高频知识点50条

1、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。**

2、专家系统【ES】

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。

3、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。

4、 原型法

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。

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。

5、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。

6、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。

7、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。

8、构件的组装

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。

9、逆向工程及其相关的概念

(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 (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。

10、净室软件工程

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。

11、UML中的4种事物

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。

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。

12、UML 图(静态图/结构图)

类图：描述一组类、接口、协作和它们之间的关系。

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：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。

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。

组合结构图。

13、UML 图(动态图/行为图)

用例图：系统与外部参与者的交互

顺序图：是一种交互图，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。

通信图：也叫协作图。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：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 。

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。

14、UML 图关系之用例图关系

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。

15、UML 图关系之类图/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(继承)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。

16、 **系统建模方法**

(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。

17、 **系统测试**

一般来说，系统测试的主要内容包括功能测试、健壮性测试、性能测试、用户界面测试、安全性测试、安装与反安装测试等。

系统测试活动与步骤：制订系统测试计划(进行人员以及任务的确定，明确测试范围、测试方法、测试环境与辅助工具)、设计系统测试用例(如：等价类划分、边界值分析等测试方法的应用)、执行系统测试(执行设计好的测试用例，并记录结果)、缺陷管理与改错(消除已发现的错误)。

18、 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**

19、 **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】。

20、软件架构风格

架构风格定义了用于描述系统的术语表和一组指导构建系统的规则。

五大架构风格以及子风格划分：

数据流风格：批处理、管道-过滤器

调用/返回风格：主程序/子程序、面向对象、分层架构

独立构件风格：进程通信、事件驱动系统(隐式调用)

虚拟机风格：解释器、规则系统

以数据为中心：数据库系统、黑板系统、超文本系统

21、MVC

Model (模型)是应用程序中用于处理应用程序数据逻辑的部分。通常模型对象负责在数据库中存取数据。

View (视图)是应用程序中处理数据显示的部分。通常视图是依据模型数据创建的。

Controller(控制器)是应用程序中处理用户交互的部分。通常控制器负责从视图读取数据，控制用户输入，并向模型发送数据。

J2EE 体系结构中：

视图 (View):JSP

控制 (Controller):Servlet

模型 (Model):Entity Bean、Session Bean

22、SOA 关键技术

【UDDI】 是 Web 服务集成的一个体系框架，包含了服务描述与发现的标准规范。

【WSDL】 服务描述语言，三个基本属性(服务做些什么/如何访问服务/服务位于何处。

【SOAP】 基于XML 的协议，在分布式环境中交换信息。4个部分：SOAP 封装(消息内容，谁发的，谁接收)、SOAP 编码规则(数据类型实例)、 SOAP RPC (远程过程调用和应答协定)、SOAP 绑定(使用底层协议交换信息)。

23、REST　概念

REST(Representational State Transfer,表述性状态转移)是一种通常使用HTTP 和 XML 进行基于 Web 通信的技术，可以降低开发的复杂性，提高系统的可伸缩性。

REST 的5个原则：①网络上的所有事物都被抽象为资源；②每个资源对应一个唯一的资源标识；③通过通用的连接件接口对资源进行操作；④对资源的各种操作不会改变资源标识；⑤所有的操作都是无状态的。

24、REST API 命令

命令	用途
GET	从指定的资源请求数据，只进行数据检索，不进行其他操作
POST	将数据发送到服务器进行创建，通常用于上传文件或提交表单
PUT	更新目标资源的所有当前表示，使用上传的内容进行替换
DELETE	删除指定的资源
HEAD	与GET方法类似，但只传输状态行和头部信息
PATCH	对资源进行部分修改

25、 云原生架构设计原则

服务化原则：使用微服务

弹性原则：可根据业务变化自动伸缩

可观测原则：通过日志、链路跟踪和度量

韧性原则：面对异常的抵御能力

所有过程自动化原则：自动化交付工具

零信任原则：默认不信任网络内部和外部的任何人/设备/系统

架构持续演进原则：业务高速迭代情况下的架构与业务平衡

26、 云计算分类

按服务类型分类：

SaaS【软件即服务】	基于多租户技术实现，直接提供应用程序
PaaS【平台即服务】	虚拟中间件服务器、运行环境和操作系统

laas【基础设施即服务】	包括服务器、存储和网络等服务

按部署方式分类：

公有云：面向互联网用户需求，通过开放网络提供云计算服务

私有云：面向企业内部提供云计算服务

混合云：兼顾以上两种情况的云计算服务

27、边云协同的分类

【资源协同】边缘节点有基础设施资源的调度管理能力，可与云端协同。

【数据协同】边缘节点采集数据并初步分析，再发给云端做进一步处理。

【智能协同】分布式智能，云端做集中式模型训练，再将模型下发到边缘节点。

【应用管理协同】边缘节点提供应用部署与运行环境，云端主要提供应用开发、测试环境。

【业务管理协同】边缘节点提供模块化、微服务化的应用/数字孪生/网络等应用实例；云端主要提供按照客户需求实现应用/数字孪生/网络等的业务编排能力。

【服务协同】边缘节点按照云端策略实现部分ECSaaS 服务，通过 ECSaaS 与云端 SaaS 的协同实现面向客户的按需 SaaS 服务；云端主要提供SaaS 服务在云端和边缘节点的服务分布策略，以及云端承担的 SaaS 服务能力。

28、ADL

(1)什么是ADL

ADL 是这样一种形式化语言，它在底层语义模型的支持下，为软件系统的概念体系结构建模提供了具体语法和概念框架。

C2SADL【基于组件和消息的软件架构描述语言】

Wright【 分布、并发类型的架构描述语言】

ACME【架构互换语言】

UniCon【 基于组件和连接的架构描述语言】

Rapide【 基于事件的架构描述语言】

其他【Darwin、MetaH、Aesop、Weaves、SADL、xADL】

(2)ADL 的三个基本元素

ADL 的三个基本元素：

构件：计算或数据存储单元；

连接件：用于构件之间交互建模的体系结构构造块及其支配这些交互的规则；

架构配置：描述体系结构的构件与连接件的连接图。

29、 **基于架构的软件设计（ABSD）**

ABSD 方法是架构驱动，即强调由业务【商业】、质量和功能需求的组合驱动架构设计。ABSD 方法有三个基础。第一个基础是功能的分解。在功能分解中，ABSD 方法使用已有的基于模块的内聚和耦合技术；第二个基础是通过选择架构风格来实现质量和业务需求；第三个基础是软件模板的使用。软件模板利用了一些软件系统的结构。

开发过程：架构需求、架构设计、架构文档化、架构复审、架构实现、架构演化。

30、 **软件系统质量属性**

软件系统质量属性是一个系统的可测量或者可测试的属性，用来描述系统满足利益相关者需求的程度。基于软件系统的生命周期，可以将软件系统的质量属性分为开发期质量属性和运行期质量属性2个部分。

开发期质量属性主要指在软件开发阶段所关注的质量属性，主要包含6个方面：

易理解性、可扩展性、可重用性、可测试性、可维护性、可移植性。

运行期质量属性主要指在软件运行阶段所关注的质量属性，主要包含7个方面：

性能、安全性、可伸缩性、互操作性、可靠性、可用性、鲁棒性。

31、 **风险点与非风险点、敏感点与权衡点**

风险点：系统架构风险是指架构设计中潜在的、存在问题的架构决策所带来的隐患。

非风险点：是指不会带来隐患， 一般以“XXX 要求是可以实现【或接受】的”方式表达。

敏感点：敏感点是一个或多个构件(和/或构件之间的关系)的特性，它能影响系统的某个质量属性。

权衡点：影响多个质量属性的特性，是多个质量属性的敏感点。

32、 **质量效用树**

ATAM 方法采用效用树 (Utility tree) 这一工具来对质量属性进行分类和优先级排序。效用树的结构包括：树根一质量属性一属性分类一质量属性场景(叶子节点)。

得到初始的效用树后，需要修剪这棵树，保留重要场景(通常不超过50个)，再对场景按重要性给定优先级(用 H/M/L 的形式)，再按场景实现的难易度来确定优先级(用H/M/L 的形式)，这样对所选定的每个场景就有一个优先级对(重要度、难易度)，如 (H、L) 表示该场景重要且易实现。

33、 **三种经典缓存使用模式：**

Cache-Aside pattern【旁路缓存模式】将缓存的读取和更新操作与数据库的操作分开处理。适用场景：读多写少、更新不频繁、对数据一致性要求不是非常严格、高并发下的读性能优化

Read-Through/Write-Through【 读写穿透】将缓存与数据库紧密结合，使得应用程序在读写数据时都通过缓存层进行操作。

Write-Behind【 异步缓存写入】本机制数据先写入缓存，再异步批量写入数据库。

34、Redis 和 MemCache 对比

工作	MemCache	Redis
数据类型	简单key/value结构	丰富的数据结构
持久性	不支持	支持
分布式存储	客户端哈希分片/一致性哈希	多种方式，主从、Sentinel、Cluster等
多线程支持	支持	Redis5.0及以前版本不支持
内存管理	私有内存池/内存池	无
事务支持	不支持	有限支持
数据容灾	不支持，不能做数据恢复	支持，可以在灾难发生时，恢复数据

35、 **负载均衡**

负载均衡技术：

(1)应用层负载均衡： http 重定向、反向代理服务器；

(2)传输层负载均衡： DNS 域名解析负载均衡、基于NAT 的负载均衡；

(3)硬件负载均衡： F5；

(4)软件负载均衡： LVS、Nginx、HAproxy。

负载均衡算法：

静态算法(不考虑动态负载)：

轮转算法：轮流将服务请求(任务)调度给不同的节点(即：服务器)。

加权轮转算法：考虑不同节点处理能力的差异。

源地址哈希散列算法：根据请求的源IP 地址，作为散列键从静态分配的散列表找出对应的节点。

动态算法(考虑动态负载)：

最小连接数算法：新请求分配给当前活动请求数量最少的节点，每个节点处理能力相同的情

况下。

加权最小连接数算法：考虑节点处理能力不同，按最小连接数分配。

加权百分比算法：考虑了节点的利用率、硬盘速率、进程个数等，使用利用率来表现剩余处理能力。

36、 软件架构复用

软件架构复用类型：

(1)机会复用：开发过程中，只要发现有可复用资产，就对其进行复用。

(2)系统复用：开发之前，要进行规划，以决定哪些需要复用。

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

可复用资产范围：需求、架构设计、元素、建模与分析、测试、项目规划、过程、方法和工具、人员、样本系统、缺陷消除。

37、 信息安全的5个基本要素

机密性是指网络信息不泄漏给非授权的用户、实体或程序，能够防止非授权者获取信息。

完整性是指网络信息或系统未经授权不能进行更改的特性。

可用性是指合法许可的用户能够及时获取网络信息或服务的特性。

可控性是指可以控制授权范围内的信息流向及行为方式。

可审查性是指对出现的信息安全问题提供调查的依据和手段。

38、 国产密码算法

SM1： 对称加密，分组长度和密钥长度都为128比特，广泛应用于电子政务、电子商务及国民经济的各个应用领域。

SM2： 非对称加密，用于公钥加密算法、密钥交换协议、数字签名算法，国家标准推荐使用素数域256位椭圆曲线。

SM3： 杂凑算法，杂凑值长度为256比特，适用于商用密码应用中的数字签名和验证。

SM4： 对称加密，分组长度和密钥长度都为128比特，适用于无线局域网产品。

SM9： 标识密码算法，不需要申请数字证书，适用于互联网应用的各种新兴应用的安全保障。

39、 区块链的特点

去中心化：由于使用分布式核算和存储，不存在中心化的硬件或管理机构，任意节点的权利和义务都是均等的，系统中的数据块由整个系统中具有维护功能的节点来共同维护。

开放性：系统是开放的，如：区块链上的【交易信息是公开的】，不过【账户身份信息是高度加密的】。

自治性：区块链采用基于协商一致的规范和协议(比如一套公开透明的算法)使得整个系统中的所有节点能够在去信任的环境自由安全的交换数据，使得对“人”的信任改成了对机器的信任，任何人为的干预不起作用。

安全性(信息不可篡改)：数据在多个节点存储了多份，篡改数据得改掉51%节点的数据，这太难。同时，还有其它安全机制，如：比特币的每笔交易，都由付款人用私钥签名，证明确实是他同意向某人付款，其它人无法伪造。

匿名性(去信任)：由于节点之间的交换遵循固定的算法，其数据交互是无需信任的(区块链中的程序规则会自行判断活动是否有效)，因此交易对手无须通过公开身份的方式让对方自己产生信任，对信用的累积非常有帮助。

40、 区块链的四大基础技术支柱

分布式存储：简单来说，就是一种将数据分散存储到多个地方的数据储存技术，而且存储的数据可在多个参与者之间共享，人人可以参与，并具有相同的权力， 一起记录数据，主要起到了数据储存的功能。

共识机制：因为区块链的分布式网络中，没有中央权威。因此，网络需要一个决策机制来促成参与者达成一致。而共识机制就是一种协调大家处理数据的机制。共识机制就决定了这些数据中，谁获得数据的记账权。

智能合约：是一种旨在以信息化方式传播、验证或执行合同的计算机协议。智能合约可以在没有第三方的情况下，也能进行可信的交易，而且这些交易可追踪且不可逆转。所以，智能合约在系统中，主要起到了数据的执行作用。【执行智能合约的机制：去中心化网络+共识机制+受激励的矿工】

密码学：是一种特殊的加密和解密技术，区块链系统中，应用了多种多样的密码学技术，包括哈希算法、公钥私钥、数字签名等等，以此来保证整个系统的数据安全，并且证明了数据的归属。

41、TCP/IP 协议簇

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

42、嵌入式微处理器

嵌入式微处理器主要用于处理相关任务。由于嵌入式系统通常都在室外使用，可能处于不同环境，因此选择处理器芯片时，也要根据不同使用环境选择不同级别的芯片。其主要因素是芯片可适应的工作环境温度。通常，我们把芯片分为民用级、工业级和军用级。

民用级器件的工作温度范围：0~70℃

工业级器件的工作温度范围：　-40~85℃

军用级器件的工作温度范围：　-55~150℃

当然，除了环境温度外，环境湿度、震动、加速度等也是应考虑的因素。

43、嵌入式操作系统的分类

嵌入式操作系统通常分为两类：

（1）嵌入式实时操作系统（面向控制、通信等领域）

兼具嵌入式操作系统的特点和实时操作系统的特点。实时操作系统的最核心特点是实时性强。

如 WindRiver 公司VxWorks、ATI 公司Nucleus 等；

（2）非实时嵌入式操作系统（面向消费电子产品）

如 Google 公司的Android、Apple 公司的 iOS，以 及 Microsoft 公司的WinCE 等 。

44、嵌入式数据库分类

按照数据库存储位置的不同而进行分类是目前广泛采用的分类方法，它可以划分为三类：

基于内存方式 (MMDB)： 基于内存的数据库系统是实时系统和数据库系统的有机结合。

基于文件方式 (FDB)： 基于文件的数据库系统就是以文件方式存储数据库数据，即数据按照一定格式储存在磁盘中。使用时由应用程序通过相应的驱动程序甚至直接对数据文件进行读写。

基于网络方式 (NDB)： 基于网络的数据库系统是基于手机4G/5G 的移动通信基础之上的数据库系统，在逻辑上可以把嵌入式设备看作远程服务器的一个客户端。嵌入式网络数据库系统的特点是：无需解析 SQL 语句；支持更多的SQL 操作；客户端小、无需支持可剪裁性；有利于代码重用。

45、 两阶段提交协议2PC:

2PC 事务提交的两个阶段：

表决阶段，目的是形成一个共同的决定。

执行阶段，目的是实现这个协调者的决定。

两条全局提交规则：

只要有一个参与者撤销事务，协调者就必须做出全局撤销决定。

只有所有参与者都同意提交事务，协调者才能做出全局提交决定。

46、分表和分区

分表和分区的共性：1、都针对数据表；2、都使用了分布式存储；3、都提升了查询效率；4 、都降低了数据库的频繁I/O 压力值、

分表和分区的差异：分区逻辑上还是一张表，分表逻辑上已是多张表。

47、 视图的优点

(1) 视图能简化用户的操作

(2) 视图机制可以使用户以不同的方式查询同一数据

(3) 视图对数据库重构提供了一定程度的逻辑独立性

(4) 视图可以对机密的数据提供安全保护

其中物化视图：将视图的内容物理存储起来，其数据随原始表变化，同步更新。

它不是传统意义上虚拟视图，是实体化视图，其本身会存储数据。同时当原始表中的数据更新时，物化视图也会更新。

48、 反规范化

技术手段：增加派生性冗余列、增加冗余列、重新组表、分割表。

反规范化的优点：连接操作少，检索快、统计快；需要查的表减少，检索容易。

反规范化的缺点：数据冗余，需要更大存储空间；插入、更新、删除操作开销更大；数据不一致；可能产生添加、修改、删除异常；更新和插入代码更难写。

49、 备份

(1)冷备份也称为静态备份，是将数据库正常关闭，在停止状态下，将数据库的文件全部备份(复制)下来。

(2)热备份也称为动态备份，是利用备份软件，在数据库正常运行的状态下，将数据库中的数据文件备份出来。

(3)完全备份：备份所有数据。

(4)差量备份：仅备份上一次完全备份之后变化的数据。

(5)增量备份：备份上一次备份之后变化的数据。

(6)日志文件：事务日志是针对数据库改变所做的记录，它可以记录针对数据库的任何操作，并将记录结果保存在独立的文件中。

50、 使用许可

按照被许可使用权的排他性强弱不同，可以将使用许可分为以下三种：

独占使用许可-仅1个授权对象可用，著作权人不可用

排他使用许可-仅1个授权对象和著作权人可用

普通使用许可-多个授权对象和著作权人可用