一种新型的基于JAVA技术的WEB应用服务体系结构模型

上海 同济大学计算中心 胡 泳 张志浩 陈福民

三层结构中虽然引入了应用服务器的概念，比原来的B/S结构在性能、管理、扩展方面都有了很大的进步，但是也存在着某些问题，主要表现在Java Applet程序的下载时间比较长和网络资源的访问受到JAVA安全性限制两个问题。当浏览器发出HTTP请求时，Java Applet程序自动地从WEB服务器下载到客户端，由于JAVA的类文件相对比较大，如果网络带宽不够大（现在有许多地方存在这种情况，比如通过电话线与INTERNET连接），下载时间就可能相对较长，这对于企业级网络计算是不能容忍的。因此需要一种机制在靠近客户端的位置来缓存经常使用的Java Applet和一些静态数据，以减少下载时间和网络流量。

1. 新型多层软件体系结构的逻辑组成

常见的企业级网络一般是由分布在各地机构中的一组计算机组成一个局域网，各个局域网又组成一个庞大的广域网，数据分布存储在各地；或者整个企业网就是一个大的园区网，而它是由多个局域网组成的。因此可以提出这样一种设想：把整个企业网的逻辑中心（包括作为业务逻辑处理中心的应用服务层和数据库层）放在企业的中心机房以处理复杂的逻辑计算；同时在各个局域网中配置一个缓存设备，用来存放该局域网经常使用到的Java Applet程序和一些静态数据（如文件资源等），这样可以减少不必要 的网络数据传输，也可以避免Java Applet安全性的限制。基于以上思想，给出一种新型的多层软件体系结构如图1所示：

整个系统由四层组成：客户层、顶端WEB服务层、应用服务层和数据库层，分别实现用户界面、代理/缓存、业务逻辑和持续存储的功能。与基本的三层结构相比，该多层体系多了一个称作顶端WEB服务层的部分，它主要实现代理/缓存的功能。下面对各个层次的功能及它们之间的通信方式做以下介绍：

(1)客户层。客户层通常向用户提供应用的接口，是一个图形用户界面。这一层运行的是Java Applet程序，它们可以运行在WEB浏览器环境下，也可以运行在任何可以运行JAVA软件的环境下（如网络计算机）。客户层不需要完成任何重要的业务逻辑，也不以任何方式直接和数据库交互，同时也不保存任何本地的状态信息，它只提供与用户交互的功能，提供一个良好的人机界面。这样就保证了系统中的客户机是一个真正的“瘦”客户机。(见图1)

(2)顶端WEB服务层。顶端WEB服务层是多层体系结构中重要的一层，它是对基本三层体系结构的重大改进，它主要起代理和缓存的作用。通常它被放置在一个局域网内部，为这个局域网内部的多台客户机提供服务。一台顶端WEB服务器用缓存来存储应用需要的Java Applet程序和静态数据，提供访问本地资源（例如用户文件和打印机）的能力，起了一个Java Applet主机和访问其他服务的代理作用。

顶端WEB服务层主要包括以下几个部分：代理服务器、服务定位Servlet (Service Locator Servlet)、本地服务(Local Service)和代理Servlet(Proxy Servlet)。

①代理服务器的作用是缓存本地各个客户机经常使用的Java Applet程序和静态数据，与普通代理服务器的作用相同。

②服务定位Servlet的功能是根据客户机发来的请求寻找合适的服务，从而完成对客户机需求的数据和网络资源的存取。

③本地服务主要包括文件存取、打印、等录、配置和会话等，这些服务的功能是根据各个客户机的请求完成对本地资源的访问。例如，一个客户机需要使用本地的打印机，它向服务定位Servlet发出请求，并从服务定位Servlet得到一个访问打印服务的“句柄(Handle)”，然后就可以使用这个句柄来访问打印服务，完成打印。

④代理Servlet的功能是访问远端数据。如果客户机需要访问远端数据，那么它向服务定位Servlet发出请求，并从服务定位Servlet获得一个访问“句柄”，从而通过代理Servle访问远端数据。

在顶端WEB服务层中，代理服务器应该完全由JAVA语言编写，服务定位Servlet、本地服务和代理Servlet都由Servlet技术实现。这样可以充分利用JAVA语言自身的优点，实现各个部分之间的无逢结合。

(3)应用服务层。应用服务层是多层体系结构中最重要的一层。因为所有的逻辑业务处理功能都由这一层提供，整个系统中所有对数据库的操作也都是由这一层来完成。

应用服务层包括完成业务逻辑处理所需要的各种服务，这些服务以API的方式提供，客户通过调用这些API来完成对数据库的操作。例如认证服务通过访问企业的认证数据

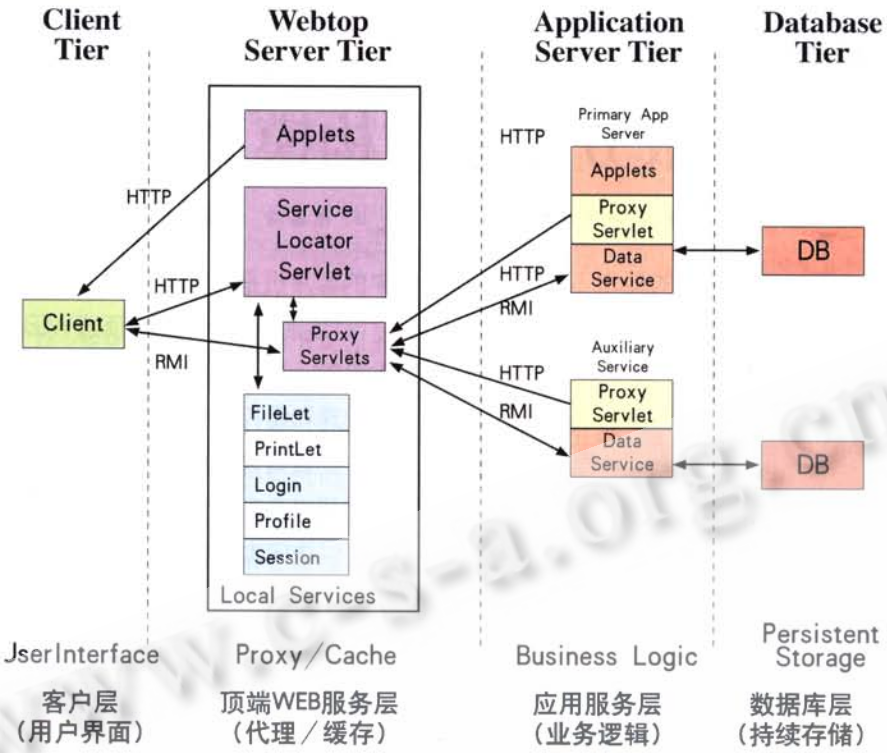

图 1 新型的多层软件体系结构示意图

库来验证用户口令数据是否正确。

在本层中，对于每一种应用服务，都有一个代理Servlet相对应，它会客户第一次发出请求时自动下载到顶端WEB服务层。当某个客户端请求某种服务时，顶端WEB服务层的服务定位Servlet会传给它对应这种服务的一个“句柄”。客户端利用这个“句柄”向一个代理Servlet发送请求，这个代理Servlet将这个请求发送到应用服务层，再由应用服务层中的某个服务完成对这个请求的响应。

由于客户端的Java Applet是通过顶端WEB服务器由应用服务器下载到本地，因此可以将顶端WEB服务器看成它的宿主机，所以Java Applet可以和顶端WEB服务器直接通信，利用本地资源，也可以通过顶端WEB服务器与本地局域网中的其他机器通信，避免了广域网上大量的数据传输。另外，由于客户端的Java Applet程序和顶端WEB服务器中的代理Servlet都是从应用服务器中自动下载来的，因此多层体系结构中的配置也符合客户端配置最小的B/S结构原则。

应用服务层中的大多数应用服务是由servlet的集合和一个后者多个RMI对象来完成的，也可以利用EJB模式。值得注意的是，由于任何一个应用服务可以调用其他的服务，所以一个实际的应用可以演变为更多的层次结构。

(4)数据库层。最后一层是数据库，它的功能是存储应用中的数据。它一般采用关系型数据库(RDB)或面向对象数据库(OODB)。数据库和应用服务层共同完成业务规则、验证和持续存储的实现，它们之间通过JAVA的数据库接口JDBC实现。

(5)各层之间的通信方式。多层体系结构由于包括的层数比较多，因此其内部的通信机制相对比较复杂。客户层与顶端WEB服务层之间使用HTTP和RMI两种通信协议。客户端使用HTTP协议将存储在顶端WEB服务器的Java Applet程序下载到本地运行。客户层还可以使用HTTP协议调用顶端WEB服务器中的服务定位Servlet，服务定位Servlet返回一个RMI对象索引，然后客户机利用这个索引与代理Servelt来通信，这里使用的协议是RMI，从而获得需要的数据或服务。顶端WEB服务层与应用服务层之间使用的协议也是HTTP和RMI。顶端WEB服务层使用HTTP协议从应用层下载Java Applet程序和Java Servelt程序（代理Servelt）。当顶端WEB服务层中的代理Servelt通过RMI获得客户层的请求时，它便使用RMI与应用服务层中的某种服务通信，转发请求，获得应用服务层发送的结果后再转发给客户层。数据库和应用服务层之间通过JAVA的数据库接口JDBC实现。

2. 多层软件体系结构的实现

上面对多层软件体系结构的组成和内部通信方式做了比较详细的讨论，那么作为一个系统，这些部件怎样在一个企业网内分布呢？下面给出一种新型的多层应用软件体系结构中各层的分布图(见图2)：

客户层是用户所在地，顶端WEB服务层通常是在物理上与客户端接近的服务器，每台顶端WEB服务器可以支持多个用户，具体的数量应该由一定区域内的客户数量和顶

端WEB服务器提供的额外服务所决定。

多层结构的具体实现过程如下:

当一个用户第一次访问某一个应用时，对应于该应用的applets和代理servlet自动地从应用服务器下载到顶端WEB服务器。这样做有两个好处：把applets缓存在接近客户端的顶端WEB服务器中可以为以后的用户减少applets下载的时间，并且这种机制使顶端WEB服务器成为applets的宿主机，从而可以实现applets与顶端WEB服务器的通信。

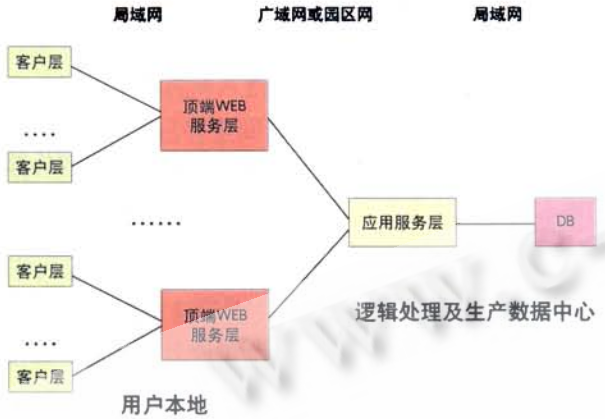

图 2 新型的多层软件体系结构各层分配图

一旦applet从顶端WEB服务器下载到客户端，它便在客户端运行并且向顶端WEB服务器中的Servlets(这些Servlets根据需要自动从应用服务器中下载)发送请求信息。一个servlet接到请求信息后，或者在本地执行相关的工作，或者把该请求信息传递给另一个服务。在一个典型的用户任务中，客户机可以调用一个主要的应用和多个辅助服务。

诸如打印和用户文件访问的服务请求由顶端WEB服务器中的服务处理，而需要访问远端数据的请求则被传递给应用服务器。通过使用多重服务，一个客户端可以访问来自多个数据源的数据。多层体系结构的一个重要特色是位于顶端WEB服务器中和位于应用服务器中的服务都是可重用的并且可以通过应用程序平衡。例如，一个应用服务能够提供货币兑换利率的信息，组织顾客数据或者帐户信息。每一种服务只需要实现一次并且可以被所有的客户应用程序重用。而且，客户端的applets和服务端的servlets都可以自动下载，这就使企业级应用程序的配置和发布变的相当容易。

3. 多层结构的优点

(1)可伸缩性好。多层体系结构通过增加可同时存在的客户连接数来提高应用程序的可伸缩性和性能。在两层模型中，每一个客户都与数据库保持一个连接，而在多层体系中，客户机只与应用服务器连接，应用服务器通过一个预先分配的数据库连接池与数据库连接来处理客户请求。这样可以减轻数据库的负担，因为系统可以根据客户端请求的多少来动态调整连接池中的连接数，使系统消耗较少的资源来完成客户端的请求；另外可以适当的增加应用服务器的数目来平衡应用服务层的负载，从而提高系统的可伸缩性。

(2)客户机范围广。由于在多层体系结构中，大部分逻辑计算都集中在中间层，因此客户机需要很少的资源，这就可以使多种客户平台能够运行应用程序。

(3)网络效率高。多层体系结构通过如下几种途径来减少网络流量，从而增强整个系统的性能。首先，对于相对静态的数据，可以下载到顶端WEB服务层进行缓存，以供客户快速存取，这种情况下，数据只在广域网中传输一次，而传统的C/S或B/S结构中，对于每一次客户请求，数据都需要在广域网中传输。另外，诸如打印和文件访问等需要访问靠近客户端的网络资源的服务，可以在靠近那些资源的顶端WEB服务层实现。

(4)安全性高。多层体系结构中，实现安全的应用也比较容易。首先，安全认证服务是一个可以被所有应用程序调用的辅助服务，不必为每个应用单独编写。安全认证服务提供了访问应用服务器的证书，而应用服务器是一个数据库的安全网关，使用防火墙技术，数据库的访问只能由应用服务器进行，各个客户机无法直接访问到数据库。最后，在多层体系结构中，顶端WEB服务器可以设置在一个防火墙中，提供了一种既工作在内部网又工作在外部网中的模式。

(5)可管理性强。多层体系结构对于应用程序的配置来说不需要对客户端有任何了解，客户端唯一的要求就是要安装支持HTTP-S(HTTP的安全版)、JAVA技术和SSL(安全套接层)。从管理的角度看，应用程序的分发和管理都集中在应用服务器上，在客户端不需要安装和配置任何应用程序。

既然应用系统运行过程中，在顶端WEB服务器中不需要任何的程序配置和管理，它也实现了"零管理"。顶端WEB服务器唯一的管理任务是在安装的时候，包括安装原始的操作系统，Java Web服务器，打印配置和其他的一些必要的服务。所有的应用程序代码在需要时自动下载，没有特殊的应用程序代码长期驻留在顶端WEB服务器中。

(6)可重用性好。整个系统由许多服务组成，每种服务可以被不同的应用重用。构建系统时采用面向对象的组件模式，各种服务又由许多可重用的组件构成，进一步增加了系统的可重用性。例如所有的应用服务都可以共享安全认证服务，而不必在每个应用中重新开发，这就避免了在每个应用程序中都进行管理和配置。

参考文献

[1] http://www.sun.com [2] 计算机世界 1999年第35期