调制解调：模拟信号转数字信号，再将数字信号转模拟信号

CMMI(能力成熟度模型):

初始级：过程不可预测

已管理级：项目级管理

已定义级：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

优化级：持续改进，创新导向

## 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**

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

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

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

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

## 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，迭代增量的软考开发过程

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

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

ABSD基于体系结构的软件设计

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

企业集成:网络集成，数据集成，会聚集成，服务集成，接口集成

架构复用：构造/获取可复用资产， 管理..， 使用..

安全审计的4要素：控制目标，安全漏洞，控制措施，控制测试

## 计算机网络

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

有干扰下最大传输速率(香农公式)=带宽\*log2(1+S/N) bps

## 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

## 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

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：段大小可变

## 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

进程视图：多进程怎么交互(性能)

物理(部署)视图：服务器部署

逻辑视图：功能怎么分

开发(实现)视图：代码怎么实现

场景视图：活动图，序列图，使用场景

**开发模型**

瀑布模型：软件需求明确

原型(演化)模型:需求不明确，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：系统模块化程度较高时使用

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

**分析方式**

结构化分析：自顶向下，逐层分解，面向数据流(数据流图)

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)

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

开发期质量属性：易理解、可扩展、可重用、可测试、可维护、可移植

运行期质量属性：性能、安全性、可伸缩性、互操作性、可靠性、可用性、鲁棒性

可用性：可用时间和故障时间

架构评估中的质量属性：性能、可靠性、可用性、安全性、可修改性、功能性、可变性、互操作性

网络信息安全

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：控制访问权限

安全审计：各种操作的记录

设计模式

创建型：单例，原型，工厂方法，抽象工厂，建造者

结构型：桥接，适配器，组合，装饰，外观，享元，代理

行为型：剩下的

权衡点：对多个质量属性进行权衡

风险点：存在潜在隐患

敏感点：会影响某个质量属性

**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

## 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

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)

## 法律

国家机密保密期：20年

## 数据库

模式

内模式(机器): 分区，页，块，底层结构

模式/概念模式(开发)：索引，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

## 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

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

















