第一章系统工程与信息系统基础

1系统工程

1.1 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。 【明确问题-确定目标-系统综合-系统分析-优化-系统决策-实施计划】 时间维：时间维即工作进程。 【规划阶段-拟定方案-研制阶段-生产阶段-安装阶段-运行阶段-更新阶段】 知识维：知识维即专业科学知识。【工程、医药、建筑、商业、法律、管理等】 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】**

1.2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

1．3系统工程生命周期方法

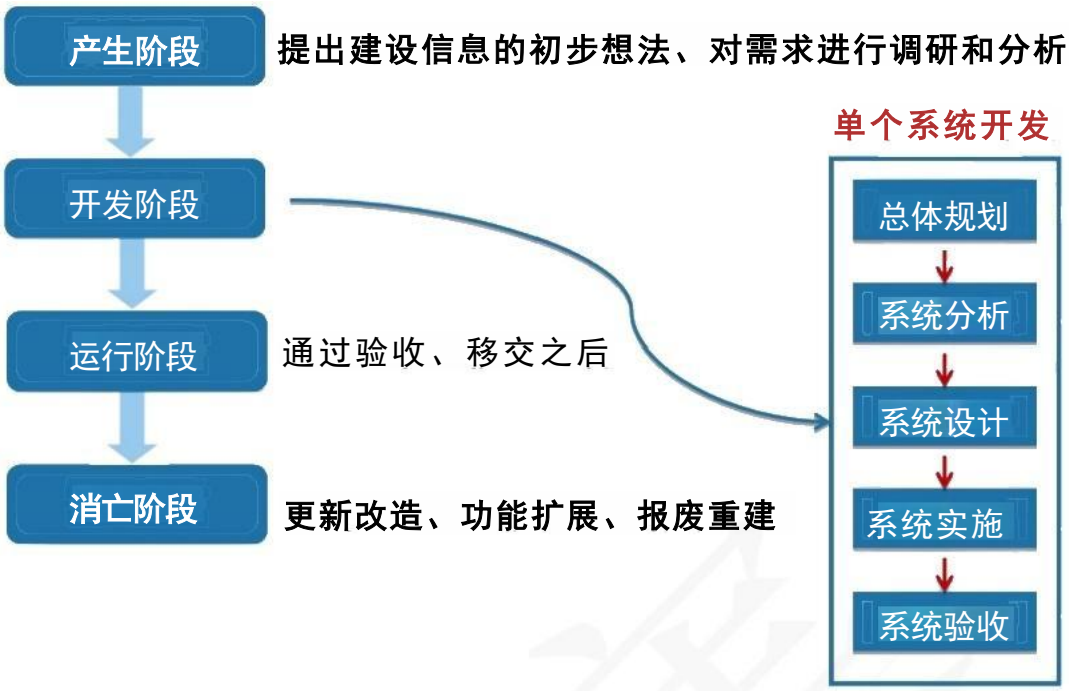
计划驱动方法：需求->设计->构建->测试->部署

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信息系统生命周期



2.1信息系统建设原则

高层管理人员介入原则	如：CIO介入
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
自顶向下规划原则	以此减少信息不一致的现象
工程化原则	引入【软件工程】
其他原则	创新性原则、整体性原则、发展性原则、经济性原则

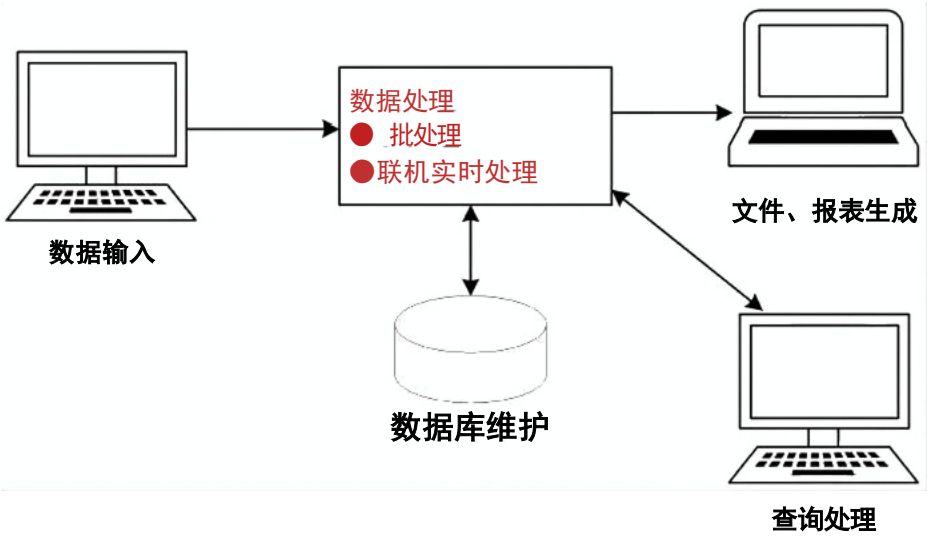
3 信息系统的分类

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	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。

【OAS】	
企业资源计划 【ERP】	打通供应链，集成，整合。

3.1 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。

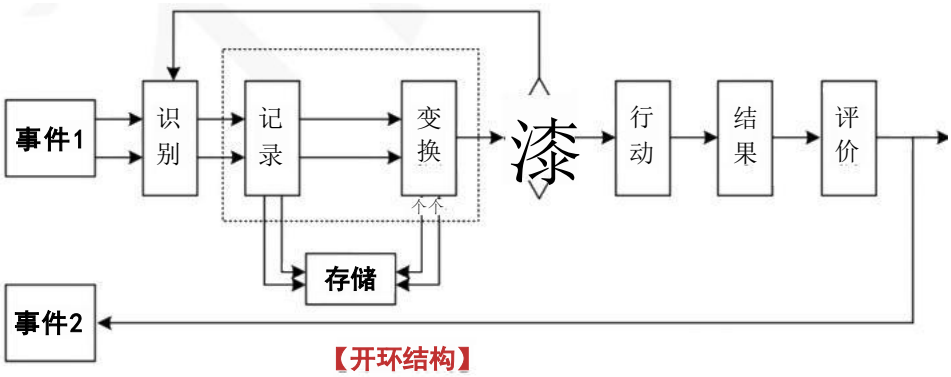


TPS 是服务于组织管理层次中最低层、最基础的信息系统。

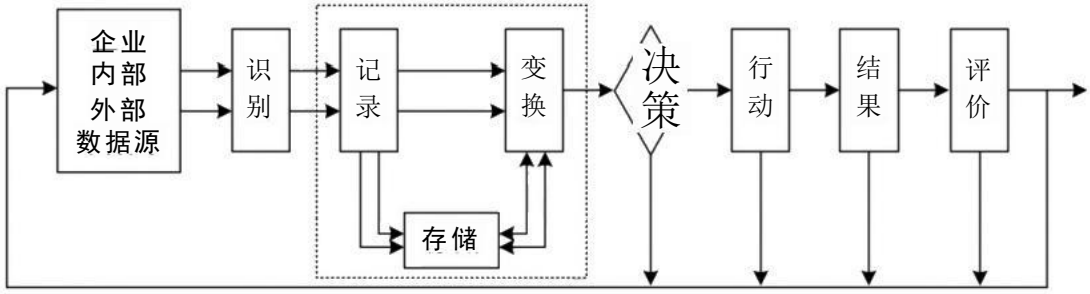
3.2管理信息系统【MIS】

【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。



【开环结构】



【闭环结构】

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】

3.3 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。

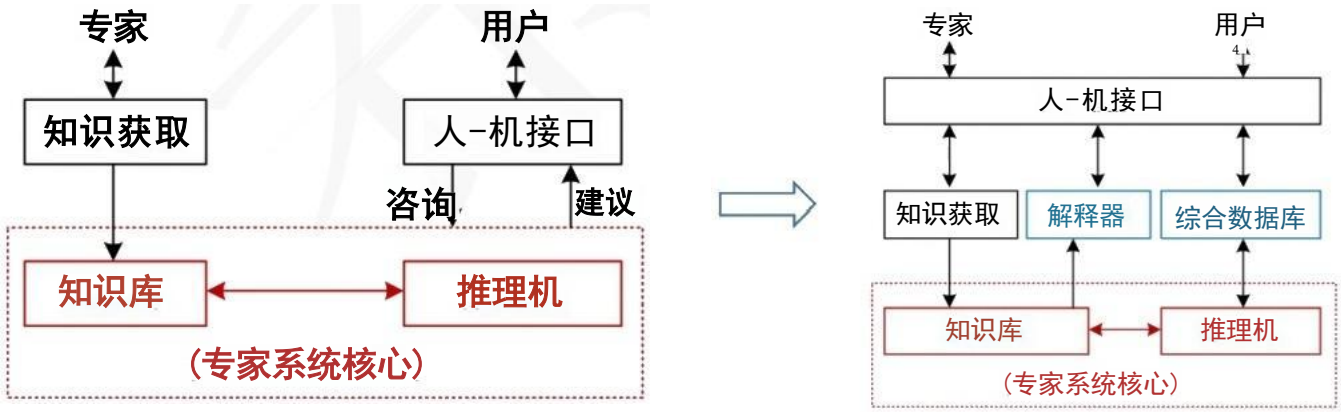
(3)DSS 主要用于解决半结构化及非结构化问题。

(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。

3.4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。

知识表 示 + 知 识推理技 术 = 专 家 系 统



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。

4政府信息化与电子政务

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。

5企业信息化与电子商务

5．1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。

5．2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。

5．3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。

5.4 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。

5．5信息系统战略规划 (ISSP)

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。**

5.5.1第一个阶段

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。

5.5.2 **第二个阶段**

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)。

5.5.3第三个阶段

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)。

5.6企业资源计划（ERP）

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。

5.7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，目的是与客户建立长期和有效的业务关系，最大限度地增加利润；核心是客户价值管理，提高客户忠诚度和留存率。

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。

5.8 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) 数据挖掘方法：

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。

5.9企业应用集成

5.9.1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

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

5.9.2企业门户

企业信息门户 (Enterprise Information Portal,EIP)：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（Enterprise Knowledge Portal,EKP）：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（Enterprise Application Portal,EAP）：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。

第二章软件工程

1 软件开发方法

(1)结构化开发方法

特点：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，系统开发过程工程化，文档资料标准化。**

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，此类原型在系统真正实现以后就放弃不用了。

演化型原型，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)。

(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。

从概念上讲，SO 方法有三个主要的抽象级别： 操作、服务、业务流程。

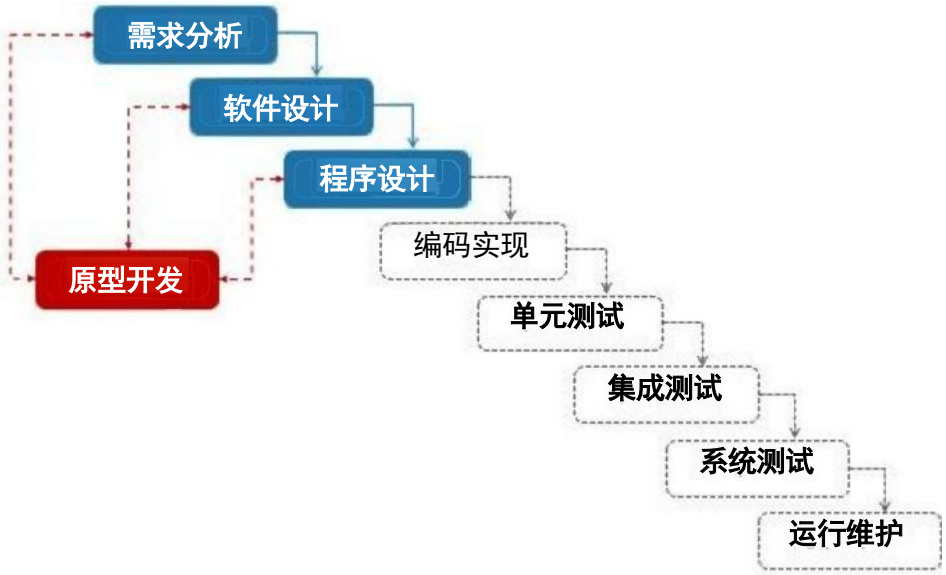
操作：代表单个逻辑工作单元 (LUW) 的事务。

服务：代表操作的逻辑分组。

业务流程：为实现特定业务目标而执行的一组长期运行的动作或活动。

2 软件过程模型

2.1 原型模型



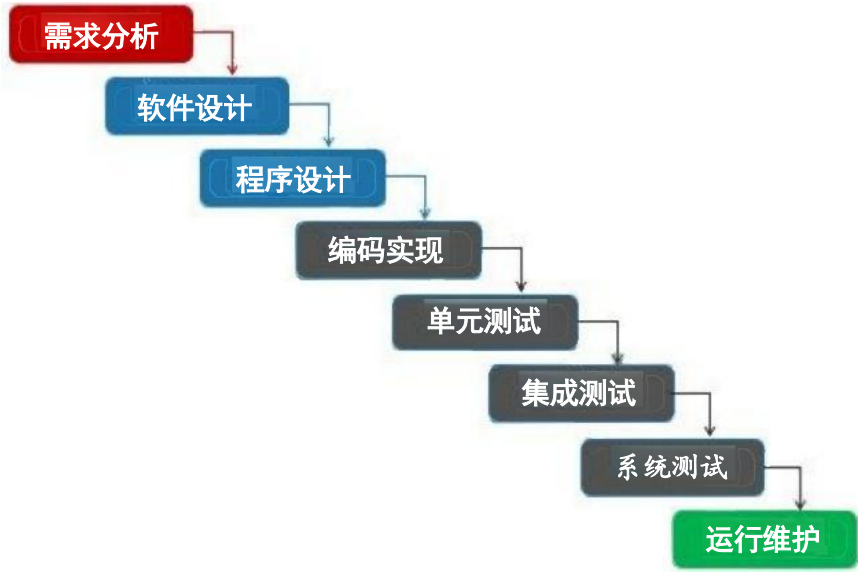
典型的原型开发方法模型。适用于需求不明确的场景，可以帮助用户明确需求。可以分为【抛弃型原型】与【演化型原型】

原型模型两个阶段：

1、原型开发阶段；

2、目标软件开发阶段。

2.2 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以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；

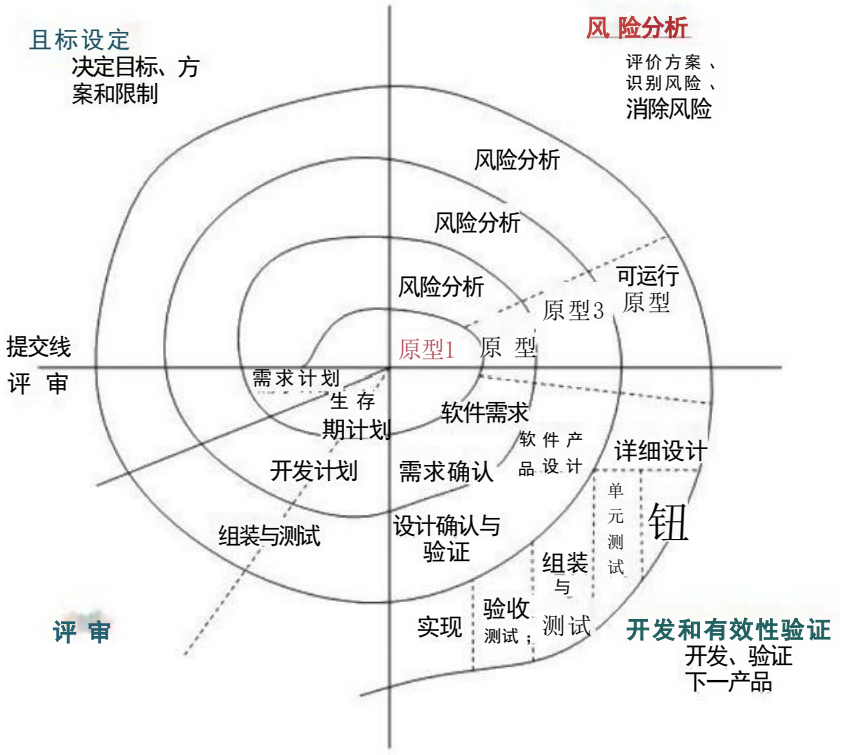
严格串行化，很长时间才能看到结果；

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。

2.3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。

2.4 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。

2.5V 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】。

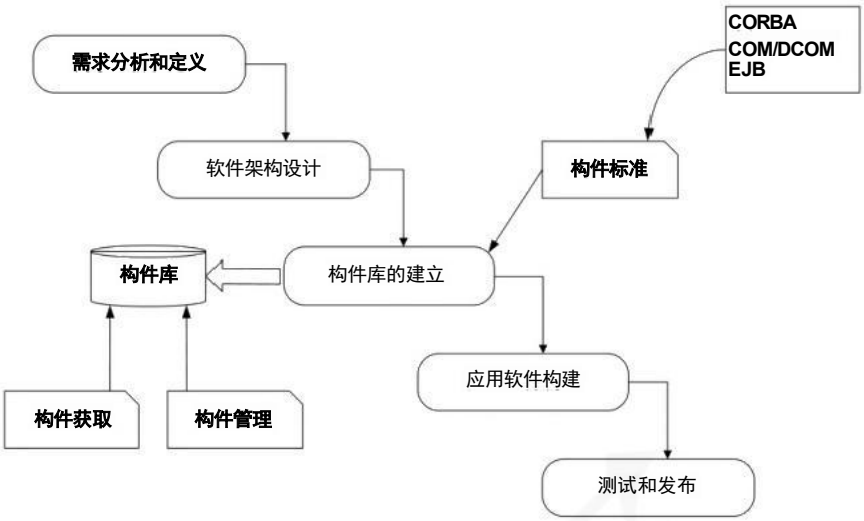
2.6 快速应用开发 RAD

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。

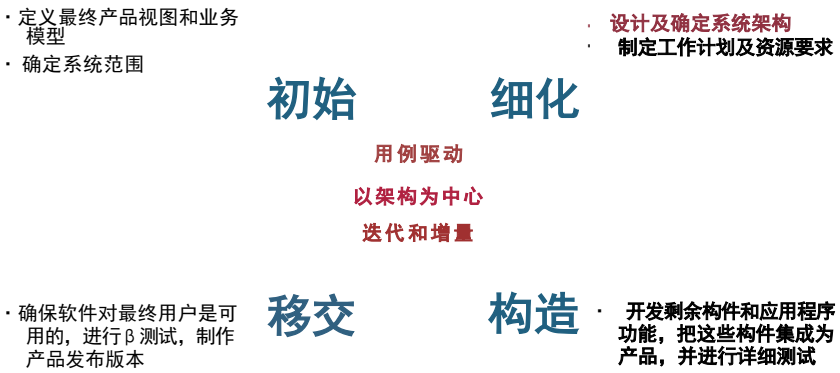
2. 7构件组装模型



【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。

2.8 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。

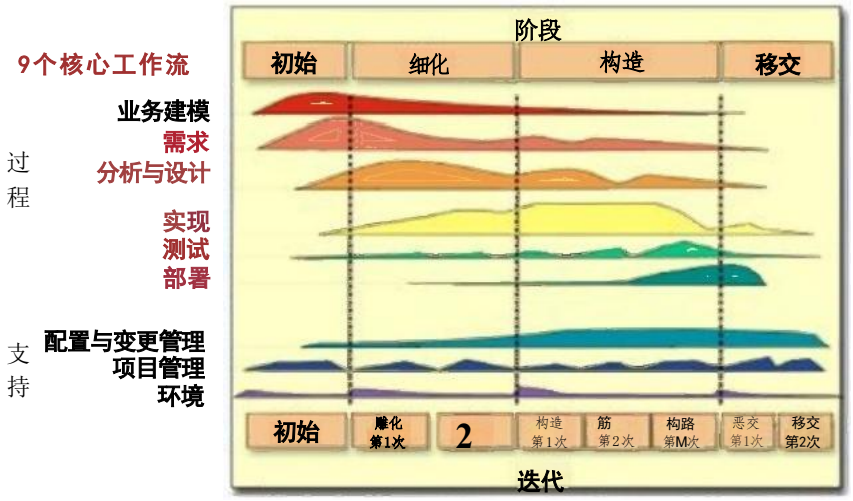
统一过程把一个项目分为四个不同的阶段：

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。

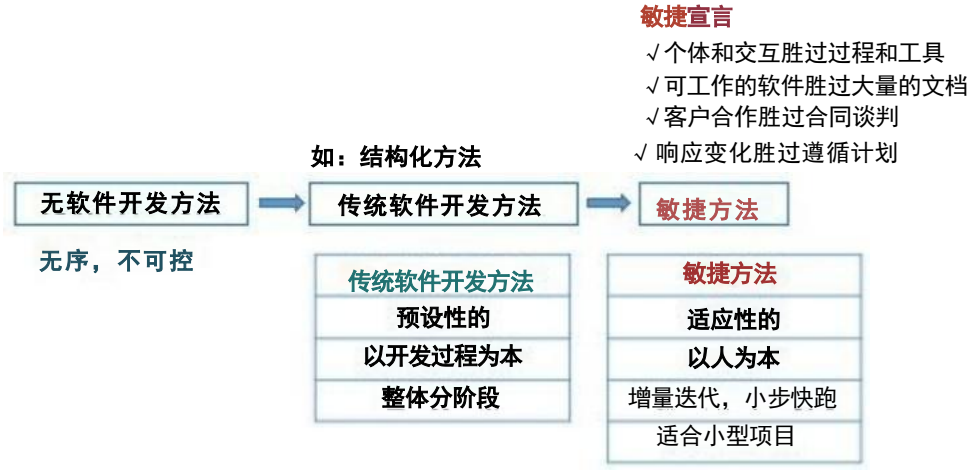
构造阶段：开发剩余构件和应用程序功能，把这些构件集成为产品，并进行详细测试。

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。

2.9 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)：在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。

3 基于构件的软件工程（CBSE）

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】：

1、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。

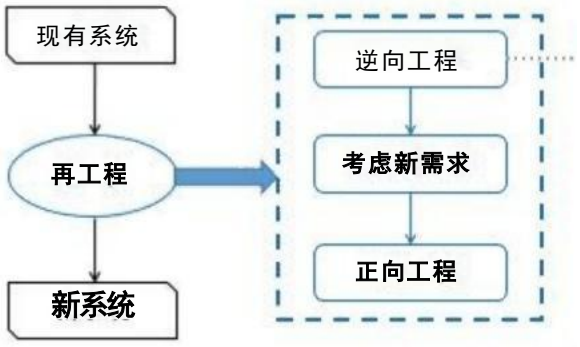
2、层次组装：被调用构件的“提供”接口必须和调用构件的“请求”接口兼容。

3、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、使用信息、部署。

4逆向工程



逆 向工程是设计的恢复过程

【逆向工程恢复信息的方法】
用户指导下的搜索与变换方法：实现级、结构级
变换式方法：实现级、结构级、功能级
基于领域知识的方法：功能级、领域级
铅板恢复法：实现级、结构级

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。

5 净室软件工程

强调以合理的成本开发出高质量的软件。理论基础主要是函数理论和抽样理论。它提倡开发者不需要进行单元测试(但还是需要传统的模块测试)，而是进行正确性验证和统计质量控制。因为高质量改进管理，降低风险及成本，满足用户需求，提供竞争优势。

技术手段：

统计过程控制下的增量式开发： 控制迭代

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）

正确性验证：净室工程的核心，它使软件质 量有了极大提高。

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。

6 需求工程

6.1 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。

软件需求是指用户解决问题或达到目标所需的条件或能力，是系统或系统部件要满足合同、标准、规范或其他正式规定文档所需具有的条件或能力，以及反映这些条件或能力的文档说明。

需求开发包括：需求获取、需求分析、形成需求规格、需求确认与验证

需求管理包括：变更控制、版本控制、需求跟踪、需求状态跟踪

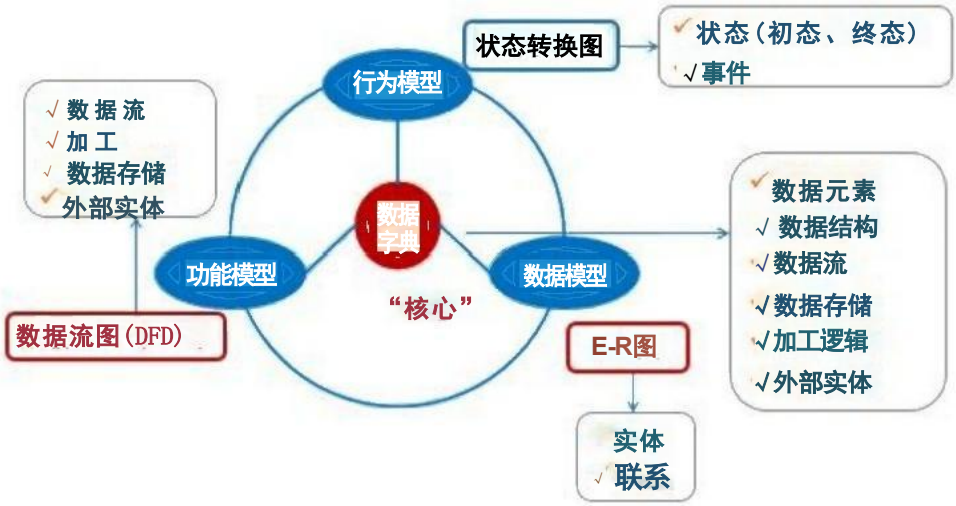
6.2 需求开发

6.2.1 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。

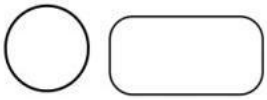
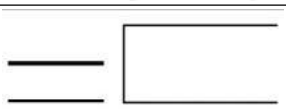
联合需求计划(JRP)	高度组织的群体会议，各方参与，了解想法，消除分歧，交互好，成本高。
问卷调查	用户多，无法——访谈，成本低。
现场观察	针对较为复杂的流程和操作。
原型化方法	通过简易系统方式解决早期需求不确定问题。
头脑风暴法	一群人围绕新业务，发散思维，不断产生新的观点。

6.2.2 需求分析



结构化需求分析（SA)

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	

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)

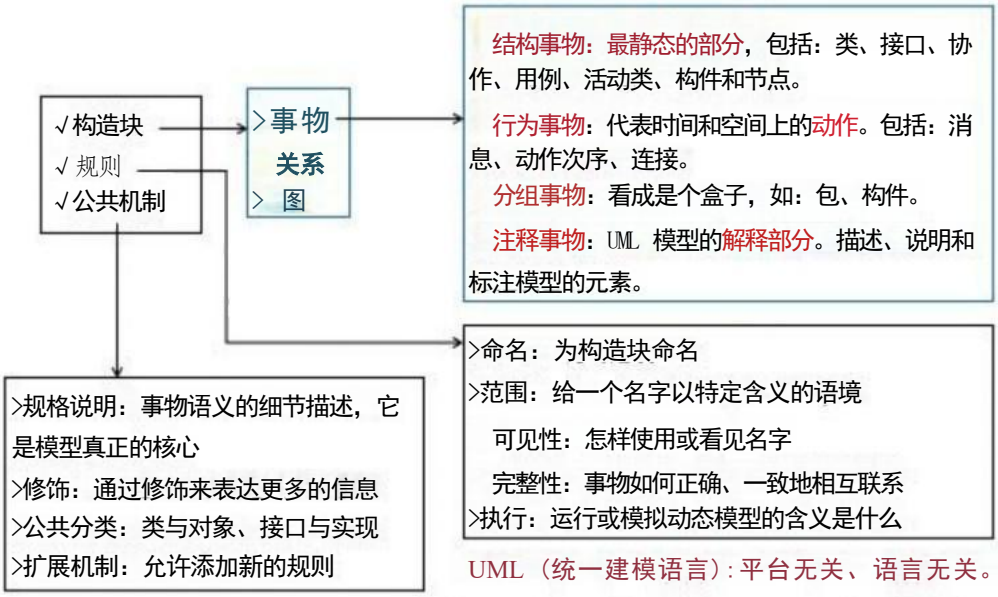
实体类映射需求中的每个实体，实体类保存需要存储在永久存储体中的信息，例如，在线教育平台系统可以提取出学员类和课程类，它们都属于实体类。

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。

6.2.3 UML

(1)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，分别是包和构件。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。

(2)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：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

其中继承关系可分为：

【取代继承（Replacement Inheritance）】指子类可以替换父类（如遵循 Liskov 替换原则），但子类的能力通常与父类相同或兼容，并不强调添加新内容导致能力扩展。

【受限继承（Restricted Inheritance）】子类继承父类，但限制了某些功能（如缩小方法范围或值域），导致子类的能力小于父类。

【特化继承（Specialization Inheritance）】子类完全继承父类的所有特性（属性和方法），并在此基础上添加新的特性或修改行为，从而使子类的能力大于或等于父类。这体现了“is-a”关系，子类是父类的特化或扩展。

【包含继承（Inclusion Inheritance）】通常指子类包含父类的所有特性，但更侧重于类型包含或子类型关系（如接口继承），并不强调添加新内容。

6.3需求定义

严格定义法： 所有需求都能够被预先定义

开发人员与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

原型法：并非所有的需求都能在开发前被准确的说明

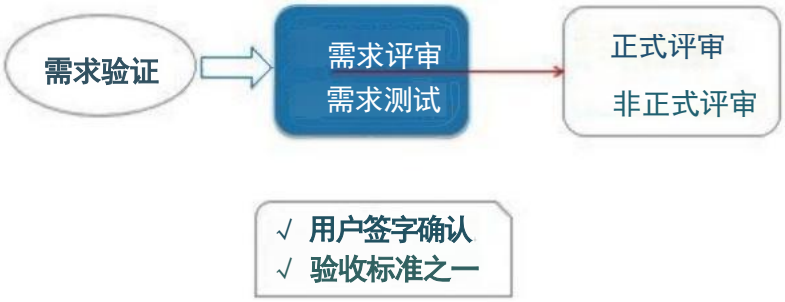
项目参加者之间通常都存在交流上的困难

需要实际的、可供用户参与的系统模型

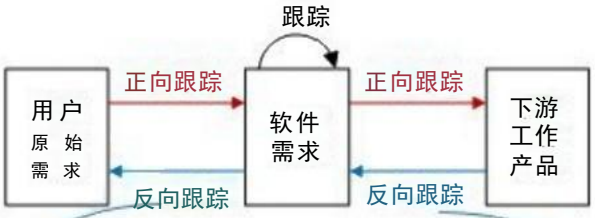
有合适的系统开发环境

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

6.4需求确认与验证



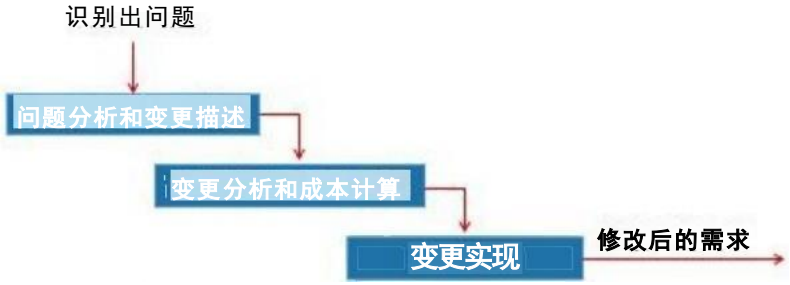
6.5需求跟踪



用例 / 原始需求	UC-1	UC-2	UC-3		UC-n
FR-1	√				
FR-2			√		
……					
FR-m					

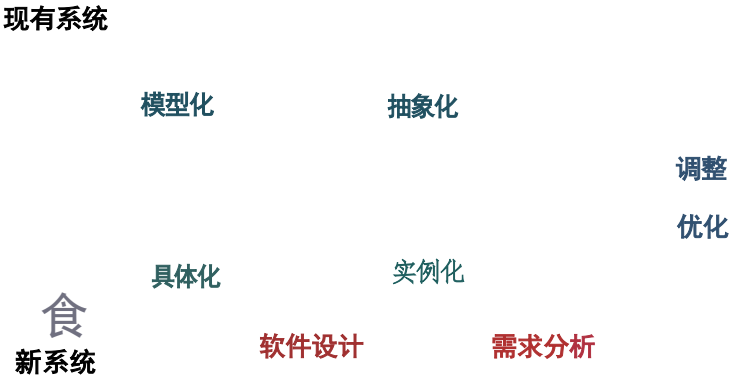
元素 / 用例	功能点	设计元素	代码模块	测试用例
UC-1				
UC-2				
……				
UC-n				

6.6 需求变更管理过程



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。

6.7软件系统建模



7 系统设计

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。**

黄金三法则： **置于用户控制之下、减少用户的记忆负担、保持界面的一致性。**

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。

7.3 面向对象设计

类的分类：

面向对象设计原则：

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(最少知识原则)：一个对象应当对其他对象有尽可能少的了解

8 系统测试

(1)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】(最强)				
	【逻辑覆盖】(由强到弱排列)		条件覆盖	
修正条件/判定	条件组合	条件/判定覆盖	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。

(2)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。

(3)软件测试与软件调试

软件调试方法：

蛮力法：主要思想是“通过计算机找错”，低效，耗时。

回溯法：从出错处人工沿控制流程往回追踪，直至发现出错的根源。复杂程序由于回溯路径多，难以实施。

原因排除法：主要思想是演绎和归纳，用二分法实现。

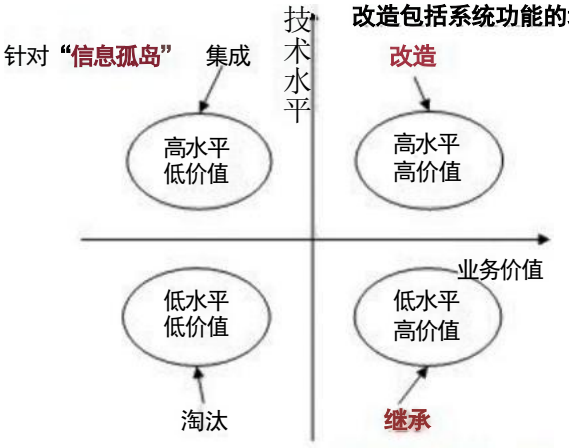
软件测试与软件调试的区别： 调试是测试之后的活动，测试和调试在目标、方法和思路上都有所不同。

软件测试	软件调试
目的是找出存在的错误	目的是定位错误并修改程序以修正错误
从一个已知的条件开始，使用预先定义的过程，有预知的结果	从一个未知的条件开始，结束的过程不可预计
测试过程可以事先设计，进度可以事先确定	调试不能描述过程或持续时间

9系统运行与维护

9．1遗留系统与系统转换计划

遗留系统演化策略：



改造包括系统功能的增强和数据模型的改造两个方面

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

新旧系统是转换策略：

现有系统

新系统

直接转换策略

现有系统

新系统

并行转换策略

现有系统 新系统

分段转换策略

9.2 软件维护

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。

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】: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，以使系统适应各类变化而不被淘汰。经典实例： 【专用】 改 【通用】。

第三章计算机系统基础知识

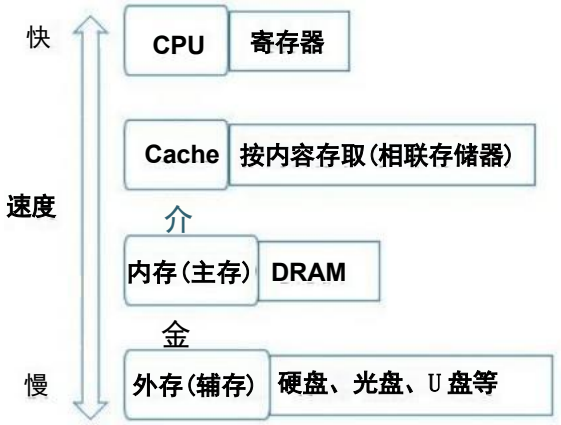
1 计算机系统组成

1.1 存储系统

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。



1.2计算机系统组成

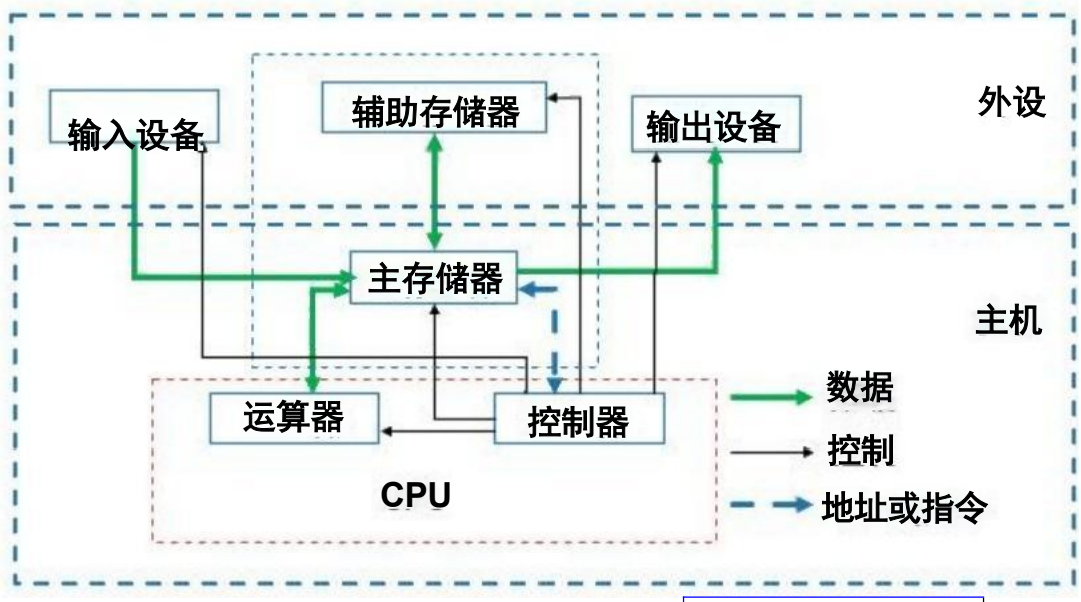
处理器、存储器、总线、接口和外部设备等

操作系统、编译工具等

办公软件、娱乐软件、信息系统软件等

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13、15、17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。	指令与数据分开存储，可以并行读取，有较高的数据吞吐率。 有4条总线：指令和数据的数据总线与地址总线。	一般用于嵌入式系统处理器。 注：DSP属于哈佛结构



冯 · 诺依曼计算机组成

· 冯 · 诺依曼结构
· 哈佛结构

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。

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：

(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

2 操作系统

2.1操作系统概述

操作系统（OS,Operating System)

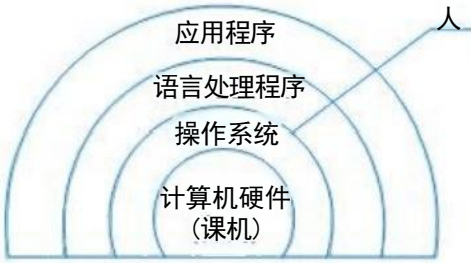
◆人机之间的接口

◆ 应用软件与硬件之间的接口

◆ 为应用程序的开发和运行提供一个高效率的平台

◆ 管理系统的硬件、软件、数据资源

控制程序运行



◆ 进程管理

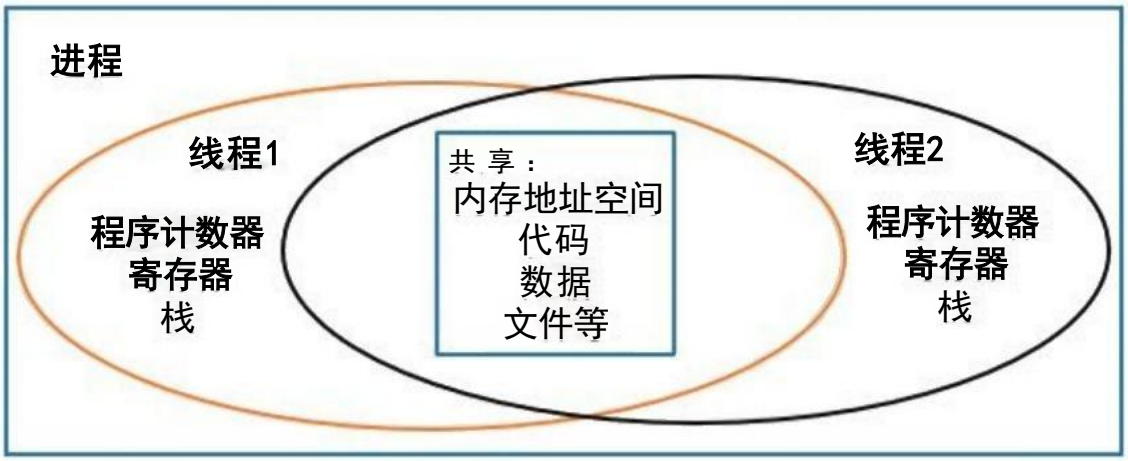
◆存储管理 **◆ 作业管理**

◆ 文件管理 **◆ 设备管理**

2.2 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)**

2.3进程管理



运行：当一个进程在CPU 上运行时。

（单处理机处于运行态的进程只有一个，多进程在CPU 上交替运行）

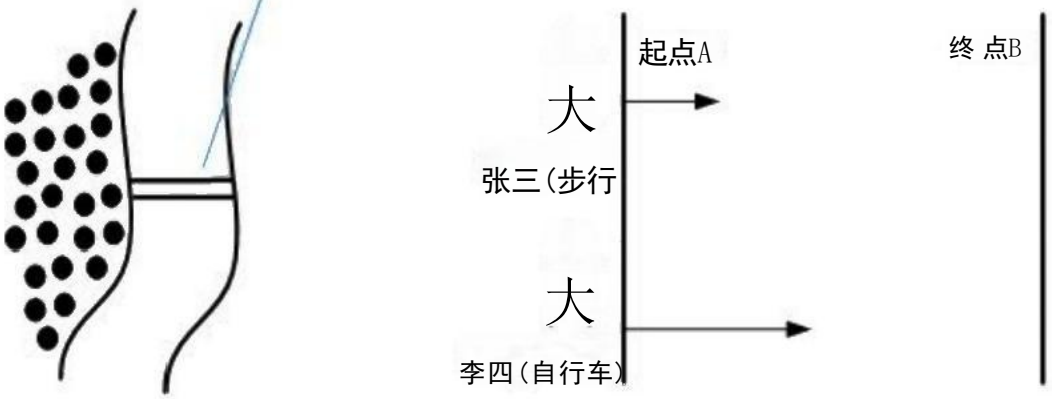
就绪：一个进程获得了除CPU 外的一切所需资源，一旦得到处理机即可运行。

阻塞：阻塞也称等待或睡眠状态， 一个进程正在等待某一事件发生（例如请求I/O、等待I/O 完成等）而暂时停止运行，此时即使把 CPU 分配给进程也无法运行，故称进程处于阻塞状态。

2．4进程同步与互斥

▲临界资源：诸进程间需要互斥方式对其进行共享的资源。

（进程中访问临界资源的那段代码称为临界区）



互斥：如千军万马过独木桥　　同步：速度有差异，在一定情况停下等待。

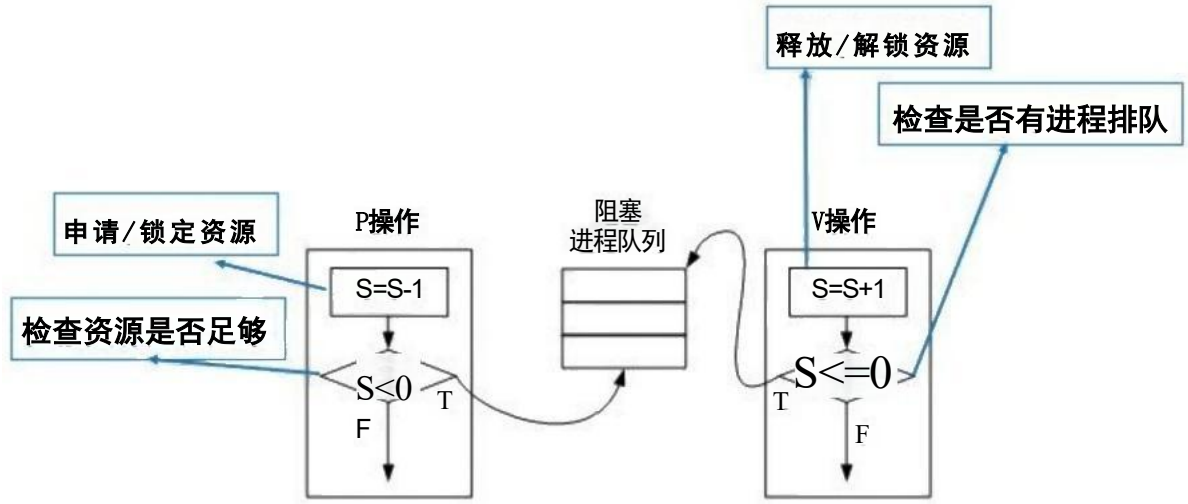
▲间接制约关系　　▲直接制约关系

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。

P 是荷兰语的Passeren,V　是荷兰语的Verhoog。



前驱图的表示：

1个箭头表示1个前驱关系

A 有箭头指向D，　则记录为（A,D）

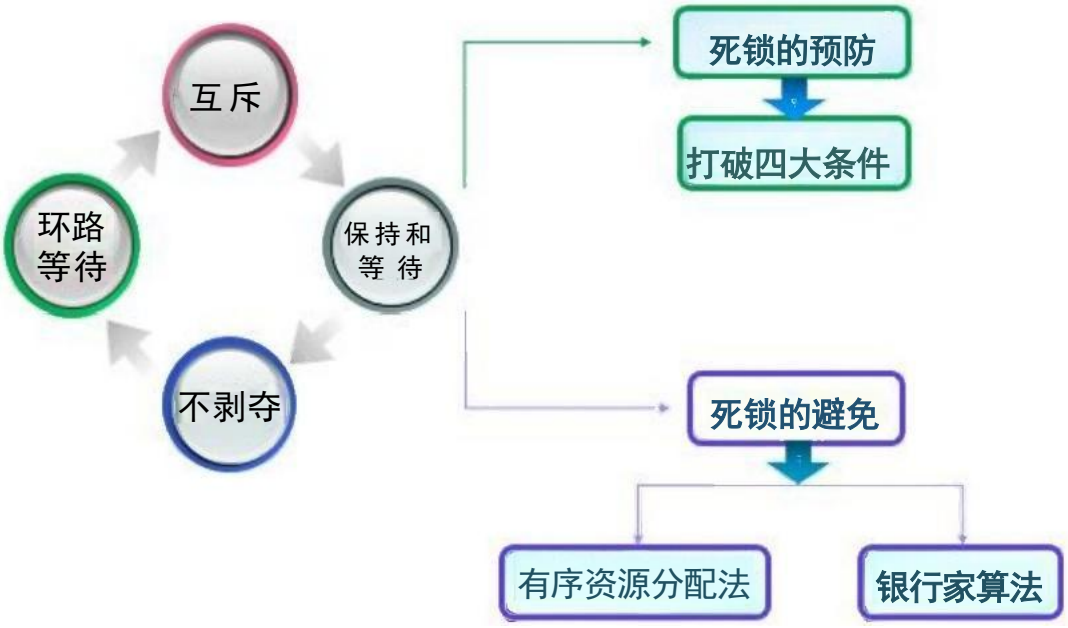
没有前驱进程的节点是起始进程，没有后继进程的节点是终结进程。

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，有前驱就有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。

2.5 死锁



2.6银行家算法

(1)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；

(2)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；

(3)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。

2.7 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

2.8 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）。

3文件系统

位示图 √ 空闲区表法(空闲文件目录)

√ 空闲链表法

√ 位示图法

√ 成组链接法

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	0	0	0	1	1	1	0	0	1	0	1	1	1	0
1	0	0	0	1	1	1	1	1	1	0	0	0	0	1	1	1
2	1	1	1	0	0	0	1	1	1	1	1	1	1	0	0	0
3																
…																
15																

位示图

4 系统性能

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)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

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。