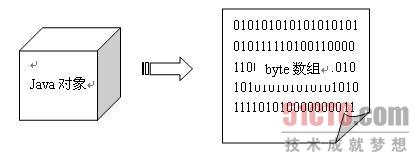
1. 我们不禁要问，什么是"服务集群"？什么是"企业级开发"？   
既然说了EJB 是为了"服务集群"和"企业级开发"，那么，总得说说什么是所谓的"服务  
集群"和"企业级开发"吧！  
这个问题其实挺关键的，因为J2EE 中并没有说明白，也没有具体的指标或者事例告诉  
广大程序员什么时候用EJB 什么时候不用。于是大家都产生一些联想，认为EJB"分布式运  
算"指得是"负载均衡"提高系统的运行效率。然而，估计很多人都搞错了，这个"服务群集"  
和"分布式运算"并没有根本解决运行负载的问题，尤其是针对数据库的应用系统。  
为什么？  
我们先把EJB 打回原形给大家来慢慢分析。

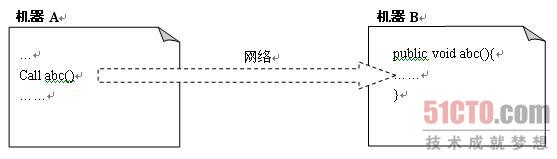
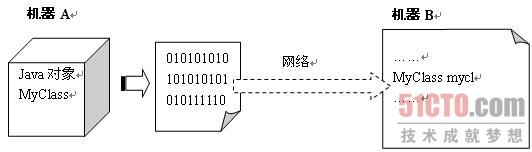
2. 把EJB 掰开了揉碎了   
我们把EJB 的概念好好的分析一下，看看能发现些什么蛛丝马迹。

3.1 EJB 概念的剖析  
我们先看一下，EJB 的官方解释：  
商务软件的核心部分是它的业务逻辑。业务逻辑抽象了整个商务过程的流程，并使用计  
算机语言将他们实现。  
……  
J2EE 对于这个问题的处理方法是将业务逻辑从客户端软件中抽取出来，封装在一个组  
件中。这个组件运行在一个独立的服务器上，客户端软件通过网络调用组件提供的服务以实  
现业务逻辑，而客户端软件的功能单纯到只负责发送调用请求和显示处理结果。在J2EE 中，  
这个运行在一个独立的服务器上，并封装了业务逻辑的组件就是EJB（Enterprise Java  
Bean）组件。  
这其中我们主要关注这么几点，我们来逐条剖析：  
剖析1：所谓："业务逻辑"   
我们注意到在EJB 的概念中主要提到的就是"业务逻辑"的封装，而这个业务逻辑到底是  
什么？说的那么悬乎，其实这个所谓的"业务逻辑"我们完全可以理解成执行特定任务的"类  
"。  
剖析2：所谓："将业务逻辑从客户端软件中抽取出来，封装在组件中……运行在一个服  
务器上"  
既然我们知道了"业务逻辑"的概念就是执行特定任务的"类"，那么，什么叫"从客户端  
软件中抽取出来"？其实，这个就是把原来放到客户端的"类"，拿出来不放到客户端了，放  
到一个组件中，并将这个组件放到一个服务器上去运行。  
3.2 把EJB 这个概念变成大白话   
变成大白话就是，"把你编写的软件中那些需要执行制定的任务的类，不放到客户端软  
件上了，而是给他打成包放到一个服务器上了"。  
3.3 发现问题了   
不管是用"八股文"说，还是用大白话说这个EJB 概念都提到了一个词--"客户端软件"。  
"客户端软件"？难道EJB 的概念中说的是C/S 软件？  
是的，没错！  
EJB 就是将那些"类"放到一个服务器上，用C/S 形式的软件客户端对服务器上的"类"进  
行调用。  
快崩溃了吧！  
EJB 和JSP 有什么关系？EJB 和JSP 有关系，但是关系还真不怎么大，至多是在JSP 的  
服务器端调用远端服务上的EJB 类，仅此而已。

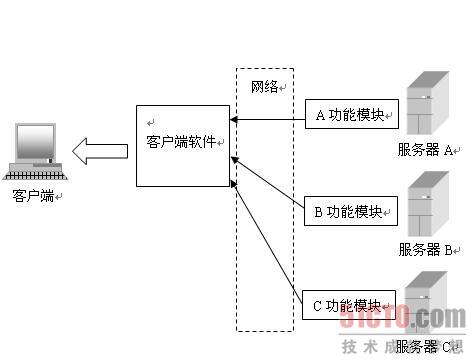
4 .1 EJB 的最底层究竟是什么   
我们揭开了EJB"八股"概念的真谛，那么，再来分析EJB 的底层实现技术，通过底层实  
现技术来分析EJB 的工作方式。

4.2 EJB 的实现技术  
EJB 是运行在独立服务器上的组件，客户端是通过网络对EJB 对象进行调用的。在Java  
中，能够实现远程对象调用的技术是RMI，而EJB 技术基础正是RMI。通过RMI 技术，J2EE  
将EJB 组件创建为远程对象，客户端就可以通过网络调用EJB 对象了。  
4.3 看看RMI 是什么东东   
在说RMI 之前，需要理解两个名词：  
对象的序列化  
分布式计算与RPC  
名词1：对象的序列化   
对象的序列化概念：对象的序列化过程就是将对象状态转换成字节流和从字节流恢复对  
象。将对象状态转换成字节流之后，可以用java.io 包中的各种字节流类将其保存到文件中，  
或者通过网络连接将对象数据发送到另一个主机。  
上面的说法有点"八股"，我们不妨再用白话解释一下：对象的序列化就是将你程序中实  
例化的某个类的对象，比如，你自定一个类MyClass，或者任何一个类的对象，将它转换成  
字节数组，也就是说可以放到一个byte 数组中，这时候，你既然已经把一个对象放到了byte  
数组中，那么你当然就可以随便处置了它了，用得最多的就是把他发送到网络上远程的计算  
机上了。如图2 11所示。

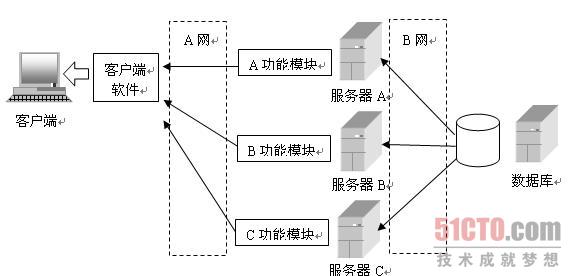
  
名词2：分布式计算与RPC   
RPC 并不是一个纯粹的Java 概念，因为在Java 诞生之前就已经有了RPC 的这个概念，RPC  
是"Remote Procedure Call"的缩写，也就是"远程过程调用"。在Java 之前的大多数编程语  
言，如，Fortran、C、COBOL 等等，都是过程性的语言，而不是面向对象的。所以，这些编  
程语言很自然地用过程表示工作，如，函数或子程序，让其在网络上另一台机器上执行。说  
白了，就是本地计算机调用远程计算机上的一个函数。  
如图2 12所示。

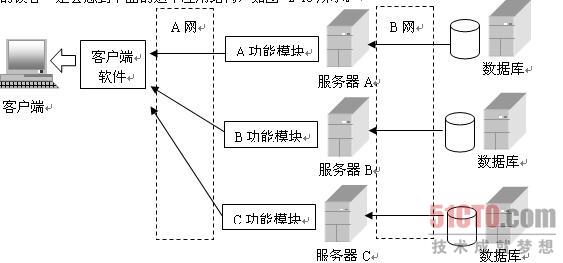
  
名词3：二者结合就是RMI   
RMI 英文全称是"Remote Method Invocation"，它的中文名称是"远程方法调用"，它就  
是利用Java 对象序列化的机制实现分布式计算，实现远程类对象的实例化以及调用的方法。  
说的更清楚些，就是利用对象序列化来实现远程调用，也就是上面两个概念的结合体，利用  
这个方法来调用远程的类的时候，就不需要编写Socket 程序了，也不需要把对象进行序列  
化操作，直接调用就行了非常方便。  
远程方法调用是一种计算机之间对象互相调用对方函数，启动对方进程的一种机制，使用这  
种机制，某一台计算机上的对象在调用另外一台计算机上的方法时，使用的程序语法规则和  
在本地机上对象间的方法调用的语法规则一样。  
如图2 13所示。  


4.4 优点  
这种机制给分布计算的系统设计、编程都带来了极大的方便。只要按照RMI 规则设计程  
序，可以不必再过问在RMI 之下的网络细节了，如：TCP 和Socket 等等。任意两台计算机  
之间的通讯完全由RMI 负责。调用远程计算机上的对象就像本地对象一样方便。  
RMI 可将完整的对象作为参数和返回值进行传递，而不仅仅是预定义的数据类型。也就  
是说，可以将类似Java 哈西表这样的复杂类型作为一个参数进行传递。  
4.5 缺点   
如果是较为简单的方法调用，其执行效率也许会比本地执行慢很多，即使和远程Socket  
机制的简单数据返回的应用相比，也会慢一些，原因是，其在网络间需要传递的信息不仅仅  
包含该函数的返回值信息，还会包含该对象序列化后的字节内容。  
4.6 EJB 是以RMI 为基础的  
通过RMI 技术，J2EE 将EJB 组件创建为远程对象，EJB 虽然用了RMI 技术，但是却只需  
要定义远程接口而无需生成他们的实现类，这样就将RMI 技术中的一些细节问题屏蔽了。  
但不管怎么说，EJB 的基础仍然是RMI，所以，如果你想了解EJB 的原理，只要把RMI  
的原理搞清楚就行了。你也就弄清楚了什么时候用EJB 什么时候不需要用EJB 了。

5. EJB 中所谓的"服务群集"   
既然已经知道了，RMI 是将各种任务与功能的类放到不同的服务器上，然后通过各个服  
务器间建立的调用规则实现分布式的运算，也就明白EJB 所谓的"服务群集"的概念。  
就是将原来在一个计算机上运算的几个类，分别放到其他计算机上去运行，以便分担运  
行这几个类所需要占用的CPU 和内存资源。同时，也可以将不同的软件功能模块放到不同的  
服务器上，当需要修改某些功能的时候直接修改这些服务器上的类就行了，修改以后所有客  
户端的软件都被修改了。如图2 14所示。  


6. 这种部署难道是无懈可击   
图2 14所示的这个"服务群集"看似"无懈可击"，其实是它这个图没有画完整，我们来  
把这个图画完整，再来看看有什么问题没有。  
6.1 瓶颈在数据库端   
仔细观察之后，发现这种配置是有瓶颈的，如图2 15所示。

  
我们看看图2 15的结构图，现在如果想实现各个服务器针对同一个数据库的查询，那  
么，不管你部署多少个功能服务器，都需要针对一个数据库服务器进行查询操作。也就是说，  
不管你的"计算"有多么"分布"也同样需要从一台服务器中取得数据。虽然，看起来将各个功  
能模块分布在不同的服务器上从而分担了各个主计算机的CPU 资源，然而，真正的瓶颈并不  
在这里，而是，数据库服务器那里。数据库服务器都会非常忙的应付各个服务器的查询及操  
作请求。  
因此，通过这个结构图使我们了解到了EJB 根本不能完全解决负载的问题，因为，瓶颈  
并不在功能模块的所在位置，而是在数据库服务器这里。  
6.2 假如分开数据库，数据共享怎么办   
有的读者一定会想到下面的这个应用结构，如图2 16所示。

  
就是把每一个功能服务器后面都部署一个数据库，这样不就解决了上节所说的问题了  
吗？是的解决了数据库查询负载的问题，然而又出现了新的问题，就是"数据共享"的问题就  
又不容易解决了。  
6.3 网络面临较大压力，让你的应用慢如老牛  
我们再向前翻看看如图2 15所示的这种架构中存在两个网络，一个是"A 网"一个是"B  
网"，这两个网络是不同的。"B 网"往往是局域网，一般带宽是10M/100M，速度较快，因此  
到还好说，然而，"A 网"往往是互联网或者是利用电信网络互联VPN 网或称广域网。"A 网"  
的特点是带宽一般较窄，如ADSL 的网络仅仅有512K-2M 的带宽，由于广域网互联的成本较  
高，所以一般不会有较高的带宽。  
而在这个网络上恰恰跑的是功能模块和客户端软件之间交换的数据，而这部分数据恰恰  
优势非常占用带宽的。  
因此，这个应用架构其运行速度可以想见是多么的慢了。说句不夸张的话，有点想老牛  
拉破车一样的慢。  
一个如老牛的系统：  
目前在中国互联网做运营商网络管理系统的一个大公司，它的一个早期的网管软件就是  
采用了这种架构来做的C/S 结构的应用系统。  
有一次，我作为评估者来对其应用系统进行评估，将其部署到一个非运营商大型的网络  
中的时候，便出现了我们上述描述的情况，速度已经到了难以忍受的地步，打开一个流量图，  
有时候需要用15分钟的时间才能呈现完整。然而，该系统在开发阶段并没有发现这个问题，  
为什么呢？因为，他们没有考虑到应用的实际用户连接网络的复杂性，从而给该公司造成较  
大损失，以至于，这个开发架构被最终遗弃。

7. EJB 活学活用，J2EE 不是必须使用EJB   
通过上面小节的讲解似乎好像EJB 和开发Web 应用的B/S 结构的系统关系并不大，其实  
倒也不然。我们如果把"客户端程序"理解成某一台服务器，这样也是可以被应用的，而且，  
如果是服务器互相之间做EJB 的调用的话，也就不存在广域网带宽限制的问题了。  
但是，如下情况尽量就不要使用EJB 了：  
1、较为简单的纯Web 应用开发，不需要用EJB。  
2、需要与其他服务程序配合使用的应用，但调用或返回的自定义的网络协议可以解决  
的应用程序,不需要使用EJB。  
3、较多人并发访问的C/S 结构的应用程序，尽量不要使用EJB。

**总结：**

a.EJB实现原理： 就是把原来放到客户端实现的代码放到服务器端，并依靠RMI进行通信。

b.RMI实现原理 ：就是通过Java对象可序列化机制实现分布计算。

c.服务器集群： 就是通过RMI的通信，连接不同功能模块的服务器，以实现一个完整的功能。