# EJB概念剖析

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服务器上，并封装了业务逻辑的组件就是EJB组件。这其中我们主要关注这么几点，我们逐条剖析：

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？说的那么玄乎，其实这个所谓的业务逻辑，我们完全可以理解为执行特定任务的“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。

# 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特定任务的类，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”。

# 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。

# 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。

# 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。

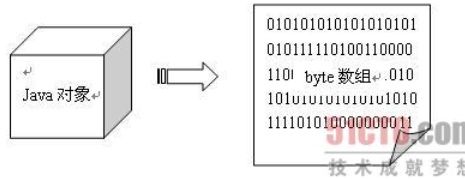
# RMI

在说RMI之前，需要理解两个名词：

## 对象的序列化

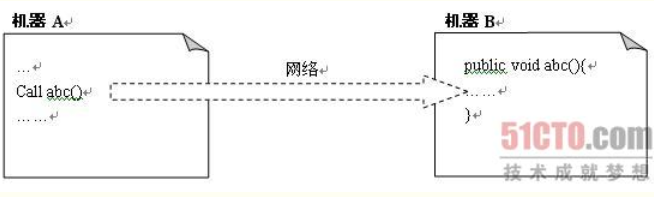
对象序列化的概念：对象的序列化过程就是将对象状态转换成字节流和从字节流恢复对象。将对象状态转换成字节流之后，可以用java.io包中的各种字节流类将其保存到文件中，或者通过网络连接将对象数据发送到另一台主机。

上面的说法有点“八股”，我们不妨用白话解释一下：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，将他转换成字节数组，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byte数组中，那么你当然就可以随便处置它了，用的最多的就是把它发送到网络上的远程计算机上了。如图所示：



## 分布式计算与RPC

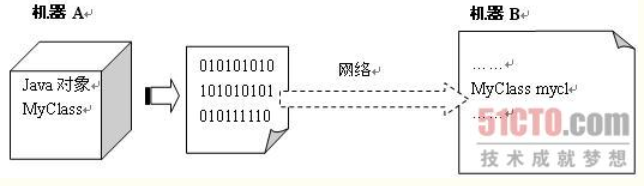
RPC并不是一个存粹的java概念，因为在java诞生之前就已经有RPC这个概念，RPC是由“Remote Procedure Call”的缩写，也就是远程过程调用。在java之前的大多数编程语言，如：Fortran，C，Cobol等等，都是过程性的语言，而不是面向对象的。所以，这些编程语言很自然地用过程表示工作，如：函数或子程序，让其在网络上另一台机器上执行。说白了，就是本地计算机调用远程计算机上的一个函数。如图：



## 二者结合就是RMI

RMI的英文全称“Remote Method Invovation”他的中文名称是“远程方法调用”，他就是利用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：



# 优点

这种机制给分布计算机的系统设计、编程都带来了极大的方便。只要按照RMI的规则设计程序，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哈希表这样的复杂类型作为一个参数进行传递。

# 缺点

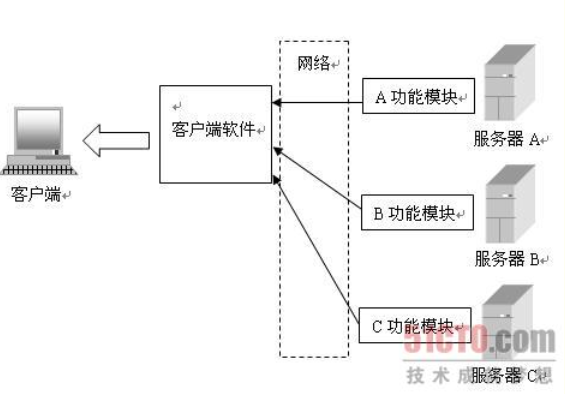
如果较为简单的方法调用，其执行效率也许会比本地执行慢很多，即使和远程socket机制的简单数据返回的应用相比，也会慢一些，原因是，其在网络间需要传递的信息不仅仅包含该函数的返回值信息，还会包含该对象序列化后的字节内容。

# 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，所以如果想了解EJB的原理，只需要把RMI的原理搞清楚就行了。

# EJB中所谓的服务集群

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“服务集群”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，修改以后所有的客户端的软件都被修改了。如图：



# 这种部署难道是无懈可击

上图所示的这个“服务集群”看是“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，我们来把这个图画完整，再来看看有什么问题

# 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：

