Web服务器与数据库服务器之间通信的不同方案

本文介绍了实现Web服务器与数据库服务器之间通信的不同解决方案：CGI，API，中间集成件。对它们的概念、工作流程作了阐述，并对它们的性能进行了比较。

广州通信学院　王兰波　雷渭侣　孙正严

随着WWW网的日益普及，人们越来越多地接触到了动态网页的制作。动态网页的内容不是固定的，而且一般要与后台数据库服务器打交道，需要后台数据库服务器根据浏览器提供的不同参数才能确定具体显示内容。例如，可以编写一个外部应用程序让浏览器用户选择你的信息资料库中的内容。Web应用服务器支持Web服务器与数据库服务器之间的通信，提供应用服务。它有许多种不同的解决方案，目前使用得最广泛的有三种方式：CGI，API和集成中间件。下面分别对它们加以介绍。

CGI方式

(1)CGI概念。通用网关接口CGI（Common Gateway Interface），是一个用于定义Web服务器与外部程序之间通信方式的标准，它使得外部程序能够生成HTML、图象或者其他内容。简单地说，一个CGI接口的功能就是在超文本文件和服务器主机应用程序之间传递信息。

(2)CGI的性能评价。这种技术提供了一种基本的数据库连接机制。简单的数据库连接方案使用CGI作为与数据库连接的一种手段，通过将CGI程序与数据库连接，使用标准输入参数构造一个事务并处理之，然后以HTML格式生成结果，这是一种将CGI体系结构扩展到数据库的相当容易的方法。

CGI是一个定义良好并被广泛支持的标准。它的跨平台性能极佳，几乎可以在任何操作系统上实现，例如，DOS、WINDOWS、UNIX、OS/2、Macintosh等。而且，实现CGI的编程语言也有很多选择。目前最为流行的CGI程序设计语言主要有：C，Shell，Perl和VisualBasic。其他一些语言也有许多人在使用，例如，TCL，Fortran及AppleScript等。用于开发CGI的几种常用语言，除了C语言是编译型语言以外，其他几种都是解释型的。解释型语言没有编译型语言的效率高，但可移植性较强。

但是这种体系结构存在着一些无法解决的问题，基于CGI的数据库接口有许多限制：

①无法保持连接状态。CGI应用进程是为Web浏览器的每一个请求而fork产生的，它必须每次重建与数据库的连接。要求一个持续的数据库连接的应用不得不使用复杂的技巧来解决这个问题。例如，有一个应用允许用户每次只看到查询结果集中的一部分并为用户提供了“向前”、“向后”翻页功能。CGI应用进程很难完成这个任务，因为它无法将结果集及其状态(目前显示的子集位于结果集的什么位置)保存下来。设计人员不得不使用某种令牌传递算法指出用户是谁，它们提出了什么请求，它们位于结果集的什么位置，等等。

当应用变得更复杂和要求处理更复杂事务的能力时，例如，创建多人玩的游戏，是不可能使用这种基于CGI的体系结构的。

②性能上的考虑。执行多数据库连接的多用户应用可能会导致不可预知的代价。对于有限的用户系统，这不是一个问题。但随着用户的增多，性能问题越来越突出。CGI的应用程序一般都是一个独立的可执行程序，它和WWW服务器各自占据着不同的进程，而且一个CGI程序只能处理一个用户请求。这样，每当有一个用户请求，都会激活一个CGI进程，当用户请求数量非常多时，会大量挤占系统的资源，例如，内存、CPU时间等，造成效能低下。而且，由于数据库连接不是持续的，因而必须不断地重新获得结果集。这可能会导致应用与数据库之间更多的通信量。

③CGI的安全性。CGI规范本身并不是引起不安全的原因，但它代表了许多Web浏览器的特性。这些特性使得支持CGI的Web服务器在提供信息的同时，也赋予了Web浏览器用户对服务器一定程度的控制能力。一个对安全问题缺少考虑的CGI程序将使得Web浏览器用户获得程序设计人员预计的对Web服务器的控制。CGI安全方面的漏洞主要是由于程序设计人员有意或无意在主机

系统中遗漏安全漏洞给非法黑客创造条件。

API方式

API方式中最著名的是Microsoft的ISAPI和Netscape的NSAPI。API克服了CGI的某些局限，比CGI性能更高，开发人员可以更简单地开发复杂应用，也提供了安全措施。下面以Microsoft的ISAPI为例，简单介绍API方式的原理和性能。

(1)ISAPI概念。Internet服务器应用程序接口ISAPI(Internet Server Application Program Interface)，是微软提供的一套面向Internet服务的API接口，它能实现CGI所提供的全部功能，并在此基础上进行了扩展，例如，提供了过滤器应用程序接口。

(2)ISAPI性能评价。ISAPI可以通过参数选择永久连接，即服务器在处理完一个用户请求后不断开连接，继续等待处理另一个HTTP请求的到来。在一个DLL里可以设置多个用户请求处理函数，API函数共享资源，占用空间较小。还可以通过数据缓冲区加速API函数的执行速度。此外，ISAPI的DLL应用程序和WWW服务器处于同一个进程中，效率明显要高于CGI。

不过目前ISAPI还不具备跨平台的功能，只能用于微软自己的Windows和NT操作系统，服务器平台也仅限于IIS(Internet Information Server)和MS personal Web server以及NT workstation上的peer Web server。能用来开发ISAPI应用的语言也不如CGI多，主要有VisualC++4.1以上版本、VisualBasic5.0和BorlandC++5.0。

而且，这种解决方案仍然依赖于中间件脚本访问数据库，因此和CGI方式一样，存在着许多性能问题。同时，由于API不能提供扩展的数据类型，对于一些复杂问题的描述，API方式就有些力不从心了。

中间集成件方式

中间集成件方式通常是由数据库厂商根据自己数据库的特点开发的Web应用程序，例如，Informix公司的IWAS(Informix Web Application Server)和Oracle公司的OWAS(Oracle Web Application Server)。这种解决方案的优点在于将Web应用与数据库系统做到了紧密的结合，克服了CGI、ISAPI、NSAPI的局限性和不兼容性问题，能充分发挥Web的潜在能力，提供高性能和增强的可管理性。不同的中间集成件有着不同的解决方案。下面分别介绍Informix公司的中间集成件IWAS和Oracle公司的中间集成件OWAS，最后对两者进行简单的比较。

(1)IWAS

①工作原理。Informix公司的中间集成件IWAS的工作原理如图1所示。

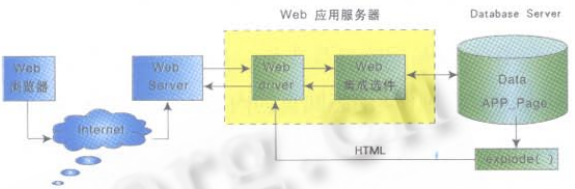

图1 IWAS的工作原理图

当一个最终用户通过Internet向Web服务器发出第一个应用请求时，系统运行Web驱动模块(Web driver)。Web驱动模块是一个运行在Web应用服务器上的轻量进程，它初始化并管理从最终用户的Web浏览器到具体的应用之间的连接(这个连接被称为一个会话)。Web驱动模块在共享内存中为客户指定一块区域，并为这个会话产生一个会话ID，然后将用户请求的页面标识和参数送给Web集成选件处理。Web集成选件收到Web驱动模块的请求后，从数据库中读出指定的APP_Page并fork那个页面应用，然后执行Web explode( )函数。Web explode( )函数执行APP_Page中的查询，格式化结果，形成一个标准的HTML文件，然后将这个HTML文件返回给Web驱动模块，最后由驱动模块将这个HTML文件返回给Web浏览器。将HTML页面返回给用户后，Web驱动进程终止。

②IWAS的主要特点

· 高性能和可伸缩性。Web集成选件与数据库服务器紧密集成，充分利用Informix的多线程结构和并行处理能力。因此，当Web数据库应用用户数量增加时，数据库服务器将充分利用所有系统可用资源，以保证性能最高。对每个客户请求无需启动一个应用拷贝。Web集成选件提供一套完整的连接和会话管理机制，它可以由多个客户共享，从而降低系统开销。用户可以根据负载的轻重设置允许启动的Web集成选件的数量。Web集成选件本身提供负载平衡算法，在所有的Web应用之间均匀分布客户请求，起到了动态平衡负载的作用。

作用于表和索引的存取方法也是由数据库服务器负

责管理的，它通过用户身份认定和读写级的存取，保证了对整个Web应用的安全访问，只有被授权的用户可以对一个结点的任何内容进行查找、编辑、删除或拷贝。

·保持连接状态。当一个最终用户通过请求应用的第一个HTML页面开始一个会话时，Web集成选件在共享内存中为这个会话指定一个共享内存段存放这个会话的特定信息，并为这个会话指定一个会话ID。当最终用户请求第二个及随后的页面时，Web集成选件通过这个会话ID继续这个会话。通过这种方式，Web集成选件保持了Web浏览器与数据库服务器之间的连接状态。这对于一些复杂应用，例如，在上一次查询的基础上进行进一步的查询，有很大意义。

而且，IWAS不是每当一个用户提交一个查询请求就产生一个新的应用。一旦应用大量产生了它的应用进程，它就常住内存。共享内存跟踪所有的状态信息以及用户和应用之间的关系，在Web浏览器与数据库服务器交互的过程中减少了不断地重新打开与数据库的连接的代价，因而克服了传统的CGI方式的一个重要限制。

·其他特点。程序设计方便，可管理性强，开发环境开放、强有力，运行环境安全，与Web浏览器和数据库服务器相独立，也是IWAS的重要特点。

(2)OWAS

①工作原理。Oracle公司的中间集成件OWAS的工作原理如图2所示。

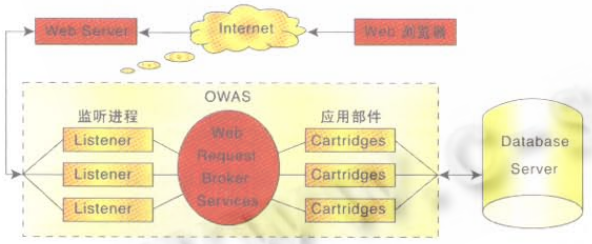

图 2 OWAS的工作原理

来自Web浏览器的客户请求首先由监听进程(Listener)进行处理，然后转交给WRB(Web Request Broker)。WRB是一个进程集合，它用于确保系统中的所有构件都有一个统一的配置、权限验证和资源集合。WRB将不同类型的服务请求送给相应的应用部件Cartridges进行处理。数据库服务器通过存储过程的调用，生成HTML文件返回给WRB，然后再由WRB返回给Web浏览器客户。

②主要特点

·高性能。OWAS允许管理和创建一个完整的应用进程池。这种应用进程池是一个可重用的进程集合，它能够根据工作负载的轻重被适当地调整，并为大量用户提供当今最快的往返(Round - trip)响应速度。同时，OWAS还确保了一个在所有各处都一致的管理和权限验证机制。

·可伸缩性。无论应用进程是如何被有效地管理，在OWAS系统中都会存在一个极限点。系统资源限制包括CPU、磁盘I/O或网络带宽，这些限制通常发生在高负荷情况下。OWAS允许用户通过将系统的规模扩大来克服这些局限性。用户可以将OWAS的不同构件(监听进程，应用部件)放在不同的服务器上，而且还可以让这些构件的多个集合来分别处理工作负荷(例如，多个监听进程，多个应用服务器)。

(3)IWAS和OWAS的体系结构的比较

①连接层的处理时间主要体现在对客户请求的处理、保持数据库连接的处理以及对负载均衡的调度，我们称这些处理为连接层的核心处理。IWAS的核心处理是在数据库服务器上进行的，连接层对Web服务器的影响很轻。OWAS的核心处理是在Web服务器(或者专用的中间应用服务器)上，它加重了Web服务器的负担，但相对来说在数据库服务器上的处理要轻些。

②从可伸缩性来讲，IWAS主要是通过在数据库服务器端的参数来进行调整，而OWAS主要是通过在Web服务器(或者专用的中间应用服务器)端的参数来进行调整。连接层的可伸缩性，OWAS比IWAS更加灵活，但OWAS比IWAS要消耗更多的系统资源。

③在系统的硬件环境方面，如果连接层选用OWAS，则需要采用高性能的机器来做Web服务器(或者专用的中间应用服务器)；如果连接层选用IWAS，则需要配置高性能的数据库服务器。从总体的硬件资源需求来看，OWAS比IWAS的要求更高。■

参考文献

1 Daniel J. Berlin ,et al·精通CGI编程·清华大学出版社

2 王文婷，唐世渭·利用Informix Web DataBlade开发Web应用·《计算机世界》·1997年第3期学习与实践版

3 龚建勇·ISAPI与CGI的比较与实现

4 Informix和Oracle的各种联机资料