低了费用，提高了可维护性。

缺点：由于存在多种构件标准，缺乏通用的构件组装结构标准，如果自行定义标准的话，会引入较大的风险；构件可重用性和软件系统高效性之间不易协调，需要权衡。

（5）负载测试和容量测试是一个概念吗？

答：二者不是同一个概念。

负载测试：负载测试是通过逐步增加系统负载，测试系统性能的变化，并最终确定在满足性能指标的情况下，系统所能承受的最大负载量的测试。

容量测试，主要用于测试系统可同时处理的在线最大用户数量。

第三章 系统可靠性分析与设计

1 考情分析

根据对历年的考试真题进行分析，本章要求考生掌握以下几个方面的知识：

(1) 了解可靠性相关基本概念。

(2) 了解可靠性指标，掌握可靠性计算，了解系统可靠性模型。

(3) 熟悉可靠性设计相关策略方法。

1.1 本章重点

序号	知识领域	知识点详情
1	可靠性相关基本概念（占占）	基本概念
1	可靠性分析（占占占占）	可靠性指标
2		可靠性计算
3		系统可靠性模型
1	软件可靠性设计（占占占占）	可靠性设计

2 考点精讲

2.1 相关基本概念（占占）

可靠性：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。

可用性：可用性是**系统能够正常运行的时间比例。**

软件可靠性 ≠ 硬件可靠性

软硬件对比

复杂性：软件复杂性比硬件高，**大部分失效来自于软件失效。**

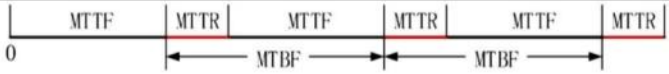
物理退化：硬件失效主要是物理退化所致，软件不存在物理退化。

唯一性：软件是唯一的，每个 COPY 版本都一样，而两个硬件不可能完全一样。

版本更新周期：硬件较慢，软件较快。

2.2 系统可靠性分析（占占占占）

2.2.1 可靠性指标



平均无故障时间 → (MTTF)系统**无故障运行**的平均时间。

平均故障修复时间 → (MTTR) 从**出现故障到修复成功**中间的这段时间。

平均故障间隔时间 → (MTBF) MTBF = MTTR + MTTF

平均检测时间 → (MTTD) **故障发生到被检测出来**的时间【潜伏期】。

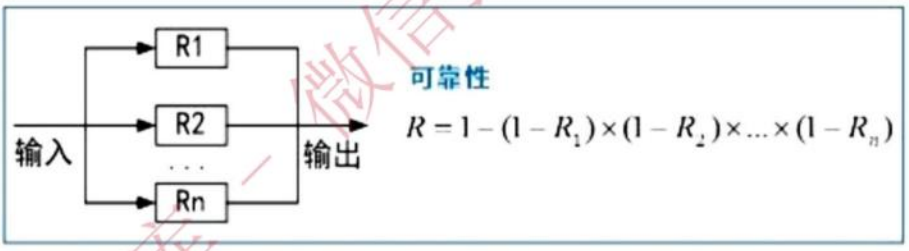
系统可用性 → MTTF/(MTTR+MTTF)×100%

在实际应用中，一般 MTTR 很小，所以**通常认为 MTBF≈MTTF**。

2.2.2 可靠性计算

串联系统

并联系统



N 模混联系统

先将整个系统划分为多个部分串联 R1、R2…等，再计算 R1、R2内部的并联可靠性，带入原公式。

2.2.3 系统可靠性模型

可靠性模型大致可分为如下10类：

种子法模型：预先有意“播种”一些设定的错误“种子”，再看种子发现比例。

失效率类模型：如：Jelinski-Moranda 的 De-eutrophication 模型、Schick-Wolverton 模型。

曲线拟合类模型：用回归分析的方法研究软件复杂性、程序中的缺陷数、失效率、失效间隔时间。

可靠性增长模型：预测软件在检错过程中的可靠性改进，用增长函数来描述软件的改进过程。

程序结构分析模型：根据程序、子程序及其相互间的调用关系，形成一个可靠性分析网络。

输入域分类模型：选取软件输入域中的某些样本“点”运行程序，根据这些样本点在“实际”使用环境中的使用概率的测试运行时的成功/失效率，推断软件的使用可靠性。

执行路径分析方法模型：先计算程序各逻辑路径的执行概率和程序中错误路径的执行概率，再综合出该软件的使用可靠性。

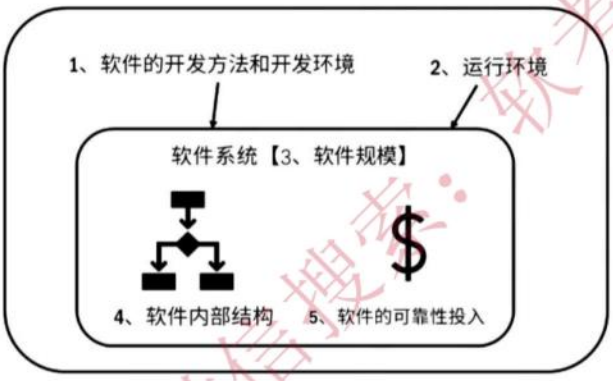
非齐次泊松过程模型：以软件测试过程中单位时间的失效次数为独立泊松随机变量，来预测在今后软件的某使用时间点的累计失效数。

马尔可夫过程模型：如：完全改错的线性死亡模型、不完全改错的线性死亡模型。

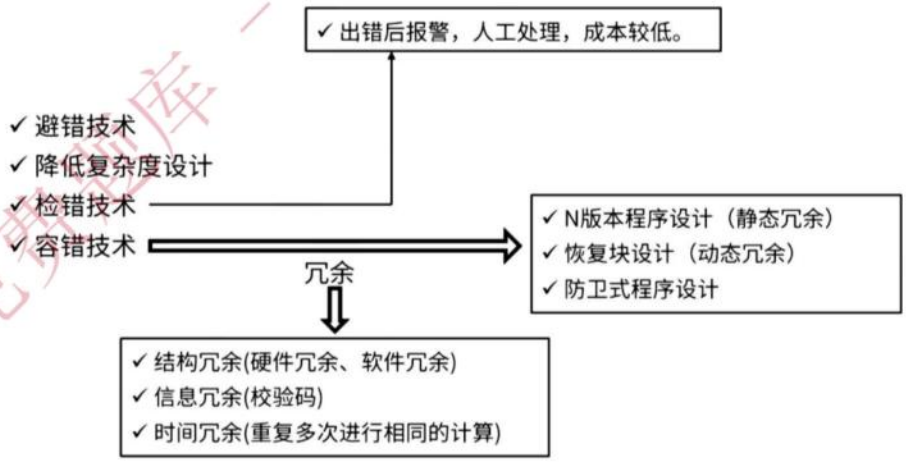
贝叶斯分析模型：利用失效率的试验前分布和当前的测试失效信息，来评估软件的可靠性。

2.3 可靠性设计（占占占占）

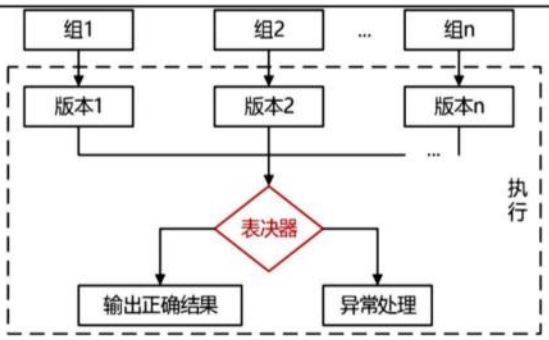
（1）影响可靠性的因素



（2）可靠性设计技术



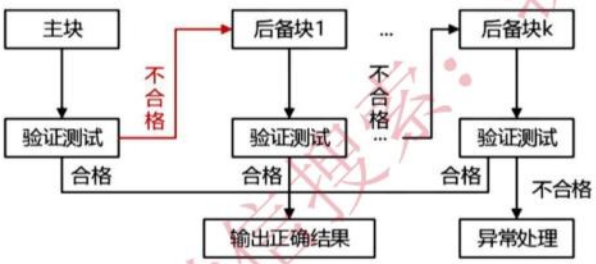
2.3.1 N 版本程序设计



与通常软件开发过程不同的是，N 版本程序设计增加了三个新的阶段：相异成份规范评审、相异性确认、背对背测试

N 版本程序的同步、N 版本程序之间的通信、表决算法（全等表决、非精确表决、Cosmetie 表决）、一致比较问题、数据相异性

2.3.2 恢复块方法



设计时应保证实现主块和后备块之间的独立性，避免相关错误的产生，使主块和备份块之间的共性错误降到最低程度。

必须保证验证测试程序的正确性。

（1）恢复方法分类

前向恢复：使当前的计算继续下去，把系统恢复成连贯的正确状态，弥补当前状态的不连贯情况

后向恢复：系统恢复到前一个正确状态，继续执行

（2）恢复块方法与 N 版本程序设计对比

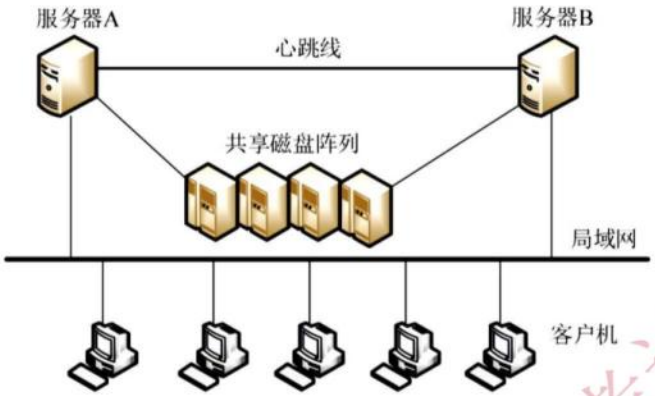
	恢复块方法	N 版本程序设计
硬件运行环境	单机	多机
错误检测方法	验证测试程序	表决
恢复策略	后向恢复	前向恢复
实时性	差	好

2.3.3 防卫式程序设计

对于程序中存在的错误和不一致性，通过在程序中包含错误检查代码和错误恢复代码，使得一旦错误发生，程序能撤销错误状态，恢复到一个已知的正确状态中去

实现策略：错误检测、破坏估计、错误恢复

2.3.4 双机容错



双机热备模式（主系统、备用系统）

双机互备模式（同时提供不同的服务，心不跳则接管）

双机双工模式（同时提供相同的服务，集群的一种）

双机模式是集群的前身

3 章节问答

（1）主动冗余是什么？

答：主动冗余是指，处理结点出故障之后，结点本身或控制器会协调没有出故障的结点提供处理服务。

被动冗余，则是靠请求者自己完成这个工作，请求者如果发现一台服务器不能正常提供服务，转而找另一台服务器。

（2）负载均衡是否算是提高可靠性的策略？提高可用性的策略是否也算是提高可靠性设计的策略？

答：提高可靠性最主要的技术是检错技术、容错设计和降低复杂度设计。

可靠性和可用性这两个质量属性是相互促进的作用。提高可用性的同时，间接在提高系统的可靠性，所以，他们的设计策略在一定程度上是有相互包含的。

负载均衡在某些角度来看算是提高可靠性的策略。比如使用带有负载平衡的多个服务器组件，取代单一的组件，可以通过冗余提高可靠性。

（3）一个好的可靠性模型需要具备什么特性？可靠性模型有哪些？

答：一个好的软件可靠性模型应该具有如下重要特性：基于可靠的假设；简单；计算一些有用的量；给出未来失效行为的好的映射；可广泛应用。

可靠性模型大致可以分为如下10类：种子法模型、失效率类模型、曲线拟合类模型、可靠性增长模型、程序结构分析模型、输入域分类模型、执行路径分析方法模型、非齐次泊松过程模型、马尔可夫过程模型、贝叶斯分析模型。

（4）广义的可靠性测试与狭义的可靠性测试是什么？

答：广义的软件可靠性测试是指为了最终评价软件系统的可靠性而运用建模、统计、试验、分析和评价等一系列手段对软件系统实施的一种测试。

狭义的软件可靠性测试是指为了获取可靠性数据，按预先确定的测试用例，在软件的预期使用环境中，对软件实施的一种测试。狭义的软件可靠性测试也叫“软件可靠性试验”，它是面向缺陷的测试，以用户将要使用的方式来测试软件，每一次测试代表用户将要完成的一组操作，使测试成为最终产品使用的预演。

第四章 计算机系统基础

1 考情分析

根据对历年的考试真题进行分析，本章要求考生掌握以下几个方面的知识：

(1) 了解计算机系统的基本组成。

(2) 了解操作系统的功能、操作系统的分类和特点。

(3) 了解进程的三态模型，了解进程中多线程资源的特点。

(4) 了解 PV 操作的基本概念，掌握前趋图的分析。

(5) 掌握死锁资源数的计算和银行家算法的分析。

(6) 了解页式存储、段式存储、段页式存储的特点，掌握页式存储逻辑地址与物理地址的转化，了解页式存储中页面淘汰的原则。

(7) 了解磁盘的基本结构，掌握磁盘平均存取时间的计算，了解磁盘移臂调度算法对调度序列的影响。

(8) 掌握索引文件、位示图的分析。

(9) 了解常见性能指标的分类和描述，了解性能设计和优化的一些原则，了解性能评估的一些方法。

1.1 本章重点

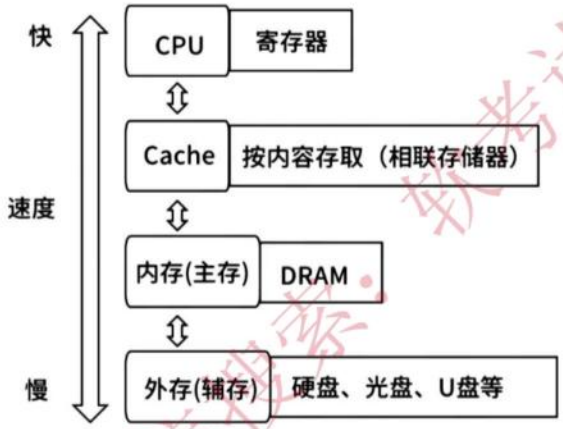
序号	知识领域	知识点详情
1	计算机系统概述	计算机系统组成
1	操作系统（占占占占）	操作系统概述、进程管理、存储管理、磁盘管理
1	文件系统（占占）	文件系统概述、索引文件、位示图
1	系统性能（占）	性能指标、性能设计、性能评估

2 考点精讲

2.1 计算机系统概述

2.1.1 计算机系统组成

2.1.2 存储系统



中央处理器（CPU，Central Processing Unit）

时间局部性：指程序中的某条指令一旦执行，不久以后该指令可能再次执行，典型原因是由于程序中存在着大量的**循环操作**。

空间局部性：指一旦程序访问了某个存储单元，不久以后，其附近的存储单元也将被访问，即程序在一段时间内所访问的地址可能集中在一定的范围内，其典型情况是程序**顺序执行**。

工作集理论：工作集是进程运行时被频繁访问的页面集合。

2.1.3 处理器体系结构

体系结构分类	定义	特点	典型应用
冯·诺依曼结构	冯·诺依曼结构也称普林斯顿结构，是一种将程序指令存储器和数据存储器合并在一起的存储器结构。	指令与数据存储器合并在一起。 指令与数据都通过相同的数据总线传输。	一般用于 PC 处理器，如 I3、I5、I7 处理器。 **注：常规计算机属于冯·诺依曼结构**
哈佛结构	哈佛结构是一种并行体系结构，它的主要特点是将程序和数据存储在不同	指令与数据分开存储，可以并行读取，有较高的数	一般用于嵌入式系统处理器。

	的存储空间中，即程序存储器和数据存储器是两个独立的存储器，每个存储器独立编址、独立访问。	据吞吐率。 有 4 条总线：指令和数据的数据总线与地址总线。	**注：DSP 属于哈佛结构**

2.1.4 总线

总线的基本概念：总线是一组能为多个部件**分时共享**的信息传送线，用来连接多个部件并为之提供信息交换通路。【总线通常是半双工的】

特点：

挂接在总线上的多个部件只能分时向总线发送数据，但可同时从总线接收数据。

通过总线复用方式可以减少总线中信号线的数量，以较少的信号线传输更多的信息。

总线分类：

总线通常分为以下几类。

芯片内总线。用于集成电路芯片内部各部分的连接元件级总线。用于一块电路板内各元器件的连接。

系统总线，又称内总线。用于计算机各组成部分（CPU、内存和接口等）的连接。

外总线，又称通信总线。用于计算机与外设或计算机与计算机之间的连接或通信。

（1）从功能上来对总线进行划分：数据总线、地址总线和控制总线

（2）从数据传输的方式划分为并行总线和串行总线

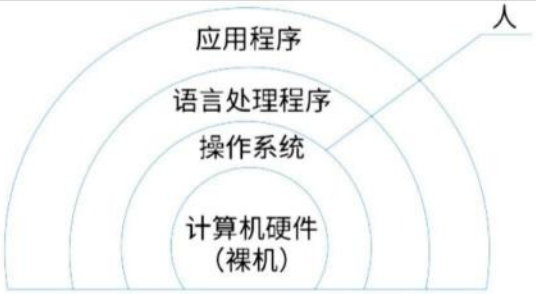
串行总线：数据是一位一位地进行传输的，在传输中每一位数据都占据一个固定的时间长度。常见串行总线：RS232、SPI、I2C、USB、CAN、IEEE 1394等。【**长距离**，传输波特率可调整，正确性依赖于校验码，数据传输方式可以使用多种】

并行总线：将数据字节的各位用多条数据线同时进行传送。常见并行总线 ISA、PCI、VME 等。【**短距离**】

2.2 操作系统概述（占占占占）

2.2.1操作系统概述

（1）软件分层



（2）操作系统作用

管理系统的硬件、软件、数据资源

控制程序运行

人机之间的接口

应用软件与硬件之间的接口

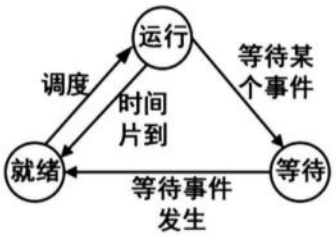
（3）特殊的操作系统

分类	特点
批处理操作系统	单道批：一次一个作业入内存，作业由程序、数据、作业说明书组成 多道批：一次多个作业入内存，特点：多道、宏观上并行微观上串行
分时操作系统	采用时间片轮转的方式为多个用户提供服务，每个用户感觉独占系统 特点：多路性、独立性、交互性和及时性
实时操作系统	实时控制系统和实时信息系统 交互能力要求不高，可靠性要求高（**规定时间内响应并处理**）
网络操作系统	方便有效共享网络资源，提供服务软件和有关协议的集合 主要的网络操作系统有：Unix、Linux 和 Windows Server 系统
分布式操作系统	任意两台计算机可以通过通信交换信息 是网络操作系统的更高级形式，具有透明性、可靠性和高性能等特性
微机操作系统	Windows：Microsoft 开发的图形用户界面、多任务、多线程操作系统 Linux：免费使用和自由传播的类 Unix 操作系统，多用户、多任务、多线程和多 CPU 的操作系统
嵌入式操作系统	运行在智能芯片环境中 特点：微型化、**可定制（针对硬件变化配置）**、实时性、可靠性、**易移植性（HAL 和 BSP 支持）**

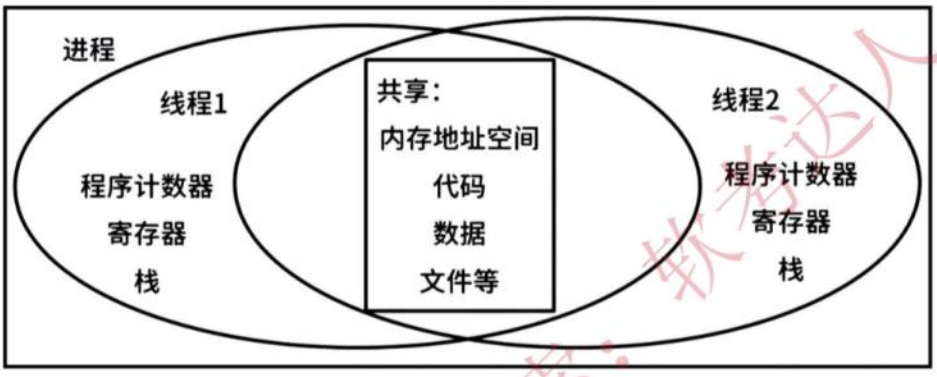
2.2.2 进程管理

2.2.2.1 进程的状态

（1）操作系统三态模型如下图所示：



（2）进程与线程的概念：



（3）进程的同步与互斥

相关概念：互斥、同步、临界资源、临界区、信号量。

互斥：如千军万马过独木桥，同类资源的竞争关系。间接制约关系

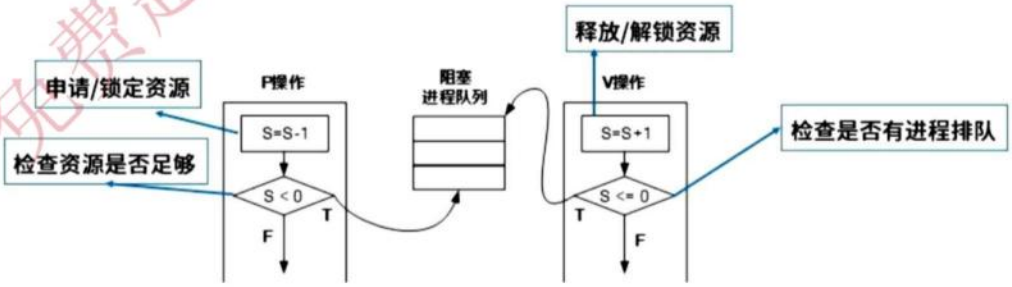
同步：速度有差异，在一定情况停下等待，进程间的协作关系。直接制约关系

临界资源：诸进程间需要互斥方式对其进行共享的资源，如打印机、磁带机等

临界区：每个进程中访问临界资源的那段代码称为临界区

信号量：是一种特殊的变量。信号量可以表示资源数量；信号量为负数时还可以表示排队进程数

PV 操作对应的过程（如下图所示）：



前趋图与 PV 操作结合，根据前趋图箭线标注信号量，再根据进程图填空。

（4）进程调度算法

时间片轮转

先来先服务

短作业优先

高响应比优先

优先级调度

抢占式&非抢占式

2.2.2.2 前趋图

（1）前趋活动和后继活动：在前趋图中，前趋活动完成后通知所有后继活动；后继活动开始之前要检查是否前趋活动已经全部完成。

（2）前趋图的表示：

- **1个箭头表示1个前趋关系**
- **A 有箭头指向 D，则记录为（A，D）**
- **没有前趋进程的结点是起始进程，没有后继进程的结点是终结进程。**

（3）技巧：

- **并发图中某活动有后继就有 V 操作释放资源并通知后继活动，有前趋就有 P 操作检查资源是否足够。**
- **实现并发的信号量初值一般为0。有几个箭头就有几个信号量。**

2.2.2.3 死锁及银行家算法

了解死锁的条件和预防概念：进程管理是操作系统的核心，但如果设计不当，就会出现死锁的问题。如果一个进程在等待一件不可能发生的事，则进程就死锁了。而如果一个或多个进程产生死锁，就会造成系统死锁。

银行家算法：分配资源的限制

当一个进程对资源的最大需求量不超过系统中的资源数时可以接纳该进程。

进程可以分期请求资源，但请求的总数不能超过最大需求量。

当系统现有的资源不能满足进程尚需资源数时，对进程的请求可以推迟分配，但总能使进程在有限的时间里得到资源。

2.2.3 存储管理

（1）页式存储：将程序与内存均划分为同样大小的块，以页为单位将程序调入内存。

优点：利用率高，碎片小，分配及管理简单

缺点：增加了系统开销；可能产生抖动现象

（2）段式存储：按用户作业中的自然段来划分逻辑空间，然后调入内存，段的长度可以不一样。

优点：多道程序共享内存，各段程序修改互不影响

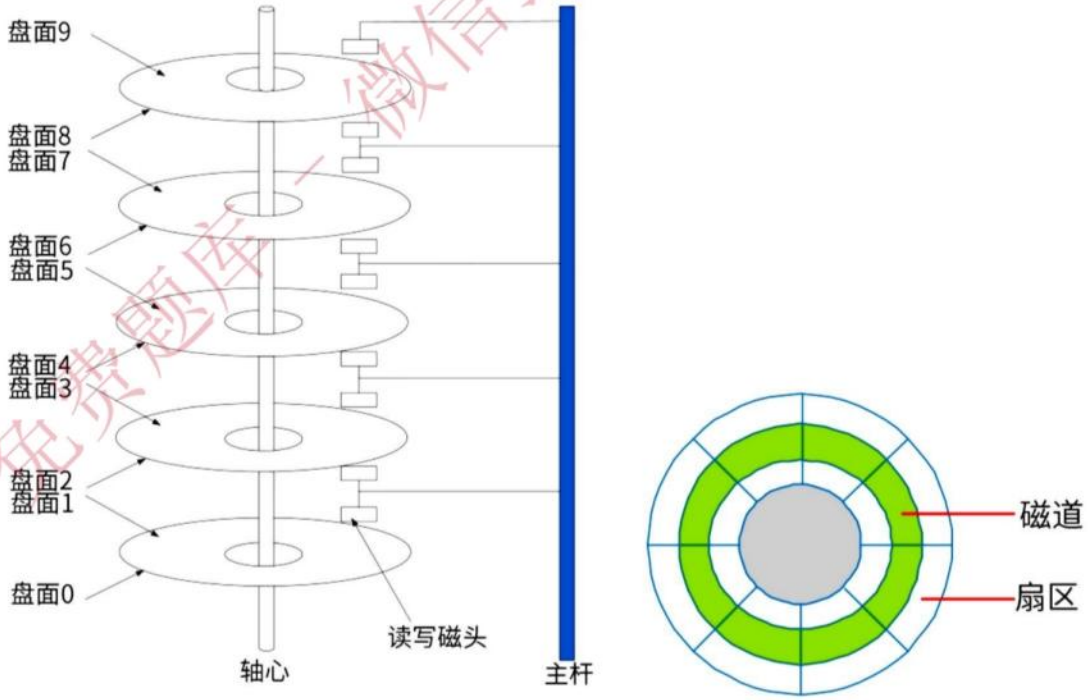
缺点：内存利用率低，内存碎片浪费大

（3）段页式存储：段式与页式的综合体。先分段，再分页。1个程序有若干个段，每个段中可以有若干页，每个页的大小相同，但每个段的大小不同。

优点：空间浪费小、存储共享容易、存储保护容易、能动态连接

缺点：由于管理软件的增加，复杂性和开销也随之增加，需要的硬件以及占用的内容也有所增加，使得执行速度大大下降

2.2.4 磁盘管理



（1）存取时间=寻道时间+等待时间，寻道时间是指磁头移动到磁道所需的时间；等待时间为等待读写的扇区转到磁头下方所用的时间。

（2）读取磁盘数据的时间应包括以下三个部分：

找磁道的时间。

找块（扇区）的时间，即旋转延迟时间。

传输时间。

（3）磁盘移臂调度算法：

先来先服务 FCFS（谁先申请先服务谁）；

最短寻道时间优先 SSTF（申请时判断与磁头当前位置的距离，谁短先服务谁）；

扫描算法 SCAN（电梯算法，双向扫描）；

循环扫描 CSCAN（单向扫描）。

2.3 文件系统（占占）

2.3.1 索引文件

（1）索引结点对应的索引方式一般题干会给出，没有给出的默认按照如图所示方式理解，下面的文件大小依图给出计算过程。

（2）根据物理块大小（假设1KB）和地址项长度（假设4B），可以计算存放间接索引的物理块可以存放的地址项个数：物理块大小/地址项长度，向下取整（1KB/4B=256，注意单位和进制转换）。

（3）直接索引（即索引结点直接指向实际存储文件的物理块），能够表示的逻辑页号范围是0~9，能够表示的文件大小是10*1KB。

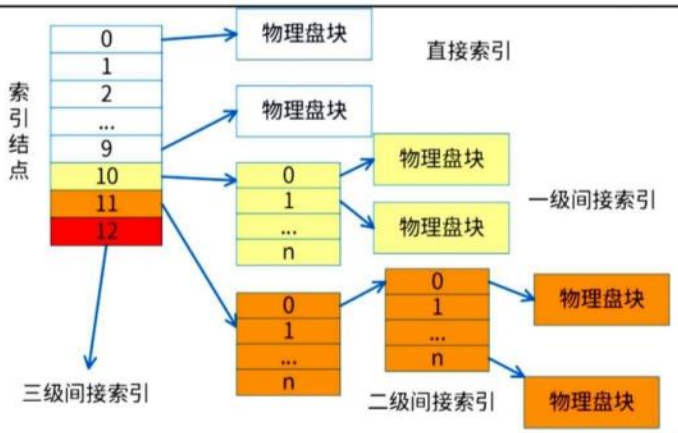
（对于直接索引方式，直接通过索引结点访问1次对应磁盘即可找到数据）

（4）一级间接索引（即索引结点指向的物理块存放的是地址项，对应地址项个数256个，可以指向256个实际存储文件的物理块），能够表示的逻辑页号范围是10~265，能够表示的文件大小是256*1KB。

（对于一级间接索引，需要先访问1次索引磁盘，再通过对应索引访问第2次数据磁盘，从而找到对应数据位置）

（5）二级间接索引（即索引结点指向的物理块存放的是间接索引的地址项，共256个，可以指向256个存放地址项的物理块，每个物理块指向实际存储文件的地址项有256个，最终指向的物理块共有256*256个），能够表示的逻辑页号范围是266~65801，能够表示的文件大小是65536KB。

（对于二级间接索引，需要先访问1次二级地址索引表所在磁盘，再通过对应索引访问第2次一级地址索引表，然后第3次访问数据磁盘，从而找到对应数据位置）



2.3.2 位示图

（1）对于位示图，每一个 bit 位可以表示一个磁盘的占用情况，“0”表示空闲，“1”表示占用。

（2）对于字的长度与具体机器字长有关，由题目指定，假设机器字长16位，则每个字可以表示16个磁盘块的占用情况；

（3）指定序号为 n 或第 n+1个磁盘，占用情况需要用 m=（n+1）/16（向上取整）个字表示，字的序号为 m-1。注意其中磁盘序号、字的序号、对应位号都是从0开始，计算过程中会有加1或减1处理。

【一般根据选项选择最恰当的选项】

2.4 系统性能（占）

2.4.1 性能指标

分类		指标
硬件	计算机	**时钟频率（主频）**；运算速度与精度；内存的存储容量；存储器的存取周期；**数据处理速率 PDR**；**吞吐率**；各种响应时间；各种利用率；RASIS 特性；平均故障响应时间；兼容性；可扩充性；性能价格比。
	路由器	设备吞吐量、**端口吞吐量**、全双工线速转发能力、**丢包率、时延、时延抖动**、VPN 支持能力、端口硬件队列数、基于 Web 的管理、网管类型等。
	交换机	交换机类型、配置、支持的网络类型、最大 ATM 端口数、**支持协议和标准等。**
	网络	设备级性能指标；网络级性能指标；应用级性能指标；用户级性能指标；**吞吐量。**
软件	操作系统	系统的可靠性、**系统的吞吐率（量）**、**系统响应时间**、系统资源利用率、可移植性。
	数据库管理系统	衡量数据库管理系统的主要性能指标包括数据库本身和管理系统两部分，有：数据库的大小、数据库中表的数量、单个表的大小、表中允许的记录（行）数量、单个记录（行）的大小、表上所允许的索引数量、数据库所允许的索引数量、**最大并发事务处理能力、负载均衡能力、最大连接数等。**

	WEB 服务器	**最大并发连接数、响应延迟、吞吐量。**

时钟频率（主频）：主频又称为时钟频率，时钟周期是时钟频率的倒数。如主频为1GHz，则说明1秒钟有1G 个时钟周期，每个时钟周期为1000*1000*1000/1G=1ns。（主频 = 外频 * 倍频）

运算速度：运算速度是计算机工作能力和生产效率的主要表征，取决于给定时间内 CPU 所能处理的数据量和 CPU 的主频。一般用 MIPS 和 MFLOPS 表示。

MIPS（百万条指令/秒）：每秒处理的百万级的机器语言指令数，主要用于衡量标量机性能。**MIPS = 指令条数/（执行时间×10^6）= 主频/CPI = 主频×IPC**

MFLOPS（百万次浮点运算/秒）：每秒百万个浮点操作，不能反映整体情况，只能反映浮点运算情况，主要用于衡量向量机性能。**MFLOPS = 浮点操作次数/（执行时间×10^6）**

平均每条指令的平均时钟周期个数（CPI，clock per instruction）

每（时钟）周期运行指令条数（IPC，instruction per clock）

响应时间（RT，Response Time）

2.4.2 系统性能设计

对于**数据库系统**，性能调整主要包括 CPU/内存使用状况、优化数据库设计、优化数据库管理以及进程/线程状态、硬盘剩余空间、日志文件大小等；

对于**应用系统**，性能调整主要包括应用系统的可用性、响应时间、并发用户数以及特定应用的系统资源占用等。

对系统中某组件采用某种更快的执行方式，所获得的系统性能的改变程度，取决于该组件被使用的频率，或所占总执行时间的比例。加速比计算公式如下：

$$R = \frac{T_p}{T_i} = \frac{1}{(1 - F_e) + F_e / S_e}$$

其中，T_p 表示不使用改进组件时完成整个任务的时间，T_i 表示使用改进组件时完成整个任务的时间。加速比主要取决于两个因素：

（1）在原有的系统上，能被改进的部分在总执行时间中所占的比例。这个值称为改进比例，记为 F_e，它总是小于1。

（2）通过改进的执行方式所取得的性能提高，即如果整个系统使用了改进的执行方式，那么，系统的执行速度会有多少提高，这个值等于在原来的条件下系统的执行时间与使用改进组件后系统的执行时间之比，记为 S_e，它总大于1。

2.4.3 性能评估

方法	描述	特点

时钟频率法	以时钟频率高低衡量速度	仅考虑 CPU
指令执行速度法	表示机器运算速度的单位是 MIPS	仅考虑 CPU
等效指令速度法（吉普森混合法）	通过各类指令在程序中所占的比例（W_i）进行计算得到的。	仅考虑 CPU，**综合考虑指令比例不同的问题。**
数据处理速率法（PDR）	PDR 值的方法来衡量机器性能，PDR 值越大，机器性能越好。	PDR = L/R **仅考虑 CPU+存储**
综合理论性能法（CTP）	CTP 用 MTOPS 表示。CTP 的估算方法是，首先算出处理部件每个计算单元的有效计算率，再按不同字长加以调整，得出该计算单元的理论性能，所有组成该处理部件的计算单元的理论性能之和即为 CTP。	**仅考虑 CPU+存储**
基准程序法	把应用程序中用**得最多、最频繁的那部分核心程序**作为评估计算机系统性能的**标准程序，称为基准测试程序（benchmark）。**	**综合考虑多部分**，基准程序法是目前一致承认的测试系统性能的较好方法。

【测试精确度排名】 真实的程序 > 核心程序 > 小型基准程序 > 合成基准程序

在 Web 服务器的测试中，反映其性能的指标主要有：**最大并发连接数、响应延迟和吞吐量等。**

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。

系统监视：进行系统监视通常有3种方式：一是通过**系统本身提供的命令**，如 UNIX/Linux 系统中的 W、ps、last，Windows 中的 netstat 等；二是通过**系统记录文件**查阅系统在特定时间内的运行状态；三是集成命令、文件记录和可视化技术的监控工具，如 **Windows 的 Perfmon 应用程序。**

3 章节问答

（1）移臂调度过程中，同一柱面有多个扇区怎么办？

答：一般而言，涉及移臂调度算法时，只需要考虑移臂过程，查看柱面顺序即可。旋转等待时间跟多磁头（盘面）是否并发读写有关系，从目前磁盘技术来看，是支持多个盘面同时读写的。所以只要是同一柱面，扇区顺序一般没有要求。如果考试时遇到如果仅凭柱面判断不了顺序的情况下，可以参考扇区号（从小到大）。

（2）为什么说高速缓存对程序员来说是透明的？什么是透明？

答：这里的透明指的是不需要了解其中的机制甚至是看不到这个层次，也就是不直接操作。高速缓存是由硬件自动完成的，所以对程序员来说是透明的。

（3）PV 操作中信号量表示什么含义？

答：PV 操作中，信号量一般可以表示资源的数量。当信号量小于0，表示资源数不足，此时信号量可以表示等待资源的进程数量。

（4）位示图计算过程中，编号是从0开始还是从1开始?

答：根据近几年的其他科目真题来看，目前考查时会给出编号形式是从0还是从1开始。如果没有给出，则默认编号从0开始计算，个数从1开始计算。

（5）死锁的计算公式是什么?

答：假设有 m 个共享资源，n 个进程，每个进程所需的最大资源数为 w，那么仅是 m>n*（w-1）时，不会死锁。m<=n*（w-1）时，可能死锁。

（6）死锁产生的4个必要条件中，不能作为预防死锁措施的是哪一个?

答：死锁产生的4个必要条件：互斥、请求和保持、不可抢占、循环等待。

产生死锁需要四个条件，那么，只要这四个条件中至少有一个条件得不到满足，就不可能发生死锁了。由于互斥条件是非共享资源所必须的，不仅不能改变，还应加以保证，所以，主要是破坏产生死锁的其他三个条件。

第五章 计算机网络

1 考情分析

根据对历年的考试真题进行分析，本章要求考生掌握以下几个方面的知识：

(1) 掌握通架信系统构。

(2) 掌握 TCP/IP 协议簇相关知识。

(3) 了解网络规划与设计的阶段及各阶段任务，了解逻辑网络设计、物理网络设计阶段的产物。

(4) 了解组网技术相关知识。

1.1 本章重点

序号	知识领域	知识点详情
1	通信系统架构（占）	通信系统架构
1	TCP/IP 协议簇（占占占）	常规 TCP/IP 协议
2		DNS
3		DHCP
1	网络规划与设计（占）	网络冗余设计
1	组网技术（占）	组网技术

2 考点精讲

2.1 通信系统架构（占）

2.1.1 通信系统网络架构

通信网络主要包括局域网、广域网、移动通信网等网络形式。不同的网络会采用不同的技术进行网络构建。

局域网通常由计算机、交换机、路由器等设备组成。局域网的几种典型架构风格：单核心架构，双核心架构，环型架构，层次局域网架构。

广域网属于多级网络，通常由骨干网、分布网、接入网组成。在网络规模较小时，可仅由骨干网和接入网组成。广域网的几种典型架构风格：单核心广域网，双核心广域网，环型广域网，半冗余广域网，对等子域广域网，层次子域广域网

移动通信网为移动互联网提供了强有力的支持，尤其是5G 网络为个人用户、垂直行业等提供了多样化的服务。5GS（5G System）在为移动终端用户 UE（User Equipment）提供服务时通常需要 DN（Data Network）网络，如 Internet、IMS（IP Media Subsystem）、专用网络等互连来为 UE 提供所需的业务。从 UE 通过5GS 接入 DN 的方式来说，存在两种模式，即透明模式和非透明模式。

2.1.2 网络分类

局域网（LAN）

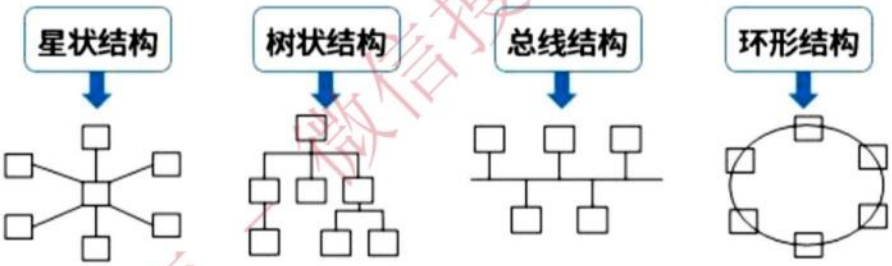
局域网是指在有限地理范围内将若干计算机通过传输介质互联成的计算机组（即通信网络），通过网络软件实现计算机之间的文件管理、应用软件共享、打印机共享、工作组内的日程安排、电子邮件和传真通信服务等功能。局域网是**封闭型**的。

城域网（MAN）

广域网（WAN）

因特网（Internet）

局域网专用性非常强，具有比较稳定和规范的拓扑结构。常见的局域网拓扑结构有**星状结构、树状结构、总线结构和环形结构。**



	特点	缺点
星状结构	网络中的每个结点设备都以中心结点为中心，通过连接线与中心结点相连，如果一个结点设备需要传输数据，它首先必须通过中心结点；中心结点是控制中心，**任意两个结点间的通信最多只需两步；传输速度快、网络结构简单、建网容易、便于控制和管理**	**可靠性低**，网络**共享能力差**，并且一旦中心结点出现故障则导致全网瘫痪。
树状结构	被称为分级的集中式网络；网络成本低，结构简单；在网络中，**任意两个结点之间不产生回路，每个链路都支持双向传输**，结点**扩充方便、灵活，方便寻查链路路径**	除叶结点及其相连的链路外，任何一个工作站或链路产生故障都会影响**整个网络系统的正常运行**
总线结	网络是将各个结点设备和**一根总线相连**；网络中所有	作为数据通信必经的总线的**负载能力**

构	的结点设备都是**通过总线进行信息传输**的	**是有限度的**，这是由通信媒体本身的物理性能决定的；它的**故障将影响总线上每个结点的通信**
环状结构	网络中各结点**通过一条首尾相连的通信链路连接起来，形成一个闭合环形结构网**；网络中各结点设备的地位相同，信息按照**固定方向单向流动，两个节点之间仅有一条通路**，系统中无信道选择问题	网络**不便于扩充**，系统响应延时长，且信息传输**效率相对较低，任一结点的故障将导致物理瘫痪**
网状结构	任何结点彼此之间均存在一条通信链路，任何结点故障不会影响其他结点之间的通信（可靠性高）	网络布线较为**繁琐**，且建设成本高，控制方法复杂

2.1.3 5G 技术

5G 网络的主要特征

（1）**服务化架构**：5G 核心网中引入了 SBA 服务化架构，实现网络功能的灵活定制和按需组合。

（2）**网络切片**：通过网络切片技术在单个独立的物理网络上切分出多个逻辑网络，从而避免了为每一个服务建设一个专用的物理网络，极大地降低了建网和运维成本。

2.1.4 软件定义网络架构

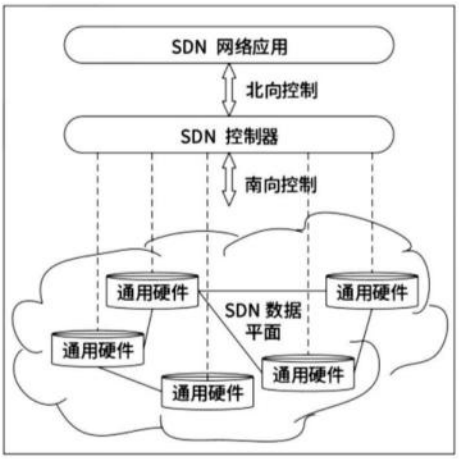
软件定义网络（Software Defined Network，SDN）：一种新型网络创新架构，其核心是将网络设备的控制面与数据面分离开来，从而实现了网络流量的灵活控制。

SDN 架构如下图所示，由下至上分为数据平面、控制平面和应用平面。

应用平面：在网络上运行的应用，用户无需关心底层细节就可编程、部署应用。

控制平面：包含了 SDN 控制器，它掌握着全局网络信息，是网络的“大脑”。

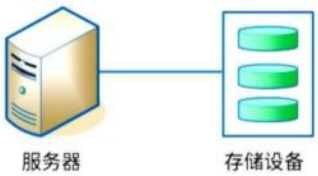
数据平面：交换机、路由器等物理硬件，单纯负责数据的转发。



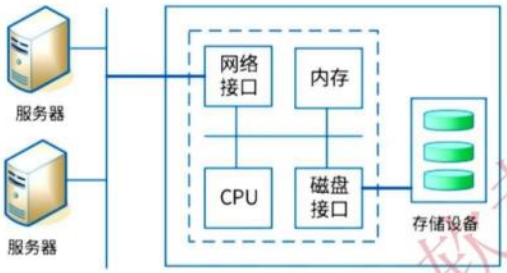
2.1.5 存储网络架构

一般来说，计算机访问磁盘存储有3种方式：

直连式存储（Direct Attached Storage,DAS）：计算机通过 I/O 端口直接访问存储设备的方式。



网络连接的存储（Network Attached Storage,NAS）：计算机通过分布式文件系统访问存储设备的方式。

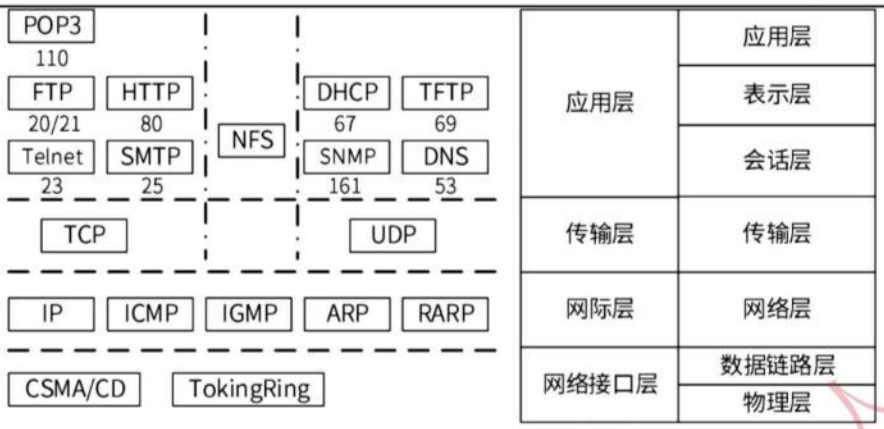


存储区域网络（Storage Area Network,SAN）：计算机通过构建的独立存储网络访问存储设备的方式。

其中 SAN 和 NAS 都可以用于集中管理存储，并供多主机（服务器）共享存储。但是，NAS 通常是基于以太网，而 SAN 可使用以太网和光纤通道。此外，NAS 注重易用性、易管理性、可扩展性和更低的总拥有成本（TCO），而 SAN 则注重高性能和低延迟。

2.2 TCP/IP 协议簇（占占占）

2.2.1 常见协议及功能



网际层是整个 TCP/IP 体系结构的关键部分，其功能是使主机可以把分组发往任何网络并使分组独立地传向目标。

POP3：110端口，邮件收取

SMTP：25端口，邮件发送

FTP：20数据端口/21控制端口，文件传输协议

HTTP：80端口，超文本传输协议，网页传输

DHCP：67端口，IP 地址自动分配

SNMP：161端口，简单网络管理协议

DNS：53端口，域名解析协议，记录域名与 IP 的映射关系

TCP：可靠的传输层协议，**TCP 协议可以依据端口号将报文交付给上层的某一进程，可以对应用层进程进行寻址。**

UDP：不可靠的传输层协议

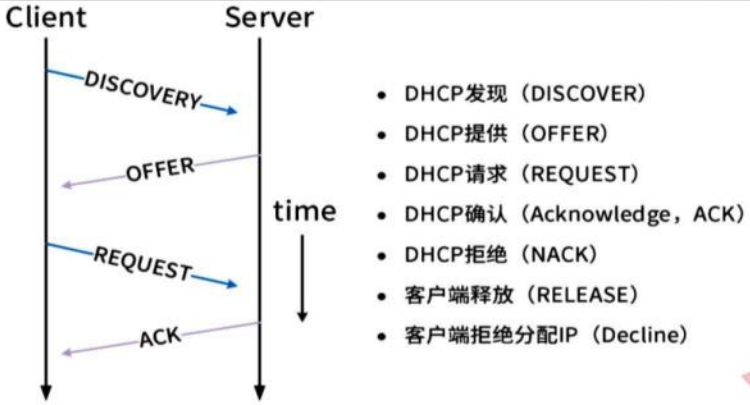
ICMP：因特网控制协议，PING 命令来自该协议

IGMP：组播协议

ARP：地址解析协议，IP 地址转换为 MAC 地址

RARP：反向地址解析协议，MAC 地址转 IP 地址

2.2.2 DHCP



动态主机配置协议（DHCP，Dynamic Host Configuration Protocol）

（1）客户机/服务器模型

（2）租约默认为8天

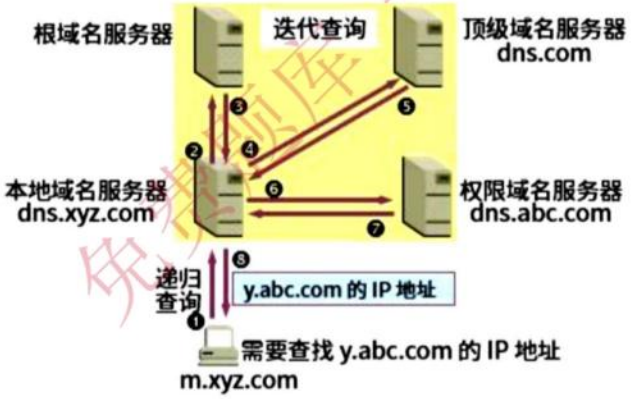
（3）当租约过半时，客户机需要向 DHCP 服务器申请续租；

（4）当租约超过87.5%时，如果仍然没有和当初提供 IP 的 DHCP 服务器联系上，则开始联系其他的 DHCP 服务器。

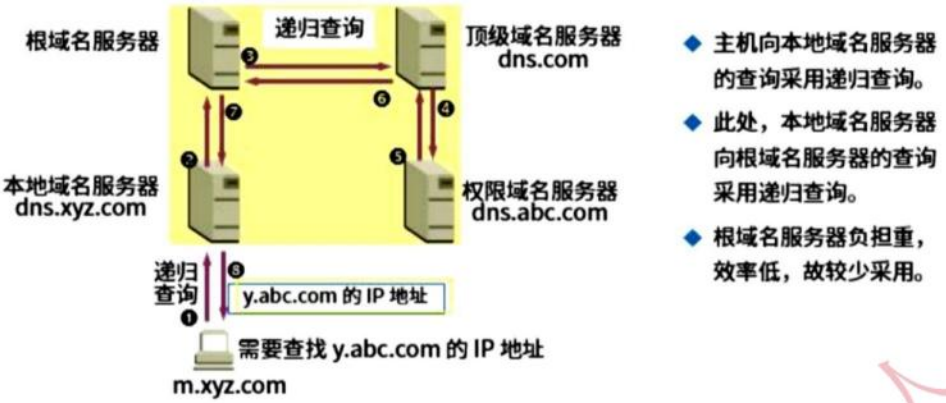
（5）分配方式：固定分配【管理员分配】、动态分配【有效期限的 IP 地址】和自动分配。【无限长的 IP 地址】

（6）无效地址：169.254.X.X（Windows）和0.0.0.0（Linux）

2.2.3 DNS 用法



- 主机向本地域名服务器的查询采用递归查询。
- 本地域名服务器向根域名服务器的查询通常采用迭代查询。



递归查询：服务器必须回答目标 IP 与域名的映射关系。

迭代查询：服务器收到一次迭代查询回复一次结果，这个结果不一定是目标 IP 与域名的映射关系，也可以是其它 DNS 服务器的地址。

在 Linux 系统中，DNS 配置文件 resolv.conf 的关键字主要有四个，分别是：

nameserver：定义 DNS 服务器的 IP 地址

domain：定义本地域名

search：定义域名的搜索列表

sortlist：对返回的域名进行排序

2.3 网络规划与设计（占）

2.3.1 网络构建关键技术

（1）SDN 技术

控制平面技术：控制器是控制平面核心部件，也是整个 SDN 体系架构的逻辑中心。随着 SDN 网络规模的扩大，单一控制器结构的 SDN 网络处理能力遇到了性能瓶颈，因此需要对控制器进行扩展。通 常存在两种控制器扩展方式：一种是对网络中单一控制器本身进行扩展，另一种是采用多控制器方式。

数据平面技术：SDN 转发设备（如交换机等）的数据转发形态可分为硬件和软件两种。硬件处理方式相比软件处理方式具有更快的速度，但灵活性有所降低。而软件的处理速度低于硬件，但软件方式可以提升转发规则处理的灵活性。

转发规则一致性更新技术：在 SDN 网络中不同转发设备转发规则更新可能会出现不一致现象。针对这种问题一般采用“两段提交”的方式来更新规则。

（2）网络高可用设计

网络部件（设备和线路）：网络部件是组成网络的基本要素，典型代表有各种交换机、路由器等网络设备。网络部件的高可用性是网络高可用性的关键。通常网络部件包括硬件结构和软件系统。因此硬件高可用性和软件系统高可用性，就直接影响着网络部件的高可用性。

网络连接模型：除了网络部件本身的高可用性外，网络物理拓扑连接形式也影响网络的可用性程度。

网络协议及配置：高可用性离不开运行于网络中的路由、链路检测等协议。

（3）IPv4与 IPv6融合组网技术

双协议栈：IPv4/IPv6双协议栈机制就是使 IPv6 网络节点具有一个 IPv6协议栈和一个 IPv4协议栈，同时支持 IPv4和 IPv6协议的处理。

隧道技术：在 IPv4和 IPv6融合组网时通常采用下述隧道技术：ISATAP 隧道、6to4 隧道、4over6隧道、6over4隧道。

网络地址翻译技术：网络地址翻译技术将 IPv4地址和 IPv6 地址分别看作内部地址和外部地址，或者相反，以实现地址转换。

2.3.2 网络构建和设计

（1）网络需求获取

业务需求、用户需求、应用需求、计算机平台需求、网络需求。

（2）网络技术遴选及设计

局域网技术遴选、广域网技术遴选、地址规划模型、路由协议选择、层次化网络模型设计、网络高可用设计方法。

（3）网络安全

防火墙部署、VPN 技术、访问控制技术、网络安全隔离、网络安全协议、网络安全审计。

（4）绿色网络

标准化、集成化、虚拟化、智能化、权限性、可靠性。

（5）高可用网络构建

网络可靠性通常是由组成网络的各功能部件稳定提供连续性服务保证的。只要单独提升每个部件的稳定性即可提升整个网络的可靠性；而网络可用性，通常需要构成网络的各部件相互协同，冗余备份等来提供的。这需要通过复杂的网络连接来保证。【设备冗余、链路冗余：矩形连接方式与三角形连接方式】

2.3.3 网络冗余设计

在网络冗余设计中，对于通信线路常见的设计目标主要有两个：一个是备用路径，另一个是负载分担。

备用路径，提高可用性，由路由器、交换机等设备之间的独立备用链路构成，**一般情况下备用路径仅仅在主路径失效时投入使用。**设计时主要考虑：

（1）备用路径的带宽；

（2）切换时间；

（3）非对称；

（4）自动切换；

（5）测试。

负载分担，是对备用路径方式的扩充，通过并行链路提供流量分担（冗余的形式）来提高性能，主要的实现方法是利用两个或多个网络接口和路径来同时传递流量，设计时注意考虑：

（1）网络中存在备用路径、备用链路时，可以考虑加入负载分担设计；

（2）对于主路径、备用路径都相同的情况，可以实施负载分担的特例—负载均衡；

（3）对于主路径、备用路径不相同的情况，可以采用策略路由机制，让一部分应用的流量分摊到备用路径上。

2.3.4 层次化网络设计

核心层：主要是高速数据交换，实现高速数据传输、出口路由，常用冗余机制。

汇聚层 ：网络访问策略控制、数据包处理和过滤、策略路由、广播域定义 、寻址。

接入层：主要是针对用户端，实现用户接入、计费管理、MAC 地址认证、MAC 地址过滤、收集用户信息。

2.4 组网技术（占）

OSI/RM 七层模型

层次	名称	主要功能	主要设备及协议
7	应用层	实现具体的应用功能	POP3、FTP、HTTP、Telnet、SMTP、DHCP、TFTP、SNMP、DNS
6	表示层	数据的格式与表达、加密、压缩	
5	会话层	建立、管理和终止会话	
4	传输层	端到端的连接	TCP、UDP
3	网络层	分组传输和路由选择	三层交换机、路由器 ARP、RARP、IP、ICMP、IGMP
2	数据链路	传送以帧为单位的信	网桥、交换机（多端口网桥）、

Internet 网络核心采取分组交换

	层	息	网卡 PPTP、L2TP、SLIP、PPP	以太网帧最小长度为 64 字节
1	物理层	二进制传输	中继器、集线器（多端口中继器）	

交换技术：数据在网络中转发通常离不开交换机。交换机的功能包括：集线功能、中继功能、桥接功能、隔离冲突域功能等。

基本交换原理：交换机是一种基于 MAC 地址识别，能完成封装转发数据包功能的网络设备。交换机可以“学习”MAC 地址，并把其存放在内部地址表中，通过在数据的始发者和目标接收者之间建立临时的交换路径，使数据直接由源地址到达目的地址。

交换机需要实现的功能如下所述。

（1）转发路径学习。根据收到数据中的源 MAC 地址建立该地址同交换机端口的映射写入 MAC 地址表中。

（2）数据转发。如果交换机根据数据中的目的 MAC 地址在建立好的 MAC 地址表中查询到了，就向对应端口进行转发。

（3）数据泛洪。如果数据中的目的 MAC 地址不在 MAC 地址表中，则向所有端口转发也就是泛洪。广播帧和组播帧向所有端口（不包括源端口）进行转发。

（4）链路地址更新。MAC 地址表会每隔一定时间（如300s）更新一次。

3 章节问答

（1）TCP 与 UDP 有什么区别?

答：

	TCP	UDP
共同点	基于 IP 协议的传输层协议，可以端口寻址	
不同点	面向连接（连接管理）、三次握手、流量控制、差错校验和重传、IP 数据报按序接收不丢失不重复、可靠性强、牺牲通信量、效率低。	不可靠、无连接、错误检测功能弱，无拥塞控制、无流量控制，有助于提高传输的高速率性。 不对无序 IP 数据报重新排序、不负责重传、不消除重复 IP 数据报、不对已收到的数据报进行确认、不负责建立或终止连接，这些由 UDP 进行通信的应用程序进行处理。

相关协议	HTTP、FTP、Telnet、POP3、SMTP	DNS、DHCP、TFTP、SNMP

（2）DNS 查询的过程为什么有的是先查区域记录 hosts，有的是先查本地缓存？

答：本机（浏览器客户端）查询一般先查找本机 HOST 文件，没有相关映射时，查询域名服务器。域名服务器接收到请求时先查询本地缓存。

（3）递归查询跟迭代查询怎么区分？

答：递归查询：如果不知道 IP 地址，那么就以 DNS 客户的身份向其他域名服务器继续查询，适用于本地域名服务器和中介域名服务器。迭代查询：如果不知道 IP 地址，直接告知查询另外一个域名服务器 ，适用于根域名服务器。递归查询会向下探索，最终返回答案，迭代查询不会向下探索，会立即返回消息，可以只返回线索。

（4）存储网络架构里面的 DAS、NAS 和 SAN 怎么区分？

答：DAS 采用 I/O 总线架构，如 IDE 或 ATA 等将存储设备挂接在计算机中，实现数据存储。多种存储设备适合用作主机连接存储；包括硬盘驱动器、RAID 阵列、CD、DVD 和磁带驱动器。对主机连接存储设备进行数据传输的 I/O 指令是针对特定存储单元（例如总线 ID 和目标逻辑单元）的逻辑数据块的读和写。

NAS 和 SAN 都是基于网络构建存储系统的。网络存储采用面向网络的存储体系结构，使数据处理和数据存储分离，由专门的系统负责数据处理，存储设备或子系统负责数据的存储。网络存储结构通过网络连接服务器和存储资源，具有灵活的网络寻址能力和远距离数据传输能力，实现了在单一区域或多个区域可靠的数据存储、恢复，以及不同主机不同存储设备之间的资源共享。SAN 和 NAS 都可以用于集中管理存储，并供多主机（服务器）共享存储。但是，NAS 通常是基于以太网，而 SAN 可使用以太网和光纤通道。此外，NAS 注重易用性、易管理性、可扩展性和更低的总拥有成本（TCO），而 SAN 则注重高性能和低延迟。