熟悉网络与分布式系统基本概念：

分布式计算系统是由多个相互连接的计算机组成的一个整体，这些计算机在一组系统软件（分布式操作系统或中间件）环境下，合作执行一个共同的或不同的任务，最少依赖于集中的控制过程、数据和硬件。其中，分布式系统可以用硬件、控制和数据三个维度加以检验：分布式计算系统=分布式硬件+分布式控制+分布式数据。

这个定义中表现出的分布式系统的特征：

①系统是由多个计算机或计算机族（集群）组成，这些计算机或集群可以是异构的

②这些计算机在物理上是独立的，在地理上是分散的，计算机运行其自身的操作系统，称为局部操作系统。局部操作系统也可以是异构的，一个作业或程序可以在一个或多个计算机节点运行，每个节点具有自治能力

③各个计算机的地位是平等的。除了受全局分布式操作系统或中间件协同工作以外，不存在或很少有主从控制或某种集中控制环节

④这些计算机组成一个整体，对用户是透明的，尽力呈现出单一系统视图。用户使用任何资源不必知道这些资源的位置，即感觉不到计算机内部的组成结构、边界或网络的存在。

分布式系统透明性的概念和机制

概念：透明性定义向用户和应用程序隐藏了分布式计算系统部件的差异，系统被认为是一个整体而不是独立部件的集合。

机制：？

分布式系统的五个要求：

①开放性：能否用各种方法进行扩展和重新实现。分布式系统的开放性主要由下列事实决定，即新的共享资源或服务能否被加入并为各客户程序所利用。

②可扩展性：指在资源数量和用户数量增加时仍能有效工作。

③异构性：允许用户访问运行在异构计算机和网络上的服务，用户程序可以运行在这类异构的结构上。异构表现在在：异构网络、异构操作系统、异构计算机硬件、异构程序设计语言、不同开发商的实现

④透明性：透明性定义向用户和应用程序隐藏了分布式计算系统部件的差异，系统被认为是一个整体而不是独立部件的集合。

⑤安全性：基于开放环境下的分布式计算系统应有安全措施，它们的安全至关重要

熟悉中间件技术的类型

中间件的类型：？

中间件为应用程序提供一系列服务，不同的中间件提供的服务可能是不同的：

①命名服务

②作业调度

③资源管理

④数据的持久性

⑤分布式事务

⑥分布式文档系统

⑦安全服务

⑧高级通信服务

实体：实体在一个计算机系统中指范围广泛的事务，包括计算机主机、外围设备、进程、数据、文件、数据库、服务、服务器和用户等。在计算机中称呼实体有很多中种办法，比如：名字、地址、标识符

名字：实体的名字是一个用户可读的、便于记忆的字符串

地址、：如果要对实体进行操作，就要访问该实体，实体就需要一个访问点。访问点在分布式系统种是一个特殊实体，它的名字称为地址。

了解从URL查找Web网页的全过程（画图）

标识符：除了实体名和地址外，实体还可以用内部标识符来标识

属性：一个实体有若干属性，用属性的<类型-值>对表示

名字服务的形式：①名字服务：根据实体的名字查找它的属性（地址）②目录服务：既可以根据实体的名字查找实体的属性，当不知道实体名时也可以更具实体的一个或多个属性及其值查找并得到一个匹配这些属性的实体列表③合约服务：是一种增强的目录服务，通过技术规范来定位一个命名实体，也叫做绿叶服务。如Web服务

名字空间：？

名字图：DAG图（有向无环图）——叶子节点、目录节点、根节点

挂接与挂载（被挂接的一定是一个目录节点，会找挂接点和挂载点）

名字服务器：数据库（实现实体名和其地址的绑定，还保存了实体的其他信息）+名字解析软件

名字空间划分和多副本：

名字服务器的组成：

域名系统：

熟悉名字解析中，递归名称解析和迭代名称解析的区别与工作原理，可以给出解析实体名的过程

可以结合具体的例子给出工作过程

迭代名称解析（画图）：客户名字解析程序将被访问实体的整个名字传送给根名字服务器，根名字服务器尽力解析实体名，将其解析出的名字服务器的地址和实体名的剩余部分返回给客户端名字解析程序。客户端名字解析程序收到响应后在向由此响应中指明的名字服务器发送实体名中的剩余部分，该名字服务器收到请求后同样会尽力解析这一剩余部分，并将其解析出的下一个名字服务器的地址和实体名中的剩余部分返回给客户端名字解析程序，以此类推。最终实体名会被全部解析，客户端名字解析程序会得知完整的文件服务器地址，并向其发出请求，文件服务器为客户返回请求的文件。

递归名称解析（画图）：与递归解析不同，每次名字服务器解析得到的下一个名字服务器地址不是返回给客户而是根据得到的地址直接找寻下一个名字服务器，并将实体名的剩余部分传送给下一个名字服务器进行解析。最后一个名字服务器解析得到的实体服务器最后一节的地址会返回给上一个名字服务器。以此类推，根名字服务器将实体的地址返回给客户。

比较： ①递归名字解析要求每台名字服务器都具有较高的性能，除了能进行名字解析和与客户通信以外还要与其他名字服务器通信，并缓存自己和其他名字服务器的解析结果；而迭代名字解析只要求名字服务器可以进行名字解析和通信

②递归名字解析中，各名字服务器的缓存结果的使用更为有效，例如：。。。；而迭代名字解析的过程中，客户调用名字解析程序得到的解析结果一般是缓存在客户的地址空间，其他客户无法使用。

③。。。

熟悉名字服务的目录服务基本概念

名字：用来共享资源、标识实体和指向实体的位置。 典型的名字服务：DNS、目录服务X.500、活动目录域服务、CORBA的命名服务

目录服务：目录服务是一种特殊类型的名字服务。除了根据实体名查找它的属性（如IP地址）外，用户可以基于属性描述来查找实体，而不用完整的实体名。

目录服务模型：没有看懂

目录服务协议：目录访问协议DAP、目录系统协议DSP、目录信息镜像协议DISP、目录操作绑定管理协议DOP

LDAP模型的概念与基本原理

轻量目录访问协议（LDAP）是用户用来访问目录服务的一个协议。其最初的目标是向客户提供目录服务时避免DAP的大量开销以代替DAP访问X.500目录。同时也支持其他目录服务。

LDAP模型图（画图）中，客户使用LDAP对LDAP服务器发出服务请求。LDAP服务器对目录执行指定的操作活动，然后向客户报告结果。LDAP服务器自行处理目录服务器（DSA）送来的转交，只向客户返回正确的查询结果或错误消息，不再返回转交。LDAP模型的实现有两种方法：①LDAP服务器是一个独立的实体，用目录服务器提供的接口请求目录服务器执行查询操作 ②LDAP服务器就是目录服务器的一个组成部分

活动目录域服务ADDS：没看

分布式进程概念：进程是操作系统中独立存在的实体，它可以拥有自己独立的资源，没有经过进程本身允许的情况下，其他进程不能访问这些资源（对比线程）

进程与程序的区别：程序是一个静态的指令集合，而进程是一个正在系统中活动的指令集合（即执行中的程序），若干个进程可以在单处理机状态上并发执行

进程状态：

线程：是包含在进程中的一种实体，有自己的运行线索，可以完成一定的任务，可与其他线程共享进程中的共享变量以及部分环境，相互之间协同来完成进程所要完成的任务。线程比进程有更高的性能，同一个进程中的线程都有共性；多个线程将共享同一个进程虚拟空间。

熟悉进程迁移的原理与基本概念

概念：（动态）进程迁移是将一个正在运行的进程挂起，它的状态从源处理机节点转移到目标处理机节点，并在目标处理机上恢复该进程运行。相对于静态放置（如进程远程执行），进程迁移具有灵活且应用广泛的优点，例如支持动态负载平衡、系统容错、高效使用本地资源等诸多系统功能。但缺点是运行开销相对比较大。

原理（概念模型）：进程迁移被分为两部分的工作，一是在源处理机采集迁移进程的进程状态并转移到目的处理机，并用这些进程状态在目的处理机重建迁移进程，使之从断点处继续运行；二是通知与迁移进程有通信关系的其他进程，重建它们与迁移进程正确的通信连接。

进程迁移步骤（机制）：①迁移协商②创建恢复进程③中断迁移进程的运行④在源处理机上收集迁移进程状态⑤将迁移进程状态传输到目标处理机⑥恢复被迁移状态⑦通知被迁移进程的新位置⑧迁移进程在目标节点恢复运行⑨操作转发，利用转发机制保证进程可以在远程处理机透明执行

进程迁移策略：没看

分布式系统通信：是由一个通信层实现的，它建立在节点操作系统和网络传输层协议之上，提供一组通信原语供应用程序调用。通信层介乎应用程序和操作系统程序之间，实现各种通信协议，实际上是一种中间件。

消息：

可靠多播通信中，解决反馈拥塞的方法

没看懂

RPC概念与基本工作原理

远程过程调用，思想是尽量使远程过程调用具有与本地过程调用相同的形式。分布式系统希望RPC是透明的，即客户不应该知道被调用的过程是在另一台机器上执行，反之亦然。只不过调用时是从例程的另一个版本——客户桩中获取。服务器端服务器也为远程客户提供了一个例程——服务器桩。实际上，客户桩是服务器在客户机上的代理；服务器桩是客户在服务器机上的代理。

原理（要理解并会描述画图）：对于客户，客户执行一个远程调用会调用客户桩。客户桩的编码模块从堆栈中将调用参数读出到一个缓冲区，形成调用参数适用于网络上传输的扁平形式。然后将它打包（Pack）成“请求信息”，通过发送原语“send”，由操作系统网络接口传送“请求信息”。然后发送端执行接收原语“receive”后被阻塞，等待“应答消息”到来。对于服务器，服务器桩先执行接收原语receive后被阻塞，等待客户请求。“请求消息”到达服务器端由操作系统根据接收原语指定的端口，将他送到服务器桩。服务器桩解包（Unpack）“请求消息”，解码调用参数，恢复成堆栈形式，然后调用服务器的A例程，服务进程从堆栈读取调用参数，实现对服务的调用。服务器执行完A例程之后，准备将结果返回给客户，将调用服务器桩将结果编码成适于网络上传输的格式。然后Pack成“应答消息”，通过发送原语send由操作系统和网络接口传送“应答消息”。“应答消息”到达客户端，由操作系统根据接收原语指定的端口，将它送到客户桩。客户桩Unpack“应答消息”，解码调用结果，并将结果返回给用户

远程对象引用ROR：是访问对象的一种机制，有点像但不是对一个对象的标识或者说命名。

RMI的基本概念和工作原理，RMI与RPC的关系与区别

语义?

远程方法调用RMI（常用的是CORBA和Java/RMI），将用户绑定到对象之后，就可以通过代理来调用对象的方法。其中分为静态和动态两类。

静态（画图）：分别在用户端和服务器端生成代理Proxy和骨架Skeleton，分别对应客户桩和服务器桩。由接口定义语言生成客户端的代理和服务器端骨架。客户端代理由用户应用进程调用，对调用结果解码，将结果返回给客户进程；服务器端骨架接收和解包请求消息，解码调用参数，调用对象方法。对象返回调用结果给骨架，骨架对调用结果进行编码，并打包成应答消息，将应答消息传送给客户端代理。

动态：

关系与区别：？

网络时间协议NTP

Lamport时间戳：定义了一个关系称作”先发生“，只能表明事件之间的时序关系，无法表述其因果关系。

向量时间戳的概念与原理，及其表示事件的方法

向量时间戳是通过让每个进程维护一个向量V来实现的。其中V有以下三个性质：

①向量时钟是一个数组，其元素的个数是系统的进程数，元素是升序的整数。元素在数组中是按进程号排序。系统的每个进程维持一个向量时钟。

②Vi[i]是到目前为止进程Pi发生的事件数量，它是在进程Pi中新事件发生时递增Vi[i]来维护的

③如果Vi[j] = k， 进程Pi便知道进程Pj中已经发生了k个事件。这是通过所发送消息携带向量来维护的。例如进程Pi发送消息时，将它当前的向量时钟作为时间戳与消息一同发送

图5.7理解会画图

掌握欺负算法基本原理、概念与工作流程，熟悉选举触发的条件、选举条件和触发场景

欺负算法（画图）：涉及了三种消息——选举消息E、应答消息OK和协调者消息C 当一个进程P发现协调者不再相应请求时，它就启动一次选举。进程P按照如下的过程主持一次选举：

①进程P向所有编号比它高的进程发送一个选举消息E

②如果没有响应进程，那么P就是老大，获胜成为新的协调者

③如果有进程响应，即有进程的编号比P大，响应进程接过选举主持工作，进程P的任务结束

任何时候，一个进程只能从编号比他小的进程里接收到选举消息E。它接收到选举消息后发回一个应答消息OK，然后接替主持选举，发出新的选举消息E。最后除了一个编号最大的进程外，其他进程都放弃选举的主持工作，这个编号最大的进程便成为新的协调者。凡是编号最大的进程，它总是会当选协调者，所以称作欺负算法。

另外，算法中还需要一个可靠的失效检测器。假定Ttran是最大的消息传输延时，Tprocess是处理消息的最大延时。如果在T = 2Ttran + Tprocess时间内没有应答消息到达，本地失效检测器认为接收选举消息的进程已经失效，如果在T时间内没有收到任何消息OK，那么该进程便认为自己成为了新的协调者，并发出协调者消息C，通知进程组内所有幸存进程

协调者的作用是?

掌握分布式互斥算法中，Ricart和Agarawala算法的原理与临界区死锁问题

基于Lamport时间戳，只要2(N-1)个消息，其中N是竞争资源的进程数目。

此算法最重要的特点是其对称性、完全的分布式控制和对通信链路相对速度的不敏感性

原理：

①进程Pi发送资源请求消息Request（Ti： Pi）到其他竞争该资源的进程，并将消息至于他自己的请求队列中。Ti是它的时间戳

②当进程Pj接收到资源请求消息Request（Ti： Pi），按时间戳顺序将消息置于它的消息队列中，并做W1：如果没有资源请求或请求的有效顺序晚于收到的请求消息的优先顺序，便向发请求消息的进程回送一个应答消息Reply（Tj： Pj），否则就推迟返回应答消息 W2：进程从临界区退出，向有关请求资源的进程补发一个应答消息

③请求进程从其他所有竞争进程得到应答消息Reply（Tj： Pj），便可进入临界区

临界区死锁问题：？

理解一致性模型中，以数据为中心和以用户为中心的模型原理与区别

一致性模型是数据存储和访问数据存储的进程之间的一种契约

以数据为中心的一致性，就是多个进程并发访问同一个共享数据存储时，保持共享数据存储的一致性。（从强到弱：严格一致性、顺序一致性、因果一致性、FIFO一致性、弱一致性、释放一致性、入口一致性）

以客户为中心的一致性，其是为了解决一个客户访问数据存储的不同副本时，如何保持数据存储副本间的一致性（几个方法：单调读、单调写、写后读、读后写）

单调读：一个进程读数据项x的值，该进程的任何后续对x的读操作总是返回前一次读同样的值或更加新的值。

单调写：

掌握法定多数表决的复制写协议工作原理，可以给出正确的读、写集团

复制写协议：写操作同时在多个副本上进行。其包括主动多副本协议和基于法定数量的协议。前者在所有副本上进行，后者在一个法定数量（多数服务器也即超过一半数量的服务器）的副本上进行。

基于法定数量的协议：一种基于法定多数表决的复制写协议是由Thomas提出的。其基本思想是要求客户在读或写一个多副本共享数据项之前，向多个副本服务器提出请求并得到它们的同意。

具体原理：一个分布式文件系统被复制在N个服务器上。要更新一个文件，客户至少必须取得N/2+1个服务器（多数服务器）的同意。一旦它们同意更新，该文件将被修改，并用一个新的版本号相关联。在读一个多副本文件时，客户也必须至少联系N/2+1个服务器，请求它们返回该文件的版本号。如果返回的所有版本号一致，该版本必定是最新的版本，因为此时剩余的服务器数量不足总数的一半，不可能执行写操作。

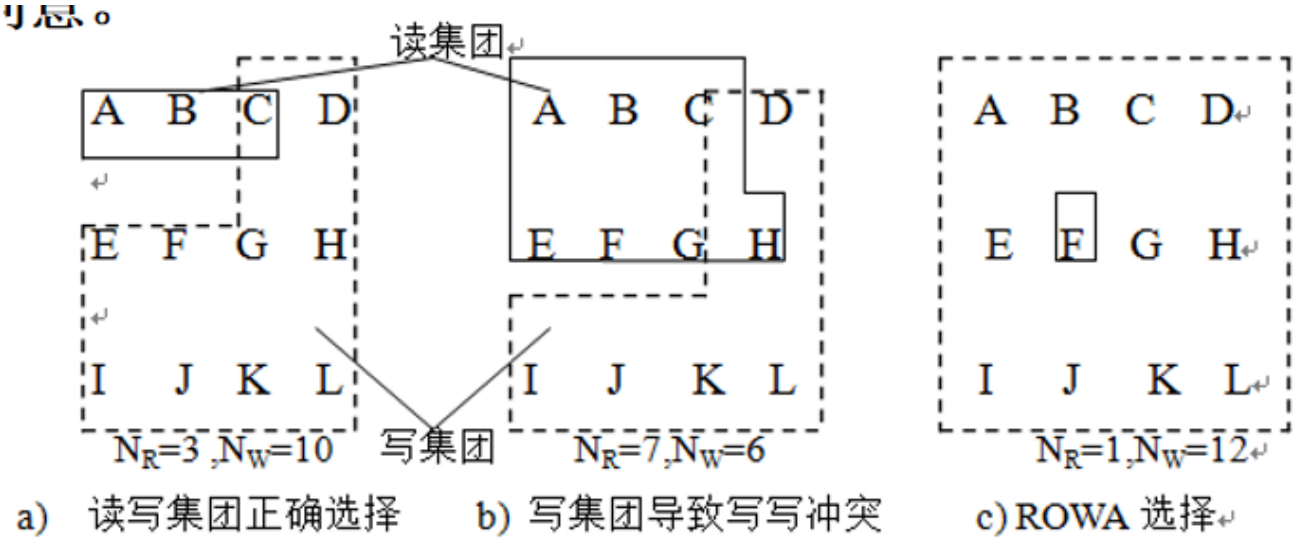
Gifford的改进：一个客户要读取一个具有N个副本的文件，他必须将Nr个服务器组成一个“读集团”；要修改一个文件，必须将Nw个服务器组成一个“写集团”。其中，

①Nr + Nw > N 用于防止读-写冲突

②Nw > N/2 用于防止写-写冲突

只有在适当个数的服务器同意参与文件的读写操作后，客户才能执行对文件的读或写

具体如何给出正确的读、写集团参照课本P215图8.21



NFS体系结构与概念，清楚解决高效缓存一致性的原理与方法

网络文件系统NFS（使用RPC）

体系结构：图9.3 会画

NFS底层模型是远程文件访问模型，它为客户对远程服务器所管理的文件系统提供透明服务

高速缓存一致性的原理与方法：（以下三个方面的原理 ）

高速缓存一致性的检测策略：

①静态方法：此方法假定编译器在客户软件运行之前执行必要的分析，确定哪些数据可能因为被缓存而导致不一致性。编译器所做的仅仅是输入一些避免不一致的指令。

②动态方法：此方法是在运行时检测不一致性，是分布式系统常用的方法。

高速缓存一致性的实施策略——为保持高速缓存与服务器各副本保持一致——两种方法：

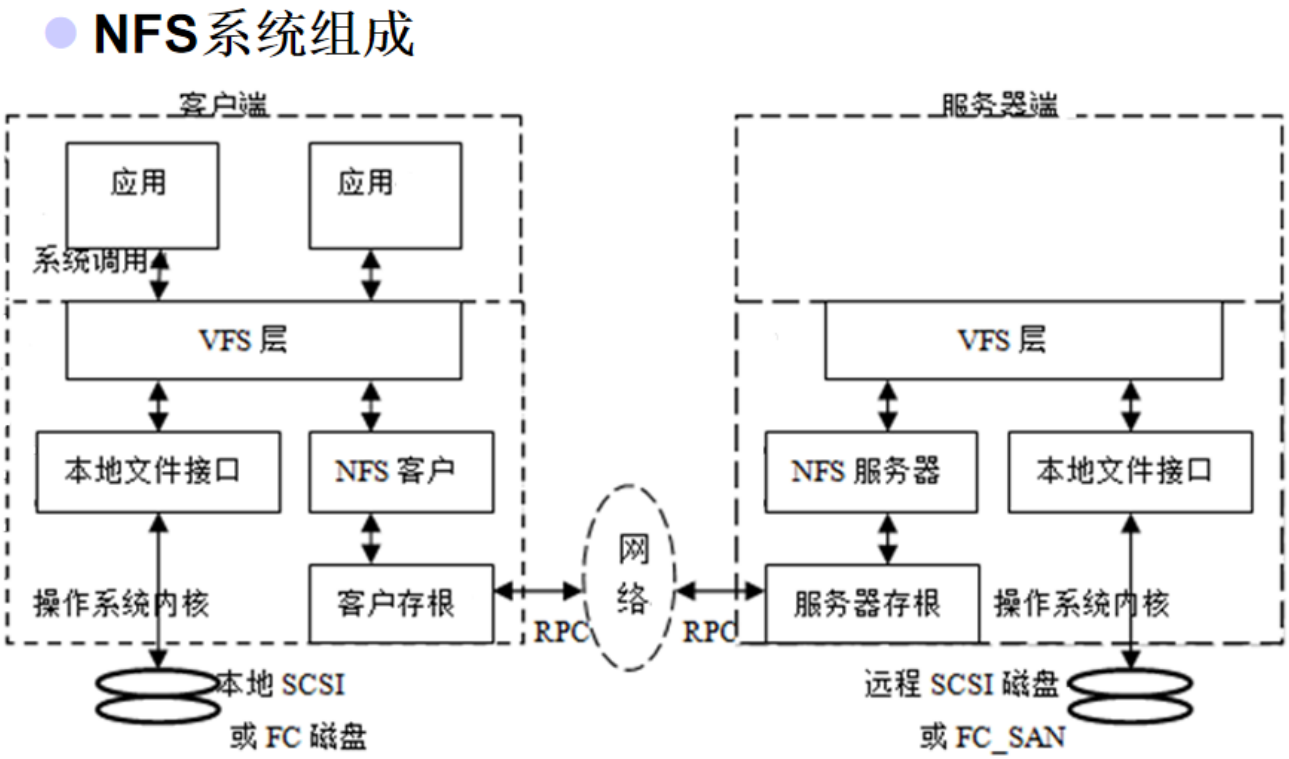
①第一种方法是当服务器一个数据项被修改后，服务器向所有高速缓存发送一个无效消息，通知高速缓存该数据项已经过时

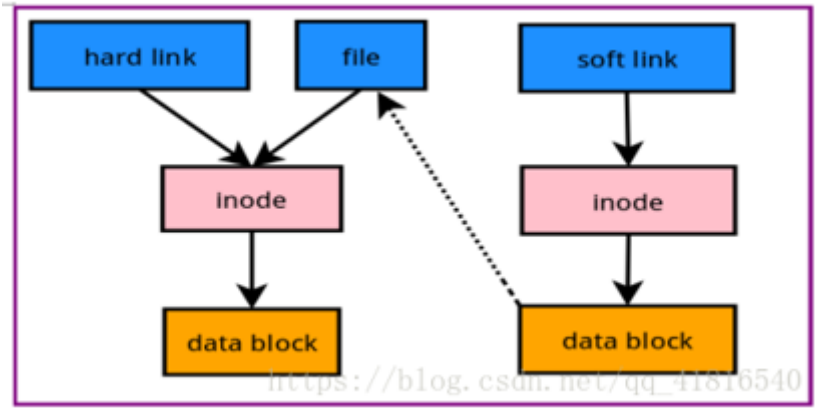
②第二种方法是直接传播数据更新。

最后，还要考虑用户进程修改高速缓存更新数据时如何使服务器及时更新

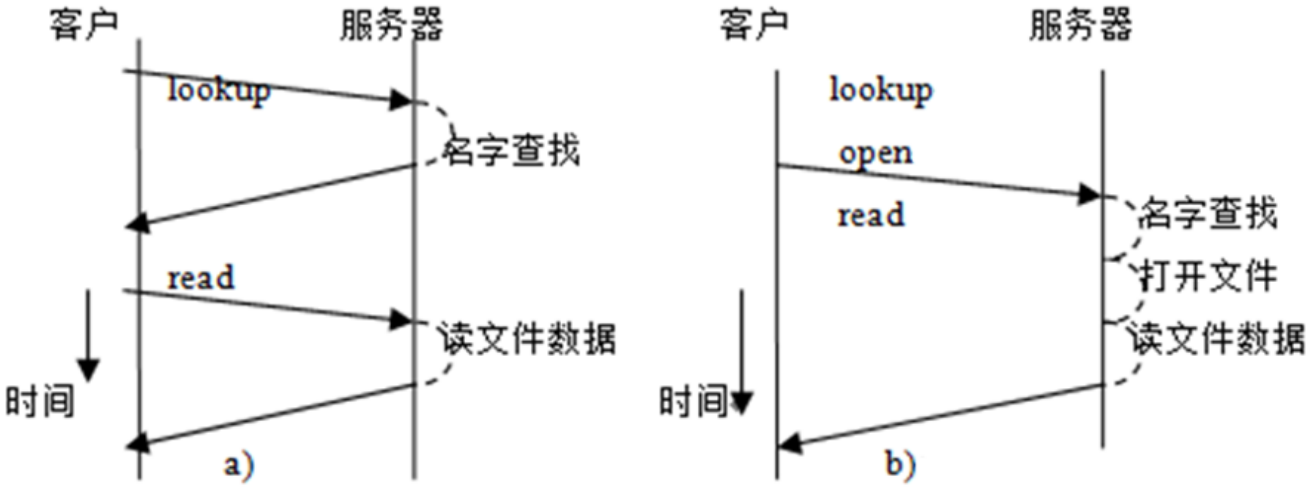
①直写式高速缓存：是允许客户在修改被缓存数据时，将更新数据发给副本服务器。

②另外还有回写式高速缓存

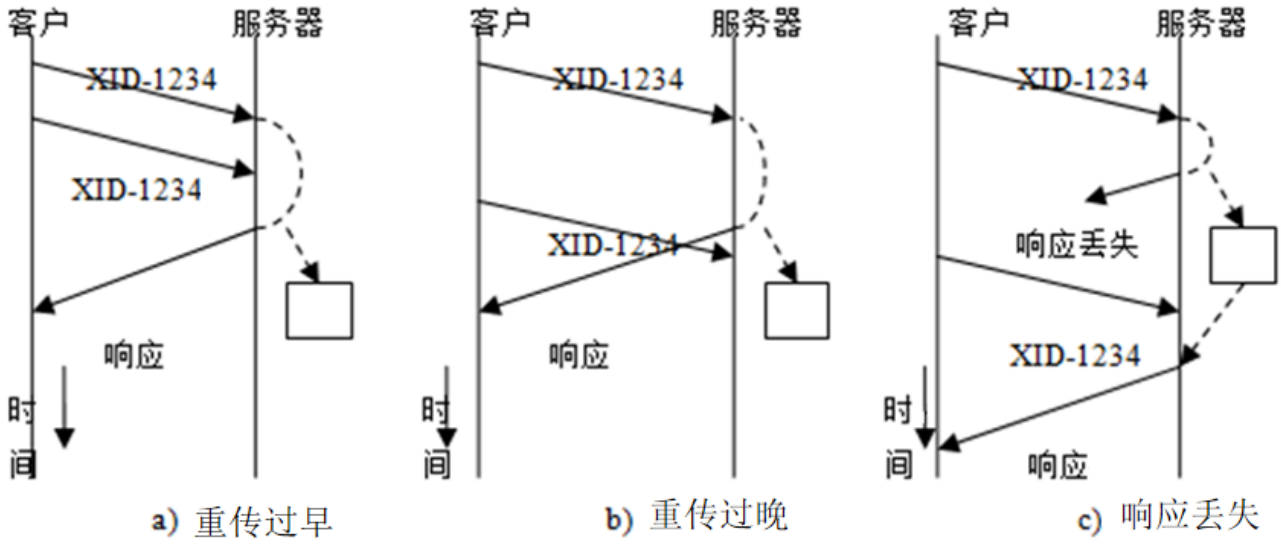




- 常规调用vs.复合调用



- RPC重传： RPC头部携带事务标识符（XID）；服务器处理请求后，将响应放入缓存

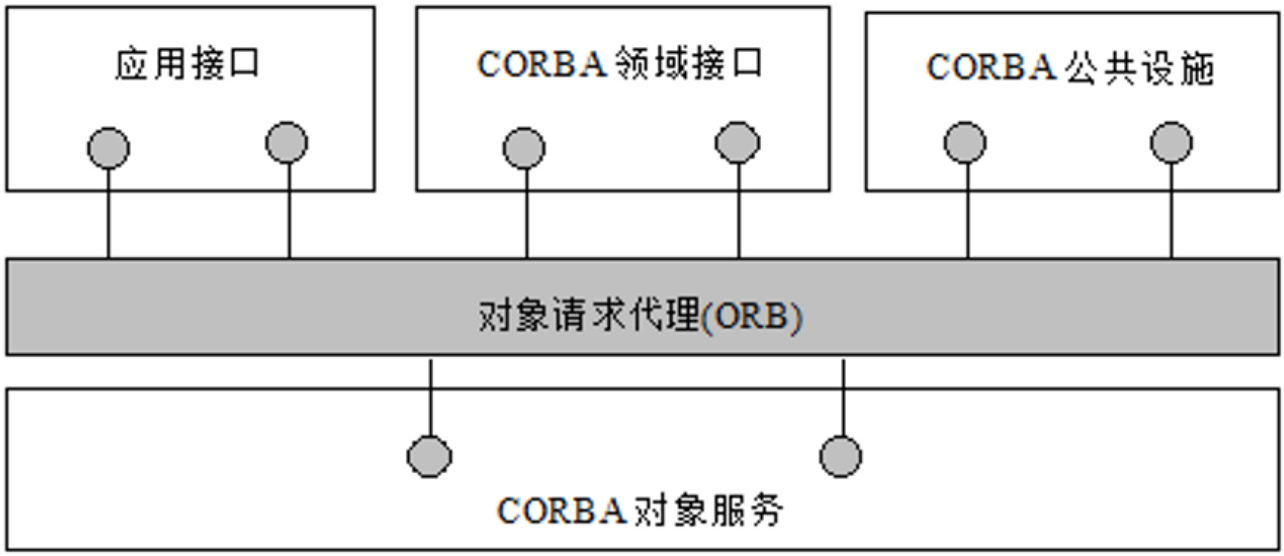


CORBA（远程对象方法调用的一种）的体系结构与概念，掌握接口的类型与作用

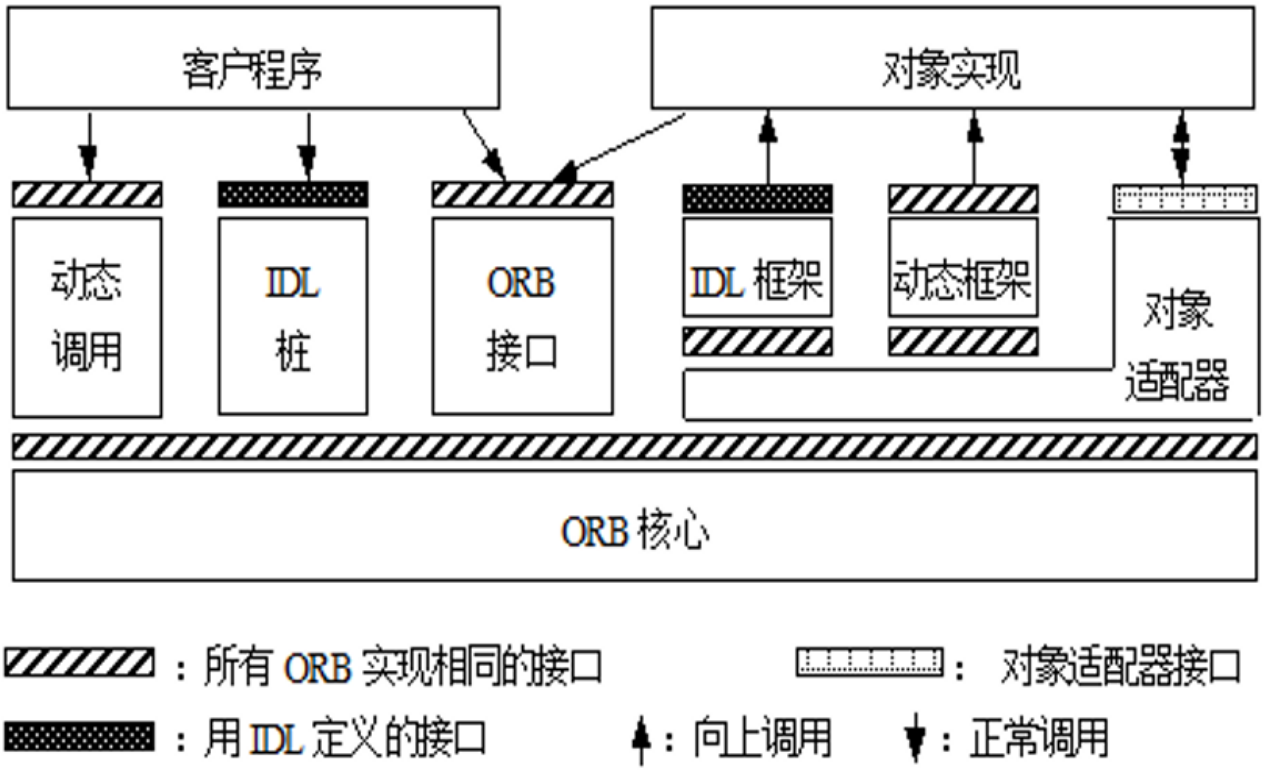
分布式对象是一些独立代码的封装体（对象、组件），它向外提供了一个包括一组属性和方法的接口，远程客户程序通过远程方法调用来访问它。

公共对象请求代理结构CORBA，目的是允许分布式对象在异构环境下互操作，对象可以用不同的编程语言实现，部署在不同的平台上。

CORBA是OMA的一部分，作为规范分布式对象间互操作的标准



CORBA组成



WebService：主要目的是跨平台的可互操作性

①是由企业发布的完成其特定商务需求的在线应用服务

②是一种新的Web应用程序分支，是自包含、自描述、模块化的应用

③是一个应用组件，它逻辑性地为其他应用程序提供数据与服务

XML

- 扩展标记语言
- XML文档组成
  - 元素和相关属性
  - 语法规则：必须有根元素；正确嵌套；元素必须有结束符
- XML名字空间
  - 名字冲突解决方法：限定名字
  - 默认名字空间
- XML Schema模式语言

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE, SYSTEM "Bookstore">
<bookstore xmlns="http://www.w3c.org/bookstore">
        <book category="CHILDREN">
                <title>Harry Potter</title>
                <author>J K Rowling</author>
                <year>2005</year>
                <price>29.99</price>
        </book>
        …
</bookstore>
```

XML

- XML Schema模式语言（.xsd）
- 用于描述和定义数据
- 包括XML文档的元素及其子元素的个数、次序和是否为空、元素中间的属性和属性的数据类型等
- 引用XML模式

WebService中WSDL的概念，XML基本原理与概念

WebService中UDDI的基本概念，理解商业注册中心类型

UDDI是Web服务提供者和消费者之间的接洽点

SOAP的基本概念和工作原理，掌握SOAP XML Schema基本代码的编写规范

云计算的概念和其关键技术和服务，SaaS、PaaS、laaS

能够根据给定的需求设计一个分布式系统，并阐述所涉及到的相关技术