(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 104767834 B
(45)授权公告日 2018.09.14

(21)申请号 201510221918.3

(22)申请日 2007.04.11

(65)同一申请的已公布的文献号
申请公布号 CN 104767834 A

(43)申请公布日 2015.07.08

(30)优先权数据
60/744720 2006.04.12 US

(62)分案原申请数据
200780021971.2 2007.04.11

(73)专利权人 思杰系统有限公司
地址 美国弗罗里达州

(72)发明人 B·J·佩德森 P·桑达拉贾
R·辛哈 T·特里德

(74)专利代理机构 北京泛华伟业知识产权代理有限公司 11280
代理人 王勇

(51)Int.Cl.
H04L 29/08(2006.01)
H04L 29/06(2006.01)

(56)对比文件
WO 2006/005078 A2,2006.01.12,
CN 1625734 A,2005.06.08,
WO 2005/088476 A1,2005.09.22,
US 2003/0043777 A1,2003.03.06,

审查员 马晔

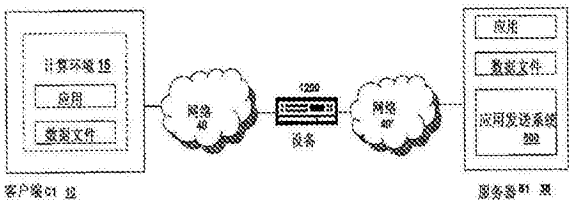
权利要求书3页 说明书136页 附图62页

(54)发明名称

用于加速计算环境到远程用户的传送的系统和方法

(57)摘要

本发明相关于计算环境到远程用户的传送的加速，该远程用户位于远程位置客户端。该计算环境可包括应用和被该应用使用或处理的数据文件。该应用和数据文件可被存储或通过远程服务器提供给客户端。用户可从服务器请求计算环境，由用户通过远程计算机该服务器提供应用的执行。例如，服务器可流发送应用到远程客户端。客户端和服务器可通过应用通信，该应用可加速客户端和服务器间的通信。例如，该应用可加速应用到远程用户的流发送。在一些情况中，该应用或远程用户也可以从服务器请求数据文件，并且该应用加速数据文件到远程用户的传送。同样的，在远程位置的用户通过任何将设备连接到应用和数据文件的网络获得加速的访问，该应用和数据文件对于用户位于远程位置。



CN 104767834 B

1.一种用于加速传送计算环境到远程客户端的方法,所述方法包括:

由部署为在服务器和远程客户端之间的中间设备的设备拦截多个文件,所述多个文件包括由所述服务器响应于来自所述远程客户端的执行应用的第一请求流式传输的应用程序;

由所述设备确定所述设备是否可以加速传输所拦截的包括由所述服务器响应于所述第一请求流式传输的应用程序的多个文件;

由所述设备并且响应于确定所述设备可以加速传输所述应用程序,确定是否使用加速程序配置所述远程客户端;以及

由所述设备并且响应于确定所述设备可以加速传输所述应用程序以及使用加速程序配置所述远程客户端,通过向所拦截的包括由所述服务器响应于所述第一请求流式传输的应用程序的多个文件应用一个或多个传输层传输加速技术,来加速传输所述应用程序。

2.根据权利要求1所述的方法,其中,所述一个或多个传输层传输加速技术包括:

传输控制协议多路复用;

传输控制协议池化;或

传输控制协议缓冲。

3.根据权利要求1所述的方法,还包括:

由所述设备并且响应于确定没有使用加速程序配置所述远程客户端,将所述加速程序传输到所述远程客户端。

4.根据权利要求3所述的方法,还包括由所述远程客户端在接收到来自所述设备的加速程序时自动地安装和执行所述加速程序。

5.根据权利要求4所述的方法,还包括:

由所述设备向所述远程客户端传输授权由所述远程客户端在预定时间执行所述应用程序的许可;

由所述设备从所述远程客户端接收指示所述应用程序正被执行的心跳消息;以及

由所述设备并且响应于接收所述心跳消息,传输授权由所述远程客户端在预定时间执行所述应用程序的更新的许可。

6.根据权利要求1所述的方法,还包括:

由所述设备高速缓存包括所述应用程序的多个文件;以及

由所述设备拦截来自所述远程客户端的对包括所述应用程序的多个文件中的一个或多个的第二请求,并且响应于所述第二请求向所述远程客户端传输高速缓存的多个文件中的一个或多个。

7.根据权利要求1所述的方法,其中,加速传输应用程序还包括:

通过向由所述服务器流式传输的多个文件应用第一一个或多个传输层传输加速技术来加速从所述服务器到所述设备的应用程序的传输,以及通过向传输到所述远程客户端的多个文件应用第二一个或多个传输层传输加速技术来加速从所述设备到所述远程客户端的应用程序的传输。

8.根据权利要求1所述的方法,还包括由所述设备经由在所述服务器和所述设备之间的池化的传输层连接接收来自所述服务器的包括所述应用程序的多个文件,其中包括所述应用程序的多个文件是与所述服务器经由所述设备通信的多个远程客户端使用的。

9.根据权利要求8所述的方法，还包括：

由所述设备经由池化的传输层连接，接收包括在多个远程客户端和所述服务器之间的多路复用的通信会话的一个或多个数据包；以及

由所述设备将所述多路复用的通信会话解复用为在所述服务器和所述多个远程客户端中的每一个之间的一个或多个通信会话。

10.根据权利要求1所述的方法，其中，确定所述设备是否可以加速传输所述应用程序还包括：

响应于应用于包括所述应用程序的多个文件的一个或多个在先传输的传输层传输加速技术的过去性能的日志，确定所述设备是否可以加速传输所述应用程序。

11.一种用于加速传送计算环境到远程客户端的系统，所述系统包括：

被部署为在一个或多个远程客户端和一个或多个服务器之间的中间设备的设备，其拦截多个文件，所述多个文件包括由所述服务器响应于来自所述远程客户端的执行应用的第一请求流式传输的应用程序；确定所述设备是否可以加速传输所拦截的包括由所述服务器响应于所述第一请求流式传输的应用程序的多个文件；并且响应于确定所述设备可以加速传输所述应用程序，确定是否使用加速程序配置所述远程客户端；以及

所述设备的包引擎，其响应于确定所述设备可以加速传输所述应用程序以及使用加速程序配置所述远程客户端，通过向所拦截的包括由所述服务器响应于所述第一请求流式传输的应用程序的多个文件应用一个或多个传输层传输加速技术，来加速传输所述应用程序。

12.根据权利要求11所述的系统，其中，所述一个或多个传输层传输加速技术包括：

传输控制协议多路复用；

传输控制协议池化；或

传输控制协议缓冲。

13.根据权利要求11所述的系统，其中，所述设备还包括：

存储客户端侧加速程序的存储元件；并且所述设备：

响应于确定没有使用加速程序配置所述远程客户端，将所述加速程序传输到所述远程客户端。

14.根据权利要求13所述的系统，其中，所述远程客户端还安装所述加速程序，并且执行所述加速程序。

15.根据权利要求14所述的系统，其中，所述加速程序被配置为提供用于将包括所述应用程序的多个文件存储到高速缓存中的高速缓存；以及

所述加速程序接收在预定时间周期内执行在所述高速缓存中的包括所述应用程序的多个文件的授权，以及

响应于预定时间周期的未终止，将对包括所述应用程序的多个文件的请求重定向到在所述高速缓存中的所存储的多个文件。

16.根据权利要求11所述的系统，其中，所述设备还包括存储包括所述应用程序的多个文件的存储元件；以及

所述包引擎拦截来自所述远程客户端的对包括所述应用程序的多个文件中的一个或多个的第二请求，并且响应于所述请求向所述远程客户端传输所存储的多个文件中的一个

或多个。

17.根据权利要求11所述的系统，其中，所述包引擎还通过向由所述服务器流式传输的多个文件应用第一一个或多个传输层传输加速技术来加速从所述服务器到所述设备的应用程序的传输，以及通过向传输到所述远程客户端的多个文件应用第二一个或多个传输层传输加速技术来加速从所述设备到所述远程客户端的应用程序的传输。

18.根据权利要求11所述的系统，其中，所述设备还经由在所述服务器和所述设备之间的池化的传输层连接接收来自所述服务器的包括所述应用程序的多个文件，其中包括所述应用程序的多个文件是与所述服务器经由所述设备通信的多个远程客户端使用的。

19.根据权利要求18所述的系统，其中，所述设备还经由池化的传输层连接，接收包括在多个远程客户端和所述服务器之间的多路复用的通信会话的一个或多个数据包；以及

所述包引擎将所述多路复用的通信会话解复用为在所述服务器和所述多个远程客户端中的每一个之间的一个或多个通信会话。

20.根据权利要求11所述的系统，其中，所述设备响应于应用于包括所述应用程序的多个文件的一个或多个在先传输的传输层传输加速技术的过去性能的日志，确定所述设备是否可以加速传输所述应用程序。

用于加速计算环境到远程用户的传送的系统和方法

[0001] 本申请是思杰系统有限公司于2007年4月11日提交的申请号为200780021971.2、发明名称为“用于加速计算环境到远程用户的传送的系统和方法”的专利申请的分案申请。

[0002] 相关申请

[0003] 本申请要求美国临时专利申请NO.60/744,720的利益和优先权，该美国临时专利申请标题为“用于加速计算环境到远程用户传送的系统和方法”，并且在2006年4月12日提出申请，该申请通过引用在此加以结合。

技术领域

[0004] 本发明相关于向客户端的远程用户加速包括应用和数据文件的计算环境的传送的系统和方法，该客户端相对于服务器位于远程。

背景技术

[0005] 管理和控制企业环境花费时间、金钱、和资源。在许多情况中，这是因为应用和数据管理过程是分散的和劳动力密集型的。例如，管理人员的大部分时间花在提供更多的存储空间或为公司数据执行备份，或更新服务器以处理企业数据的增长上。同样的，管理人员需要生成和提供新的服务器以处理数据的增长。另外，管理人员花费时间更新或配备服务器以提供特定用户应用。另外，公司数据的相当大部分可以驻留在公司的数据中心的外部。例如，公司公文、文件和数据可存在或分散到各种相对于数据中心为远端的各种计算机中。

[0006] 在努力减少管理和控制企业的数据和应用所需要的时间、金钱和资源，许多公司有统一和集中化的服务器、企业数据和应用。虽然统一和集中化已经减少了一些花费并已经产生了一些利益，集中化数据和应用在提供对数据和应用访问中提出了额外的挑战。一个这样的挑战包括远程用户试图通过广域网(WAN)连接访问文件。例如，在分支机构的远程用户试图通过WAN打开存储在企业数据中心中的微软办公(Microsoft Office)文件，该分支机构通常有到企业数据中心的网络连接，其操作速度慢于局域网连接许多。由于WAN的等待，可靠性和带宽，该远程用户通过网络对于文件的访问可能延迟。对于较大的文件延迟可能更大。此外，随着在远程用户和企业数据中心间距离的增加，访问文件中网络延迟的频率和长度也会增加。在WAN上增加虚拟专有网络，安全和其他网络层，可进一步减少到远程用户的可用的带宽并增加在访问文件中的延迟。远程办公室的更低的速度和带宽可在访问远程文件中引起无法接受的延迟。为了避免在远程文件访问中的延迟，远程用户可以拷贝并在本地使用文件，消除集中化操作的目的。另外，WAN连接可能没有LAN连接可靠，造成包丢失和网络连接断开。在文件操作期间，例如保存或打开文件期间，可发生WAN中断，进一步使得远程用户经历延迟。

[0007] 因此，需要改进远程用户到集中化应用和数据文件的访问的系统和方法，包括加速应用和数据文件到远程用户的传送。

发明内容

[0008] 本发明涉及用于加速应用和数据文件的计算环境到远程用户的传送的系统和方法。该应用和数据文件可通过相对于客户端为远端的服务器存储或提供。例如用户，诸如远程雇员，可在分支机构里使用没有本地可用的应用和/或数据文件的计算机。该用户可能想要使用该远程客户端不可用的字处理应用编辑一个企业文件。该用户可从服务器请求计算环境，由用户通过远程客户端该服务器提供需要的应用的执行。例如，该服务器可流式发送该应用到远程客户端。远程客户端和服务器可通过一个设备通信，该设备可加速远程客户端和服务器间的通信。例如，该设备可加速应用到远程用户的流式发送。在一些情况中，应用或远程用户也可以从服务器请求数据文件，并且该设备加速数据文件到远程用户的传送。就这点而论，本发明通过任一网络连接的设备为在远程位置的用户提供对应用和数据文件的加速的访问，该应用和数据文件对于用户位于远程位置。

[0009] 一方面，本发明涉及用于加速应用和数据文件的计算环境到位于远端位置的客户端的用户的传送的方法。该方法包括通过服务器接收来自远程客户端执行应用的请求。远程客户端和服务器通过一个设备通信。该方法还包括通过服务器流式发送用于执行的应用到远程客户端。该客户端向服务器传输对该应用可使用的数据文件的请求，该设备加速该数据文件到远程客户端的传输。

[0010] 在本发明的一个实施例中，该方法包括通过该设备加速应用到远程客户端的流式发送。在另一个实施例中，通过执行下述加速技术之一，该设备加速数据文件的传输或应用的流式发送：1)压缩；2)解压缩；3)传输控制协议池；4)传输控制协议多路复用技术；5)传输控制协议缓冲；以及6)高速缓冲存储。在另一个实施例中，该方法包括通过在远程客户端上的加速程序加速在远程客户端和服务器间的通信。在该方法的一些实施例中，该设备建立与远程客户端的虚拟专有网络连接或加密套接字协议层(SSL)连接。在其他实施例中，该方法包括通过该设备加速网络包有效负载，该网络包通过在远程客户端和服务器之间的传输层连接进行传送。

[0011] 在本发明的一个实施例中，该方法包括在收到来自远程客户端的与服务器建立连接或会话的请求时通过该设备传输加速程序到远程客户端。在一些实施例中，在收到该设备的请求时该远程客户端自动安装并执行加速程序。在一些实施例中，该方法包括通过在远程客户端上的加速程序执行下述加速技术之一：1)压缩；2)解压缩；3)传输控制协议池；4)传输控制协议多路复用技术；5)传输控制协议缓冲；以及6)高速缓冲存储。在一些实施例中，该远程客户端对于应用或服务器透明地执行加速程序。

[0012] 在本发明的一些实施例中，该方法包括由该设备确定该应用能够被加速，以及响应该确定而向远程客户端传输加速程序。在另外的实施例中，该设备高速缓冲数据文件。在一个实施例中，该设备拦截对于数据文件的请求，并响应该请求传输高速缓存的数据文件到远程客户端。

[0013] 在另一个方面，本发明涉及用于加速应用和数据文件的计算环境到位于远端位置的客户端的远程用户的传送的系统。该系统包括用于加速一个或多个远程客户端和一个或多个服务器间通讯的设备。该系统也包括用于接收来自远程客户端执行应用的请求的服务器。该远程客户端和服务器通过该设备通信。该服务器流式发送用于执行的应用到远程客户端。该客户端向服务器传输对于该应用可使用的数据文件的请求，并且该应用加速该数据文件到远程客户端的传输。

[0014] 在本发明的一些实施例中，该设备加速应用到远程客户端的流式发送。在一个实施例中，通过执行下述加速技术之一，该设备加速数据文件的传输或应用的流式发送：1)压缩；2)解压缩；3)传输控制协议池；4)传输控制协议多路复用技术；5)传输控制协议缓冲；以及6)高速缓冲存储。在另一个实施例中，该系统包括在远程客户端上的加速程序，该加速程序加速在远程客户端和服务器间的通信。在一个实施例中，该设备建立与远程客户端的虚拟专有网络连接或加密套接字协议层(SSL)连接。

[0015] 在本发明的一些实施例中，该设备加速网络包有效负载，该网络包通过在远程客户端和服务器之间的传输层连接进行传送。在一个实施例中，该设备在收到来自客户端的请求时传输加速程序到远程客户端，以与服务器建立连接或会话。在一些实施例中，在收到来自设备的请求时该远程客户端自动安装并执行加速程序。在远程客户端上的加速程序执行下述加速技术之一：1)压缩；2)解压缩；3)传输控制协议池；4)传输控制协议多路复用技术；5)传输控制协议缓冲；以及6)高速缓冲存储。在一个实施例中，该远程客户端对于应用或服务器透明地执行加速程序。

[0016] 在本发明系统的另一个实施例中，该设备确定应用能够被加速，以及响应该确定向远程客户端传输加速程序。在一个实施例中，该设备包括用于高速缓冲数据文件的高速缓存。在一些实施例中，该设备拦截对于数据文件的请求，并响应该请求传输高速缓存的数据文件到远程客户端。

附图说明

[0017] 该发明的这些和其他方面从下述细节描述和附图将会十分明显，下述细节描述和附图用于示出而非限制本发明，并且其中：

[0018] 图1A是描述网络环境的框图；

[0019] 图1B是描述在网络环境中远端的计算环境的一个实施例的框图；

[0020] 图1C和1D是与描述的实施例相关的计算机的具体实施例的框图；

[0021] 图1E是描述适合传送计算环境到客户端的环境的框图；

[0022] 图1F是描述系统的一个实施例的框图，该系统用于提供多个应用程序，这些应用程序通过在网络(web)服务目录中的GUI的公布为本地机器可用；

[0023] 图2是描述步骤的一个实施例的流程图，该步骤用于选择应用程序的执行方法；

[0024] 图3A是描述本地机器的一个实施例的框图，该本地机器通过环球网(World Wide Web)启动程序邻居(Program Neighborhood)应用的执行；

[0025] 图3B是描述步骤的一个实施例的流程图，本地机器使用这些步骤访问使用web服务目录所列举的应用程序；

[0026] 图4A是网络环境的实施例的框图，该网络环境为本地机器提供对应用程序的基于策略的访问；

[0027] 图4B是描述策略引擎的更详细的实施例的框图；

[0028] 图4C是步骤的一个实施例的流程图，策略引擎使用这些步骤基于接收到的关于本地机器的信息作出访问控制决定；

[0029] 图4D是描述计算机网络的实施例的框图，在该计算机网络中提供对多个应用会话的授权的远程访问；

[0030] 图4E是描述步骤的一个实施例的流程图，会话服务器使用这些步骤将本地机器和它的相关联的应用会话连接；

[0031] 图5是描述了步骤的一个实施例的流程图，会话服务器使用这些步骤将客户端节点和它的相关联的应用会话连接；

[0032] 图6是描述了远程机器的一个实施例的框图，该远程机器包括提供应用列举的管理服务；

[0033] 图7是描述步骤的一个实施例的流程图，使用这些步骤访问包括应用程序的多个文件；

[0034] 图8A是描述计算机的一个实施例的框图，该计算机在操作系统的控制下运行，该操作系统减少了应用的兼容性和应用群集度(sociability)的问题；

[0035] 图8B是描述多用户计算机的一个实施例的框图，该多用户计算机拥有减少了的应用的兼容性和应用群集度的问题；

[0036] 图8C是描述步骤的一个实施例的流程图，这些步骤在方法中用于连接过程与一个孤立的范围；

[0037] 图9是描述步骤的一个实施例的流程图，这些步骤在用于执行应用程序的方法中采用；

[0038] 图10是描述驻留在远程机器中的多个应用文件的一个实施例的流程图；

[0039] 图11是步骤的一个实施例的流程图，这些步骤在用于本地响应对与远程存储的文件相关的文件元数据的请求的方法中采用；

[0040] 图12是描述系统的一个实施例的框图，该系统用于本地响应对与远程存储的文件相关的文件元数据的请求；

[0041] 图13是描述步骤的一个实施例的流程图，这些步骤在用于访问在目录结构中的远程文件的方法中采用，该目录结构与本地执行的应用程序相关联；

[0042] 图14是描述系统的一个实施例的框图，该系统用于访问与应用相关联的目录结构中的文件；

[0043] 图15是远程机器的一个实施例的框图，该远程机器包括许可管理子系统；

[0044] 图16是描述在远程机器上管理服务中的部件的一个实施例的框图；

[0045] 图17是描述步骤的一个实施例的流程图，这些步骤用于从远程机器请求和保持许可；

[0046] 图18是描述状态的一个实施例的框图，该状态可与由管理服务监控的会话相关联；

[0047] 图19是描述包的实施例的框图，该包包括两个对象，每个对象包括包含应用的多个应用文件；

[0048] 图20是描述步骤的一个实施例的流程图，这些步骤在不重新启动操作系统而安装应用程序的基于策略的方法中采用；

[0049] 图21是描述步骤的一个实施例的流程图，这些步骤在不重新启动操作系统而安装应用程序的基于策略的方法中采用；

[0050] 图22是描述将在本地机器上执行的脚本的列举的一个实施例的屏幕照片；

[0051] 图23是描述系统的一个实施例的框图，该系统包括向独立环境中执行安装程序的

打包机制；

[0052] 图24是描述在环境中使用的步骤的一个实施例的流程图，在该环境中安装程序的执行需要重新启动操作系统；

[0053] 图25是描述远程机器的一个实施例的框图，打包机制向该远程机器中安装应用程序；

[0054] 图26是描述步骤的一个实施例的流程图，使用这些步骤将应用安装到应用独立环境中；

[0055] 图27示出了执行集成高速缓冲存储的设备的机构的一个具体实施例的框图；

[0056] 图28A是在方法的具体实施例中使用的步骤的流程图，该方法用于包处理和包处理计时器与设备操作的集成；

[0057] 图28B是在方法的具体实施例中使用的步骤的流程图，该方法用于实现图3A中的无效粒度(inval idat ion granulari ty)技术；

[0058] 图29A是步骤的流程图，这些步骤在使用无效命令以无效失时效对象(stale object)的方法的实施例中使用；

[0059] 图29B是步骤的流程图，这些步骤在合并对象组的无效的方法的具体实施例中使用；

[0060] 图29C是步骤的流程图，这些步骤在方法的具体实施例中使用，该方法中为对象确定因子(object determinant)分析客户端请求；

[0061] 图29D是步骤的流程图，这些步骤在方法的具体实施例中使用，该方法使用对象确定因子合并对象的组的无效；

[0062] 图30是在连接池的方法的一个实施例中使用的步骤的流程图；

[0063] 图31是在变换客户端和服务器请求的方法的一个具体实施例中使用的步骤的流程图；

[0064] 图32示出了内容长度参数的一个具体实施例；

[0065] 图33示出了大块尺寸域的一个具体实施例；

[0066] 图34是描述连接池的一个具体实施例的消息流程图；

[0067] 图35示出了步骤的一个实施例的细节流程图，这些步骤使用内容长度参数以增加在客户端和服务器之间的连接池的效率；

[0068] 图36是描述步骤的一个实施例的流程图，这些步骤使用内容长度参数以增加在客户端和服务器之间的连接池的效率；

[0069] 图37示出了步骤的一个实施的细节流程图，这些步骤使用大块尺寸域以增加在客户端和服务器之间的连接池的效率；

[0070] 图38是描述步骤的一个实施例的流程图，这些步骤使用大块尺寸域以增加在客户端和服务器之间的连接池的效率；

[0071] 图39是步骤的一个实施例的流程图，这些步骤提供集成的高速缓存功能；

[0072] 图40A是客户端侧加速程序的具体实施例的框图；

[0073] 图40B是设备的实施例的框图，该设备用于提供客户侧加速程序；

[0074] 图41A是方法的具体实施例的步骤图，该方法用于动态提供并自动安装和执行客户端侧的加速程序；

[0075] 图41B是方法的具体实施例的步骤图，该方法用于确定可被加速的应用；

[0076] 图41C是方法的另一个具体实施例的步骤图，该方法通过加速程序执行多个加速技术，以在传输层拦截并使用核心层数据结构；

[0077] 图42A是方法的另一个具体实施例的步骤图，该方法以通过第一程序在客户端自动安装和执行加速程序；

[0078] 图42B是方法的具体实施例的步骤图，该方法用于第一程序和加速程序，以提供虚拟专有网络连接并执行一个或多个加速技术；

[0079] 图43是方法的具体实施例的步骤图，该方法用于重定向到服务器的客户端通信，以旁路已被确定不可用于向服务器传输该通信的中介；

[0080] 图44是方法的具体实施例的步骤图，该方法用于执行传输控制协议缓冲器的客户端侧加速技术；

[0081] 图45A是方法的具体实施例的步骤图，该方法用于执行传输控制协议连接池的客户端侧加速技术；

[0082] 图45B是一组HTTP事务的简图，该事务通过在一个具体实施例中的一个或多个传输层连接的池由多个应用执行；

[0083] 图46是方法的具体实施例的步骤图，该方法用于执行传输控制协议多路复用技术的客户端侧的加速技术；

[0084] 图47是传输层包的内容长度标识符的具体实施例的简图；

[0085] 图48是通过多个大块传输的消息的内容长度标识符的另一个具体实施例的简图；

[0086] 图49A是描述联网的计算机系统的举例性具体实施例的框图，该联网的计算机系统用于加速计算环境到远程客户端的传送；并且

[0087] 图49B是描述方法的步骤的一个具体实施例的流程图，该方法用于加速计算环境到远程客户端的传送。

具体实施方式

[0088] 为了阅读下述各种具体实施例的描述，下述对于说明书的部分以及它们各自内容的描述是有用的：

[0089] —A部分描述了网络环境和计算环境，这对于实施此处描述的具体实施例是有用的；

[0090] —B部分描述了用于传送计算环境到远程用户的系统和方法的具体实施例；

[0091] —C部分描述了用于在客户端和服务器间加速通信的系统和方法的具体实施例；并且

[0092] —D部分描述了说明性的举例性的具体实施例，该实施例使用B部分和C部分描述的系统和方法加速计算环境到远程用户的传送。

[0093] A.网络和计算环境

[0094] 在讨论本系统和方法的具体实施例的细节之前，讨论具体实施例在其中配置的网络和计算环境可能是有帮助的。现在引用图1A，描述了网络环境5。概括来讲，网络环境5包括通过一个或多个网络40，40”、一个或多个客户端10－10”(同样一般指客户端10，或本地机器10)与一个或多个服务器30－30”通信(同样一般指服务器30，或远程机器30”)。在一些

实施例中，客户端10通过设备1250与服务器30通信。

[0095] 虽然图1A示出了在客户端10－10”和服务器30－30”间的网络40和网络40”，客户端10－10”和服务器30－30”可以在同一个网络40之上。网络40和40”可以是相同类型的网络或不同类型的网络。网络40和/或40”可为局域网(LAN)，例如公司内的企业内部互联网、城域网(MAN)、或者广域网(WAN)，例如因特网或万维网。在一个实施例中，网络40”可为专有网络并且网络40可为公网。在一些实施例中，网络40可为专有网络并且网络40”可为公网。在另一个实施例中，网络40和40”可都为专有网络。在一些实施例中，客户端10－10”可位于公司分支机构中，在网络40之上通过WAN连接与位于公司数据中心的服务器30－30”通信。

[0096] 网络40和/或40”可为网络的任何类型和/或形式，可包括任何的下述内容：点对点网络，广播网络，广域网，局域网，电信网络，数据通信网络，计算机网络，ATM(异步传输模式)网络，SONET(同步光网络)网络，SDH(同步数字体系)网络，无线网络和有线网络。网络40和/或40”的拓扑可为总线型，星形，或环形网络拓扑。网络40和/或40”以及网络拓扑可以是任何对于本领域普通技术人员所熟知的、可以支持此处描述的操作的任何这样的网络或网络拓扑。

[0097] 如图1A所示，示出设备1250(此处也指为接口单元1250)在网络40和40”之间。在一些实施例中，设备1250可位于网络40上。例如，公司的分支机构可在分支机构中部署设备1250。在其他实施例中，设备1250可位于网络40”上。例如，设备1250可位于公司的数据中心。在另一个实施例中，多个设备1250可在网络40上部署。在一些实施例中，多个设备1250可在网络40”上部署。在一个实施例中，第一设备1250与第二设备1250’通信。在另一个实施例中，设备1250可为位于相同或不同的网络40，40’上的任一客户端10－10’或服务器30－30”的一部分，作为客户端10－10”。一个或多个设备1250可位于客户端10－10”和服务器30－30”之间的网络或网络通信路径中的任何点上。

[0098] 在一个实施例中，系统可包括多个逻辑分组远程机器30，一个或多个远程机器30可用于以本地机器10名义执行应用。在一些实施例中，远程机器的逻辑分组可指服务器群组(farm)38或群组38。在这些实施例中的一些，远程机器30可为地理上分散的。可将群组38作为单独的实体进行管理。

[0099] 在每个群组38中的远程机器30可为不同种类的。即，一个或多个远程机器30可根据一种类型的操作系统平台(例如位于Redmond，Washington的微软公司制造的WINDOWS NT)操作，而其他远程机器30的一个或多个可根据另一类型的操作系统平台(例如，Unix或Linux)操作。包括在每个群组38内的远程机器30不需要与其群组38内的每个其他远程机器30物理上接近。因此，逻辑分组的作为群组38的远程机器30的组可使用广域网(WAN)连接或城域网(MAN)连接互联。例如，群组38可包括物理上位于不同陆地或陆地的不同区域、国家、州、城市、校园或房间。如果远程机器30使用本地网(LAN)连接或一些形式的直接连接进行连接，在群组38中的远程机器30间的数据传输速度将增加。

[0100] 远程机器30可指服务器、文件服务器、应用服务器、或远程机器。在一些实施例中，远程机器30可有能力或者如应用服务器或者如管理应用服务器那样运行。在一个实施例中，远程机器30可包括活动目录。本地机器10也可称为客户端节点或端点。在一些实施例中，本地机器10有能力如客户端节点以及如应用服务器那样运行，其中作为客户端节点搜索对于应用的访问，作为应用服务器提供对用于其他本地机器10的主机应用的访问。

[0101] 在一个实施例中，本地机器10可直接与群组38中的远程机器30之一直接通信。在另一个实施例中，本地机器10执行程序邻近应用以与群组38内的远程机器30通信。还是在另一个实施例中，远程机器30提供主节点的功能。在一些实施例中，本地机器10通过网络40与群组38中的远程机器30通信。通过网络40，本地机器10可以，例如，请求执行寄存在群组38中的远程机器30、30'、30"和30"'中的各种应用，并接收应用执行结果的输出用于显示。网络40可包括同步或异步连接并可为LAN、MAN(中等区域网络，Medium－Area Network)，或广域网。另外，网络40可包括无线连接，例如红外信道或卫星频带。在一些实施例中，只有主节点提供要求确认和提供地址信息的功能，其中的地址信息与寄存所请求的应用的远程机器30'相关。

[0102] 在一些实施例中，本地主机10与远程机器30"'通信。在这些实施例之一中，远程机器30"'提供网络(Web)服务器的功能。在这些实施例的另一个中，远程主机30"'接收来自本地主机10的请求，将该请求转发到远程机器30并以来自远程机器30对该请求的响应通过本地机器10响应该请求。仍然在这些实施例中的另一个中，远程机器30获得本地机器10可用的应用的列举以及地址信息，该地址信息与远程机器30'相关，该远程机器30'寄存由该应用的列举确定的应用。也在这些实施例的另一个中，远程机器30"'使用网络接口呈现对本地机器10的请求的响应。在一个实施例中，本地机器10直接与远程机器30'通信以访问所确认的应用。在另一个实施例中，本地机器10从远程机器30"'接收应用输出数据，该应用输出数据通过在远程机器30'上的所确认的应用的执行而产生。

[0103] 现在看图1B，示出了在客户端10上用于传送和/或操作计算环境的网络环境。概括来看，服务器30包括用于向一个或多个客户端传送计算环境或应用和数据文件的应用传送系统500。客户端10可包括用于执行应用的计算环境15，该应用使用或处理数据文件。客户端10通过网络40、40'和设备1250与服务器30通信，该客户端可从服务器30请求应用和数据文件，或者设备1250可转发来自客户端10的请求到服务器30。例如，客户端10可能本地没有本地存储或本地可访问的应用和数据文件。响应请求，服务器30可传送应用和数据文件到客户端10。例如，在一个实施例中，服务器30可以应用流传递应用，以操作在客户端10上的计算环境15。

[0104] 图1C和1D是描述通用计算机135的体系结构的具体实施例的框图，该通用计算机135用于客户端计算设备10和服务器计算设备30。如图1C和1D所示，每个计算机135包括中央处理单元102，以及主存储单元122。每个计算机135也包括其他可选择的部件，例如一个或多个输入/输出设备130a－130－b(总的使用参考数字130表示)，以及与中央处理单元102通信的高速缓存140。

[0105] 中央处理单元102是任何逻辑电路，响应并处理从主存储单元122取得的指令。在许多实施例中，中央处理单元由微处理器单元提供，例如由California，Mountain View的Intel公司制造的；由IIIinois，Schaumburg的Motorola公司制造的；由Califirnia的Santa Clara的Transmeta公司制造的处理器Crusoe和Efficeon线；由New York，White Plains的International Business Machines公司制造的处理器线；或者由California，Sunnyvale的Advanced Micro Devices公司制造的处理器线。

[0106] 主存储单元122可为一个或多个存储芯片，这些存储芯片可以存储数据并允许微处理器102直接访问任何存储位置，微处理器可为，例如静态随机存储器(SRAM)、猝发

(Burst)SRAM或同步猝发(SynchBurst)SRAM(BSRAM)，动态随机存储器(DRAM)、快页模式(Fast Page Mode)DRAM(FPM DRAM)、增强(Enhanced)DRAM(EDRAM)、扩展数据输出(Extended Data Output)RAM(EDO RAM)、猝发扩展数据输出DRAM(BEDO DRAM)、增强DRAM(EDAM)、同步DRAM(SDRAM)、JEDEC SRAM、PC100SDRAM、双数率SDRAM(DDR SDRAM)、增强SDRAM(ESDRAM)、同步连接DRAM(SLDRAM)、直接存储总线(DirectRambus)DRAM(DRDRAM)或者铁电RAM(FRAM)。在图1C所示的实施例中，处理器102通过系统总线120(下面将更详细的介绍)与主存储器122通信。图1B描述了计算机系统135的具体实施例，其中处理器通过存储端口直接与主存储器122通信。例如，在图1B中的主存储器122可为DRDRAM。

[0107] 图1C和1D描述的具体实施例中，主处理器102通过次级总线，有时称为“背端”总线，直接与高速缓存存储器140通信。在其他实施例中，主处理器102使用系统总线120与高速缓冲存储器140通信。高速缓冲存储器140通常比主存储器122有更快的响应时间，并通常通过SRAM，BSRAM，或EDRAM提供。

[0108] 在图1C所示的具体实施例中，处理器102通过本地系统总线120与各种I/O设备130通信。各种总线用于连接中央处理单元102和I/O设备130，包括VESA VL总线、ISA总线、EISA总线、微通道结构(MCA)总线、PCI总线、PCI－X总线、PCI－Express总线、或者NuBus。对于这样的实施例，其中I/O设备是视频显示器，处理器102可使用高级图形端口(AGP)与显示器通信。图1D描述了计算机系统135的具体实施例，其中主处理器102通过HyperTransport，Rapid I/O或者InfiniBand与I/O设备130B直接通信。图1D也描述了一个具体实施例，其中本地总线和直接通信是混合的：处理器102使用本地互联总线与I/O设备130a通信，同时与I/O设备130b直接通信。

[0109] 在计算机系统135中将提供I/O设备130的多个种类。输入设备包括键盘、鼠标、轨道垫、轨道球、麦克风、以及绘画板。输出设备包括视频显示器、扬声器、喷墨打印机、激光打印机、及染料升华(dye－sublimation)打印机。I/O设备也可为计算机系统135提供大容量存储器，例如硬盘驱动器，用于接受软盘例如3.5-英寸、5.25－英寸磁盘或ZIP磁盘的软盘驱动器，CD－ROM驱动器、CD－R/RW驱动器、DVD－ROM驱动器、各种格式的磁带驱动器，以及USB存储设备，例如由California，Los Alamitos的TwintechIndustry有限责任公司制造的USB闪存驱动器。

[0110] 在进一步的实施例中，I/O设备130可为在系统总线120和外部通信总线间的桥梁，该外部通信总线可为，例如USB总线、Apple Desktop总线、RS-132串行连接、SCSI总线、FireWire总线、FireWire800总线、Ethernet总线、AppleTalk总线、Gigabit Ethernet总线、异步传输模式总线、HIPPI总线、Super HIPPI总线、SerialPlus总线、SCI/LAMP总线、FibreChannel总线、或者串行连接(Serial Attached)小计算机系统接口总线。

[0111] 图1C和图1D中描述的通用计算机通常在操作系统的控制下操作，该操作系统控制任务和对系统资源的访问的调度。典型的操作系统包括：由Washington，Redmond的Microsoft公司制造的MICROSOFT WINDOWS；由California，Cupertino的Apple计算机公司制造的MacOS；由New York，Armonk的International Business Machines制造的OS/2；以及由Utah，Salt Lake City的Caldera公司的可自由获得的操作系统Linux，以及其他操作系统。

[0112] 对于实施例，其中客户端机器10或服务器30包括移动设备，该设备可为JAVA操作

的移动电话，例如i55sr、i58sr、i85s或者i88s，这些都是由Illinois，Schaumburg的Motorola公司制造的；由Japan，Kyoto的Kyocera制造的6035或7135；或者由Korea，Seoul的Samsung Electronics有限公司制造的i300或者i330。在包括移动设备的其他实施例中，移动设备可为在PalmOS操作系统控制下的个人数字助理(PDA)，正如Tungsten W，VII，VIIx，所有这些都由California，Milpitas的palmOne公司制造。在其他实施例中，客户端113可以是在PocketPC操作系统控制下的个人数字助理(PDA)，诸如iPAQ4155、iPAQ5555、iPAQ1945、iPAQ2215以及iPAQ4255，所有这些都是由California，Palo Alto的Hewlett-Packard公司制造的；由California，Walnut的ViewSonic制造的ViewSonic V36；或者由New York，New York的Toshiba American有限公司制造的Toshiba PocketPC e405。仍然在其他实施例中，移动设备是组合PDA/电话设备，例如Treo180、Treo270、Treo600、Treo650或者Treo700w，所有这些都是由California，Milpitas的palmOne有限公司制造的。仍然在进一步的实施例中，移动设备是在PocketPC操作系统控制下操作的移动电话，该PocketPC操作系统可为，例如由Motorola公司制造的MPx200。一个典型的移动设备可包括如以上在图1C和1D中描述的许多元件，包括处理器102和主存储器104。

[0113] B.用于传送计算环境的系统和方法

[0114] 一个实施例涉及用于从服务器30传送计算环境到位于远程位置的客户端10的远程用户的系统和方法。尽管在该部分中该方法和系统总的称为服务器30，下述方法和系统可使用或者服务器30、网络设备1250、或它们的任意组合。

[0115] 现在看图1E，系统的一个实施例中远程机器30包括图1A中所示所描述的群组38。每个远程机器30包括网络侧接口202和群组侧接口204。远程机器30的网络侧接口202可与一个或多个本地机器10或网络210通信。网络210可为WAN、LAN或者例如因特网或环球网的国际网络。本地机器10可使用网络210建立与远程机器30的连接。

[0116] 远程机器30的群组侧接口204通过通信链路200彼此互联，以便远程机器30可彼此通信。在每个远程机器30上，群组侧接口204与网络侧接口202通信。群组侧接口204也与持久存储器230通信(通过箭头220指出)并且，在一些实施例中，也可与动态存储器240通信。，远程机器30、持久存储器230、以及动态存储器240(当提供时)的结合共同称为群组38。在一些实施中，远程机器30与持久存储器230通信，以及其他远程机器30’与远程机器30通信通信，以访问存储在持久存储器中的信息。

[0117] 持久存储器230可在磁盘、磁盘组、独立磁盘冗余阵列(RAID)、可写紧密磁盘或其他容许数据读出和写入并且如果存储设备没电则保存写入的数据的任何其他设备上物理实现。单个物理设备可提供用于多个持久存储设备的存储，例如，单个物理设备可用于提供用于多于一个群组38的持久存储器230。持久存储器保存与在群组38中的每个远程机器30相关联的静态数据，以及在群组38中的所有远程机器30使用的全局数据。在一个实施例中，持久存储器230可在轻量级目录访问协议(LDAP)数据模型中保存远程机器数据。在另一个实施例中，持久存储器230在ODBC－相容数据库中存储远程机器数据。为了该说明书的目的，术语“静态数据”指不经常改变的数据，例如，数据仅基于每小时、每天、或者每星期改变，或者从不改变的数据。每个远程机器使用持久存储子系统从持久存储器30中读数据和写入数据。

[0118] 持久存储器230存储的数据出于可靠的目的可物理上或逻辑上复制。例如，使用一

组冗余的镜像磁盘可提供物理冗余，每个冗余的镜像磁盘提供数据的副本。在另一个实施例中，使用标准数据技术本身复制数据库，以提供多个数据库的副本。在进一步的实施例中，物理和逻辑复制可并发的使用。

[0119] 动态存储器240(例如，所有记录表的集合)可以各种方式实现。在一个实施例中，动态存储器240被集中化；即，所有的运行时数据存储在群组38中的一个远程机器30的存储器中。该远程机器作为主网络节点操作，当寻找对该运行时数据的访问时，在群组38中的所有其他远程机器30与该远程机器通信。在另一个实施例中，在群组38中的每个远程机器30保存动态存储器240的一个完全备份。此处，每个远程计算机30与每个其他远程机器30通信，以保持其动态存储器240副本的更新。

[0120] 在另一个实施例中，每个远程机器30保持它自己的运行时数据，当寻找以便获得每个其他远程机器30的运行时数据时，与每个其他远程机器30通信。因此，例如，远程机器30试图寻找被本地主机10请求的应用程序，该远程机器30可直接与在群组38中的每个其他远程机器30直接通信，以发现一个或多个保存被请求的应用的远程机器。

[0121] 对于群组38有大量的远程机器30，这些实施例产生的网络通信量变得十分繁重。一个实施例通过在群组中指定远程机器30的子集，代表性的为两个或更多个，作为“收集点”，来减轻繁重的网络通信量。通常，收集点是收集运行时数据的远程机器。每个收集点存储从群组38中的某些其他远程机器30收集的运行时数据。群组38中的每个远程机器30可以作为收集点操作，并因此可以被指定为收集点。在一个实施例中，每个收集点存储全部动态存储器240的副本。在另一个实施例中，每个收集点存储动态存储器240的一部分，例如，保持特定数据类型的运行时数据。由远程机器30存储的数据的类型根据一个或多个标准可被预先确定。例如，远程机器30可保存基于启动顺序的数据的不同类型。可选的，远程机器30存储的数据类型可由管理员使用管理工具140配置。在这些实施例中，动态存储器240分布于群组38中的两个或多个远程机器30中。在另一个实施例中，设备1250通过加速在远程机器30、动态存储器16240和持久存储器230间数据的传送减轻繁重的网络通信量。通过在C部分进一步讨论的任何技术可提供这样的加速。例如，设备1250可用于减轻繁重的网络通信量。

[0122] 没有指定作为收集点的远程机器30知道在群组38中指定作为收集点的远程机器30。没有指定作为收集点的远程机器180当传送和请求运行时数据时可与特定收集点通信。因此，当寻求以访问运行时数据时，由于在群组38中的每个远程机器30与单一的收集点远程机器30通信，而不是与每个其它的远程机器30通信，收集点减轻了网络通信量。

[0123] 每个远程机器30可作为收集点操作用于多于一种类型的数据。例如，对于许可信息和负载信息，远程机器30”可作为收集点操作。在这些实施例中，每个收集点可收集不同类型的运行时数据。例如，为了说明这种情形，远程机器30”’可收集许可信息，而远程机器30”收集负载信息时。

[0124] 在一些实施例中，每个收集点存储在群组38内的所有远程机器30间共享的数据。在这些实施例中，特定数据的类型的每个收集点与群组38中的每个用于该数据类型的其他收集点交换由该收集点收集的数据。因此，一旦完成了这样的数据的交换，每个收集点30”和30处理相同的数据。也在这些实施例中，每个收集点30和30”也保存每个其他收集点最新的对于运行时数据的任何更新。

[0125] 浏览使得本地机器能够查看群组38、远程机器30以及群组中的应用，并访问可用的信息，诸如整个群组38的会话。每个远程机器30包括ICA浏览子系统260，为本地机器10提供浏览的能力。在本地机器10建立任何远程机器30的ICA浏览器子系统260的连接之后，该浏览子系统支持多种本地机器请求。这样的本地机器请求包括：(1)列举在群组中的远程机器的名字，(2)列举群组中公开的应用的名字，(3)将远程机器名字和/或应用名字解析为远程机器地址，这对于本地机器10是有用的。ICA浏览器子系统260也支持由本地机器10运行程序邻近应用所作出的请求，根据请求，该程序邻近应用为本地机器10提供群组38中的对于用户授权的那些应用的视图。ICA浏览器子系统260转发所有上述提及的本地机器请求到远程机器30中的适当的子系统。

[0126] 在一个实施例中，在群组38中有程序邻近子系统270的每个远程机器30，可为本地机器10的用户提供在群组38中的应用的视图。程序邻近子系统270可以限制对于这些本地机器10的用户有权访问的应用的视图。通常，该程序邻近服务器以列表或图标组为用户呈现这些应用。

[0127] 本地机器的两种类型可用到由程序邻近子系统270提供的功能，(1)程序邻近起动的本地机器，其可直接从本地机器桌面访问该功能，以及(2)非程序邻近起动的本地机器(例如，传统本地机器)，通过运行在远程机器上的程序邻近起动的桌面访问该功能。

[0128] 在程序邻近起动的本地机器和程序邻近子系统270间的通信可通过专用虚拟通道发生，该专用虚拟通道在ICA虚拟通道的上部建立。在另一个实施例中，该通信使用XML服务发生。在这些实施例的一个中，程序邻近起动的本地机器和XML子系统通信，该XML子系统，例如下述参考图6描述的XML服务器516，提供远程机器30上的程序邻近功能。

[0129] 在一个实施例中，程序邻近起动的本地机器没有与带有程序邻近子系统270的远程机器的连接。对于该实施例，本地机器10向ICA浏览器子系统260发送请求，以建立到远程机器30的ICA连接，以确定本地机器10可用的应用。本地机器10接着运行需要用户证书的客户端侧对话。该证书通过ICA浏览器子系统260接收，并发送到程序邻近子系统270。在一个实施例中，程序邻近子系统270向用户管理子系统发送证书用于认证。该用户管理子系统可返回一组区别性的名字，这些区别性的名字描述用户所属的列表。一旦认证，程序邻近子系统270建立程序邻近虚拟通道。该通道直到应用过滤结束一直保持开放。在一些实施例中，在C部分描述的加速程序6120响应本地机器10的请求也可被传送到本地机器10。

[0130] 程序邻近子系统270接着从普通应用子系统524请求程序邻近信息，该子系统524与这些说明相联系。普通应用子系统524从持久存储器230获得程序邻近信息。一旦接收到程序邻近信息，程序邻近子系统270格式化程序邻近信息，并且通过程序邻近虚拟通道向本地机器返回程序邻近信息。接着部分ICA连接被关闭。

[0131] 对于另一个实施例，考虑本地机器10的选择群组38的用户，其中程序邻近起动的本地机器与远程机器建立部分ICA连接。群组38的选择从本地机器10发送请求到ICA浏览器子系统260，以便与被选择的群组38中的远程机器30之一建立ICA连接。该ICA浏览器子系统260发送请求到程序邻近子系统270，该子系统选择在群组38中的远程机器30。与远程机器30相关联的地址信息经由ICA浏览器子系统260被确认并返回到本地机器10。本地机器10随后可连接到相应于接收到的地址信息的远程机器30。

[0132] 在另一个实施例中，程序邻近起动的本地机器10程序建立一个ICA连接，邻近虚拟

通道在该ICA连接上被建立并在ICA连接持续的时候保持打开。通过该程序邻近虚拟通道，程序邻近子系统270将程序邻近信息更新推到本地机器10上。这种将更新推入本地机器10可根据此处讨论的任何加速技术被加速。为了获得更新，程序邻近子系统270从普通应用子系统524预定事件，以使得程序邻近子系统270探测所公布的应用的改变。

[0133] 看图1F，描述了系统结构的另一个实施例的框图，该实施例用于通过在web服务目录中的GUI的公布提供对本地机器可用的多个应用程序。该系统包括本地机器10，以及多个远程机器30。一个远程机器30作为内容服务器那样运行。远程机器30'提供web服务器功能。远程机器30"提供用于提供对应用文件访问的功能，并且作为应用服务器或文件服务器工作。本地机器10可从内容服务器30、web服务器30'、应用服务器30"通过网络155下载内容。在一个实施例中，本地机器10可从应用服务器30"通过客户端－应用服务器通信通道1150下载内容(例如应用)。

[0134] 在一个实施例中，本地机器10上的web浏览器11使用加密套接字协议层(SSL)支持到内容服务器30和/或web服务器30'的通信。SSL是由California，Mountain View的Netscape Communication公司开发的安全协议，并且现在是由Internet工程任务组(IEFT)发布的标准。该web浏览器11可选的可使用其他安全协议连接到内容服务器30和/或web服务器30'，例如，但不限于，CA，Los Altos的Terisa Systems开发的安全超文本传输协议(SHTTP)，HTTP over SSL(HTTPS)，Washington，Redmond的Microsoft公司开发的私人通信技术(PCT)，以及IEFT发布的传输层安全(TLS)标准。在其他实施例中，web浏览器11使用没有加密的通信协议，例如超文本传输协议(HTTP)与服务器30通信。

[0135] 另外，本地机器10包括应用客户端13，该应用客户端13用于通过客户端－应用服务器通信通道1150与应用服务器30"建立并交换通信。在一个实施例中，应用客户端13是GUI应用。在一些实施例中，应用客户端13是独立计算结构(ICA)客户端，由Florida，Fort Lauderdale的Citrix Systems有限公司开发，并且也指下述的ICA客户端13。其他应用客户端13的实施例包括远程显示协议(RDP)客户端(由Washington，Redmond的Microsoft公司开发)，X－Windows客户端13、客户端侧显示器、可执行多媒体应用、email、Java、或者.NET代码的解释器或仿真器。更多的，在一个实施例中在应用服务器30"上执行的应用的数据可通过ICA客户端13在本地机器10上显示。在一些实施例中，应用客户端13是例如结合图5详细描述的应用流客户端552的应用客户端。在一些实施例中，依照此处描述的6120的任何实施例，应用客户端13包括加速程序用于在客户端10和服务器30之间加速通信。

[0136] 本地机器10查找web服务目录160以获得网络服务。在一个实施例中，此查找是手动查找。可选的，此查找可为自动查找。web服务目录(web service directory)160也可提供基于服务的视图，例如，白色和黄色的页面，以在web服务目录中查找网络服务(web service)。在另一个实施例中，基于机构化服务名字和用于GUI应用的服务查找，web服务目录160支持层次化的浏览。在一个实施例中，web服务目录160在一个远程机器上独立于内容服务器30(例如目录服务器)执行。在其他实施例中，web服务目录160在多个服务器上执行。

[0137] 在一些实施例中，基于额外的分析或信息，通过在web服务目录160中提供该信息或分析，内容服务器30使得远程机器10选择网络服务。web服务目录160可列出的服务信息实例包括，但不限于，提供服务的商务的名字、服务类型、服务的文本描述、一个或多个服务接入点(SAP)、网络类型、使用的路径(例如TCP或HTTPS)、以及服务质量(QoS)信息。进一步

的，服务信息可是具体的客户端设备类型或用户(例如角色)。因此，服务选择可基于一个或多个上述属性。

[0138] 在一个实施例中，服务类型指本地机器10必须使用以访问网络服务的程序接口。例如，服务类型可规定通过接口描述语言(例如网络服务描述语言(WSDL))所编码的服务。

[0139] 服务接入点，或SAP是用于应用的唯一的地址。SAP使得计算机系统支持在本地机器10和每个远程机器30的多个应用。例如，应用服务器30”可支持电子邮件(例如，e—mail)应用、文件传输应用、和/或GUI应用。在一个实施例中，这些应用每个有SAP，该SAP在应用服务器30”中是唯一的。在一个实施例中，SAP是web或Internet地址(例如，域名系统(DNS)名字、IP/端口、或者统一资源定位器(URL))。因此，在一个实施例中，SAP识别web服务器30’的地址作为用于存储在web服务器30’上的应用的地址的一部分。在一些实施例中，如下面描述的，SAP识别公布服务器插件165的地址，作为用于存储在web服务器30’上的应用的地址的一部分。在一个实施例中，SAP是来自于UDDI登记的“接入点”。

[0140] 为了准备在web服务目录160中发表的项目，内容服务器30包括web发表工具170。在一个实施例中，web发表工具170是软件模块。可选的，web发表工具170是另一个服务器，该服务器位于内容服务器30的外部或内部。

[0141] 在一个实施例中，web服务器30’向本地机器10分发web页。该web服务器30’可为任何能够向本地机器10提供web页的远程机器30。在另一个实施例中，web服务器30’为企业信息门户(例如，公司企业内部网或安全的商户到商户企业外部网)。公司门户为公司网站，用于为用户聚集、个人化并服务应用、数据和内容，同时提供管理工具用于更有效的组织和使用信息。在一些公司中，门户以基于浏览器的对虚拟工作场所的访问代替传统桌面软件。在一些实施例中，设备1250加速网页的提供的传送是使用此处讨论的任何加速技术来加速的。在其他实施例中加速程序6120加速网页的传送。

[0142] web服务器30’也包括发表服务器插件165，使得进行图象用户接口(GUI)应用的发表。更特别的，发表服务器插件165翻译新的web服务入口URL到GUI应用服务，以使得GUI可通过web服务目录160被访问。在一个实施例中，发表服务器插件165是通用网关接口(CGI)脚本，其是设计为接收和返回符合CGI规范的数据的程序。该程序可以任何程序语言，例如C、Perl、Java或者Visual Basic编写。在另一个实施例中，发表服务器插件165是Java服务器页(JSP)。使用发表服务器插件165以方便远程GUI应用的发表，远程机器10因此可以访问网络服务，而不通过程序接口或网页，但通过完全的GUI接口，例如带有Citrix的ICA或者Microsoft的RDP。在一些实施例中，设备1250或者加速程序6120加速上述GUI到客户端的传送是使用此处在C部分讨论的任一加速技术来加速的。

[0143] 应用服务器30”寄载一个或多个本地机器10可用的应用。这样的应用的例子包括字处理程序，例如MICROSOFT WORD以及电子表格程序，例如MICROSOFT EXCEL，两个都是由Washington，Redmond的Microsoft公司制造的，财务报告程序、客户注册程序、程序提供技术支持信息、客户数据库应用、或者应用集合管理器。

[0144] 在一个实施例中，web发表工具170存储关于应用的信息，该web发表工具170在持久大容量存储器225中的web服务目录160中发表该应用。在一个实施例中该信息是用于动态发表服务器插件165的URL。持久大容量存储器225可为磁盘或磁性—光学驱动器。在一个实施例中，持久大容量存储器225是数据库服务器，其存储在一个或多个服务数据库中的与

发表的应用相关的数据。该持久大容量存储器225可为一个位于任一或所有远程机器30内部或外部的部件。

[0145] 在其他实施例中，内容服务器30或者web服务器30’与群组38中的远程机器30通信，以获得应用列举。在这些实施例的一个中，内容服务器30或者web服务器30’与群组38而非与持久大容量存储器225通信。

[0146] 现在看图2，描述了用于选择执行一个应用程序的方法的步骤的一个实施例的流程图。总的来说，以具有可用于本地机器执行的应用的列举的请求(步骤202)，接收与本地机器相关联或与本地机器的用户相关联的证书。相应于接收到的证书，提供本地机器可用的多个应用的列举(步骤204)。接收请求以执行所列举的应用(步骤206)。响应于一个策略，选择执行所列举的应用的预定数量的方法之一，预定数量的方法包括用于所列举的应用的应用流的方法(步骤208)。

[0147] 以具有可用于本地机器执行的应用的列举的请求(步骤202)，接收与本地机器相关联或与本地机器的用户相关联的证书。在一个实施例中，远程机器从本地机器10接收带有证书的可用应用的列举的请求。在另一个实施例中，在远程机器30上的XML服务接收请求和证书，并传送该请求和证书到远程机器30上的管理服务。

[0148] 在一些实施例中，作为web服务器运行的远程机器30从本地机器10接收通信并将该通信转发到远程机器30’。在这些实施例之一中，web服务器转发该通信到远程机器30’上的XML服务。在这些实施例的另一个中，web服务器驻留在本地机器上。在其他的由web服务器将来自于本地机器10的通信传送到远程机器30’的实施例中，响应本地机器10的因特网协议(IP)地址选择该远程机器30。

[0149] 在一些实施例中，本地机器10请求对驻留在远程机器30上的应用的访问。在这些实施例之一中，本地机器10请求由远程机器30对驻留在远程机器30上的应用的执行。在这些实施例的另一个中，本地机器10请求对包括该应用的多个应用文件的获取。

[0150] 在一些实施例中，用户通过由远程机器提供给本地机器10上的图形用户接口向远程机器30提供证书。在其他实施例中，有web服务器功能的远程机器30”’为本地机器10提供图形用户接口。仍然在其他实施例中，由远程机器30传输到本地机器10的收集代理从本地机器10收集证书。在一个实施例中，证书指用户名和密码。在另一个实施例中，证书不限于用户名和密码，但包括而不限于本地机器10的机器ID、操作系统类型、操作系统补丁的存在、安装的网卡的MAC地址、在客户端设备上的数字水印、在活动目录中的成员、病毒扫描器的存在、个人防火墙的存在、HTTP头、浏览器类型、设备类型、网络连接信息，例如因特网协议地址或者地址的范围、远程机器30的机器ID、包括调整用于改变时区的访问请求的日期或时间，以及授权证书。

[0151] 在一些实施例中，与本地机器相关联的证书与本地机器的用户相关联。在这些实施例的一个中，证书是由用户支配的信息。在这些实施例的另一个中，证书是用户鉴别信息。在其他实施例中，与本地机器相关联的证书与网络相关联。在这些实施例的一个中，证书是与本地机器连接到的网络相关联的信息。在这些实施例的另一个中，证书是与收集关于本地机器的信息的网络相关联的信息。仍在其他实施例中，与本地机器相关联的证书是本地机器的特征。

[0152] 响应于接收到的证书，提供本地机器可用的多个应用程序的列举(步骤204)。在一

个实施例中，本地机器10的用户可以在不知道何处去寻找这样的应用和没有连接到这样的应用的所需要的技术信息的情况下，了解由网络40中的远程机器30所寄载的应用程序的可用性。这些可用的应用程序包括用户的“程序邻近”。用于确定用于本地机器的程序邻近的系统包括应用程序(以下称为“程序邻近”应用)、用于存储应用程序部件的存储器、以及用于执行应用程序的处理器。该程序邻近(PN)应用如下所述可安装在本地机器10的存储器中和/或远程机器30上。

[0153] 根据程序邻近应用操作的远程机器30从在群组38中的每个远程机器30中收集应用相关信息。用于每个所寄载的应用的应用相关信息可为多种信息，包括，例如，寄载该应用的远程机器的地址、应用名字、授权使用该应用的用户或用户组、以及在建立连接以运行该应用之前本地机器10需要的最小能力。例如，该应用可流传送视频数据，因此需要的最小能力为本地机器可支持视频数据。其他例子的要求可以是本地机器支持音频数据或有能力处理加密的数据。应用相关信息可存储在数据库中。

[0154] 当本地机器10与网络40连接时，本地机器10的用户提供用户证书。用户证书可包括本地机器10的用户的用户名、用户密码、以及用户被授权的域名。可选的，用户证书可为智能卡、基于时间的令牌、社会保险号、用户密码、个人身份证(PIN)号码、基于对称键值或椭圆曲线密码术的数字证书、用户的生物学特征、可用于获得本地机器10的用户身份并提交用于验证的任何其他方式。响应于本地机器10的远程机器30可基于用户证书验证用户。用户证书可存储于程序邻近应用执行的任何地方。对于本地机器10执行程序邻近应用的实施例，用户证书可存储于本地机器10。对于远程机器30执行程序邻近应用的实施例，用户证书可存储于远程机器30。

[0155] 从用户证书和应用相关信息，远程机器30也可确定远程机器30寄载的哪些应用程序可被本地机器10的用户使用。远程机器30传输代表可用应用程序的信息到本地机器10。该过程消除了本地机器10的用户建立应用连接的需要。另外，远程机器30的管理员可在本地机器10的多个用户中控制应用的访问。

[0156] 在一些实施例中，由远程机器30执行的用户验证足可以授权提供给本地机器10的每个所寄载的应用程序的使用，尽管这样的应用程序可驻留在另一远程机器30’上。因此，当本地机器10启动(例如初始化执行)所寄载的应用之一时，由本地机器10进行用户证书的另外输入对于验证该应用程序的使用是必要的。因此，用户证书的一次输入可服务于确定可用的应用，以及在没有用户其他的手动登录验证处理的情况下授权这样的应用程序的启动。

[0157] 本地机器10或者远程机器30可启动程序邻近应用。结果显示在本地机器10、20的显示屏12、22上。在基于图形窗口的实现中，结果可显示在程序邻近图形窗口中，每个授权的应用程序由该窗口中的图标表示。

[0158] 在一个实施例中，程序邻近应用过滤出本地机器10没有被授权执行的应用程序，并只显示授权的(即可用的)应用程序。在其他实施例中，程序邻近应用可显示授权的和未授权的应用程序。当未授权的应用没有从显示中过滤时，可提供这样的通知，指示这种应用程序是不可用的。可选的，在没有标识哪些应用被授权或未被授权给本地机器10执行的情况下，程序邻近应用可以向本地机器10的用户报告由远程机器30所寄载的所有应用。当本地机器10试图运行这些应用程序之一时，授权可被随后确认。

[0159] 本地机器10从远程机器30请求应用列举。应用列举可使本地机器10的用户观看每个发表的应用的名字。在一个实施例中，无论用户是否有权执行该应用，本地机器10的用户可查看该应用的名字。在另一个实施例中，用户只查看用户被授权执行的那些应用的名字。

[0160] 取决于由本地机器10运行的特定过程，对于应用列举的请求传送到ICA浏览器子系统260、程序邻近子系统270、或者到普通应用子系统524。例如，当本地机器10正运行程序邻近应用时，对于应用列举的请求被发送到在远程机器30上的程序邻近子系统270。当本地机器10通过网页提交请求列举时，该请求传到过普通访问点子系统524。对于这些实施例，当本地机器10需要列举应用时，普通应用子系统524作为用于程序邻近子系统270、ICA浏览器子系统260、以及普通应用子系统的初始化访问点服务。在一些实施例中，当本地机器10通过网页提交列举请求时，寄载web服务器的中间远程机器30接收该请求并向远程机器30'转发该请求。

[0161] 一旦接收到列举请求，普通应用子系统524在持久存储器230中查找所有应用的列举。对于从程序邻近子系统270和普通访问点645子系统接收到的请求，根据本地机器10的用户的证书过滤该程序列表(即，用户只能查看那些用户被授权的应用)。

[0162] 本地机器10也可请求远程机器列举。远程机器列举可使本地机器10的用户查看在群组38中的远程机器的列表。在一个实施例中，远程机器的列表可根据远程机器的类型过滤，如由在该远程机器上的专门的远程机器子系统确定。

[0163] 依赖于本地机器120正在运行的特定过程，对于远程机器列举的请求传给ICA浏览器子系统260或者普通访问点子系统645。例如，当本地机器120通过网页提交远程机器列举请求时，该请求传给普通访问点子系统645。对于这些实施例，普通远程机器子系统300作为用于ICA浏览器子系统260和普通访问点645子系统服务的初始访问点。一旦接收到远程机器列举请求，普通远程机器子系统在持久存储器230中查找所有远程机器列表。可选的，根据远程机器类型过滤远程机器列表。

[0164] 图3A是描述处理的另一个实施例的框图，通过该处理本地机器10初始化程序邻近应用的执行，在此实例中是通过万维网。本地机器10执行web浏览器应用80，例如由California,Mountain View的Netscape Communication有限公司制造的NETSCAPE NAVIGATOR,或者由Washington,Redmond的Microsoft公司制造的MICROSOFT INTERNET EXPLORER或者由California,Mountain View的Mozilla基金公司制造的FIREFOX,或者由Norway,Oslo的Opera Software ASA制造的OPERA,或者California,Cupertino的Apple Computer有限公司制造的SAFARI。

[0165] 本地机器10通过web浏览器80发送请求82以访问相应于驻留在远程机器30上的HTML页的统一资源定位器(URL)。在一些实施例中，通过远程机器30返回84到本地机器10的第一HTML页为寻求识别本地机器10的验证页。

[0166] 仍然看图3A，一旦本地机器10被远程机器30验证，该远程机器准备并向本地机器10传送HTML页88，该页包括程序邻近窗口58，其中显示表示本地机器访问的应用程序的图标57,57'。本地机器10的用户通过点击图标57调用图标57代表的应用的执行。

[0167] 在一些实施例中，远程机器30代表本地机器10的用户执行程序邻近应用。在这些实施例之一中远程机器30是位于本地机器10和远程机器30'之间的中间远程机器。

[0168] 参看图3B，其描述通过在web服务目录中GUI的发表提供本地机器可用的多个应用

程序的步骤的流程图。为了发表一个应用(步骤300),web发表工具170接收用于应用(例如GUI应用)的web服务说明和访问信息。在一个实施例中,web服务说明包括以上描述的服务信息(例如,提供web服务的商家的名称、服务类型、服务的文字说明、以及SAP)。访问信息可包括,例如,发表的应用名字、传输控制协议(TCP)浏览服务器群组地址、以及MetaFrame服务器IP地址。在一些实施例中,访问信息指示使用的地址和用于穿越网络或安全网关或桥设备的标签。

[0169] web发表工具170接着构造服务发表请求以请求web服务的发表(例如GUI应用)(步骤305)。在一个实施例中,服务发表请求包括SAP。在一些实施例中,SAP是包括web服务器30'的网络地址和发表服务器插件165的URL。进一步的,网络地址可为统一资源标识符(URI),其是用于指向web上的对象的名字和地址的类型的通用术语。URL是一种URI。URI的一个实例是web服务器30'的名字(例如,"web服务器")以及用于发表服务器插件165的CGI脚本名字(例如"dynamic-component"("动态部件"))。

[0170] web发表工具170在持久大容量存储器225中存储与SAP相关联的SAP条目(步骤310)。在一些实施例中,web发表工具170也将所发表的应用信息(例如,ICA-published-app-info)与GUI应用相关联。在进一步的实施例中,web发表工具170还在服务发表请求中包括键值,以标识内容服务器30在持久大容量存储器225中存储的SAP条目。例如,键值可以有"123456677"的值。作为SAP标识web服务器30'、发表服务器插件165的CGI脚本名字,以及以上描述的键值的SAP的一个例子是"http://web-server/dynamic-component/?app=123456677"。

[0171] 与以上描述的SAP相关联的SAP条目的实例是"key=123456677,value=ICA-published-app-info"。该键值可以任意长(例如,56位键值,128位键值)。在一个实施例中,键值是用加密的随机数字。键值也为键值持有者提供访问权限。尽管以键值的方式进行了示意,可使用任何方式为存储在持久大容量存储器225中的SAP条目提供安全形式。

[0172] web发表工具170为内容服务器30提供用于在web服务目录160中发表的服务发表请求(步骤315)。更进一步的,在一个实施例中,内容服务器30向本地机器10传输SAP的键值,以请求特定的网络服务,用于随后用于定位SAP条目。在一个实施例中,服务发表请求的发表使得本地机器10的用户访问该服务。在一个实施例中,可使用Florida,Fort Lauderdale的CitrixSystem有限公司开发的NFUSE将GUI应用发表在web服务目录160上。在一些实施例中,GUI应用的发表者使用应用启动和嵌入(ALE)定制web服务目录160上的GUI应用的发表,ALE也是由Citrix System有限公司开发的。ALE使得GUI应用从HTML页启动或者将应用嵌入HTML页中。

[0173] 本地机器10接着从web服务目录160查找服务名字(步骤320)。内容服务器30从本地机器10接收该查找(325)并在web服务目录160中寻找所请求的服务的名字。在另一个实施例中,本地机器10的用户导航web服务目录160,直到定位到一个具体服务的名字,该名字是本地机器10的用户试图去寻找的。虽然以本地机器10示意,但是任何web服务目录客户端(例如,UDDI客户端或者LDAP浏览器)可查询或导航该web服务目录160以发现发表的web服务。

[0174] 一旦定位与接收的查找相关联的SAP,内容服务器30向本地机器10传输该SAP(步骤330)。本地机器10接收该SAP(步骤335)并确定来自于SAP的发表服务器插件165的地址。

本地机器10接着向web服务器30’传输用于GUI应用的请求(步骤340)。在一些实施例中,来自于本地机器10的请求是HTTP请求,从web浏览器11传输到web服务器30’。在其他实施例中,在本地机器10上执行的应用(例如,通用目录浏览器或HTML UI)从内容服务器30接收SAP并作为变元向web浏览器11提供SAP。web浏览器11接着可以向web服务器30’自动传输HTTP请求(对于GUI应用)。沿着前述实例的思路,对web服务器30’的应用请求的具体实例是http://web-server/dynamic-component/?app=123456677)。

[0175] web服务器30’,和更具体的,发表服务器插件165,接收与应用请求相关联的SAP(步骤345),并确定与该请求相关联的SAP项目(步骤350)。在一个实施例中,发表服务器插件165从本地机器10接收请求,并获得与已被存储在持久大容量存储器225中的该请求(作为SAP入口的一部分)相关联的已发表的应用信息。在一些实施例中,发表服务器插件165使用SAP(或SAP的一部分)作为访问存储在持久大容量存储器225中的适当服务项目(例如,已发表的应用信息)的键值,该SAP(或SAP的一部分)是本地机器10从内容服务器30接收的。

[0176] 发表服务器插件165接着构造有已发表的应用信息(例如,应用服务器30”的HTTP地址)的文件或文档,并将该文档传输到本地机器10(步骤355)。该发表服务器插件165构造文件以便该文件的格式与应用客户端13兼容。在一个实施例中,该文件是多用途的因特网邮件扩充协议(MIME)或安全MIME(S/MIME)文件。在另一个实施例中,该文件是包含ICA web客户端被嵌入对象HTML标签的HTML文档。仍然在另一个实施例中,该文件是包含应用流客户端被嵌入对象HTML标签的HTML文档。

[0177] web浏览器11随后接收文件并试着打开该文件。在一个实施例中,如果应用客户端13没有在本地机器10上安装,本地机器10与应用服务器30”通信以下载并安装该应用客户端13。一旦安装了该应用客户端13或者,可选的,如果应用客户端13已经在本地机器10上安装,本地机器10启动该应用客户端13以查看从web服务器30’接收的文档(步骤360)。

[0178] 一旦应用客户端13在本地机器10上安装并被执行,应用服务器30”接着执行该应用并在应用客户端13上显示该应用(步骤365)。如以下结合图7进一步详细描述的,在一个可选的实施例中,该应用服务器30”传输包括该应用的多个应用文件到应用客户端13,以在本地机器10上执行。在另一个实施例中,本地机器10查看该文档(即使在启动该应用客户端13之前)并使用在文档中的信息以便从应用服务器30”获得GUI应用。在此实施例中,GUI应用的显示包括应用客户端30”的安装和执行。更多的,该文档的查看对于本地机器10的用户可以是透明的。例如,本地机器10可从web客户端30’接收文档,并在从应用服务器30”自动请求GUI应用之前解释该文档。

[0179] 因此,该应用客户端13为所发表的应用、桌面、桌面文档、以及应用客户端13支持的任何其他应用提供基于服务的访问。应用客户端13可提供访问的应用的实例包括,但不限于,WINDOWS桌面、WINDOW文件,例如MICROSOFT EXCEL、WORD以及POWERPOINT(其所有都是由Washington,Redmond的Microsoft Corporation开发的),Unix桌面,例如SUN SOLARIS(其由California,Palo Alto的Sun Microsystems开发),以及由North Carolina,Durham的Red Hat有限公司发布的GNU/Linux,以及其他应用。

[0180] 在一些实施例中,响应于策略引擎关于是否以及怎样本地机器可访问一个应用的决定,提供本地机器10可用的多个应用程序的列举(步骤204)。该策略引擎在作出决定之前可收集关于本地机器的信息。现在看图4A,描述了计算机网络的一个实施例,其包括本地机

器10、收集代理404、策略引擎406、策略数据库408、群组38、以及应用服务器30’。在一个实施例中，策略引擎406为远程机器30。在另一实施例中，应用服务器30’是远程机器30’。虽然在图4A所示的实施例中描述只有一个本地机器10、收集代理404、策略引擎406、群组38、以及应用服务器30’，可以理解的是，系统可提供这些部件的每个或任意的多个。

[0181] 总的来看，当本地机器10向策略引擎406传输请求410用于访问应用时，收集代理404与本地机器10通信，检索关于本地机器10的信息，并向策略引擎406传输本地机器信息412。该策略引擎406通过向接收到的信息412应用来自于策略数据库408的策略作出访问控制决定。

[0182] 更详细的，本地机器10向策略引擎406传输对于资源的请求410。在一个实施例中，策略引擎406驻留在应用服务器30’上。在另一个实施例中，该策略引擎406是远程机器30。仍然在另一个实施例中，应用服务器30’从本地机器10接收请求410，并向策略引擎406传输该请求410。还是在另一个实施例中，本地机器向远程机器30”’传输对于资源的请求410，该远程机器30’向策略引擎406传送该请求410。

[0183] 一旦接收到该请求，策略引擎406通过收集代理404启动信息收集。收集代理404收集关于本地机器10的信息，并向策略引擎406传输信息412。

[0184] 在一些实施例中，收集代理404通过网络连接收集和传送信息412。在一些实施例中，收集代理404包括字节码，例如以字节码编程语言JAVA写的应用。在一些实施例中，收集代理404包括至少一个脚本。在那些实施例中，收集代理404通过在本地机器10上运行至少一个脚本收集信息。在一些实施例中，收集代理包括本地机器10上的Active X控件。Active X控件是专门的组件对象模型(COM)对象，该对象实现为一组接口使得其看起来像控件并且类似控件工作。

[0185] 在一个实施例中，策略引擎406向本地机器10传输收集代理404。在另一个实施例中，设备1250可存储或高速缓存收集代理。然后设备1250可向本地机器10传输收集代理。在其他实施例中，设备1250可拦截收集代理404的传输。仍然在另一个实施例中，设备1250可加速收集代理的传送。在一个实施例中，策略引擎406在收集代理404向策略引擎406传输信息412后要求收集代理404的第二次执行。在该实施例中，策略引擎406可能没有足够的信息412以确定本地机器10是否满足特定条件。在其他实施例中，策略引擎406响应于接收到的信息412要求收集代理404的多次执行。

[0186] 在一些实施例中，策略引擎406向收集代理404传输指令以确定收集代理404收集的信息的类型。在那些实施例中，系统管理员可以配置从策略引擎406传输到收集代理404的指令。这提供了对收集的信息的类型的更大的控制。由于对所收集的信息的类型的更大的控制，也扩大了策略引擎406可做出的访问控制决定的类型。收集代理404收集信息412，所述信息412包括但不限于，本地机器10的机器ID、操作系统类型、操作系统补丁的存在、安装的网卡的MAC地址、客户端设备上的数字水印、在活动目录中的成员、病毒扫描器的存在、个人防火墙的存在、HTTP头、浏览器类型、设备类型、例如因特网协议地址或者地址的范围的网络连接信息、远程机器30的机器ID，包括用于改变时区的调整的访问请求的日期或时间、以及授权证书。在一些实施例中，收集代理收集信息以确定使用加速程序6120是否可加速在客户端上的应用。

[0187] 在一些实施例中，设备类型是个人数字助理。在其他实施例中，设备类型是蜂窝电

话。在其他实施例中，设备类型是膝上型电脑。在其他实施例中，设备类型是桌上型电脑。在其他实施例中，设备类型是网络亭子(InternetKiosk)。

[0188] 在一些实施例中数字水印包括数据嵌入。在一些实施例中，水印包括插入到文件中提供关于文件的源信息的数据的模式。在其他实施例中，水印包括数字散列文件以提供篡改检测。在其他实施例中，水印提供关于文件的版权信息。

[0189] 在一些实施例中，网络连接信息涉及带宽能力。在其他实施例中，网络连接信息涉及因特网协议地址。仍然在其他实施例中，网络连接信息包括因特网协议地址。在一个实施例中，网络连接信息包括标识登录代理的网络时区，本地机器向该登录代理提供验证证书。

[0190] 在一些实施例中，授权证书包括多种类型的验证信息，包括但不限于，用户名、客户端名、客户端地址、密码、PIN、语言样本、一次性验证码、生物学数据、数字证书、标签等、及其组合。在接收到收集的信息412后，策略引擎406基于接收到的信息412做出访问控制决定。

[0191] 现在看图4B，其描述策略引擎406的一个实施例的框图，包括第一部件420，该部件包括条件数据库422以及登录代理424；还包括第二部件430，该部件包括策略数据库432。第一部件420将来自于条件数据库422的条件应用到接收到的关于本地机器10的信息，并且确定接收到的信息是否满足该条件。

[0192] 在一些实施例中，条件可能要求本地机器10执行特定的操作系统以满足该条件。在一些实施例中，条件可以要求本地机器10执行特定的操作系统补丁以满足该条件。仍然在另一个实施例中，条件可以要求本地机器10为每个安装的网卡提供MAC地址以满足该条件。在一些实施例中，条件可以要求本地机器10指出在特定激活目录中的成员以满足该条件。在另一个实施例中，条件可以要求本地机器10执行病毒扫描器以满足该条件。在其他实施例中，条件可以要求本地机器10执行个人防火墙以满足该条件。在一些实施例中，条件可以要求本地机器10包括特定的设备类型以满足该条件。在其他实施例中，条件可以要求本地机器10建立特定类型的网络连接以满足该条件。

[0193] 如果接收到的信息满足条件，第一部件420为该条件在数据集426中存储一个标识符。在一个实施例中，如果接收到的信息使条件为真，则该信息满足该条件。例如，条件可以要求安装特定的操作系统。如果本地机器10有该操作系统，则该条件为真并且被满足。在另一个实施例中，如果接收到的信息使一个条件为假，则该信息满足该条件。例如，条件可以针对恶意软件(spyware)是否存在于本地机器10上。如果本地机器10不包括恶意软件(spyware)，则条件为假并且被满足。

[0194] 在一些实施例中，登录代理424驻留在策略引擎406之外。在另一实施例中，登录代理424驻留在策略引擎406中。在一个实施例中，第一部件420包括登录代理424，该代理发起关于本地机器10的信息的收集。在一些实施例中，登录代理424进一步包括数据存储。在这些实施例中，数据存储包括收集代理收集信息的条件。该数据存储与条件数据库422明显不同。

[0195] 在一些实施例中，登录代理424通过执行收集代理404发起信息收集。在其他实施例中，登录代理424通过向本地机器10传输收集代理404发起信息收集，用于本地机器10上的执行。仍然在其他实施例中，登录代理424在接收信息412后发起其他条件信息收集。在一个实施例中，登录代理424也接收信息412。在该实施例中，登录代理424基于接收到的信息

412产生数据集426。在一些实施例中，登录代理424通过将来自数据库422的条件应用到从收集代理404接收的信息而产生数据集426。

[0196] 在另一实施例中，第一部件420包括多个登录代理424。在该实施例中，多个登录代理424中的至少一个驻留在每个网络域，从这些网络域，本地机器10可传输资源请求。在该实施例中，本地机器10向特定的登录代理424传输资源请求。在一些实施例中，登录代理424通过网络域传输到策略引擎406，本地机器10从该网络域访问登录代理424。在一个实施例中，网络域被称为本地机器10的网络区域，本地机器10从该网络域访问登录代理424。

[0197] 条件数据库422存储第一部件420将应用于接收到的信息的条件。策略数据库432存储第二部件430将应用到接收到的数据集426的条件。在一些实施例中，条件数据库422和策略数据库432在ODBC一相容数据库中存储数据。例如，条件数据库422和策略数据库432作为ORACLE数据库被提供，该数据库由Calif，Redwood Shores的Oracle公司制造。在其他实施例中，条件数据库422和策略数据库432可为Microsoft ACCESS数据库或Microsoft SQL server数据库，上述产品由Wash，Redmond的Microsoft公司制造。

[0198] 在第一部件420将接收到信息应用到条件数据库422中的每个条件之后，第一部件向第二部件430传输数据集426。在一个实施例中，第一部件420仅仅向第二部件430传输数据集426。因此，在该实施例中，第二部件430不接收信息412，仅仅接收所满足的条件的标识符。第二部件430接收数据集426，并基于在数据集426中识别的条件，通过应用来自策略数据库432的策略做出访问控制决定。

[0199] 在一个实施例中，策略数据库432存储应用到接收到的信息412的策略。在一个实施例中，通过系统管理员至少部分的指定存储在策略数据库中的策略。在另一实施例中，用户指定存储在策略数据库432中的至少一些策略。用户一指定的一个策略或多个策略作为优先选择被存储。策略数据库432可存储于易失性或非易性存储器中，或者例如分布在多个服务器上。

[0200] 在一个实施例中，只有在一个或多个条件被满足时，策略才允许对资源的访问。在另一实施例中，策略允许对资源的访问但严禁向本地机器10传输资源。另一个策略可能在本地机器10上做出临时连接，其要求在安全网络内访问。在一些实施例中，资源为应用程序并且本地机器10已请求执行应用程序。在这些实施例的一个中，策略允许本地机器10上的应用程序的执行。在这些实施例的另一个中，策略可使得本地机器10接受包括该应用程序的文件流。在这个实施例中，文件流可在独立的环境中被存储和执行。仍然在这些实施例的另一个中，策略可以仅仅允许在远程机器上执行该应用程序，例如应用服务器，以及要求该远程机器向本地机器10传输应用输出数据。

[0201] 现在看图4C，其描述了步骤的一个实施例的流程图，由策略引擎406基于接收到的关于本地机器10的信息使用该步骤做出访问控制决定。一旦接收到所收集的关于本地机器10的信息(步骤450)，策略引擎406基于该信息产生数据集(步骤452)。该数据集包含用于每个被接收到的信息412满足的条件的标识符。该策略引擎406应用策略到数据集426内的每个所识别的条件。该应用产生资源列举，本地机器10可访问该资源列举(步骤454)。策略引擎406接着将该列举提供给本地机器。在一些实施例中，策略引擎406创建超文本标记语言(HTML)文件，用于将所述列举提供给本地机器。

[0202] 现在看图4D，描述了网络的一个实施例，其包括本地机器10、收集代理404、策略引

擎406、策略数据库408、条件数据库410、本地机器20、会话服务器420、所存储的应用数据库422、远程机器30'、第一数据库428、远程机器30"、以及第二数据库432。概括地说，当本地机器10向访问控制服务器406传输用于访问应用程序的请求412时，收集代理404与本地机器10通信，查找关于本地机器10的信息，并向策略引擎406传输本地机器信息414。如上述在图4A和4B中讨论的，策略引擎406做出访问控制决定。本地机器10接收与本地机器10相关联的可用的应用的列举。

[0203] 在一些实施例中，会话服务器420在本地机器10和多个应用会话间建立连接，该多个应用会话与本地机器10相关联。在另一个实施例中，策略引擎406决定本地机器10有权检索包含该应用的多个应用文件，以及有权在本地执行该应用程序。在一些实施例中，策略引擎406决定是否通过向本地机器10传输加速程序6120以加速应用文件的传送。在这些实施例的一个中，远程机器30'存储应用会话数据以及包括该应用程序的多个应用文件。在这些实施例的另一个中，本地机器10与远程机器30'建立应用流会话，该远程机器30'存储应用会话数据和包含该应用程序的多个应用文件。在一些实施例中策略引擎406决定是否通过向本地机器10传输加速程序6120以加速会话流的传送。在一些实施例中，策略引擎406决定是否通过向本地机器10传输加速程序6120以加速数据文件的传送。

[0204] 现在看图4E，描述了步骤的流程图，会话服务器420使用该步骤为本地机器10提供对其相关的应用会话的访问。该会话服务器420从策略引擎406接收关于本地机器10的包括策略引擎406做出的访问控制决定的信息(步骤480)。会话服务器420产生与应用相关联的列举(步骤482)。会话服务器420可将本地机器10连接到相关应用(步骤484)。在一个实施例中，该信息也可包括本地机器信息414。在另一个实施例中，该信息包括在本地执行应用程序的授权。

[0205] 会话服务器420产生相关应用列举(步骤482)。在一些实施例中，策略引擎406识别已经和本地机器10相关联的多个应用会话。在另一实施例中，会话服务器420识别与本地机器10相关联的所存储的应用会话。在这些实施例的一些中，一旦从策略引擎406接收到该信息，会话服务器420自动识别所存储的应用会话。在一个实施例中，存储的应用数据库422驻留在会话服务器420中。在另一实施例中，存储的应用数据库422驻留在策略引擎406中。

[0206] 存储的应用数据库422包含数据，该数据与群组38中的多个远程机器相关联，该远程机器执行应用会话或提供对应用会话数据和应用文件的访问，该应用文件包括应用程序。在一些实施例中，识别与本地机器10相关联的应用会话要求参考与一个或多个远程机器相关联的所存储的数据。在这些实施例的一些中，会话存储器420参考与一个或多个远程机器相关联的所存储的数据。在这些实施例的其他一些实施例中，策略引擎406参考与一个或多个远程机器相关联的所存储的数据。在一些实施例中，第一应用会话在远程机器30'上运行，并且第二应用会话在远程机器30"上运行。在其他实施例中，所有的应用会话在群组38中的单个远程机器30上运行。

[0207] 会话服务器420包括与应用会话相关的信息，该应用会话由用户启动。该会话服务器可存储于易失性或非易失性存储器中，或者例如，通过分布在多个服务器上。表1示出包括在示意性会话服务器420的一部分中的数据：

[0208]

应用会话	应用会话 1	应用会话 2	应用会话 3
用户 ID	用户 1	用户 2	用户 1
客户端 ID	第一客户端		第一客户端

[0209]

客户端地址	172.16.0.50		172.16.0.50
状态	激活	断开	激活
应用	字处理	数据库	电子表格
进程号	1	3	2
服务器	服务器 A	服务器 A	服务器 B
服务器地址	172.16.2.55	172.16.2.55	172.16.2.56

[0210] 表1

[0211] 表1中示出的会话服务器420包括将每个应用会话与启动该应用会话的用户相关联的数据，包括客户端计算机10或20的标识，以及客户端计算机10或20的IP地址，若存在客户端计算机10或20的标识的话，从该客户端计算机10或20的标识，用户一般可连接到远程机器30’。该示出的会话服务器420也包括每个应用会话的状态。应用会话状态可为，例如，“激活”(意味着用户连接到应用会话)，或者“断开”(意味着用户没有连接到该应用会话)。在可选的实施例中，应用会话的状态也可被设置为“执行－断开”(意味着用户已经从应用会话断开，但是在应用会话中的应用仍然在执行)，或者“停止－断开”(意味着用户断开，并且在应用会话中的应用没有正在执行，但是它们在断开之前的即时操作状态已被存储)。会话服务器420进一步存储信息和数据，该信息指示应用116正在每个应用会话内执行，并且该数据指示服务器上每个应用的过程。在远程机器30’是群组38一部分的实施例中，会话服务器420至少是动态存储的一部分，并且该会话服务器也包括在表1的最后两行中的数据，该数据指示在群组38中的远程机器30上的每个应用现在正在/过去正在执行，以及远程机器30的IP地址。在可选的实施例中，会话服务器420包括用于在每个应用会话中的每个应用的状态指示。

[0212] 例如，在表1的例子中，存在三个应用会话，应用会话1、应用会话2以及应用会话3。应用会话1与用户1相关联，用户1正在使用终端1。终端1(terminal one)的IP地址是152.16.2.50。应用会话1的状态是激活，并且在应用会话1中正执行字处理程序。字处理程序在服务器A上作为处理号码1正在被执行。服务器A’的IP地址是152.16.2.55。表1中的应用会话2是断开的应用会话118的实例。应用会话2与用户2相关联，但应用会话2没有连接到本地机器10或者20。应用会话2包括正在IP地址152.16.2.55的服务器A上执行的数据库程序，作为处理代码3。应用会话3是一个用户怎样与在不同远程机器30上操作的应用会话相互作用的实例。应用会话3像应用会话1一样与用户1相关联。应用会话3包括在IP地址152.16.2.56的服务器B上正在被执行的电子数据表程序，为处理代码2，但是包括在应用会话1中的应用会话正在服务器A上执行。

[0213] 在另一个例子中，当通信越过具有诸如服务器B的第二远程机器30”的应用流会话来获取来自第二远程机器30”的第二应用程序用于本地执行时，用户可以通过在诸如服务器A的远程机器30’上执行的应用会话来访问第一应用程序。当不满足第一应用程序的执行所必须的条件时，本地机器10的用户可能拥有获得的在本地执行第二应用程序的授权。

[0214] 在一个实施例中，配置会话服务器420接收断开请求以断开与本地机器10相关联的应用会话，并且响应该请求断开该应用会话。该会话服务器420在从应用会话断开本地机器10后，继续执行应用会话。在该实施例中，会话服务器420访问所存储的应用数据库422，并更新与每个断开的应用会话相关联的数据记录，以便该记录指示与本地机器10相关联的应用会话被断开。

[0215] 在接收到与连接到网络的本地机器相关联的验证信息之后，会话服务器420参考所存储的应用数据库422以识别与本地机器的用户相关联的任何激活的应用会话，如果授权信息与例如本地机器20相关联，则用于识别连接到例如本地机器10的不同本地主机的任何激活的应用会话。在一个实施例中，如果会话服务器420识别任何这样激活的应用会话，该会话服务器420从本地机器10自动断开应用会话，并将该应用会话连接到现在的本地机器20。在一些实施例中，接收到的验证信息将限制本地机器10可重新连接的该应用会话。在其他实施例中，接收到的验证信息授权在本地机器20上的应用程序的执行，其中，本地机器10可能已经拒绝该授权。在这些实施例的一个中，会话服务器420可提供本地机器访问信息以获取用于本地执行的应用程序。

[0216] 接收请求以执行所列举的应用(步骤206)。在一个实施例中，本地机器10的用户从可用应用的接收列举中选择应用用于执行。在另一个实施例中，用户独立于接收到的列举选择应用用于执行。在一些实施例中，用户通过选择应用的图形表示来选择该应用用于执行，该图形表示通过客户端代理呈现在本地机器10上。在其他实施例中，用户通过选择应用的图形表示选择该应用用于执行，该图形表示呈现给web服务器或其他远程机器30”’上的用户。在一些实施例中，设备1250或加速程序6120加速图形表示的传送。在一些实施例中，设备1250高速缓存或存储该图形表示。在一些实施例中，设备可高速缓存或存储任意的和所有的相关应用或部分相关应用。

[0217] 仍然在其他的实施例中，用户请求访问文件。在这些实施例的一个中，请求执行应用以提供给用户对文件的访问。在这些实施例的另一个中，在选择用于执行的文件后，该应用被自动选择用于执行。仍然在这些实施例的另一个中，在对文件的访问请求之前，应用与文件的类型相关联，在识别与请求的文件相关联的文件类型后，启用应用的自动选择。在一些实施例中，可使用设备1250或加速程序6120加速一个或多个文件的传送。在一些实施例中设备1250可高速缓存或存储一些或所有的文件。

[0218] 在一个实施例中，所列举的应用包括多个应用文件。在一些实施例中，该多个应用文件驻留在远程机器30’上。在其它实施例中，该多个应用文件驻留在单独的文件服务器或远程机器30”上。仍然在其它实施例中，该多个应用文件可被传输到远程机器10。仍在其他实施例中，在该多个应用文件中的文件可先于该多个应用文件中的第二个文件传输到本地机器10上之前被执行。在一些实施例中，可使用设备1250或加速程序6120来加速一个或多个应用文件的传送。

[0219] 在一些实施例中，远程机器30从远程机器30’获取关于所列举的应用的信息。在这

些实施例的一个中，远程机器30接收寄载多个应用文件的远程机器30”的标识。在这些实施例的另一个中，远程机器30接收多个应用文件的位置标识，该标识符合通用命名标准(UNC)。仍然在这些实施例的另一中，该标识包括网络位置和用于应用流协议的套接字。

[0220] 在一个实施例中，远程机器30获取包含关于所列举的应用的信息的文件。该文件可包括寄载所列举应用的服务器的的位置的标识。该文件可包括所列举的应用的多个版本的标识。该文件可包括多个应用文件的列举，该多个应用文件包括所列举的应用。该文件可包括压缩文件的标识，该压缩文件包括包含所列举的应用的多个应用文件。该文件可包括必须条件的标识，这些必须条件通过执行所列举的应用的机器来满足。该文件可以包括与所列举的应用相关联的数据文件的列举。该文件可包括脚本的列举，该脚本被用于执行所列举的应用的机器执行。该文件可包括与所列举的应用相关联的登记数据的列举。该文件可包括规则的列举，该规则用于所列举的应用在隔离环境之内执行的一个实施例中。在一个实施例中，该文件可被指为“清单(manifest)”文件。文件包含的信息在下面结合图21将进一步详细描述。

[0221] 在一些实施例中，远程机器30向本地机器10的识别特征应用策略。在这些实施例的一个中，响应所识别的特征，远程机器30识别所列举的应用的版本用于执行。在这些实施例的另一个中，远程机器30做出执行与本地机器10的特征相兼容的所列举的应用的版本的决定。仍然在这些实施例的另一个中，远程机器30做出执行与在本地机器10上执行的操作系统相兼容的所列举的应用的版本的决定。仍在这些实施例的另一个中，远程机器30做出执行与本地机器10上的操作系统的修订级相兼容的所列举的应用的版本的决定。在这些实施例的一个中，远程机器做出执行与本地机器10上的操作系统指定的语言相兼容的所列举的应用的版本的决定。

[0222] 响应策略，选择用于执行所列举的应用的预定数量的方法之一，预定数量的方法包括用于所列举应用的应用流的方法(步骤208)。在一个实施例中，响应策略的应用，对与本地机器10相关联的接收到的证书做出选择。在一些实施例中，通过策略引擎做出选择，例如在上述图4A、4B、4C中描述的策略引擎406。在其他实施例中，接收证书和请求以执行所列举应用的远程机器30进一步包括这样的策略引擎406。

[0223] 在一个实施例中，预定数量的方法包括用于执行在远程机器30’上的所列举应用的方法。在另一个实施例中，预定数量的方法包括用于执行本地机器10上的所列举应用的方法。仍然在另一个实施例中，预定数量的方法包括用于执行第二远程机器30’上的所列举的应用的方法。

[0224] 在一些实施例中，预定数量的方法包括用于通过应用流会话向本地机器10提供所列举的应用的方法。在这些实施例的一个中，本地机器10包括流服务代理，该代理可以启动与远程机器30’的连接，并从远程机器30’接收传输的数据包的流。

[0225] 数据包的流可以包括应用文件，该应用文件包括所列举的应用。在一些实施例中，应用文件包括与应用程序相关联的数据文件。在其他实施例中，应用文件包括执行应用程序所需的可执行文件。仍然在其他实施例中，应用文件包括元数据，该元数据包括关于文件的信息，例如位置、兼容性需求、配置数据、登记数据、用于使用在隔离环境中的执行脚本规则的标识、或者授权需求。在一个实施例中，数据包的流通过诸如TCP/IP包的有效载荷的传输层连接来传输。

[0226] 在一些实施例中，流应用先于包括流应用的多个应用文件中的每个应用文件的传输执行。在这些实施例的一个中，通过本地机器10接收多个应用中的一个应用文件后，开始流应用的执行。在这些实施例的另一个中，通过本地机器10接收到多个应用文件中的可执行应用文件后，开始流应用的执行。仍然在这些实施例的另一个中，本地机器10执行多个应用文件中的第一个接收到的应用文件，并且第一个接收到的应用文件请求对多个应用文件中的第二个应用文件访问。

[0227] 在一个实施例中，流应用在本地机器10上执行，无需永久驻留在本地机器10上。在这个实施例中，流应用可在本地机器10上执行，并且一旦中止流应用，则从本地机器10上移除。在另一个实施例中，在每个应用文件的预先部署的副本存储在本地机器10上之后，流应用在本地机器10上执行。仍然在另一实施例中，在每个应用文件的副本存储在本地机器的隔离环境中之后，流应用在本地机器10上执行。仍然在另一个实施例中，在每个应用文件的副本存储在本地机器10上的高速缓存中之后，流应用在本地机器10上执行。

[0228] 在一个实施例中，响应于本地机器10可接收流应用文件的决定，从预定数量的方法中选择用于将应用流式传输到本地机器10的方法。在另一实施例中，响应于本地机器10授权本地执行流应用文件的决定，从预定数量的方法中选择用于将应用流式传输到本地机器10的方法。

[0229] 在其他实施例中，预定数量的方法包括用于向本地机器10提供应用输出数据的方法，应用输出数据从在远程机器30上所列举应用的执行产生。在这些实施例的一个中，远程机器30是接收用于执行所列举应用的请求的远程机器30。在这些实施例的另一个中，远程机器30是第二远程机器30'，例如文件服务器或应用服务器。在一些实施例中，所列举的应用驻留在执行所列举的应用的本地机器30'上。在其他实施例中，执行所列举的应用的远程机器30'首先从第二远程机器30'通过应用流会话接收所列举的应用。在这些实施例的一个中，远程机器30'包括流服务代理，该流服务代理可以开始与第二远程机器30'的连接，并从第二远程机器30'接收所传输的数据流。在这些实施例的另一个中，可使用负载平衡技术识别第二远程机器30'。仍然在这些实施例的另一中，可基于接近远程机器30'而识别第二远程机器30'。这些实施例将结合下图9更详细的描述。

[0230] 在一些实施例中，远程机器30从预定数量的用于执行所列举的应用的方法中选择一种方法，该方法用于将所列举的应用流传输到远程机器30，在远程机器30上执行所列举的应用，以及向本地机器10提供由执行所列举的应用而产生的应用输出数据。在这些实施例的一个中，远程机器30响应本地机器10的评价来选择方法。在这些实施例的另一个中，响应策略的应用，做出本地机器10的评价的决定。仍然在这些实施例的另一个中，响应接收到的证书的评价做出决定。在一个实施例中，远程机器30接收包括所列举的应用的多个应用文件。在另一个实施例中，远程机器30通过表示层协议，例如ICA表示层协议或远程桌面窗口表示层协议或者X－窗口表示层协议，提供应用输出数据。

[0231] 在一些实施例中，远程机器30也提供与所列举的应用相关联的访问信息，响应所选择的方法产生该访问信息。在这些实施的一个中，访问信息向本地机器10提供选择的方法的指示，该选择的方法用于所列举的应用的执行。在这些实施例的另一个中，访问信息包括所列举的应用的位置的标识，该标识服从通用命名标准(UNC)。仍然在这些实施例的另一个中，访问信息包括会话管理服务器的标识。

[0232] 在一些实施例中，访问信息包括启动票据，该启动票据包括验证信息。在这些实施例的一个中，本地机器10可使用该启动票据验证从远程机器30接收的访问信息。在这些实施例的另一个中，本地机器10可使用该启动票据将其本身验证到寄载所列举的应用的第二远程机器30。仍然在这些实施例的另一个中，响应来自于本地机器10的用于启动票据的请求，远程机器30包括在访问信息中的启动票据。

[0233] 现在看图5，描述了实施例的框图，其中本地机器10请求应用程序的执行并且包括远程机器30的应用传送系统500选择执行应用程序的方法。在一个实施例中，远程机器30从本地机器10接收证书。在另一个实施例中，远程机器30从本地机器10接收用于可用应用列举的请求。

[0234] 在一些实施例中，提供多个、冗余的远程机器30，30’，30”，30”’以及30””。在这些实施例的一个中，例如，存在多个文件服务器、多个会话管理服务器、多个中间集结机器(staging machine)、多个web接口、或者多个访问组控制台。在这些实施例的另一个中，如果远程机器失效，选择冗余远程机器30提供失效机器的功能。在其他实施例中，虽然远程机器30、30’、30”、30”’以及30””、以及网络接口558和访问组控制台520被描述为具有管理服务器、会话管理服务器、中间集结机器、文件服务器、web服务器、以及访问组控制台的单独功能的单独远程机器30，但是可提供具有所有这些机器功能的单个远程机器30。仍然在其他实施例中，远程机器30可提供一个或多个其他远程机器的功能和服务。

[0235] 现在更详细的看图5，该框图描述了应用传送系统500的一个实施例，该系统提供对应用程序的访问。应用传送系统500可包括一个或多个远程机器30、设备1250或它们的任意组合。除了上述结合图1D描述的接口和子系统之外，远程机器30进一步包括管理通信服务514、XML服务516以及管理服务504。管理服务504包括应用管理子系统506、服务管理子系统508、会话管理子系统510以及许可管理子系统512。远程机器30可与访问组控制台520通信。

[0236] 在一个实施例中，管理服务504进一步包括专用远程过程调用子系统、元帧(Metaframe)远程过程调用(MFRPC)子系统522。在一些实施例中，MFRPC子系统522在远程机器30上的子系统间路由通信，例如XML服务516以及管理服务504。在其他实施例中，MFRPC子系统522提供远程过程调用(RPC)接口用于调用管理功能，向管理服务504传送RPC调用，并且向做出调用的子系统返回结果。

[0237] 在一些实施例中，远程机器30与协议引擎，例如图4B中描述的协议引擎406通信。在这些实施例的一个中，远程机器30与驻留在远程机器30’上的协议引擎406通信。在其他实施例中，远程机器30进一步包括协议引擎406。

[0238] 远程机器30可与访问组控制台520通信。访问组控制台520可寄载管理工具到远程机器30或群组38的管理员。在一些实施例中，远程机器30使用XML与访问组控制台520通信。在其他实施例中，远程机器30使用简单对象访问协议(SOAP)与访问组控制台520通信。

[0239] 对于例如图1D和图5中描述的例子，其中远程机器30包括子系统的子集，管理服务504包括多个子系统。在一个实施例中，每个子系统或者为单线程或者为多线程子系统。线程是运行在多任务环境中的独立的可执行流。单线程子系统在一个时间只可以执行一个线程。多线程子系统可以支持多个并发执行线程，例如，多线程子系统可同时执行多个任务。

[0240] 应用管理子系统506管理与多个应用相关联的信息，上述多个应用可以被流式传

输。在一个实施例中，应用管理子系统506控制来自于其他部件的请求，例如用于存储、删除、更新、列举、或解析应用的请求。在另一实施例中，应用管理子系统506控制由与可被流式传输的应用相关的部件发出的请求。这些事件可分为三种类型的事件：应用发表、应用列举以及应用启动，下面将进一步详细的描述每一个。在其他实施例中，应用管理子系统506进一步包括对于应用解决(resolution)、应用公布(publication)以及应用发表(publishing)的支持。在其他实施例中，应用管理子系统506使用数据存储器来存储应用的性质和策略。

[0241] 服务器管理子系统508控制专用于服务器群组配置中的应用流的配置。在一些实施例中，服务器管理子系统508也控制请求获取与群组38的配置相关联的信息的事件。在其他实施例中，服务器管理子系统508控制由与远程机器相关的其他部件发送的事件，该远程机器提供对通过应用流的应用的访问和这些远程机器的性质。在一个实施例中，服务器管理子系统508存储远程机器性质和群组性质。

[0242] 在一些实施例中，远程机器30进一步包括一个或多个公共应用子系统520，该子系统为一个或多个专用应用子系统提供服务。这些远程机器30也可具有一个或多个公共远程机器子系统，该子系统为一个或多个专用远程机器子系统提供服务。在其他实施例中，不提供公共应用子系统，每个专用应用和远程机器子系统实现所有请求的功能性。

[0243] 在一个实施例中，其中远程机器30包括公共应用子系统524，公共应用子系统524管理用于发表的应用的公共性质。在一些实施例中，公共应用子系统524控制事件，该事件请求获取与发表的应用或公共特征相关联的信息。在其他实施例中，公共应用子系统524控制所有由与公共应用及它们的属性相关的其他部件发送的事件。

[0244] 公共应用子系统524可向群组38“发表”应用，这使得通过本地机器10每个应用可用于列举和启动。通常，应用被安装在每个远程机器30上，在远程机器30上期望应用的可用性。在一个实施例中，为了发表应用，管理者运行指定信息的管理工具，这些信息诸如远程机器30寄载应用、在每个远程机器上的可执行文件的名称、用于执行应用(例如，音频、视频、加密等)的本地机器的所请求的能力、以及能够使用该应用的用户列举。此指定的信息被分类为应用专用信息以及公共信息。应用专用信息的实例为：用于访问应用的路径名和用于运行应用的可执行文件的名称。公共信息(例如公共应用数据)包括，例如，应用的用户友好名称(例如“Microsoft WORD2000”)，应用的唯一标识以及应用的用户。

[0245] 应用专用信息和公共信息可被发送到专用应用子系统，该子系统控制在寄载应用的每个远程机器上的应用。专用应用子系统可向持久存储器240写入应用专用信息和公共信息。

[0246] 当被提供时，公共应用子系统524也提供用于管理群组38中的发表的应用的设备。通过公共应用子系统524，管理员可使用例如访问组控制台520的管理工具管理群组38的应用来配置应用组并且产生这些应用组的应用树分级。每个应用组可以被表示为应用树分级中的文件夹。在应用树分级中的每个应用文件夹可包括一个或多个其他应用文件夹和远程机器的指定实例。公共应用子系统524提供建立、移动、重命名、删除、以及列举应用文件夹的功能。

[0247] 在一个实施例中，公共应用子系统524在控制应用列举和应用解决请求中支持应用管理子系统506。在一些实施例中，响应一类数据文件和用于处理该类数据文件的应用之

间的映射，公共应用子系统524提供用于识别用于执行的应用的功能性。在其它实施例中，第二应用子系统提供用于文件类型关联的功能。

[0248] 在一些实施例中，远程机器30进一步包括策略子系统。策略子系统包括用于在本地机器10请求应用的执行时确定是否应用可被流式传输到本地机器10的策略规则。在一些实施例中，策略子系统识别与发表在访问组控制台520中的流应用相关联的服务器访问选项。在这些实施例的一个中，策略子系统服务器使用服务器访问选择作为策略以代替策略规则。

[0249] 会话监控子系统520保持并更新与本地机器10相关联的应用流会话的会话状态，并执行用于应用流会话的许可请求。在一个实施例中，会话管理子系统510监控会话和记录事件，例如应用的启动或者应用流会话的终止。在另一个实施例中，会话监控子系统510接收例如心跳信息的通信，从本地机器10传输到远程机器30。仍然在另一实施例中，会话管理子系统510响应来自管理工具关于会话的询问，例如访问组控制台520内的工具。在一些实施例中，管理服务504进一步包括许可证管理子系统，该子系统与会话管理子系统通信以便为了应用的执行向本地机器提供并保存许可。

[0250] 在一个实施例中，管理服务504提供用于应用列举和应用解决的功能。在一些实施例中，管理服务504也提供用于应用启动、会话监控和跟踪、应用发表以及认证执行的功能。

[0251] 现在看图6，描述了远程机器30的一个实施例的框图，该远程机器30包括提供应用列举的管理服务。管理服务504通过使用与XML服务516交互的web接口可以提供应用列举。在一个实施例中，XML服务516列举用于本地机器10的用户的应用。在另一实施例中，XML服务516实现以上描述的ICA浏览器子系统以及程序邻近子系统的功能。XML服务516可以与管理通信服务514相互作用。在一个实施例中，XML服务516使用管理通信服务514产生应用列举请求。应用列举请求可以包括客户类型，该客户类型指示当执行所列举的应用时所使用的执行方法。应用列举请求被发送到公共应用子系统524。在一个实施例中，公共应用子系统524返回与应用列举请求的客户类型相关联的应用的列举。在另一个实施例中，公共应用子系统524向本地机器10的用户返回可用的应用列举，响应策略的应用向与本地机器10相关联的证书返回该选择的列举。在此实施例中，如上述结合图4B所描述的，策略引擎406可向由收集代理404收集的证书应用该策略。仍然在另一个实施例中，直到请求该列举的应用的执行，返回该应用的列举并推迟到本地机器10的策略的应用。

[0252] 管理服务504可提供用于识别寄载应用的第二远程机器30的应用解决服务。在一个实施例中，第二远程机器30’为文件服务器或应用服务器。在一些实施例中，管理服务504查阅文件，该文件包括用于寄载应用的多个远程机器30的标识符。在一个实施例中，管理服务504响应来自本地机器10的对于执行应用的请求来提供应用解决服务。在另一个实施例中，管理服务504识别第二远程机器30’，该第二远程机器30’能实现执行应用的、与第一远程机器30的不同方法。在一些实施例中，管理服务504识别第一远程机器30’，该第一远程机器30’可以向本地机器10流式传输应用程序，并且第二远程机器30’可以执行应用程序，并向本地机器10提供响应应用程序的执行而产生的应用输出数据。

[0253] 在一个实施例中，web接口向XML服务516传输应用解决请求。在另一个实施例中，XML服务516接收应用解决请求并向MFRPC子系统522传输该请求。

[0254] 在一个实施例中，MFRPC子系统522识别包括带有接收到的应用解决请求的客户端

类型。在另一实施例中，MFRPC子系统向该客户端类型应用策略并确定将该应用“流式传输”到本地机器10。在该实施例中，MFRPC子系统522可向应用管理子系统506转发该应用解决请求。在一个实施例中，一旦接收到来自于MFRPC子系统522的应用解决请求，应用管理子系统506可为本地机器10识别远程机器30””，该远程机器30””作为会话管理服务器562运行。在一些实施例中，本地机器向会话管理服务器562传输心跳消息。在另一个实施例中，应用管理子系统506可识别寄载多个应用文件的远程机器30’，这些应用文件包括被流式传输到本地机器10的应用。

[0255] 在一些实施例中，应用管理子系统506使用文件来列举寄载多个应用文件的多个远程机器以识别该远程机器30’。在其它实施例中，应用管理子系统506识别远程机器30’，该远程机器30’的IP地址相似于本地机器10的IP地址。仍然在其它实施例中，应用管理子系统506识别远程机器30’，该远程机器30’的IP地址在本地机器10可访问的IP地址范围内。

[0256] 仍然在另一个实施例中，MFRPC子系统522向客户端类型应用策略，并确定可在远程机器30’上执行应用，远程机器30’向本地机器10传输由应用的执行而产生的应用输出数据。在该实施例中，MFRPC子系统522可向公共应用子系统524转发该应用解决请求，以获取用于本地机器30’的主机地址的标识符。在一个实施例中，识别的远程机器30’可使用例如ICA或RDP或X-Windows的表示层协议向本地机器传输应用输出数据。在一些实施例中，远程机器30’通过应用流会话从第二远程机器30’接收应用。

[0257] 在一个实施例中，一旦完成应用列举和应用解析，访问信息被传输到本地机器10，包括用于列举应用的执行方法的识别以及寄载所列举应用的远程机器30’的标识符。在一个实施例中，此处管理服务504确定所列举的应用将会在本地机器10上执行，web接口建立并向本地机器10传输包括关于所列举的应用的名称解析信息的文件。在一些实施例中，文件可使用扩展名“.rad”标识。本地机器10可响应于接收到的文件的内容来执行所列举的应用。表2描述包含在文件中的信息的一个实施例：

[0258]

域	描述	源
UNC 路径	指向文件服务器上的容器主清单文件	XML 服务
初始程序	程序从容器启动	XML 服务
命令行	用于使用 FTA 启动文件	XML 服务
web 服务器 URL	用于从 RADE 客户端到 WI 的消息	WI 配置
群组 ID	应用属于的群组 - 心跳消息所需	WI 配置
启动票据	应用流客户端使用启动票据获得授权程序执行的许可	XML/IMA
ICA 后退启动信息	如果后退是被允许的，嵌入 ICA 文件用于后退	XML 服务

[0259] 表2

[0260] 如表2所示，文件也可包括启动票据，本地机器在执行应用中使用该启动票据。在一些实施例中，启动票据在预定时间周期之后终止。在一个实施例中，本地机器向寄载将被

执行的列举的应用的远程机器提供启动票据。本地机器的用户使用启动票据授权对列举的应用的访问，可有助于禁止用户重新使用文件或者产生文件的未授权版本以对应用进行不适当的访问。在一个实施例中，启动票据包括大量随机产生的数字。

[0261] 结合图2所述，当接收到与本地机器10或本地机器10的用户相关联的证书，开始用于选择应用程序的执行方法的方法(步骤202)，并且响应接收到的证书，提供对本地机器10有用的多个应用程序的列举(步骤204)。接收请求以执行列举的应用(步骤206)，并且响应策略，选择用于执行列举应用的预定数量的方法的一个，预定数量的方法包括用于列举应用的应用流的方法(步骤208)。

[0262] 现在参见图7，描述访问包括应用程序的多个文件所采用的多个步骤的一个实施例的流程图。本地机器执行本地机器的启动前分析(步骤210)。在一个实施例中，在获取和执行包括应用程序的多个应用文件之前，本地机器10执行启动前分析。在另一个实施例中，响应所接收到的指示启动前分析是用于授权访问包括应用程序的多个应用文件的必要条件，本地机器10执行启动前分析。

[0263] 在一些实施例中，本地机器10从远程机器30接收与多个应用文件相关联的访问信息。在这些实施例的一个中，访问信息包括寄载多个应用文件的远程机器30'的位置的标识。在这些实施例的另一个中，本地机器10接收包括一个或多个版本的应用程序的多个应用的标识。仍在这些实施例的另一个中，本地机器10接收包括一个或多个应用程序的多个应用文件的标识。在其它实施例中，本地机器10接收本地机器10可用的用于获取和执行的应用程序的列举。在这些实施例的一个中，由本地机器10的评价产生列举。仍然在其他实施例中，本地机器10响应于包括应用程序的多个应用文件的所获取的标识来获取至少一个特征。

[0264] 在一些实施例中，访问信息包括启动票据，该启动票据可以授权本地机器访问多个应用文件。在这些实施例的一个中，响应于本地机器10的评价，向本地机器10提供启动票据。在这些实施例的另一个中，在本地机器10进行本地机器10的启动前分析之后向本地机器10提供启动票据。

[0265] 在其他实施例中，本地机器10获取执行多个应用文件所必需的至少一个特征。在这些实施例的一个中，访问信息包括至少一个特征。在这些实施例的另一个中，访问信息指示用于通过远程机器10获取的文件的位置，该文件列出至少一个特征。仍然在这些实施例的另一个中，所列举的至少一个特征的该文件进一步包括多个应用文件的列举和寄载该多个应用文件的远程机器30的标识。

[0266] 本地机器10确定本地机器上的至少一个特征的存在。在一个实施例中，本地机器10将该确定作为启动前分析的一部分。在另一个实施例中，本地机器10确定本地机器10是否有至少一个特征。

[0267] 在一个实施例中，确定本地机器10上至少一个特征的存在包括确定设备驱动是否被安装在本地机器上。在另一个实施例中，确定本地机器10上至少一个特征的存在包括确定操作系统是否安装在本地机器10上。仍然在另一个实施例中，确定本地机器10上至少一个特征的存在包括确定特定的操作系统是否安装在本地机器10上。也在另一个实施例中，确定本地机器10上至少一个特征的存在包括确定特定修订版的操作系统是否安装在本地机器10上。

[0268] 在一些实施例中，确定本地机器10上至少一个特征的存在包括确定本地机器10是否已经获得授权以执行所列举的应用。在这些实施例的一个中，本地机器10作出确定，关于本地机器10是否已接收到许可以执行所列举的应用。在这些实施例的另一个中，本地机器10作出确定，关于本地机器10是否已接收到许可通过应用流会话接收包括所列举应用的多个应用文件。在其它实施例中，确定本地机器上至少一个特征的存在包括确定本地机器10是否有足够的可用带宽来获取和执行所列举的应用。

[0269] 在一些实施例中，确定本地机器10上至少一个特征的存在包括本地机器10上脚本的执行。在其他实施例中，确定本地机器10上至少一个特征的存在包括本地机器10上软件的安装。仍然在其他实施例中，确定本地机器10上至少一个特征的存在包括本地机器10上登记的修改。仍在其他实施例中，确定本地机器10上至少一个特征的存在包括向本地机器10传输收集代理404，用于在本地机器10上执行以收集与本地机器10相关联的证书。

[0270] 本地机器10从远程机器30请求用于执行多个应用文件的授权，该请求包括启动票据(步骤212)。在一些实施例中，本地机器10响应于确定至少一个特征存在于本地机器10上而作出请求。在这些实施例的一个中，本地机器10确定在本地机器10上存在多个特征，该多个特征与所列举的应用相关联，并且响应于请求被接收以执行所列举的应用。在这些实施例的另一个中，是否本地机器10接收了用于执行所列举的应用文件的授权的指示，依赖于本地机器10上至少一个特征的存在。在一个实施例中，本地机器10接收应用列举，请求所列举的应用的执行，并且接收访问信息，该访问信息包括至少一个特征和启动票据，该启动票据在确定本地机器10上至少一个特征的存在时授权所列举的应用的执行。

[0271] 在一个实施例中，本地机器10从远程机器30接收授权执行多个应用文件的许可。在一些实施例中，该许可授权执行特定的时间周期。在这些实施例的一个中，许可请求传输心跳信息以维持授权用于多个应用文件的执行。

[0272] 在另一个实施例中，本地机器10从远程机器30接收许可和与监控多个应用文件的执行的远程机器30相关联的标识符。如上图5中描述的，在一些实施例中，远程机器为会话管理服务器562。在这些实施例的一个中，会话管理服务器562包括会话管理子系统510，该子系统监控与本地机器10相关联的会话。在其它实施例中，单独的远程机器30""是会话管理服务器562。

[0273] 本地机器10接收并执行多个应用文件(步骤214)。在一个实施例中，本地机器10通过应用流会话接收多个应用文件。在另一个实施例中，本地机器10在本地机器10上的隔离环境中存储多个应用文件。仍然在另一个实施例中，本地机器10在接收到多个应用文件的第二个之前，执行多个应用文件的一个。在一些实施例中，远程机器向多个远程机器传输多个应用文件，多个远程机器中的每个已经与远程机器建立了单独的应用流会话。

[0274] 在一些实施例中，本地机器10在高速缓存中存储多个应用文件，并延迟应用文件的执行。在这些实施例的一个中，本地机器10以在定义前的时段接收执行应用文件的授权。在这些实施例的另一个中，当本地机器10缺少对网络的访问时，本地机器10在定义以前的时段接收授权以执行应用文件。在其他实施例中，本地机器在高速缓存中存储多个应用文件。在这些实施例的一个中，应用流客户端552建立内部应用流会话以从高速缓存获取多个应用文件。在这些实施例的另一个中，当本地机器10缺少对网络的访问时，本地机器10在定义以前的时段接收授权以执行应用文件。

[0275] 本地机器10向远程机器传输至少一个心跳消息(步骤216)。在一些实施例中,本地机器10传输至少一个心跳消息以保留授权来执行包括列举的应用的多个应用文件。在其他实施例中,本地机器10传输至少一个心跳消息以保留授权来在多个应用文件中获取应用文件。仍然在其他实施例中,本地机器10接收在确认前时段授权多个应用文件的执行的许可。

[0276] 在一些实施例中,本地机器10向第二远程机器30""传输心跳消息。在这些实施例的一个中,该第二远程机器30""可包括会话管理服务器562,该会话管理服务器562监控多个应用文件的获取和执行。在这些实施例的另一个中,该第二远程机器30""响应于该传输的心跳消息,可以更新用于授权多个应用文件的执行的许可。仍然在这些实施例的另一个中,第二远程机器30""响应于传输的心跳消息,可以向本地机器10传输命令。

[0277] 现在回看图5,本地机器10可包括应用流客户端552、流服务器554以及隔离环境556。应用流客户端552可为可执行程序。在一些实施例中,应用流客户端552可以启动另一个可执行程序。在其他实施例中,应用流客户端552可开始流服务554。在这些实施例的一个中,应用流客户端552可向流服务554提供与执行应用程序相关联的参数。在这些实施例的另一个中,应用流客户端552可使用远程过程调用来开始流服务554。

[0278] 在一个实施例中,本地机器10请求应用程序的执行并从远程机器30接收关于执行的访问信息。在另一个实施例中,应用流客户端552接收访问信息。仍然在另一个实施例中,应用流客户端552向流服务554提供访问信息。也在另一个实施例中,访问信息包括文件位置的标识,文件与包括应用程序的多个应用文件相关联。

[0279] 在一个实施例中,流服务554获取与多个应用文件相关联的文件。在一些实施例中,获取的文件包括多个应用文件的位置的标识。在这些实施例的一个中,流服务554获取多个应用文件。在这些实施例的另一个中,流服务554执行本地机器10上的获取的多个应用文件。在其他实施例中,流服务554向远程机器10传输心跳消息以维持授权来获取和执行多个应用文件。

[0280] 在一些实施例中,获取的文件包括多于一个的多个应用文件的位置的标识,每个多个应用文件包括不同的应用程序。在这些实施例的一个中,流服务554获取包括与本地机器10兼容的应用程序的多个应用文件。在这些实施例的另一个中,响应于本地机器10的评价,流服务554接收授权以获取特定的多个应用文件。

[0281] 在一些实施例中,多个应用文件被压缩并存储在文件服务器上档案文件中,例如CAB、ZIP、SIT、TAR、JAR或其他档案文件中。在一个实施例中,存储在档案文件中的多个应用文件包括应用程序。在另一个实施例中,存储在档案文件中的多个应用文件每个包括不同版本的应用程序。仍然在另一个实施例中,存储在档案文件中的多个应用文件每个包括不同的应用程序。在一些实施例中,档案文件包括元数据,该元数据与该多个应用文件中的每个文件相关联。在这些实施例的一个中,流服务554响应于包括的元数据产生目录结构。如下面结合附图12更详细描述,可使用元数据来满足应用程序对目录列举的请求。

[0282] 在一个实施例中,流服务554解压缩档案文件以获得该多个应用文件。在另一个实施例中,在从该多个应用文件获取文件之前,流服务554确定该多个应用文件中的文件的本地副本是否存在于本地机器10的高速缓存中。仍然在另一个实施例中,文件系统过滤器驱动器564确定本地副本是否存在于高速缓存中。在一些实施例中,在该多个应用文件中获取文件之前,流服务554修改登记入口。

[0283] 在一些实施例中，流服务554在本地机器10上的高速缓存中存储多个应用文件。在这些实施例的一个中，一旦接收到对该多个应用文件高速缓存的请求，流服务554可提供高速缓存多个应用文件的功能。在这些实施例的另一个中，流服务554可提供保护本地机器10上的高速缓存的功能。在这些实施例的另一个中，流服务554可使用算法调整高速缓存的尺寸和位置。

[0284] 在一些实施例中，流服务554在本地机器10上生成隔离环境。在这些实施例的一个中，流服务554使用隔离环境应用编程接口来生成隔离环境556。在这些实施例的另一个中，流服务554在该隔离环境556中存储该多个应用文件。仍然在这些实施例的另一个中，流服务554执行在隔离环境中的该多个应用文件中的文件。仍然在这些实施例的另一个中，流服务554在隔离环境中执行应用程序。

[0285] 对于接收授权以在本地机器10上执行应用的实施例，该应用的执行可发生在隔离环境556中。在一些实施例中，在执行应用前，包括该应用的多个应用文件存储在本地机器10上。在其他实施例中，在执行该应用之前，该多个应用文件的子集存储在本地机器10上。仍然在其他实施例中，该多个应用文件没有驻留在隔离环境556中。仍然在其他实施例中，该多个应用文件的子集没有驻留在本地机器10上。无论该多个应用文件的子集或者多个应用文件中的每个应用文件是否驻留在本地机器10上或隔离环境556中，在一些实施例中该多个应用文件中的应用文件可在隔离环境556中被执行。

[0286] 隔离环境556可由核心系统组成，该核心系统可以提供Fi le System Virtual izat ion(文件系统虚拟)、Registry System Virtual izat ion(登记系统虚拟)、以及Named Object Virtualization(命名对象虚拟)，不需要请求对应用源代码作任何改变而减少应用兼容性问题。隔离环境556可使用用于登记和命名对象虚拟的用户模式中和使用用于文件系统虚拟的文件系统过滤器驱动器的内核中二者的钩子(hooking)，重定向应用资源请求。下述是对隔离环境556的一些实施例的描述。

[0287] 现在看图8A，示出了在操作系统100控制下运行的计算机的一个实施例，该计算机系统100已经减少了应用兼容性和应用群集度问题。操作系统100通过它的系统层108使得各种原生资源到应用程序112、114。系统层108包含的资源的范围(view)可称为“系统范围”。为了避免应用程序112、114对原生资源102、104、106、107访问的冲突，提供隔离环境200。如图8A所示，隔离环境200包括应用隔离层200以及用户隔离层240。概念上的，隔离环境200通过应用隔离层220提供带有唯一范围(view)的原生资源，例如文件系统102、登记104、对象106、以及窗口名称107。每个隔离层修改提供给应用的原生资源的范围(view)。通过层提供的原生资源的修改范围指层的“隔离范围”。如图8A所示，应用隔离层包括两个应用隔离范围222、224。范围222表示提供给应用112的原生资源的范围，并且范围224表示提供给应用114的原生资源的范围。因此，在图8A所示的实施例中，当提供带有其特定的另一个文件系统102”的APP2 114时，提供带有特定范围的文件系统102’的APP1 112。在一些实施例中，应用隔离层220为在操作系统100顶部上执行的每个单独的应用程序提供原生资源102、104、106、107的特定范围。在其他实施例中，应用程序112、114可被分为集合为集并且，在这些实施例中，应用隔离层220为每个应用程序集提供特定范围的原生资源。冲突应用程序放入单独的组中以增强应用的兼容性和群集度。仍在进一步的实施例中，属于集合的应用通过管理员配置。在一些实施例中，“通过”隔离范围可被定义恰恰对应于系统范围。换句

话，在通过隔离范围执行的应用直接在系统范围内操作。

[0288] 在一些实施例中，应用隔离范围进一步被分为分层的子范围。主要的子范围包括基本应用隔离范围，并且附加子范围包括对多个应用的执行实例可能是可见的该范围的各种修改。例如，子范围可包括对该范围的修改，该范围具体化了对应用的补丁标准的改变或者附加特征安装或者移动。在一些实施例中，配置附加子范围的子集，该附加子范围的子集对执行应用的实施可见。在一些实施例中，无论代表是哪个应用正在执行的用户，用于执行应用的所有实例的可见子范围的集合是相同的。在另一个中，对于执行应用的不同用户，可见子范围的集可改变。仍然在其他实施例中，可定义子范围的各种子集，并且用户有机会选择使用哪个子集。在一些实施例中，当子集不再需要时可被丢弃。在一些实施例中，包括在子范围的集合中的修改可一起合并以形成单个子范围。

[0289] 现在看图8B，描述了具有减少的应用兼容性和应用群集度问题的多用户计算机。该多用户计算机包括在系统层108中的原生资源102、104、106、107、以及刚刚讨论的隔离环境200。该应用隔离层220如上讨论的运行，提供带有修改范围的原生资源的应用或应用组。用户隔离层240，概念上的，提供带有原生资源的范围的应用程序112、114，该原生资源进一步基于用户的用户标识改变，代表该用户正在执行应用。如图8B所示，可认为用户隔离层240包括许多的用户隔离范围242’、242”、242’”、242’”’、242’””、242”””(通称242)。用户隔离范围242提供原生资源的应用特定范围的用户特定范围。例如，提供带有文件系统范围102’(a)的代表用户‘a’的用户会话110中执行的APP1 112，该文件系统范围102’(a)通过用户隔离范围242’和应用隔离范围222被改变和修改。

[0290] 另一种方式，用户隔离层240通过“层化”用户特定范围修改来改变用于每个单独用户的原生资源范围，该用户特定范围修改由用户隔离范围242’在应用特定范围修改“之上”提供，该应用特定范围修改由应用隔离范围222提供，其由系统层提供的原生资源的系统宽范围“之上分层”。例如，当APP1112的第一实施例在登记数据库104中访问入口时，考虑对第一用户会话和应用104’(a)为专用的登记数据库的范围。如果在登记104’(a)的用户特定范围内发现了请求的登记键值(registry key)，该登记键值返回到App1 112。如果没有，考虑应用104’专用的登记数据库的范围。如果请求的登记键值在登记104’的应用特定范围内被找到，则该登记键值返回到APP1 112。如果没有，则存储在系统层108中的登记数据库104中的该登记键值(例如，原生登记键值)返回给APP1 112。

[0291] 在一些实施例中，用户隔离层240为每个单独的用户提供隔离范围。在其他实施例中，用户隔离层240为用户群提供隔离范围，可通过在组织内的角色定义该用户群，或者由管理员确定。仍然在其他实施例中，不提供用户隔离层240。在这些实施例中，通过应用隔离层220提供应用程序可见的原生资源范围。虽然通过多个用户关于支持应用程序并发执行的多用户计算机来描述隔离环境200，但是隔离环境200也可用于单用户计算机上以处理应用兼容性和群集度的问题，这些问题来源于在同一计算机系统上的不同的用户对应用程序的顺序执行，并且这些问题来源于相同的用户的不兼容问题的安装和执行。

[0292] 在一些实施例中，用户隔离范围进一步被分为子范围。通过用户隔离范围对呈现给在该范围内执行的应用的范围的修改是包括在范围内的每个子范围中的修改的集合。每个子范围层叠在彼此之上，并且在集合中在更高子范围中的对资源的范围修改超越在较低层中的对相同资源的修改。

[0293] 在这些实施例的一些中，一个或多个这些子范围可包括对用户专用的范围的修改。在这些实施例的一些中，一个或多个子范围可包括对用户集专用的范围的修改，其可以由系统管理员限定或者被限定为操作系统中的用户的群。在这些实施例的一些中，这些子范围的其中之一可包括对特定登录会话专用的范围的修改，并且因此当会话结束时该子范围被丢弃。在这些实施例的一些中，通过与用户隔离范围相关联的应用实例改变的原生资源总是影响这些子范围之一，并且在其他实施例中，依赖于特定资源的改变，这些改变可影响不同的子范围。

[0294] 以上讨论的概念上结构允许代表用户正在执行的应用将被呈现为集合的、或统一的、虚拟的原生资源范围，对应用和用户的联合是特定的。该集合的范围可指“虚拟范围”。代表用户正在执行的应用实例呈现为带有单一范围的原生资源，该原生资源影响原生资源的所有运转着的虚拟实例。概念上的该集合的范围首先由在系统范围内的操作系统提供的原生资源的集合组成，覆盖包含在能够执行应用的应用隔离范围内的修改，进一步覆盖包括在能够执行代表用户的应用的用户隔离范围内的修改。除了操作系统许可对特定的用户或应用拒绝访问以外，在系统范围内的该原生资源的特点对在系统上所有用户和应用是共同的。对包含在应用隔离范围内的资源范围的修改的特点是对应用的所有实例是共同的，应用与该应用隔离范围相关联。对包含在用户隔离范围内的资源范围的修改的特点是对与可应用的应用隔离范围相关联的所有应用是共同的，该应用隔离范围代表与用户隔离范围相关联的用户正在执行。

[0295] 该概念可延伸到子范围；对在用户子范围内包含的资源范围的修改对所有应用都是公共的，上述应用与可应用的隔离子范围相关联，隔离子范围代表用户或用户群正执行，并与用户隔离子范围相关联。通过所有描述，应该理解的是任何时候存在于此处的通常所指的“范围”也指子范围。

[0296] 当应用请求原生资源的列举时，例如文件系统或登记数据库的一部分，通过首先列举原生资源的“系统范围”实例来构建虚拟列举，即，若有的话，存在于系统层中的实例。接着，请求的资源的“应用范围”实例，即在该适当的应用隔离范围内存在的实例，若存在，则被列举。在应用隔离范围内存在的任何列举的资源被加入到范围内。如果列举的资源已经存在于范围内（最好还是因为它已经存在于系统范围内），它被应用隔离范围内存在的资源的实例所替代。相似的，请求资源的“用户范围”实例，即在适当的用户隔离范围内发现的实例，若存在，则被列举。再一次的，在用户隔离范围内存在的任何列举的资源被加入到范围内。如果原生资源已经存在于范围内（由于它已经呈现在系统中或者在适当的应用隔离范围内），它被在用户隔离范围内存在的资源的实例所替代。以此方式，原生资源的任何列举将会正确的反映列举的原生资源的虚拟。概念上的，相同的方法可应用到包括多个子范围的隔离范围的列举。列举单独的带有来自更高子范围的资源的子范围，该更高的子范围代替来自集合范围中的更低子范围的匹配的实例。

[0297] 在其他实施例中，从用户隔离范围层下至系统层执行列举，而非反过来。在这些实施例中，列举用户隔离范围。接着列举应用隔离范围，并且出现在应用隔离范围内没有在用户隔离范围中列举的任何资源实例被加入到结构之下的集合范围。相似的处理过程可被重复用于仅出现在系统范围内的资源。

[0298] 仍然在其他实施例中，同时列举所有的隔离范围并且合并各个列举。

[0299] 如果应用试图打开存在的原生资源实例，但不试图修改该资源，则返回到应用的特定的实施例是在虚拟范围内发现的一个，或等效的，该实施例将出现在请求的资源的父虚拟列举中。从隔离环境的范围点，该应用被告知去请求打开“虚拟资源”，并且用于满足该请求的原生资源的特定实例被称为对应于请求的资源的“文字资源”。

[0300] 如果代表用户正执行的应用试图打开资源并指示它正在做的带有改变该资源的意图，当在应用隔离范围和系统范围内的资源对代表其他用户正在执行的应用是公共的时，通常的给该应用实例将要修改的资源的私有副本。代典型地，除非用户范围实例已经存在，否则制作资源的用户范围副本。通过虚拟范围提供的集合范围的限定意味着向用户隔离范围拷贝应用范围或系统范围的资源的行为没有改变集合范围，该集合范围通过虚拟范围提供给正被讨论的用户或应用，不给其他用户，也不给任何其他应用实例。后来由代表用户正在执行的应用实施例作出的对该副本资源的修改不影响任何其他应用实施例的集合范围，上述任何其他应用实施例不共享相同的用户隔离范围。换句话说，这些修改没有改变原生资源的集合范围，该原生资源用于其他用户或者用于与相同的应用隔离范围不相关的应用实例。

[0301] 应用可安装于特定的隔离范围内(以下更详细的描述)。安装在隔离范围内的应用总是与该范围相关联。可替代地，应用可在特定隔离范围内或者许多的隔离范围内启动。有效的，启动应用，并且该应用与一个或多个隔离范围相关联。相关联的隔离范围提供带有原生资源特定范围的处理。应用也可在系统资源范围内启动，即，它们可以与非隔离范围相关联。这使得在隔离范围内可选择性的执行操作系统应用，例如Internet Explorer，以及第三方应用。

[0302] 无论应用安装在何处在隔离范围内启动应用的能力减少了应用兼容性和群集度问题，从而不要求在隔离范围内单独安装应用。选择性的启动在不同隔离范围内的已安装的应用的能力提供拥有应用的能力，该应用需要帮助者应用(例如Word，Notepad等)使这些帮助者应用以相同规则集合启动。

[0303] 另外，在多个隔离环境中启动应用的能力允许在隔离的应用和公共的应用之间更好的集成。

[0304] 现在看图8C，总的来说，用于将过程和隔离范围相关联的方法包括在暂停状态中启动过程的步骤(步骤882)。获取与期望的隔离范围相关联的规则(步骤884)并且在存储元件中存储用于过程和获取规则的标识符(步骤886)，并且重新开始暂停过程(步骤888)。拦截或挂起由过程作出的访问原生资源的随后调用(步骤890)并且与过程标识符相关联的规则，如果有的话，被用于对请求资源的虚拟访问(步骤892)。

[0305] 仍然看图8C，并且更详细的，过程在暂停状态中被启动(步骤882)。在一些实施例中，使用定制的启动程序完成此项任务。在这些实施例的一些中，启动装置被特别的设计以启动过程到选择的隔离环境中。在其他实施例中，启动当做期望的隔离环境的说明的输入，例如，通过命令行选项。

[0306] 获取与期望的隔离范围相关联的规则(步骤884)。在一些实施例中，从持久存储元件中获取规则，例如硬盘驱动器或其他固态存储元件。规则可作为关系数据库、平面文件数据库、树形结构数据库、二进制树结构或其他持久性数据结构被存储。在其他实施例中，规则可存储于为存储它们特别定制的数据结构中。

[0307] 用于过程的标识符，例如过程标识符(PID)，以及获取规则被存储在存储元件中(步骤886)。在一些实施例中，提供内核模式驱动器以接收关于新过程生成的操作系统消息。在这些实施例中，PID和获取的规则可被存储在驱动器的上下文中。在其他实施例中，提供文件系统过滤器驱动器，或迷你过滤器拦截原生资源请求。在这些实施例中，PID和获取规则可存储在过滤器中。仍然在其他实施例中，通过用户模式挂起执行所有的拦截，并且根本不存储PID。在初始化过程期间，通过用户模式挂起设备加载规则，因为规则关联完全在过程中执行，所以不需要其他的部件知道应用到PID的规则。

[0308] 重新开始暂停过程(步骤888)，并且拦截或挂起由过程作出的访问原生资源的随后调用(步骤890)，并且与过程标识符相关联的规则，如果有的话，用于对请求资源的虚拟访问(步骤892)。在一些实施例中，文件系统过滤器驱动器，或者迷你过滤器，或者文件系统驱动器拦截访问原生资源的请求，并确定与拦截请求相关联的过程标识符是否已经与规则集合相关联。如果是这样，与存储的过程标识符相关联的规则被用于虚拟访问原生资源的请求。如果不是，访问原生资源的请求可不被修改的通过。在其他实施例中，动态链接库被载入到新生成的过程中，并且该库载入隔离的规则。仍然在其他实施例中，内核模式技术(挂起、过滤器驱动器、迷你驱动器)以及用户模式技术都被用于拦截对访问原生资源的调用。对于文件系统过滤器驱动器存储规则的实施例，该库可载入来自于文件系统过滤器驱动器的规则。

[0309] 与隔离范围相关联的过程的“子”过程与它们的“父”过程的隔离范围相关联。在一些实施例中，通过内核模式驱动器在子过程生成时通报文件系统过滤器驱动器可实现该关联。在这些实施例中，文件系统过滤器驱动器确定父过程的过程标识符是否与隔离范围相关联。如果这样，文件系统过滤驱动器存储用于新生成的子过程的过程标识符和父过程的隔离范围间的关联。在另一个实施例中，文件系统过滤器驱动器可从系统直接调用而不需使用内核模式驱动器。在其他实施例中，在与隔离范围相关联的过程中，生成新过程的操作系统功能被挂起或拦截。当从这样的过程接收到生成新过程的请求时，存储新的子过程和父过程的隔离范围间的关联。

[0310] 在一些实施例中，范围或子范围与单独的线程相关联而非与整个过程，这使得隔离在每一线程的基础上被执行。在一些实施例中，每一线程隔离可以被用于服务或COM+服务器。

[0311] 在一些实施例中，使用隔离环境向应用流客户端552提供附加的功能。在这些实施例的一个中，应用程序在隔离环境中被执行。在这些实施例的另一个中，获取的多个应用文件驻留在隔离环境中。仍然在这些实施例的另一个中，在隔离环境中作出本地机器10上的登记的改变。

[0312] 在一个实施例中，应用流客户端552包括隔离环境556。在一些实施例中，应用流客户端552包括文件系统过滤器驱动器564，该文件系统过滤器驱动器564拦截对于文件的应用请求。在这些实施例的一个中，文件系统过滤器驱动器564拦截应用请求以打开已存在的文件并确定该文件没有驻留在隔离环境556内。在这些实施例的另一个中，文件系统过滤器驱动器564响应于确定该文件没有驻留在隔离环境556中而重定向对流服务554的请求。流服务554可从多个应用文件中析取文件，并在隔离环境556中存储该文件。文件系统过滤器驱动器564可接着响应对于带有的文件已存储副本的文件的请求。在一些实施例中，文件系

统过滤器驱动器564响应于流服务554没有获取到文件或多个应用文件的指示和该文件没有驻留在隔离环境556中的决定，可以将文件的请求重定向到文件服务器540。在一些实施例中，流服务554可包括加速程序6120来执行下述讨论的一些或所有的加速技术以加速文件和应用的存储和传送。

[0313] 在一些实施例中，文件系统过滤器驱动器564使用严格的隔离规则来阻止冲突或不一致数据出现在隔离环境556中。在这些实施例的一个中，在用户隔离环境中拦截对于资源的请求的文件系统过滤器驱动器564可重定向该请求到应用隔离环境。在这些实施例的另一个中，文件系统过滤器驱动器564不重定向该请求到系统范围。

[0314] 在一个实施例中，流服务554使用IOCTL命令与过滤器驱动器通信。在另一个实施例中，到文件服务器540的通信使用Microsoft SMB流协议被接收。

[0315] 在一些实施例中，包机制530在清单文件(manifest fi le)中存储已公布文件类型的列表作为可用的应用并且使得此信息对应用公布软件可用。在这些实施例的一个中，包机制530接收从监控应用程序安装到分段机器上的隔离环境中的信息。在这些实施例的另一个中，包机制530的用户向包机制530提供此信息。在另一个实施例中，在访问组控制台520中的应用公布软件查找该清单文件以便向访问组控制台520的用户呈现可能的文件类型，该文件类型与被请求的正在被公布的应用相关联。用户选择文件类型以关联特定公布的应用。在应用列举时文件类型被呈现给本地机器10。

[0316] 本地机器10可包括客户端代理560。客户端代理560提供将文件类型与应用程序相关联的功能，和响应于关联选择应用程序的执行的方法的功能。在一个实施例中，客户端代理560为程序邻近应用。

[0317] 当应用程序被选择用于执行时，本地机器10确定执行的方法，该执行的方法与应用程序的文件类型相关联。在一个实施例中，本地机器10确定文件类型与执行的方法相关联，该执行的方法要求在隔离环境中获取应用文件和执行的应用流会话。在该实施例中，本地机器10可重定向该请求到应用流客户端552，而不是启动应用程序的本地版本。在另一个实施例中，客户端代理560作出决定。仍然在另一个实施例中，客户端代理560重定向该请求到应用流客户端552。

[0318] 在一个实施例中，应用流客户端552从远程机器30请求与应用程序相关联的访问信息。在一些实施例中，应用流客户端552接收包含访问信息的可执行程序。在这些实施例的一个中，应用流客户端552接收可执行程序，该可执行程序可以在本地机器10上显示应用输出数据，该应用输出数据从在远程机器上的应用程序的执行中产生。在这些实施例的另一个中，应用流客户端552接收可执行程序，该可执行程序可以通过应用流会话获取该应用程序，并在本地机器10上的隔离环境中执行该应用程序。在这个实施例中，应用流客户端552可执行接收到的可执行应用程序。仍然在这些实施例的另一个中，如上所述的，远程机器30响应于执行应用程序解决，选择可执行程序以便提供给本地机器10。

[0319] 现在看图9，描述了用于执行应用的方法中所使用的步骤的一个实施例的流程图。如上述在图7中所述，关于步骤214，本地机器10接收并执行多个应用文件。总的来说，本地机器10接收包括访问信息的文件以访问多个应用文件并执行可以接收应用流的第一客户端(步骤902)。响应于该文件，本地机器10获取多个应用文件的标识(步骤904)。响应于该文件，本地机器10获取在多个应用文件的执行中需要的至少一个特征(步骤906)。本地机器10

确定本地机器10是否包括至少一个特征(步骤908)。响应于确定本地机器10缺少至少一个特征,本地机器10执行第二客户端,第二客户端请求在远程机器上的多个应用文件的执行(步骤910)。

[0320] 现在看图9,更详细的,本地机器10接收包括访问信息的文件,访问信息用于访问多个应用文件并用于执行的第一客户端,该第一客户端可以接收应用流(步骤902)。在一个实施例中,本地机器10接收包括多个应用文件的位置标识的访问信息,这些应用文件包括应用程序。在另一个实施例中,本地机器10响应于应用程序的执行请求来接收文件。仍然在另一个实施例中,访问信息包括指示该多个应用文件驻留在本地机器30'上,例如应用服务器或文件服务务器。仍在另一个实施例中,访问信息指示本地机器10可在应用流会话上从远程机器30获取该多个应用文件。

[0321] 本地机器10响应于文件获取该多个应用文件的标识(步骤904)。在一个实施例中,响应于包括访问信息的文件,本地机器10识别其上有该多个应用文件驻留的远程机器。在另一个实施例中,本地机器10从远程机器30获取识别该多个应用文件的文件。在一些实施例中,该多个应用文件包括应用程序。在其他实施例中,该多个应用文件包括多个应用程序。仍然在其他实施例中,该多个应用文件包括单个应用程序的多个版本。

[0322] 现在提前看图10,示出了驻留在例如文件服务器540的远程机器30'上驻留的多个应用文件的一个实施例的流程图。在图10中,多个应用文件,指定为包,包括包含一个或多个应用程序的三个不同版本的应用文件。

[0323] 在一个实施例中,包括一个或多个应用程序的版本并被存储在包中的应用文件的每个子集被指定作为目标。目标1,例如,包括字处理应用程序和电子数据表程序的版本,该版本与Microsoft Windows 2000操作系统的英语版本相兼容。目标2包括字处理应用程序和电子数据表程序的版本,该版本与Microsoft XP操作系统的英语版本相兼容。目标3包括字处理应用程序和电子数据表程序的版本,该版本与带服务包3的Microsoft Windows 2000操作系统的日语版本相兼容。

[0324] 现在转而看图9,在一些实施例中,从寄载该多个应用文件的远程机器30获取的文件包括包的描述和包括在该多个应用文件中的目标。在其他实施例中,从远程机器30获取的文件识别该多个应用文件,该多个应用文件包括由本地机器10请求执行的应用程序。

[0325] 响应于该文件,本地机器10获取执行该多个应用文件需要的至少一个特征(步骤906)。在一些实施例中,本地机器10可以不执行应用程序,除非本地机器包括特定特征。在这些实施例的一个中,不同的应用程序要求本地机器10包括不同的特征,这些特征来自于被其他应用程序需要的特征。在这些实施例的另一个中,本地机器10接收执行该多个应用文件需要的至少一个特征的标识,该多个应用文件包括本地机器10请求的应用程序。

[0326] 本地机器确定本地机器10是否包括该至少一个特征(步骤908)。在一个实施例中,本地机器10评价本地机器10上的操作系统以确定本地机器10是否包括至少一个特征。在另一个实施例中,本地机器10识别本地机器10上的操作系统所使用的语言,以确定本地机器10是否包括该至少一个特征。仍然在另一个实施例中,本地机器10识别本地机器10上的操作系统的修订级,以确定本地机器10是否包括至少一个特征。仍然在另一个实施例中,本地机器10识别驻留在本地机器10上的应用程序的应用版本以确定本地机器10是否包括至少一个特征。在一些实施例中,本地机器10确定本地机器10是否包括设备驱动器以确定本地

机器10是否包括该至少一个特征。在其他实施例中，本地机器10确定本地机器10是否包括操作系统以确定本地机器10是否包括该至少一个特征。仍然在其他实施例中，本地机器10确定本地机器10是否包括执行该多个应用文件的许可以确定本地机器10是否包括至少一个特征。

[0327] 响应于确定本地机器10缺少该至少一个特征，本地机器10执行第二客户端，该第二客户端请求在远程机器30上的该多个应用文件的执行(步骤910)。在一个实施例中，当本地机器10确定该本地机器10缺少该至少一个特征时，本地机器10不执行能够接收应用流的第一客户端。在另一个实施例中，当本地机器10缺少至少一个特征时，策略禁止本地机器10在应用流上接收该多个应用文件。在一些实施例中，本地机器10确定本地机器10包括该至少一个特征。在这些实施例的一个中，本地机器10执行第一客户端，该第一客户端从远程机器30接收应用流用于在本地机器上执行，该应用流包括该多个应用文件。

[0328] 在一些实施例中，一旦确定本地机器10缺少该至少一个特征，本地机器10执行第二客户端，该第二客户端请求在远程机器上该多个应用文件的执行。在这些实施例的一个中，第二客户端向寄载该多个应用文件的远程机器30传输该请求。在这些实施例的另一个中，远程机器30执行包括应用程序的该多个应用文件，并产生应用输出数据。仍然在这些实施例的另一个中，第二客户端接收通过在远程机器上的该多个应用文件的执行产生的应用输出数据。在一些实施例中，第二客户端通过独立计算结构表示层协议或远程桌面窗口表示层协议或X-Windows表示层协议接收应用输出数据。仍在这些实施例的另一个中，第二客户端在本地机器10上显示应用输出。

[0329] 在一些实施例中，第二客户端向远程机器30传输请求，该远程机器30没有寄载该多个应用文件。在这些实施例的一个中，远程机器30可从寄载该多个应用文件的第二远程机器30请求该多个应用文件。在这些实施例的另一个中，远程机器30可通过应用流会话从第二远程机器30接收该多个应用文件。仍然在这些实施例的另一个中，远程机器30在隔离环境中存储接收的该多个应用文件并在隔离环境中执行应用程序。仍然在这些实施例的另一个中，远程机器30向在本地机器上的第二客户端传输产生的应用输出数据。

[0330] 现在回看图5，在一个实施例中，可以接收应用流的第一客户端是应用流客户端552。应用流客户端552接收文件，响应于该文件，获取多个应用文件的标识和执行该多个应用文件需要的至少一个特征，并确定本地机器10是否包括至少一个特征。在另一个实施例中，第二客户端是客户端代理560。在一些实施例中，响应于应用流客户端552作出的本地机器10缺少该至少一个特征的确定，客户端代理560从应用流客户端552接收文件。

[0331] 在一些实施例中，在本地机器10上执行的应用566使用Win32FindFirstFile()和FindNextFile()API调用来列举与应用566相关联的文件。在这些实施例的一个中，多个应用文件包括应用566。在这些实施例的另一个中，不是所有的在多个应用中的文件都驻留在本地机器10上。仍然在这些实施例的另一个中，流服务554获取在存档文件中的多个应用文件但只提取该多个应用文件的子集。仍在这些实施例的另一个中，即使在请求的文件没有驻留在本地机器10上时，流服务554和文件过滤器驱动器564仍然提供用于满足列举请求的功能。

[0332] 在一个实施例中，通过拦截列举请求并且好像在该多个应用文件内的所有的文件都驻留在本地机器10上那样提供数据来提供功能。在另一个实施例中，通过由文件系统过

滤器驱动器564拦截以IOCTL命令传输的列举请求，例如IRP_MJ_DIRECTORY_CONTROL IOCTL，来提供功能。当文件系统过滤器驱动器564拦截该调用时，文件系统过滤器驱动器564将该请求重定向到该流服务554。在一个实施例中，在将该请求重定向到该流服务554前，文件系统过滤器驱动器564确定该被请求的列举驻留在本地机器10上的隔离环境中。在另一个实施例中，流服务554使用在该多个应用文件中的文件实现该请求，该文件包括与该多个应用文件相关联的目录结构的列举。仍然在另一个实施例中，流服务554向文件系统过滤器驱动器564提供该请求的响应用于满足列举请求。

[0333] 现在看图11，描述了方法中所采用的步骤的一个实施例的流程图，该方法用于本地响应对与远程存储的文件相关联的文件元数据的请求。总的来说，(i)表示远程机器存储的应用程序的目录结构，和(ii)元数据被从远程机器接收，该元数据与包括存储的应用程序的每个文件相关联(步骤1102)。该目录结构和元数据被存储(步骤1104)。接收至少一个请求，该请求用于访问与目录结构中的特定文件相关联的元数据(步骤1106)。使用所存储的元数据响应该至少一个请求(步骤1108)。

[0334] 现在更详细的看图11，从远程机器接收代表被远程机器存储的应用程序的目录结构和与包括存储的应用程序的每个文件相关联的元数据(步骤1102)。在一个实施例中，流服务554接收该目录结构和该元数据。在另一个实施例中，当流服务554获取包括存储的应用程序的多个应用文件时，流服务554接收该目录结构和元数据。仍然在另一个实施例中，目录结构和元数据被存储在多个应用文件中的文件中。

[0335] 在一个实施例中，与每个文件相关联的元数据包括用于至少一个文件的选用名称。在另一个实施例中，与每个文件相关联的元数据包括用于至少一个文件的短名称，该名称的长度为八个字符、点以及三个字符的扩展名。仍然在另一个实施例中，与每个文件相关联的元数据包括在用于至少一个文件的选用名称和用于至少一个文件的短名称之间的映射。在一些实施例中，在该多个应用文件中的文件有选用文件名。在这些实施例的一个中，当流服务554获取到本地机器的文件时，响应于用于该文件的选用名称和用于至少一个文件的短名称之间的映射，该文件与短名称相关联。

[0336] 目录结构和元数据被存储(步骤1104)。在一个实施例中，目录结构和元数据存储在隔离环境556中。在另一个实施例中，目录结构和元数据存储在高速缓存元件中。仍然在另一个实施例中，表示应用程序被远程机器存储的目录结构被用于产生目录结构的列举，该目录结构表示在本地机器上执行的应用程序。

[0337] 接收访问元数据的至少一个请求，该元数据与在目录结构中的特定文件相关联(步骤1106)。在一个实施例中，该请求是文件列举请求。在另一个实施例中，该请求是确定文件副本是否本地驻留的请求，该文件副本包括存储的应用程序。

[0338] 在一个实施例中，由在本地机器上的隔离环境中执行的应用566作出请求。在另一个实施例中，由应用流客户端552作出请求。仍然在另一个实施例中，代表应用566作出请求。

[0339] 在一个实施例中，文件系统服务器过滤器564拦截该请求。在另一个实施例中，通过文件系统过滤器驱动器564转发该请求到应用流客户端552。仍然在另一个实施例中，通过文件系统过滤器驱动器564转发该请求到流服务554。

[0340] 在一些实施例中，该请求被代替操作系统功能或代替用于列举目录功能的功能挂

起。在另一个实施例中，使用挂起动态链接库拦截该请求。在用户模式或内核模式执行该挂起功能。对于挂起功能在用户模式中执行的实施例，当过程被生成时该挂起功能被载入到过程的地址空间中。对于挂起功能在内核模式执行的实施例，挂起功能与操作系统资源相关联，该操作系统资源用于分派对于文件操作的请求。对于为文件操作的每种类型提供单独的操作系统功能的实施例，每个功能可被分别的单独挂起。可选的，可提供单独的挂起功能，其拦截用于一些类型的文件操作的生成或打开调用。

[0341] 使用存储的元数据响应该至少一个请求(步骤1108)。在一个实施例中，文件系统过滤器驱动器564响应该请求。在另一个实施例中，应用流客户端552响应该请求，仍然在另一个实施例中，流服务554响应该请求。在一个实施例中，访问存储的元数据以响应该至少一个请求。在另一个实施例中，以文件的远程副本本地驻留的错误指示响应该请求。

[0342] 在一个实施例中，响应于接收到的元数据，满足Windows操作系统的FindFirst操作。在另一个实施例中，响应于接收到的元数据，满足Windows操作系统的FindNext操作。仍然在另一个实施例中，响应于接收到的元数据，满足在目录结构中根节点的识别操作。在一些实施例中，使用例如WIN32_FIND_DATA API的应用层API来响应该操作。在其它实施例中，使用内核层API(例如FILE_BOTH_DIR_INFORMATION)来响应该操作。

[0343] 在一个实施例中，元数据满足用于标识与目录结构中的节点相关联的访问次数的操作。在另一个实施例中，元数据满足用于标识与目录结构中的节点相关联的修改次数的操作。仍然在另一个实施例中，元数据满足用于识别在目录结构中的所修改的节点的操作。

[0344] 现在看图12，描述了系统的一个实施例的框图，该系统用于本地响应对与远程存储的文件相关联的文件元数据的请求，该系统包括流服务554、文件系统过滤器驱动器564、目录结构570、多个应用文件572、元数据574、以及高速缓存元件576。总的来说，目录结构570识别与至少一个应用程序相关联的多个文件。元数据574与该多个文件的至少一个相关联，多个文件中的至少一个驻留在远程机器上。在一个实施例中，目录结构570包括元数据574。高速缓存元件576存储目录结构570。文件系统过滤器驱动器564拦截对访问元数据的请求，该元数据与至少一个远程存储的文件相关联，文件系统过滤器驱动器564访问高速缓存元件，并使用存储的目录结构来响应该至少一个请求。

[0345] 在一些实施例中，流服务554接收目录结构570和元数据574。在这些实施例的一个中，目录机构570表示与应用程序相关联的多个应用文件572，该多个应用文件572驻留在远程机器上，例如远程机器30。在这些实施例的另一个中，元数据574包括用于响应Windows操作系统的FindFirst请求的信息。仍然在这些实施例的另一个中，元数据574包括用于响应Windows操作系统的FindNext请求的信息。也在这些实施例的另一个中，元数据574包括用于响应对在目录结构中的根节点的标识的请求的信息。在这些实施例的另一个中，元数据574包括用于响应对在目录结构中的节点的标识的请求的信息。在一些实施例中，使用应用层API(例如WIN32_FIND_DATA API)来响应操作。在其它实施例中，使用内核层API(例如FILE_BOTH_DIR_INFORMATION)来响应操作。

[0346] 在一些实施例中，关于文件的少量元数据574可以文字的文件名直接存储，例如通过为带有元数据指示器的虚拟名称添加后缀，此处元数据指示器是唯一的与特定元数据状态相关联的字符串。元数据指示器可指示或编码元数据的一个或几个位。由于元数据指示器的存在，通过虚拟文件名访问文件的请求检查文字的文件名可能的变化，并且为了使用

文字的文件名响应，获取文件本身的名称的请求被挂起或拦截。在其他实施例中，用于文件的一个或多个选用名称由虚拟文件名和元数据指示器构成，并使用文件系统提供的硬链接或软链接设备生成。如果给出使用链接的名称访问文件的请求，则由隔离环境通过指示该文件没有找到从而向应用隐藏这些链接的存在。特定链接的存在或不存在可指示用于每个元数据指示器的元数据的一个位，或者存在带有元数据指示器的链接，该指示器能够采用多个状态以指示元数据的一些位。仍然在其他实施例中，此处文件系统支持可选的文件流，生成可选的文件流以具包含元数据，该文件流带有指示元数据几个位的流尺寸。仍然在其他实施例中，文件系统可直接提供存储一些第三方元数据用于在文件系统中的每个文件的能力。仍然在其他实施例中，可以使用单独的子范围记录删除的文件，在该子范围内存在的文件(没有标记为占位符)意味着该文件被删除。

[0347] 在一个实施例中，合并在用户隔离环境、应用隔离环境和系统范围内的数据以形成表示应用的目录结构的本地列举。在另一个实施例中，流服务554访问元数据574和目录结构570以总装(populate)应用隔离环境。仍然在其他实施例中，文件系统过滤器驱动器564产生目录结构的本地列举。仍然在另一个实施例中，目录结构的本地列举识别在该多个应用文件572中的至少一个文件，该至少一个文件驻留在远程机器而非本地机器上。在一些实施例中，目录结构的本地列举被存储在高速缓存元件576中。在其他实施例中，流服务554产生应用隔离环境和目录结构的本地列举。

[0348] 在一个实施例中，文件系统过滤器驱动器564拦截请求，该请求传输到系统范围用于访问目录结构的本地列举。在另一个实施例中，文件系统过滤器驱动器564在拦截该请求后产生本地列举。仍然在另一个实施例中，文件系统过滤器驱动器564将对本地列举的请求重定向到用户隔离环境。仍然在另一个实施例中，文件系统过滤器驱动器564将对本地列举的请求重定向到应用隔离环境。

[0349] 在一些实施例中，文件系统过滤器驱动器564拦截对文件访问的请求，该文件在目录的本地列表中识别，该文件驻留远程机器上。在这些实施例的一个中，如下述结合图13更详细描述的，文件系统过滤器驱动器564通过流服务554请求文件的获取。

[0350] 当在隔离环境中运行的应用做出对于文件的请求时，过滤器驱动器拦截这些请求。如果该请求是用于打开文件，过滤器驱动器将首先重定向该请求到隔离环境，以确定该请求是否可被隔离环境满足。如果调用成功，过滤器驱动器将使用位于隔离环境中的该文件的实例来响应请求。

[0351] 然而，如果该请求的文件没有驻留在隔离环境中，过滤器驱动器向流服务554发送请求以从该多个应用文件、程序块中获取该文件，直到该请求完成，并且接着重试最初的打开。在一些实施例中，流服务554的功能被称为“随选高速缓存”，该流服务554用于一旦从过滤器驱动器接收到请求则从该多个应用文件中获取文件。

[0352] 现在看图13，描述了方法中所采用步骤的实施例的流程图，该方法用于访问在目录机构中的远程文件，该目录结构与本地执行的应用程序相关联。总的来说，拦截通过应用访问文件的请求(步骤1302)。该请求被重定向到第一隔离环境(步骤1304)。做出被请求的文件在第一隔离环境中不存在的决定(步骤1306)。响应文件在目录结构的列举中识别，该请求被重定向到第二隔离环境，该目录结构与驻留在远程机器上的多个应用文件关联(步骤1308)。响应于第二隔离环境不包括该文件以及该文件在列举中被识别的决定，而从远程

机器获取所请求的文件(步骤1310)。

[0353] 现在看图13,更详细的,拦截应用对于访问文件的请求(步骤1302)。在一个实施例中,通过文件系统过滤器驱动器拦截该请求。在另一个实施例中,文件系统驱动器过滤器拦截对于访问文件的所有请求。仍在另一个实施例中,应用流客户端552拦截该请求。在一些实施例中,拦截应用对于访问可执行文件的请求。在其他实施例中,拦截应用对于文件、在本地机器10上执行的应用一部分的访问的请求。

[0354] 该请求被重定向到第一隔离环境(步骤1304)。在一个实施例中,应用在第一隔离环境中执行。在一个实施例中,应用是应用程序,例如字处理程序或电子数据表程序。在另一个实施例中,该应用是应用流客户端552。仍然在另一个实施例中,应用是在应用流客户端552内的部件,试图启动代表本地机器10上的用户的应用程序。在另一个实施例中,文件系统过滤器驱动器重定向该请求到第一隔离环境。

[0355] 做出该请求的文件没有存在于第一隔离环境中的决定(步骤1306)。在一个实施例中,文件系统过滤器驱动器接收请求的文件没有存在于第一隔离环境中的指示。

[0356] 响应于该文件在目录结构列举中被识别的决定,该请求被重定向到第二隔离环境,该目录结构与驻留在远程机器上的多个应用文件相关联(步骤1308)。在一个实施例中,接收带有关于第一应用执行的访问信息的目录结构列举。在另一个实施例中,该列举识别包括第二应用的多个应用文件。在此实施例中,第一应用是第二应用的本地副本。

[0357] 响应于第二隔离环境不包括文件以及文件在列举中被识别的决定,从远程机器获取请求的文件(步骤1310)。在一个实施例中,从第二远程机器获取请求的文件。在另一个实施例中,从文件服务器获取请求的文件。在一些实施例中,目录结构的列举识别驻留在本地机器上的多个应用文件。在其他实施例中,目录结构的列举指示该多个应用文件驻留在本地机器上。在这些实施例的一个中,目录结构的列举已经指示该多个应用文件驻留在本地机器上,当应用请求存储在多个应用文件中的文件时,一旦拦截访问请求则从文件服务器获得该文件。在这些实施例的另一个中,文件服务器将请求的文件流式传输到本地机器。仍然在这些实施例的另一个中,一旦接收到请求的文件,该请求的文件被存储在第二隔离环境中。仍然在其他实施例中,目录结构的列举已经指示该多个应用文件驻留在本地机器上,当应用请求存储在多个应用文件中的文件时,文件的副本从本地高速缓存提供给应用。

[0358] 在一些实施例中,请求的文件被加密。在其它实施例中,请求的文件以加密的形式被存储。仍然在其他实施例中,如果应用程序缺少对请求的文件访问的授权,则阻止请求该文件的应用解密被请求的文件。

[0359] 在一个实施例中,作出目录结构列举没有识别该文件的决定。在此实施例中,访问文件的请求可以被重定向到在第一隔离环境之外和第二隔离环境之外的环境中。

[0360] 在一些实施例中,拦截访问文件的第二请求。在这些实施例的一个中,通过第二应用作出访问文件的请求。在这些实施例的另一个中,在第三隔离环境中执行第二应用。仍然在这些实施例的另一个中,响应于该文件在列举中被列出以及第二隔离环境包括该文件的决定,该请求被重定向到第二隔离环境。一旦从文件服务器接收到文件,则可以作出本地机器在第二隔离环境中存储文件的决定。仍然在另一个实施例中,该文件被存储在第三隔离环境中。

[0361] 现在看图14,描述了系统的一个实施例的框图,该系统用于访问与应用相关联的

目录结构中的文件。总的来说，本地机器10包括应用流客户端552、流服务554、隔离环境556、文件系统过滤器驱动器564以及第一应用556。本地机器10可与文件服务器540、远程机器30、网络接口558以及第二应用556’相互作用。

[0362] 本地机器10初始化应用流客户端552以执行第一应用556。在一个实施例中，应用流客户端552初始化流服务554以获取并执行第一应用566。在一些实施例中，多个应用文件包括第一应用566。在这些实施例的一个中，流服务554获取该多个应用文件并将它们存储在隔离环境566中。在这些实施例的另一个中，流服务554识别远程机器的位置，该多个应用程序文件驻留在该远程机器上但不获取该多个应用程序文件。仍然在这些实施例的另一个中，流服务554在该多个应用程序文件中获取文件的子集。仍然在这些实施例的另一个中，流服务554获取包括多个应用程序文件的档案文件。

[0363] 在一个实施例中，第一应用566包括驻留在远程机器30上的第二应用566’的本地副本。在另一个实施例中，多个应用文件驻留在远程机器30上并且包括驻留在远程机器30上的第二应用566’。仍然在另一个实施例中，为了执行第二应用566’，本地机器10获取多个应用文件，在本地机器上生成第一应用566，并执行第一应用566。在一些实施例中，应用566和566’是用户应用，例如字处理应用或电子数据表应用或显示应用。

[0364] 在一些实施例中，多个应用文件包括识别目录结构的文件，该目录结构与在远程机器30上的该多个应用文件相关联。在这些实施例的一个中，该文件包括元数据，该元数据涉及在该多个应用文件中的每个应用文件。如上述结合图12的描述，在这些实施例的另一个中，流服务554从文件中获取元数据以生成目录结构的列举，该目录结构与该多个应用文件相关联。仍然在这些实施例的另一个中，流服务554存储目录结构的列举，该目录结构与包括第二应用556’的多个应用文件相关联。在一些实施例中，流服务554在第二隔离环境中存储列举。

[0365] 在一个实施例中，流服务554获取与第一应用566相关联的初始可执行文件。在另一个实施例中，一旦获取初始可执行文件，流服务554执行在本地机器10上的第一应用566。仍然在另一个实施例中，当需要该文件以继续第一应用566的执行时，第一应用566请求在多个应用程序文件中的其他文件的访问。在一些实施例中，第一应用566在隔离环境556中执行。

[0366] 文件系统过滤器驱动器564通过第一应用566拦截请求，该第一应用566在隔离环境556中执行，用于访问在多个应用文件中的文件。文件系统过滤器驱动器564重定向该请求到隔离环境556。如果请求的文件驻留在隔离环境556中，向第一应用566提供对请求的文件的访问。

[0367] 如果请求的文件没有驻留在隔离环境556中，文件系统过滤器驱动器564重定向该请求到第二隔离环境。在一个实施例中，第二隔离环境包括目录结构的列举，该目录结构通过流服务554产生并与包括第二应用566’的多个应用文件相关联。在另一个实施例中，做出请求的文件在目录结构的列举中被识别的决定。

[0368] 在一些实施例中，流服务554向隔离环境556提供信号量。在这些实施例的一个中，文件系统过滤器驱动器564使用该信号量来向流服务554指出需要访问在多个应用文件中的文件。在其它实施例中，文件系统过滤器驱动器564使用线程向流服务554指示需要访问文件。

[0369] 一旦从文件系统过滤器驱动器564接收到通知，流服务554从多个应用文件获取请求的文件。仍然在这些实施例的另一个中，流服务554在第二应用隔离环境中存储请求的文件。在一个实施例中，用于访问文件的请求使用文件的实施例满足，该文件的实施例从多个应用文件中获取并存储在第二隔离环境中。在另一个实施例中，请求的文件也被存储在第一隔离环境中。

[0370] 在一些实施例中，做出第二隔离环境不包括文件以及该文件在列举中被识别的决定。在这些实施例的一个中，文件在目录结构的列举中识别，该目录结构与包括第二应用566'的多个应用文件相关联，并且该文件是该多个应用文件中的文件。在这些实施例的另一个中，流服务554不从远程机器上获取文件。仍然在这些实施例的另一个中，流服务554不获取包括请求的文件的多个应用文件。仍然在这些实施例的另一个中，流服务554在档案文件中获取多个应用文件，但不获取来自档案文件的请求的文件。

[0371] 在一个实施例中，流服务554包括收发器，与文件系统过滤器驱动器通信。在另一个实施例中，收发器从文件系统过滤器驱动器接收重定向的请求。仍然在另一个实施例中，收发器向寄载请求的文件的远程机器转发对于文件的请求。在一个实施例中，远程机器是文件服务器540。在另一个实施例中，请求被转发到远程机器30，该远程机器30路由该请求到文件服务器540。在一些实施例中，文件服务器540将请求的文件流式传输到本地机器10上的转发器。在其他实施例中，远程机器30将请求的文件流式传输到本地机器10上的转发器。仍然在其他实施例中，一旦从文件服务器540接收到请求的文件，转发器在第二隔离环境中存储接收的文件。

[0372] 在一个实施例中，文件系统过滤器驱动器564拦截用于访问文件的第二请求，该第二请求由第三应用566"做出，并在本地机器上第三隔离环境中执行。在另一个实施例中，文件系统过滤器驱动器564重定向用于访问文件的请求到第二隔离环境。仍然在另一个实施例中，在通过第三应用566"拦截用于访问的请求之前，文件系统过滤器驱动器564确定流服务554在第二隔离环境中存储接收的文件。

[0373] 在一些实施例中，一旦初始化，在执行应用程序之前，流服务554在隔离环境中可以总装高速缓存。在这些实施例的一个中，流服务554将登记文件安装到隔离环境中。在这些实施例的另一个中，流服务554存储在文件的长名称和短文件名称之间的映射。

[0374] 在一个实施例中，为了节省本地机器上的空间，限制高速缓存的尺寸。在一些实施例中，当高速缓存接近它的尺寸限定时，高速缓存中最旧的文件将被自动删除以获得空间用于新的文件。在这些实施例的一个中，文件的年龄通过时间戳确定，该时间戳由操作系统保持，表示"最后一次访问"的时间戳的时间。除了文件的年龄之外，也考虑文件类型－二进制可执行文件(.EXE，.DLL等)比其他类型相同年龄的文件保存的更长。

[0375] 一旦初始化，流服务554可列举在高速缓存中的现有文件，并确定高速缓存的总尺寸。在文件被加入高速缓存之后，或者通过隔离环境556或者通过流服务554，流服务554调用通知高速缓存系统新文件以及其位置和尺寸的功能。每个最近高速缓存的文件的尺寸被加入现有高速缓存尺寸的运行总量中。新的总量接着与高速缓存尺寸限定相比较，并且如果超过了限定，代码发出线程以增加高速缓存的年龄(age the cache)。任何时候可以只有该线程的一个实施例在任何给定的时间运行。

[0376] 线程一般在高速缓存中产生所有文件的列表，通过最近访问的时间戳排序该列

表，并接着开始顺着该列表向下删除文件，直到有足够的磁盘空间满足用于该线程的退出标准。退出标准基于减少高速缓存尺寸下降到低于限定的级，该限定作为限定的百分比(缺省值是10％)被确定。删除比所需要的更多以防止超过该限定，该限定防止高速缓存在每次加入新文件时过载。

[0377] 在一些实施例中，流服务554向本地机器10以压缩文件格式提供在多个应用文件中的每个文件拷贝的能力，这些应用文件包括应用程序。这种能力被称为“预高速缓存”。在这些实施例的一个中，当应用程序被顺序的执行，所有的包请求去到本地副本而非遍历网络。这些实施例使得本地机器10的用户在用户没有访问网络时每次执行应用程序。

[0378] 远程机器30包括监控被本地机器10使用的应用的功能。远程机器30可监控本地机器10所使用的每个应用的状态，例如当应用的执行或终止时。在一个实施例中，远程机器30要求本地机器10传输关于应用状态的消息，该应用被本地机器10执行。在另一个实施例中，当本地机器10连接到远程机器驻留的网络上时，本地机器10传输消息，该消息指示本地机器10已经连接到网络。

[0379] 在一个实施例中，当本地机器10与远程机器30交互并执行一个或多个应用时，本地机器10拥有会话。在另一个实施例中，远程机器30要求本地机器维持许可，该许可用于会话的期间，授权从远程机器接收的应用的执行。仍然在另一个实施例中，会话具有由远程机器分配的唯一的会话标识符。

[0380] 在一个实施例中，本地机器10向远程机器30传输消息，与远程机器交互以接收并执行应用程序。在另一个实施例中，本地机器10从远程机器30接收第二远程机器的标识符，例如会话管理服务器562，第二远程机器接收并存储所有传输的消息，该消息与本地机器10上的会话相关联。

[0381] 在一些实施例中，会话管理服务器562为提供许可管理和会话监控服务的远程机器30。在这些实施例的一个中，会话管理服务器562包括提供这些服务的服务器管理子系统508。

[0382] 在一个实施例中，本地机器10直接向会话管理服务器562传输消息。在另一个实施例中，本地机器10向远程机器30传输消息，远程机器30向会话管理服务器562转发该消息，该会话管理服务器562带有本地机器10的标识。

[0383] 本地机器10可向远程机器30传输心跳消息。在一个实施例中，该心跳消息包括对许可的请求。在此实施例中，本地机器10在接收到与应用程序相关联的访问信息后可传输心跳信息，本地机器10请求授权以执行该应用程序。本地机器10在执行该应用之前传输该心跳消息。在一个实施例中，本地机器10包括带有心跳信息的启动票据，该启动票据带有访问信息被接收。在此实施例中，一旦成功验证启动票据，远程机器30授予本地机器552许可。

[0384] 在另一个实施例中，心跳信息包括本地机器已经开始应用执行的指示。仍然在另一个实施例中，心跳信息包括本地机器已经终止应用程序执行的指示。仍然在另一个实施例中，心跳信息包括应用执行失败的指示。

[0385] 在一个实施例中，心跳信息包括第二会话管理服务器的标识的请求，例如会话管理服务器562。在另一个实施例中，心跳消息包括本地机器10已经连接到远程机器30驻留的网络的指示。

[0386] 在一些实施例中，心跳消息包括重启应用流会话的请求。在这些实施例的一个中，

当发生错误，在远程机器30驻留的网络和本地机器10间的连接被停止时，本地机器10传输该心跳消息。在这些实施例的另一个中，本地机器10传输带有与会话相关联的心跳消息信息。仍然在这些实施例的另一个中，如果会话没有终止，远程机器30可向本地机器10传输会话相关数据。

[0387] 在这些实施例的另一个中，如果远程机器30从在其上应答的网络上断开，本地机器10不接收传输到远程机器30的心跳信息的应答。在一个实施例中，本地机器10通过传输请求到远程机器30的会话复位的消息可以重新重建会话。在另一个实施例中，远程机器10通过传输请求到第二远程机器30的会话复位的消息可以重建会话。在一些实施例中，当远程机器30重连接到网络中时，将为在远程机器30被断开时接收的每个会话重置请求生成新的会话。在这些实施例的一个中，新会话被关联到重连接和未许可状态。在这些实施例的另一个中，不获得用于新的会话的新许可。仍然在这些实施例的另一个中，当本地机器10执行应用时，将获得新的许可，并且与本地机器10相关联的所有会话将与活动的和许可的状态相关联。

[0388] 在一些实施例中，本地机器10上的应用流客户端552生成心跳消息。在这些实施例的一个中，应用流客户端552向web接口558转发该心跳消息用于向本地机器10传输以向远程机器30传输。在其他实施例中，在远程机器30上的管理服务504从本地机器10通过web接口558接收心跳消息。仍然在其他实施例中，包括收集点240的远程机器30(上述结合图1D描述)接收并存储心跳消息。

[0389] 在一些实施例中，应用流客户端552从远程机器30请求许可。在这些实施例的一个中，该许可授权本地机器552上的应用程序的执行。在这些实施例的另一个中，远程机器30可访问第二远程机器以提供该许可。仍然在这些实施例的另一个中，远程机器30可向本地机器提供许可。仍然在这些实施例的另一个中，远程机器30可向第二远程机器提供可接受的用于授权的许可。在一些实施例中，一旦应用程序的执行终止，许可被撤回。

[0390] 在一些实施例中，在群组38中的远程机器30包括许可管理子系统，用于配置和维持用于这些子系统的许可并用于控制到这样的子系统的连接的数量，这些子系统需要操作许可。在其他实施例中，远程机器30包括在其他子系统(例如应用管理子系统和会话管理子系统)中的许可管理子系统的功能。在一个实施例中，每个远程机器30包括许可管理子系统或者与许可管理子系统相关联的功能。许可管理子系统管理两种类型的许可(1)特征许可，以及(2)连接许可。总的来说，许可管理子系统使用特征许可来控制访问许可的软件产品的特征，例如负荷管理，并使用连接许可来控制这些许可软件产品允许的用户连接的数量。特征可为软件产品的一些方面或特定功能，或者该特征可为整个的产品，该产品在没有特征许可时不工作。

[0391] 图15示出了在群组38中的远程机器30的一个实施例，其中远程机器30包括许可管理子系统1510，组子系统1520，持久存储系统服务模块1570，动态存储系统服务模块1580，关系子系统1530，特定远程机器子系统1540，以及与事件总线通信的公共接入点子系统524。图15中示出的这些子系统用于描述许可管理子系统1510的行为。远程机器30包括其他类型的子系统。

[0392] 许可管理子系统1510通过事件总线与组子系统1520通信，从而建立和维持许可的逻辑分组(此处，“许可组”)以促进许可池、分配和组。许可组包括以下描述的许可串的集合

和/或其他许可组。许可组收集相似特征的许可并从而实现许可池化。池化的许可是可被群组38中的任何远程机器30使用的许可。每个许可组保存在许可组和其他许可子组(例如在许可组中的其他许可组)中的许可的收集性能。在一个实施例中,许可池相关信息被保持在动态存储器240中。在该实施例中,每个许可管理子系统1610本地存存储的许可总数量和分配给群组38中的远程机器30的许可的数量。一旦授予池化的许可,该授予许可管理子系统1510在动态存储器240中产生入口,指示该池化的许可"正使用中"。每个其他许可管理子系统1510识别这样的池化的许可不可用于授予的。在一个特定实施例中,动态存储240存储与每个许可组相关联的远程机器ID/客户端ID对以识别使用中的池化的许可。

[0393] 关系子系统1530在许可和远程机器30间以及在许可组和远程机器30间维持关联。关联定义了用于每个许可和许可组的许可的数量,只有相关联的远程机器30可获得该关联(例如"本地许可")。本地许可是分配给群组38中的一个远程机器的许可,并且没有被其他远程机器38分享。许可管理子系统1510与关系子系统1530通信以建立、删除、查询并更新这样的关联。公共接入点子系统524提供每个驻留在远程机器30上的软件产品所使用的远程过程调用(RPC)。这些RPC接口使得这样的软件产品通过公共接入子系统524通信来访问许可信息。

[0394] 仍然看图15,特定远程机器子系统1540与许可管理子系统1510通信以获得特征许可,该特征许可用于需要许可的特定远程机器子系统1540的每个性能。这发生在特定远程机器子系统1540初始化和任何许可事件后。如果不能获得特征许可,特定远程机器子系统1540限制子系统将提供许可的功能。同样的,只要客户端会话使用远程机器30开始,特定远程机器子系统1540使用许可管理子系统1510获得客户端连接许可。

[0395] 当许可串依照命名规定形成时,许可管理子系统1510与持久存储系统服务模块352通信从而在许可存储库1550中存储特征和连接许可。许可存储库1550驻留在持久存储器230中。当这样的许可被存储在许可存储库1550中时,循环冗余码校验(CRC)禁止许可的篡改。许可管理子系统1510也存储与许可存储库1550中的许可串相关的信息。例如,该信息可指示哪个许可被分配给群组38的哪个远程机器30,和在一些实施例中,每个许可的激活状态。在一个实施例中,连接许可表1560存储这些已经获得了连接许可的本地机器的标识符。

[0396] 在一个实施例中,许可管理子系统1510支持来自请求使用许可性能的子系统的事件,例如对可用的池化许可的请求。事件包括请求许可的子系统的UID和该子系统驻留的远程机器30的UID。该事件也包括以许可组ID形式被请求的许可类型(例如,特征或连接许可)。存储在持久存储器230中的实际的许可组ID是任意的,但符合命名规则,为远程机器30提供新软件产品(例如子系统)将来附加的便利。

[0397] 由寻找许可的请求子系统发送的事件包括(1)许可组类型的指示,请求许可的本地机器和远程机器的标识,以及"强制获得"标志。许可组类型的指示可包括特征许可的标识,例如负荷管理,或者连接类型许可,例如软件应用产品。识别寻找许可的本地机器和远程机器的域可包括与远程机器和本地机器相关联的唯一的标识符。强制获得标记可被用于,例如,在许可改变事件之后重新获得连接许可。许可改变事件指示在持久存储器230中的许可信息已经改变;例如,许可已经被删除、增加或者分配。一旦许可改变事件发生,因为远程机器不知道许可改变事件的特定原因,每个远程机器30试图重新获得在许可改变事件

之前所持有的所有的连接许可。该标记，如果设置，指示必须获得连接许可，即使这样做增加了到远程机器30的连接数量，超过允许连接的预定最大量。直到使用中的连接许可量下降到低于预定的最大量，否则后来不授予新的连接许可。在此方式中，由于许可改变事件，本地机器连接许可在会话中将不被终止。

[0398] 现在看图16，描述了包括在许可执行中的部件的一个实施例的框图。远程机器30包括服务器管理子系统508和许可管理子系统512。在一些实施例中，该服务管理子系统508和许可管理子系统512提供上述的许可管理子系统1510的功能。在另一个实施例中，应用管理子系统506和会话管理子系统510提供上述的许可管理子系统1510的功能。仍然在其他实施例中，其他子系统提供上述的许可管理子系统1510的功能。

[0399] 在一个实施例中，服务器管理子系统508包括许可部件，该许可部件用于请求许可的发布和撤回。在另一个实施例中，许可管理子系统512可应用策略到从服务器管理子系统508接收的许可发布和撤回的请求。仍然在另一个实施例中，许可管理子系统512可向提供许可执行功能的远程机器30传输该请求。在一些实施例中，管理服务504可维持与提供许可执行功能德第二远程机器30的连接。在其他实施例中，远程机器30提供许可执行功能。

[0400] 在一些实施例中，一旦本地机器10向远程机器传输预定数量的心跳信息失败，许可的终止或停止将有效。在这些实施例的一个中，许可的终止撤回由本地机器10执行应用程序的授权。

[0401] 在其他实施例中，在预定时间周期终止时，会话将终止。在一个实施例中，在许可满期后，直到会话满期，管理服务504维持会话相关的数据。在一些实施例中，会话相关数据可包括信息例如会话名称、会话ID、客户端ID、客户端名称、会话开始时间、服务器名称(文件服务器的UNC路径)、应用名称(本地机器基于浏览器名称产生的唯一名称)、别名、会话状态(激活/许可、激活/未许可、重连接/未许可)。在另一个实施例中，本地机器10停止心跳消息的传输并在稍后的时间点重新开始心跳消息的传输。仍然在另一个实例中，如果在会话满期之前本地机器10重新开始心跳消息的传输，则管理服务504可重新发行认证并使得保持的会话相关数据对本地机器10可用。

[0402] 现在看图17，描述了本地机器10上的会话的持续期间请求并保持来自远程机器30的许可所采用步骤的一个实施例的流程图。总的来说，应用流客户端请求许可(步骤1702)。远程机器30接收对许可的请求，验证与该请求相关联的票据，并产生许可(步骤1704)。远程机器30向本地机器10提供许可和与许可相关联的信息(步骤1706)。本地机器10执行结合上述在图7中的步骤214描述的应用。本地机器传输心跳消息，该心跳消息指示本地机器已经执行了应用(步骤1708)。远程机器30接收该心跳消息并验证与心跳信息一起被传输的识别信息(步骤1708)。远程机器30建立与执行的应用和与本地机器10相关联的会话(步骤1710)。建立会话的结果被传输到本地机器10(步骤1712)。如上述结合图7中的步骤216描述的，在整个应用执行期间，本地机器传输心跳消息。本地机器接收对传输的心跳消息的响应(步骤1714)。本地机器传输心跳消息，该心跳消息指示该应用的执行的终止(步骤1716)。远程机器30接收该心跳消息并确定是否移除会话相关数据以及是否释放与本地机器10和终止应用相关联的许可(步骤1718)。远程机器10作出的决定的结果被传输到本地机器10(步骤1720)。

[0403] 现在看图17，更详细的，在本地机器10上的应用流客户端请求许可(步骤1702)。在

一些实施例中，一旦接收到与应用程序相关联的访问信息，本地机器10请求该许可。在这些实施例的一个中，本地机器请求来自远程机器30的许可，该许可授予本地机器10执行应用程序的授权。在一些实施例中，该许可的请求包括从带有访问信息的远程机器30接收的启动票据。在其他实施例中，本地机器10上的应用流客户端552向web接口558传输该请求，并且web接口558向远程机器30传输该请求。仍然在其他实施例中，远程机器上的会话管理子系统510接收并处理该对许可的请求。

[0404] 远程机器30接收对该许可的请求，验证与该请求相关联的票据，并产生许可(步骤1704)。在一个实施例中，远程机器30验证本地机器10被授权以执行该应用。在另一个实施例中，远程机器30确定本地机器10是否已经与存在的许可相关联。仍然在另一个实施例中，远程机器30确定本地机器10与存在的许可相关联并为本地机器10提供标识符，用于会话管理服务器562管理该存在的许可。在又一个实施例中，远程机器30产生并向本地机器10提供新许可、会话标识符、以及管理该新许可的会话管理服务器562的标识。

[0405] 在一些实施例中，远程机器30使用许可管理子系统1510来响应在一个实施例中的许可请求。许可管理子系统1510接收许可请求。该请求可用于特征许可或者用于连接许可。该许可管理子系统1510确定该许可已是否经被授予，即该特征已经被开始或者用于本地机器的连接已经存在。如果该许可已经被授予，该许可管理子系统1510向该许可请求者发送“授予”事件。如果许可没有事先授予，该许可管理子系统1510确定本地许可，即，已经永久指定给该远程机器30的许可，是否可用。在一些实施例中，该许可管理子系统1510通过检查本地存储器来执行该确定。如果本地许可是可用的，即，远程机器30的许可永久分配比当前授予多，该许可管理子系统1510向该许可请求者发送“授予”事件。

[0406] 远程机器30向本地机器10提供许可和与该许可相关联的信息(步骤1706)。在一个实施例中，一旦从远程机器30接收到该许可、该会话标识符以及该会话管理服务器562的标识，本地机器10执行该应用。如结合上述图7中的步骤214所描述的，本地机器10可执行该应用。本地机器传输心跳消息，该心跳消息指示本地机器已经执行应用(步骤1708)。在一个实施例中，本地机器向传输该心跳消息到会话管理服务器562的远程机器30传输该心跳消息。在另一个实施例中，响应于从远程机器30接收的会话管理服务器562的标识符，本地机器10直接向会话管理服务器562传输心跳消息。

[0407] 远程机器30接收该心跳消息并验证与该心跳消息一起传输的识别信息(步骤1708)。在一个实施例中，远程机器30’为会话管理服务器562。在另一个实施例中，会话管理服务器562验证服务器标识符，该标识符与心跳消息一起由本地机器10提供，仍然在另一个实施例中，服务器标识符是远程机器30向本地机器10提供的标识符。

[0408] 远程机器30建立与所执行的应用和本地机器10相关联的会话(步骤1710)。在一个实施例中，会话管理服务器562一旦接收到该心跳消息，则建立与执行的应用相关联的新会话。在另一个实施例中，第三方远程机器30建立新会话。在一些实施例中，会话管理服务器562在生成新会话时，存储会话相关信息。

[0409] 建立会话的结果被传输到本地机器10(步骤1712)。在一些实施例中，该结果确定会话的建立。在其他实施例中，该结果识别该应用或与会话相关联的应用。如上述结合图7中步骤216描述的，本地机器在该应用执行的整个期间传输心跳消息。在一个实施例中，本地机器10在有规律的间隔连续的传输心跳消息，在整个应用程序的执行期间定期的时间间

隔到会话管理服务器562。本地机器接收对已传输的心跳消息的响应(步骤1714)。在一个实施例中，本地机器10从会话管理服务器562接收心跳消息的确认。在另一个实施例中，响应于经由会话管理服务器562的心跳消息的接收，本地机器10从会话管理服务器562接收用于执行的命令。

[0410] 本地机器传输心跳消息，该心跳消息指示该应用的执行的终止(步骤1716)。远程机器30接收该心跳消息并决定是否移除会话相关数据以及是否释放与本地机器10和终止的应用相关联的许可(步骤1718)。远程机器30作出的决定的结果被传输到本地机器10(步骤1720)。

[0411] 现在看图18，描述了可与管理服务504监控的会话相关联的状态的实施例的框图。在一个实施例中，在管理服务504上的会话保持子系统510监控本地机器10的会话并为该会话分配状态。在另一个实施例中，会话保持子系统510保持许可相关数据列表，该列表包括与本地机器相关联的标识符、与会话相关联的标识符、会话状态以及指示远程机器30从本地机器10接收消息的上一次的时间戳。在一些实施例中，会话保持子系统510包括会话监控线程。在这些实施例的一个中，会话监控线程在定期的许可超时间隔唤醒，以扫描该许可相关数据的列表和更新会话的会话状态。

[0412] 会话可处于的第一状态是激活以及被许可状态。在一个实施例中，当在此状态中，本地机器10已经维持了授权应用的执行的有效的许可。在另一个实施例中，会话管理服务器562维持会话相关数据。在一些实施例中，会话管理服务器562在第二远程机器上存储会话相关数据。在一个实施例中，当本地机器10最初执行应用时，该用于本地机器的会话处于激活和许可状态。

[0413] 会话可处于的第二状态是激活和未许可状态。在一个实施例中，当本地机器10传输心跳消息失败并且到本地机器10的许可已经终止时，会话处于此状态。在一个实施例中，如果会话处于这个状态，尽管该许可已经过期，用于会话的不充足的时间已经过去以致过期，并且该会话视为激活。在一些实施例中，当会话处于该状态，远程机器30或会话管理服务器562可存储代表本地机器10的会话相关数据。在其它实施例中，如果本地机器10在会话终止之前传输心跳消息，则会话相关数据被传输到具有新许可的本地机器10，并且该会话返回到激活和被许可状态。在一个实施例中，远程机器30使用会话标识符和与本地机器相关的标识符以验证会话没有终止，并向本地机器提供适当的会话相关数据。

[0414] 会话可处于的第三状态是未连接和不存在状态。当会话终止，删除会话相关数据。

[0415] 会话可处于的第四状态是重连接和未许可状态。在一个实施例中，当本地机器10上的会话终止，则删除会话相关数据。在另一个实施例中，当本地机器10传输新的心跳信息时，产生用于本地机器10的新会话标识符和本地机器标识符。在一些实施例中，本地机器10重授权远程机器30，接收新许可，并进入激活和被许可状态。

[0416] 表3总结了与会话相关的状态。

[0417] 会话状态 描述

[0418] 激活\被许可 操作的正常模式

[0419] 激活\未被许可 缺少心跳的持续时间〉

[0420] 许可超时

[0421] 并且

[0422] 缺少心跳的持续时间

[0423] 〈会话超时

[0424] 重连接\未被许可 缺少心跳的持续时间〉

[0425] 会话超时

[0426] 或者寄载会话的CPS/RADE下线和重新在线(down

[0427] and back online)

[0428] 表3

[0429] 在一些实施例中，包机制启用与应用程序相关联的多个应用文件的建立。在这些实施例的一个中，包机制启用多个应用文件的标识。在这些实施例的另一个中，包机制启用将单个的应用文件分组到多个应用文件中。仍然在这些实施例的一个中，包机制启用在例如文件服务器或应用服务器的远程机器上寄载该多个应用文件。

[0430] 在一个实施例中，包机制在被描述为“中间集结机器”的远程机器上执行。在另一个实施例中，包机制在“干净的机器”上执行。干净的机器可为其上只安装了操作系统的远程机器，而没有附加的软件、驱动器、登记入口或其他文件。仍然在另一个实施例中，包机制在远程机器上执行，该远程机器类似应用程序可在其上执行的本地机器。在一些实施例中，包机制在其上执行的远程机器包括隔离环境，即使远程机器本身不是干净的机器，该隔离环境可提供应用程序可安装在其中的干净的机器环境。

[0431] 在一个实施例中，该多个应用文件被指定作为“包”。在其他实施例中，该包可为存储多个应用文件的档案文件。仍然在另一个实施例中，包可为档案文件，该档案文件存储该多个应用文件和文件，该文件包括与在该多个应用文件中的至少一个文件相关联的元数据。在一些实施例中，包包括多个应用文件，该多个应用文件包括应用程序。在其它实施例中，包包括多个应用文件，该多个应用文件包括一组应用程序。在又一些实施例中，包包括多个应用文件，该多个应用文件包括应用程序和执行该应用程序需要的先决条件。

[0432] 在一个实施例中，包机制在隔离环境中开始安装程序的执行。在另一个实施例中，包机制监控安装程序产生的对隔离环境的改变。仍然在另一个实施例中，包机制通过安装程序监控在隔离环境中文件的生成。仍然在另一个实施例中，包机制通过安装程序监控在隔离环境中的文件的修改。在一些实施例中，该多个应用文件包括通过安装程序生成或修改的文件。在其他实施例中，包机制实现文件系统过滤器驱动器564以监控该隔离环境。

[0433] 在一些实施例中，包机制可产生多种该多个应用文件，每种包括应用程序的不同版本，该应用程序被配置用于在不同目标环境中执行。在这些实施例的一个中，配置多个应用文件从而在本地机器上执行，该本地机器有特定的操作系统、修订级、语言配置和主驱动器(例如配置一个多个应用文件以在本地机器上执行，该本地机器有带有修订级SP2及以上的Windows XP Professional操作系统、使用英语并有主驱动器C:\)。在这些实施例的另一个中，多于一种多个应用文件可合并为一个档案文件。仍然在这些实施例的另一个中，每种多个应用文件可被指定为“目标”。仍然在这些实施例的另一个中，包含一种或多种多个应用文件的档案文件可被指定为“包”。

[0434] 现在看图9，描述了包括两个目标的包的框图，每个目标包括多个应用文件，该多个应用文件包括应用。在图19中，该应用程序“Foo”被包入两个目标中。两个目标间的差别是“目标语言”。特别地，目标1支持“英语”，并且目标2支持“德语”。在一个实施例中，可用应

用程序的列举可列出该应用程序“Foo”。在另一个实施例中，该合适的多个文件被传输到请求访问该应用程序的本地机器。仍然在另一个实施例中，响应于本地机器的评价，做出向本地机器传输特定目标的决定。仍然在另一个实施例中，与包相关联的文件识别与包中目标相关联的、本地机器上的执行请求的至少一个特征。

[0435] 在一些实施例中，包机制530通过执行与应用程序相关联的安装程序，预备用于流式传输的应用程序。在这些实施例的一个中，包机制产生远程机器30上的隔离环境，该包机制在该远程机器30上执行。在这些实施例的另一个中，包机制执行在隔离环境中的应用程序。仍然在这些实施例的另一个中，包机制识别由安装程序产生或修改的多个应用文件。仍然在这些实施例的另一个中，包机制建立包括该多个应用文件的档案文件。在这些实施例的一个中，包机制建立包括该多个应用文件的.CAB文件。在这些实施例的另一个中，包机制建立目录并在该目录中存储该多个应用文件。在一些实施例中，包机制在文件服务器或其他远程机器30上存储该多个应用文件。在其他实施例中，该包机制在多个远程机器上存储该多个应用文件。

[0436] 现在看图20，描述了基于策略的方法中采用的步骤的一个实施例的流程图，用于有效的安装应用程序而不用重新启动操作系统。总的来说，包机制在隔离环境中执行安装程序，该安装程序将与第二应用相关联的至少一个应用文件安装到隔离环境中(步骤2002)。通过安装程序对至少一个应用编程接口(API)的调用被拦截，在重新启动操作系统后该调用需要动作的执行(步骤2004)。不用重新启动操作系统，执行至少一个拦截的调用的动作(步骤2006)。接收至少一个应用文件的文件类型的标识(步骤2008)。响应于识别的文件类型，至少一个执行方法与该至少一个被安装的应用文件相关联(步骤2010)。该至少一个被安装的应用文件被存储在至少一个服务器上(步骤2012)。产生第二应用、该至少一个被安装的应用文件、该至少一个服务器的位置以及该至少一个执行方法的列举(步骤2014)。

[0437] 现在看图20，更详细的，包机制在隔离环境中执行安装程序，该安装程序将与第二应用相关联的至少一个应用文件安装到隔离环境中(步骤2002)。在一个实施例中，在隔离环境中执行安装程序启用包机制来隔离安装程序对本地机器上的文件或登记所作出的改变。在另一个实施例中，包机制拦截安装程序请求的改变，并将该改变重定向到隔离环境以防止改变发生在本地机器上。仍然在另一个实施例中，包机制在隔离环境中执行第二安装程序，该第二应用向隔离环境中安装与第三应用相关联的至少一个应用文件。

[0438] 在一些实施例中，包机制在隔离环境中执行该安装程序，该安装程序在隔离环境内执行与应用相关联的至少一个可执行应用。在一个实施例中，其中安装程序执行应用，应用的执行启用第二应用的安装。

[0439] 在这些实施例的另一个中，除了安装程序的执行之外，应用的安装需要执行至少一个可执行应用。仍然在这些实施例的另一个中，除了安装程序的执行之外，应用的安装需要因特网浏览器应用的执行。在一些实施例中，执行安装程序以安装程序，并且安装程序的执行包括第二程序的执行，需要该第二程序以安装该程序。在这些实施例的一个中，该程序是插件。在这些实施例的另一个中，该程序是ActiveX部件。仍然在这些实施例的另一个中，该程序是Flash部件。仍然在这些实施例的另一个中，该程序是定制的工具栏，例如Yahoo!或者Google工具栏。在其他实施例中，该程序是安装到第二程序中的部件，并且独立于第二

程序不可执行。

[0440] 通过安装程序对至少一个应用编程接口(API)的调用被拦截，在重新启动操作系统后该调用需要动作的执行(步骤2004)。不用重新启动操作系统执行至少一个拦截的调用的动作(步骤2006)。在一些实施例中，动作的执行包括执行安装期间被修改的登记入口的动作。关于不重新启动操作系统时至少一个拦截调用的执行的进一步的细节下面结合图25提供。

[0441] 接收至少一个应用文件的文件类型的标识(步骤2008)。响应于识别的文件类型，至少一个执行方法与该至少一个被安装的应用文件相关联(步骤2010)。在一个实施例中，该至少一个执行方法启用至少一个应用文件流式传输到客户端。在另一个实施例中，该至少一个执行方法启用客户端上的该至少一个被安装的应用文件执行。仍然在另一个实施例中，该至少一个执行方法启用服务器上的该至少一个被安装的应用文件执行。在又一个实施例中，该至少一个执行方法启用该至少一个应用文件流式传输到服务器。

[0442] 该至少一个被安装的应用文件被存储在至少一个服务器上(步骤2012)。在一些实施例中，在至少一个服务器上存储该至少一个被安装的应用文件之前，该被安装的应用程序在该隔离环境中被执行。在这些实施例的一个中，响应于该被安装的应用程序的执行，产生附加的应用文件。在这些实施例的另一个中，产生数据文件。仍然在这些实施例的另一个中，被安装的应用程序需要信息以完成安装，在最初的安装过程之后需要该信息。仍然在这些实施例的另一个中，需要诸如软件产品标识符、许可标识符或者其他证书的信息。

[0443] 在一些实施例中，提供标识符以识别在该至少一个服务器上的该至少一个被安装的应用文件的位置。在这些实施例的一个中，标识符符合通用命名标准(UNC)。在其他实施例中，在例如.CAB文件的档案文件中放置该至少一个被安装的应用文件。在这些实施例的一个中，多个应用文件被存储在档案文件中，并且该档案文件被存储在该至少一个服务器上。仍然在这些实施例的另一个中，该至少一个被安装的应用文件被存储在多个服务器上。仍然在其他实施例中，该至少一个应用文件被放置在存储应用文件的目录中。

[0444] 产生第二应用、该至少一个被安装的应用文件、该至少一个服务器的位置以及该至少一个执行方法的列举(步骤2014)。在一些实施例中，该列举被存储在文件中。在其他实施例中，该列举被存储在清单文件中。仍然在其他实施例中，该列举被存储在XML文件中。

[0445] 在一个实施例中，产生多个应用、与多个应用中的每个相关联的多个被安装的应用文件以及存储多个被安装的应用文件的至少一个服务器的位置的列举。在其他实施例中，产生列举包括第二应用和多个被安装的应用文件之间的关联。仍然在另一个实施例中，产生的列举包括在第二应用和压缩文件间的关联，该压缩文件包括至少一个被安装的应用文件。

[0446] 现在看图21，描述了基于策略的方法中使用的步骤的一个实施例的流程图，用于不重启操作系统而安装应用程序。总的来说，包机制在隔离环境中执行安装程序，该安装程序向该隔离环境安装至少一个应用文件，该应用文件与第二应用相关联(步骤2102)。拦截由安装程序对至少一个应用编程接口(API)的调用，在重启操作系统之后，该调用需要动作的执行(步骤2104)。该至少一个被拦截的嗲用的动作不用重启操作系统而被执行(步骤2106)。接收该至少一个应用文件的特征的标识(步骤2108)。响应于被识别的特征，至少一个执行必须条件与该至少一个被安装的应用文件相关联(步骤2110)。该至少一个被安装的

应用文件被存储在至少一个服务器上(步骤2112)。产生第二应用、该至少一个被安装的应用文件、该至少一个服务器的位置以及该至少一个执行必须条件的列举(步骤2114)。

[0447] 现在看图21,更详细的,包机制在隔离环境中执行安装程序,该安装程序向该隔离环境中安装至少一个应用文件,该应用文件与第二应用相关联(步骤2102)。在一个实施例中,在该隔离环境中执行该安装程序启用包机制来隔离由安装程序产生对本地机器上的文件或登记表的改变。在另一个实施例中,包机制拦截安装程序请求的改变并将该改变重定向到隔离环境中以防止该改变发生在本地机器上。在另一个实施例中,包机制在隔离环境中执行第二安装程序,该第二应用将至少一个应用文件安装到隔离环境中,该至少一个应用文件与第三应用相关联。

[0448] 在一些实施例中,包机制在隔离环境中执行安装程序,该安装程序在隔离环境内执行至少一个可执行应用,该至少一个可执行应用与应用相关联。在一个实施例中,其中安装程序执行应用,应用的执行启用安装第二应用。在这些实施例的另一个中,除了安装程序的执行之外,安装应用需要该至少一个可执行应用的执行。仍然在这些实施例的另一个中,除了安装程序的执行之外,应用的执行需要因特网浏览器应用的执行。

[0449] 提前看图23,描述了系统的一个实施例的框图,该系统包括执行安装程序2350到隔离环境532内的包机制530,以及与包机制530通信的文件系统过滤器驱动器534,和隔离系统532。

[0450] 在一个实施例中,包机制530通过向隔离环境532中安装应用程序来产生包(如上述结合图21所描述的)。在另一个实施例中,包机制530通过执行安装程序2350向隔离环境532中安装应用程序。在一些实施例中,包机制530包括图形用户界面。在这些实施例的一个中,图形用户界面启用包机制530的用户通过包机制530来定制包的产生。在这些实施例的另一个中,包机制与访问控制台520上的图形用户界面通信,使得访问控制台520的用户通过包机制530定制包的产生。

[0451] 在一些实施例中,文件系统过滤器驱动器532启用在隔离环境532中安装应用程序。在这些实施例的一个中,文件系统过滤器驱动器532拦截安装程序2350的请求。在这些实施例的另一个中,文件系统过滤器驱动器532重定向该安装程序2350的请求到隔离环境532中。仍然在这些实施例的另一个中,文件系统过滤器驱动器532存储安装程序2350做出的请求的记录。在这些实施例的又一个中,文件系统过滤器驱动器532存储安装程序2350建立或修改的文件的副本。在一些实施例中,文件系统过滤器驱动器532产生的被存储的记录作为多个应用文件被存储在一些,上述多个应用文件包括应用程序。在其它实施例中,该多个应用文件被存储在文件服务器540上。

[0452] 现在回头看图21,拦截安装程序对至少一个应用编程接口(API)的调用,该调用在重启操作系统之后需要动作的执行(步骤2104)。不用重新操作系统而执行该至少一个被拦截的调用的动作(步骤2106)。在一些实施例中,动作的执行包括驱动器的安装,该驱动器被配置为在启动计算机系统时被启动。在其他实施例中,动作的执行包括执行在安装期间修改的登记入口的动作。

[0453] 接收至少一个应用文件的特征的标识(步骤2108)。在一些实施例中,接收操作系统类型的标识。在其他实施例中,接收操作系统使用的语言的标识。仍然在其他实施例中,接收第二应用的版本的标识。

[0454] 响应于被识别的特征，至少一个执行的必要条件与至少一个被安装的应用文件相关联(步骤2110)。在一个实施例中，响应于到特征的策略的应用，该至少一个执行的必要条件与该至少一个被安装的应用文件相关联。在另一个实施例中，脚本与该至少一个被安装的应用文件相关联，该脚本包括可执行程序，该可执行程序确定客户端上的至少一个执行必要条件的存在。现在提前参见图22，屏幕图片描述了在本地机器上执行的脚本列举的一个实施例。脚本2202的类型指示何时脚本将被执行，例如，或者在应用程序的执行之前，或应用程序的执行终止之后。隔离指示器24指示了脚本是否应该在本地机器10上的隔离环境中被执行。如图22所示，在一些实施例中，在该多个应用文件被打包到一起并存储在寄载该多个应用文件的远程机器30’上时，脚本与该应用程序相关联。

[0455] 在一些实施例中，该至少一个执行必要条件需要在执行至少一个被安装的应用文件的系统上安装操作系统的版本。在其它实施例中，该至少一个执行必要条件需要在执行至少一个被安装的应用文件的系统上安装第二应用的版本。在又一些实施例中，指令与至少一个被安装的应用文件相关联，该指令指示通过客户端所使用的第二被安装应用文件不能满足至少一个执行的必要条件。在又一些实施例中，指令与至少一个被安装的应用文件相关联，该指令指示第二执行方法不能满足该至少一个执行必要条件，该第二执行方法用于在客户端上的至少一个被安装的应用文件的执行。在这些实施例的一个中，执行方法与该至少一个被安装的应用程序文件相关联，该执行方法授权流式传输包括第二应用的多个应用文件到本地机器用于在本地机器上的执行。在这些实施例的另一个中，本地机器的评价识别至少一个特征，该特征与至少一个被安装的没有包括在本地机器上的应用文件相关联。仍然在这些实施例的另一个中，撤回用于该多个应用文件的执行的授权。在这些实施的又一个中，提供第二执行方法用于执行该多个应用文件，该第二执行方法启用远程机器上该多个应用文件的执行以及从远程机器向本地机器传输应用输出数据。

[0456] 该至少一个被安装的应用文件被存储在至少一个服务器上(步骤2112)。在一些实施例中，在至少一个服务器上存储该至少一个被安装的应用文件之前，该被安装的应用程序在该隔离环境中被执行。在这些实施例的一个中，响应于该被安装的应用程序的执行产生附加的文件。在这些实施例的另一个中，产生数据文件。仍然在这些实施例的另一个中，该被安装的应用程序需要信息以完成安装，在初始安装过程之后需要这些信息。在这些实施例的又一个中，需要信息，例如软件产品标识符、许可标识符、或其它证书。

[0457] 在一些实施例中，提供标识符识别至少一个服务器上的至少一个被安装的应用文件的位置。在这些实施例的一个中，标识符服从通用命名标准(UNC)。在其他实施例中，该至少一个被安装的应用文件被放置于档案文件中，例如.CAB文件。在这些实施例的一个中，多个应用文件被存储在档案文件中，并且该档案文件被存储在至少一个服务器上。仍然在这些实施例的另一个中，至少一个被安装的应用文件被存储在多个服务器上。仍然在其他实施例中，至少一个被安装的应用文件被放置在存储应用文件的目录中。

[0458] 产生第二应用、至少一个被安装的应用文件、至少一个服务器的位置以及至少一个执行必要条件的列举(步骤2114)。在一些实施例中，该列举被存储在文件中。在其他实施例中，该列举被存储在清单文件中。仍然在其他实施例中，该列举被存储在XML文件中。

[0459] 在一个实施例中，产生多个应用、与多个应用中的每个相关联的多个被安装应用文件、以及存储该多个被安装的应用文件的至少一个服务器的位置的列举。在另一个实施

例中，产生的列表包括在第二应用和多个被安装的应用文件间的关联。仍然在另一个实施例中，产生的列举包括在第二应用和压缩文件间的关联，该压缩文件包括至少一个被安装的应用文件。

[0460] 回头看步骤2106，此处没有重启操作系统而执行至少一个被拦截的调用的动作，在一些实施例中，提供虚拟化的安装和执行环境，其在执行被安装的应用之前移除重启系统的需要。

[0461] 现在看图24，描述了实施例的一个流程图，其中安装程序的执行需要重启本地机器上的操作系统，安装程序在该本地机器上执行。传统应用安装复制文件到该应用被安装的远程机器上(步骤2402)。在一些实施例中，拷贝文件可使得远程机器重启。应用安装试着将至少一个文件拷贝以锁定文件(步骤2404)。在一个实施例中，当操作系统被执行(或“重启”)时锁定的文件只能被写入。MOVE_FILE_DELAY_UNTIL_REBOOT选项可在MoveFileEx()Win32Api中被设置(步骤2406)，并且应用安装调用系统关闭/重启功能(步骤2408)。随着重启，在重启时，初始锁定文件接着被安装(步骤2410)。

[0462] 现在看图25，描述了远程机器30的一个实施例的框图，包机制安装应用文件到该远程机器上。远程机器30包括系统资源2502、系统API 2504和用于安装应用的应用安装程序2506。远程机器30也包括钩住功能(function-hooking)机制2508，安装后处理器模块2510以及应用隔离环境2512。在一些实施例中，向隔离环境2512中安装应用程序启用不使用重启远程机器30而安装。在这些实施例的一个中，在隔离环境2512中实现的对系统资源2502做的改变不改变远程机器30上的相应系统资源2502。由于在远程机器30上的系统资源没有改变，不需要重启机器以保护系统资源被不合适的改变。

[0463] 现在看图25，更详细的，系统资源2502可以包括登记人口、系统DLL以及其他被锁定文件，当远程机器30被执行时，阻止操作系统写入该被锁定文件。系统API 2504包括用于重新启动系统的API，这些API被应用安装程序2506调用并被钩住功能机制2508钩住以防止重新启动远程机器30。

[0464] 应用隔离环境2512向应用安装程序2506提供带有操作系统资源视图的环境。在一个实施例中，应用隔离环境2512是隔离环境556。在一些实施例中，应用隔离环境2512提供诸如文件系统、登记和命名对象的操作系统资源的虚拟化。在一个实施例中，应用安装程序2506在应用隔离环境2512中执行。在另一个实施例中，应用安装程序2506向应用隔离环境2512中安装应用程序。仍然在另一个实施例中，应用安装程序2506在应用隔离环境2512外执行，并在应用隔离环境2512内安装应用程序。

[0465] 在一些实施例中，当应用安装程序2506向应用隔离环境2512中安装应用时，应用隔离环境2512避开对于重启远程机器30的需要。在一个实施例中，应用隔离环境2512拦截向锁定的文件拷贝应用文件的请求。在另一个实施例中，应用隔离环境2512将该拷贝应用文件的请求重定向到未锁定的文件。在另一个实施例中，应用隔离环境将该拷贝应用文件的请求重定向到虚拟文件。在又一个实施例中，重定向拷贝应用文件的请求启用不需要重新启动远程机器30而安装应用文件。在一个实例中，如果应用安装程序2506试图写到锁定文件，例如C:\windows\system32\mfc40.dll，应用隔离环境2512拦截该请求并将该文件重定向到另一个未锁定的位置。这种避免锁定文件的能力意味着文件不必使用MoveFileEX()API以及MOVE_FILE_DELAY_UNTIL_REBOOT标记而被安装。这种能力消除了重启远程机器

30的需要。

[0466] 在一个实施例中，钩住功能机制2508是文件系统过滤器驱动器564。在另一个实施例中，文件系统过滤器驱动器564包括钩住功能机制2508。仍然在另一个实施例中，钩住功能机制2508拦截来自应用安装程序2506的请求以重启远程机器30。在一些实施例中，应用隔离环境2512提供向未锁定文件拷贝应用文件。然而，应用隔离环境2512不处理由应用安装程序2506提出的用于重启远程机器30的请求。钩住功能机制2508可提供实施动作的功能并且对应于应用的安装程序2506。

[0467] 应用隔离环境2512启用向未锁定文件拷贝应用文件。然而，在一些实施例中，需要用于安装应用的其它动作，并且一旦重启，这些动作可发生。阻止重启不阻止在安装过程中完成这些动作的需要。钩住功能机制2508可以提供用于执行和安装应用相关联的动作的功能性。

[0468] 例如，在应用安装期间，可写入登记入口，例如HKLM\SYSTEM\CurrentControlSet\Control\Session_Manager\Pending-Fi leRenameOperations。其他服务可安装服务或驱动器，一旦重启机器需要开始的这些服务或驱动器。安装后处理器模块2510识别应用文件，这些应用文件在安装期间已被修改，并完成与应用文件相关联的动作。

[0469] 现在看图26，描述了步骤的实施例的流程图，随着上述步骤在应用隔离环境2512中安装应用。应用隔离环境2512还将为应用安装程序提供服务器操作系统的虚拟视图(步骤2602)。在服务器上涉及系统重启和关闭的API被钩住(步骤2604)以阻止引起应用安装程序2506的重启。应用安装程序2506请求文件拷贝操作以锁定文件，该请求被拦截并被重定向到无冲突位置(步骤2606)。当应用安装程序2506试图通过调用系统API重启，该请求被拦截并且该重启被阻止(步骤2608)。安装后处理器模块2510执行通常在重启后发生的动作(步骤2610)并且不用重启远程机器30，该应用可接着在应用隔离环境2512中被执行(步骤2612)。

[0470] 在一些实施例中，在向应用隔离环境2512中安装应用程序之后，包机制识别在应用程序安装期间建立或修改的多个应用文件。在这些实施例的一个中，该多个应用文件被存储在远程机器上。在这些实施例的另一个中，获取该多个应用文件的本地机器可执行该应用程序。

[0471] 在一些实施例中，包机制530在远程机器上执行，该远程机器包括隔离环境532和文件系统过滤器驱动器534，并且包机制530向隔离环境532中安装应用程序。在这些实施例的一个中，远程机器被指定作为“干净的机器”或者“中间集结的机器”。在这些实施例的另一个中，隔离环境532包括应用隔离范围，提供原生资源可修改、虚拟化的实例，由在干净的机器上的操作系统提供该原生资源。仍然在这些实施例的另一个中，隔离环境532包括系统隔离范围，提供原生资源的只读视图。在这些实施例的又一个中，原生资源的只读视图包括文件系统的快照和驻留在干净机器上的登记表。

[0472] 在一个实施例中，转向器拦截用于改变原生资源的请求。在一些实施例中，转向器为文件系统过滤器驱动器534。在另一个实施例中，由包机制530执行的安装程序做出用于改变的请求。仍然在另一个实施例中，需要对原生资源改变以向干净的机器上安装应用程序。在又一个实施例中，转向器将请求重定向到隔离环境532。

[0473] 在一些实施例中，将改变原生资源的请求重定向到隔离环境532导致与应用程序

的安装相关联的改变的隔离。在其他实施例中，改变原生资源的请求被记录并被存储在存储元件中。在这些实施例的一个中，与应用程序的安装相关联的所有改变驻留在存储元件中。在这些实施例的另一个中，获取存储元件的内容并实施驻留在本地机上552的隔离环境556中的原生资源的改变导致本地机器552上的应用程序的安装。

[0474] 在一些实施例中，需要本地机器10的启动前分析。在这些实施例的一个中，本地机器10验证包括在本地机器10中的至少一个特征。在这些实施例的另一个中，在启动前分析确定本地机器10缺少至少一个特征之后，将至少一个特征加入到本地机器中。仍然在这些实施例的另一个中，至少一个特征被包括在寄载应用程序的远程机器上，并且本地机器不包括至少一个特征将阻止应用程序的执行。在又一个实施例中，应用程序需要本地机器上的至少一个特征的存在用于执行。

[0475] 在一些实施例中，包机制启用至少一个特征的标识，该至少一个特征在本地机器的启动前分析中使用。在另一个实施例中，包机制启用至少一个特征与可用于本地机器上执行的应用程序的关联。在另一个其他实施例中，包机制启用可执行脚本与应用程序间关联，本地机器执行该可执行脚本以完成启动前分析。在又一些实施例中，在应用程序的执行之后，需要至少一个特征存在于本地机器上。

[0476] 包机制可提供用于签名多个应用文件的功能。在一个实施例中，签名多个应用文件启用本地机器以验证多个应用文件的完整性。在另一个实施例中，签名多个应用文件防止本地机器执行被破坏的应用程序。在一些实施例中，计算在多个应用文件中的文件的密码校验和，例如MD4hash，MD5hash或者SHA－1hash。

[0477] 在其他实施例中，计算在多个应用文件中的每个文件的密码校验和。在这些实施例的一个中，密码校验和被存储在第二文件中。在这些实施例的另一个中，第二文件与多个应用文件关联。在一些实施例中，第二文件被加到多个应用文件。在其他实施例中，第二文件使用证书，例如X.509证书被签名。仍然在其他实施例中，获取多个应用文件的本地机器使用证书的公开部分验证签名。在又一些实施例中，本地机器接收证书的公开部分以及证书信任列表的标识用于签名的验证。在这些实施例的一个中，本地机器接收登记键值，该登记键值包括证书信任列表的标识。

[0478] 在一个实施例中，包机制提供用于定制隔离环境的功能。在另一个实施例中，包机制提供产生存储隔离环境的定义的文件的功能。仍然在另一个实施例中，包机制包括带有多个应用文件的文件，该多个应用文件包括应用程序。在又一个实施例中，本地机器从远程机器接收带有访问信息的文件。

[0479] 在一些实施例中，多个应用文件被存储在档案文件中。在这些实施例的一个中，档案文件为CAB文件格式。在这些实施例的另一个中，档案文件格式不提供通过文件的短文件名称的应用程序对详细说明的支持。仍然在这些实施例的另一个中，操作系统，例如WINDOWS2000，可能不提供通过文件的短文件名称的应用程序对详细说明的支持。在其他实施例中，操作系统，例如WINDOW XP，提供通过文件的短文件名称的应用程序对详细说明的支持。在这些实施例的一个中，执行文件的请求必须包括校正短文件名称。

[0480] 在一个实施例中，产生映射以将在多个应用文件中的文件的长文件名称与文件的短名称关联。在另一个实施例中，该映射被存储在该多个应用文件中的文件中。仍然在另一个实施例中，仅当文件的长文件名称大于十二个字符，文件有短文件名。在一些实施例中，

短文件名是与文件相关联的虚拟文件名。在这些实施例的一个中，文件被传输到本地机器10用于执行，此处与长文件名一起被存储。在这些实施例的另一个中，本地机器10上的应用文件使用短文件名请求文件的执行。仍然在这些实施例的另一个中，虽然用于执行文件的请求不使用本地机器上的文件的名称(长文件名称)，但映射启用文件的执行。

[0481] 在一些实施例中，包机制530产生映射。在这些实施例的一个中，包机制为有长文件名的文件选择短文件名。在这些实施例的另一个中，包机制530正执行的远程机器30'上的操作系统为有长文件名的文件选择短文件名。仍然在这些实施例的另一个中，选择唯一的短文件名称，该唯一的短文件名称与远程机器30'上的第二短文件名称不冲突。在这些实施例的又一个中，被包机制530执行的安装程序产生包括在长文件名和短文件名间的映射的文件。在其他实施例中，映射被发送到获取该文件的本地机器10。在这些实施例的一个中，本地机器10指向正在执行的文件。

[0482] 下述示范性实施例示出了上述讨论的方法和系统可怎样被用于选择、流式传输到本地机器，并在本地机器上执行包括应用程序的多个文件。这些实施例只意味着示出而非限制。

[0483] 实例1

[0484] 在一个实施例中，本地机器10的用户请求访问应用程序，例如字处理程序、web浏览应用或者电子数据表程序，该应用程序在应用程序列举中被识别。在此实施例的一个例子中，本地机器10执行程序邻近应用，该程序邻近应用从远程机器30接收可用于本地机器10的应用列举。在此实施例的另一个例子中，本地机器10与例如远程机器30"'的web服务器通信以接收该应用列举。通过选择代表被列举的应用程序的图解，本地机器10的用户可请求访问所列举的应用程序。本地机器10的用户可请求访问不是预先安装在本地机器10上的应用程序。

[0485] 本地机器10向远程机器30传输对应用程序的访问请求。本地机器10接收远程机器30"的识别，该远程机器30"提供对包括应用程序的多个应用文件的访问。本地机器10识别至少一个特征，在应用程序的执行中需要上述至少一个特征。在此实施例的一个实例中，本地机器10接收带有远程机器30"的标识的该至少一个特征，该至少一个特征通过远程机器30被传输到本地机器10。在此实施例的另一个实例中，在接收远程机器30"的标识之后，本地机器10从远程机器30"获取该至少一个特征。在接收授权以获取多个应用文件之前，可请求本地机器10以包括至少一个特征。可替代的，在执行多个应用文件之前，可请求本地机器10以包括至少一个特征。在此实施例的一个实例中，在执行该多个应用文件的整个过程中，可请求本地机器10以包括该至少一个特征。

[0486] 一旦本地机器10验证本地机器10包括至少一个特征，本地机器10在该多个应用文件中获取至少一个应用文件并执行获取到的应用文件以执行应用程序。

[0487] 实例2

[0488] 远程机器30从本地机器10接收对应用程序的访问请求。远程机器30验证本地机器10。在该实施例的一个实例中，远程机器30从本地机器10请求证书，例如用户名和密码。在此具体实施例的另一个实例中，远程机器30向本地机器10传输收集代理404。收集代理404收集关于本地机器10的信息并向远程机器30传输此信息用于验证本地机器10。仍然在此实施例的另一个实例中，远程机器30向策略引擎406提供关于本地机器10的信息用于验证本

地机器10。远程机器30可包括策略引擎406。可替代的，远程机器30可与包括策略引擎406的远程机器30’通信。

[0489] 远程机器30选择应用程序的执行方法。远程机器30响应于本地机器10的验证，作出选择。在此实施例的一个实例中，远程机器30向收集的关于本地机器10的信息应用策略。在此实施例的另一个实例中，远程机器30响应于应用到应用程序的策略作出选择。仍然在此实施例的另一个实例中，远程机器30响应于应用到与应用程序相关联的文件类型的策略作出选择。远程机器30可参考文件以作出对应用程序的执行方法的选择。

[0490] 远程机器30可选择应用程序的执行方法，该执行方法启用本地机器10接收应用输出数据，该应用输出数据通过远程机器30’上的应用程序的执行而产生。远程机器30可选择应用程序的执行方法，该执行方法使得本地机器10在获取包括应用程序的多个应用文件之后本地执行应用程序。

[0491] 在一个实施例中，远程机器30选择执行应用程序的执行方法，当通过应用流会话获取包括应用程序的多个应用文件时，该执行方法启用本地机器10本地执行应用程序。在此实施例的一个实例中，本地机器10建立与远程机器的应用流会话，该远程机器寄载多个应用文件，本地机器10通过应用流会话开始对多个应用文件的获取，并且在该多个应用文件中获取第二应用文件时，本地机器10执行在该多个应用文件中的获取到的第一应用文件。在此实施例的另一个实例中，一旦从第一应用文件接收对第二应用文件的访问的请求，本地机器10执行在该多个应用文件中的第一应用文件并获取在多个应用中的第二应用文件。

[0492] 对于被选择的执行方法启用本地机器10在多个应用文件中获取至少一个应用文件的实施例中，其中多个应用文件包括应用程序，远程机器30识别寄载应用程序的远程机器，该应用程序对于本地机器10的访问有用。远程机器30”寄载包括应用程序的多个应用文件。远程机器30”可寄载包括多个应用程序的该多种多个应用文件。在此实施例的一个实例中，远程机器30”寄载多个应用文件，此多个应用文件用于应用程序的不同版本的每一个。

[0493] 远程机器30”寄载与多个应用文件相关联的文件，多个应用文件包括带有应用程序说明的特定应用程序。在向机器传输该多个应用文件之前，文件也可识别将在机器上被识别的一个或多个执行必要条件。文件可进一步包括远程机器30”的网络上的位置的标识。在此实施例的一个实例中，远程机器30参考文件以识别远程机器30”的网络上的位置。

[0494] 远程机器30选择远程机器30”。远程机器30可以选择具有本地机器10可访问的网络上的位置的远程机器30”。远程机器30可选择远程机器30”，该远程机器30”寄载与本地机器10兼容的应用程序的版本。响应于接收到对于应用程序的访问请求，远程机器30向本地机器10传输被选择的执行应用程序的方法的识别以及远程机器30”的标识。远程机器30也可向本地机器10传输文件。

[0495] 实例3

[0496] 在一个实施例中，本地机器10接收被选择的执行应用程序的方法的标识以及远程机器30”的标识，该远程机器30”提供对多个应用文件的访问，多个应用文件包括应用程序。本地机器10验证对应用程序访问的授权。在此实施例的一个实例中，本地机器10执行自身的启动前分析。本地机器10识别至少一个特征，并验证本地机器10上的该至少一个特征的存在。该至少一个特征是维持授权以访问并执行应用程序的必要条件。验证本地机器10上

的至少一个特征的存在可确保本地机器10的特征和应用程序的系统需求间的兼容性，并附加的确保与安全策略或者许可协定的适应性。

[0497] 一旦成功完成启动前分析，本地机器10建立与远程机器30”的应用流会话，提供对该多个应用文件的访问。应用流会话可为任何连接，通过该连接本地机器10可请求并接收该多个应用文件中的文件。在获取该多个应用文件中的所有文件之前，应用流会话的建立可以启动本地机器10执行在该多个应用文件中的第一应用文件。当连续的在该多个应用文件中获取附加文件时，本地机器10可开始应用程序的执行。可替代的，当从档案文件中提取第二应用文件时，本地机器10可在档案文件中获取该多个应用文件并执行第一被提取的应用文件。

[0498] 实例4

[0499] 在一个实施例中，本地机器10上的应用流客户端552从远程机器30获取多个应用文件。应用流客户端包括流服务554、隔离环境556以及文件系统过滤器驱动器564。流服务554建立与远程机器30的应用流会话，用于请求和获取该多个应用文件。流服务554在隔离环境556中执行应用文件。通过拦截来自于执行应用文件的请求并将该请求重定向到隔离环境556中，文件系统过滤器驱动器564启用在隔离环境556中执行应用文件。

[0500] 在该实施例的一个实例中，流服务554获取包括该多个应用文件的档案文件，该该多个应用文件包括应用程序。流服务554从档案文件中提取来自于该多个应用文件的第一应用文件。第一应用文件可为可执行文件。流服务554可在隔离环境556中执行第一应用文件。第一应用文件的执行可开始应用程序的执行。

[0501] 在另一个实施例中，在隔离环境556内执行的第一应用文件从本地机器10请求该多个应用文件的列举。文件系统过滤器驱动器564拦截用于列举的请求并将该请求重定向到流服务554。在流服务554获取该多个应用文件的实施例中，流服务554可产生该多个应用文件的列举。在实施例中，此处流服务554获取包括该多个应用文件的档案文件，流服务554响应于包括在被获取的档案文件中的列举而产生该多个应用文件的列举。在其他实施例中，当该多个应用文件中的至少一个应用文件驻留在远程机器30上，并且没有通过流服务554获取到本地机器10时，流服务554只获取该多个应用文件的列举。在这些实施例中，响应于获取到的列举，流服务554可产生该多个应用文件的列举。在这些实施例的一个实例中，虽然只有列举驻留在本地机器10上，流服务554向第一应用文件指示该多个应用文件驻留在本地机器10上。

[0502] 实例5

[0503] 在一个实施例中，在隔离环境556中执行的第一应用文件从本地机器10请求访问通过该多个应用文件的列举被识别的文件。如果被请求的文件驻留在第一应用文件可访问的隔离环境556中的用户范围内，则第一应用文件访问被请求的文件。

[0504] 如果被请求的文件没有驻留在用户范围或隔离环境556内，则文件系统过滤器驱动器564拦截该请求并将该请求重定向到流服务554。如果被请求的文件为包括该多个应用文件的档案文件内的文件，则流服务554提取被请求的文件并将该被请求的文件存储在本地机器10上。流服务554可在隔离环境556中存储文件。当文件被存储在隔离环境556中时，满足对于文件的请求。

[0505] 如果被请求的文件没有驻留在隔离环境556或包括该多个应用文件的档案文件

中，则流服务554从远程机器30请求文件。流服务554通过应用流会话可从远程机器30接收文件。流服务554在隔离环境556中存储接收的文件。当文件被存储在隔离环境556中时，满足对于文件的请求。

[0506] 在此实施例的一个实例中，第二应用文件在隔离环境556中的第二用户范围内执行。第二应用文件请求访问初始被第一应用文件请求的文件。如果被请求文件的副本没有驻留在第二用户范围内，使用存储在隔离环境556中的被请求文件的副本满足对于应用文件的请求。

[0507] 实例6

[0508] 在一个实施例中，本地机器10从远程机器30接收应用程序执行选择的方法的标识以及远程机器30'的识别，该远程机器30'提供对多个应用文件的访问，该多个应用文件包括应用程序。本地机器10成功完成本地机器10的启动前分析。本地机器10从远程机器30接收许可，该许可授权应用程序的执行。在此实施例的一个实例中，许可需要本地机器10向会话管理服务器562传输心跳消息以维持执行应用程序的授权。心跳消息可包括指示应用程序开始执行的消息，应用程序执行终止的消息，以及整个应用程序执行过程中以周期性为基础的被发送的消息。心跳消息也可包括关于本地机器10状态的消息，例如本地机器10何时连接到网络或者本地机器10何时终止与网络的连接。在此实施例的另一个实例中，许可指定事先确定的时间周期，在此时间周期内本地机器10已被授权执行应用程序。

[0509] 本地机器10建立与远程机器30'的应用流会话，并在该多个应用文件中获取至少一个应用文件。在至少一个应用文件执行期间，在实施例中此处被接收的许可需要心跳消息的传输，本地机器10向会话管理服务器562传输心跳消息以维持至少一个应用文件的执行的授权。

[0510] 实例7

[0511] 在一个实施例中，本地机器10接收执行应用程序的选择的方法的标识以及远程机器30'的识别，该远程机器提供对包括应用程序的多个应用文件的访问。本地机器10成功完成本地机器10的启动前分析。本地机器10接收指定预定时间周期的许可，在此时间周期内本地机器10已授权执行应用程序。

[0512] 本地机器10建立与远程机器30'的应用流会话，并在该多个应用文件中获取至少一个应用文件。在此实施例的一个实例中，本地机器获取该多个应用文件的子集，当本地机器10没有被连接到网络时，该子集包括执行应用程序必须的每个文件。本地机器10在本地机器10的高速缓存中存储该子集。

[0513] 在预先确定的时间周期的时间点，本地机器10从网络上断开，并从本地机器10的用户接收访问应用程序的请求。在此实施例的一个实例中，本地机器10是例如膝上型电脑的设备，并且本地机器10的用户处于禁止网络连接的环境，例如飞机。一旦从用户接收请求，本地机器10可从高速缓存中获取来自该多个应用文件的应用文件，并执行该应用程序。

[0514] 实例8

[0515] 在另一个实施例中，本地机器10接收执行应用程序的选择的方法的标识，以及远程机器30'的标识，该远程机器提供对包括应用程序的多个应用文件的访问。本地机器10可接收驻留在本地机器10上的第一客户端代理的标识以执行该多个应用文件的获取，例如应用流客户端。

[0516] 在此实施例的一个实例中，本地机器10不能成功完成自身的启动前分析。本地机器10可能缺少与应用程序的需求兼容所需的特征，例如特定的设备驱动器或操作系统。本地机器10可以缺少服从安全策略必需的特征，例如，用于授权访问专用网络或者在特定激活目录中的成员资格。本地机器10可能是与应用程序的需要不兼容的机器类型，例如个人数字助理，其试图访问计算密集型应用程序，或者在亭子(kiosk)中的公共计算机，其试图执行由专用网络上的远程计算机寄载的安全应用。

[0517] 响应于确定本地机器10缺少访问应用程序必需的至少一个特征，本地机器10做出不通过应用程序流获取该多个应用文件的决定。本地机器10执行驻留在本地机器10上的第二客户端代理，而不是执行被识别的第一客户端代理。在此实施例的一个实例中，在不能成功完成启动前分析的事件中，本地机器10接收第二客户端代理的标识以执行。本地机器10请求在远程机器30”上的应用程序的执行。第二客户端代理接收应用输出数据，该应用输出数据由远程机器30”上的应用程序的执行产生。第二客户端代理在本地机器10上显示应用输出数据。

[0518] 实例9

[0519] 在一个实施例中，网络管理员为本地机器10的用户提供对应用程序的访问。管理员在远程机器30’上执行应用以产生包括应用程序的多个应用文件。应用可包括图形用户界面。管理员可使用图形用户界面以识别应用程序和与应用程序相关联的安装程序，定义策略，以及指明关于被提供的访问的类型的特征，上述策略将被应用于授权对应用程序访问中，上述特征包括由本地机器10满足的试图访问或执行应用程序的必需条件。管理员可以识别安装整个应用程序或者应用程序的部分的安装程序，诸如更新或者补丁。

[0520] 在此实施例的一个实例中，远程机器30包括包机制530。包机制530在远程机器30上的隔离环境中执行安装程序。安装程序的执行导致向隔离环境532中安装至少一个应用文件，该应用文件与应用程序相关联。远程机器30可包括文件系统过滤器驱动器534，该文件系统过滤器驱动器534通过拦截安装程序的请求保证向隔离环境532中安装应用文件，从而在本地机器10上安装应用文件，并将该请求重定向到隔离环境532中。包机制530可使用文件系统过滤器驱动器534以保持安装到隔离环境532中的每个应用文件的记录。

[0521] 安装程序可向隔离环境532中安装多个应用文件。包机制530产生包括应用文件列举的文件，该应用文件在多个应用文件中。该文件可包括与多个应用文件相关联的信息，例如多个应用文件包括的应用程序的类型，应用程序的版本，与应用程序相关联的执行必需条件，以及策略需求，例如特定应用程序必需的执行方法。包机制530在远程机器30’上存储多个应用文件和文件。

[0522] 在一个实施例中，网络管理员识别包括现有的应用程序的更新版本的应用程序，或者在包括应用程序的多个应用文件中的应用文件。

[0523] C.用于加速客户端-服务器通信的系统和方法

[0524] 本发明的一个实施例指向用于加速客户端-服务器通信的系统和方法。这些系统和方法可被单独或一起使用，并且可与以上讨论的用于传送计算环境的任何系统和方法结合使用。更具体地，将讨论四个一般种类的加速技术。

[0525] 1.动态生成对象的高速缓存：在一些实施例中，通过设备1250在数据通信网络中执行高速缓存动态生成对象来加速客户端-服务器的通信。

[0526] 2.连接池:在一些实施例中,通过设备1250执行连接池技术加速客户端-服务器的通信。

[0527] 3.集成高速缓存:在另一个实施例中,通过设备1250执行与多个加速技术相集成的高速缓存来加速客户端-服务器的通信。

[0528] 4.客户端加速:在又一个实施例中,通过客户端10上执行的程序执行一个或多个加速技术来加速客户端-服务器的通信。

[0529] 1.高速缓存动态生成对象

[0530] 就像此处更详细描述的,在一个实施例中,设备1250可在带有一个或多个处理任务的操作系统的内核级别集成高速缓存功能,这些处理任务包括但不限于解码、解压缩、或验证和/或授权。这样的实例结构此处结合图27描述,但其它实例可用于实现此处描述的操作。

[0531] 图27示出了设备1250的实例结构3200。如上所述的,结构3200仅仅通过示出的方式被提供,并不试图被限制。如图2所示的,实例结构3200包括硬件层3206和分为用户空间3202和内核空间3204的软件层。

[0532] 硬件层3206提供硬件元件,在内核空间3204内的程序和服务在该硬件元件上被执行。硬件层3206也提供结构和元件,这使得在内核空间3204和用户空间3202内的程序和服务关于设备1250既在内又在外通信数据。如图27所示,硬件层3206包括用于执行软件程序和服务的处理单元3262,用于存储软件和数据的存储器3264,用于通过网络传输和接收数据的网络端口3266,以及用于执行涉及数据通过网络被传输和接收的加密套接字协议层处理功能的加密处理器3260。在一些实施例中,中央处理单元3262可在单独的处理器中执行加密处理器3260的功能。另外,硬件层3206可包括用于每个处理单元3262和加密处理器3260的多处理器。虽然示出的设备1250的硬件层3206通常带有加密处理器3260,但是处理器3260可为执行涉及任何加密协议的功能的处理器,例如加密套接字协议层(SSL)或者传输层安全(TLS)协议。在一些实施例中,处理器3260可为通用处理器(GPP),并且在进一步的实施例中,可为用于执行任何安全相关协议处理的可执行指令。

[0533] 虽然在图27中示出的设备1250的硬件层带有特定元件,但是设备1250的硬件部分或部件可包括任何类型和形式的元件、硬件或软件,任何类型或形式的计算设备,例如此处结合图1C和1D讨论和示出的计算设备135。在一些实施例中,设备1250可包括服务器、网关、路由器、开关、桥接器或其它类型的计算或网络设备,并且拥有与此相关的任何硬件和/或软件元件。

[0534] 设备1250的操作系统分配、管理或另外分离可用的系统存储器到内核空间3204和用户空间3202。在示例的软件结构3200中,操作系统可为任何类型和/或形式的Unix操作系统。同样的,设备1250可正运行任何操作系统,例如Microsoft®Windows操作系统的任何版本,Unix和Linux操作系统的不同版本,用于Macintosh计算机的Mac OS®的任何版本,任何被嵌入的操作系统,任何网络操作系统,任何实时操作系统,任何开放源操作系统,任何专有操作系统,用于移动计算设备或网络设备的任何操作系统,或者可以在设备1250上运行并执行此处所述操作的任何其它操作系统。

[0535] 预定内核空间3204用于运行内核3230,内核3230包括任何设备驱动器,内核扩展或其他内核相关软件。就像本领域技术人员所知的,内核3230是操作系统的核心,并提供对

资源以及设备1250的相关硬件元件的访问、控制和管理。根据实施例，内核空间3204也包括与高速缓存管理器3232协同工作的许多的网络服务或处理。有时也称为集成高速缓存，其益处此处将进一步详细描述。另外，内核3230的实施例将依赖于通过设备1250安装、配置或其他使用的操作系统的实施例。

[0536] 在一个实施例中，设备1250包括一个网络堆栈3267，例如基于TCP/IP的堆栈，用于与客户端10和/或服务器30通信。在一个实施例中，使用网络堆栈3267与例如网络40的第一网络以及第二网络40通信。在一些实施例中，设备1250终止第一传输层连接，例如客户端10的TCP连接，并建立到服务器30的第二传输层连接，客户端10使用该第二传输层连接，例如，在设备1250和服务器30终止第二传输层连接。可通过单独的网络堆栈3267建立第一和第二传输层连接。在其他实施例中，设备1250可包括多个网络堆栈，例如3267或3267'，并且在一个网络堆栈3267可建立或终止第一传输层连接，在第二网络堆栈3267'上可建立或者终止第二传输层连接。例如，一个网络堆栈可用于在第一网络上接收和传输网络包，并且另一个网络堆栈用于在第二网络上接收和传输网络包。在一个实施例中，网络堆栈3267包括用于排队一个或多个网络包的缓冲器3243，其中网络包由设备1250传输。

[0537] 如图27所示，内核空间3204包括高速缓存管理器3232、高速层2－7集成包引擎3240、加密引擎3234、策略引擎3236以及多协议压缩逻辑3238。在内核空间3204或内核模式而不是用户空间3202中运行这些部件或处理3232、3240、3234、3236和3238提高这些部件中的每个单独的和结合的性能。内核操作意味着这些部件或处理3232、3240、3234、3236和3238在设备1250的操作系统的核地址空间中运行。例如，在内核模式中运行加密引擎3234通过移动加密和解密操作到内核可改进加密性能，从而可减少在存储空间或在内核模式中的内核线程与存储空间或在用户模式的线程的传输的数量。例如，在内核模式获得的数据不需要传输或拷贝到运行在用户模式的处理或线程，例如从内核级数据结构到用户级数据结构。在另一个方面，也可减少内核模式和用户模式之间的上下文开关的数量。另外，在任何部件或处理3232、3240、3235、3236和3238间的同步和通信在内核空间3204中可被执行的更有效率。

[0538] 在一些实施例中，部件3232、3240、3234、3236和3238的任何部分可在内核空间3204中运行或操作，而这些部件或处理3232、3240、3235、3236和3238的其他部分可在用户空间3202中运行或操作。在一个实施例中，使用内核级数据结构来提供对一个或多个网络包的任何部分的访问，例如，网络包包括来自客户端10的请求或者来自服务器30的响应。在一些实施例中，由包引擎3240通过传输层驱动器接口或到网络堆栈3267的过滤器获得内核级数据结构。内核级数据结构可包括任何接口和/或通过与网络堆栈3257相关的内核空间3204可访问的数据，由网络堆栈3267接收或传输的网络话务量或包。在其他实施例中，任何部件或处理3232、3240、3234、3236和3238可使用内核级数据结构来执行部件或处理的需要的操作。在一个实例中，当使用内核级数据结构时，部件3232、3240、3234、3236和3238在内核模式3204中运行，而在另一个实施例中，当使用内核级数据结构时，部件3232、3240、3234、3236和3238在用户模式中运行。在一些实施例中，内核级数据结构可被拷贝或传输给第二内核级数据结构，或任何需要的用户级数据结构。

[0539] 高速缓存管理器3232可包括软件、硬件或软件和硬件的任意组合，以提供对任何类型和形式的内容的高速缓存访问、控制和管理，例如对象或由源服务器30提供服务的动

态产生的对象。由高速缓存管理器3232处理和存储的数据、对象或内容可包括任何格式的数据，例如标记语言，或通过任何协议通信。在一些实施例中，高速缓冲管理器3232复制存储在其它地方的原始数据或先前被计算、产生或传输的数据，其中相对于读高速缓存元件，原始数据可能需要更长的访问时间以便取得、计算或以别的方式获得。一旦数据被存储在高速缓存存储元件中，将来的使用可通过访问高速缓存的副本而非再取回或再计算的原始数据而获得，因而减少了访问时间。在一些实施例中，高速缓存存储元件奈特(nat)包括设备1250的存储器3264中的数据对象。在其它实施例中，高速缓存存储元件可包括有比存储器3264更快的存储时间的存储器。在另一个实施例中，高速缓存存储元件可以包括设备1250的任何类型和形式的存储元件，诸如硬盘的部分。在一些实施例中，处理单元3262可提供被高速缓存管理器3232使用的高速缓存存储器。在又一个实施例中，高速缓存管理器3232可使用存储器、存储区或处理单元的任何部分和组合来高速缓存数据、对象或其它内容。

[0540] 另外，高速缓存管理器3233包括用于执行此处描述的技术的任何实施例的任何逻辑、功能、规则或操作。例如，基于无效时间周期的终止，或者从客户端10或服务器30接收无效命令，高速缓存管理器3232包括无效对象的逻辑或功能。在一些实施例中，高速缓存管理器3232可作为在内核空间3204中执行的程序、服务、处理或任务来操作，并且在其它实施例中在用户空间3202中操作。在一个实施例中，当高速缓存管理器3232的第一部分在用户空间3202内执行时，第二部分在内核空间3204内执行。在一些实施例中，高速缓存管理器3232可包括任何形式的通用处理器(GPP)，或者任何其它形式的集成电路，例如现场可编程门阵列(FPGA)，可编程逻辑电路(PLD)，或者专用集成电路(ASIC)。

[0541] 策略引擎3236可包括，例如，智能统计引擎或其它可编程应用。在一个实施例中，策略引擎3236提供配置功能以允许用户识别、指定、定义或配置高速缓存策略。策略引擎3236，在一些实施例中，也访问存储器以支持数据结构，例如备份表或hash表，以启用用户选择的高速缓存策略决定。在其它实施例中，除了对安全、网络话务量、网络访问、压缩或其它任何由设备1250执行的功能或操作的访问、控制和管理之外，策略引擎3236可包括任何逻辑、规则、功能或操作以决定和提供对设备1250所高速缓存的对象、数据、或内容的访问、控制和管理。在一些实施例中，策略引擎3236可与策略引擎406的功能相集成。在一个实施例中，策略引擎3236可基于收集代理404提供的信息，确定高速缓存的策略决定。在一些实施例中，策略引擎3236可基于应用执行的类型，确定高速缓存策略决定。在一个实施例中，策略引擎可基于应用是否被流式传输到客户端10，确定高速缓存的策略决定。特定高速缓存策略的其它实施例此处进一步描述。

[0542] 加密引擎3234包括用于控制任何安全相关协议处理，例如SSL或TLS，或其相关的任何功能的任何逻辑、商业规则、功能或操作。例如，加密引擎3234加密并解密通过设备1250通信的网络包，或其任何部分。加密引擎3234也可代表客户端10、服务器30或设备1250安装或建立SSL或TLS连接。同样的，加密引擎3234提供SSL处理的卸载和加速。在一个实施例中，加密引擎3234使用隧道协议来提供在客户端10和服务器30间的虚拟专用网络。在一些实施例中，加密引擎3234与加密处理器3260通信。在其它实施例中，加密引擎3234包括运行在加密处理器3260上的可执行指令。

[0543] 多协议压缩引擎3238包括用于压缩一个或多个网络包协议(例如被设备1250的网

络堆栈3267使用的任何协议)的任何逻辑、商业规则、功能或操作。在一个实施例中,多协议压缩引擎3238双向压缩在客户端10和服务器30间任何基于TCP/IP的协议,包括消息应用编程接口(MAPI)(电子邮件,email)、文件传输协议(FTP)、超文本传输协议(HTTP)、通用Internet文件系统(CIFS)协议(文件传输)、独立计算结构(ICA)协议、远程桌面协议(RDP)、无线应用协议(WAP)、移动IP协议以及IP上语音(VoIP)协议。在其它实施例中,多协议压缩引擎3238提供基于超文本标记语言(HTML)的协议的压缩,并且在一些实施例中,提供任何标记语言的压缩,例如可扩展标记语言(XML)。在一个实施例中,多协议压缩引擎3238提供任何高性能协议的压缩,例如为设备1250设计的用于设备1250通信的任何协议。在另一个实施例中,多协议压缩引擎3238使用修改的传输控制协议来压缩任何载荷或任何通信,例如事务TCP(T/TCP)、带有选择确认的TCP(TCP-SACK)、带有大窗口的TCP(TCP-LW)、例如TCP-Vegas协议的拥塞预报协议以及TCP欺骗协议。

[0544] 同样的,多协议压缩引擎3238通过桌面客户端,例如Micosoft Outlook和非web瘦客户端,例如由通用企业应用像Oracle,SAP和Siebel启动的任何客户端,甚至移动客户端(例如便携式计算机),来加速用户访问应用的性能。在一些实施例中,多协议压缩引擎3238通过在内核模式3204内部执行并与访问网络堆栈3267的包处理引擎3240集成,可以压缩TCP/IP协议携带的任何协议,例如任何应用层协议。

[0545] 高速层2-7集成包引擎3240,通常也称为包处理引擎,或包引擎,通过网络端口3266负责内核级包的处理的管理,该包由设备1250接收和传输。高速层2-7集成包引擎3240可包括在处理期间用于排队一个或多个网络包的缓冲器,例如用于网络包的接收或者网络包的传输。另外,高速层2-7集成包引擎3240通过网络端口3266与一个或多个网络堆栈3267通信以发送和接收网络包。高速层2-7集成包引擎3240与加密引擎3234、高速缓存管理器3232、策略引擎3236和多协议压缩逻辑3238协同工作。更具体地,配置加密引擎3234以执行包的SSL处理,配置策略引擎3236以执行涉及话务量管理的功能,例如请求级内容切换以及请求级高速缓冲重定向,并配置多协议压缩逻辑3238以执行涉及数据压缩和解压缩的功能。

[0546] 高速层2-7集成包引擎240包括包处理定时器3243。在一个实施例中,包处理定时器3242提供一个或多个时间间隔以触发输入处理,例如,接收或者输出(即传输)网络包。在一些实施例中,高速层2-7集成包引擎3240响应于定时器3242处理网络包。包处理定时器3242向包引擎3240提供任何类型和形式的信号以通知、触发或传输时间相关事件、间隔或发生。在许多实施例中,包处理定时器3242以毫秒级操作,例如100ms、50ms、或25ms。例如,在一些实例中,包处理定时器3242提供时间间隔或者其它使得由高速层2-7集成包引擎3240以10ms时间间隔所处理的网络包,而在其它实施例中,以5ms时间间隔,并且在进一步的实施例中,短到3、2或1ms时间间隔。高速层2-7集成包引擎3240在操作期间可与加密引擎3234、高速缓存管理器3232、策略引擎3236以及多协议压缩引擎3238连接、集成或通信。同样的,响应于包处理定时器3242和/或包引擎3240,可执行加密引擎3234、高速缓存管理器3232、策略引擎3236以及多协议压缩引擎3238的任何逻辑、功能或操作。因此,在由包处理定时器3242提供的时间间隔粒度,可执行加密引擎3234、高速缓存管理器3232、策略引擎3236以及多协议压缩引擎3238的任何逻辑、功能或操作,例如,时间间隔少于或等于10ms。例如,在一个实施例中,响应于高速层2-7集成包引擎240和/或包处理定时器3242,高速缓

存管理器3232可执行任何所高速缓存的对象的无效。在另一个实施例中，高速缓存的对象的终止或无效时间被设定为与包处理定时器3242的时间间隔相同的粒度级，例如每10ms。

[0547] 与内核空间3204不同，用户空间3202是操作系统的存储区域或部分，被用户模式应用或用户模式运行的程序所使用。用户模式应用不能直接访问内核空间3204和为了存储内核服务而使用用户服务调用。如图27所示，设备1250的用户空间3202包括图形用户界面(GUI)3210、命令行接口(CLI)3212、壳服务(shell service)3214、健康监控程序3216以及守护(daemon)服务3218。GUI 210和CLI 3212提供方法，通过该方法系统管理员或其他用户可与设备1250的操作作用并控制该设备1250的操作，例如通过设备1250的操作系统，或者是用户空间3202或者内核空间3204。GUI 3210可为任何类型和形式的图形用户界面，并且可通过文本、图形由任意类型的程序或应用呈现，例如浏览器。CLI 3212可为任何类型和形式的命令行或基于文本的接口，例如通过操作系统提供的命令行。例如，CLI 3212可包括壳，该壳是启用用户与操作系统相互作用的工具。在一些实施例中，可通过bash、csh、tcsh或者ksh类型的壳提供CLI 3212。，壳服务3214包括程序，服务，任务，处理或可执行指令以支持由用户通过GUI 3210和/或CLI 3212与设备1250或操作系统交互。

[0548] 使用健康监控程序3216监控、检查、报告并确保网络系统运行正常，用户正通过网络接收请求的内容。健康监控程序3216包括一个或多个程序、服务、任务、处理或可执行指令，为监控设备1250的任何行为提供逻辑、规则、功能或操作。在一些实施例中，健康监控程序3216拦截并检查通过设备1250传递的任何网络话务量。在其他实施例中，健康监控程序3216通过任何合适的方法和/或机制与一个或多个下述设备连接：加密引擎3234，高速缓存管理器3232，策略引擎3236，多协议压缩逻辑3238，包引擎3240，守护服务3218以及壳服务3214。同样的，健康监控程序3216可调用任何应用编程接口(API)以确定设备1250的任何部分的状态、情况或健康。例如，健康监控程序3216可基于周期性的查验或发送状态查询以检查程序、处理、服务是否被激活并当前正在运行。在另一个实施例中，健康监控程序3216可检查由任何程序、处理、服务或任务提供的任何状态、错误或历史日志以确定设备1250任何部分的任何条件、状态或错误。

[0549] 守护服务3218是连续的或在背景中运行的程序，并且操纵该程序控制设备1250接收的周期性服务请求。在一些实施例中，守护服务向其他程序或处理(例如合适的另一个守护服务3218)转发请求。对于本领域技术人员所公知的，守护服务3218可无人监护的运行，以执行连续的或周期性的系统范围程序，例如网络控制，或者执行任何需要的任务。在一些实施例中，一个或多个服务3218运行在用户空间3202中，而在其他实施例中，一个或多个守护服务3218运行在内核空间。

[0550] 动态内容，例如一个或多个动态产生的对象，可通过服务器产生，称为应用或源服务器30和/或后端数据库，如图1A所描述的，该数据库处理来自一个或多个本地或远程的客户端10的对象请求。当这些应用或数据库处理数据时，包括涉及到输入从客户端接收的数据、被这些数据库或应用服务的响应对象可改变。由这些应用或数据库在源服务器中产生的在前的对象不再新鲜并且因此不再被高速缓存存储。例如，假定相同输入集，第一实例的动态产生对象与第二实例的动态产生对象不同。在另一个实例中，使用不同的输入集动态产生相同的对象，使得对象的第一实例被产生不同于对象的第二实例。

[0551] 为了获得提高的网络性能，设计并配置设备1250以通过以下详细描述的各种方法

解决问题，该问题出现在高速缓存动态产生内容中。在此处描述的一些实施例中，设备1250包括一个或多个技术，这些技术用于使得存储在高速缓存中的动态产生内容的无效更加有效率和有效果。另外，设备可包括用于执行控制和用于突发访问的高速缓存的技术。高速缓存存储器典型的存储每个对于对象的请求的响应，只要这种请求没有标记为不可高速缓存的。如此处描述的，动态产生内容的有效率的高速缓存需要可使得在高速缓存中的对象及时无效的技术，该对象在源服务器上已经经历了改变。及时无效允许高速缓存避免服务失时效内容——带有动态产生内容的特定相关的任务，特别是内容的改变不规则发生的地方。以下提出的是一些保证动态产生内容的及时无效的技术。

[0552] a.集成功能

[0553] 在一个方面，动态产生对象的高速缓存涉及到高速缓存管理器3232，策略引擎3236，加密引擎3234，和/或多协议压缩引擎3238的集成功能、逻辑或操作的技术，上述多协议压缩引擎3238带有响应于包处理定时器3242的高速层2－7集成包引擎3240的包处理操作。例如，可在用于包处理操作的包处理定时器3242的时间间隔内执行高速缓存管理器3232的操作，例如在接收或传输网络包之上。就像下述将详细描述的，在一个实施例中，通过与包处理操作集成和/或使用包处理定时器，高速缓存管理器3232可高速缓存带有终止时间降至非常小的时间间隔的对象。在其他实施例中，响应于包处理定时器3242的高速缓存管理器3232也可接收无效命令从而在高速缓冲对象的非常短的时间周期内无效对象。

[0554] 图28A描述的方法3300示出了技术的一个实施例，该技术用于请求高速缓存管理器3232、策略引擎3236、加密引擎3234和/或多协议压缩引擎3238以执行在处理期间或与时间间隔相关联的操作，用于通过高速层2－7集成包引擎或包处理引擎3240处理网络包。总的来说，在方法3300的步骤3310，设备1250接收网络包或被请求传输网络包。在步骤3315，设备31250请求包处理引擎3240响应于包处理定时器3242处理网络包。作为包处理操作的一部分，或与之关联，在步骤3320，包处理引擎240请求高速缓存管理器3232、策略引擎3236、加密引擎234，和/或多协议压缩引擎3238在高速缓存的对象上执行操作。在步骤3325，高速缓存管理器3232、策略引擎3236、加密引擎3234和/或多协议压缩引擎3238执行被请求的操作，该操作可包括此处描述技术的任何一个或者组合。在一个实施例中，高速缓存管理器3232确定高速缓存的对象的无效，并标记高速缓存的对象无效。在一些实施例中，高速缓存管理器3232响应于包处理引擎3240的请求来清除无效对象。当高速缓存管理器3232响应于包处理定时器3242正执行这些操作时，对象的无效可发生在毫秒级的时间段内，并带有拥有时间间隔量级中终止的对象，该时间间隔终止由包处理定时器3242提供，例如10ms。

[0555] 在方法3300进一步的细节中，在步骤3310，设备1250接收一个或多个网络包，和/或传输一个或多个网络包。在一些实施例中，设备1250请求通过网络40或网络40’传输一个或多个网络包。在另一个实施例中，设备1250在一个端口3266接收网络包并在相同的端口3266或不同的端口3266’传输网络包。在一些实施例中，设备1250的包引擎3240传输或请求传输一个或多个网络包。在一个实施例中，设备1250在第一网络40接收或传输包，而在另一个实施例中，设备1250在第二个网络40’上接收或传输包。在其它实施例中，设备1250在相同的网络40中接收和传输包。在一些实施例中，设备1250向一个或多个客户端10接收和/或传输网络包。在其它实施例中，设备1250向一个或多个服务器30接收和/或传输网络包。

[0556] 在步骤3315，一旦在设备1250的网络端口3266接收到网络包，或者一旦请求从设备1250传输网络包，或者一旦一个或多个网络包的接收和/或传输的任何组合，设备1250可请求或触发包处理引擎3240的包处理操作。在一些实施例中，包处理引擎3240的包处理操作由包处理定时器3242提供的信号触发。在一个实施例中，包处理定时器3242可提供涉及到一个或多个网路包的接收和/或传输的中断驱动或者事件驱动定时器功能。在一些实施例中，通过由设备1250接收和/或传输的网络包的速率，或者通过处理每个包或一批包的速率，驱动包处理定时器3242。同样的，在一个或多个包处理操作的每个设置之后，包处理定时器3242可被触发和复位。在另一个实施例中，包处理定时器3242提供时间间隔，相同或者不同的时间间隔，以触发、唤醒或信号通知包处理引擎3240以执行功能或操作，例如控制被接收的包或者传输被提交的包。如上结合图27中的设备1250所讨论的，包处理定时器3242可以毫秒级操作，例如以时间间隔10ms或更少引发包处理操作的时间间隔或触发。包处理定时器的时间粒度功能可以不同的方式提供，并用于包处理引擎3240的包处理操作的操作中。

[0557] 在方法3300的步骤3320，包处理引擎3240请求一个或多个高速缓存管理器3232、策略引擎3236、加密引擎3234和/或多协议压缩引擎3238以执行操作。在一个实施例中，包处理引擎3240或包处理定时器3242产生一个或者多个信号向一个或者多个高速缓存管理器3232、策略引擎3236、加密引擎3234和/或多协议压缩引擎3238。在网络包或一个或多个网络包的包处理操作之前、之中或之后的任何时间点，包处理引擎3240可执行请求或发信号通知操作。在一个实施例中，一旦包处理定时器3242触发或包处理定时器3242提供时间间隔的终止，并且在网络包上执行包处理操作之前，包处理引擎240做出请求。在另一个实施例中，在执行一个或多个网络包操作的过程中，包处理引擎3240做出请求。例如，在操作执行过程中，例如在功能调用中，包处理引擎240可向高速缓存管理器3232、策略引擎3236、加密引擎3234和/或多协议压缩引擎238做出应用编程接口(API)调用。在其它实施例中，一旦完成网络包处理操作，包处理引擎3240做出请求。

[0558] 在步骤3325，通过一个或多个高速缓存管理器3232、策略引擎3236、加密引擎3234和/或多协议压缩引擎3238执行请求操作。在一些实施例中，通过内核3204提供的任何功能或操作可被请求从而被执行，例如通过内核应用编程接口(API)。同样，可结合通过包处理定时器3232的包处理的计时或计时间隔，执行设备1250的任何功能。在一些实施例中，被请求的操作可同步并结合包处理引擎3240的包处理操作执行。例如，一旦完成被请求的操作，或者从被请求的操作获得响应，包处理操作等待并持续。在其它实施例中，被请求的操作与包处理操作异步执行。例如，包处理引擎3240发送请求以执行操作但不阻塞或等待以接收来自该操作的响应。如将进一步结合图28B的方法3350进一步详细描述的，包处理引擎3240可请求高速缓存3232以执行任何高速缓存管理功能，例如检查对象的终止或无效，标记对象为无效，或清除无效或终止的对象。

[0559] 在一些实施例中，在步骤3320包处理引擎3240发送多个请求，例如第一请求到高速缓存管理器232，并且第二请求到加密引擎3234。在其它实施例中，包处理引擎3240，在步骤3320，发送单个请求，该单个请求包括将被设备1250分发的多个请求，例如通过内核3230到设备1250的指定部分。在一个实施例中，请求持续在彼此间传送。在另一个实施例中，请求可依赖于在前请求的状态、结果、成功或完成。例如，到策略引擎3236的第一请求可被用

于确定策略，该策略用于处理来自另一个设备或与网络包相关联的用户的网络包。基于策略引擎3236的策略，依赖于第一请求的结果，可获得或不获得到高速缓存的第二请求。高速缓存管理器3232、策略引擎3236、加密引擎3234和/或多协议压缩引擎3238与包处理引擎3240一起集成在设备1250的内核空间204中，就像此处所描述的，存在设备1250的各种操作，这些操作通过包处理操作触发和与包处理操作集成。

[0560] b.无效粒度

[0561] 在另一个方面，动态产生对象的高速缓存涉及和包括配置被高速缓存存储的对象的终止时间到合适的粒度时间间隔的能力，例如由包处理定时器提供的时间间隔的粒度。该特征称为“无效粒度”。同样，在一个实施例中，无效时间降至非常小的时间间隔的对象可被高速缓存。在其他实施例中，响应于包处理定时器的高速缓存管理器也可接收无效命令从而在高速缓存对象的非常短的时间周期内无效对象。通过在终止时间内提供该微小的粒度，该高速缓存可高速缓存并保存频繁改变的对象，有时在一秒内甚至几次。在一个实施例中设备使用的一项技术可影响包处理定时器，该技术可以毫秒级的时间增量操作，允许无效或终止粒度降至10ms或更少。传统的高速缓存，与之对比，通常不设置少于一秒的终止或无效粒度。

[0562] 现在看图28B，描述方法3350的实施例，用于响应于包处理定时器3242和/或包处理引擎3240而无效或终止高速缓存的对象。同样，在一些实施例中，高速缓存的对象可以毫秒级被无效或终止，例如10ms或更少。总的来说，在方法3350的步骤3355，响应于包处理定时器3242，高速缓存管理器3232接收信号或者请求从而通过包处理引擎3240执行操作。在步骤3360，高速缓存管理器3232确定高速缓存的对象(例如动态产生对象)是否无效或者被终止。在步骤3365，如果对象是无效的，高速缓存管理器3232标识该对象为无效，并且在步骤3370，从高速缓存管理器3232清除该无效的对象。

[0563] 在步骤3355进一步的细节中，在一些实施例中，在网络包处理期间的任何时间点，高速缓存管理器3232可被发送信号或者被请求以执行高速缓存涉及的操作。在一个实施例中，在步骤3355，在由设备1250接收或传输的网络包处理之前，高速缓存管理器3232接收操作请求。在另一个实施例中，一旦完成网络包的处理，高速缓存管理器3232接收操作请求。例如，包处理引擎3240完成网络包处理，并且在等待定时器3242的另一个时间间隔之前，或者在处理下一个包之前，请求高速缓存以执行操作。在其他实施例中，在包处理操作期间，包处理引擎3240向高速缓存管理器3232传送操作请求。在另一个实施例中，高速缓存管理器3232例如从包处理引擎3240或者从包处理定时器3242接收信号，以触发高速缓存管理器3232来执行操作。在一些实施例中，信号指示高速缓存对象无效或者指示终止高速缓存的对象的终止。

[0564] 在一些实施例中，高速缓存管理器3232可从高速缓存管理器3232的外部实体接收请求，从而执行高速缓存操作，例如请求对象无效，该对象被服务器30传送，并被包处理引擎3240处理。在一个实施例中，高速缓存管理器3232可在10ms或更少时间之内接收高速缓存对象的无效请求，而在另一个实施例中，短到5ms、2ms或1ms。在其他实施例中，高速缓存管理器3232可响应于高速缓存管理器3232的操作或功能以执行高速缓存操作，例如在任何高速缓存命令的处理期间，定时器的终止使得对象被无效。在其他实施例中，高速缓存管理器3232使用设备1250的包处理定时器3242触发高速缓存操作。例如，在可被定时器3242设

置的任何时间间隔，定时器2342可触发或发信号给高速缓存以检查高速缓存的对象的无效或终止。在一个实例中，在设置为10ms或更少的时间内，定时器3242可被设置触发或发信号给高速缓存，或者在另一个实施例中，设置短至5ms、2ms或者1ms。在一些实施例中，源服务器30可设置对象的终止时间。在其他实施例中，设备1250或客户端10可设置对象的终止时间。

[0565] 在步骤3360，高速缓存管理器3232确定存储在高速缓存中的对象的无效或终止。在一些实施例中，在高速缓存中的对象基于定时器的终止被无效。在一个实施例中，高速缓存管理器3232基于定时器的终止可发出关于对象无效的命令。在另一个实施例中，响应于定时器的终止，被存储在高速缓存中的对象通过高速缓存管理器3232被自动无效，例如具有包处理定时器3242的定时器设定。在一些实施例中，响应于包处理定时器3242，高速缓存管理器3232检查用于高速缓存的对象的任何定时器的终止。在一个实施例中，高速缓存管理器3232决定对象定时器已经终止，而在另一个实施例中，高速缓存管理器3232对象定时器还没有终止。在进一步的实施例中，响应于包处理定时器3242的第二触发或第二时间间隔，如果先前被检查的对象定时器已经终止，高速缓存管理器3232将检查第二定时器。

[0566] 在一些实例中，高速缓存管理器3232解析、拦截、访问、读取或其他方式处理无效命令或者请求以识别在高速缓存中的对象的无效。在一个实施例中，高速缓存管理器3232外的实体向高速缓存管理器3232发送无效命令以无效对象。在另一个实施例中，响应于包处理定时器3242，外部实体可发送无效命令。如果对象是有效的和/或者还没有被无效掉，高速缓存管理器3232响应于请求来无效该对象。在一些实施例中，由高速缓存管理器3232处理的无效请求响应于处理请求的包处理引擎3240的包处理操作，其接着可能也响应于包处理定时器3242。

[0567] 在步骤3365，高速缓存管理器3232标记对象为无效。高速缓存管理器3232可以以任何合适或者期望的方式标记每一对象为无效。在一个实施例中，通过设置存储对象的标志、属性或者性质，将对象标记为无效。例如，标志可以被设为向高速缓存管理器3232标识对象无效的任何值。在另一个实施例中，通过移动对象到用于存储无效对象的高速缓存的区域或者部分，可以将对象标记为无效。在其他实施例中，高速缓存管理器3232可以通过数据库或者链接列表或者任何类型和形式的数据结构来识别或者跟踪存储对象的无效和/或有效状态。在一些实施例中，高速缓存管理器3232使用一个或者多个对象来识别或者跟踪存储在高速缓存中的一个或者多个对象的有效性或者无效性。在另一个实施例中，通过改变、修改或者变更存储的对象将对象标记为无效，例如删除或者移动对象的部分使得其不可以被使用，或者通过改变或者修正对象的名称。

[0568] 在步骤3370，高速缓存管理器3232，在一些实施例中，从高速缓存清除这些被标记为无效的对象。在另一个实施例中，一旦请求对象，例如通过客户端10，高速缓存管理器3232从高速缓存清除无效对象。在一些实施例中，高速缓存管理器3232使用在对象的无效或终止之后接收的对象的更新的副本或版本改写无效对象。在另一个实施例中，高速缓存管理器3232通过存储另一个到高速缓存的相同部分来重新使用被无效对象占用的高速缓冲存储器。在又一个实施例中，高速缓存管理器3232不清除被标记为无效的对象，而是保持存储在存储器或者高速缓存存储部件中的对象。

[0569] 尽管响应于包处理定时器和/或结合包处理操作来提供无效粒度，方法3350描述

了无效和高速缓存的对象的清除，但是此处描述的高速缓存的任何操作和高速缓存管理的任何技术，以及设备1250的任何其他操作可在合适的粒度时间间隔内被执行，该粒度时间间隔由包处理定时器提供。在一些实施例中，高速缓存的对象的无效或终止可发生在短至100ms的时间间隔，而在另一个实施例中，短至50ms时间间隔。在一些实施例中，高速缓存的对象的无效或终止可发生在短至25ms的时间间隔，并且在其它实施例中，短至10ms时间间隔。然而在其他实施例中，高速缓存的对象的无效或终止可发生在短至5ms的时间间隔，并且仍然在进一步在实施例中，短至3、2或1ms时间间隔。

[0570] 如在结合上述图28A和28B的方法3300和3350中所描述的，在耗用非常小的时间增量之后，通过包括无效对象的功能，启用动态产生内容的改进的高速缓存。一些动态内容事实上是可修订的，以用于在非常短的时间周期从高速缓存被存储和被服务。但是，为了成功的高速缓存这样的内容，在对象被无效并从高速缓冲存储器中清除之前，根据一个实施例的方法提供在非常短的时间周期内对象的高速缓存。例如，某些动态产生对象可被高速缓存的时间长为1秒，但是对于经常改变的内容更长的任何时间是经常不可接受的。在一个实施例中，在1秒的几分之一的时间之后，方法包括无效或终止高速缓存的内容。作为例子，如果应用花费100毫秒产生动态响应，则高速缓存可存储并服务该响应持续时间少于或等于100毫秒周期，不伤害数据的新鲜度。由于100毫秒少于产生新对象的时间，在100毫秒周期内将不会产生新的对象。在此期间，设备1250因此可被建立以服务于先前的对象。降至非常小的时间增量的设备1250的这种无效的能力经常对于这样的应用环境非常有用，在该环境中设置数据库处理隔离级别从而允许重复的读或串行的读。

[0571] C.无效命令

[0572] 基于用于内容的预先定义的终止时间，传统的高速缓存技术无效被存储的内容，该技术通常由管理员配置或从服务对象的服务器接收。下面描述的是另一个用于无效对象的技术，该技术为了更有效率的高速缓存动态产生的内容。技术包括在设备1250接收无效命令的能力，该命令实时识别一个或多个先前存储在高速缓存中的对象为无效。例如，无效命令可通过传输到客户端的网络包传送到客户端或通过服务器调用的应用编程接口(API)到设备。这与传统的方法不同，通过传统的方法，在对象被服务时服务器简单的设置包括在对象头部中的高速缓存终止时间。

[0573] 图29A和29B中更具体地示出了技术。图29A示出了用于维持高速缓存的方法的流程图，例如计算机存储器高速缓存。总的来说并根据步骤3410，先前被源服务器30服务的动态产生对象被存储在高速缓存中。例如，动态产生对象可不被识别为可高速缓存的或者包括任何高速缓存或高速缓存控制信息。在步骤3420，在高速缓存或高速缓存管理器3232接收无效命令。无效命令识别一个或多个先前被服务的对象为无效。在步骤3430，响应无效命令，高速缓存或高速缓存管理器3232标记被识别的对象为无效。

[0574] 在步骤3410进一步的细节中，高速缓存管理器3232在高速缓冲存储器元件中存储从任何源接收、获得或传递的动态产生对象。在一些实施例中，动态产生对象可从服务器30产生和服务。在其它实施例中，动态产生对象可被客户端10产生和传递。在一些实施例中，设备1250的另一个部分、部件或处理产生对象并在高速缓存中存储该对象。在进一步的实施例中，动态产生对象可由另一个设备1250或网络上的另一个计算设备产生，并传输或传送到设备1250。在一些实施例中，动态产生对象没有识别为可高速缓存的或者识别为不可

高速缓存的。在其它实施例中，动态产生对象被识别为可高速缓存的或者在高速缓存的控制下。

[0575] 在步骤3420，高速缓存管理器3232接收识别对象为无效的无效命令，这样的动态产生对象被存储在高速缓存中。在一个实施例中，无效命令可包括任何类型的命令或指令，指示高速缓存该对象无效或者可能以其他方式失时效。在一些实施例中，无效命令识别对象并且也识别对象无效的时间，以及对象的哪一部分无效。在一个实施例中，高速缓存管理器3232提供应用编程接口(API)，该应用编程接口可被源服务器30远程调用。在一些实施例中，高速缓存管理器3232可提供任何类型和形式的协议，用于通过一个或多个网络包接收命令并答复命令。在一个实施例中，高速缓存管理器3232或设备1250提供可扩展标记语言(XML)API接口用于接收和处理无效命令。例如，高速缓存管理器3232可提供web服务接口。在一些实施例中，高速缓存管理器3232通过向源服务器30发送确认、状态或其它响应以答复无效命令。在其它实施例中，高速缓存管理器3232不答复无效命令。在一个实施例中，如果运行在源服务器30的应用执行使被存储的对象失时效的动作，例如通过产生新的或更新版本的对象，则该对象被标记为无效。这可发生在，例如，当新闻编辑对快速发展中的新闻故事做出改变并因此想确定故事的最新版本已经被提供给客户端时。

[0576] 无效命令可从源服务器通过产生该对象的应用，通过另一个服务器30或者另一个设备1250来发送。在一个实施例中，源服务器30响应于对源服务器30上的动态产生对象的改变，自动向高速缓存3232发送或传送无效命令。无效命令也可通过服务器30或者设备1250外部的管理控制产生。例如，管理控制可为运行在网络上并于设备1250通信的任何类型和形式的程序或应用，例如管理员控制台。另外，客户端10可向设备1250或高速缓存管理器3232发出或传达无效命令。例如如果客户端将采取客户端10承认的动作，该动作将引起在源服务器的被请求对象的改变，客户端可传达无效命令。任何被存储在高速缓存中的对象可通过将用户命令传输到高速缓存而被无效，该命令在高速缓存被本地执行或使用XML API内部结构被远程调用。

[0577] 根据步骤3430，存储在高速缓存中的已经被识别为无效的对象，例如先前被服务的动态产生对象，则响应于无效命令被标记为无效。无效的对象不会从高速缓存中提供给请求客户端，而是改为从源服务器直接服务。高速缓存管理器3232可以任何合适或期望的方式标记每个对象为无效。在一个实施例中，通过设置被存储对象的标志、属性或性质来标记对象的无效。例如，标志可设置为给高速缓存管理器3232识别对象是无效的任何值。在另一个实施例中，通过将对象移动到存储无效对象的高速缓存的区域或部分来标记对象无效。在其它实施例中，高速缓存管理器3232可通过数据库或者链接表或者任何类型和形式的数据结构来识别或跟踪被存储对象的无效和/或有效状态。在一些实施例中，高速缓存管理器3232使用一个或多个对象来识别或跟踪存储在高速缓存中的一个或多个对象的有效或无效。在另一个实施例中，通过改变、修改或变更被存储的对象标记对象为无效，例如删除或移除对象的一部分以便使其不再使用，或通过改变或修正对象的名称。

[0578] 在一些实施例中，设备1250连续的从高速缓存清除被标记为无效的对象。在另一个实施例中，一旦请求该对象，例如由客户端10，设备1250从高速缓存中清除该无效对象。在一些实施例中，设备1250使用对象的更新的副本或版本重写无效对象。在另一个实施例中，通过向高速缓存的相同部分存储另一个动态产生对象，设备1250重新使用被无效对象

所占用的高速缓冲存储器。

[0579] 利用高速缓存管理器3232的命令无效API，任何与设备1250通信的计算设备或用户可请求来无效一个对象，例如动态产生的存储在高速缓存中的对象。同样，存储在高速缓存中的对象的无效可被实时控制，而非使用预先确定的配置终止或无效时间周期。因此，使用这些技术被高速缓存的对象的寿命可从例如数据库的外部应用处理节点或发端应用服务器控制。例如，可配置设备1250与数据库一起工作，以便数据库的改变自动触发从数据库(或应用)到设备1250的无效命令以清除特定的对象。

[0580] d.使用无效命令的组的无效

[0581] 在进一步的实施例中，在相同的时间，设备1250识别并无效存储在高速缓冲中的对象组。存储在传统高速缓冲存储器中的每个对象被高速缓存单独、分别处理，确定对象是否失时效。当每个对象达到它的指定终止时间(通常由服务器设置并被高速缓冲存储在表中)时，该项目被从高速缓冲存储器中清除。此传统的方法效率低，并且最终不足以成功的控制在试图高速缓存动态产生内容时出现的挑战。

[0582] 图29B示出了用于维持高速缓存的方法的另一个实施例，例如计算机存储器高速缓存，其中设备1250有能力建立、存储和无效之前已经被源服务器30服务的相关对象组。总的来说，在步骤3410，对象，例如，从源服务器30服务的动态产生对象被存储在高速缓存中。在步骤3412，高速缓存管理器3232形成先前被服务的存储在高速缓存中的对象的组。在一个实施例中，如下述进一步详细描述的，该组可与一个或多个对象确定因子相关联或被一个或多个对象确定因子确定。在步骤3414，高速缓存管理器3232维护对象组的记录。在步骤3422，高速缓存管理器3232接收无效命令以无效对象组。在步骤3432，高速缓存管理器3232响应于无效命令，标记对象组为无效。

[0583] 步骤3410与在图29A中一样，其中对象存储在设备1250的高速缓存中，例如先前从源服务器30服务的动态产生对象。在一些实施例中，一个或多个对象不能识别为当可高速缓存的或其他的，或者可以以其他方式不具有任何高速缓存或者高速缓存控制信息。例如，服务器30可以假设动态产生对象不被高速缓存。

[0584] 根据步骤3412，设备1250形成组，该组在先前由源服务器30服务并存储在高速缓存中的对象集之外。任何对象的合适或期望的集可与彼此关联以形成组。例如，产生用于服务web页面或者与服务web页面关联的任何动态产生对象可以形成组。在一些实施例中，对象可与多个组相关联。在其他实施例中，对象的一个组可形成对象的另一个组的子集。在一些实施例中，对象的形成的组拥有从相同的服务器30服务的对象，而在其他实施例中，对象的形成的组拥有从不同服务器30服务的对象。在进一步的实施例中，对象的形成的组可包括来自客户端10的对象，来自服务器30的对象，或者由客户端10和服务器30产生或服务的对象。在一个实施例中，在组中的一个对象是静态的，而在组中的另一个对象是动态产生的。在一些实例中，组中的一个对象不被识别为可高速缓存的，而组中的另一个对象被识别为可高速缓存的。在其他实例中，组中的对象根据服务器30提供的功能或应用可以逻辑相关。在另一个实例中，组中的对象当与相同的客户端10或相同的用户相关联时相关。

[0585] 在步骤3414，维护对象组的记录。在实现此处描述的一些实施例的操作中，可使用用于记录和维护对象组的记录或者其他相关对象的各种技术。在一个实施例中，可在例如查询表中直接维持记录。在另一个实施例中，记录可以hash表格式出现。在一些实施例中，

高速缓存管理器3232维持在数据库、数据结构中对象、或者在存储器中维持对象的相关性。在进一步的实施例中，组中每个对象的标志、性质或属性被分配或设定为识别该组的值，例如值等于，识别或引用组的名称或标识符，例如将在下面更详细的描述的组的对象确定因子。在一些实施例中，对象组在被识别为持有该组的高速缓存存储器的一部分中被布置、放置或定位。

[0586] 在步骤3422，在设备1250或高速缓存管理器3232接收无效命令。根据图29B描述的实施例，无效命令识别一个或多个对象是无效的，或者是失效的。在一些实施例中，无效命令引用、识别或指定对象组的名称或标识符。在一个实施例中，无效命令包括无效组中的所有对象的单独的无效请求。在另一个实施例中，无效命令识别组中的一个对象无效。在其他实施例中，无效命令包括多个无效请求，这些无效请求无效在组中的多个对象。

[0587] 根据步骤3432，如果无效命令引用、识别或指定组的对象为无效，组中的每个对象无效，或者组无效，则先前被服务的对象组被标识为无效。在一些实施例中，如果无效命令识别组中的对象为无效，则高速缓存管理器3232标记对象为无效。在其他实施例中，如果无效命令识别组中的对象为无效，则高速缓存管理器3232标记对象的组为无效或组中的每个对象为无效。在又一个实施例中，当通过一个或多个无效命令识别多个对象为无效时，高速缓存管理器3232仅无效对象的组。在另一个实施例中，无效命令可指定组的名称或标识符，并且高速缓存管理器3232标记该组为无效，或组中的每个对象为无效。

[0588] 在一个实施例中，设备1250或高速缓存管理器3232从高速缓冲存储器中清除已经标记为无效的对象组。在一些实施例中，仅当组中的每个对象被标记为无效时，在组中的对象可被从高速缓冲存储器中清除。在其他实施例中，如果组中的一个对象已经被标记为无效，则整个组被清除。在另一个实施例中，一旦通过客户端10接收到对于对象组或者组中的任何对象的请求，标记为无效的对象组或标记为无效的组中的任何对象被清除。在其他实施例中，一旦从提供组中的一个或多个新对象的服务器30接收到响应，标记为无效的对象组或标记为无效的组中的任何对象可被清除。

[0589] 接着是上述描述的实施例的实例。客户资源管理(“CRM”)应用被许多商业使用以追踪和评价资源管理的所有方面。经常，通过专用或包括Internet的公共网络，实施或访问CRM应用。提供对大量数据的访问的这些应用经常被访问，因此受益于由这样的应用产生的数据的高速缓存。例如，经常产生销售报告并提供给远程连接用户。这些销售报告由相关应用通过编译来自销售的信息建立，该信息被邮往这样的应用服务器和/或它们的主要数据库。当许多用户请求相同的文件(例如某个销售报告)时，没有高速缓存，应用服务器必需为每个应用重新产生对象。但是，当高速缓存放置于更接近该请求客户端时，如果这样的对象可存储在高速缓存中，则可保留应用和数据库处理，包括潜在的有价值的带宽。

[0590] 由于每次新的销售都邮往运行在源服务器(或到其主要数据库)的应用，在销售报告中的信息需要被更新，则出现了用于高速缓存这样的对象的挑战。结果是，已经存储在任何支持这些应用服务器的高速缓存中的所有的销售报告必需被无效并从高速缓冲存储器中清除其内容。然而，用于高速缓存的传统的方法不能精确确定对主要数据库或程序的改变何时将发生并且因此不能合理评价动态内容的新鲜度。每次在数据库或应用程序或源服务器中发生改变，该高速缓存必需能够识别已经作出的改变，以及哪个对象组将被无效以作为这个改变的结果。无效命令的产生，如上所述，可满足该需要，该无效命令包括链接到

前述服务的对象的组的对象确定因子。

[0591] 相关对象的多个组可在单一的分级级别中形成。可选的，可形成对象的子组以建立多个分级级别。在一个实施例中，对象的组或子组可通过用户设定。在另一个实施例中，用户可建立规则，通过该规则设备1250自动形成相关对象的组，以及关联其中的对象确定因子。

[0592] e.在客户端请求或响应中的对象确定因子的标识

[0593] 通过对象的组的产生以及实现参数化的无效，实施例也满足可以识别所有的对象的需要，这些对象被在源应用服务器30(和/或主要数据库)的状态改变所影响。在此实施例中，通过例如来自客户端的拦截的HTTP请求，任何对象或对象的预先定义的组可被无效，该高速缓存解析该HTTP请求以识别该对象确定因子。术语“对象确定因子”指唯一的或者以其他方式引用、识别或指定一个对象或一组对象的任何信息、数据、数据结构、参数、值、数据模式、请求、回复或命令。在一些实施例中，对象确定因子是在通信中的字节或字符的模式，该通信可与对象相关联或用于唯一的识别该通信与该对象相关联或引用该对象。在一个实施例中，对象确定因子指示在源服务器中是否对先前服务的对象的组的改变已经发生或将要发生，该先前服务的对象的组存储在对象确定因子相关联的高速缓存管理器3232中。在一些实施例中，在对象组中的对象被关联，以使它们与至少一个对象确定因子相关联。特别的，下面更全面的描述对象确定因子非限定的实施例以及它们应用的进一步示例。

[0594] 在本实施例的一些实例中，对象确定因子为包括或嵌入到客户端中的请求或回应中的某些预先定义的参数或数据结构。在另一个实施例中，客户端10、服务器30或设备1250嵌入到与一个或多个对象确定因子的通信中，例如代表对象确定因子的预先定义的字符串或参数集。对象确定因子指示这样的请求是否将对在对象状态中引起的改变产生影响，该对象的状态存储在源服务器30或其链接的数据库中。在一个实施例中，在请求中的对象确定因子的存在指示改变已经或将要在对象上发生。在另一个实施例中，对象确定因子的语法、结构、参数或值指示改变已经或将要在对象上发生。在一个实施例中，高速缓存从客户端10接收对象请求。该请求可包括某些参数或值(对象确定因子)，这些参数和值由高速缓存识别，将改变源服务器或应用服务器的状态，这些服务器随后将使高速缓存管理器3232存储的某些相关对象失时效，通过这样的源服务器或应用服务器30已经被预先产生某些相关对象。依赖于由用户设置的无效策略，该参数(对象确定因子)可通过源服务器请求已经被存储在高速缓存中的一个或多个预先服务的对象或对象组的无效。配置该高速缓存以识别将被这些状态改变影响的相关对象(例如，这些对象或对象组被链接到对象确定因子)，并且通过标记每个对象为无效和/或从高速缓冲存储器清除这样的对象的方法来无效这些对象。

[0595] 以上描述的技术在图29C中示出。作为此处描述的其他实施例，步骤3410包括在高速缓存中存储对象，例如从源服务器的先前服务的动态产生对象。该对象可通过运行在源服务器30上的应用产生，或例如，通过源服务器30访问数据库中提取。在一些实施例中，动态产生对象被识别为不能高速缓存的或以其他方式不识别为可高速缓存的。

[0596] 根据步骤3421，高速缓存在客户端和服务器之间拦截或以其他方式接收通信，诸如来自客户端的请求或来自服务器的响应。在一些实施例中，请求用于特定的对象，该对象已经被预先服务并存储在高速缓存中。在另一个实施例中，通信包括来自带有被请求的对

象的服务器的响应。在一个实施例中，这样的接收或拦截根据建立的高速缓存协议或通信标准发生。虽然高速缓存管理器3232或设备1250通常被描述为接收请求、响应或通信，在接收这样的请求、响应或通信中，高速缓存3232或设备1250可通过任何合适的方法或机制拦截或获得请求、响应或通信，甚至不直接或明确的与高速缓存通信。

[0597] 在步骤3423，在拦截的通信中识别对象确定因子。高速缓冲管理器3232可提取、解释、解析、访问、读取或以其他方式处理被拦截的通信以确定或识别在通信中的一个或多个对象确定因子。可使用通信的一个或多个字符的任何参数、值、语法、数据、结构或集来识别对象确定因子。在一个实施例中，高速缓存管理器3232可识别从客户端10到服务器30的请求中的对象的名称或标识符，其中该客户端请求该对象。在另一个实施例中，高速缓存管理器3232可识别在客户端10的请求或者来自服务器30的响应中的第一对象的名称或标识符，该响应指示对存储在高速缓存中的第二对象的改变已经发生或将要发生。在其他实施例中，高速缓存管理器3232确定请求中的任何模式的字符与高速缓存中的对象组或者对象相关联的对象确定因子是否相匹配。在一些实施例中，对于当前没有存储在高速缓冲中的对象可以确定对象确定因子。在其他实施例中，对于当前标记为无效的对象可以确定对象确定因子。在其他实施例中，用于请求的对象的对象确定因子被确定为将与高速缓存的对象的对象确定因子相关联。在又一实施例中，根据用于在通信中的对象的第一引用、请求或应答，高速缓存管理器3232建立所识别的对象确定因子作为用于该对象的对象确定因子。

[0598] 通过接收和解析通信，例如客户端请求或服务器应答，以识别对象确定因子，高速缓存管理器3232或设备1250可有效的确定已经与识别对象确定因子相关联的高速缓存的对象是否标记为无效。因此，根据步骤3425，做出关于对象确定因子指示高速缓存的对象是否改变的决定。在一些实施例中，被识别的对象确定因子可为通信的一部分，该部分没有改变、修改或产生对象。在其他实施例中，被识别的对象确定因子是通信的一部分，该部分指示与对象确定因子相关联的对象的改变已经发生或将要发生。例如，通信可为用于动态产生对象的获得请求或者提交请求，该提交请求将改变用于一个或多个动态产生对象的数据。在一些实施例中，在通信中的对象确定因子的存在指示在一个或多个对象上的改变已经或将要发生。在另一个实施例中，在通信中的命令、指示或指令的类型或名称连同对象确定因子指示在一个或多个对象上的改变已经或将要发生。在进一步的实施例中，参数的存在、值或设定，或命令、指示或指令的变量指示在与对象确定因子相关联的一个或多个对象上的改变已经或将要发生。

[0599] 在其他实施例中，高速缓存管理器3232在拦截的通信或对象确定因子上执行hash函数、算法或操作，以确定改变是否在对象中已经发生。在一些实施例中，该hash值与先前被存储的用于对象的hash值相比较，如果不同则高速缓存管理器3232识别对象已经改变。在又一个实施例中，用于对象的hash值可被包括在通信或对象确定因子中。在一个实施例中，通信指示对象已经被参数的值或设定改变，例如带有Boolean标志。在其他实施例中，以下将详细描述的实体标签控制和确认机制可被用于识别对象并确认对象是否已经改变。

[0600] 如果改变被指示，则在步骤3431，接着，与对象确定因子相关联或被对象确定因子识别的对象被标记为无效。在一些实施例中，根据步骤3431，由被拦截的通信请求的对象被标记为无效，并根据步骤3440从源服务器30获取。否则，在其他实施例中，被请求的对象根据步骤3450从高速缓存中被获取。在一个实施例中，被标记为无效的任何对象将从高速缓

存中清除。

[0601] f.基于对象确定因子对象组的无效

[0602] 以上的实施例描述了基于在客户端请求中的对象确定因子的标识，无效在高速缓冲管理器3232中先前服务的对象。在另一个实施例中，这个一般的概念可被用于识别或无效对象组，一个或多个对象确定因子已经与该无效对象组相关联。该实施例将在图29D中示出。

[0603] 在图29D中描述的方法以与方法图29C同样的方式开始。步骤3410包括在高速缓存中存储对象，例如先前从源服务器服务的动态产生对象。在一些实施例中，一个或多个对象没有被识别为可高速缓存的。根据步骤3412和类似于图29B，先前被服务的对象被形成组。在一个实施例中并且根据对象确定因子技术，对象组与至少一个对象确定因子相关联并被该至少一个对象确定因子识别。就像以下更充分描述的，在一些实施例中，组和对象确定因子的关系依赖于用户高速缓存策略的性质和细节，例如被策略引擎3236使用、定义、控制的策略。在其他实施例中，组的一个或多个对象确定因子包括组中的对象的一个或多个对象确定因子。在另一个实施例中，组的对象确定因子包括组中的对象的对象确定因子的组合。

[0604] 根据步骤3414，维护组的记录，连同其相关联的对象确定因子，如果可应用的。该步骤相似于图29B中示出的步骤3414。在一个实施例中，记录和/或组的任何对象确定因子在查询表中被维护。在其他实施例中，记录和/或组的任何对象确定因子可以hash表的格式被维护。可设计该hash表有效率的存储非邻近关键字，在字母和数字顺序中，hash表有很大的差距。在另一个实施例中，可在hash表之上建立索引系统。在一些实施例中，高速缓存管理器232维护作为组的对象与在数据库或数据结构中的一个或多个对象确定因子或在存储器中的对象的关系。在进一步的实施例中，在组中的每个对象的标志、属性或性质被指定或设定为识别该组的值，例如值等于，识别或引用组的名称或标识符，或者组的对象确定因子。在一些实施例中，对象的组在被识别作为保持该组的高速缓冲存储器的部分中被布置、放置或定位。在另一个实施例中，与对象的组相关联的一个或多个对象确定因子被存储。

[0605] 步骤3421和3432类似于如图29C示出的步骤3421和3423。根据步骤3421，高速缓存管理器3232或设备1250在客户端10和服务器30之间拦截或以其他方式接收通信，例如来自于客户端用于先前服务和存储在高速缓存中的对象的请求。在一个实施例中，高速缓冲管理器3232拦截从客户端10到服务器30的请求。在一些实施例中，该请求用于存储在高速缓存中的对象。在其它实施例中，该请求是到服务器30的指令、命令或指示，将引起存储在高速缓存中的对象的改变，使得对象被动态产生。在另一个实施例中，高速缓存管理器3232拦截从服务器30到客户端10的包括或识别存储在高速缓冲中的对象的响应。

[0606] 在步骤3432，在拦截的通信中识别对象确定因子。如上所注释的，对象确定因子指示在源服务器30被请求的对象中是否已经发生或将要发生改变。然而，在图29D的实施例中，对象确定因子可与对象的组相关联。这启用有效的将存储在高速缓存中的全部对象无效，其可能被特定对象确定因子所影响。在一些实施例中，识别在组中的对象的对象确定因子。在其他实施例中，识别用于对象的组的对象确定因子，例如，组对象确定因子。在另一个实施例中，识别组中的一个或多个对象的对象确定因子的组合。

[0607] 因此，根据步骤3427，做出关于对象确定因子是否指示先前被服务的对象的组中的改变的决定。在一些实施例中，在拦截的通信中的组的对象确定因子的存在指示组中的

一个或多个或所有的对象的改变已经发生或将要发生。在其他实施例中，在拦截的通信中的命令、指示或指令的名称和类型指示这个改变。在又一个实施例中，在通信中的任何参数或变量的存在、值或设定也可指示这样的改变。

[0608] 如果在步骤3427，对象确定因子指示组中的改变，接着根据步骤3435，先前被服务的对象的组在高速缓存中被标记为无效。在一些实施例中，根据步骤3440，组中的一个或多个或所有的对象被从源服务器30请求和获取。如果在步骤3427，对象确定因子没有指示组中的变化，则根据步骤3450，在一些实施例中，从高速缓存管理器3232中获取对象作为被拦截的通信的一部分被请求的任何对象，并且该对象先前被服务和存储在高速缓存中。在实施例中，标记为无效的任何对象或对象组被高速缓存管理器3232从高速缓存中清除。

[0609] g.组的指示

[0610] 高速缓冲的管理员可特别指定被包括在特定组中的对象。无论任何时候对象被存储在高速缓存中，管理员可依赖于配置使该对象为配置或隐式(implicit)组之一的成员。被配置的组可基于配置，该配置为管理员先前建立或可选的基于应用行为和涉及对象无效的其他数据。如果对象的配置组是动态的，则对象也可为隐式组的部分。在隐式组中的对象通过有效的无效参数的值被分组。

[0611] 通过允许对象的非常灵活的分组，高速缓冲可以获得无效中的灵活和协调水平，这对于高速缓冲动态产生内容是有必要的。高速缓存可同时无效对象的非常特别的组，因此使得高速缓存更能响应于无效动态产生内容的频繁需要。这时，为了关联的一个或多个对象确定因子，高速缓冲分配对象到组中，该组确定许多涉及到该对象的事情，包括无效参数和命中确定因子。

[0612] 在客户资源管理(“CRM”)实例中，高速缓存管理员可预先指定每个分组。例如，管理员配置高速缓存以便通过名称分组每个销售部门。因此管理员可以指定汽车部门、摩托车部门等，并且在每次到达高速缓存的请求中识别对象确定因子，该高速缓存可接着无效存储在指定组中的所有对象，该指定组通过对象行列市链接到合适的部门。

[0613] h.基于规则分组

[0614] 可选的，高速缓存管理员可建立规则，该规则允许高速缓存设备在运行中确定包括在特定组中的对象。这种基于规则的分组可通过虚拟建立的规则依赖于组的指定，该已建立的规则将重要的对象确定因子连接到该对象，高速缓存使用该重要的对象确定因子生成相关的组。该方法的实例可包括以规则配置高速缓存，高速缓存使用该规则来识别什么对象放于每个组中。

[0615] 再一次看CRM实例，规则可规定，建立在应用上的销售部门的细分将被高速缓存识别为自身的分组。在这种方式中，在没有高速缓存管理员必需特定识别每个分组时，建立分组，但是允许高速缓存基于相关规则确定。此技术生成很方便并常常为小的工作强度的方式来指定分组。高速缓存管理员可配置规则，该规则规定每个销售部门的细分(例如，销售\汽车，销售\摩托车等)通过高速缓存产生新分组。当来自汽车销售部门的请求通过高速缓存被处理并由应用返回时，高速缓存基于预先配置的规则，可识别销售的每个子分组并为其自动生成分组。

[0616] 通过高速缓存可实施该规则，每次高速缓存获取用于报告/销售/汽车或报告/销售/摩托车等的类型的对象的新请求。当高速缓存识别这些子分组并为它们中的每个建立

对象分组时，当摩托车销售部门请求示出它是销售部门的子分组时，该过程可接着被重复，接着自行车销售部门等等。当已知的无效请求来到链接到这些分组中的一个的高速缓存时，或者如果相关对象确定因子在客户端请求中被识别时(例如，在解析该请求中发现邮送到摩托车销售部门销售/摩托车的商业报告)，高速缓存确定来无效摩托车销售部门分组中的所有高速缓存的对象。

[0617] 在此方式中，当高速缓存认识到对被应用服务的数据的改变已经发生或将要发生(或者由于高速缓存认识到被高速缓存接收的请求的内容将触发在应用的改变或者由于一些外部改变的发生)时，上述技术启用高速缓存来快速并简单的识别通过分组过程哪个对象需要无效。以此方式，高速缓存可以无效由于应用或数据库状态中的改变而不再新鲜的大量动态产生的对象。

[0618] 使用智能统计引擎也可增强高速缓存成功存储和服务其高速缓冲存储器动态产生内容的能力，为了在一段时间内确定对象的设置，该智能统计引擎检查请求模式和响应话务量，该对象的设置将提供最大高速缓存利益。该引擎或者可以集成到自身的高速缓存设备中，或者作为启发式运行在单独的计算机中，以选择对象的一些子集，用于进一步探测以确定用于动态存储的适宜性。

[0619] i.对象确定因子的进一步使用

[0620] 如以上描述的，对象确定因子可为指示在源服务器中的改变是否已经发生或将要发生的任何数据结构，该改变为对存储在对象确定因子与之关联的高速缓存中的先前被对象的组。对象确定因子可基于嵌入到请求中的预先定义的字符串值建立。例如，当请求带有特定USERID来到时，该USERID可被链接在高速缓存存储器中的对象的组，每次从该特定USERID送来邮寄或其他请求时该对象的组被无效。用于对象确定因子的可能候选也可包括使用服务器的服务标识符，该服务器是最初服务该对象的。该服务标识符包括服务IP地址、TCP端口和出现在HTTP请求中的服务标识符。

[0621] 另一个可能的对象确定因子出现在请求统一资源定位器(“URL”)中。在高速缓存静态对象例子中，请求的URL通常足以唯一的识别对象。然而，用于动态产生内容的请求，出现在URL中的信息可能不足以识别高速缓存对象。高速缓存必需因此在请求中检查其他信息，以发现包括在HTTP头、cookie头在其他常用HTTP头中的对象确定因子。高速缓存可另外在客户端请求中的多个其他位置中查找相关参数信息的子集，包括但不限于：在URL查询串中，在POST体中，在cookie头中，或者在任何其它请求或响应头中。

[0622] 解析用于对象确定因子的URL的问题是URL和其他头可包括除了与高速缓存的决定相关的信息之外的很多信息。因此，高速缓存必需能够解析大量信息，以能够识别合适的对象确定因子。另外，在头中的数据经常被任意的排序，这意味着数据放置在HTTP头中没有标准的方式，因此简单的比较常常不足以定位在这样的串中的相关对象确定因子。

[0623] 如果没有预先配置的策略使特定对象确定因子与存储在高速缓存中的相关对象或对象的组匹配，则在另一个实施例中，高速缓存可仍然作出这样的决定。例如，高速缓存可检查并解析请求的各个方面以发现是否任何其他对象确定因子在这样的请求中被找到并用于链接这样的请求到存储在应该被无效的高速缓冲存储器中的特定对象。可选的，也可启用高速缓存检查用于某些对象确定因子的请求，该对象确定因子是高速缓存基于某些预先定义的启发式确定可有目的的链接到特定对象或对象的组。例如，当请求到达高速缓

存用于更新与特定USERID相关联的日历时，可建立实施例以识别USERID等于更新日历请求的USERID的所有高速缓存的对象，以及用于任何一个特定日子的包括用户的日历的(对象)将需要被无效。

[0624] 高速缓存也可假定对象确定因子作为name＝value或者相似对的组以非特定顺序中出现在URL词干(Stem)中、在URL中提供的查询中、在POST体中或在Cookie头中。在实施例中，假定查询被格式化为name＝value对的列表。用户可以因此配置哪个参数名称是重要的。每个高速缓存的对象使用第一个它的访问URL键入。URL可看起来像/site/application/special/file.ext?p1＝v1&p2＝v2&p3＝v3。/site/application/special/file.ext部分是URL词干。p1＝v1&p2＝v2&p3＝v3部分是URL查询并包括参数-值对。这些参数-值对也可出现在POST体中或者Cookie头中。

[0625] 在一个实施例中，用户或管理员建立该P1和P2将为无效的参数或对象确定因子。高速缓存其后将自动分组拥有匹配的p1和p2值的对象。实施该分组的一种方法是将p1和p2映射到在数据库表中的主要关键字，即，在表中唯一的可以识别的对象，该高速缓存为了确定无效状态将知道怎样引用该对象。为了更新这些数据库表中的内容，为了反映存储在高速缓存中的数据不再有效的事实，高速缓存将为p1和p2指明新值并且当高速缓存识别这个新值时，下一时间将服务这个内容，高速缓存将知道无效存储在它的存储器中的被链接的对象。当高速缓存遇到这样的请求时，当见到更新请求则高速缓存知道必须无效具有匹配的p1和p2值的组——因为高速缓冲知道在源端的数据将改变，因此影响涉及到这些p1和p2对象确定因子的所有对象。

[0626] 为了解决更复杂的情况，其中管理员没有预先配置的嵌入到请求中作为对象确定因子的特定参数，高速缓存可部署用户配置策略以从请求中提取相关对象确定因子从而帮助识别何时无效对象的分组。接着使用确定因子串以便定位存储在高速缓存中的对象的组以及无效这样的对象。可使用这些对象确定因子来配置高速缓存以产生重要的参数值的列表。如果输入的写请求已经匹配用于这些重要参数的值，则连接到这些参数名称的对象将被无效。可选的，用户可指定策略架构动作，该动作可从请求中提取对象确定因子串。对象确定因子串从写请求中提取，并且带有匹配的确定因子串的所有对象被无效。在该可选方法中，请求到达高速缓存，高速缓存做出请求串是否匹配无效策略的决定。无效策略指定其中的内容组将被无效的对象。

[0627] 可选的，高速缓存可使用可被呈现在客户端请求中的任何其他用户信息。如上所述的，验证和授权的集成允许高速缓存访问用户信息。如果高速缓存的对象的相关分组被链接到用户或用户的组，USERID或者GROUPID可为确定因子之一。虽然用户信息常常是重要的对象确定因子，但是用户信息常常不能被呈现在HTTP请求中。在进一步的实施例中，动态高速缓冲方面可与系统和方法相结合，该系统和方法用于将带有各种其他网络元件与高速缓存集成，该元件包括执行特定种类验证、访问控制与核查(AAA)下部构造的能力。因此，根据由应用产生的数据的安全级别被应用到改为从高速缓存服务的数据上。此技术允许应用来高速缓存敏感的、访问控制信息，该访问控制信息不能以其他方式被高速缓存。

[0628] 该方法允许高速缓存识别用户，该用户在HTTP请求中不包括可确认的用户信息，但该用户可通过在集成高速缓存专利中描述的AAA方法识别。这样的方法启用高速缓存来通过检查可从AAA处理中共享的授权状态信息以识别相关用户到特定请求。在进一步的实

施例中，集成启用安全策略的应用到存储在高速缓存中的信息以禁止非授权用户访问存储在高速缓存上的信息。

[0629] 该方法也提出由这样的事实所引起的挑战，即动态产生数据的重要的部分需要客户端请求在高速缓存响应来自于客户端的相关请求之前被验证和授权的这样的数据。高速缓存必需有由验证用户授权请求的能力，以便应用可以高速缓存访问控制对象，并通过集成带有验证和授权信息与这样的动态高速缓存技术，而获得安全性。如果对象为用户或用户组定制，则USERID或者GROUPID将是对象确定因子之一。因此根据由应用产生的数据的安全级别也被应用到高速缓存的信息。这个技术允许应用来高速缓存敏感的、存储控制信息，该信息不能以其它方式被高速缓存。

[0630] 最后，其他信息像日子的时间、在发端数据库的状态等可从请求中解析并用作对象确定因子来确定是否存储在高速缓存中的对象仍然有效。高速缓存可以通过在对象分组中配置合适的终止行为来管理这种情形，对象被配置从而对这样的外部变化敏感。

[0631] 为了进一步提出呈现在请求必须通过高速缓存解析和拦截的动态内容事实面前的挑战，根据实施例的动态缓存能够限定哪个参数一定是用于高速缓存的相关对象确定因子。以此方式，用于从高速缓存服务对象而非转发这样的请求到可应用的应用服务器的成功的比率被增强。通过示例，从客户端查询的请求可包括城市和状态参数。然而，可配置高速缓存满足应用的需求，其中高速缓存正存储内容以识别响应可被提供给从客户端来的请求，该查询示出该来自于所有客户端的没有考虑到城市值的请求。为了这个目的，城市参数不是相关的，并且高速缓存可以识别此事实。可选的实施例包括配置高速缓存以便如果只有城市参数匹配而无论状态参数指明什么，服务可从高速缓存响应。

[0632] 总之，高速缓存实施普通的参数化对象匹配。在此方式中，配置高速缓存以识别在请求中的信息的子集，该子集作为对象确定因子是有用的，并且该子集被链接到特定的对象，以便当这样的对象确定因子被识别时，高速缓存可以使用在评价对象或对象的组保持最新并可以从高速缓存被服务中该确定因子的存在(或相反这样的确定因子的缺乏)。该高速缓存维护表，该表在请求进入时查询，以检查配置参数从而确定是否被请求的数据保持最新，并且也允许高速缓存匹配相关数据到存储在高速缓存存储器中的固有对象。

[0633] j.具体化数字(Incarnation Number)

[0634] 在又一个实施例中，高速缓存可使用具体化数字无效对象组。由于初始时状态的改变，在高速缓存需要同时改变对象每个组的状态的情况下，具体化数字为实现此无效提供了简单的技术。尽管识别每个对象并改变每个的状态是确定存储在高速缓存中的数据的新旧程度的没有效率的方法，使用具体化数字使得利用更加简单和有效的方法来无效对象组。本实施例描述了每个对象怎样指向代表组的数据结构，并且因此服务器仅仅需要发送改变用于组的数据结构中的状态的命令。当用于高速缓存的对象的随后请求从客户端到达时，高速缓存必须首先指出状态是否已经改变。为了做这个，高速缓存查询数据结构以指出用于组的状态是否已经改变。

[0635] 为了有效地实施数据结构，高速缓存必须能够确定是否查询状态改变。因此，高速缓存必须可以决定是否已经在组中看到状态改变，或者没有。这就是具体化数字有益之处。高速缓存将动态产生对象关联到内容组中。每个这样的内容组可通过hash表查找过程使用包括在数据结构中的特定的索引值或者“具体化数字”而被呈现。之后，无论任何时候高速

缓存接收高速缓存识别引起状态改变的客户端请求，客户端解析相关参数的客户端请求，基于识别的对象确定因子执行hash查找，并增加在数据结构中的索引或具体化数字。每次当存储在指定的分组中的对象被客户端请求时，高速缓存在对象上执行hash算法，并将其与用户内容分组的数据结构中的初始存储值相比较。如果存储的值与为该对象由高速缓存计算的数字相等，则高速缓存知道内容仍然保持最新，并且可被提供给请求者。如果高速缓存探测到为这样的对象计算的当前具体化数字和数据结构中用于这样的内容分组所存储的数字间存在差异，高速缓存知道被存储的对象不再是最新的。高速缓存接着无效被存储的对象，并向应用服务器发送该请求。当响应返回高速缓存时，设备将在高速缓冲存储器中存储新的响应，并再次将该响应链接到新的数据结构。之后，每次高速缓存在该分组中对象的请求，高速缓存可做出比较并假定没有对数据结构作出进一步的改变，高速缓存可服务最新的存储对象。

[0636] 通过以此方式使用对象的组的无效，高速缓存可以非常快速的无效一并且无论被无效的对象的数量是多少，花费的时间是常量。通过该更快速更有效率的无效的处理，该技术启用高速缓存来更有效的控制动态产生对象。由于这样的数据中的迅速改变，该方法允许位于应用前的高速缓存设备更加积极的存储和服务动态产生对象而不用服务无效或失时效的内容。该实施例启用高速缓存来服务经常或不可预知的改变以提供高速缓存的性能的数据。该高速缓存也可使用用户命令并且也可通过检验和分组各种web话务量来无效存储在高速缓冲存储器中的对象以及对象组。

[0637] 2.连接池

[0638] 在一个实施例中，网络设备1250(此处也成为接口单元1250)减轻服务器30大量的处理负载，引起处理负载的原因是通过打开一个或多个与服务器的连接来重复打开和关闭到客户端的连接并保持这些连接从而使得客户端通过Internet可以进行重复的数据访问。该技术此处称为“连接池”。

[0639] 为了完整，下面关于图30简单的描述连接池的操作。当客户端10请求访问在接口单元1250管理的服务群组中的一个服务器时，图30中的处理开始。如在步骤4302中所示的，在接口单元1250和请求客户端之间的连接被打开，并且接口单元1250接收客户端请求以访问服务器。如步骤4304所示的，接口单元1250确定被请求的服务器的标识。在一个实施例中，这是通过检查由客户端请求指定的目的网络地址实现的。在另一个实施例中，这是通过检查由客户端请求指定的网络地址和路径名实现的。

[0640] 如在步骤4306中所示的，在确定客户端请求将被定向的服务器30的标识之后，接口单元1250确定是否到服务器的空闲连接(即，没有不使用的一个)已经打开。如果打开，处理在步骤4310重新开始。如果没有，如在步骤4308中所示的，接口单元1250向服务器打开连接。如在步骤4310中所示的，接口单元1250接着翻译客户端请求并传递该请求到服务器，并且以下将结合图31更全面的描述。如在步骤4312中所示的，在服务器处理之后，接口单元从服务器接收响应。如在步骤4314中所示的并且下面将进一步描述的，服务器响应被翻译并传输到请求客户端。最后，如在步骤4316中所示的，接口单元1250关闭与客户端的连接。然而，接口单元1250和服务器之间的连接没有断开。通过与服务器维持打开连接，并根据需要打开和关闭与客户端的连接，接口单元1250解决了服务器30几乎所有的与服务Internet上客户端相关联的连接负载问题。

[0641] 就像以下将进一步详细描述的，一些实施例涉及到步骤4316，此处接口单元1250关闭与客户端10的连接。存在多种情况导致接口单元1250关闭与客户端的连接。例如，客户端可开始FIN(完成)命令或RST(复位)命令。在这两个情况中，接口单元1250等待直到其失去本身和客户端之间的连接之前接收到这些命令中的一个。当客户端不使用或结束该连接但是一段时间内不向接口单元1250传递该信息时，发生无效率的连接池。由于接口单元1250为了重新使用用于另一个客户端的连接而正等待来自客户端的命令，该连接不能成功地连接。

[0642] 为了下面更详细的描述，即使在接收对请求的服务器响应之后，超文本传输协议(HTTP)1.1(默认)以及HTTP 1.0(带有连接:Keep-Alive Technique(保持激活技术))启用客户端和/或接口单元1250以保持与服务器的连接打开。客户端和/或接口单元1250可接着通过相同的连接立即或者大量时间(或者“思考时间”)之后发出其他请求。当客户端的人类操作者正决定浏览器上的下一个连接的点击以及类似时，客户端处于“思考时间”。即使服务器不处理通过连接的任何请求，这可导致服务器保持连接。此处，服务器管理员可能被迫通过设置保持激活超时时间而监视服务器上的太多的同时的连接，在该暂时休息时间之后已经被空置的或处于“思考时间”的连接被关闭。一个实施例允许当客户端10处于“思考”中时，通过客户端10’使用到服务器的连接。当然，如果当客户端10正使用服务器连接时客户端10’做出请求，则客户端10’必须使用到服务器的不同的连接。然而，当非常小数量的连接被超过并移到通常实例中时，将实现一个实例的有效连接池。一般情况是当为‘n’个客户端连接可为统计复用到‘m’个服务器连接，此处‘n’大于‘m’。

[0643] 如在步骤4310和4314中所示的(图30)，图31是示出翻译客户端和服务器请求的一个实施例的操作的流程图。在一个实施例中，消息话务量是以TCP/IP包的形式的本领域公知的协议组。TCP/IP协议组支持许多应用，例如Telnet、文件传输协议(FTP)、e－mail以及HTTP。该实施例以HTTP协议的术语描述。然而在阅读完该说明书之后，该概念可同样适用于其他TCP/IP应用，这对于本领域的技术人员是显而易见的。

[0644] 每个TCP包包括TCP头和IP头。IP头包括32位源IP地址和32位目标IP地址。TCP头包括16位源端口号和16位目的端口号。源IP地址和端口号，共同指向源网络地址，唯一的识别包的源接口。同样，目的IP地址和端口号，一起指向目的网络地址，唯一的识别包的目的接口。包的源和目的网络地址唯一识别连接。TCP头也包括32位序列号以及32位确认号。

[0645] 包的TCP部分称为TCP段。TCP段包括TCP头和体。TCP段的体部分包括HTTP头和消息。有两个机制用于确定消息长度，包括基于块传输编码的一个和基于内容长度的另一个。在HTTP头中找到内容长度头文件。如果内容长度头的域存在，它的字节值代表消息体的长度。可选的，如果块传输编码头出现在HTTP头中，并且示出“块”传输编码已经被应用，则消息的长度通过块编码限定。为了将消息作为一连串块传输，每一块带有包含在块尺寸域中的自身的指示器，块编码修改消息体。

[0646] 作为以下将详细描述的，一个实施例使用内容长度参数和/或块传输编码头，通过避免客户端处于“思考时间”的情形而增加在服务器和客户端之间连接池的效率。没有此实施例，接口单元1250或者在重用用于一个客户端的链接之前等待来自另一个客户端的命令，或者当连接已经空闲太长的时间时连接被中止。

[0647] 以上提到的32位序列号识别在从发送TCP到接收TCP的数据串中的字节，数据的第

一个字节在TCP段中表示。由于被交换的字节是被编号的，因此确认号包括确认的发送者希望接收的下一个序列号。这就是因此序列号加上最后一个成功接收的数据的字节之一。校验和覆盖TCP段，即，TCP头和响应数据(或体)。这是强制域，必须被发送者计算和存储，并且接着被接收者验证。

[0648] 为了从客户端成功的路由入站包到目的服务器，或者从服务器路由出站包到客户端，接口单元1250使用公知的“网络地址翻译”过程。网络地址翻译为本领域所公知，并且通过Internet标准(草案)RFC 1631指定，可在URLhttp://WWW.safety.net/RFC1631.txt中找到。

[0649] 然而，为了无缝的结合客户端和服务器连接，在共有的美国专利申请No.09/188709中详细描述了新的翻译技术，该申请1998年10月10日申请，题目为“因特网客户端-服务器多路复用器”，此处成为“连接复用”。根据该技术，通过在TCP协议层修改包的序列号和确认号来翻译该包。该技术显著的优点在于不需要应用层交互。

[0650] 现在看图31，如在步骤4402中所示的，翻译包的网络地址。在入站包的例子中(即，从客户端接收包)，包的源网络地址被改变到接口单元1250的输出端口的地址，并且目的网络地址被改变到目的服务器的地址。在出站包的实例中(即，从服务器接收的包)，源网络地址从服务器的地址被改变到接口单元1250的输出端口，并且目的地址从接口单元1250的地址被改变到请求客户端的地址。包的序列号和确认号也被翻译，如步骤404和406中所示并在下面进一步详细描述。最后，如在步骤4408中所示，重新计算包的校验和以说明这些翻译。

[0651] 如以上所提及的，实施例特别涉及用于通过内容-长度参数和/或块传输编码头来快速池化网络客户端-服务器连接的设备、方法和计算机程序产品以增加服务器和客户端之间的有效的连接池。效率中的增长其结果可以在客户端处于“思考时间”时避免占用连接。在一个实施例中，使用内容长度参数确定消息的长度。在另一个实施例中，使用块传输编码确定消息的长度。接着，两个实施例将分别结合图32和33加以描述。

[0652] 图32示出了被称为TCP段4500的TCP包的TCP部分。TCP段4500包括TCP头4502和体4504。体4504在其它信息中包括HTTP头和消息。内容长度参数4506在HTTP头中找到。实施例怎样使用内容长度参数4506提供更有效率的连接池将在下面结合图35和36加以描述。

[0653] 图33示出了被称为TCP段4600的TCP包的TCP部分。如上所述，如果块传输编码头出现在HTTP头中，并且示出“块”传输编码已经被应用，则消息的长度通过块编码被限定。为了作为块序列传输消息，每个块带有包含在块尺寸域中的自身的指示符，块编码修改消息的体。TCP段4600包括TCP头(未示)以及体。除了其它信息之外，体还包括HTTP头4602A-4602C以及消息。HTTP头4602A-4602C由七个块尺寸域4606A-4606G组成；以及六个块消息数据4604A-4604F。

[0654] 如图33所示，块尺寸域4606A-4606G被连接到一起。块尺寸域4606A示出在块消息数据4604A中的消息的长度，块尺寸域4606C示出在块消息数据4604C中的消息的长度，依此类推。最后的块尺寸域4606G总是包括长度值零，指示没有消息数据跟随。这指示所有的消息已经被发往客户端。实施例怎样使用块尺寸域4606A-4606G来提供更有效率的连接池，下面将结合图37和38加以描述。值得重点指出的是图33中的TCP段4600仅仅用于说明。

[0655] 在详细描述实施例怎样使用内容长度参数来增加连接池的效率之前，连接池在美

国专利申请NO.09/188709中首先讨论完整性，该专利在1998年11月10日申请，题目为“因特网客户端-服务器多路复用器”。图34是示出连接池的消息流程图。图34示出了将两个客户端C1和C2连接到服务器S的接口单元1250。两个客户端C1和C2可包括此处讨论的任何客户端10，并且服务器S可包括此处讨论的任何服务器30。首先，如流程4702所示的，接口单元1250使用客户端C1提供的网络地址1打开与客户端C1的网络连接。由于TCP/IP使用多阶段握手打开连接，所以流程线4702所示为两路流程。

[0656] 如通过流程线所示的，一旦连接被打开，接口单元1250从客户端C1接收GET请求，指出/sales/forecast.html的路径名。由于在接口单元1250和服务器S之间没有空闲连接打开，接口单元1250打开与服务器S的连接。如流程线4706所示的，接口单元1250将该请求映射到网络地址2，该网络地址2指服务器S。如流程线4708所示的，接口单元1250也向该服务器传送GET请求。如流程线4710所示的，服务器S以被请求的web页面来响应。如流程线4712所示的，接口单元1250向客户端C1转发该web页面。最后，如流程线4714所示的，关闭在客户端C1和接口单元1250间的连接。根据TCP/IP协议，关闭网络连接可包括多阶段过程。因此流程线4714示为双向的。值得重点指出的是，接口单元1250不关闭与服务器S的连接，而是保持其打开以调节进一步的数据流。

[0657] 接着，如流程线4716所示的，使用由客户端C2提供的网络地址1打开接口单元1250和客户端C2间的连接，接着，如流程线4718所示的，接口单元1250从客户端C2接收GET请求，指定web页面/sales/forecast.html。由于在接口单元1250和服务器S之间的空闲连接已经打开，则不需要接口单元1250负担带有打开的进一步连接的处理负载的服务器S。接口单元1250仅仅使用空闲打开的连接。如流程线4720所示的，接口单元1250将GET请求映射到服务器S，传输它，并转发到服务器S。如流程线4722所示的，接口单元从服务器S接收响应，并如流程线4724所示，向客户端C2转发该响应。最后，如流程线4726所示的，接口单元1250关闭与客户端C2的连接。再一次的，接口单元1250不关闭与服务器S的连接。代替的，接口单元1250保持该连接打开以调节进一步的数据流。

[0658] 如上所讨论的，如流程线4724所示的，有许多的情况导致接口单元1250关闭与客户端C2的连接。例如，客户端可开始FIN(完成)命令，这发生在一旦客户端已经获取到所有被请求的数据(或消息)时。客户端也可开始RST(复位)命令。除了关闭接口单元1250和客户端之间的连接之外，RST命令导致执行许多的内务操作以保持服务器端连接情况正常。更具体地，TCP协议保证RST命令将有正确的SEQ(序列)号，以便服务器将接收TCP段；然而，RST命令不能保证有正确的ACK(确认)号。为了维护这种情形，接口单元1250跟踪服务器发送的数据的字节以及客户端确认的字节。如果客户端仍然没有确认通过服务器的所有的数据，接口单元1250计算该未确认的字节，并向服务器发送ACK。另外，服务器端PCB可被放置于超时队列中，以允许任何未决的服务器数据全部传输完。

[0659] 另外，虽然没有在图34中示出，服务器也可关闭在其自身和接口单元1250间的连接。服务器将向接口单元1250发送FIN命令。在此例子中，在服务器和接口单元1250之间的连接以及在接口单元1250和客户端之间的连接将被关闭。

[0660] 另一个方面是通过最小化服务器关闭连接的机会，最大化来自服务器的连接处理的卸载。有三个例子：

[0661] (1)使用协议版本HTTP/1.1。在该实例中，不需要明确的Keep-Alive(保持激活)头

部。通过缺省值，服务器保持连接打开；直到客户端关闭连接。实例通过重新使用服务器端连接来卸载服务器。由于直到客户端关闭连接，当客户端完成连接但在一段时间内没有向接口单元1250转播该消息时，发生连接池的无效率。由于接口单元1250正等待来自客户端的命令以便重新使用用于其他客户端的连接，该连接不需要被连接。

[0662] (2)使用协议版本HTTP/1.0，并且客户端提供“Connection:Keep-Alive”(“连接：保持激活”)头。在该实例中，服务器保持连接打开；直到客户端关闭该连接。实例通过重新使用服务器端连接来卸载服务器。如使用协议版本HTTP/1.1，当客户端完成连接但在一段时间内没有向接口单元1250转播该消息时，发生连接池的无效率。

[0663] (3)使用协议版本HTTP/1.0，并且客户端不提供“Connection:Keep-Alive”(“连接：保持激活”)头。在该实例中，服务器在完全完成一个GET请求后正常关闭连接。如果服务器在每次请求后，关闭连接，这就使得接口单元1250没有重新使用服务器端连接的机会。由于这个原因，因特网的许多仍然使用没有“连接：保持激活”的HTTP/1.0。在该说明书中新的技术用于允许重用服务器端连接，重要的例子在共有的美国专利申请No.09/188709中详细描述，该申请1998年10月10日申请，题目为“因特网客户端-服务器多路复用器”。接口单元1250检查GET包以探测该情况。当探测到这种情况时，接口单元1250向GET包中插入“连接：保持激活”。由于该操作对客户端不可见，接口单元1250必须跟踪在服务器端连接上的“字节增加”的数量。由于序列号是在第一字节中，则“字节增加”不影响GET包中的序列号。然而，接口单元1250必须随后向从客户端到服务器的包的序号增加“字节增加”。与之相反，服务器将确定附加字节，但是接口单元1250在向客户端发确认时必须减去它们--客户端不知道这些字节增加。

[0664] 如上所提出的，通过操作序列号和确认号来实现连接多路复用。由接口单元1250接收的段的序列号和确认号被修改和映射到接收者所希望的值。对于客户端，数据看来是来自于服务器并且反之亦然。例如，如果“流入”指出被接口单元1250接收的段，并且“流出”指出相应的出站段，序列号和确认号以下列的方式被改变：

[0665] 流出序列号＝流入序列号-流入开始序列号+流出开始序列号

[0666] 流出确认号＝流入确认号-流入开始确认号+流出开始确认号

[0667] 为了处理用于HTTP/1.0包的“连接：保持激活”的附加值，接口单元1250跟踪连接的合适一半上的“字节增加”-在此例中为服务器端。序列号和确认号以下列的方式被改变：

[0668] 流出序列号＝流入序列号-流入开始序列号+流出开始序列号+流出字节增加

[0669] 流出确认号＝流入确认号-流入开始确认号+流出开始确认号-流入字节增加

[0670] 当包括用于提供更多有效率的连接池的实施例的内容长度参数技术在下面结合附图35和36(涉及内容长度参数)和图37和38(涉及块尺寸域)被描述时，使用这些等式获得翻译的特定实施例。

[0671] 图35详细的描述由实施例结合内容长度参数技术执行的确认号和序列号的翻译的流程图。在图35中的用于每个流程的标记形式为T:S,A(L)，其中T表示TCP段类型，S为序列号，A为确认号，并且L为内容长度参数。内容长度参数描述了在消息中的数据的字节的数量。

[0672] 流程4802A-4802C提供打开客户端C1和接口单元1250间连接的一个方法。每个流程代表TCP段。在TCP段4802A中，设置在TCP头中的SYN标志，指示来自客户端C1的新连接请

求。客户端C1已经建立了开始序列号为2000和确认号为2000。如流程4802B所示，接口单元1250响应以SYN ACK段，指定开始序列号为4000，并增加确认号到2001。本领域所公知的，在网络内的每个实体(例如，客户端，服务器，接口单元)设置自身唯一的序列号和/或确认号。如流程4802所示，客户端C1以ACK端响应，指定序列号为2001并且增加确认号为4001。如流程4804所示，客户端C1接着发出指定49字节长度的GET段。

[0673] 如流程所示，假设接口单元1250确定没有与服务器S空闲的打开连接存在，并因此向服务器S发送SYN段，指出开始序列号1950。如在4806B中所示，服务器S响应SYN ACK段，指定开始序列号6000并将确认号增加到1951。如流程8060所示，接口单元1250响应以ACK段。如流程线4808所示，在根据以上描述的翻译等式修改序列号和确认号后，接口单元1250接着从客户端C向服务器S转发GET段。

[0674] 如流程所示，服务器S响应以被请求的数据，指定序列号为6001，确认号为2000以及内容长度参数为999。如流程线4812A所示，接口单元1250接收RESP段，翻译序列号和确认号，并向客户端C1转发RESP段。

[0675] 在该点，接口单元1250接收客户端C2的请求以打开连接。如上，流程4816A-4816C提供打开客户端C2和接口单元1250间连接的一个方法。再一次的，每个流程表示TCP段。在TCP段4816A中，设置TCP头中的SYN标志，指示来自客户端C2的新的连接请求。客户端C2已经建立了开始序列号999，以及确认号为999。如流程4816B所示，接口单元1250响应以SYN ACK段，指定开始序列号为4999，并将确认号增加到1000。如流程4816C所示，客户端C2响应以ACK段，指定序列号为1000并将确认号增加到5000。如流程4818所示，客户端C2接着发送GET段，指定50字节的长度。

[0676] 假设在该点接口单元1250没有到服务器S的可用连接。如果客户端C1已经完成连接或正处于“思考时间”，则目标是重用到服务器S的先前用于客户端C1的相同的连接。接口单元1250使用内容长度参数以确定客户端C1已经接收了所有被请求的数据，而不是等待C1来开始FIN(完成)命令或RST(复位)命令以释放连接。此处，在流程4812B，接口单元1250从客户端C1接收确认，客户端C1事实上已经接收了所有的被请求数据。这向接口单元1250指示即使在客户端C1发出FIN或RST之前由于一些原因被暂停，客户端C1也可完成连接。如流程所示，接口单元1250修改确认号和序列号并向服务器S转发该RESP ACK段。

[0677] 如流程线4820所示，使用与客户端C1使用的相同的连接，在根据以上描述的翻译等式修改序列号和确认号之后，接口单元1250接着从客户端C2向服务器S转发GET段。如流程822所示，服务器S响应以被请求的数据，指定序列号为7000，确认号为2050以及内容长度参数为500。

[0678] 如流程线4824A所示，接口单元1250接收RESP段，翻译序列号和确认号，并向客户端C2转发RESP段。此处，在流程4824B，接口单元1250从客户端C2获得确认，客户端C2事实上已经接收了所有的被请求数据。如流程4824C所示，接口单元1250修改序列号和确认号，并向服务器S转发RESP ACK段。

[0679] 如流程4826A-4826D所示，一旦接口单元1250从客户端C2接收FIN或RST命令，在客户端C2和接口单元1250间的连接接着被关闭或断开。与之相同，如流程4814A-4814D所示，一旦接口单元1250从客户端C1接收FIN或RST命令，在客户端C1和接口单元1250间的连接接着被关闭或断开。值得重点指出的是，然而，接口单元1250保持与服务器S的连接。需要重点

指出的，结合图36描述的事件的序列仅仅用于说明。

[0680] 图36示出了根据实施例在客户端和服务器之间使用内容长度参数以增加连接池化的效率的操作的流程图。接口单元1250保持与多个服务器的连接，并且基于在客户端请求中指明的路径名路由客户端请求到这些服务器。如步骤4902所示，首先，接口单元1250打开与服务器的连接。如步骤4904所示，接着响应于客户端C1的请求，接口单元1250打开到客户端C1的连接并从客户端C1以使用路径名接收请求来获取数据。

[0681] 如步骤4906所示，接口单元1250接着选择寄载由路径名指定内容的服务器。在替代实施例中，接口单元1250查询其它预定策略以选择合适的服务器，例如服务器的负载和服务器的状态。接口单元1250管理和保持服务器和所连接服务群组的数据库。在其它事务中，该数据库中的信息包括当前激活的策略和规则，允许接口单元1250将输入的包定向到正确的服务器。依赖于期望的网络条件和服务，这些策略和规则可被非常快的改变。

[0682] 如步骤4908所示，接口单元1250接着翻译请求并向选择的服务器传输该翻译的请求。如步骤4910所示，接口单元1250从服务器S接收响应。如步骤4912所示，接口单元1250接着翻译响应并向客户端C1传输该翻译的响应。

[0683] 假设为了说明目的，在该点接口单元1250从客户端C2接收请求以获取数据。如步骤4914所示，接口单元1250，响应客户端C2的请求，打开到客户端C2的连接，并从客户端C2接收请求以使用路径名获取数据。如步骤4916所示，接口单元1250接着选择寄载由路径名指定内容的服务器。

[0684] 在步骤4918，接口单元1250确定客户端C2是否已经选择了与客户端C1相同的服务器。如果到步骤4918的结果是否定的，则接口单元1250以足以满足客户端C2的请求的方式继续进行(对该实施例这是不重要的)。在该点，在图36中的流程结束。如步骤中所示，可替代的，如果步骤4918的结果是肯定的，则接口单元1250确定是否有任何打开的到选择的服务器的连接。

[0685] 如果步骤4920的结果是肯定的，则接口单元1250以需要满足客户端C2的请求的方式继续进行(对该实施例这是不重要的)。在该点，图9中的流程结束。如在步骤4922中所示，可替代的，如果步骤4920的结果是否定的，则接口单元1250使用内容长度参数确定客户端C1接收到了客户端C1请求的所有数据。需要重点指出的是，为了确定客户端C1已经完成连接或处于“思考时间”内，接口单元1250不等待客户端C1发送FIN或RST命令。由于与接口单元1250在重新使用用于另一个客户端的连接之前，等待客户端关闭连接相比，接口单元1250可更快速的使用每个连接的事实，允许更有效率的连接池。

[0686] 在步骤4924，接口单元1250接着翻译请求，并使用与客户端C1所用相同的连接向被选择的服务器传输被翻译的请求。如步骤4926所示，接口单元1250从服务器S接收响应。如步骤4928所示，接口单元1250接着翻译该响应并向客户端C2上传输该被翻译的响应。如步骤4930所示，接口单元1250使用内容长度参数来确定客户端C2接收了所有客户端C2请求的数据。

[0687] 接着，接口单元1250在步骤4932关闭或断开于客户端C2的连接。最后，在步骤4934，接口单元1250关闭或者断开与客户端C1的连接，并且在图36中的流程结束。就像以上结合图35所描述的，结合图36的事件序列仅仅用于说明。

[0688] 图37示出了当结合块尺寸域技术时，实施例执行确认号和序列号的翻译的详细流

程图。在图37中的每个流程的标记的形式是T:S,A(L),其中T表示TCP段类型,S为序列号,A为确认号,并且L为块尺寸域。块尺寸域的总值描述了TCP段中的数据字节的数量。

[0689] 为了简单,假设在到客户端C1和客户端C2二者的连接已经建立。客户端C1接着发送GET段,指定49字节的长度,如流程4002所示。接口单元1250确定和服务器S没有空闲打开的连接,并且因此打开与服务器S的连接(在图37中没有示出)。如流程线4004所示,根据以上描述的翻译等式修改序里号和确认号之后,接口单元1250接着从客户端C1向服务器S转发GET段。

[0690] 为了说明目的,假设在响应段中的数据的总体内容数据长度为999。进一步假设,数据将被传输到两个300的数据块和一个399的数据块中。注意,这只是用于说明目的,而非限制。因此,如流程4008A所示,服务器S首先用被请求数据(或消息)的块响应,指定序列号6001,确认号为2000,以及块尺寸域为300。如流程线4006A所示,接口单元1250接收RESP段,翻译序列号和确认号,并向客户端C1转发RESP段。如流程线4006B所示,客户端C1向接口单元1250确认数据的接收。如流程线4008B所示,接口单元1250向服务器S回馈传输该确认。

[0691] 如流程4012A所示,服务器S接着用被请求数据的第二个块响应,指定序列号6301,确认号为2001,以及块尺寸域为300。如流程线4010A所示,接口单元1250接收RESP段,翻译序列号和确认号,并向客户端C1转发RESP段。如流程线4010B所示,客户端C1向接口单元1250确认数据的接收。如流程线4012B所示,接口单元1250将该确认回馈传递到服务器S上。

[0692] 如流程4016A所示,服务器S接着用被请求数据的第三个块响应,指定序列号为6601,确认号为2002,以及块尺寸域为399。如流程线4014A所示,接口单元1250接收RESP段,翻译序列号和确认号,并向客户端C1转发RESP段。如流程线4014B所示,客户端C1向接口单元1250确认数据的接收。如流程线4016B所示,接口单元1250将该确认回馈传递到服务器S上。

[0693] 如流程4020所示,最后,服务器S用零数据的最后一块(由为零的块尺寸域来指示)响应,指出序列号为7000,确认号为2003,以及块尺寸域为0。如流程线4018所示,接口单元1250接收RESP段,翻译序列号和确认号,并向客户端C1转发给RESP段。这向接口单元1250和客户端C1示指示所有被请求的数据已经被传输。

[0694] 如流程4022所示,在该点,客户端C2接着发送GET段,指明长度为50字节。假设在该点接口单元1250没有到服务器S的可用连接。如果客户端C1完成了连接或者处于“思考时间”,则目标是重用到服务器S的相同连接,该连接先前用于客户端C1。接口单元使用等于零的块尺寸域以确定所有被请求的数据已经被客户端C1接收,而不是等待客户端C1开始FIN(完成)命令或RST(复位)命令来释放连接。这向接口单元1250示出了,即使客户端C1在发送FIN或RST命令之前由于一些原因被暂停了,客户端C1被完成该连接。如流程4024所示,接口单元1250修改序列号和确认号并向服务器S转发GET段。

[0695] 为了示出的目的,假定在响应段中的数据总的内容数据长度是500。进一步假定数据将被传输到一个300数据块和一个200的数据块中。注意这只是仅仅用于说明目的,而不是限定。因此,如流程线1028A所示,服务器S首先用被请求数据的块响应,指明序列号为7000,确认号为2050,并且块尺寸域为300。如流程线1026A所示,接口单元1250接收RESP段,翻译序列号和确认号,并且将RESP段转发给客户端C2。如流程线4026B所示,客户端C2向接口单元1250确认接收到数据。如流程线4028B所示,接口单元1250将该确认回馈传递到服务

器S上。

[0696] 如流程4032A所示，服务器S接着用被请求数据的第二个块响应，指定为序列号7300，确认号为2051，以及块尺寸域为200。如流程线4030A所示，接口单元1250接收RESP段，翻译序列号和确认号，并向客户端C2转发RESP段。如流程线4030B所示，客户端C2向接口单元1250确认数据的接收。如流程线4032B所示，接口单元1250将该信息回馈传递到服务器S上。

[0697] 如流程4036所示，最后，服务器S用零数据的最后一块(由块尺寸域为零来指示)响应，指出序列号为7500，确认号为2052，以及块尺寸域为0。如流程线4034所示，接口单元1250接收RESP段，翻译序列号和确认号，并向客户端C2转发给RESP段。这向接口单元1250和客户端C2示出，所有被请求的数据已经被传输。

[0698] 如流程4038所示，一旦接口单元1250从客户端C2接收到FIN或RST命令，则在客户端C2和接口单元1250间的连接接着被关闭或断开。类似的，如流程4040所示，一旦接口单元1250从客户端C1接收到FIN或RST命令，则在客户端C1和接口单元1250间的连接接着被关闭或断开。然而需要重点指出的是，接口单元1250保持与服务器S的连接。同样需要重点指出的是，结合图37描述的事件的序列仅仅用于说明，不用于限制。

[0699] 图38示出了根据实施例使用块尺寸域的操作以增加在客户端和服务器之间的连接的池化的效率的操作的流程图。接口单元1250保持与多个服务器的连接，并且基于在客户端请求中指出的路径名路由客户端请求到这些服务器。首先，如步骤4102所示，接口单元1250打开与服务器的连接。接着，如步骤4104所示，响应与客户端C1的请求，接口单元1250打开与客户端C1的连接并从客户端C1接收请求以使用路径名获取数据。

[0700] 如步骤4106所示，接口单元1250接着选择寄载由路径名指明的内容的服务器。如步骤4108所示，接口单元1250接着翻译该请求并向被选择的服务器传输该被翻译的请求。如步骤4110所示，接口单元1250从服务器S接收响应。如步骤4112所示，接口单元1250接着翻译该响应，并向客户端C1传输该被翻译的响应，直到块尺寸域等于零。

[0701] 假设为了示出的目的，在该点，接口单元1250接收来自客户端C2的请求以打开连接。如步骤4114所示，接口单元1250，响应于客户端C2的请求，打开到客户端C2的连接并从客户端C2接收请求，以便使用路径名获取数据。如步骤4116所示，接口单元1250接着选择寄载由路径名指定内容的服务器。

[0702] 在步骤4118，接口单元1250确定客户端C2是否已经选择了与客户端C1相同的服务器。如果步骤4118的结果是否定的，则接口单元1250以满足客户端C2的请求的方式继续进行。在该点，图38中的流程图结束。可替代的，如步骤4120所示，如果步骤4118的结果是肯定的，则接口单元1250确定是否存在到被选择服务器的任何打开的连接。

[0703] 如果步骤1120的结果是肯定的，则接口单元1250以满足客户端C2的请求的方式继续进行。在该点，图38中的流程图结束。作为选择的，如步骤4120所示，如果步骤4118的结果是否定的，则接口单元1250使用在步骤4112中块尺寸域等于零的事实来确认客户端C1接收了客户端C1请求的所有消息数据。值得重点指出的是，接口单元1250不等待客户端C1发出FIN或RST命令，用于确定客户端C2已经完成连接或处于“思考时间”。

[0704] 在步骤4122，接口单元1250接着翻译请求，并使用客户端C1所使用的相同的连接来传输该被传输的请求到被选择的服务器。如步骤4124所示，接口单元1250从服务器S接收

响应。如在步骤4126中所示，直到块尺寸域等于零，接口单元1250接着翻译该响应，并传输该被翻译的响应到客户端C2上。接口单元1250使用块尺寸域来确定客户端C2接收了客户端C2请求的所有消息数据。

[0705] 接着，在步骤4128中，接口单元1250关闭或断开与客户端C2的连接。最后，在步骤4130，接口单元1250关闭或断开与客户端C1的连接，并且在图38中的流程图结束。如上述结合图37所述的，结合图38描述的事件的序列仅仅用于示出的目的，并不用于限制。

[0706] 前述的实施例被特定描述为在接口单元内实施，例如接口单元1250，该接口单元连接到在群组中的服务器，用于从服务器上卸载连接处理开销。然而，它们也可被应用到其他类型的设备中，这些设备在客户端和服务器间的网络连接路径上。当网络话务量通过这样的设备时，它们都有机会卸载连接处理。这样的设备的一些实施例是：

[0707] —负载平衡器，在服务器群组中的一组服务器(本地的或地理上分布的)间分布客户端网络连接。

[0708] —带宽管理器，监控网络话务量和测量包流。

[0709] —防火墙，监控包并使得仅仅授权的包通过。

[0710] —路由器和交换器，也位于网络话务量的路径上。行业趋向可为在这些设备内集成附加的功能(例如负载平衡，带宽管理和防火墙功能)。

[0711] 实施例也可在计算机系统内实施，该计算机系统为网络间接的末端。在该实例中，可使用附加卡卸载在计算机系统内主要的处理元件。

[0712] 3.集成高速缓存

[0713] 图39示出了根据实施例发生在设备内的事件序列的流程图5300，该设备提供集成高速缓存功能。然而，该实施例不被流程图5300提供的描述所限制。更正确的，此处提供的教导对于相关领域的技术人员是显而易见的，以及其他功能流程在本实施例的范围和精神之内。这些其他的功能流程可包括在集成高速缓存上的不同的处理、不同序列和其他变化。

[0714] 路程图5300的方法可在一个或多个装置中实施，这些装置可通信的耦合到数据通信网络。例如，流程图5300的方法可在设备中实施，例如上述关于图1A描述的设备1250，包括上述关于图27描述的软件结构3200。将继续参考这些示例性实施例描述流程图5300的方法。

[0715] 如在图39中示出的，流程图5300的方法在步骤5302开始，其中设备1250从一个客户端10接收加密包。在实施例中，设备1250被配置充当用于服务器30的代理SSL端点，解密从客户端10接收的加密包，并接着根据需要发送到那里用于进一步的处理，并基于在加密包中的地址信息最终发送到合适的资源。合适的资源可为，例如，任何服务器30或由设备1250管理的高速缓存。在步骤5304，设备1250在包上执行解密处理。

[0716] 在步骤5306，设备1250验证和/或授权从其接收加密包的客户端，该设备根据实施例被配置以实施AAA策略用于访问控制。

[0717] 在步骤5308，设备1250在解密的包上实施包处理，以减少由设备网络协议产生的连接开销的处理需求，该设备1250根据实施例被配置以执行特定类型的包处理。

[0718] 在步骤5310，设备1250解压缩与包相关联的请求，该设备根据实施例被配置以压缩和解压缩内容。在实施例中，请求包括web对象请求。

[0719] 在步骤5312，设备1250接着可以激活高速缓存功能，该设备接收用于对象的清除

和/或授权的和/或解压缩和/或打包处理请求。由于在步骤5302,5304,5306,5308,5310中描述的先前的处理,高速缓存管理逻辑可基于清除和/或授权的和/或解压缩和/或打包处理请求做出是否对象已经被高速缓存或者可高速缓存的决定,并且因此比传统的高速缓存可以处理更广泛的请求队列,并比传统的方法更有效率的实施高速缓存。另外,由于高速缓存管理逻辑连同其他处理工作在内核空间内运行,它涉及到作为数据结构的相关对象,该数据结构带有涉及到这样的数据结构的同等状态,作为每个其他应用,并因此可在十分有效率的方式获得集成。

[0720] 如步骤5314所示,如果对象不是已经存在于高速缓存存储器中,设备1250发送请求到一个或多个服务器30上。然而,在请求被发送之前,可发生多个附加处理步骤。

[0721] 例如,在步骤5316,设备1250可选的执行连接处理以保证有效率的向服务器传输请求,并且在步骤5318,设备1250可选的做出负载平衡决定以保证请求被发送到最合适的服务器。同样的,在实施例中,在请求通过后端加密处理被发送到服务器之前,该请求被加密,因此提供端到端网络安全。在步骤5320,请求被发送到服务器。

[0722] 在步骤5322,设备1250接收服务器30之一返回的响应。如果后端加密如上所描述的被支持,设备1250解密来自服务器的响应。

[0723] 在步骤5324,设备1250压缩与来自服务器的响应相关联的对象。在实施例中,该对象包括web对象。

[0724] 在步骤5326,设备1250中的高速缓存管理逻辑以压缩形式在高速缓存中存储对象。由于处理能力,高速缓存管理逻辑可以以该方式存储被压缩的对象——旦对象被存储在高速缓存中,可在没有执行如上描述的步骤5316,5318,5320,5322,5324,5326时,从高速缓存服务用于对象的未来的客户端请求。这通过在流程图5300中的直接连接从判断步骤5314到步骤5328的线指示。

[0725] 在步骤5328,在已经从服务器接收对象以后或者从高速缓存获取对象之后,设备1250在连接上执行包处理,以便更有效率的服务初始客户端的请求。在步骤5330,该响应对象接着被重加密并传送回客户端。

[0726] 以上描述的每个处理步骤发生在设备1250的内核/OS级。通过在内核/OS空间中的其他处理步骤中间以及与其结合而实施高速缓存,实施例可以带来附加的功能并提高速缓存的性能。

[0727] 根据实施例这样的集成允许高速缓存实施以执行附加功能,这些附加功能传统上超出高速缓存的操作功能。例如,实施例允许高速缓存与加密和/或被压缩的对象一起工作。

[0728] 通过实施例可获得附加功能的另一个实施例包括端对端被加密的HTTPS话务量的高速缓存。典型的,高速缓存仅仅存储来自服务器的未加密的HTTP响应。在一些情况中,某些高速缓存可支持从高速缓存到客户端SSL加密HTTPS的传送,但是在任何情况中,传统的高速缓存不能够高速缓存已经被服务器加密的响应,因此不能支持端到端(即服务器到客户端)的加密。典型的,当响应被服务器以HTTPS的形式加密时,高速缓存不能够解密这样的响应,并且因此不能够在它的高速缓存存储器中存储该响应。因为这个原因,传统的高速缓存不能在面对实施例中的端对端被加密的话务量中提供任何有益之处,该集成高速缓存设备用作用于SSL所加密的HTTPS话务量的两路终端端点。

[0729] 例如，在实施例中，集成高速缓存设备作为在服务器和设备之间的被加密的话务量的终端端点，和在设备和客户端之间的被加密的话务量的终端端点。在此种方式中，设备可以解密和高速缓存从服务器接收的SSL－加密响应，并且当向客户端服务这样的响应时，重加密这样的响应，并向请求客户端安全地传输该响应，因此启用端对端加密并因此提高了对更多的web话务量的高速缓存的适应能力。

[0730] 在实施例中，设备也可以在SSL虚拟专用网络(SSL VPN)中作为端点。更具体地，设备可以在专用数据通信网络中作为代理SSL端点用于任何资源，解密从客户端接收的被加密的包并接着基于在加密的包中的地址信息，向合适的目的服务器源发送。在客户端和网关之间建立的数据通信会话可被加密，网关用作本申请前述的附图中所描述的作为加密的端点。就像描述的，客户端可使用安全套接字(SSL)、IPSec或者一些其他的加密方法，以建立加密数据通信会话，通过该会话在客户端上的拦截机制直接将话务量导到网关，而使得客户端浏览器认为其直接与目的服务器或目的网络通信。在这样的实施例中，可在网关终止加密的数据通信会话，如此处所描述的也包括集成的高速缓存。在该方式中，高速缓存功能可被集成到SSL VPN功能中。

[0731] 网关也可执行任何对于请求可应用的AAA策略，并且因此网关将把高速缓存的对象只服务到合适的所验证的客户端，也仅允许用于授权访问特定高速缓存对象的用户的请求。这可能是因为高速缓存以这样的方式被集成：即在高速缓存见到任何特定请求之前，网关的访问控制策略被增强。因此，在不需要高速缓存本身执行验证和授权时，高速缓存的对象获得访问控制的利益。通过带有其他功能的高速缓存的集成，高速缓存本身变得更加有效率并且在控制通过现今的网络的数据的变化上更有效。实施例通过将高速缓存的功能益处引入到web话务量的更宽阔的队列，也可提高整个网络性能的效率。

[0732] 上述与实施例结合描述的集成模式的一些其他唯一的结果如下。一个结果是高速缓存预压缩的数据并将其传送到知道压缩的客户端的能力。另一个结果是高速缓存访问所控制的数据的能力。又一个结果是与外部高速缓存协同提供高速缓存的可量测性。由于高速缓存在网关与重定向和流量管理能力集成，所以外部高速缓存可被部署来提供高速缓存的第二层，从而显著扩展高速缓存的能力(以及益处)。通过实施例，高速缓存模块本身不用必须明确的执行高速缓存重定向策略，而生成该能力。

[0733] 在性能方面，如上所述通过集成高速缓存，高速缓存的处理器可从执行多个任务处理连接中解放出来，该高速缓存，在网络上作为节点，通常被要求执行，并且因此可以以它们的最高性能级别执行自身的高速缓存功能。事实上，通过启用压缩数据的高速缓存，高速缓存可以更有效率的运行，并允许用户实现更高的性能。

[0734] 就像在该申请中之前提及的，作为高速缓存与其他网络服务和技术结合的方式的结果产生效率，上述技术包括负载平衡技术、加密、AAA、压缩和其他类型的加速以及包处理。结果，避免通过集成的传统模式而产生的处理复制和其他无效率。通过不需要的拷贝和上下文切换引起的这种无效率，由于通过设备接收的每个对象必须拷贝到消息并接着在被相关应用处理之前进入处理器存储器而产生。该请求接着必需拷贝回引入附加的存储器拷贝的高速缓存处理的对象或包级别。相反，实施例在OS或内核级别实施集成，因此启用高速缓存在对象上操作作为数据结构，其中高速缓存具有和涉及到并处理这样的数据结构的其他应用和/或处理相等的状态，并且当所有的处理与相同的数据结构一起工作时，此处对这

样的附加的存储器拷贝的需要被消除。结果是更有效率的集成。

[0735] a.在数据通信网络中具有主动验证的高速缓存

[0736] 由于web对象能够随着时间改变，每个潜在的可高速缓存的对象被认为具有有用的寿命或“新鲜度(freshness)”，新鲜度的概念指这样的事实：即初始产生内容的应用服务器也确定可被高速缓存服务的该对象的时间周期，该高速缓存可以存储这样的对象－高速缓存必需能够确定存储在其存储器中的对象的副本是否仍然是“新鲜”，或者高速缓存是否需要从源服务器获取对象的新副本。实施例实现保证对象的新鲜度的新方法。许多传统的高速缓存的实现试图通过从在预定表上的源取来内容而保持高速缓存的内容的新鲜。基于一个或两个下述方法，从源取来内容发生在由高速缓存管理器建立时：或者在(i)正常指定的间隔或者(ii)当内容将终止时

[0737] 存在与上述通常应用的方法相关联的典型的两个问题。第一，由于服务器需要向请求刷新的高速缓存提供内容(无论这样的更新发生在指定的时间间隔或者内容将终止时)，不考虑这样的内容是否最终被服务到客户端，则不必需的处理负载强加在源服务器上。第二，由于高速缓存需要跟踪必须被刷新的元件以及它们必须被刷新的时间，高速缓存基于产生的额外处理导致附加的处理器负载。

[0738] 根据实施例的高速缓存使用新的预取得(pre-fetching)方式来解决上述问题。内容的预取得不依照之前定义的方式或刚刚前述的内容的终止前而执行。代替的，实施例仅仅当下述两个条件都满足时才执行预取得：(1)客户端已经做出对指定内容的请求，并且(2)该内容‘将终止’。

[0739] 该方法处理上述描述了两个问题。仅仅在这样的内容被主动访问时，主动的重新生效更可能产生对于来自源服务器的内容的更新请求。这最小化了源服务器上的‘不必需’负载的数量-如上讨论的，此处高速缓存请求最终没有传送到客户端(或者依赖于高速缓存的敏感性，只是很少的获得传输)的对象的更新，该高速缓存正无效率地使用自身资源和源服务器的资源。实施例避免无效率的使用高速缓存和服务器资源，该资源仅仅被主动访问的内容请求。由于相同的原因，该方法也减少了用于预取得的带宽，因此可比传统的方法更有效率的使用网络资源。

[0740] 进一步的，实施例使用包括在高速缓存对象自身中的终止信息以确定是否请求来自源服务器的对象的更新。这样的终止信息通常包括在相关对象的头中。该实施例不像许多传统的方式，因此避免了为了获取而关注任一附加信息的无效率，传统的方式需要高速缓存保持跟踪进度表用于更新的表。使用‘基于需要的’预取得技术也增强了预取得的内在益处。这个技术减少了频繁访问的对象的高速缓存未命中的数量，是由于对象恰好在终止之前很可能经历主动的重新有效。该技术也阻止了到源服务器的话务量蜂涌，当大量要求的大规模响应过期时发生话务量蜂涌。在传统的方法中，由于高速缓存内容已经终止，对于这样内容的所有请求不能得到高速缓存并且被发送回源服务器。作为对比，在一个实施例中，高速缓存存储器的内容通常仅仅在终止之前被刷新，并且因此当高速缓存正被更新时而发生高速缓存未命中的情况将更少发生。

[0741] 在一个实施例中，在内容被确定将终止的情况下，在终止之前，通过调整持续期间的长度，能够控制预取得的主动性，以及需要由相关对象的高速缓存触发刷新的客户端请求的数量。

[0742] b.使用“负单元”优化大的不可高速缓存的响应的处理

[0743] 根据实施例，高速缓认出可并且不存储大于指定尺寸的对象，用于提高对象成功率(hit radio)。高速缓存典型的有专用于存储高速缓存的对象的受限的存储器空间，并且因此超过分配的存储器空间的特定响应作为不可高速缓存和不被高速缓存所存储而最终被拒绝。对于传统的高速缓存，高速缓存试图在其高速缓存存储器中存储大的响应，并且高速缓存一旦识别到该响应超过预定的最大尺寸，则仅异常中断响应的存储。每次在每种情况中高速缓存从服务器接收用于这样的响应的请求时，传统的高速缓存将重复的试着高速缓存该大的响应，该高速缓存将需要确定超过存储器空间的对象是不可高速缓存的，因此，这是明显无效率的方法。

[0744] 根据实施例，高速缓存使用最优化以避免在存储这样的响应中花费精力。任何时候高速缓存探测到由于响应的尺寸该响应变得不可高速缓存时，该高速缓存在称为“负单元”的数据结构中存储关于相应请求的符号。该符号指示请求是不可高速缓存的。在预定日期中，当客户端请求相同的对象时，该请求与关于存储在数据结构中的第一请求的符号相匹配。基于匹配，高速缓存将不试图存储该响应，代替的，该请求将完全旁路高速缓存。

[0745] 不存在需要指定负单元保存在高速缓存中的持续时间的用户配置。事实上，用户甚至不知道该特定机制被实行。在实施例中，高速缓存使用正常的终止信息，将应用该信息高速缓存大的响应，来高速缓存关于该响应的“负信息”。

[0746] 4.客户端侧加速

[0747] 在一个实施例中，客户端侧加速程序可执行一个或多个加速技术以加速、增强或以其他方法改进客户端与服务器的通信和/或对服务器的访问，例如访问服务器提供的应用。现在看图40A，描述有加速程序6120的客户端6205。总的来说，客户端6205在计算设备6100上操作，该计算设备6100拥有带有内核模式6202以及用户模式6203的操作系统，以及带有一个或多个层6210a-6210b的网络堆栈6210。客户端6205可包括先前讨论的任何和所有的客户端10。虽然只示出了一个客户端6205，但任何数量的客户端10可包括客户端6205。客户端6205可以已经安装和/或执行一个或多个应用6220a-6220n。在一些实施例中，一个或多个应用6220a-6220n可通过网络堆栈6210与网络通信。应用6220N之一也可包括第一程序6222，例如，可用于一些实施例中来安装和/或执行加速程序6120的程序。

[0748] 客户端6205的网络堆栈6210可包括任何类型和形式的软件、或硬件或其组合，用于提供与网络的连接和通信。在一个实施例中，网络堆栈6210包括用于网络协议组的软件实现。网络堆栈6210可包括一个或多个网络层，例如为本领域技术人员所公认和了解的开放式系统互联(OSI)通信模型的任何网络层。同样的，网络堆栈6210可包括协议的类型和形式，这些协议用于OSI模型的任何以下层：1)物理链路层，2)数据链路层，3)网络层，4)传输层，5)会话层，6)表示层，以及7)应用层。在一个实施例中，网络堆栈310可包括在因特网协议(IP)的网络层协议上的传输控制协议(TCP)，通常称为TCP/IP。在一些实施例中，可在Ethernet协议上实施TCP/IP协议，Ethernet协议可包括IEEE广域网(WAN)或局域网(LAN)协议的任何族，例如被IEEE802.3覆盖的这些协议。在一些实施例中，网络堆栈6210包括任何类型和形式的无线协议，例如IEEE 802.11和/或移动因特网协议。

[0749] 考虑基于TCP/IP的网络，可使用任何基于TCP/IP的协议，包括消息应用编程接口(MAPI)(email)、文件传输协议(FTP)、超文本传输协议(HTTP)、通用因特网文件系统(CIFS)

协议(文件传输)、独立计算框架(ICA)协议、远程桌面协议(RDP)、无线应用协议(WAP)、移动IP协议,以及IP语音(VoIP)协议。在另一个实施例中,网络堆栈210包括任何类型和形式的传输控制协议,诸如修改的传输控制协议,例如事务TCP(T/TCP),带有选择确认的TCP(TCP-SACK),带有大窗口的TCP(TCP-LW),拥塞预测协议,例如TCP-Vegas协议,以及TCP电子欺骗协议。在其他实施例中,任何类型和形式的用户数据报协议(UDP),例如IP上UDP,可被网络堆栈6210使用,诸如用于语音通信或实时数据通信。

[0750] 另外,网络堆栈6210可包括支持一个或多个层的一个或多个网络驱动器,例如TCP驱动器或网络层驱动器。网络层驱动器可被包括作为计算设备100的操作系统的一部分或者计算设备6100的任何网络接口卡或其它网络访问部件的一部分。在一些实施例中,网络堆栈6210的任何网络驱动器可被定制、修改或改调整以提供网络堆栈6210的定制或修改部分,支持此处描述的任何技术。在其他实施例中,设计并构建加速程序6120与网络堆栈6210协同操作或工作,上述网路堆栈6120由客户端205的操作系统安装或以其他方式提供。

[0751] 网络堆栈6210包括任何类型和形式的接口,用于接收、获得、提供或以其他方式访问涉及客户端6205的网络通信的任何信息和数据。在一个实施例中,与网络堆栈6210的接口包括应用编程接口(API)。接口也可包括任何函数调用、钩子或过滤机制,事件或回调机制、或任何类型的接口技术。网络堆栈6210通过接口可接收或提供任何类型和形式的数据结构,例如对象,涉及到网络堆栈6210的功能或操作。例如,数据结构可包括涉及到网络包或一个或多个网络包的信息和数据。在一些实施例中,数据结构包括在网络堆栈6210的协议层处理的网络包的一部分,例如传输层的网络包。在一些实施例中,数据结构6225包括内核级别数据结构,而在其他实施例中,数据结构6225包括用户模式数据结构。内核级别数据结构可包括在内核模式6202中操作的网络堆栈6210的部分获得或涉及的数据结构,或者运行在内核模式6202中的网络驱动器或其他软件,或者通过在操作系统的内核模式中运行或操作的服务、处理、任务、线程或其他可执行指令获得或接收的任何数据结构。

[0752] 此外,网络堆栈6210的一些部分可在内核模式6202中执行或操作,例如,数据链路或网络层,而其他部分在用户模式6203中执行或操作,例如网络堆栈6210的应用层。例如,网络堆栈的第一部分6210a可为应用6220a-6220n提供对网络堆栈6210的用户模式的访问,而网络堆栈的第二部分6210a提供对网络的访问。在一些实施例中,网络堆栈的第一部分6210a可包括网络堆栈6210的一个或多个更上层,例如任何层5-7。在其他实施例中,网络堆栈6210的第二部分6210b包括一个或多个较低的层,例如任何层1-4。网络堆栈6210第一部分6210a和第二部分6210b的每个可包括网络堆栈6210的任何部分,在任意一个或多个网络层,在用户模式6203、内核模式6202,或其组合,或在网络层的任何部分或者指向网络网络层的接口,或指向用户模式6203和内核模式6203的任何部分或者接口。

[0753] 本发明的加速程序6120可以包括软件、硬件或软件和硬件的任何组合。在一些实施例中,加速程序6120包括任何类型和形式的可执行指令,该可执行指令被构造并设计以执行或者提供如此处所描述的功能和操作。在一些实施例中,加速程序6120包括任何类型和形式的应用、程序、服务、处理、任务或线程。在一个实施例中,加速程序6120包括驱动器,例如被设计和构造与网络堆栈6210协同工作和连接的网络驱动器。加速程序6120的可执行指令的逻辑、功能和/或操作可执行一个或多个以下的加速技术:1)多协议压缩6238,2)传输控制协议池6224,3)传输控制协议多路复用6226,4)传输控制协议缓冲6228,以及5)通过

高速缓存管理器的高速缓冲6232，下面将进一步详细的描述这些技术。另外，加速程序6120可执行由客户端6205接收和/或传输的任何通信的加密6234和/或解密。在一些实施例中，加速程序6120也可在客户端6205和另一个计算设备6100，例如服务器30之间执行隧穿。在其他实施例中，加速程序6120向服务器30提供虚拟专用网络连接。

[0754] 在一些实施例中，加速程序6120在网络堆栈6210的一个或多个层操作，例如传输层。在一个实施例中，加速程序6120包括过滤器驱动器、钩子机制，或任何类型和形式的合适的网络驱动器接口，连接网络堆栈的传输层，例如通过传输驱动器接口(TDI)。在一些实施例中，加速程序6120连接到第一协议层，例如传输层和另一协议层，例如在传输协议层上的任何层，例如，应用协议层。在一个实施例中，加速程序6120可包括遵守网络驱动器接口规范(NDIS)的驱动器，或NDIS驱动器。在另一个实施例中，加速程序6120可包括最小过滤器或者迷你端口驱动器。在一个实施例中，加速程序6120，或其部分，在内核模式6202中操作。在另一个实施例中，加速程序6120，或其部分，在用户模式6203中操作。在一些实施例中，加速程序6120的部分在内核模式6202操作，而加速程序6120的另一部分在用户模式6203操作。在其他实施例中，加速程序6120在用户模式6203操作，但连接到内核模式驱动器，操作系统的处理、服务、任务或部分，例如获得内核级别数据结构6225。在进一步的实施例中，加速程序6120为用户模式应用或程序，例如应用6220a-6220n。

[0755] 加速程序6120可以对网络堆栈6210的任何其他协议层透明的方式操作协议层或与协议层连接。例如，在一个实施例中，对于传输层之下的任何协议层透明的，例如网络层和传输层之上的任何协议层例如会话、表示或应用层协议，加速程序6120操作网络堆栈6210的传输层或与之连接。这允许网络堆栈6210的其他协议层按照需要操作并且无需由于使用加速程序6120的修改。同样的，加速程序6120可以与传输层连接，以加速通过由传输层实施的任何协议提供的任何通信，例如TCP/IP上的应用层协议。

[0756] 进一步的，以对任何应用6220a-6220n、客户端6205的用户以及与客户端6205通信的例如服务器的任何其他计算设备透明的方式，加速程序6120可以操作网络堆栈6210或与网络堆栈6210连接。以诸如加速程序6120在没有修改应用6220a-6220n时可以加速应用6220a-6220n的任意方式，加速程序6120可在客户端6205上安装和/或执行。在一些实施例中，客户端6205的用户或与客户端6205通信的计算设备没有意识到计算程序6120的存在、执行或操作。同样的，在一些实施例中，加速程序6120的安装、执行和/或操作对应用6220a-6220n、客户端6205的用户、例如服务器的另一个计算设备或被加速程序6120连接的协议层之上和/或之下的任何协议层透明。

[0757] 在一些实施例中，加速程序6120以集成方式或样式执行一个或多个加速技术6224、6226、6228、6232。在一个实施例中，加速程序6128包括任何类型和形式的机制用于在网络堆栈6210的传输协议层拦截、钩住、过滤或接收通信。通过在传输层拦截客户端6205的网络包，并且在传输层通过例如内核级别数据结构6225的数据结构连接到网络堆栈6210，加速程序120可在网络包上执行涉及传输层的加速技术，例如传输控制协议(TCP)缓冲、TCP池化以及TCP多路复用。另外，加速程序6210可在任何协议或多协议中执行压缩6225，上述协议作为传输层协议的网络包的有效载荷被携带。

[0758] 在一个实施例中，加速程序6120使用内核级别数据结构6225提供对一个或多个网络包的任意部分的访问，例如，网络包包括来自于客户端6205的请求或者来自服务器的响

应。在一个实施例中，可通过加速程序6120使用内核级别数据结构，以执行期望的加速技术。在一个实施例中，当使用内核级别数据结构6225时，加速程序6120运行在内核模式6202中，而在另一个实施例中，当使用内核级别数据结构6225时，加速程序6120运行在用户模式6203中。在一些实施例中，内核级别数据结构可被拷贝或传递到第二内核级别数据结构，或任何期望的用户级别数据结构。虽然加速程序6120在图40A中一般的描述为操作在用户模式6203中的第一部分和操作在内核级别6202中的第二部分，但在一些实施例中，加速程序6120的任何部分可以运行在用户模式6203或内核模式6202中。在一些实施例中，加速程序6120可只在用户模式6203中操作，而在其他实施例中，加速程序6120可只在内核模式6202中操作。

[0759] 另外，通过在传输层拦截网络堆栈6210或通过内核级别数据结构6225获得对网络包的访问，加速程序6120可在单独的接口点或在单独的执行点或执行加速程序6120的任何可执行指令的时间，执行或应用多个加速技术。例如，在一个实施例中，在加速程序6120的功能或指令集中，可执行多个加速技术，例如通过调用一组可执行指令，该指令被构造和设计以执行加速技术。在一些实施例中，加速程序6120在一个接口点、执行地点或者一组指令中调用一个或多个应用编程接口(API)到任何程序、服务、处理、任务、线程或可执行指令，其被设计和构造以提供1)多协议压缩6238，2)传输控制协议池6224，3)传输控制协议多路复用6226，4)传输控制协议缓存6228，以及5)通过高速缓存管理器6232的高速缓存，以及在一些实施例中的加密6234。

[0760] 通过在加速程序6120的可执行指令中的一个地点或位置或者在网络堆栈6210的一个例如传输层的协议层执行多个加速技术，可更有效率和有效的执行这些加速技术的集成。在一个方面，可减少在处理之间的上下文切换的数量，以及减少使用的数据结构或者存储器必需的或者在其他方式中使用的数据结构的副本的数量。另外，在任何加速技术之间的同步和通信可被更有效率的执行，例如以更紧密耦合的方式，在加速程序6120的可执行指令集中。同样的，关于要执行的加速技术顺序的任何逻辑、规则、功能或操作，以及可在技术间共享或传输的数据和信息可被更有效率的执行。加速技术6120可在传输层拦截TCP包，通过内核级别的数据结构6225获得TCP包的有效载荷，以及之后以期望的顺序执行期望的加速技术。例如，网络包可首先被压缩，接着被高速缓存。在另一个实施例中，通过被缓存、被池化和的/或到服务器的TCP连接的多路复用，可通信被压缩的高速缓存数据。

[0761] 在一些实施例中，仍旧看图40A，可使用第一程序6222自动、静默的、透明的或以其他方式安装和/或执行加速程序6120。在一个实施例中，第一程序6222包括插件部件，例如ActiveX控件或Java控件或脚本，其由应用6220a-6220n载入并执行。例如，第一程序包括被web浏览器应用6220运行和载入的ActiveX控件，例如在存储器空间或应用6220的上下文中。在另一个实施例中，第一程序6222包括可执行指令组，该可执行指令组被例如浏览器的应用6220a-6220n载入并执行。在一个实施例中，第一程序6222包括被设计和构造的程序以安装加速程序6120。在一些实施例中，第一程序6222通过网络从另一个计算设备获得、下载、或接收加速程序6120。在另一个实施例中，第一程序6222是用于客户端6205的操作系统上安装程序或者安装程序的插件或者播放管理器，例如网络驱动器。

[0762] 在其他实施例中，第一程序6222可包括此处在B部分描述的任何和所有功能。在一个实施例中，第一程序6222可包括收集代理404。在另一个实施例中，第一程序可包括用于

安装收集代理的程序。在另一个实施例中，第一程序6222可包括计算环境15。在一个实施例中，第一程序6222包括用于安装诸如执行环境或者虚拟执行环境的计算机环境。在一个实施例中，第一程序6222可包括如前所述的应用流客户端442。在另一个实施例中，第一程序6222可包括将在客户端10上执行的应用。

[0763] 在其他实施例中，第一程序6222可包括加速程序6120的功能、操作和逻辑的部分，以推动或执行此处描述的加速程序6120的任何功能、操作和逻辑，例如任何加速技术。在一些实施例中，第一程序6222用于建立连接，例如传输层连接，或与设备或服务器的通信会话，例如加密套接字协议层(SSL)通信会话。在一个实施例中，第一程序6222用于建立或推动虚拟专用网络连接和通信会话。

[0764] 加速程序6120或客户端6205的高速缓存管理器6232如图40A所描述的可包括软件、硬件和软件和硬件的任何组合，以提供对任何形式和类型的内容的高速缓存访问、控制和管理，例如由服务器30服务的对象或动态产生对象。被高速缓存管理器6232处理和存储的数据、对象或内容可包括任何格式的数据，例如标记语言或通过任何协议的通信。在一些实施例中，高速缓存管理器6232复制存储在其他地方的原始数据或先前计算、产生或传输的数据，其中相对于读高速缓冲存储器元件，需要更长的访问时间以取得、计算或以其他方式得到。一旦数据被存储在高速缓冲存储元件中，通过访问高速缓存的副本而不是重新获得或重新计算原始数据而可做出未来的使用，因此而减少了访问时间。在一些实施例中，高速缓冲存储器元件可包括客户端6205的存储器中的数据对象。在其他实施例中，高速缓冲存储器元件可包括比客户端6205以其他方式使用的存储器有更快访问时间的存储器。在另一个实施例中，高速缓冲存储器元件可包括客户端6205的任何类型和形式的存储元件，例如硬盘的一部分。在又一个实施例中，高速缓存管理器6232可使用存储器、或用于高速缓存数据、对象和其他内容的处理单元的任何部分和组合。

[0765] 此外，高速缓存管理器6232可包括任何逻辑、功能、规则或操作以执行此处描述的技术的任意实施例。例如，基于无效时间周期的终止，或者从客户端6205a－6205n或者服务器接收到命令，高速缓存管理器6232包括用于无效对象的逻辑或功能。在一些实施例中，高速缓存管理器6232可作为程序、服务、处理或任务操作执行在内核空间6202中，并且在其他实施例中，在用户空间6203中。在一个实施例中，高速缓存管理器6232的第一部分在用户空间6203中执行，而第二部分在内核空间6202中执行。在一些实施例中，高速缓存管理器6232可包括任何类型的通用处理器(GPP)，或任何其他类型的集成电路，例如现场可编程门阵列(FPGA)，可编程逻辑设备(PLD)，或者专用集成电路(ASIC)。

[0766] 加速程序6120或客户端6205的加密引擎6234包括任何逻辑、商业规则、功能或操作，用于控制任何安全相关协议的处理，例如SSL或TLS，或任何与其相关的功能。例如，加密引擎6234加密或解密与客户端6205通信的网络包，或其任何部分。加密引擎6234也可代表客户端6205建立或设立SSL或TLS连接。同样的，加密引擎6234提供SSL处理的卸载和加速。在一个实施例中，加密引擎6234使用隧道协议提供客户端6205和例如服务器的另一个计算设备之间的虚拟专用网络。

[0767] 现在仍看图40A，加速程序6120或客户端6205的多协议压缩引擎6238包括任何逻辑、商业规则、功能或操作，用于压缩网络包的一个或多个协议，例如被客户端6205的网络堆栈6210使用的任何协议。例如，多协议压缩6238可包括压缩和解压缩效用，包括Gzip压缩

和解压缩，微分压缩和解压缩，或用于压缩和解压缩通过网络传输的数据的任何其他私人拥有或公共可用的效用。在一个实施例中，多协议压缩引擎6238双向压缩在客户端6205和例如服务器的另一个计算设备之间的任何基于TCP/IP的协议，包括消息应用编程接口(MAPI)(email)、文件传输协议(FTP)，、超文本传输协议(HTTP)、通用因特网文件系统(CIFS)协议(文件传输)、独立计算结构(ICA)协议、远程桌面协议(RDP)、无线应用协议(WAP)、移动IP协议，以及IP上的语音(Voice Over Protocol)(VoIP)协议。在其它实施例中，多协议压缩引擎238提供基于超文本标记语言(HTML)的协议的压缩，并且在一些实施例中，提供任何标记语言的压缩，例如可扩展标记语言(XML)。同样的，多协议压缩引擎3238通过桌面客户端，例如MicosoftOutlook和非web瘦客户端，例如，由企业应用像Oracle，SAP和Siebel，甚至例如便携式计算机的移动客户端启动的任何客户端，来加速用户访问应用的性能。

[0768] 如下述将进一步详细描述的，加速程序6120也执行缓冲、池化和多路复用的传输协议层加速技术。同样，加速程序6120包括任意类型和形式的可执行指令，该指令具有来执行此处描述的任何这些技术的逻辑、规则、功能和操作。加速程序120在网络堆栈210的传输层拦截、控制和管理应用6220a－6220n通过网络堆栈6210做出的任何传输层应用编程接口(API)调用。加速程序6120以透明的方式响应客户端6205的任何请求，以便客户端6205像希望的那样从网络堆栈6210的传输协议层接收响应。例如，在一个实施例中，加速程序6120拦截在客户端6205的网络堆栈6210中的请求，以与诸如服务器的另一个计算设备建立传输层连接，并可使用由加速程序6120建立的一个或多个传输层连接的池来响应请求。在另一个实施例中，加速程序6120通过建立的、被第二应用6220b使用的传输层连接多路传输来自第一应用6220a的请求。

[0769] 在一些实施例中，加速程序6120包括在网络传输之前在客户端6205缓冲或保持客户端6205通信的机制。例如，从网络，如从服务器，接收的通信被客户端耗用的速度少于在网络上被客户端6205传输的通信产生的速度。同样的，客户端6205可以更大的速度发送更多的请求到服务器30，该速度大于客户端6205可消耗和处理的来自这样的请求的响应。加速程序6120可拦截通信，并且确定是否消耗的速度和/或客户端6205的产生的速度低于预定阈值，例如由用户、客户端6205或另一个计算设备配置的阈值。如果确定的速度低于期望的阈值，加速程序6120向客户端的存储器元件中存储拦截的通信，直到客户端6205的性能增加消耗和/或产生的速度，使其等于或高于预定或期望的阈值。在该点，加速程序6120在网络上传输客户端的通信。同样的，基于客户端6205的通信的消耗和/或产生性能，提供客户端侧机制以减缓客户端6205的通信。

[0770] 在图40A中描述的应用6220a－6220n可为任何类型和/或形式的应用，例如任何类型和/或形式的web浏览器、基于web的客户端、客户端－服务器应用、瘦客户端计算客户端、ActiveX控件或者Java程序、或其他类型和/或形式的可执行指令，该可执行指令可以在客户端6205上执行或通过网络6204通信。应用6220a－6220n可使用任何类型的协议，并且它可为，例如，HTTP客户端、FTP客户端、Oscar客户端或Telnet客户端。在一些实施例中，应用6220a－6220n使用远程显示或表示级协议。在一个实施例中，应用6220a－6220n是ICA客户端，由Florida，FortLauderdale有限公司的CitriX Systems开发。在其它实施例中，应用6220a－6220n包括远程桌面(RDP)客户端，由Washington，Redmond的Microsoft公司开发。

在其他实施例中，应用6220a－6220n包括任何类型的涉及到VoIP通信的软件，例如软IP电话。在进一步的实施例中，应用6220a－6220n包括涉及到实时数据通信的任何应用，例如用于流式传输视频和/或音频的应用。

[0771] 图40B示出了设备1250的示例性体系结构，该体系结构与图27描述的设备的体系结构相类似。总的来说，设备1250包括硬件层6206和被分为用户空间6203和内核空间6202的软件层。硬件层6206提供硬件元件，在内核空间6202和用户空间6203中的程序和服务在该硬件元件上被执行。硬件层6206也提供结构和元件，这些结构和元件允许在内核空间6202和用户空间6203内的程序和服务既在内部又在外部与设备1250通信数据。软件层包括程序、服务、处理、任务、线程和其他可执行指令以向设备1250提供逻辑、功能和操作。

[0772] 设备1250包括应用加速确定机制6275以及客户端侧加速程序6120。应用加速确定机制6275包括软件、硬件或软件和硬件的任何组合。在一些实施例中，应用加速确定机制6275包括任何类型和形式的可执行指令，例如拥有逻辑、功能、规则的程序、服务、处理、任务或线程，或者用于确定在客户端6205和/或服务器30上执行的应用6220a－6220n是否可被加速或在客户端6205和服务器30间的访问或通信可被加速的操作。在一个实施例中，应用加速确定机制6275使用数据库以确定是否应用6220a－6220n可被加速。例如，数据库可将应用6220a－6220n与可加速应用6220a－6220n的一个或多个加速技术相关联，该数据库可进一步的基于客户端6205和/或服务器30的用户、类型、形式、位置、处理能力和其他特征。在一些实施例中，应用加速确定机制6275使用在存储器中的查询表、文件、数据结构或对象，包括识别是否通过名称、类型或种类可由加速技术加速应用6220a－6220n的信息。在其他实施例中，设备1250和/或应用加速确定机制6275包括配置机制，例如用户接口、图形、命令行或其他以接收用户输入来识别、指定或配置是否可加速应用6220a－6220n或访问服务器30。

[0773] 在一些实施例中，应用加速确定机制6275从服务器30请求信息，该信息识别是否应用6220a－6220n可被加速，并且在进一步的实施例中，通过何种加速技术以及用于何种类型和形式的客户端6205。在又一个实施例中，应用加速确定机制6275包括历史信息数据库，关于在客户端6205和服务器30之间的应用6220a－6220n的性能，并带有或不带有一个或多个客户端侧加速技术，以提供比较和启发式信息的数据库，该信息关于应用6220a－6220n使用任何客户端侧加速技术在何处被加速，或可以被加速。例如，设备1250可以从客户端6205获取涉及到应用6220a－6220n的性能的网络相关性能信息。同样的，基于网络6204的变换操作和性能特征或受其影响，可调整应用6220a－6220n是否能够被加速的决定。

[0774] 在一个方面，应用6220a－6220n或者不可以被加速，或者虽然可以加速但该加速没有效率，或非常小。在一个实施例中，应用6220a－6220n的类型和形式不能使用协议或不能使用合适的方式与加速技术通信。在另一个实施例中，应用6220a－6220n通信的协议或方式可允许执行加速技术，但基于客户端6205、设备1250或服务器30的任何可操作的或性能特征，加速技术将没有效率或者只能提供非常小的加速。同样的，基于是否应用6220a－6220n能够被加速或者是否加速将符合性能提高所期望的预定阈值，设备加速确定机制6275可决定应用6220a－6220n不期望被加速。

[0775] 在另一个方面，设备6250在设备1250的存储器元件中存储客户端侧加速程序

6120，例如由设备的硬件层6206提供的存储器。在一个实施例中，设备1250动态的通过应用加速确定机制6275决定将被客户端6205使用的或者正使用的应用6220a－6220n可被在客户端6205上正执行的加速程序6120加速，并且从设备1250的存储器向客户端6205传输或者以其他方式通信加速程序6120。在另一个实施例中，设备1250决定在客户端6205和服务器30之间的通信可被在客户端6205上正执行的加速程序6120加速，并向客户端6205传递加速程序6120。在一些实施例中，设备1250从另一个计算设备6100，例如服务器30，接收、下载或获得加速程序6120。

[0776] 在一些实施例中，加速程序6120从设备1250的策略引擎3236接收、下载或获得策略信息。在其他实施例中，加速程序6120执行和操作策略引擎，或者独立于设备1250的策略引擎，或者与之关联。在其他实施例中，包引擎3240，或其部分，可在客户端6205上被操作，例如加速程序6120的部分。同样的，如上描述的，加速程序6120可在客户端6205上根据包处理定时器3242操作。在一个实施例中，加速程序6120在执行中的一个点可执行集成的加速技术，并响应由包处理定时器3242提供的粒度时间间隔。

[0777] 在一些实施例中，健康监控程序3216可检查和确定在任何客户端6205上的任何客户端侧加速程序6120的状态、错误或历史，上述客户端6205与设备1250通信，或设备1250向客户端6205传递加速程序6120。在一些实施例中，健康监控程序3216，或其部分在客户端6205上执行。

[0778] 现在看图41A，描述了方法6300的实施例，该方法由设备1250动态提供加速程序6120，并自动通过客户端6205安装和执行加速程序6120。总的来说，在步骤6310，设备1250从客户端6205拦截请求以与服务器建立通信会话。在步骤6315，设备1250向客户端6205传输加速程序6120，用于客户端6205自动安装和执行。在步骤6320，一旦接收到加速程序6120，客户端6205自动执行或施行加速程序6120的静默(silent)安装。在步骤6325，一旦完成加速程序6120的安装，客户端6205在网络堆栈6210中自动执行加速程序6120，以拦截客户端6205和服务器30之间的通信。在步骤6330，加速程序6120执行任意的多个加速技术，并可加密和/或解密通信。

[0779] 在进一步的细节中，在步骤6310，设备1250可拦截或以其他任何合适的方法和机制从客户端6205接收请求，以与服务器30建立通信会话。在一个实施例中，设备1250的包引擎6240从客户端6205拦截通信。在其他实施例中，设备1250与客户端6205例如与加速程序6120，建立第一传输层连接，以及代表客户端6205与服务器6205建立第二传输层连接。同样的，设备1250可接收、拦截或以其他方式获得传输到服务器30的任何客户端的通信。在一些实施例中，设备1250拦截用于客户端6205的请求，以与服务器30建立传输层连接。在其他实施例中，设备1250拦截请求用于通过传输层连接之上的任何协议层建立通信会话，例如HTTP的应用层协议。使用在客户端6205的网络堆栈6210的任何协议层建立通信会话的请求，可实行该方法的实施例。

[0780] 在步骤6315，设备1250向客户端6205传输加速程序6120。设备1250可在建立由客户端6205请求的通信会话之前、之中或之后的任何点传输加速程序6120。在一个实施例中，设备1250响应于拦截客户端请求向客户端6205传输加速程序6120。在另一个实施例中，设备1250向服务器30转发请求，并向客户端6205传输加速程序6120。在一些实施例中，设备1250与服务器30建立通信会话，并且一旦通信会话建立，设备1250传输加速程序6120。在又

一个实施例中，设备1250执行客户端6205或客户端6205的用户的验证和/或授权，并且如果被验证的用户或客户端6205被授权，则设备1250向客户端6205传输加速程序6120。在一个实施例中，设备1250向服务器30转发客户端的请求，用于验证和/或授权，并且如果服务器30验证和/或授权客户端的请求，设备1250向客户端6205传输加速程序6120。

[0781] 在一些实施例中，设备1250传输来自设备1250的存储器的加速程序。在其他实施例中，设备从服务器30请求加速程序6120，并向客户端6205转发接收的加速程序1620。在另一个实施例中，服务器30向客户端6205传输加速程序6120。在一个实施例中，设备1250向客户端6205传输统一资源定位器(URL)，用于客户端6205获得、下载或接收加速程序。在一些实施例中，URL识别在设备1250的存储器中的加速程序6120的定位，而在其他实施例中，URL识别在服务器30上的加速程序6120，例如提供加速程序6120用于下载的web服务器。在一个实施例中，加速程序6120被存储在客户端6205上，并且设备1250向客户端6205传输键值，例如加密或许可键值，用于客户端6205安装和使用存储在客户端6205上的加速程序6120。在一些实施例中，设备1250向客户端6205传输将被用于在客户端6205上安装和执行加速程序6120的任何文件、配置、数据或其他信息。

[0782] 在一个实施例中，加速程序6120被设计和构建以便由客户端6205自动安装和执行。加速程序6120可包括引起加速程序6120被客户端6205的操作系统登记和识别的任何文件、著录项、配置、数据或指令，上述操作系统可与任何类型或形式的操作系统一致。在一个实施例中，例如服务器或设备的另一个计算设备，向客户端6205传输加速程序，并且客户端6205自动安装和执行加速程序6120。在一个实施例中，加速程序6120被设计和构建为即插即用(uPnP)设备，以被加入运行的计算设备6100中。在一些实施例中，加速程序6120为自安装可执行程序，例如包括安装程序程序和加速程序6120的可执行程序。在其他实施例中，加速程序6120可包括多个文件，例如安装包或安装下载，例如需要在客户端6205的操作系统中登记和安装加速程序6120的文件。例如，加速程序6120包括包括.inf文件和.sys文件。.inf文件在操作系统的Microsoft Windows族中提供Windows Setup，该.inf文件带有建立设备需要的信息，例如用于设备的有效逻辑配置的列表，以及与设备相关联的驱动器文件名称。在一些实施例中，.inf文件可包括自动运行.inf文件，该自动运行.inf文件为配置文件，告诉或通知操作系统哪个可执行程序开始，以及涉及到开始可执行程序的任何配置信息。在一个实施例中，.sys文件为驱动器文件，包括加速程序6120或其部分。

[0783] 在步骤6320，客户端6205自动安装加速程序6120。加速程序6120可以与客户端6205的操作系统相一致的任何合适的方式被安装。在一个实施例中，一旦接收到加速程序6120，客户端6205安装加速程序6120。在一些实施例中，客户端6205自动施行或执行加速程序6120的静默安装。在一个实施例中，静默安装的执行对用户或者客户端6205的应用透明。在其他实施例中，加速程序6120的静默安装不需要重新启动或重新开始客户端6205。在另一个实施例中，静默安装不需要通过用户交互来开始和/或完成安装。在其他实施例中，当客户端6205正运行并对网络堆栈6210的网络层、会话层和/或应用层透明时，加速程序6120的静默安装发生。在一些实施例中，加速程序6120是由客户端6205执行的自安装可执行程序。在其他实施例中，客户端6205使用即插即用管理器安装加速程序6120。在一个实施例中，客户端6205包括安装管理器，其接收并安装加速程序6120。在另一个实施例中被设备1250传输的加速程序6120也包括安装加速程序6120的安装程序。

[0784] 在另一个实施例中，加速程序6120通过静默安装被自动安装。在一个实施例中，静默安装包括不用用户参与的安装。在另一个实施例中，静默安装包括不需要或没有用户的交互以开始或完成安装的安装。在一些实施例中，安装是静默的从而安装过程不显示关于安装的状态或处理的信息。在一个实施例中，安装是静默的从而其对于对用户透明。在其他实施例中，安装的静默是由于加速程序6120的安装不需要客户端6205的重新启动或重新开始。在另一个实施例中，安装是静默的从而在客户端6205的操作期间安装无缝发生，在上述操作期间，没有客户端操作的中断或干扰。同样的，通过不需要重启以及不向用户显示任何安装相关信息，加速程序6120可以对客户端6205的用户或应用透明的方式安装。

[0785] 为了阻止或避免客户端6205的重新启动或重新开始，在一些实施例中，当操作系统正运行时，客户端6205，例如客户端6205的操作系统，有用于即插即用设备的即插即用管理器以安装和配置驱动器，例如在加速程序6120的一个实施例中的网络驱动器。在一个实施例中，基于加速程序6120的安装包的配置，不指示即插即用管理器重新启动或重新开始客户端6205。在另一个实施例中，.inf文件不包括重新启动或重新开始计算机的指令。在一个实施例中，加速程序6120可作为并行部件被实施，而不代替共享的、使用中的动态链接库(DLL)。在其他特定实施例中，对于加速程序6120的网络驱动器，加速程序6120使用INetCfgPnpReconfigCallback网络驱动器API，以便用户不需要重新启动操作系统使得配置改变在驱动器中生效。另外，加速程序6120可拥有通知对象，该通报对象在它的INetCfgComponentControl的ApplyPngChange方法的实施中调用SendPnpReconfig API，以便向拥有对象的网络部件的驱动器发送配置信息。SendPnpReconfig API提供带有向驱动器发送数据的机制的通知对象，并且在一些实施例中，该通知对象用于避免在配置改变产生效用之前请求用户重新启动操作系统。

[0786] 在步骤6325，一旦自动的、静态的，透明的或以其他方式完成加速程序6120的安装，加速程序120在客户端6205自动执行。在一些实施例中，安装加速程序6120的安装程序开始或执行加速程序6120。在一些实施例中，用于加速程序6120的安装程序作出系统调用以在客户端6205的存储器中装载或执行加速程序120。在一个实施例中，加速程序6120的安装包括开始加速程序6120的指令、命令或指示。在一个实施例中，加速程序6120包括自动运行配置，例如自动运行.inf文件，通知客户端6205自动运行加速程序6120。在其他实施例中，客户端6205的即插即用管理器或者操作系统一旦安装则自动执行加速程序6120。在一个实施例中，加速程序6120包括被客户端6205开始的服务、处理、线程或任务。在一些实施例中，加速程序6120是被配置为自动开始的操作系统的服务。在一个实施例中，加速程序6120包括装载在客户端的操作系统的网络堆栈的存储器中的网络驱动器。

[0787] 在另一个实施例中，加速程序6120包括装载到客户端6205的存储器中的网络驱动器。在一些实施例中，加速程序6120被装载到分配给网络堆栈6210的存储器中。在一些实例中，加速程序6120在存储器区域或空间内被装载或执行，这使得加速程序6120访问网络堆栈的协议层，例如传输层。在其他实例中，加速程序在存储器内被装载或执行，使得加速程序6120访问内核级别数据结构6225。在其他实施例中，加速程序6120被装载到应用6220a-6220n的存储器中。在另一个实施例中，加速程序6120独立地在它自身的存储器空间或上下文中执行。在一个实施例中，加速程序6120运行在应用6220a-6220n的存储器空间或上下文中。在一些实施例中，加速程序6120被装载到用户模式存储器，或分配到用户模式6203的存

储器，而在其他实施例中，加速程序6120被装载到内核模式存储器中，或分配到内核模式6202的存储器。

[0788] 在一些实施例中，加速程序6120被装载到存储器中并且/或对于客户端的用户、客户端6205的应用、设备1250或者服务器30透明的在客户端6205上执行。在其他实施例中，加速程序6120执行以与网络堆栈6210的传输层接口，并且对于传输层上的任何协议层，例如会话或应用层，以及传输层之下的任何协议层，例如网络层，透明执行。在一个实施例中，加速程序6120对于客户端6205的任何传输层连接或传输层本身透明执行。

[0789] 在步骤6330，所装载、开始或以其他方式执行的加速程序6120执行加速程序6120的任意多个加速技术，例如由以下提供的任意技术：1)多协议压缩6238，2)传输控制协议池6224，3)传输控制协议多路复用6226，4)传输控制协议缓冲器6228，以及5)通过高速缓冲管理器6232的高速缓存。加速程序6220也可执行在客户端6205和服务器30之间通信的任何加密和/或解密。在一个实施例中，加速程序6120执行多协议压缩。在另一个实施例中，加速程序6120执行传输控制协议池化，并在进一步的实施例中，加速程序6120通过被池化的传输层连接执行多路复用。在一个实施例中，加速程序6120执行传输控制协议缓存。在一些实施例中，加速程序6120执行高速缓存。在其他实施例中，加速程序6120执行高速缓冲和压缩。在一个实施例中，加速程序6120执行带有传输层池化和多路复用的高速缓存。在另一个实施例中，加速程序6120执行带有传输层池化和多路复用的多协议压缩。在另一个实施例中，加速程序6120执行带有TCP缓存的高速缓存和/或压缩，并且在进一步的实施例中，带有TCP池化和多路复用。

[0790] 同样，客户端侧加速程序6120由设备1250动态提供，并且在客户端6205上以静默的方式或者对客户端6205的用户或应用透明的方式自动安装和执行，以执行一个或多个客户端侧加速技术以便在客户端6205和服务器30之间通信。加速程序6120可对于网络堆栈的任何协议层透明并且对于客户端的用户、客户端的应用、设备或服务器透明的执行这些加速技术。

[0791] 在另一个方面，设备1250可决定被客户端请求访问的应用是否可被加速，并且如果应用可被加速则向客户端6205提供加速程序6120。现在看图41B，描述了方法的另一个实施例。一旦请求建立连接或通信会话以及请求访问服务器上的应用，可实施该方法。对方法6350总的来说，在步骤6355，设备1250拦截来自于客户端6205的、请求存储服务器30上的应用6220a－6220n的请求。在步骤6260，设备1250决定应用6220是否可以被加速。在步骤6365，如果应用6220不能被加速，则应用在步骤6267向服务器转发该请求。在步骤6265，如果应用6226可被加速，则设备1250决定加速程序6120是否被安装在客户端6205上或者已经在先前被传递到客户端6205。如果加速程序6120仍然没有提供给客户端6205，则该方法6350在以上描述的方法6300的步骤6315继续以传输、安装和执行加速程序。如果加速程序6120已经被安装并且正在客户端6205上执行，则设备1250，在步骤6375，向客户端6205上的加速程序6120发送消息以加速应用6220。在方法6250的步骤6330，加速程序6120在通信上执行多个用于应用6220的加速技术，并且可以加密和/或解密这样的通信。

[0792] 在进一步的细节中，在步骤6355，设备1250可通过任何合适的方法和机制拦截来自客户端6205的请求，以访问服务器30提供的应用。在一个实施例中，设备1250的包引擎6240拦截来自客户端6205的通信。在其他实施例中，设备1250建立与客户端6205的第一传

输层连接，例如，与加速程序6120，并且代表客户端6205建立与服务器6205的第二传输层连接。同样，设备1250可接收、拦截或以其他方式获得传输到服务器30的任何客户端的通信。在一些实施例中，设备1250拦截用于客户端6205的请求以便通过建立的与服务器30的传输层连接访问应用6220。在其他实施例中，设备6205拦截请求以便通过在传输层连接之上的任何协议层，例如HTTP的应用层协议，建立通信会话。在一个实施例中，设备6205拦截来自客户端205的请求以便通过例如ICA或RDP的远程显示协议显示并提供来自服务器30的应用6220。

[0793] 在步骤6360，设备1250确定被客户端6205请求的应用6220是否可被加速。在一些实施例中，设备1250识别、抽取或以其他方式处理来自通过名称、类型或种类识别应用的所拦截的客户端的应用标识符。在一个实施例中，应用加速确定机制6275被设备1250使用来确定应用6220是否可被加速。在一些实施例中，应用加速确定机制6275在数据库、查询表或在存储器中数据的其他结构化源(例如数据结构或对象)中执行查询或查找，以便确定应用6220是否可被加速。在另一个实施例中，设备1250向服务器30发送例如请求的通信，以便确定应用6220是否可被加速。

[0794] 在其他实施例中，设备1250拥有性能日志或历史，以确定是否应用6220已经被加速，并且是否加速已经提高了应用6220的性能和操作。同样，如果这样的加速符合提高应用6220的性能或操作的预定阈值，设备1250可决定该应用6220可被加速。在又一个实施例中，基于网络6204、客户端6205或服务器30的当前操作和性能，设备1250提供启发式规则。在一个实施例中，如果客户端6205有特定性能和操作特征或者能力，例如，特定的速度处理器或者最小数量的存储器，则应用6220可被决定可以被加速。在一些实施例中，基于配置策略或规则，例如在设备1250的策略管理器中，应用6220可以被决定能够被加速。例如，在远程用户与带有特定类型的客户端6205之间传递的应用6220可被加速，该客户端访问特定类型应用220和/或服务器30。在其他实施例中，基于用户或客户端的验证和授权，应用6220可被决定能够被加速。在又一个实施例中，应用6220可被决定不期望被加速。例如，应用6220是不经常使用的类型。

[0795] 在步骤6365，如果应用6220被决定不能被加速或以其他方式不期望向客户端6205上的应用6220施加加速技术，则设备1250在步骤6368向服务器30转发拦截的客户端请求，并且不向客户端6205传输或提供加速程序6120。在一个实施例中，设备1250可执行或提供应用6220的基于设备的加速。在其他实施例中，设备1250不执行设备1250上的应用6220的加速。在又一个实施例中，如果设备1250确定应用6220不能或不期望被加速，则设备1250可执行一些加速技术和不是用于应用6220的其它。

[0796] 在步骤6365，如果应用6220被确定可以被加速或者以其他方式期望向客户端6205上的应用施加加速技术，则设备1250确定是否加速程序6120已经提供给客户端6205。在一个实施例中，设备1250确定是否加速程序6120已经安装在客户端6205上或者正在客户端6205上执行。在一些实施例中，设备1250向客户端6205上的加速程序6120发送通信以确定是否加速程序6120正运行在客户端6205上。在其他实施例中，设备1250检查日志文件或历史文件以确定是否加速程序6120已经被传输到客户端6205。在另一个实施例中，设备1250使用设备1250或者客户端6205的健康监控程序6216检查以确定是否加速程序6120正在客户端6205上执行。

[0797] 如果设备1250确定加速程序6120没有在客户端6205上被传输、安装和/或执行，则设备1250将根据结合图41A描述的方法6300的步骤提供加速程序6120。例如，设备1250向客户端6205传输加速程序6120，客户端6205一旦接收则自动安装和执行。在一个实施例中，一旦执行方法6300的实施例的合适的步骤，设备1250可在步骤6275向加速程序传递消息，以便向应用6220施加一个或多个加速技术。在其他实施例中，如果加速程序6120已经被安装和执行，则在步骤6375设备1250向加速程序6120传递消息以便向应用6220施加一个或多个加速技术。

[0798] 在一些实施例中，加速程序6120向已经被识别的应用6120执行加速程序6120可用的任何加速技术。在其他实施例中，设备1250向加速程序6120指示为应用6220实施哪个加速技术。在一个实施例中，在每个会话的基础上，加速程序6120可向应用6120施加期望的加速技术。即，从设备1250到加速程序6120的消息仅仅通知加速程序6120为该实例或应用6220的会话执行加速技术。在其他实施例中，一旦加速程序6120接收来自设备1250的消息以便为识别的应用6220施加加速技术，则加速程序6120为应用6220的任何实例或会话应用加速技术，或者直到客户端6205重新启动或重新开始，或者设备6205重新启动或重新开始。

[0799] 在一个实施例中，在步骤6375来自设备1250的消息不是应用专用的。例如，消息通知加速程序6120以执行用于客户端6205的任何应用的一个或多个加速技术。在一些实施例中，发送到客户端6205的消息通知加速程序6120停止使用用于应用6220或者用于所有应用6220a-6220n的任意一个或多个加速技术。在另一个实施例中，设备1250向加速程序6120传递消息以便忽略特定应用6220。在又一个实施例中，设备1250向加速程序6120传递消息以便向加速程序6120提供配置数据或信息，例如对加速技术或新的加速技术应用的更新。

[0800] 在步骤6330，加速程序6120为应用6220执行任意多个加速技术，例如以下提供的技术：1)多协议压缩技术6238，2)传输控制协议池6224，3)传输控制协议多路复用6226，4)传输控制协议缓冲6228，以及5)通过高速缓存管理器的高速缓冲6232。加速程序6120也可执行在客户端605和服务器之间的应用6220的通信的任何加密和/或解密。在一个实施例中，加速程序6120执行应用相关数据的多协议压缩。在另一个实施例中，加速程序6120执行传输控制协议池，并且在进一步的实施例中，加速程序6120通过池化的传输层连接执行多路复用。在一个实施例中，加速程序6120执行传输控制协议缓冲。在一些实施例中，加速程序6120执行高速缓存。在其他实施例中，加速程序6120执行高速缓存和压缩。在一个实施例中，加速程序6120执行带有传输层池化的高速缓存，并且在进一步的实施例中也带有多路复用。在另一个实施例中，加速程序6120执行带有TCP缓冲的多协议压缩，并且在进一步的实施例中，带有传输层池化并且，在又一实施例中，也带有多路复用。在另一个实施例中，加速程序6120执行带有压缩的高速缓存，并且在进一步的实施例中，带有TCP池化，并且在又一实施例中，带有多路复用。

[0801] 同样，设备1250动态确定是否加速应用或应用是否可被加速，并传递到客户端侧加速程序6120以在客户端6205上执行用于应用6220的任意一个或多个加速技术。另外，在一些实施例中，多个加速程序6120可通过设备被动态传递到客户端6205，并且由客户端6205自动安装和执行。例如，依照用于到服务器6205的每个连接的技术和方法，或者与应用6220的每个通信会话，可提供加速程序。同样，客户端6205可自动安装和执行多个加速程序6120以便为每个服务器630或每个应用6220a-6220n处理和执行加速。

[0802] 现在看图41C，描述了以集成方式执行多个加速技术的方法6380的实施例。总的来说，在步骤6280，加速程序6120通过传输层连接在传输层拦截客户端6205和服务器30之间的通信网络包。在步骤6390，加速程序6120通过内核级别数据结构，例如通过到客户端6205的网络堆栈6210的API提供的数据结构，在传输层访问网络包。在步骤6395，加速程序6120使用内核级别数据结构在接口点或在加速程序6120的执行点以集成方式执行多个加速技术。

[0803] 在进一步的细节中，在步骤6385，加速程序6120通过任何合适的方法和机制经传输层连接拦截在客户端6205和服务器30之间通信的网络包。在一个实施例中，加速程序6120通过客户端拦截请求的网络包或涉及到请求的网络包，或者其响应，以在客户端6205和服务器30之间建立传输层连接。在另一个实施例中，加速程序6120拦截请求的网络包或涉及到请求的网络包，或者其响应，以便通过客户端6205和服务器30之间的传输层连接访问或使用应用6220。在一个实施例中，加速程序6120在传输协议层经传输驱动器接口或者以其他方式在网络堆栈6210的传输协议层处连接的网络驱动器拦截网络包。在另一个实施例中，加速程序6120在传输协议层，或网络堆栈6210的任何其他协议层通过网络驱动器接口规范(NDIS)驱动器、或者迷你端口驱动器、或者迷你过滤器驱动器拦截网络包。在一些实施例中，加速程序120在传输层通过钩子或过滤机制拦截网络包。

[0804] 在步骤6390，加速程序6120访问或以其他方式获得在传输层通过内核级别数据结构6225拦截的网络包的信息和数据。通过使用内核级别的数据结构6225，加速程序6120可获得关于有效负荷或在传输层被网络包携带或传输的一个或多个协议的信息和数据。在一些实施例中，使用内核级别数据结构来表示在传输层和/或其上的网络堆栈的层处的网络包来启用加速程序6120在传输层执行或操作多个加速技术，该加速技术用于被传输层网络包携带的协议层。在一个实施例中，使用单个的内核级别数据结构6225禁止或避免拷贝和存储器定位以及上下文切换在网络堆栈6210的多个协议层使用多个数据结构。在一个实施例中，加速程序6120向第二数据结构拷贝内核级别数据结构6225，该第二数据结构可包括另一个内核级别数据结构或用户级别数据结构。

[0805] 在步骤6395，加速程序6120在单独的接口点或在程序6210的特定区域中或在可执行指令集中或在程序6210的一个执行点，执行、或操作多个加速技术。加速技术6120执行加速程序6120的多个加速技术的任意个，例如以下提供的任意技术：1)多协议压缩技术6238，2)传输控制协议池6224，3)传输控制协议多路复用6226，4)传输控制协议缓冲6228，以及5)通过高速缓存管理器的高速缓冲6232。在加速程序6120的加速技术执行中的相同点，加速程序6120也可执行在客户端6205和服务器30之间的应用6220的通信的任意加密和/或解密。

[0806] 在一个实施例中，加速程序6120在可执行指令集中，例如功能调用或一个地点或位置，执行任何期望的彼此并发的多个加速技术。例如，加速程序6120通过内核级别数据结构获得被拦截的网络包，并接着执行指令，该指令代表彼此间并发的加速技术的逻辑、功能、规则或操作。同样，网络包的信息和数据可通过内核级别数据结构6225一次抽取或获得，并作为用于代表加速技术的加速程序6120的任何指令的输入、参数、自变量和条件而使用。虽然网络包携带更高级别的协议数据和信息，但在一些实施例中的加速程序6120在执行期间的一个点或一个时间处理网络包和更高级别的协议数据和信息。另外，加速程序

6120可以集成方式以期望的顺序执行多个加速技术中的每一个，例如向高速缓存管理器6232中存储的压缩数据，或者从高速缓存获取的压缩/或未压缩数据。

[0807] 在一个实施例中，加速程序6120执行彼此间并发的多协议压缩和高速缓存。在另一个实施例中，加速程序6120通过池化的传输层连接执行彼此间并发的操作，上述操作涉及到传输控制协议池化和多路复用。在一个实施例中，加速程序6120在压缩和高速缓存或者TCP池化和/或多路复用之后执行传输控制协议高速缓存。在一些实施例中，加速程序6120执行高速缓存。在一个实施例中，加速程序6120执行与传输层池化和多路复用并发的高速缓存。在另一个实施例中，加速程序6120在传输层池化和多路复用之后执行多协议压缩。在另一个实施例中，加速程序6120在TCP缓冲之后执行高速缓存和/或压缩，并在进一步的实施例中，在TCP池化和多路复用之后。

[0808] 虽然通常描述加速程序与执行加速技术并发，但是并发执行也可包括不涉及加速但在每个加速技术之间被集成和执行的其他逻辑、功能和操作。当用于加速技术的可执行指令和其他操作或功能在单独的接口点或在加速程序的执行点被执行，加速程序仍然获得带有这样的集成的操作和性能的效率。另外，用于在传输层携带或其上的协议层的加速技术在某一时间，在传输层的某一位置被处理。同样，由于网络包在网络堆栈6210的这些更高级别中，或者在网络堆栈6210的稍后点遍历和获得处理，则用于这些更高级别的协议的加速技术不需要再被应用。

[0809] 在其他方面，第一程序6222和加速程序6120(或者在该实施例中也称为第二程序)可被使用。在一个实施例中，可使用第一程序6222和第二程序6120来促进并建立与服务器30的虚拟专用网络连接，例如通过设备1250，通过该连接可应用客户端侧加速技术。在另一个实施例中，使用第一程序6222来安装和执行第二程序，或加速程序6120。

[0810] 现在看图42A，描述了用于实现该方面的方法6400的实施例。总的来说，在步骤6402，客户端6205登记进入并与设备6205建立通信会话。在步骤6404，设备1250发送第一程序6222到客户端6205。在步骤6406，客户端6205安装和执行第一程序6222，接着安装和执行加速程序6120，即第二程序。在步骤6407，客户端6205通过建立的加密数据通信会话与专用网络上的资源通信并对其访问。在步骤6410，客户端6205从设备1250登出并停止与设备1250的通信会话。

[0811] 在方法6400的步骤6402，客户端6205执行登入过程并通过网络6204与设备1250建立加密数据通信会话。在一个实施例中，加密数据通信会话用作隧道以桥接从客户端6205到任何服务器30的话务量，服务器30驻留在专用数据通信网络中的设备1250后。在实施例中，客户端6205使用web浏览器，例如Microsoft Internet Explorer®或者Netcape Navigator®，登入并使用安全套接层(SSL)或其他加密方法，例如IPSec，以及传输层安全(TLS)与设备1250建立数据通信会话。在另一个实施例中，可使用协议，例如安全套接字层(HTTPS)上的超文本传输协议，来开始被加密的数据通信会话。

[0812] 在步骤6404，响应于登入和加密的数据通信会话的建立，设备1250向客户端6205通过网络6204发送第一程序。设计并构建第一程序，或以其他方式配置第一程序，以便作为隧道端点，用于通过加密数据通信会话的通信。在一个实施例中，第一程序包括插件应用，该插件应用被客户端6204的浏览器自动安装和执行。例如，第一程序可包括ActiveX控件，其作为插件提供将被Mirosoft InternetExplorer®web浏览器执行。在另一个实施例中，

第一程序可包括Java小程序(applet)，其作为插件提供被NetscapeNavigator®web浏览器或在网络环境之上工作的另一个控件或设计部件执行。

[0813] 在步骤406，客户端安装和执行第一程序6222，其中执行第一程序包括在客户端6205上安装第二程序。在一个实施例中，第一程序6222被自动安装和执行，例如使用以上结合方法6300和图41A讨论的任何技术。在一些实施例中，第一程序6222从设备1250获得、下载或接收第二程序或加速程序6120。在另一个实施例中，第一程序6222包括用于第二程序的安装程序或安装管理器，例如自动安装和执行第二程序的加速程序6120，例如通过对客户端6205的用户、客户端6205的应用6220、设备1250或服务器30静默安装或透明安装的方式。

[0814] 在一个实施例中，第二程序被配置，部分地，以便从运行在客户端6205上的应用6220拦截通信，上述客户端6205预定网络6204上的资源，并且为了通过加密的数据通信会话将拦截的通信提供给第一程序6222用于发送到设备1250。第二程序也可被配置以提供内部网网络名称解析服务以及可选的分解网络话务量。通过分解话务量，实施例可以确定什么话务量被引导向第一程序6222的SSL隧道或加密隧道以及什么话务量被允许，或者在客户端6205的正常、日常事务或典型操作中，允许沿着网络堆栈6210的传输层的处理。在实施例中，第二程序包括动态拦截器(例如，过滤器设备驱动器)，该拦截器作为“钩子”插入到客户端6205的操作系统中。例如，第二程序可包括过滤器设备驱动器，该过滤器设备驱动器连接到客户端操作系统的传输层堆栈，例如MicrosoftWindows®操作系统的传输层堆栈。

[0815] 在步骤6408，一旦第一和第二程序已经被安装，则运行在客户端6205的应用通过已经建立的加密数据通信会话在专用数据通信网络6204上可与资源通信并访问资源，例如应用和数据。关于图42B下面将更详细的讨论通信发生的方式。注意，在一个实施例中，如上描述的第一程序和第二程序的功能被单独的控件或程序部件执行，客户端6205自动安装和执行上述控件或程序部件，例如加速程序6120。除了提供虚拟专用网络连接和通信之外，第一程序6222和/或例如加速程序6120的第二程序在客户端的通信之上通过虚拟专用网络连接可执行此处描述的任何加速技术，该网络连接例如到设备1250的被加密的隧道或桥接器。

[0816] 在步骤6410，客户端6205执行登出过程，从网络6204断开，客户端终止与设备1250的加密的数据通信会话。在一个实施例中，在登出时，第一程序6222自动清除对客户端6205的操作系统所作的修改，以将操作系统返回到安装第一程序6222和/或第二程序之前的状态。在一个实施例中，第一程序6222和/或第二程序也包括卸载器或卸载指令，以便从客户端6205的操作系统中移除第一和第二程序，或者以非侵入模式从客户端6205上的其它操作移除第一和第二程序到客户端6205的继续的操作。在又一个实施例中，在提供的任何通信连接或会话期间，第一程序6222和/或加速程序6120移除被客户端6205的应用使用的任何文件，例如临时文件或cookies(曲奇)。

[0817] 图42B描述了另一个方法6450的实施例，通过该实施例客户端6205与在专用数据通信网络6204上的资源通信，或访问该资源。例如，方法6450代表可实施方法6400的步骤6408的方法。总的来说，在步骤6452，通过第一程序和/或第二程序，客户端6205做出新的连接或解析域名，例如TCP/IP域名解析。在步骤6454，执行第二程序。在步骤6456，第二程序拦截从客户端6205到专用网络的通信，并将该通信再路由或发送到第一程序6222。在步骤

6458，第一程序6222通过建立的通信加密会话终止或代理该连接，隔离有效负载并封装用于传送的有效负载。在步骤6460，第一程序6222通过预先建立的加密通信会话向专用网络中的设备1250通过公共网络发送拦截的通信。在步骤6462，设备1250解密从第一程序接收的通信，并向例如服务器30的合适的目标资源转发该解密的通信。在步骤6464，目标资源处理解密的通信，并且在步骤6464，目标资源向设备1250发送响应通信(如果有的话)。在步骤6468，设备1250加密响应通信并通过预先建立的加密通信会话经由公共网络向客户端6205的第一程序6222发送加密通信。在步骤6470，第一程序6222解密响应通信并通过第二程序向合适的客户端应用转发解密的通信。

[0818] 在步骤6452，客户端6205的应用6220获得新的连接或通过客户端6205的网络堆栈6210的传输协议层解析域名。在一个实施例中，应用6220可请求建立在客户端6205和服务器30之间的传输层连接，或者在客户端6205和设备1250之间。在另一个实施例中，应用220或客户端6205可请求访问服务器30提供的应用6220。例如，通过传输代表服务器30上执行的应用6220的输出的ICA或RDP的远程显示协议，服务器30可为基于服务的计算或瘦客户计算提供。在另一个实施例中，客户端6205可请求访问服务器30的资源，例如文件或目录，或者电子邮件服务。在一些实施例中，客户端6205可在公共网络40上，并且服务器30在专用网络40’上。在其他实施例中，客户端6205和服务器30可在不同的专用网络上。

[0819] 在步骤6454，在任何传输层功能被开始之前，第二程序自动或以其他方式执行一个或多个功能。在一些实施例中，第二程序是或以其他方式包括加速程序6120。在一个实施例中，第二程序拦截或接收步骤6452的客户端请求。在一些实施例中，客户端6205的应用6220对网络堆栈6210做出API调用，上述网络堆栈6210被第二程序拦截。在网络堆栈6210的传输层处理任何API调用之前，第二程序被钩入或以其他方式接口到网络堆栈6210，以便在用于传输的通信经由传输层连接传输或处理之前，执行逻辑、规则、功能或操作。

[0820] 在步骤6456，第二程序拦截来自客户端205的通信，例如通过客户端6205上的任何应用6220a－6220n，应用6220a－6220n被预定用于网络40’上的资源，并且第二程序重路由应用6220a－6220n到第一程序6222，在一个实施例中客户端6205包括Act iveX控件插件、Java applet(小应用程序)或其他工作在网络环境中的控件或程序部件。第二程序可从提供拦截的通信的网络包或包中访问、读取或以其他方式获得目的信息，以确定通信是为网络预定的，例如在设备1250后的专用网络40’。例如，第二程序从网络包中抽取或拦截目的IP地址和/或端口。一旦确定拦截的信息是网络40’预定的，第二程序通过任何合适的接口方法和机制，例如通过任何内部处理通信接口或API调用，向第一程序6222传递拦截的通信。在一个实施例中，拦截的通信保持不变被发送到第一程序6222，或者在其他实施例中，在发送到第一程序622之前，拦截的通信被第二程序预先处理。例如，第二程序可从拦截的通信清除有效负载或者向第一程序6222转发有效负载。

[0821] 在步骤6458，第一程序6222终止或代理每个被拦截的通信，并且第一程序6222通过已建立的加密数据通信会话为传输准备拦截的通信。在一个实施例中，第一程序6222通过已建立的加密数据通信会话分离有效负载并封装该有效负载用于分发。在另一个实施例中，第一程序6222当从第二程序接收到拦截的通信时，封装它。在一些实施例中，有效负载是TCP有效负载并例如通过设备1250，在客户端6205和服务器30之间被封装入新的TCP连接。

[0822] 在步骤6460，第一程序6222通过预先建立的加密数据通信会话，在网络6204上向设备1250发送拦截的通信。在一些实施例中，第一程序6222加密拦截的通信并向设备1250发送该解密的拦截的通信。在一个实施例中，依照SSL协议实施加密。在另一个实施例中，加密基于TLS。第一程序6222或加速程序6120可使用任何类型和形式的加密和/或解密。

[0823] 在步骤6462，设备1250作为终止第一程序6222发送的连接的代理。设备1250解密从第一程序6222接收的通信，并通过第二连接将解密的通信转发到合适的目标资源，上述第二连接是设备1250与网络40'上的目标资源已经建立的连接。在一个实施例中，依照SSL协议或其他可应用的加密和解密协议可实施解密。在一些实施例中，设备1250在转发到目标资源的通信之上执行一个或多个加速技术，例如下述的一个或多个提供的技术：由1)多协议压缩6238，2)传输控制协议池6224，3)传输控制协议多路6226，4)传输控制协议缓冲6228，以及5)通过高速缓存管理器的高速缓冲6232。

[0824] 在步骤6464，目标资源处理解密通信。在一个实施例中，解密通信是建立连接或通信会话的请求。在另一个实施例中，解密通信是开始或访问代表客户端6205的应用6220的请求。在其他实施例中，解密通信是对于web页面的请求，例如从web服务器30接收web页面的HTTP请求。

[0825] 在步骤6466，如果解密通信包括存在响应的请求，则目标资源向设备1250发出响应通信。在一些实施例中，响应包括如客户端6204请求的建立连接或通信会话的确认。在其他实施例中，响应包括错误消息。在一个实施例中，响应包括验证请求或挑战响应机制。在一些实施例中，响应包括将被客户端6205使用的加速程序6120。在另一个实施例中，响应包括HTML，例如将被客户端6205显示的web页面。在其他实施例中，响应包括对象，例如动态产生对象。

[0826] 在步骤6468，设备通过预先建立的加密数据通信会话在网络40上向第一程序6220发送响应通信。在一个实施例中，设备1250加密响应通信并向第一程序6222发送加密的响应通信。在一些实施例中，依照SSL协议或其他可应用的加密和解密协议实施加密。另外，设备1250在与客户端6205通信之上可执行任何加速技术，例如多协议压缩6238'、高速缓存6232'或TCP缓冲6228'。

[0827] 在步骤6470，第一程序6222解密响应通信并通过第二程序向相应的应用6222转发该通信。第一程序6222可使用任何合适的接口方法和机制以便向第二程序通信，例如通过任何类型和形式的内部处理通信机制或API调用。第二程序可通客户端6205的网络堆栈6210向应用6220提供响应通信。同样，应用6220在没有对应用6220作任何改变或修改时透明地接收响应通信。

[0828] 依照另一个实施例，在步骤6458在通过网络40发送通信之前，客户端6205执行拦截的通信的附加处理。由于实施例在解密数据之前，提供在客户端作为终止连接的代理的VPN解决方法，所以可更有效率的执行附加处理。为了启用客户端应用使用它们选择的任何IP地址并在运行时间动态改变这些地址，上述的处理可包括拦截通信的域名服务(DNS)名称解析。这样附加的处理使得实施例可以更有效的与其他技术结合，例如全局服务负载平衡，以便在分布的网关或服务器中获得更大的可用性和更大的有效性。附加的连接处理也可启用关于拦截的通信的细节日志和统计数据的保持。

[0829] 在另一个实施例中，设备1250终止从客户端6205上的第一程序接收的通信，并进

一步处理包括在此处的一个或多个请求，而非像步骤6462所示的向网络40’上的目的地转发该通信。此进一步的处理可包括后端加密，其中在向网络40’上的合适目的地分发之前，通信被设备1250重加密，因此提供端对端网络安全。该目的地其后将适当的解密话务量和响应。更进一步的，这样的处理允许设备1250在高速缓存之外服务响应，而不需要请求目的服务器附加的工作，允许设备1250执行本地网络负载平衡、在通信上的全局服务负载平衡和/或压缩以提高网络40的效率和响应。

[0830] 依照以上描述的方法，在客户端205和网络40之间建立基于加密的数据通信会话的VPN。例如，在实施例中，通过HTTPS建立安全VPN。其后，通过该加密数据通信会话，从客户端6205到网络40的所有通信通过第一程序被路由到设备1250以及与之反向。值得注意的是，虽然使用HTTPS可建立加密数据通信会话，但是通过加密数据通信会话传输的通信不需要是HTTPS包数据或者甚至HTTP包数据。例如，通信也可包括传输控制协议/用户数据报协议(TCP/UDP)或因特网控制消息协议(ICMP)包数据，但是这些实例不是用于限定的。另外，虽然引用图42B描述的方法描述了在客户端6205上的应用和网络40上的资源之间的请求－响应类型通信，加密的通信不需要是基于请求－响应的。而是，通信可为任何类型。因此，任何可建立连接或通信会话的客户端应用，例如UDP会话，可发送和接收加密的通信。

[0831] 在另一个方面，加速程序6120可从客户端动态旁路任何的中间设备而与服务器30连接或通信。例如，客户端6205可通过一个或多个中间设备，例如设备1250与服务器连接。由于一个原因或另一个，中间设备可不再被客户端6205用于与服务器30通信，例如，设备1250可被列入维持或者可处在重新启动或重新开始的过程中。加速程序6120确定中间设备不可用并自动与服务器30建立不同的连接或通信会话。这可对于客户端6205的用户或应用透明发生，以便连接和/或通信会话不显示已经改变或以其他方式已经被干扰。

[0832] 现在看图43，描述了用于自动旁路中间设备的方法6500的实施例。总的来说，在步骤6505，加速程序6120通过例如设备1250的中间设备在客户端6205和服务器30之间建立传输层连接。在步骤6510，加速程序6120确定中间设备对于客户端6205到服务器30的通信是不可用的，上述通信通过建立的传输层连接。在步骤6515，加速程序6210在客户端6205上拦截从客户端6205到服务器30的通信。在步骤6520，加速程序6120在客户端6205和服务器30之间建立第二传输层连接，并作为结果，旁路中间设备确定对于客户端到服务器30的通信不可用。在步骤6525，加速程序6120通过第二传输层连接向服务器30传输客户端6205的拦截的通信。

[0833] 在进一步的细节中，在步骤6505，加速程序120通过中间设备在客户端6205和服务器30之间建立传输层连接。在一个实施例中，中间设备包括设备6205。在其他实施例中，中间设备包括下述之一：高速缓存器、服务器、网关、防火墙、桥接器、路由器、切换器、网络集线器、代理或作为或提供任何这些类型或形式的中间设备的功能和操作的任何软件应用或程序。在一个实施例中，中间设备可在服务器30上操作。在一些实施例中，通过相同类型和形式或不同类型和形式的多个中间设备建立传输层连接。在另一个实施例中，传输层连接包括传输层连接的池的连接，该池连接或者作为客户端6205被建立或者在设备1250被建立。

[0834] 在步骤6510，加速程序120确定中间设备对于客户端6205到服务器30的通信是不可用到的或者以其他方式不可使用，上述通信通过建立的传输层连接。加速程序6210可通

过任何合适的方法和/或机制确定中间设备的状态或可用性。在一个实施例中，通过接收错误消息或到中间设备的传输相关联的失败答复，加速程序6120确定中间设备不可用。例如，当通过已建立的传输层连接从客户端6205传输通信时，加速程序6120可接收失败的传输层通信响应。在另一个实施例中，加速程序6120可以预定的频率向中间设备传输查验(ping)命令，以便监控中间设备的状态和有效性。如果加速程序6120没有从中间设备接收答复，或者在一些实施例中，接收延迟的答复或带有比需要的等待时间更长的答复，加速程序6120可确定中间设备对于客户端6205是不可用或不能使用的。在其他实施例中，服务器30、设备1250或中间设备可向客户端6205发送消息，或提供识别中间设备消息的加速程序6120对于客户端6205是不可用或不能使用的。在一些实施例中，建立的传输层连接被干扰或被中断，或在其他实施例中，被关闭。

[0835] 在步骤6515，加速程序6120拦截从客户端6205到服务器30的通信，该通信经由中间设备通过已建立的传输层连接被预定传输。加速程序6120可在网络堆栈6210的任何点和任何协议层拦截通信。在一个实施例中，在建立的传输层连接上传输之前，加速程序6120在传输协议层拦截通信。例如，在一些实施例中，加速程序6120包括拥有传输驱动器接口或以其他方式接口到传输协议层的网络驱动器。其他实施例可包括第一程序6222和如结合图42A－42B描述的作为第二程序的加速程序6120，其中或者第一程序6222或者加速程序6120拦截通信。

[0836] 在步骤6520，为了旁路在步骤6510中被确定对于客户端6205是不可用或不能使用的中间设备，加速程序6120为客户端6205建立到服务器30的第二传输层连接。在一个实施例中，当客户端6205和服务器在客户端6205和服务器30之间可路由的相同的网络6205上或不同的网络上时，加速程序6120建立直接到服务器30的第二传输层连接。在另一个实施例中，加速程序6120建立带有第二中间设备的第二传输层连接，例如第二设备1250'。在一些实施例中，加速程序6120请求设备1250与服务器1250建立另一个传输层连接。在一个实施例中，设备1250使用传输层连接的池的第二传输层连接到服务器30。在另一个实施例中，加速程序6120请求服务器30建立第二传输层连接。在一些实施例中，加速程序6120与服务器30使用来自由加速程序6120建立的传输层连接的池的第二传输层连接。

[0837] 在一个实施例中，加速程序120在步骤6520对客户端6205的用户或应用6220透明的建立第二传输层连接，或者在一些实施例中，对于传输层之上或之下的任何协议层透明。在一些方面，一旦在步骤6510确定中间设备对于客户端6205不可用或不应该被客户端6205使用，则用于客户端6205的第二传输层连接被自动建立。在其他实施例中，一旦不能向服务器30传输拦截的通信，例如第一试图传输通信，第二传输层连接被自动建立。在一些实施例中，一旦一个或多个通信的重试传输失败，或一旦用尽预定数量的重试，则第二层连接被自动建立。在另一个实施例中，一旦确定中间设备延迟传输的速度或网络包的接收，引起等待或其他方式以不期望的方式影响传输层连接的使用，则建立第二传输层连接。在一个实施例中，加速程序6120执行负载平衡并建立旁路中间设备的第二传输层连接，以便卸载到客户端6205和/或第二中间设备的中间设备的任何处理或操作。

[0838] 在步骤6525，加速程序6120通过第二传输层连接向服务器30传输拦截的客户端6205的通信。在一个实施例中，加速程序6120直接向服务器30传输拦截的通信。在其他实施例中，加速程序6120通过第二中间设备，例如第二设备1250，传输拦截的通信。通过使用第

二传输层连接，加速程序6120旁路中间设备并继续与服务器30的客户端6205的应用6220的操作。在一个实施例中，好象应用6220一直使用先前的或第一建立的传输层连接，客户端6205的应用6220继续操作并继续与服务器30通信。同样，加速程序6120阻止、避免或避开任何通信中断、干扰、等待、延迟或其他可操作或性能问题，如果中间设备没有被加速程序6120旁路，则可发生这些问题。在另一个方面，即使对于从中间设备访问存在问题或干扰中间设备访问中存在问题，该技术向客户端6205自动提供服务器30或远程访问应用的持续的访问。

[0839] 此外，可使用以上描述的重定向和旁路技术执行客户端6205上的负载平衡和话务量管理，以访问一个或多个服务器30，服务器30向客户端6205提供应用6220a－6220n或其他内容和功能。例如，在一个实施例中，带有增长的传输层连接，以及降低的响应、性能或其他操作的速度，客户端使用的访问服务器的中间设备或设备可被超载。一旦确定中间设备或设备正降低性能，加速程序6120可将客户端重定向到另一个中间设备或设备，或服务器，以便旁路在客户端到服务器的端对端连接中的任何性能瓶颈。

[0840] 在其他方面，客户端侧加速技术可被关联到客户端的网络堆栈的传输协议层或在其上执行。加速程序6120可包括执行下述任何一个或多个技术的可执行指令：1)传输控制协议(TCP)缓冲6228，2)TCP连接池6224，3)TCP路复用6226。在一些实施例中，当加速程序6120透明的处理在客户端的网络堆栈的传输协议层拦截的通信时，加速程序6120可控制并管理客户端的TCP连接，以及在该连接之上客户端6205的应用6220a-6220n的使用和传输。图44示出了实现TCP缓冲技术的方法6600的实施例，而图45A-45B示出了TCP连接池化技术的实施例，图46，47和48描述多路复用技术。

[0841] 总的来说，图44描述的方法6600的实施例中，在步骤6605，加速程序6120拦截从客户端6205到服务器30的通信，例如客户端205访问服务器30的请求。在步骤610，加速程序6120确定在接收的服务器响应的消耗速度和客户端传输的产生速度之间的差是否下降到预定阈值以下。如果在步骤6615产生和消耗速度之间的差没有下降到对预定阈值之下，加速程序6120在步骤6617向服务器转发该通信。如果在步骤6615，速度差低于预定阈值，则在步骤6620，加速程序6120在客户端6205的存储器中存储通信。在步骤6625，加速程序6120确定是否速度差已经变得高于预定阈值了，并且如果高于则向服务器30转发存储的通信。否则，加速程序6120在客户端6205的存储器中保持通信，直到在步骤6625改变的速度差高于预定阈值的时间点。例如，如果客户端6205正以高于客户端6205可消耗产生的响应的速度向服务器30传输请求，加速程序6120保持进一步的传输直到速度差已经改变的将来的时间点。

[0842] 在进一步的细节中，在步骤6605，加速程序拦截从客户端6205到服务器30的通信。加速程序6120可在任何点和网络的任何协议层拦截通信。在一个实施例中，在已建立的传输层连接上传输之前，加速程序6120在传输协议层拦截通信。例如，在一些实施例中，加速程序6120包括拥有传输驱动器接口或以其他方式接口到传输协议层的网络驱动器。其他实施例可包括第一程序6222和如结合图42A-42B描述的作为第二程序的加速程序6120，其中第一程序6222或者加速程序6120拦截通信。在一个实施例中，通信包括客户端6205对服务器30使用或访问资源的请求，例如应用6220。

[0843] 在步骤6610，加速程序6120确定是否消耗的速度和/或客户端6205的产生的速度

的差值低于预定阈值。在一个实施例中，加速程序6120计算并追踪由客户端6205传输到服务器30传输的请求的数量，并且在另一个实施例中，加速程序6120计算并追踪客户端6205从服务器30接收的响应的数量。在一些实施例中，客户端6205追踪传输的响应和在每个应用6220的基础上接收的请求。该响应和请求可在网络堆栈6210的任何协议层被追踪。在一个实施例中，从提交点到传输层或者到客户端6205和服务器30之间的传输层连接，计算和追踪被客户端6205或应用6220传输的请求的数量。同样的，在另一个实施例中，从到达传输层的接收点或者从客户端6205和服务器30之间的传输层连接，和/或响应提供给网络堆栈6210的传输层之上的例如应用层的协议层的点，计算并追踪被客户端6205或应用6220接收的来自服务器30的响应的数量。

[0844] 在一些实施例中，加速程序6120访问、检查、或者以其他方式获得信息和数据，这些信息和数据是关于接收和发送由加速程序6120在客户端6205和服务器30之间建立的传输层连接的TCP缓冲器。例如，加速程序6120可决定任何TCP/IP缓冲器的缺省和最大尺寸，以及缓冲器现在被使用的部分，以决定从客户端6205到服务器30网络包的发送和接收之间的速度差。在其他实施例中，加速程序6120使用任何类型和形式的拥塞算法，以确定是否存在由于从客户端6502到服务器30的网络包的消耗和产生的差引起的拥塞。在另一个实施例中，例如通过网络驱动器或者TCP服务提供者，加速程序6120与传输层连接，或者从传输层连接使用的拥塞算法获得信息或数据。例如，在一个实施例中，加速程序6120决定关于连接使用的拥塞窗口的信息和数据。

[0845] 通过加速程序6120的任何合适的方法和机制可配置、指定、定义或识别预定阈值。在一个实施例中，可将阈值指定为在客户端6205和/或应用6220的产生速度和消耗速度之间的百分比、相对值、绝对值或其它。通过任何时间周期以任何粒度，用于消耗和/或产生的速度可分别被消耗的接收和产生的传输的数量所标识。在一些实施例中，可指定阈值作为在客户端6205和/或应用6220的产生速度和消耗速度之间的数量差，并且在一些实施例中，作为一段时间的数量差。例如，可指定阈值为客户端6205已经产生的请求多于客户端消耗的请求的时间点。在另一个实例中，可以指定阈值为当每一时间段客户端6205产生对服务器的请求多于在相同时间段内客户端6205消耗的请求时的时间点。

[0846] 在步骤6615，如果在客户端6205和/或应用6220的产生和消耗速度差不低于预定阈值，则在步骤6617加速程序6120向服务器6260转发该通信。在一些实施例中，加速程序为通信执行任意的加速技术。例如，可通过池化的多路技术传输层连接向服务器转发通信，另外，可压缩该通信。在其他实施例中，客户端6205可向为客户端6205提供到服务器30连接的设备1250转发通信。

[0847] 在步骤6615，如果客户端6205和/或应用6220的产生和消耗的速度差低于预定阈值，则在步骤6620，加速程序6120在客户端6205的存储器中存储通信。在一些实施例中，存储器可为客户端6205的内核模式6202的存储器，然而，在其他实施例中，存储器可在客户端6205的用户模式6203中。在一个实施例中，加速程序6120可通过高速缓存管理器6232在高速缓存中存储通信。在其他实施例中，加速程序6120可使用对象、数据结构或其他可被加速程序6120访问的数据元件来缓冲、保持或其它方式存储拦截的通信。在一个实施例中，拦截的通信可以压缩模式存储在存储器中。在另一个实施例中，加速程序6120向第一程序6222发送拦截的通信以便在存储器中存储或保持，在稍后的时间点用于传输。

[0848] 在步骤6625，加速程序6120确定何时向服务器30传输存储的通信。在一个实施例中，一旦加速程序6120在步骤6617向服务器30转发存储的通信，加速程序6120执行步骤6610和6615以确定客户端6205的产生和消耗之间的速度差是否高于阈值。在一些实施例中，加速程序6120以规则的或预定的频率或在轮流检测或事件的基础上，比较产生和消耗速度的差，并且当该差高于预定阈值时，加速程序6120向服务器30转发该通信。在其他实施例中，加速程序6120设置或配置定时器以便确定访问拦截的通信的时间长度。一旦定时器终止，加速程序6120向服务器30传输存储的通信。在另一个实施例中，在存储拦截的通信之后，加速程序6120检查客户端6102消耗的服务器响应的数量。如果消耗的响应的数量大于预定值，则加速程序6120释放来自存储器缓冲器或存储器的拦截的通信，并将该用于传输的通信提交给服务器30。

[0849] 如果在步骤6625，加速程序6120确定产生或消耗的速度没有以合适的方式改变，则加速程序6120在存储器中保持或维持拦截的通信，直到到达合适的时间点。在一个实施例中，即使产生和/或消耗的速度没有改变，加速程序6120仍然在步骤6617向服务器转发通信。例如，在等待产生和/或消耗速度改变和速度不改变的期间，加速程序6120向服务器30转发该通信。

[0850] 虽然通常结合拦截的通信或请求讨论TCP缓冲技术，但可并发的、接近同时的或同步的实施方法6600的实施例，用于客户端6205到服务器的多个拦截的通信。另外，在另一个实施例中，可在客户端上关于从客户端到多个服务器30的通信来实施方法6600。例如，方法6600的第一个实例可在客户端6205和第一服务器30’之间实施，并且方法6600的第二个实例可在客户端6205和第二服务器30”之间实施。进一步的，在一些实施例中，使用每个应用单独的产生和消耗速度，可实施方法6600用于第一应用6200a也可用于第二应用6200b。在其他实施例中，也可实施方法6600用于第一应用6200a但不用于第二应用6200n。

[0851] 根据另一个方面，客户端侧加速程序6120减少了服务器30和/或设备1250的处理负载，这些处理负载是由于重复的打开和关闭客户端连接而引起的，其时客户端通过打开一个或多个与每个服务器的连接并维持这些连接以便通过客户端6205到服务器的连接获得重复的数据访问。该技术此处通常被称为“连接池化”。现在看图45A，方法6700总的来说，在步骤6702，加速程序6120拦截访问服务器的应用的请求，并且在步骤6704，确定与请求关联的服务器的标识。在步骤6706，加速程序6120确定加速程序6120是否已经拥有到服务器30的应用6220的空闲的已建立的传输层连接。如果不存在到服务器30的应用6220的空闲的传输层连接，加速程序6220在步骤6708建立到服务器30的由客户端6205使用的传输层连接。在步骤6706，如果存在应用6220可用的传输层连接，在步骤6710，加速程序6120通过可用传输层连接翻译对于传输或通信的应用请求。

[0852] 进一步，在步骤6712，加速程序6120接收来自服务器30的请求的响应，并且在步骤6714将响应译成对应用6220的响应。在步骤6716，加速程序6120可维持或保持被客户端6205的任意应用6220a-6220n使用的传输层连接打开。通过维持客户端6205上的与服务器30的打开的传输层连接，并通过按照需要打开和关闭与应用的连接，加速程序6120释放通过网络40，例如因特网，与服务客户端6205相关联的TCP连接负载问题的服务器。在步骤6718，如果确定客户端6205的一个或多个应用6220不再使用连接访问服务器30，则在一些点的加速程序6120关闭传输层连接。

[0853] 在进一步的细节中，在步骤6702，加速程序6120通过客户端6205的任何应用6220a-6220n拦截访问服务器30的请求。在一些实施例中，在通过传输层连接建立或传输请求之前，在传输协议层拦截该请求。在其他实施例中，在传输层之上的任何协议层或在传输层连接拦截请求。在一个实施例中，应用6220的请求是打开或建立与服务器30的传输层连接的请求。在一些实施例中，响应于该请求，加速程序6120建立客户端6205的应用6220a-6220n使用的传输层连接的池的第一传输层连接。在另一个实施例中，应用请求是通过客户端6205的已建立的传输层连接访问服务器的请求。

[0854] 在步骤6704，加速程序6120通过任何合适的方法和机制从请求确定服务器30的标识。在一些实施例中，服务器30的域名或因特网协议地址被识别或以其他方式被请求的内容引用，例如请求的文本串可识别服务器30的域名。在一个实施例中，服务器30的标识被TCP包的头信息确定，例如目标因特网协议地址和端口号。在另一个实施例中，服务器30与应用6220相关联，并且加速程序6120在数据库或其他结构化信息存储中查找或寻找相关信息。

[0855] 在步骤6706，加速程序6120确定是否存在应用6220可用的或者以其他方式空闲的传输层连接。在一个实施例中，加速程序6120可以还没有建立的与服务器30的传输层连接，并且同样的，没有应用6220可用的传输层连接。在另一个实施例中，加速程序6120可拥有先前建立的与服务器30的传输层连接，但确定另一个应用6220当前正使用该连接。如下面将进一步详细讨论的，基于被服务器30接收的用于应用6220的消息的长度，例如对请求的响应，加速程序6120确定是否已建立的传输层连接对于另一个应用是可用的，或者可被应用6220a-6220n共享，和/或加速程序6120确定在服务器30和应用6220之间的通信现在是否是空闲的。

[0856] 在步骤6708，如果加速程序6120确定传输层连接对于应用6220不可用，则加速程序6120与服务器30建立传输层连接。在一些实施例中，在步骤6708建立的传输层连接是与服务器30的第一传输层连接，并且在另一个实施例中，传输层连接是到服务器30的多个传输层连接的第二传输层连接。在又一个实施中，加速程序6120等待已经建立的传输层连接变得可用或者空闲以向服务器30传送应用请求。例如，加速程序6120可决定第一应用6220a通过建立的连接即刻完成了与服务器30的交流。

[0857] 在步骤6710，加速程序6120翻译将通过传输层连接传输到服务器6106的应用请求。在一些实施例中，加速程序6120使用用于传输层连接的一个端口号，用于客户端6205的所有应用6220a-6220n共享连接。在一些例子中，加速程序6120基于应用跟踪请求和用于对应用上请求的未完成的响应。同样，加速程序6120识别哪个应用620通过到服务器30的传输层连接在任何给定时间点传输并接收网络包。在一个实施例中，每次只有一个应用6220在传输层连接上发送和接收，并且因此加速程序6220知道哪个应用6220正使用该连接。在一些实施例中，加速程序6120将请求与应用6220的处理id关联。在其他实施例中，加速程序6120提供端口号，并将之与应用6220关联，并修改将被传输到应用的指定端口号的TCP网络包中的端口号。在另一个实施例中，应用6220提供端口号并且加速程序6120改变或以其他方式提供TCP网络包中的相应的端口号。

[0858] 在步骤6712，加速程序6120接收对来自服务器30的应用的请求的响应。在一个实施例中，服务器30不响应该请求。在另一个实施例中，服务器30使用错误或失败消息响应。

在一些实施例中，服务器30使用多个响应做出响应。在其他实施例中，服务器使用包括多个网络包或多个TCP段的响应做出响应。在另一个实施例中，服务器30使用一个或多个网络包响应，上述网络包识别与应用6220相关联的或指定到应用6220的源端口号。在一个实施例中，服务器30使用一个或多个网络包响应，上述网络包识别传输层连接的源端口号并被用于客户端6205的多个应用。

[0859] 在步骤6714，加速程序6120以响应应用6220的方式翻译或以其他方式处理来自服务器30的响应。在一个实施例中，加速程序6120使用应用6220的端口号来代替接收到的网络包的或包的源端口号。在另一个实施例中，加速程序6120通过追踪机制确定应用6220正使用传输层连接，并且通过网络堆栈6210向应用6220传输响应。在一个实施例中，响应不被修改并通过在连接的传输层之上的网络堆栈6210的任何协议被传输用于处理。在一些实施例中，在处理响应和向应用6220转发响应之前，加速程序6120等待被接收的响应的多个部分，例如TCP段。在一个实施例中，加速程序6120向第一程序6222传递响应，第一程序6222与应用6220接口并向其提供响应。

[0860] 在步骤6716，加速程序6120维持或保持在从客户端6205到服务器30的一个或多个传输层连接池中的传输层连接的打开。在一个实施例中，加速程序6120或网络堆栈6210的传输层驱动器包括保持有效机制，当连接是空闲时，例如当没有数据传输的地方，该机制周期性的探测连接的另一端。为了接收响应以确定虽然连接可能是空闲但是连接仍然活动，保持有效机制可发送此消息。保持有效消息和相应的响应可包括任何类型和形式的格式、命令、指示或通信。同样，在一些实施例中，加速程序6120通过传输层驱动器向传输层连接传输保持活动消息或引起其传输。在一些实施例中，加速程序6120设置用于保持有效消息的频率，并且在其他实施例中，基于使用连接的应用6220a-6220n的行为或活动，改变保持有效的消息的频率。

[0861] 在一些实施例中，加速程序6120拦截通过传输层连接接收的任何RST和/或FIN命令，即TCP/IP命令以重新设置和/或终止TCP连接。在一个实施例中，加速程序6120忽略、不采取行动、或者以其他方式撤销、删除或者清除所拦截的RST和/或FIN命令。在另一个实施例中，加速程序6120拦截并接收RST和/或FIN命令但向连接的另一端发送消息，以便保持或维持连接打开。在其他实施例中，响应于由于处理RST和/或FIN命令而关闭已建立的传输层连接，加速程序6120建立新的传输层连接。

[0862] 在其他实施例中，加速程序6120在客户端6205的所拦截的通信中插入指令、命令或指示，以便指示服务器30保持连接打开或不关闭连接，除非客户端6205发送命令要求关闭。例如，在一个实施例中，加速程序6120拦截HTTP协议的GET请求的通信，例如协议版本1.0，并向服务器30的通信中插入保持有效(keep-al ive)头，例如“Connect ion:Keep-Al ive”。在其他实施例中，GET请求或其他HTTP命令可包括保持有效头。在这些实施例中，加速程序6120可拦截通信并检查保持有效头，接着向服务器30转发该通信。在一些实施例中，使用HTTP的1.1版本或更高版本，从而保持活动机制是隐含的，以便服务器30保持连接打开直到客户端6205请求关闭该连接。在其他实施例中，加速程序6120保持到服务器30的传输层连接打开，直到客户端6205被重新启动或重新开始，网络40变得不可用或者客户端6205从网络40断开，或者服务器30被重新启动或重新开始。

[0863] 在步骤6718，加速程序6120可在任何需要的时间点关闭客户端6205和服务器30之

间的任意一个或多个传输层连接。在一些实施例中，一旦在客户端6205上使用连接的一个或多个应用6220a-6220n终止，加速程序6120关闭传输层连接。在其他实施例中，一旦用于使用连接的应用6220a-6220n的超时期满，则加速程序6120关闭传输层连接。例如，加速程序6120可配置、设置或提供定时器以便在预定的时间周期终止，并且如果在该时间周期内连接为空闲或保持空闲，则加速程序6120关闭连接。在一些实施例中，服务器30可被重新启动、重新开始或连接被干扰或中断并且加速程序6120关闭连接。在一些实施例中，一旦完成向服务器30发送请求并从服务器30接收了所有的响应数据，则加速程序6120传输RST和/或FIN命令或引起其被传输，以便关闭连接。在其他实施例中，一旦重新启动或重新开始客户端6205，断开网络40连接或网络40不可用，或重新开始或重新启动服务器30，则关闭传输层连接或传输层连接池。

[0864] 在一些实施例中，当加速程序6120确定只有第一传输层连接需要由客户端6205的一个或多个应用6220a-6220n共享到服务器30的连接时，则到服务器30的第一传输层连接保持打开，而到服务器30的第二传输层连接被关闭。在其他实施例中，加速程序6120保持到任何服务器30的传输层连接的池，并基于增加的请求、通信或被客户端6205上的应用6220a-6220n使用的传输层连接使用，建立第二或多个到给定服务器30的连接。

[0865] 虽然通常讨论方法6700的实施例涉及从客户端6205到服务器30的一个或多个传输层连接的池，但是加速程序6120可并发的、接近同时的或同步的建立在客户端和多个服务器30中的每个之间的传输层连接池。同样，第一应用6220a和第二应用6220b可正使用到服务器30a的一个或多个传输层连接的第一池，并且第三应用6220c和第四应用6220d可正使用到服务器30b的一个或多个传输层连接的第二池。另外，方法6700的实施例的每个步骤可在不同的实例中以及不同的频率被执行。在一些实施例中，可使用加速程序6120的多个实例出力到每个服务器30的一个或多个传输层连接的每个池。

[0866] 现在看图45B，描述了加速程序6120的流程图，提供被两个应用6220a和6220b使用的传输层连接，该传输层连接在一个实施例中到服务器30，或者在另一个实施例中到设备1250。如通过步骤6752描述的，使用应用6220提供的网络地址1，在客户端6205上的加速程序6120打开在客户端6205和服务器30或设备1250之间的第一传输层连接。由于TCP/IP协议对打开的连接应用多阶段握手，则示出的步骤6752为两路步骤。

[0867] 如步骤6754所示，一旦建立传输层连接，加速程序6120从应用6220a拦截GET请求，应用6220a指定/sales/forecast.html的路径名。由于在加速程序6120和服务器30或设备6205之间没有空闲的传输层连接被打开，所以加速程序6120打开传输层连接。在一个实施例中，如步骤6756所示，加速程序6120将应用6220a的请求映射到指定服务器30的网络地址2的第二网络地址。例如，加速程序120执行网络地址翻译，以便修改到应用6220a请求的服务器30'的目的IP地址和/或目的端口，或者控制或响应到请求的另一个服务器30"的目的IP地址和/或目的端口。在另一个实施例中，当接收到请求，或应用6220s产生请求时，加速程序6120向服务器30或设备1250发送请求。

[0868] 如步骤6758所示，加速程序6120也可向服务器30或设备1250传递GET请求。在一个实施例中，设备1250向服务器30转发请求，并且在进一步的实施例中，设备1250通过在设备1250和服务器30之间的池化的或池化并多路复用的传输层连接来转发请求。如步骤6760所示，在一些实施例中，服务器30响应以被请求的web页面。如步骤6762所示，加速程序6120向

应用6220a转发web页面。在一个实施例中，如步骤6764所示，在加速程序6120和服务器30或设备1250之间的传输层连接被关闭。在其他实施例中，加速程序6120拦截关闭请求，并忽略遗留传输层连接打开的该请求。根据TCP/IP协议，关闭网络连接可包括多阶段处理。因此，步骤6764的流程线被示为双向的。在其他实施例中，根据池化方面的技术，为第一应用6220建立并被其使用的传输层连接保持打开或者以其他方式维持以提供来自相同的应用6220a或不同应用，例如第二应用6220b的进一步的数据步骤。

[0869] 在步骤6766，加速程序6120拦截从第二应用6220a到服务器30的请求。如果存在空闲的传输层连接向第二应用6220b开放并被其使用，例如在步骤6756建立的用于第一应用6220a的传输层连接，则加速程序6120使用该先前建立的传输层连接。同样，在步骤6766，第二传输层连接不需要被打开。相反，加速程序6120建立到服务器30或设备1250的第二传输层连接。在步骤6768，加速程序从第二应用6220b拦截请求，例如请求web页面/sales/forecast.html，并在步骤6770向服务器30或设备1250传输该请求。由于在加速程序6120和服务器6120之间的空闲连接已经打开，则不需要加速程序6120负担带有处理负载的服务器6120来打开进一步连接。在步骤6772，加速程序6120从服务器30拦截或接收响应，例如从传输层连接通过设备1250，并向第二应用6220b转发响应。在步骤6776，加速程序120从第二应用6220b拦截关闭请求，并且在一些实施例中，关闭连接，而在其他实施例中，忽略该请求，并使得连接保持满足来自第一应用6220a、第二应用6220b或者客户端6205的又一应用6220c-6220n的进一步的数据请求。

[0870] 在步骤6776，存在多种情况导致加速程序6120关闭与服务器30或应用1250的连结。例如，一旦确定客户端6205已经找到了用于应用6220a和6220b的所有的请求数据，或者一旦停止、关闭或退出应用6220a和6220b，则客户端6205或加速程序6120可开始FIN(完成)命令。在一些实施例中，在相似的条件下，客户端6205或加速程序6120也可开始RST(复位)命令。除了关闭在加速程序6120和服务器30或设备1250之间的连接之外，RST命令导致许多的内务处理操作被执行，以便保持服务器侧连接处于良好的顺序。特别的，TCP协议保证RST命令拥有正确的SEQ(序列)号，以便服务器将接受段。然而，RST命令不能保证拥有准确的ACK(确认)号。为了考虑这种请求，加速程序6120跟踪由服务器30或设备1250发送的数据的字节，以及被客户端6205确认的字节。如果客户端6205仍然没有被服务器30确认所有的数据，加速程序6120计算未确认的字节，并向服务器6205发送ACK。

[0871] 另外，虽然没有在图45B中示出，服务器30或设备1250可关闭自身和客户端6205之间的连接。服务器30或设备1250将向客户端6205发送FIN命令。在响应中，在一些实施例中，加速程序6120关闭连接，并且在进一步的实施例中，与服务器30或设备1250重建另一个连接。

[0872] 此外，虽然图45A的方法6700的实施例和图45B的流程图的实例通常讨论为池化被多个应用使用的一个或多个传输层连接，但是池化技术可被应用于单独的应用6220，该应用6220请求或开始多个传输层连接并通过这些连接请求。例如，在一个HTTP协议实施例中，可建立传输层连接用于来自应用的每个HTTP请求。使用该技术，在没有打开和关闭用于每一请求的传输层连接时，一个或多个传输层连接池可被应用220使用。

[0873] 在另一方面，可通过相同的或共享的传输层连接使用用于多路复用应用请求的技术，例如通过结合图45A-45B描述的池化技术而建立的传输层连接。在一些实施例中，并且

通过检查是否已经从服务器30完全接收对应用请求的响应的内容，可确认已建立的传输层连接的可用性，以及通过连接可多路复用来自多个应用的请求。如下将被进一步讨论的，在一个实施例中，使用响应的内容长度参数，并且在另一个实施例中，使用响应的块传输编码头以检查是否已经接收了响应的全部数据。在一个方面，检查是否已经接收了响应的全部数据，以便确定池化的连接是否现在是空闲的用于应用使用，和/或是否建立到服务器的连接池的另一个传输层连接，例如在图45A描述的方法6700的步骤6706和6708。在另一个实施例中，检查用于响应的内容长度的技术被用作通过相同的传输层连接对多个应用中的请求多路复用的技术。

[0874] 现在看图46，描述了通过从客户端6205到服务器30的单一的传输层连接对请求多路复用的方法6800的实施例。总的来说，在步骤6805，加速程序6120在客户端6205和服务器30之间建立传输层连接。在步骤6810，加速程序6120拦截到服务器30的第一应用6220a的第一请求。在步骤6815，加速程序6120确定是否传输层连接正被另一个应用使用，或者处于空闲。在步骤6817，如果传输层连接可被应用6220a使用，则接着在步骤6820，加速程序6120向服务器传输请求。否则，在步骤6817，如果传输层连接不能被应用6220a使用，则接着加速程序6120在步骤6819或者等待一段时间并返回步骤6815，或者建立被应用6220使用的第二传输层连接。在步骤6825，加速程序6120从服务器接收对应用请求的响应。在步骤6830，加速程序6120通过第二应用6220b拦截第二请求，并在步骤6815继续，以便确定是否传输层连接可被第二应用6220b使用。在一些实施例中，在步骤6825接收第一请求的响应之前，或者在接收所有的响应数据之前，在步骤6830，加速程序6120拦截第二应用6220b的请求。如此处进一步讨论的，在一些实施例中，加速程序6120使用内容长度检查技术来确定何时传输层连接何时空闲或者何时应用已经接收了对请求的响应的所有数据。

[0875] 在进一步的细节中，在步骤6805，加速程序6120在客户端6205和服务器30之间建立传输层连接。在一些实施例中，加速程序6120与设备1250或中间设备或通过设备1250或中间设备建立传输层连接。在一个实施例中，加速程序6120建立传输层连接，作为到服务器30的传输层连接的池。同样，在一些实施例中，传输层连接可包括到服务器30的第二或第三传输层连接。在其他实施例中，加速程序6120可像以前讨论的通过第一程序6222建立传输层连接。在一些实施例中，响应于客户端6205的第一应用6220a的请求，加速程序6120建立传输层连接。

[0876] 在步骤6810，加速程序6120拦截第一应用6220a访问服务器30的第一请求。在一些实施例中，在通过传输层连接建立或传输请求之前，该请求在传输层被拦截。在其他实施例中，请求在传输层或传输层连接之上的任何协议层被拦截。在一些实施例中，请求被第一程序6222拦截。在一个实施例中，应用6220a的请求是打开或建立与服务器30的传输层连接的请求。在另一个实施例中，应用请求是通过已建立的传输层连接或通过设备1250访问服务器的请求。

[0877] 在步骤6815，加速程序120确定传输层连接是否空闲或可被第一应用6220a使用，或者传输第一应用6220a的第一请求。在一些实施例中，加速程序6120从一个或多个传输层连接的池确定池中的哪个传输层连接是空闲的或者可被第一应用6220a使用。在一个实施例中，由于加速程序6120响应于请求或即刻在请求之前建立传输层连接，加速程序6120确定传输层连接是空闲的。在一些实施例中，加速程序6120可能还没有从应用6220接收到任

何请求，并认可该请求为将被加速程序6120截获和处理的第一请求。在另一个实施例中，加速程序6120跟踪用于任何在传输层连接上传输的请求的未完成的响应的数量，并且如果没有未完成的响应，则加速程序6120识别传输层连接可被第一应用6220a使用。在又一个实施例中，加速程序6120识别传输层连接当前空闲。例如，加速程序6120可开始对到服务器的请求保持有效，以便保持连接打开。在一些实施例中，当最后的事务已经结束，但服务器30和/或客户端6205还没有传输RST和/或FIN命令时，传输层连接空闲。

[0878] 在一些实施例中，加速程序6120可检查响应的内容长度以便确定是否从服务器30到第一应用6202a的第一请求完成与否，加速程序6120是否已经接收了响应的所有数据。如上所提及的，在实施例中的这些技术也可用于确定建立用于池化技术的另一个连接。关于该技术，将被图47和48用于在一个实施例描述响应的内容长度参数检查，或另一个实施例中响应的分度传输编码头的检查以确定是否响应的所有数据已经被接收。图47描述了称为TCP段6900的TCP包的TCP部分。TCP段6900包括TCP头6902和体6904。在一个实施例中，体6904包括其他数据和信息、HTTP头和消息，其中TCP包携带HTTP的应用层协议。在一些实施例中，内容长度参数6906位于HTTP头中，或在其中发现或在其中被引用。在一个实施例中，加速程序6120使用内容长度参数6906以便确定用于响应的所有数据是否被接收。

[0879] 图48描述了TCP包的TCP段的另一个实施例。在传输层连接上使用HTTP协议的一些实施例中，块传输编码头可被呈现并指示块传输编码已被应用到TCP段或包。同样，在该实施例中，消息的长度被分段编码定义。为了将消息转换为一系列段，每个块在块尺寸域中具有自身的长度指示器，块编码修改消息的体。TCP段7600包括TCP头(未示出)和体。体包括，在其他信息中，HTTP头7602A-7602C以及信息。HTTP头7602A-7602C包括七个块尺寸域7606A-7601C，以及六个块消息数据7604A-7604F。

[0880] 如图48所示，块尺寸域7606A-7606G被链接到一起，或以其他方式被引用或关联。块尺寸域7606A在块消息数据7604A中指示消息的长度，并且块尺寸域7606C在块消息数据7604C中指示消息的长度，以下类似。最后的块尺寸域7606G包括长度值零，指示后面没有块或任何消息。在另一个实施例中，加速程序6120通过块尺寸域确定客户端6205是否已经接收了响应的所有数据。

[0881] 虽然图47和48一般性的描述了用于检查用于对请求的响应的所有数据是否已经被接收，但是服务器30或设备1250可使用这些技术向客户端6205发送异步消息或与之通讯。另外，虽然这些技术总的结合图47和48被描述为用于HTTP协议，但是这些技术可用于任何协议层上的任何协议，这些协议层提供将被客户端605传输或接收的数据的长度的指示。同样，在一些实施例中，加速程序6120访问、抽取、检查、分析或以其他方式处理包括在任何协议层的网络包的任何部分，以确定与客户端和服务器或设备之间的请求、响应或通信相关联的所有数据是否已经被接收。在又一个实施例中，加速程序6120追踪在客户端6205和服务器30之间传输、接收和确认的字节的数量，以便确定在客户端6205和服务器30之间对于应用6220是否有字节未完成。

[0882] 通过使用以上描述的内容长度技术，加速程序6120可重新使用到服务器30先前使用的或被客户端6205的任何其他应用6220a-6220n在处理中使用的相同传输层连接。在步骤6817，加速程序6120确定是否传输层连接可用于传输第一请求，如果可以在步骤6820向服务器30传输请求。否则，在步骤6819，加速程序6120可等待直到所有用于应用的未完成的

请求的数据被接收。例如，加速程序6120可设置定时器例如到短的时间段并继续到步骤6815。在一些实施例中，加速程序6120响应于客户端6205的网络堆栈6210的包处理定时器来检查所有的数据是否已经被接收。在另一个实施例中，在步骤6819，加速程序6120建立另一个传输层连接以便传输第一应用6220a的第一请求。

[0883] 在步骤6820，加速程序6120可追踪哪个应用6220正有在连接上的未完成的请求或响应，或者正使用连接。例如，一次只有一个应用6220可在连接上传输请求或接收响应。同样，加速程序6120知道哪个应用6220正使用连接。在一些实施例中，加速程序6120使用用于传输层连接通信的一个端口号，上述传输层连接通信用于共享连接的客户端6205的所有应用6220a-6220n。在一些例子中，基于应用，加速程序6120追踪请求和在应用上对于请求的未完成的响应。在一些实施例中，加速程序6120将应用6220的处理id与请求关联。在又一个实施例中，加速程序6120传输在相同的网络包、包、TCP段或者段中的第一应用6220a的请求以及第二应用6220b的请求。在其他实施例中，通过相同的传输层连接作为一个或多个TCP段窗口的一系列TCP段的部分，加速程序6120传输应用6220a-6220n的多个请求。

[0884] 在其他实施例中，加速程序6120使用端口编号机制和/或方案来跟踪和识别接收的哪个响应或消息用于哪个应用6220a-6220n。在其他实施例中，加速程序6120为应用6220提供端口号并与将端口号与应用6220关联，并且修改TCP网络包中的端口号，该TCP网络包将被传输到应用的指定端口号。在另一个实施例中，应用6220提供端口号，加速程序6120依照TCP网络包改变或提供端口号。同样，在一些实施例中，加速程序6120可交织来自客户端6205的多个应用6220a-6220n的请求，使得应用6220a-6220n可同时使用传输层连接。

[0885] 在步骤6825，加速程序6120从服务器30接收到第一应用6220a的第一请求的响应，例如通过设备6205，并向第一应用6220a提供响应。在一些实施例中，加速程序6120通过网络堆栈6210向第一应用6220a提供响应，例如通过连接的传输层之上的协议层允许或开始响应的处理。在另一个实施例中，第一程序6222向第一应用6220a提供响应。在其他实施例中，加速程序6120可通过中间处理通信机制或例如API的接口向第一应用6220a提供应用。在一些实施例中，加速程序6120仅接收响应的一部分，例如图48中描述的在多块消息中的第一块。

[0886] 在步骤6830，加速程序6120拦截第二应用6220b的请求以便访问服务器30。在一些实施例中，加速程序6120在步骤6825之前拦截第二应用6220b的请求。在其他实施例中，加速程序6120在步骤6825接收响应期间拦截第二应用6220b的请求。在另一个实施例中，在客户端6205或加速程序6120接收用于第一应用6220a的第一请求的响应的所有数据之前，加速程序6120拦截第二应用6220b的请求。一旦拦截第二应用6220b的请求，在一个实施例中，加速程序6120继续步骤6815，以便确定是否通过传输层连接多路复用第二请求，或者是否建立另一个传输层连接，例如在连接池中的另一个连接。在其他实施例中，加速程序6120通过与第一应用6220a相同的连接传输第二应用6220b的请求，而第一应用6220a有未完成的响应或者没有接收对第一请求的响应的所有数据。在另一个实施例中，在第一应用6220a已经接收到响应之后，并且在产生与第一应用6220a相关的任何已产生的RST和/或FIN之前，加速程序6120传输第二应用6220b的请求。

[0887] 虽然对加速程序6120一般性的讨论结合客户端侧实施和加速技术的执行，但是加速程序6120与设备1250接口并与之协同工作，设备1250也实施并执行设备侧加速技术。在

一个实施例中，客户端侧加速程序6120和设备1250可彼此协同工作以便在客户端6205和服务器30之间的通信之上执行多个加速技术。在一些实施例中，客户端侧加速技术120和设备1250二者都提供TCP池化和多路复用，例如在客户端6205和服务器30之间提供级联或端对端池化以及多路复用机制。例如，加速程序6120可向设备1250提供第一池化的传输层连接，随后向服务器30提供第二池化的传输层连接。在另一个实施例中，加速程序6120可通过客户端6205上的第一池化传输层连接多路复用应用请求，第一池化传输层连接依次被设备1250通过到服务器30的第二池化传输层连接多路复用。在一些实施例中，加速程序120提供对传输来自于客户端的请求的节流机制，而设备1250为从服务器30到客户端6205的传输响应提供节流机制。在另一个实施例中，加速程序6120为客户端6205执行客户端侧高速缓存，而设备1250为客户端6205和其他客户端6205提供对象的高速缓存，例如动态产生对象。

[0888] 在一些实施例中，除了在客户端6205和/或设备上执行加速技术之外，或与之协同，加速程序6120和设备可在客户端6205和网络40之间通过设备1250提供虚拟专用网络和通信。在另一个实施例中，加速程序6120可压缩从应用6220接收的数据，并且设备1250在接收到时则可以解压缩所压缩的数据。相反，设备1250可压缩从专用数据通信网络40’上的服务器上的应用6220传送来的数据，并且加速程序6120在接收到该数据时可解压缩该压缩数据。同样，加速程序6120和设备1250在加密数据通信或隧道会话中可作为端点，其中加速程序6120加密从应用6220传送的数据，并且在接收到该加密数据时，设备1250解密该加密数据。在类似的方式中，设备1250加密从专用数据通信网络上的应用6220传送的数据，并且在接收到该数据时，加速程序6120解密该数据。

[0889] D.计算环境的加速传送的实施例

[0890] 考虑上述描述的B和C部分中的结构、功能和操作，在一些实施例中，计算环境到客户端的传送可被加速。例如，可使用此处描述的实施例可从中心公司数据中心传送被应用处理的流应用和数据文件到远程用户位置，例如公司的分支机构。设备和加速程序提供用于加速从服务器到远程客户端的任何传输层有效负载的端对端加速技术，例如被流式传输的应用和数据文件。基于多个执行方法以及基于通过策略引擎应用的任何验证和授权，应用传送管理系统提供应用传送技术以便向远程用户的桌面传送计算环境。使用这些技术，远程用户可获得计算环境并对来自任何网络连接设备的服务器存储应用和数据文件进行访问。

[0891] 现在看图49A，描述了实现以上描述的加速和应用传送的系统和方法的实施例。总的来说，客户端10通过网络40、40’和设备1250与服务器30通信。例如，客户端可驻留在公司的远程办公室里，例如分支机构，并且服务器可驻留在公司数据中心。客户端10包括客户端代理560以及计算环境15。计算环境15可执行或操作用于访问、处理或使用数据文件的应用。计算环境15、应用和/或数据文件可通过设备1250和/或服务器30传送。在一些实施例中，客户端10也可包括加速程序4120、收集代理404，以及流客户端562。服务器30包括应用传送系统500，以及在一些实施例中，策略引擎406。

[0892] 在一个实施例中，应用传送系统500可驻留在服务器30上或在其上执行。在另一个实施例中，应用传送系统500可驻留在多个服务器30-30’上或在其上执行。在一些实施例中，应用传送系统500可在服务器群组内执行。在一个实施例中，执行应用传送系统500的服务器30也可存储或提供应用和数据文件。在另一个实施例中，一个或多个服务器30的第一

集合可执行应用传送系统500，并且不同的服务器30’可存储或提供应用和数据文件。在一些实施例中，每个应用传送系统500、应用和数据文件可驻留或位于不同的服务器。在一个实施例中，应用传送系统500也可包括策略引擎406。在另一个实施例中，策略引擎406的执行与应用传送系统500分离。在一些实施例中，策略引擎406和应用传送系统500在同一服务器上。在其他实施例中，策略引擎406在设备1250上执行。在又一个实施例中，应用传送系统500和/或策略引擎406的任何部分可驻留、执行、存储于或分发到设备1250或多个设备。

[0893] 在一些实施例中，客户端代理560如上所描述的包括任何的流客户端562、收集代理404和/或加速程序6120。在一个实施例中，客户端代理560、流客户端562、收集代理404和/或加速程序6120形成或包括进可执行指令的一个单独的程序或集合中，以提供每一个的功能、逻辑和操作。在其他实施例中，流客户端562、收集代理404和加速程序6120的每个与客户端代理560分离的执行。在一个实施例中，客户端10执行客户端代理560。在另一个实施例中，客户端10执行流客户端562。在一些实施中，客户端10执行收集代理404。在一个实施例中，客户端执行加速程序6120。在一些实施例中，客户端10执行带有一个或多个流客户端562的客户端代理560、收集代理404或加速程序6120。在其他实施例中，客户端10执行流客户端562和加速程序6120。在一个实施例中，客户端10执行加速程序6120和收集代理404。

[0894] 在一些实施例中，客户端10从服务器30获得客户端560、流客户端562和/或收集代理404。在其他实施例中，客户端10从设备1250获得客户端560、流客户端562和/或收集代理404。在一个实施例中，任何客户端560、流客户端562和/或收集代理404可存储在设备1250上。例如，在一些实施例中，客户端560、流客户端562和/或收集代理404可被高速缓存在设备1250中。在其他实施例中，一旦设备1250确定应用可被加速，则设备1250可向客户端10传输客户端560、流客户端562、加速程序6120和/或收集代理404。在一些实施例中，客户端10可自动安装和执行任何的客户端560、流客户端562、加速程序6120和/或收集代理404。在又一个实施例中，任何的客户端560、流客户端562、加速程序6120和/或收集代理404可对客户端的用户或应用或者客户端网络堆栈的任何部分透明的执行。

[0895] 在一些实施例中，设备1250为客户端10建立到服务器30或网络40’的VPN或SSL VPN连接。在其他实施例中，设备1250作为代理、访问服务器或负载平衡器，以便提供对一个或多个服务器30的访问。在一个实施例中，设备1250和/或加速程序6120加速流客户端562、收集代理404和/或客户端代理560到客户端10的传送。在一个实施例中，设备1250加速到客户端10的加速程序6120的传送。在其他实施例中，设备1250和/或加速程序6120加速计算环境15、应用、和/或数据文件到客户端10的传送。在一个实施例中，客户端10拥有计算环境15，并且设备1250和/或加速程序6120加速应用和/或数据文件的传送。在一个实施例中，设备1250和/或加速程序6120加速应用的传送。在另一个实施例中，设备1250和/或加速程序6120加速数据文件的传送。在又一个实施例中，设备1250和/或加速程序6120加速计算环境15的传送，例如此处在前描述的执行环境或虚拟执行环境。

[0896] 在一个实施例中，设备1250使用收集代理404收集的信息以便确定是否计算环境15、应用和/或数据文件可被加速。在一些实施例中，设备1250的策略引擎包括策略引擎406。在其他实施例中，设备1250与策略引擎406通信或接口以便确定远程用户或远程客户端10的验证和/或授权，以访问来自服务器30的计算环境15、应用和/或数据文件。在另一个实施例中，设备1250与策略引擎406通信或接口，以便确定远程用户或远程客户端10的验证

和/或授权，使得应用传送系统500传送一个或多个计算环境15、应用和/或数据文件。在又一个实施例中，基于远程用户或远程客户端10对策略引擎的验证和/或授权，设备1250建立VPN或SSL VPN连接。在一个实施例中，基于策略引擎406的策略，设备1250控制网络话务量和通信会话的流量。例如，基于策略引擎406，设备1250可控制对计算环境15、应用或数据文件的访问。

[0897] 现在看图49B，描述了向远程位置客户端的远程用户加速传送计算环境的方法的实施例。方法8000总的来说，在步骤8005，服务器30接收执行客户端10上的应用的请求。在步骤8010，服务器30向客户端10流式传输用于执行的应用。在步骤8015，设备1250和/或客户端侧加速程序6120加速到客户端10的应用的传送或传输。在步骤8020，客户端10或应用请求被应用使用的来自服务器30的数据文件。在步骤8025，服务器30和/或设备1250向客户端10传输数据文件。在步骤8030，设备1250和/或客户端侧加速程序6120加速数据文件到客户端10的传送或传输。

[0898] 在进一步的细节中，在步骤8005，服务器30接收请求以执行客户端10上的应用。在一些实施例中，客户端10的用户作出请求。在其他实施例中，应用、操作系统或计算环境15传输该请求。在另一个实施例中，设备1250拦截来自客户端10的请求并向服务器30转发该请求。在一个实施例中，基于用户或客户端10的验证和/或授权，设备1250向服务器30转发该请求。在另一个实施例中，基于收集代理404提供的信息，设备1250向服务器30转发该请求。在一个实施例中，请求包括通过多个执行方法中的一个方法执行应用的请求。例如，客户端10的用户可请求执行应用为从服务器流式传输的应用、本地安装和执行的应用、或服务器30上执行并向客户端10远程显示的基于服务器的应用。在一些实施例中，请求基于文件类型关联。例如，用户可选择与应用相关联的文件，该应用用于读出或访问文件。

[0899] 在步骤8010，响应于步骤8005的请求，服务器30向客户端10传输用于执行的应用。在一些实施例中，服务器流式传输应用到客户端10。例如，在一些实施例中，通过流式传输应用，应用不用安装而在客户端10上执行。在其他实施例中，服务器30向客户端10传输应用用于本地安装和执行。例如，使用结合C部分中的加速程序6120描述的自动安装和执行技术，在接收到应用时，客户端10可自动安装和执行应用。在另一个实施例中，服务器30代表客户端在服务器30上执行应用，并通过远程显示或表示层协议向客户端10传输显示。在又一个实施例中，设备1250向客户端10流式传输应用或者向客户端10传输用于安装和/或执行的应用。在一些实施例中，设备1250和/或服务器30传输包括应用的计算环境15。在其他实施例中，设备1250和/或服务器30响应于请求来传输计算环境15。

[0900] 在步骤8015，设备1250和/或加速程序6120加速用于执行的应用到客户端10的传送。在一个实施例中，设备1250执行或应用在以上C部分中描述的一个或多个加速技术。在另一个实施例中，加速程序6120执行或应用在以上C部分中描述的多个客户端侧加速程序的一个或多个。在一些实施例中，加速程序1250和设备6120共同工作或彼此协同工作，以便在客户端10和设备1250二者上执行多个加速程序。例如，加速程序6120可执行一个或多个加速技术的第一集合，而设备1250执行一个或多个加速技术的第二集合。在一个实施例中，加速程序1250和设备6120执行相同的加速技术。在另一个实施例中，加速程序1250和设备6120执行不同的加速程序。

[0901] 在一个实施例中，设备1250和/或加速程序6120加速在客户端10和服务器30之间

通过传输层连接传输的任何有效负载。在一些实施例中，服务器30通过传输层连接流式传输作为一个或多个数据文件的应用，例如TCP/IP包的有效负载。在其他实施例中，服务器30通过应用层协议或在传输层连接之上的流协议流式传输应用。在另一个实施例中，服务器30通过传输层连接经ICA或RDP协议传输显示输出。在这些实施例的任意个中，设备1250和/或加速程序6120通过传输层包的有效负载加速应用的传送。

[0902] 在步骤8020，客户端10传输对于设备或计算环境15使用的数据文件的请求。在一些实施例中，对于数据文件的请求与请求一起被传输以便在步骤8005中执行应用。在一个实施例中，执行应用的请求包括对于数据文件的请求。在其他实施例中，应用或计算环境请求在执行应用或计算环境的任何功能、操作、或逻辑过程中的数据文件。例如，应用或计算环境15可请求来自服务器30的任何宏指令、脚本、配置数据、配置文件、模板或规则。在一些实施例中，应用请求作为应用的背景处理或任务的数据文件。在一个实施例中，应用或计算环境15的用户请求数据文件读取、访问或以其他方式处理带有应用或计算环境的文件。例如，用户可通过应用打开文件用于编辑，例如通过字处理应用打开文本用于编辑。在一些实施例中，用户将文件拖曳并放入计算环境的应用中以便请求数据文件。在其他实施例中，用户可通过文件和目录接口，例如在Windows操作系统中的文件浏览器，请求数据文件，放置于网络化的或远程的存储系统中，例如中心服务器的网络驱动器。

[0903] 在步骤8025，服务器30或设备1250向客户端10传输被请求的数据文件。在一些实施例中，服务器30或设备1250响应步骤8020的请求，向客户端10传输数据文件。在其他实施例中，在没有来自客户端10的请求时，服务器30或设备1250向客户端10传输数据文件。例如，服务器30可向客户端10“推入”数据文件的更新。在一个实施例中，服务器30向客户端10传输被请求的数据文件。在另一个实施例中，设备1250向客户端10传输被请求的数据文件。例如，在一个实施例中，设备1250拦截对于数据文件的请求，检查用于数据文件的设备1250的高速缓存，并向客户端10传输被高速缓存的数据文件。在又一个实施例中，加速程序6120在客户端10拦截数据文件请求并通过加速程序6120的高速缓存向客户端10提供数据文件。在一些实施例中，设备1250或服务器30通过流协议或流来传输数据文件。在其他实施例中，设备1250或服务器30通过任何类型和形式的高速缓存协议来传输数据文件。

[0904] 在步骤8030，设备1250和/或加速程序6120加速数据文件到客户端10的传送或传输。在一些实施例中，数据文件可通过任何类型和形式的协议传输，例如传输层协议之上的应用层协议。在一个实施例中，设备1250加速数据文件的传输。在另一个实施例中，加速程序6120加速数据文件的传输。在一些实施例中，设备1250与加速程序1250协同加速数据文件的传输。如此处所描述的，设备1250和/或加速程序6120可执行一个或多个在客户端10和设备30上的加速技术，以便加速数据文件的传输。在一些实施例中，设备1250和/或加速程序6120可高速缓存客户端10或设备1250上的一个或多个数据文件以供应用或计算环境15使用。

[0905] 代表性实例

[0906] 作为示范性实施例，用户位于工作在本地机器10上的分支机构。用户可能期望使用例如MICROSOFT Word的字处理应用来编辑公司文件，二者都驻留在位于中心办公室的远程机器30上。用户可接着导航web浏览器到远程机器30寄载的公司web站点。一旦用户被远程机器30验证，远程机器30可准备并向本地机器10传输HTML页，该HTML页包括如此处在图

3A和3B中描述的程序邻近窗口，在图3A和图3B中出现的图标代表本地机器10已经访问的应用程序。本地机器10的用户通过点击图标可调用应用的执行。如在图4A－4D中描述的策略引擎可接着确定本地机器10是否以及怎样访问字处理应用。使用在图20－21中描述的技术，应用程序可接着被本地安装和执行。用户可接着使用应用选来择在远程机器30上的文本用于编辑。使用此处描述的任何技术，例如TCP多路复用，设备1250可接着加速文件到本地机器10的传送。

[0907] 作为另一个实施例，第二用户可以位于工作在本地机器10上的分支机构。用户可能期望通过用户的公司帐户访问包含附件的电子邮件。电子邮件应用和电子邮件数据文件可驻留在中心办公室。一旦用户请求访问电子邮件应用，如图4A-4D中描述的策略引擎使用此处描述的流技术可确定流式传输到用户的电子邮件应用。策略引擎也可如此处描述的在本地机器10上安装来决定安装加速程序。使用此处描述的技术，例如动态高速缓存，应用流可被加速。一旦本地安装，用户可接着选择电子邮件和相应的附件查看。设备1250可通过使用加速技术，例如此处描述的TCP池化，加速文件的传送。设备也可高速缓存被传送到远程机器的一些或所有数据文件，以便加速稍后的请求。高速缓存可与加速程序结合在设备1250或者在本地机器10上实现。

[0908] 作为第三个实施例，位于分支机构的用户可能希望访问电子数据表程序(例如MICROSOFT Excel)来更新电子数据表。用户可使用本地机器10与位于中心办公室的远程机器30建立SSL连接，并如图3A和3B所描述的从程序邻近选择电子数据表应用。如图4D所描述的收集代理可接着收集关于本地机器的信息以便确定是否电子数据表应用可被流式传输到本地机器10。电子数据表应用可接着通过SSL连接被流式传输到本地机器10。SSL连接可通过此处描述的提供SSL或TCP连接池化和多路复用技术的设备1250加速。应用流也可被此处描述的提供任何动态高速缓存技术的设备1250加速。用户可接着从电子数据表应用中选择文件用于编辑。本地机器10可向远程机器传输对于文件的请求。设备1250可接着使用此处描述的压缩技术来加速文件到用户的传送。

[0909] 虽然上述一般性描述了加速计算环境到客户端的传送的应用传送系统和设备，但是应用传送系统和设备可以加速多个计算环境、应用和/或数据文件到客户端的传送。例如，应用传送系统和设备可加速与一种类型的操作系统相关联的第一计算环境到客户端的传送，以及与第二种类型的操作系统相关联的第二计算环境到客户端的传送。另外，应用传送系统和设备可加速计算环境、应用、和/或数据文件到多个客户端的传送。另外，虽然上述一般性描述了加速计算环境到远程客户或远程客户端的传送的应用传送系统和设备，但是应用传送系统和设备可以加速计算环境、应用和/或数据文件到任何本地、远程客户端或其他客户端的传送，例如在服务器的LAN上的客户端。

[0910] 此外，虽然上述一般性描述设备在客户端和应用传送系统之间，但是在一个或多个客户端和一个或多个服务器之间可使用多个设备。在一些实施例中，第一设备驻留在客户端的网络上，并且第二设备驻留在服务器网络上。在一个实施例中，第一设备和第二设备在此处描述的操作的执行中彼此通信。例如，第一设备和第二设备可通过中间节点、高性能或设备到设备通信协议来通信。应用传送系统和设备可被布置于多种类的网络环境和基础结构中。

[0911] 实施例可作为一个或多个计算机可读程序提供，这些计算机可读程序嵌入到一个

或者多个制造产品中。产品可为软盘、硬盘、紧致磁盘、数字多用途盘、闪烁存储卡、PROM、RAM、ROM或磁带。通常，计算机可读程序可以任何编程语言实现。可使用的语言的一些实例包括C、C++、C#、或JAVA。软件程序可以作为对象代码存储在一个或多个产品中。

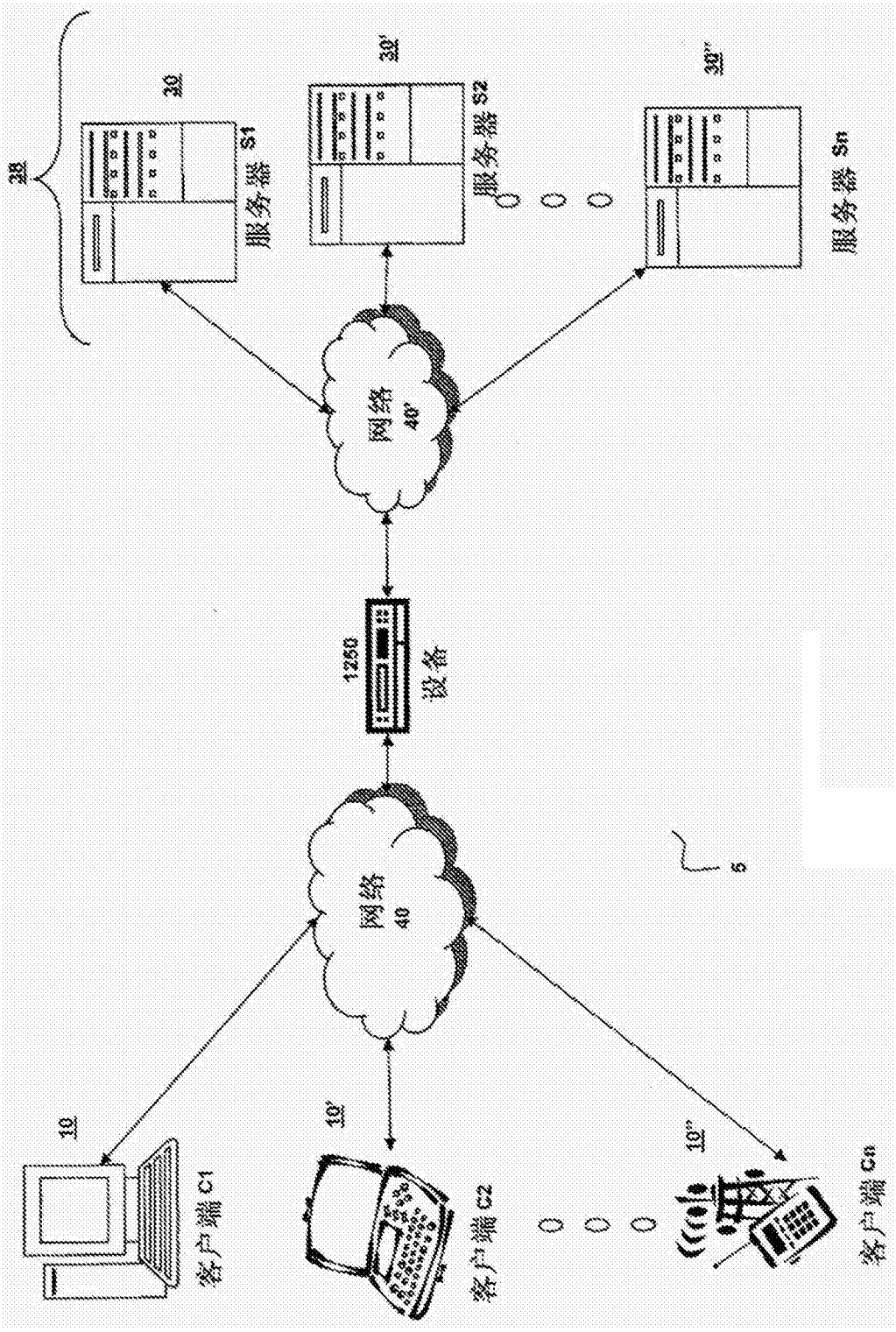

38
30
服务器 S1
30'
服务器 S2
30''
服务器 Sn
网络
40'
1250
设备
网络
40
5
10
客户端 C1
10'
客户端 C2
10''
客户端 Cn

图1A

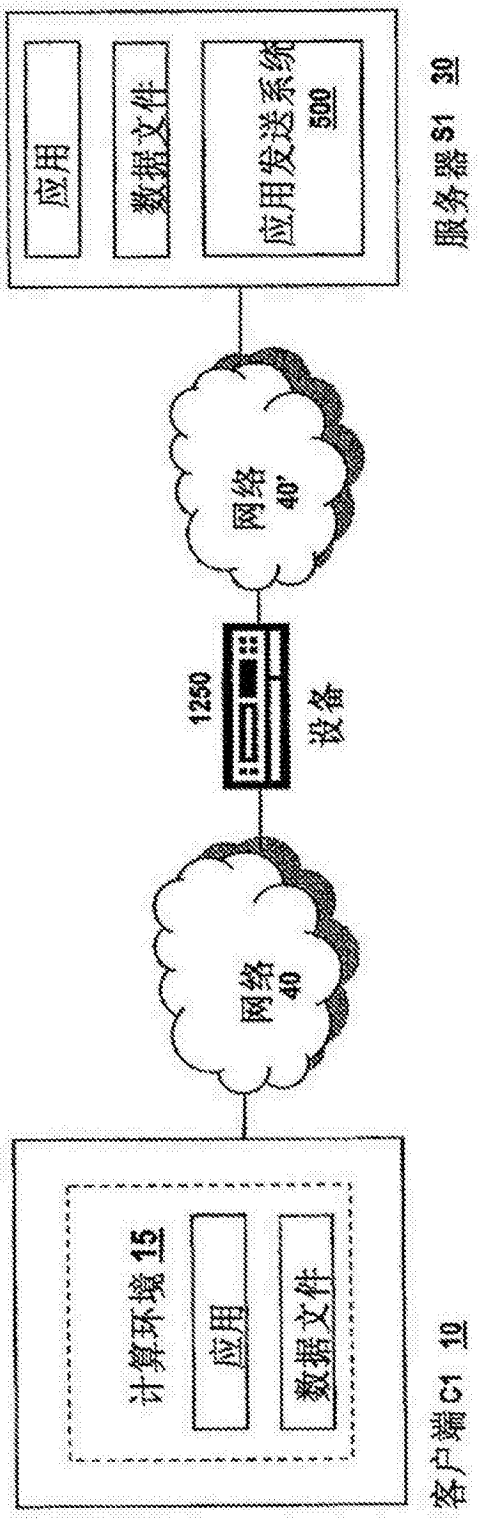

应用
数据文件
应用发送系统
500
服务器 S1 30
网络
40'
1250
设备
网络
40
计算环境 15
应用
数据文件
客户端 C1 10

图1B

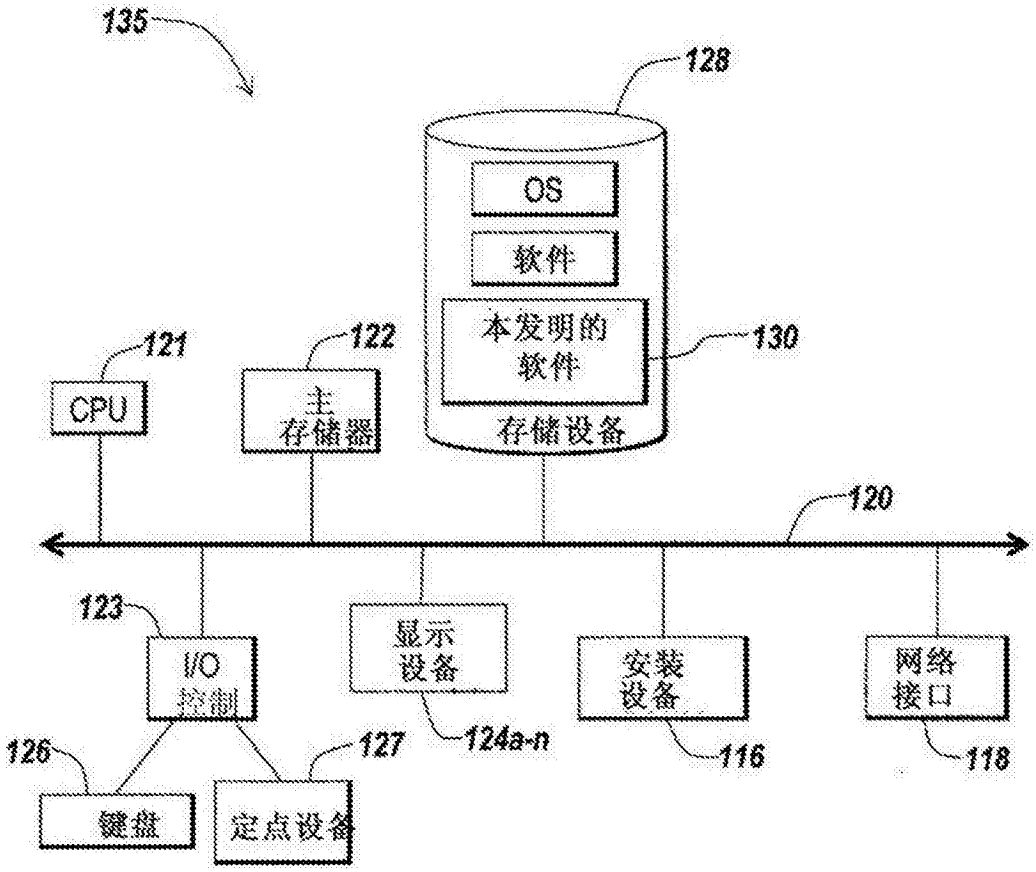

135
128
OS
软件
本发明的软件
130
存储设备
121
CPU
122
主存储器
120
123
I/O控制
显示设备
124a-n
安装设备
116
网络接口
118
126
键盘
127
定点设备

图1C

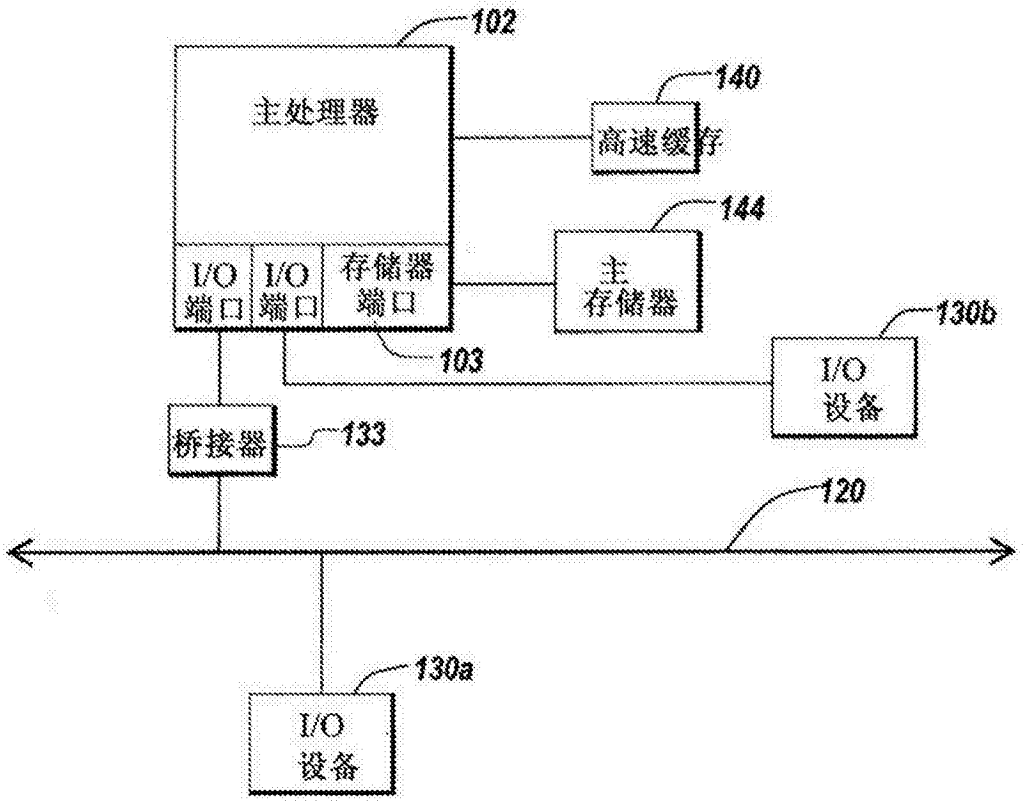

102
主处理器
140
高速缓存
144
I/O 端口
I/O 端口
存储器端口
主存储器
103
130b
I/O 设备
桥接器
133
120
130a
I/O 设备

图1D

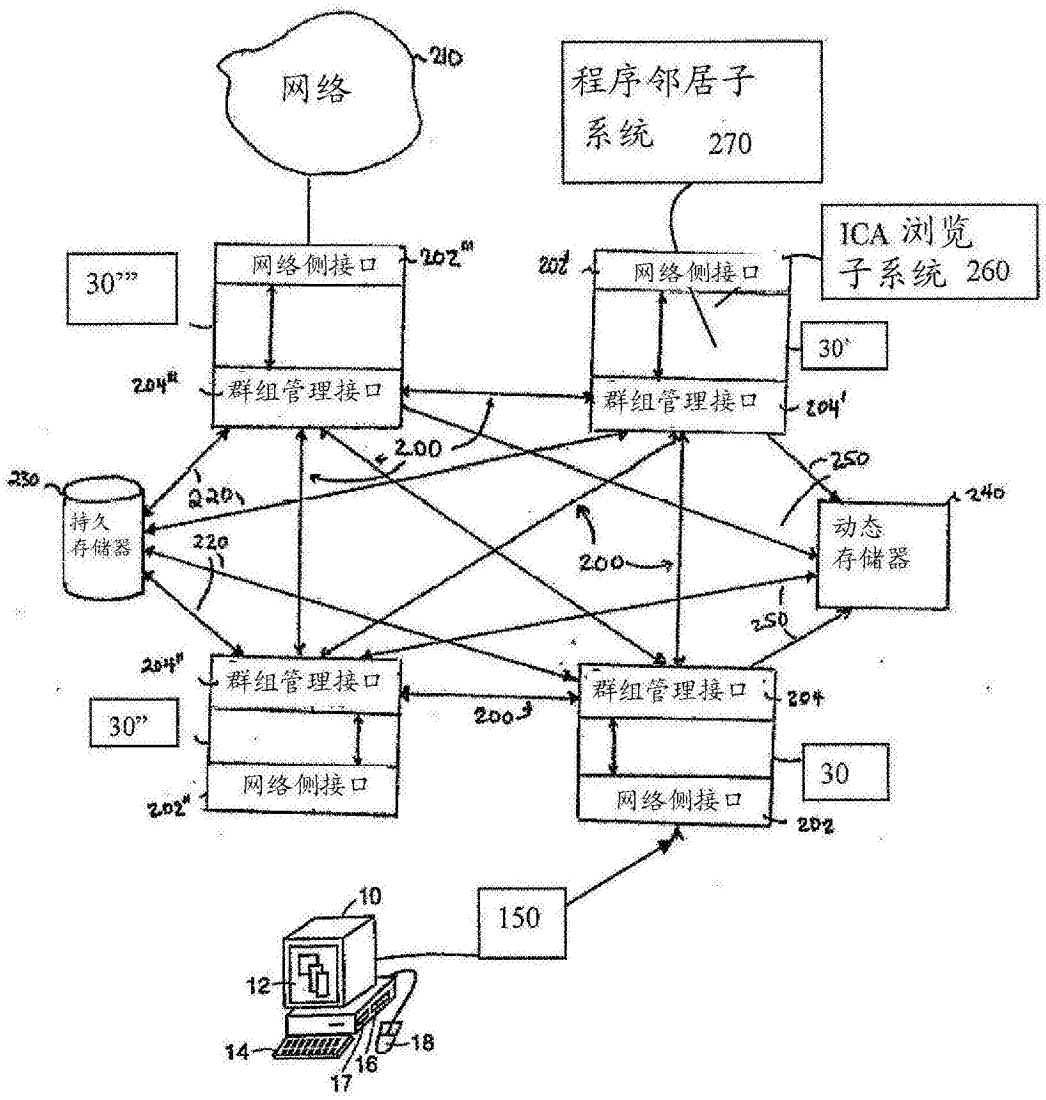

网络
210
程序邻居子系统 270
ICA 浏览子系统 260
30'''
网络侧接口
202'''
202'
网络侧接口
30'
群组管理接口
204''
群组管理接口
204'
200
230
持久存储器
220
250
240
动态存储器
200
250
群组管理接口
204''
群组管理接口
204
30''
200
网络侧接口
202''
30
网络侧接口
202
150
10
12
14
16
17
18

图1E

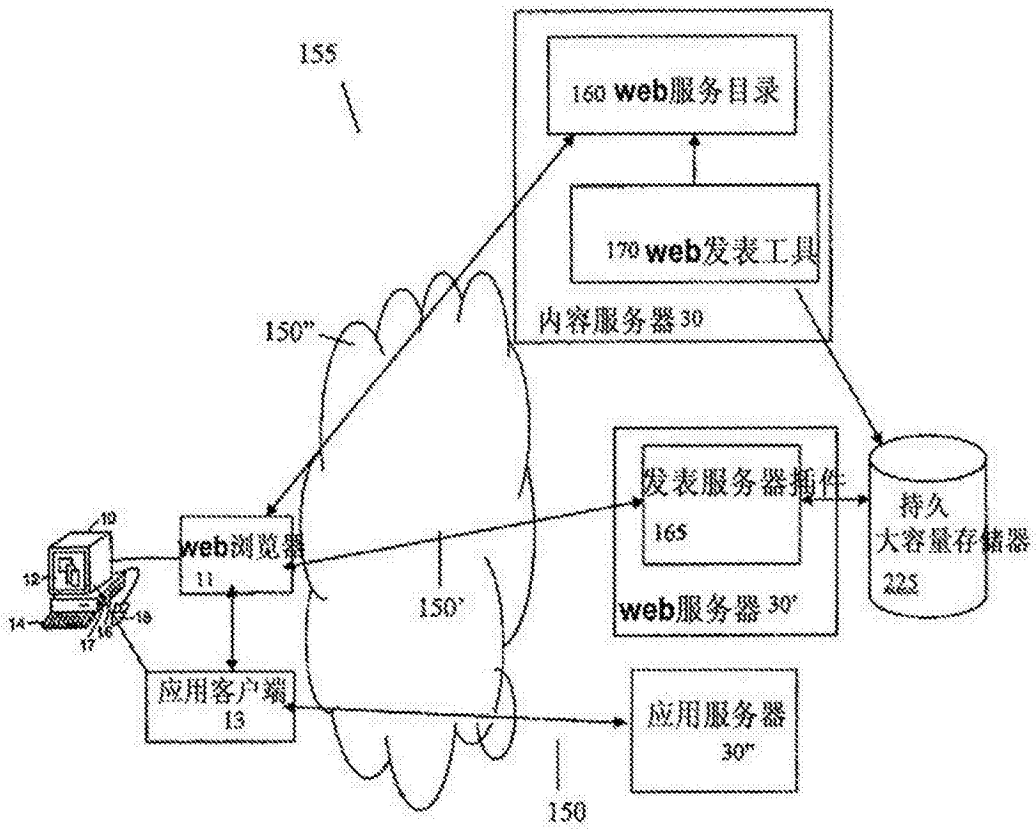

155
160 web服务目录
170 web发表工具
内容服务器 30
150"
发表服务器插件
165
持久
大容量存储器
225
web浏览器
11
150'
web服务器 30'
应用客户端
13
应用服务器
30"
150

图1F

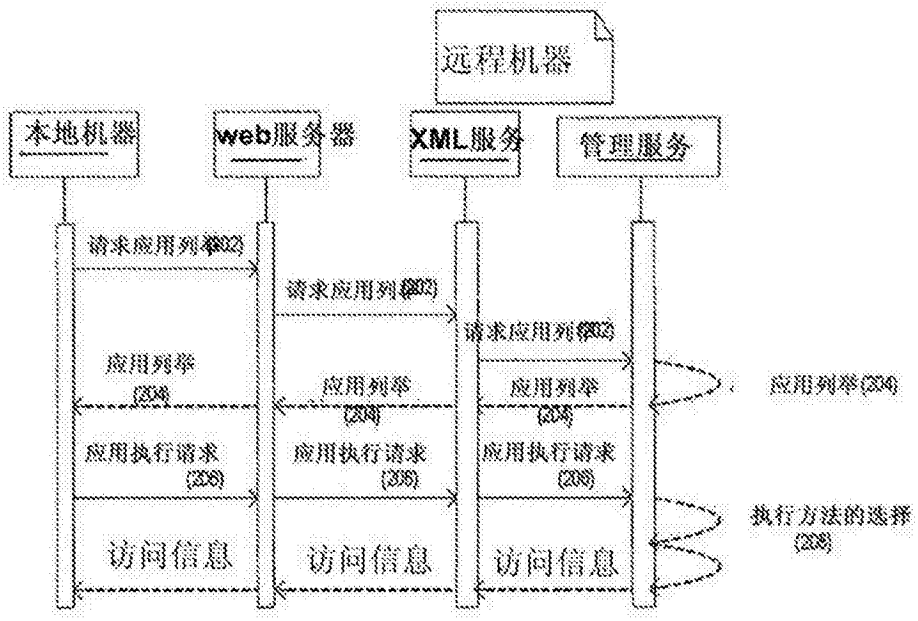

远程机器
本地机器
web服务器
XML服务
管理服务
请求应用列举(202)
请求应用列举(202)
请求应用列举(202)
应用列举(204)
应用列举(204)
应用列举(204)
应用列举(204)
应用执行请求(206)
应用执行请求(206)
应用执行请求(206)
执行方法的选择(208)
访问信息
访问信息
访问信息

图2

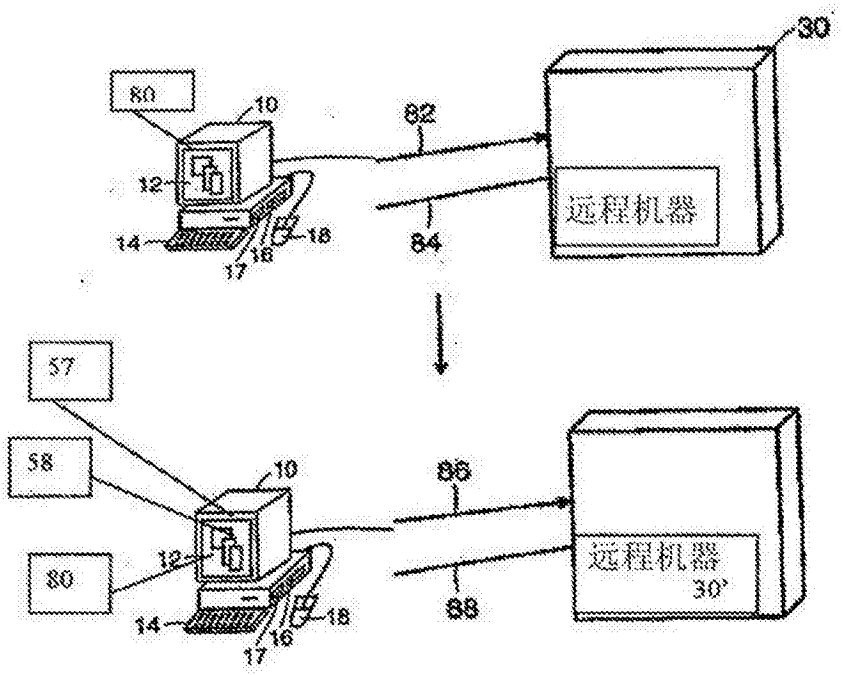

30
80
10
82
12
远程机器
14
16
18
17
84
57
58
10
86
12
80
远程机器
30'
14
16
18
17
88

图3A

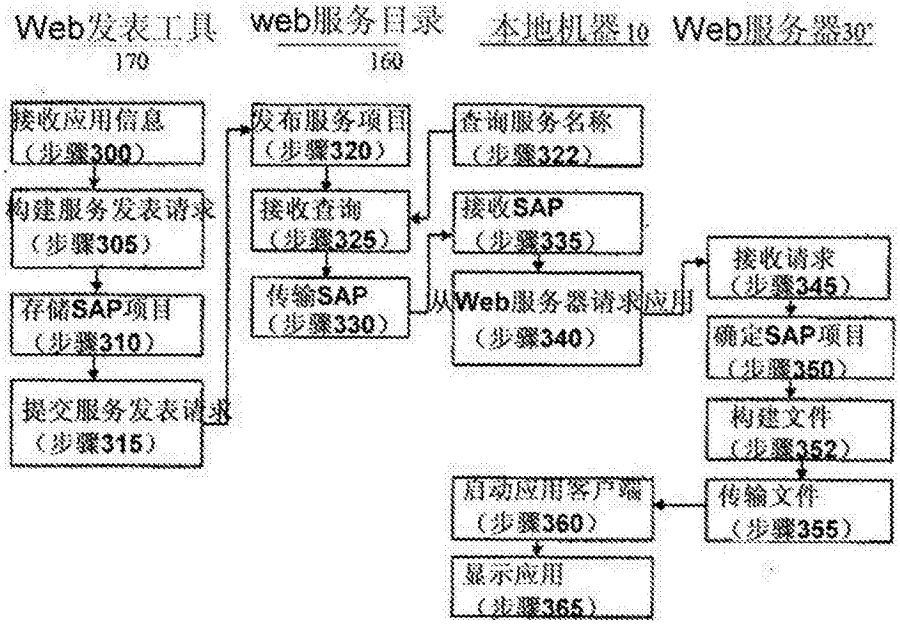

Web发表工具
170
web服务目录
160
本地机器10
Web服务器30'
接收应用信息
（步骤300）
构建服务发表请求
（步骤305）
存储SAP项目
（步骤310）
提交服务发表请求
（步骤315）
发布服务项目
（步骤320）
接收查询
（步骤325）
传输SAP
（步骤330）
查询服务名称
（步骤322）
接收SAP
（步骤335）
从Web服务器请求应用
（步骤340）
接收请求
（步骤345）
确定SAP项目
（步骤350）
构建文件
（步骤352）
传输文件
（步骤355）
启动应用客户端
（步骤360）
显示应用
（步骤365）

图3B

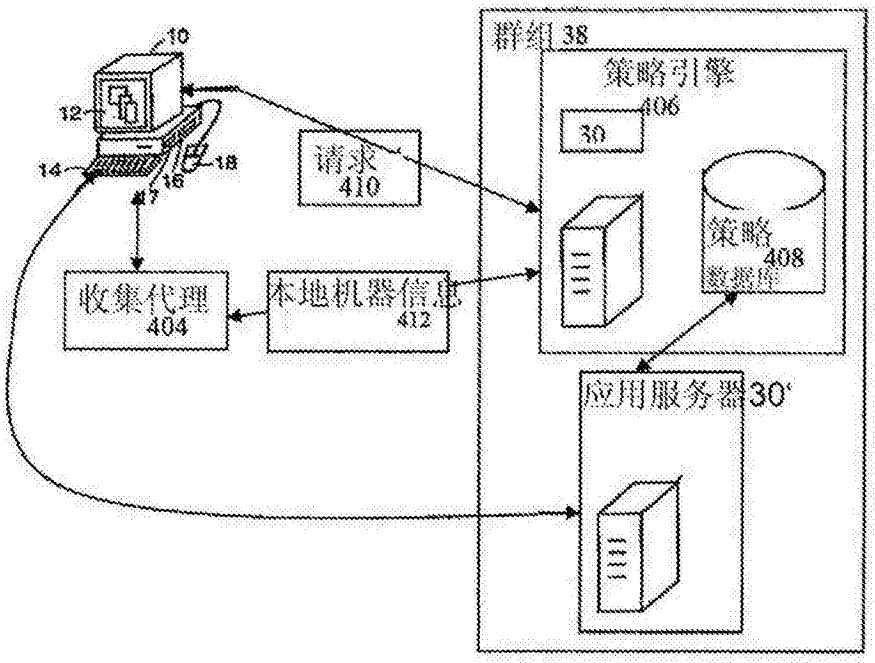

群组38
策略引擎
406
30
策略
408
数据库
请求
410
收集代理
404
本地机器信息
412
应用服务器30'
10
12
14
16
18

图4A

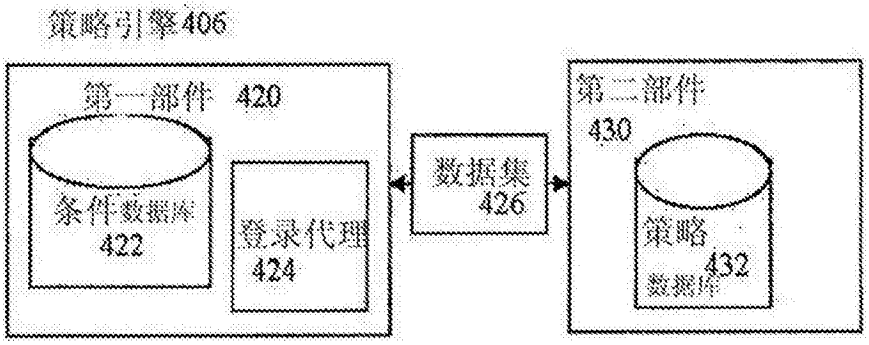

策略引擎406
第一部件 420
条件数据库
422
登录代理
424
数据集
426
第二部件
430
策略
数据库
432

图4B

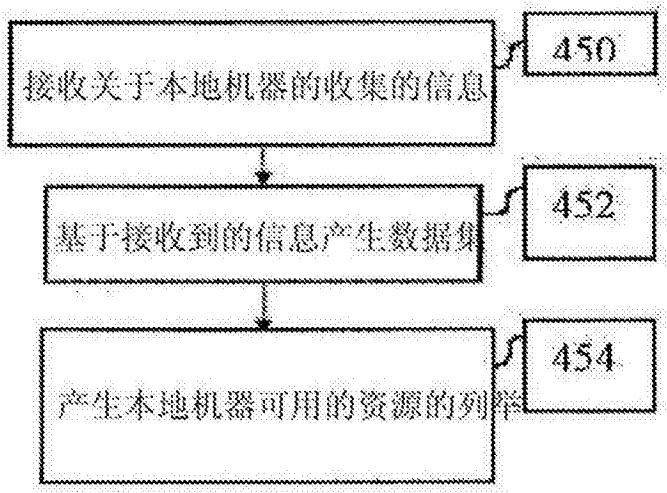

接收关于本地机器的收集的信息
450
基于接收到的信息产生数据集
452
产生本地机器可用的资源的列表
454

图4C

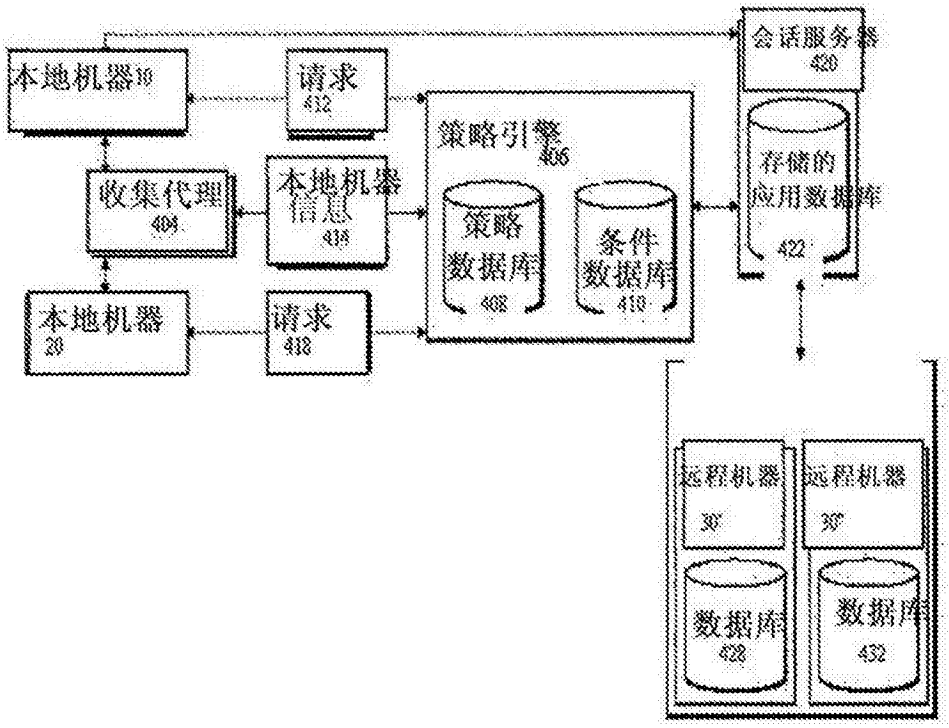

会话服务器
420
本地机器10
请求
412
策略引擎
406
存储的
应用数据库
422
收集代理
404
本地机器
信息
414
策略
数据库
408
条件
数据库
410
本地机器
20
请求
412
远程机器
30'
远程机器
30"
数据库
428
数据库
432

图4D

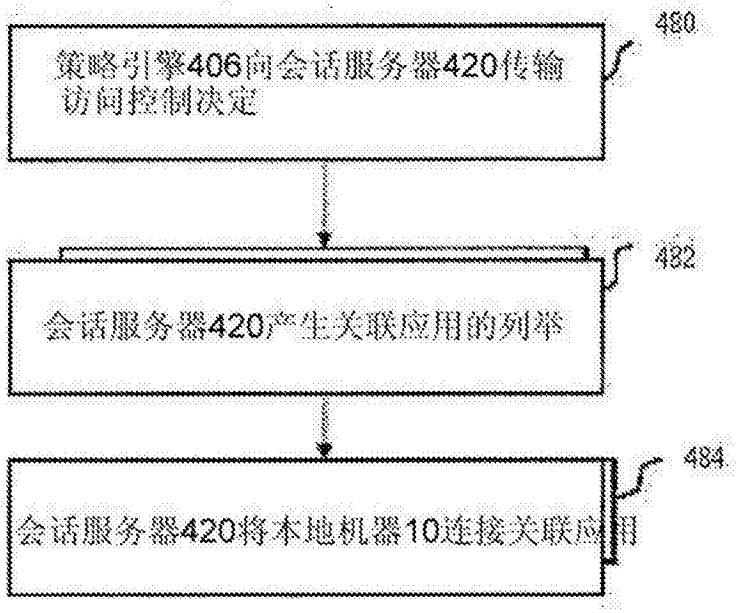

480
策略引擎406向会话服务器420传输访问控制决定
482
会话服务器420产生关联应用的列举
484
会话服务器420将本地机器10连接关联应用

图4E

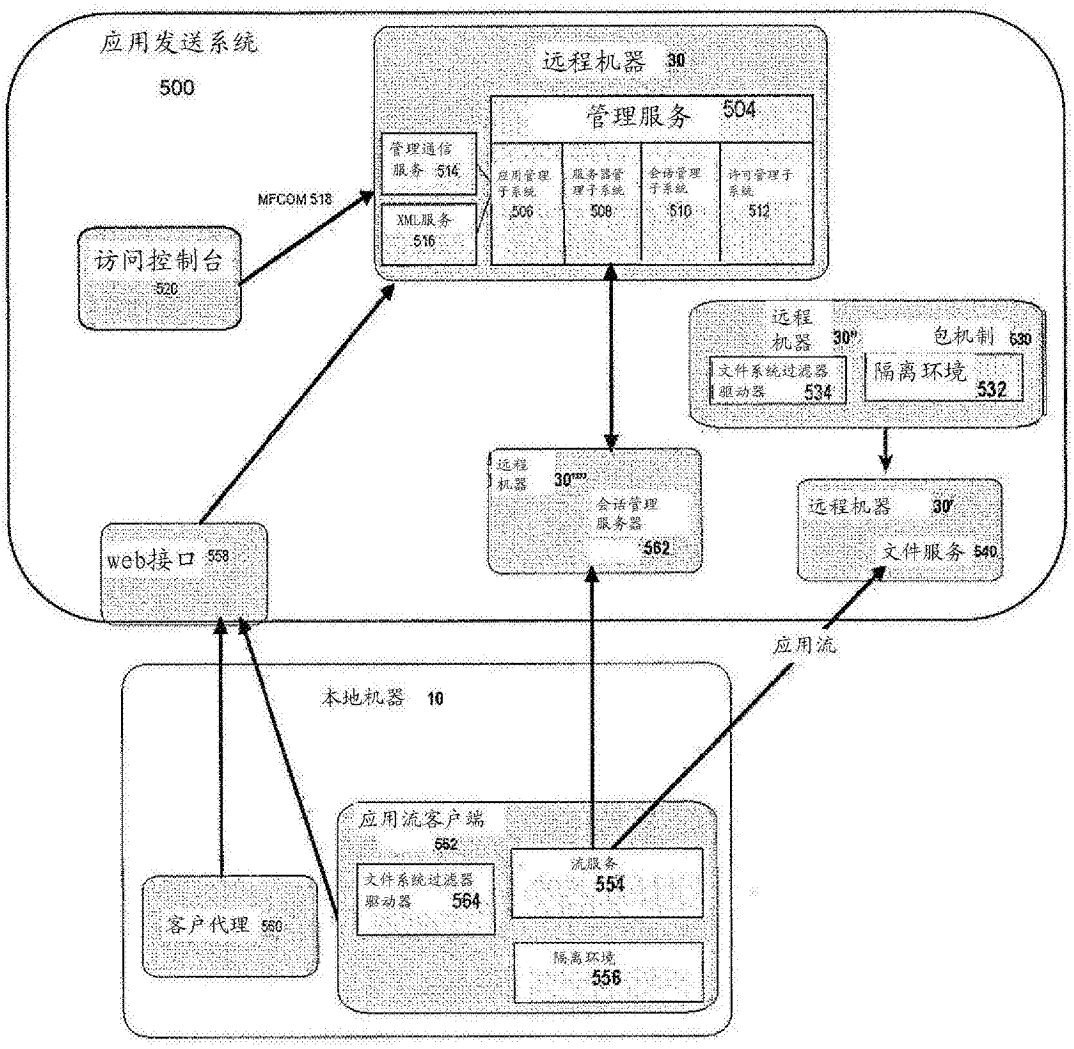

应用发送系统
500
远程机器 30
管理服务 504
管理通信服务 514
XML服务 516
应用管理子系统 506
服务器管理子系统 508
会话管理子系统 510
许可管理子系统 512
MFCOM 518
访问控制台 520
远程机器 30''
包机制 530
文件系统过滤器驱动器 534
隔离环境 532
远程机器 30'''
会话管理服务器 562
远程机器 30'
文件服务 540
web接口 558
应用流
本地机器 10
应用流客户端 552
文件系统过滤器驱动器 564
流服务 554
隔离环境 556
客户代理 560

图5

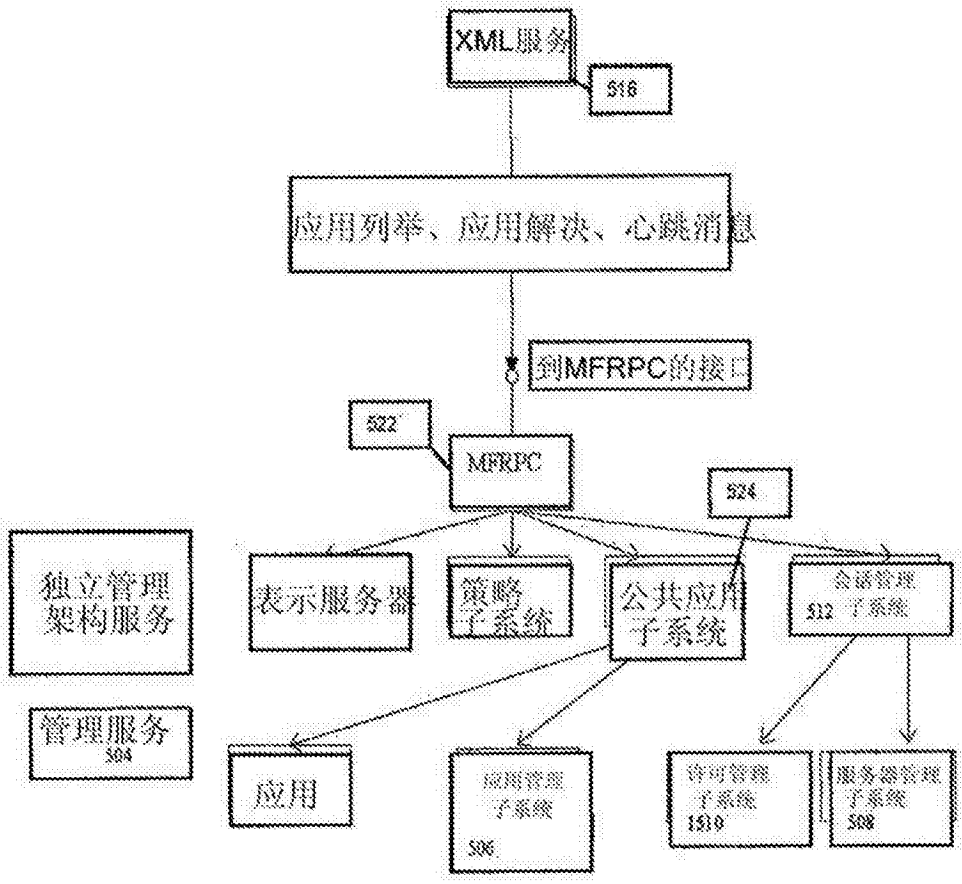

XML服务
516
应用列举、应用解决、心跳消息
到MFRPC的接口
522
MFRPC
524
独立管理架构服务
表示服务器
策略子系统
公共应用子系统
会话管理
512 子系统
管理服务
504
应用
应用管理子系统
506
许可管理子系统
510
服务器管理子系统
508

图6

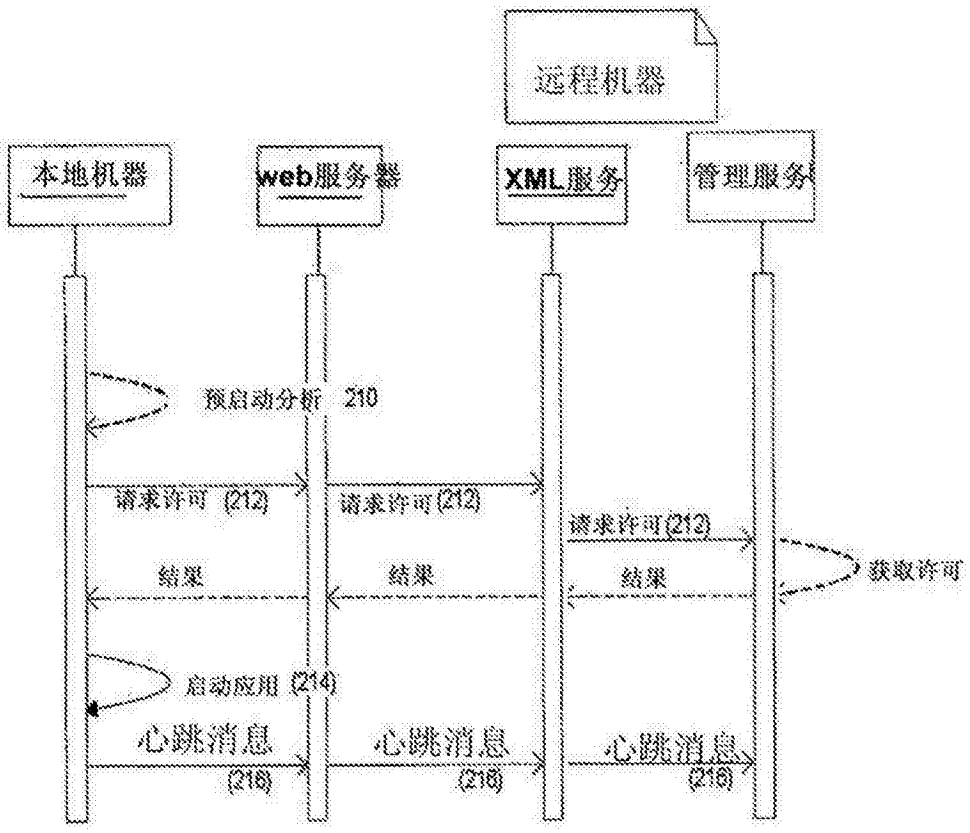

远程机器
本地机器
web服务器
XML服务
管理服务
预启动分析 210
请求许可 (212)
请求许可(212)
请求许可(212)
获取许可
结果
结果
结果
启动应用 (214)
心跳消息
(216)
心跳消息
(216)
心跳消息
(216)

图7

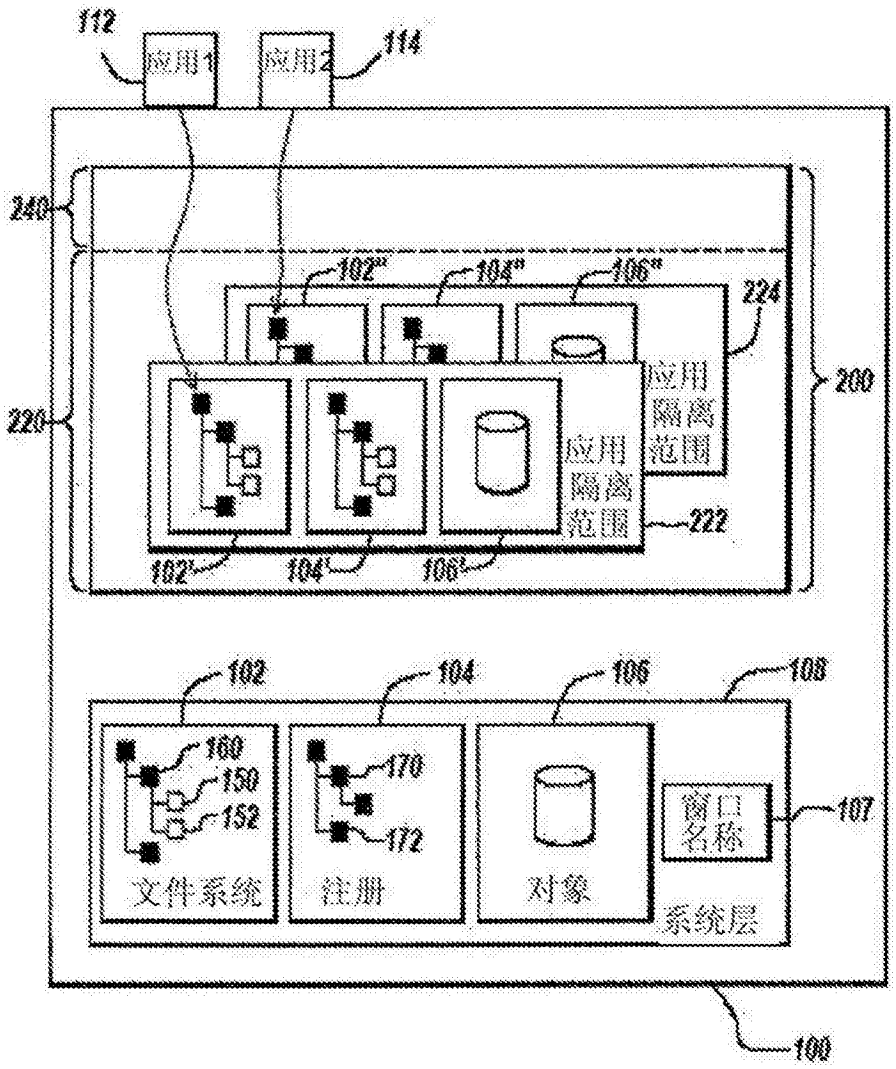

112
应用1
应用2
114
240
102"
104"
106"
224
220
200
应用隔离范围
应用隔离范围
222
102'
104'
106'
102
104
106
108
160
150
152
170
172
文件系统
注册
对象
窗口名称
107
系统层
100

图8A

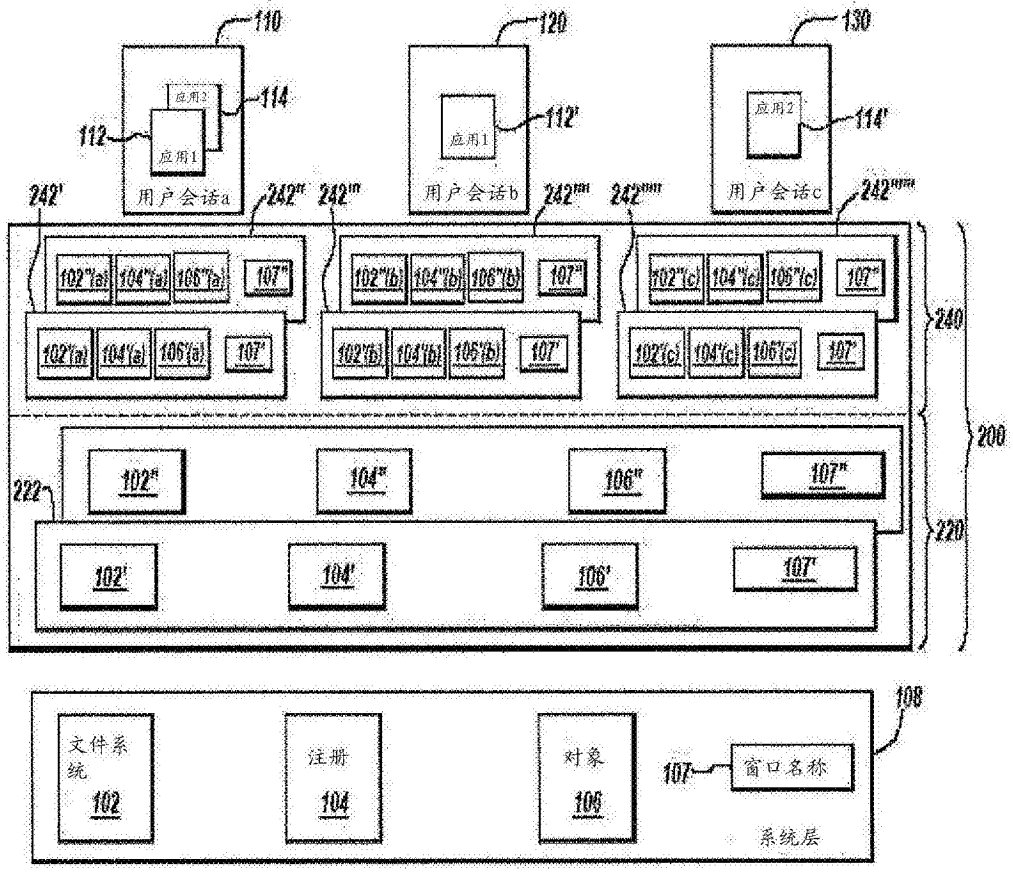

110
120
130
应用2
114
112
应用1
112'
应用2
114'
用户会话a
用户会话b
用户会话c
242'
242''
242'''
242''''
242'''''
242''''''
102''(a)
104''(a)
106''(a)
107''
102''(b)
104''(b)
106''(b)
107''
102''(c)
104''(c)
106''(c)
107''
102'(a)
104'(a)
106'(a)
107'
102'(b)
104'(b)
106'(b)
107'
102'(c)
104'(c)
106'(c)
107'
240
200
222
102''
104''
106''
107''
102'
104'
106'
107'
220
108
文件系统
102
注册
104
对象
106
107
窗口名称
系统层

图8B

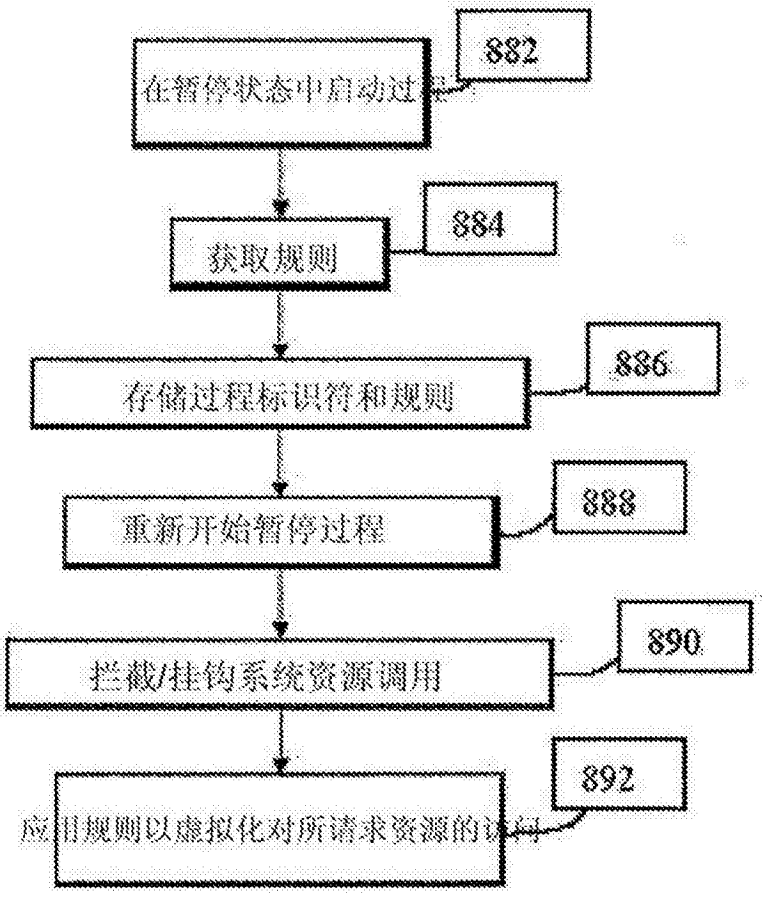

882
在暂停状态中启动过
884
获取规则
886
存储过程标识符和规则
888
重新开始暂停过程
890
拦截/挂钩系统资源调用
892
应用规则以虚拟化对所请求资源的

图8C

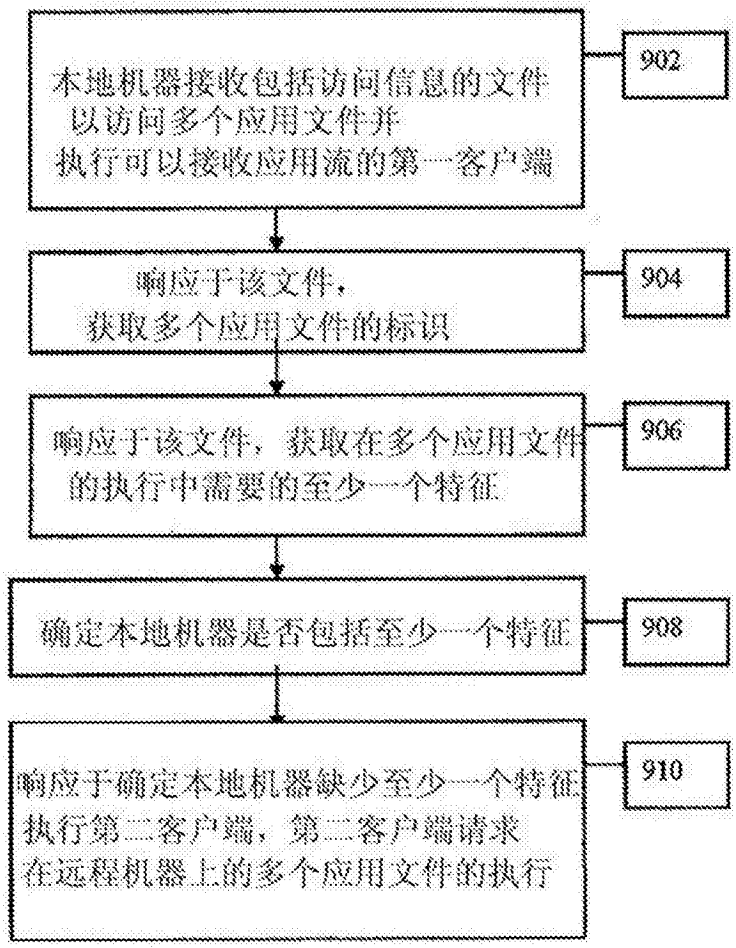

本地机器接收包括访问信息的文件
以访问多个应用文件并
执行可以接收应用流的第一客户端
902
响应于该文件，
获取多个应用文件的标识
904
响应于该文件，获取在多个应用文件
的执行中需要的至少一个特征
906
确定本地机器是否包括至少一个特征
908
响应于确定本地机器缺少至少一个特征
执行第二客户端，第二客户端请求
在远程机器上的多个应用文件的执行
910

图9

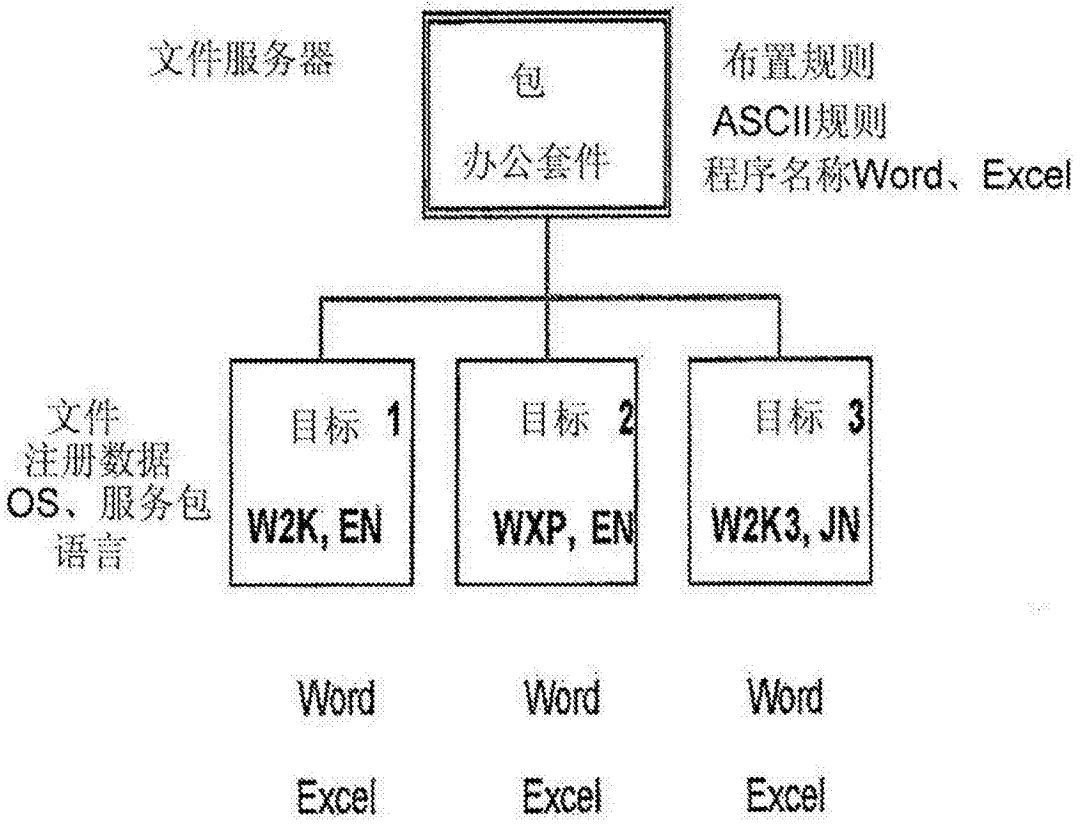

文件服务器
包
办公套件
布置规则
ASCII规则
程序名称Word、Excel
文件
注册数据
OS、服务包
语言
目标 1
W2K, EN
目标 2
WXP, EN
目标 3
W2K3, JN
Word
Excel
Word
Excel
Word
Excel

图10

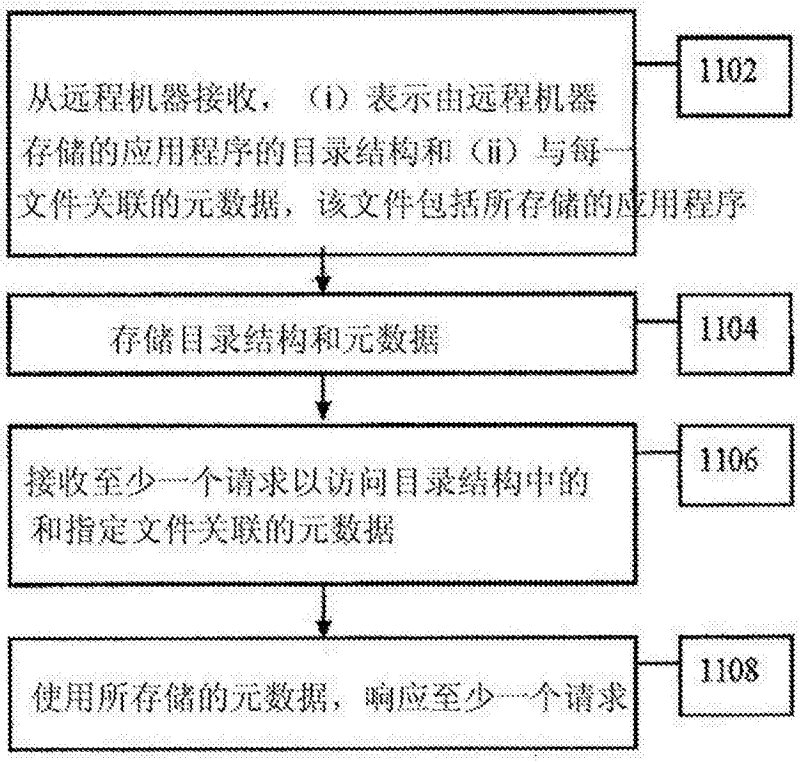

从远程机器接收，（i）表示由远程机器存储的应用程序的目录结构和（ii）与每一文件关联的元数据，该文件包括所存储的应用程序
1102
存储目录结构和元数据
1104
接收至少一个请求以访问目录结构中的和指定文件关联的元数据
1106
使用所存储的元数据，响应至少一个请求
1108

图11

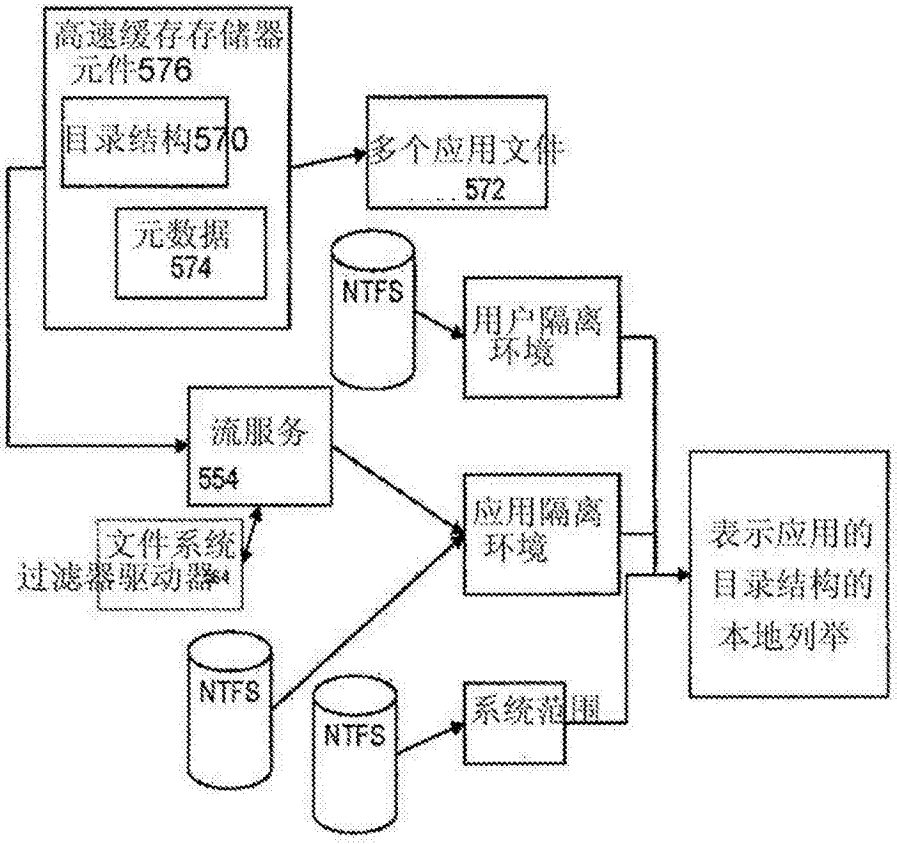

高速缓存存储器
元件576
目录结构570
元数据
574
多个应用文件
572
NTFS
用户隔离
环境
流服务
554
文件系统
过滤器驱动器
应用隔离
环境
表示应用的
目录结构的
本地列举
NTFS
NTFS
系统范围

图12

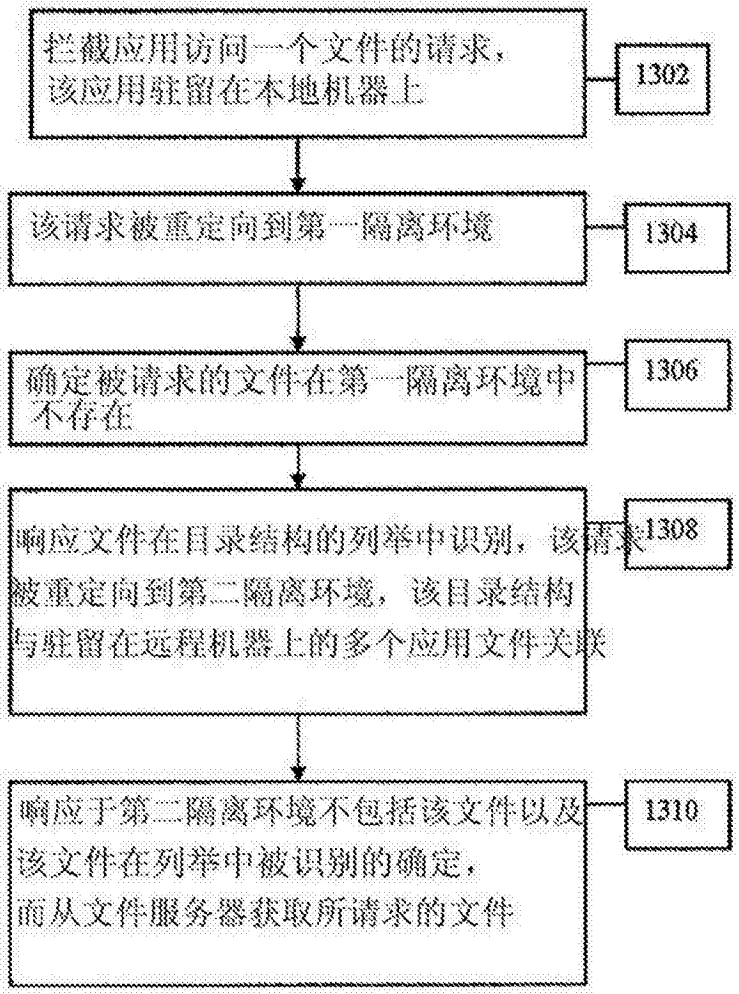

拦截应用访问一个文件的请求，该应用驻留在本地机器上
1302
该请求被重定向到第一隔离环境
1304
确定被请求的文件在第一隔离环境中不存在
1306
响应文件在目录结构的列举中识别，该请求被重定向到第二隔离环境，该目录结构与驻留在远程机器上的多个应用文件关联
1308
响应于第二隔离环境不包括该文件以及该文件在列举中被识别的确定，而从文件服务器获取所请求的文件
1310

图13

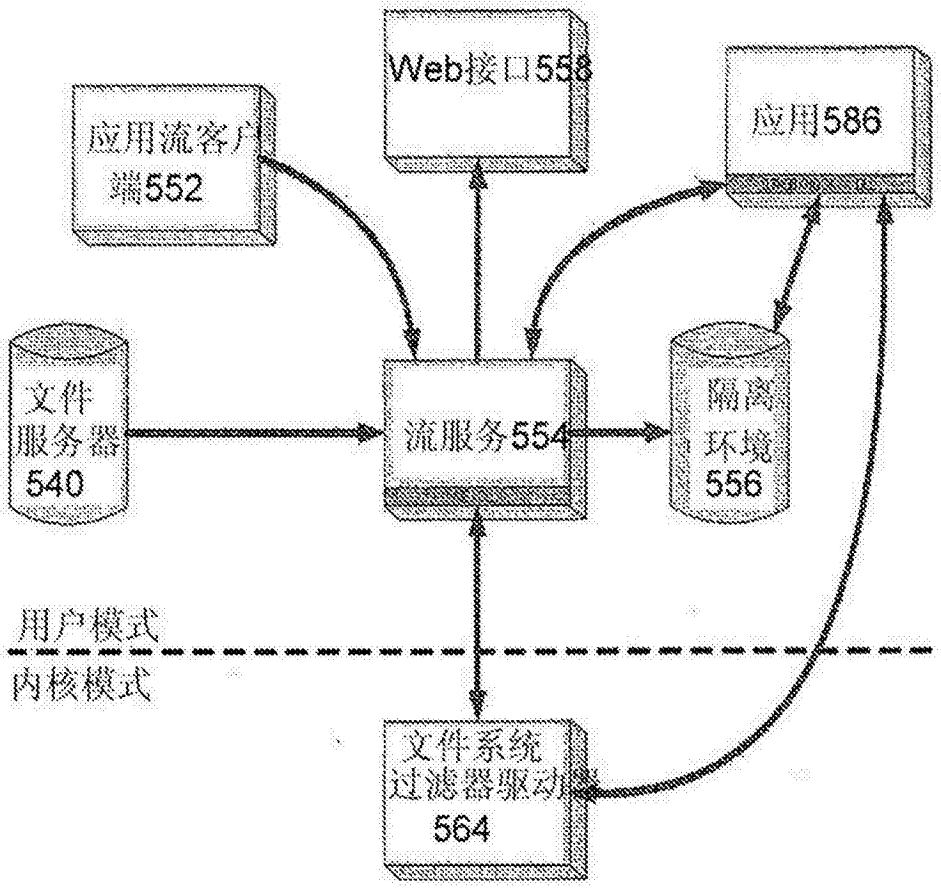

Web接口558
应用流客户端552
应用586
文件服务器540
流服务554
隔离环境556
用户模式
内核模式
文件系统过滤器驱动564

图14

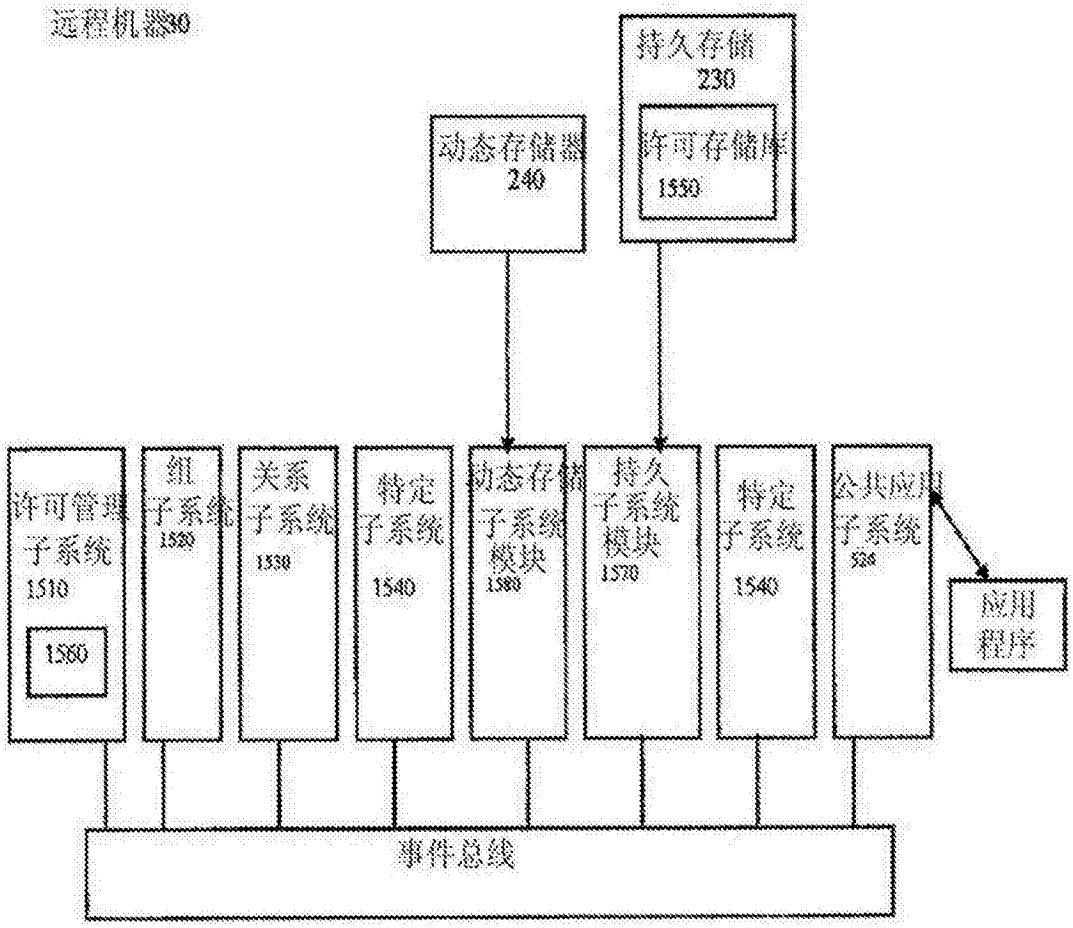

远程机器30
持久存储
230
动态存储器
240
许可存储库
1550
许可管理
子系统
1510
1560
组
子系统
1520
关系
子系统
1530
特定
子系统
1540
动态存储
子系统
模块
1580
持久
子系统
模块
1570
特定
子系统
1540
公共应用
子系统
应用
程序
事件总线

图15

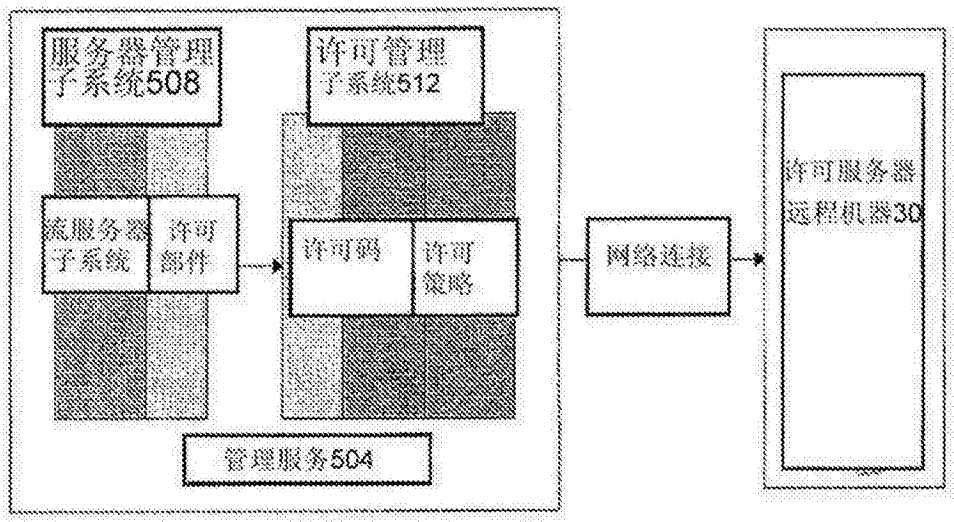

服务器管理
子系统508
许可管理
子系统512
流服务器
子系统
许可
部件
许可码
许可
策略
网络连接
许可服务器
远程机器30
管理服务504

图16

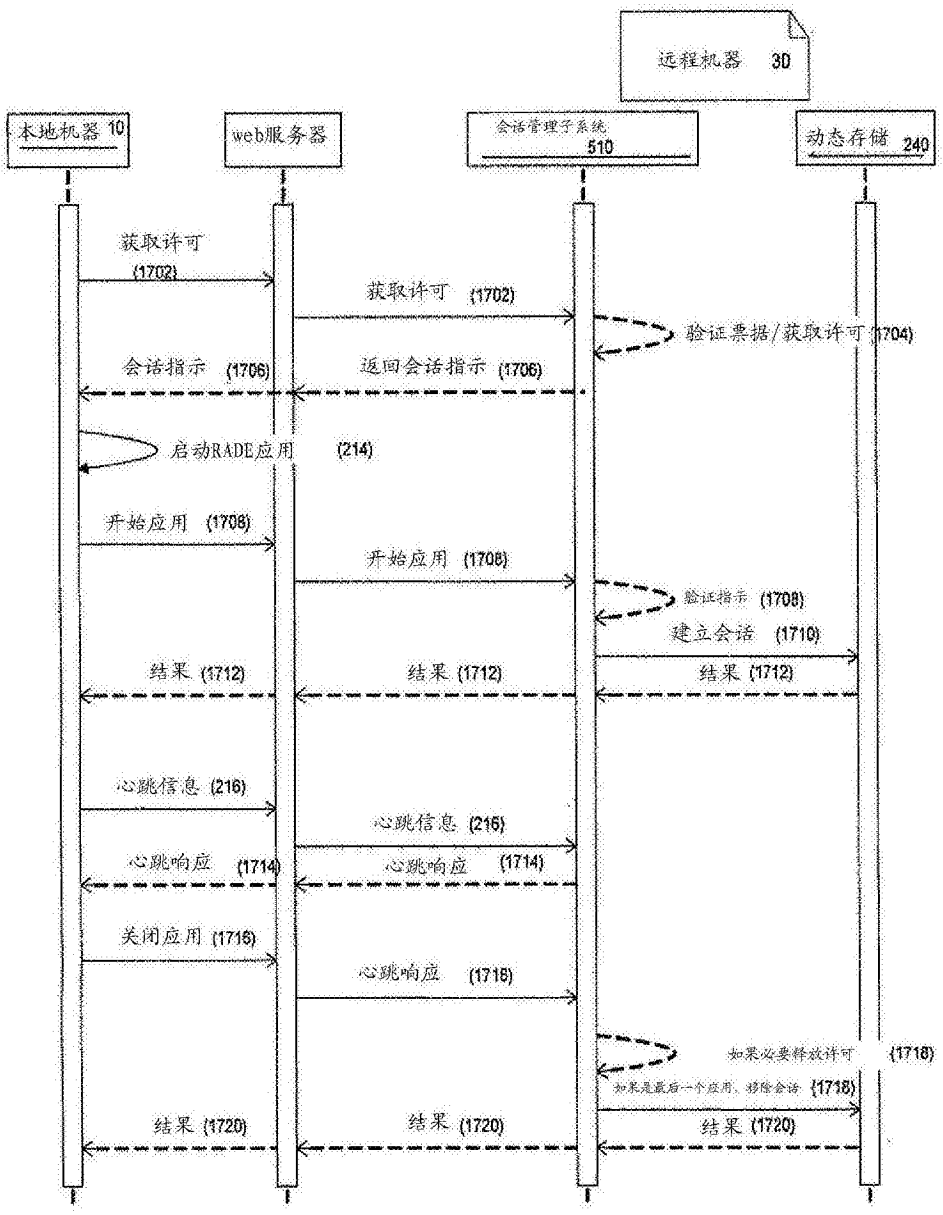

远程机器 30
本地机器 10
web服务器
会话管理子系统 510
动态存储 240
获取许可 (1702)
获取许可 (1702)
验证票据/获取许可 (1704)
会话指示 (1706)
返回会话指示 (1706)
启动RADE应用 (214)
开始应用 (1708)
开始应用 (1708)
验证指示 (1708)
建立会话 (1710)
结果 (1712)
结果 (1712)
结果 (1712)
心跳信息 (216)
心跳信息 (216)
心跳响应 (1714)
心跳响应 (1714)
关闭应用 (1716)
心跳响应 (1718)
如果必要释放许可 (1718)
如果是最后一个应用，移除会话 (1718)
结果 (1720)
结果 (1720)
结果 (1720)

图17

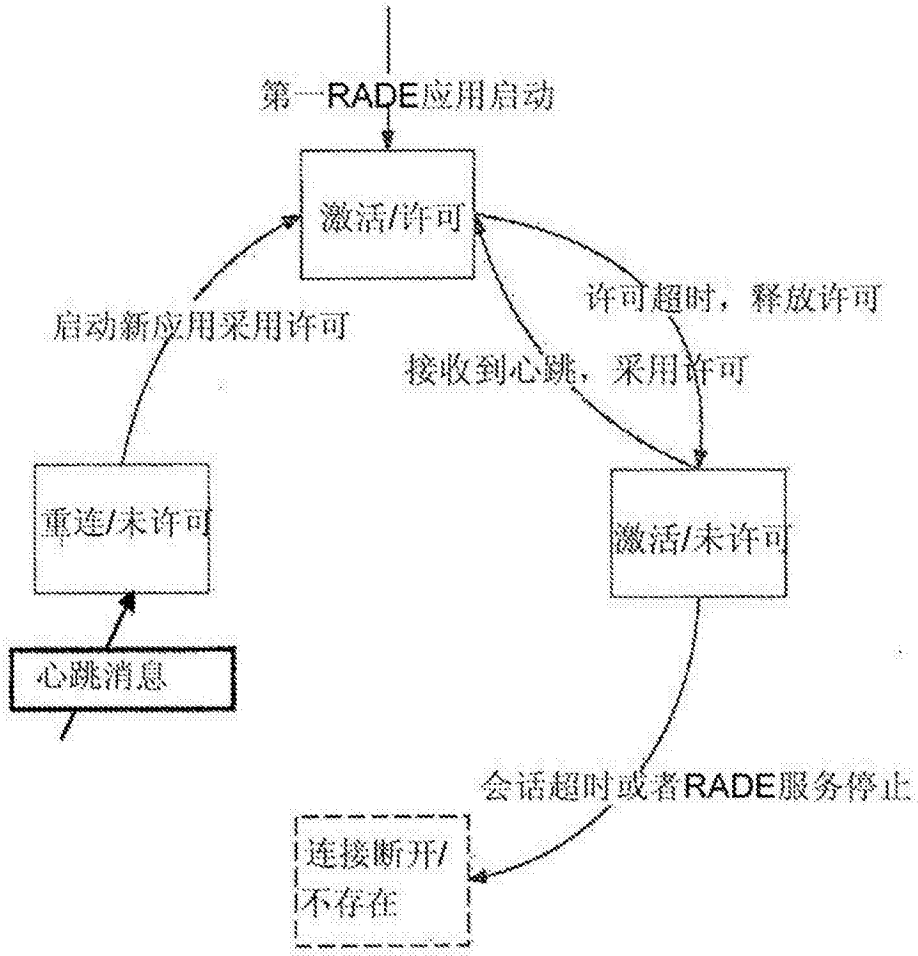

第一RADE应用启动
激活/许可
许可超时，释放许可
启动新应用采用许可
接收到心跳，采用许可
重连/未许可
激活/未许可
心跳消息
会话超时或者RADE服务停止
连接断开/
不存在

图18

包

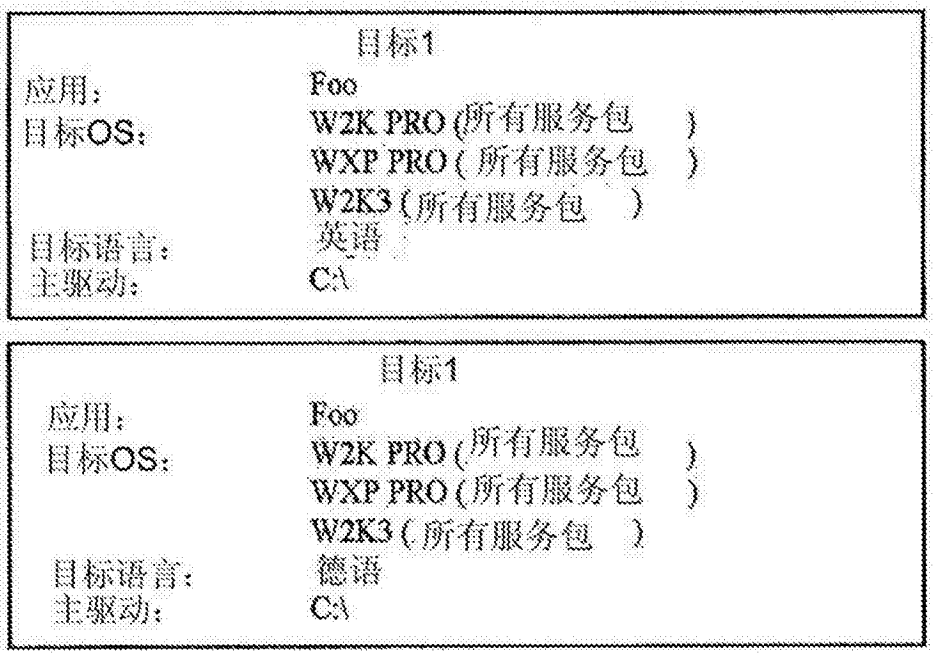

目标1
应用：
Foo
目标OS：
W2K PRO (所有服务包)
WXP PRO (所有服务包)
W2K3 (所有服务包)
目标语言：
英语
主驱动：
C:\
目标1
应用：
Foo
目标OS：
W2K PRO (所有服务包)
WXP PRO (所有服务包)
W2K3 (所有服务包)
目标语言：
德语
主驱动：
C:\

图19

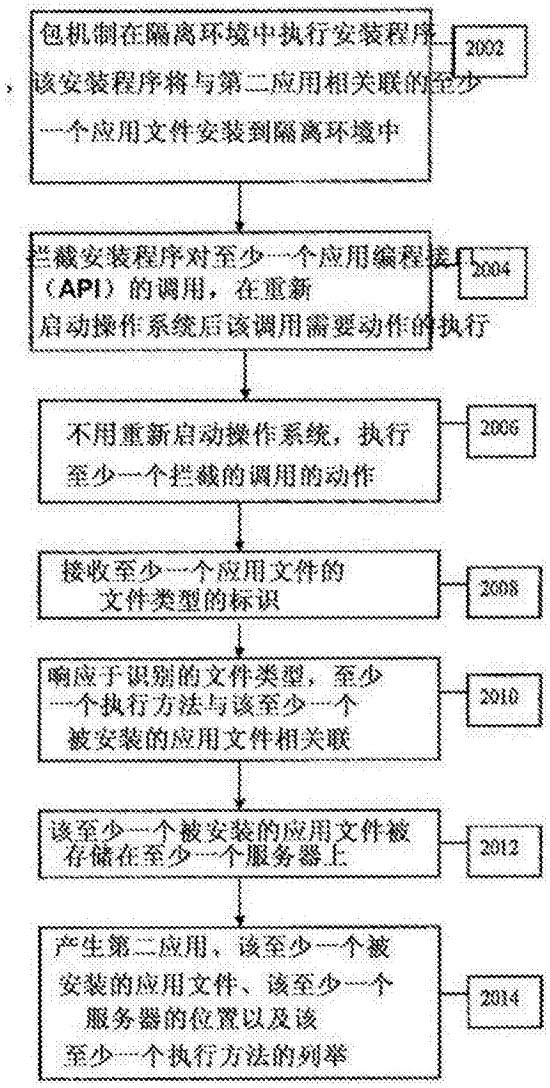

包机制在隔离环境中执行安装程序，该安装程序将与第二应用相关联的至少一个应用文件安装到隔离环境中
2002
拦截安装程序对至少一个应用编程接口（API）的调用，在重新启动操作系统后该调用需要动作的执行
2004
不用重新启动操作系统，执行至少一个拦截的调用的动作
2006
接收至少一个应用文件的文件类型的标识
2008
响应于识别的文件类型，至少一个执行方法与该至少一个被安装的应用文件相关联
2010
该至少一个被安装的应用文件被存储在至少一个服务器上
2012
产生第二应用，该至少一个被安装的应用文件、该至少一个服务器的位置以及该至少一个执行方法的列举
2014

图20

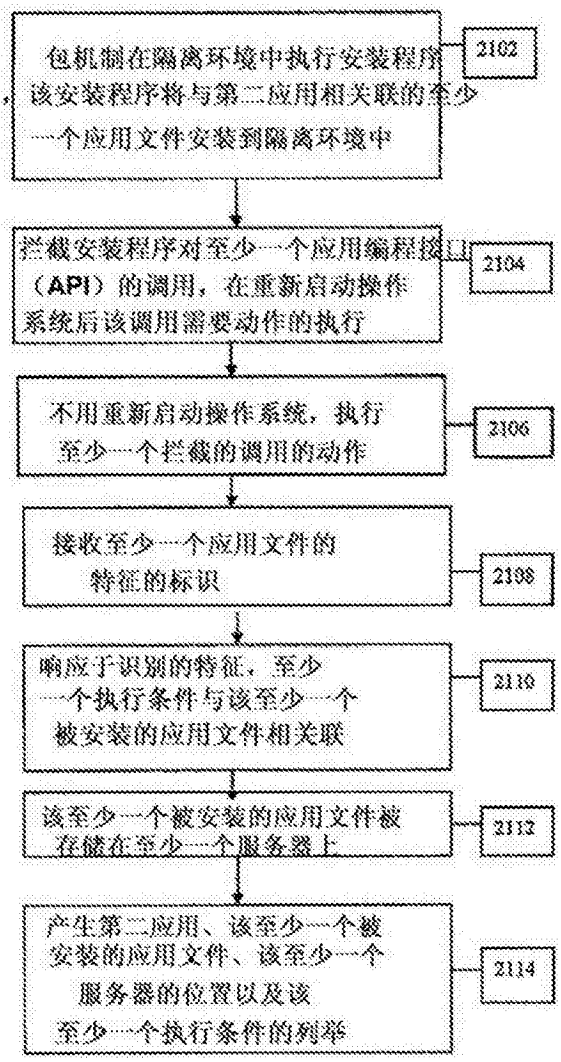

包机制在隔离环境中执行安装程序，该安装程序将与第二应用相关联的至少一个应用文件安装到隔离环境中
2102
拦截安装程序对至少一个应用编程接口（API）的调用，在重新启动操作系统后该调用需要动作的执行
2104
不用重新启动操作系统，执行至少一个拦截的调用的动作
2106
接收至少一个应用文件的特征的标识
2108
响应于识别的特征，至少一个执行条件与该至少一个被安装的应用文件相关联
2110
该至少一个被安装的应用文件被存储在至少一个服务器上
2112
产生第二应用、该至少一个被安装的应用文件、该至少一个服务器的位置以及该至少一个执行条件的列举
2114

图21

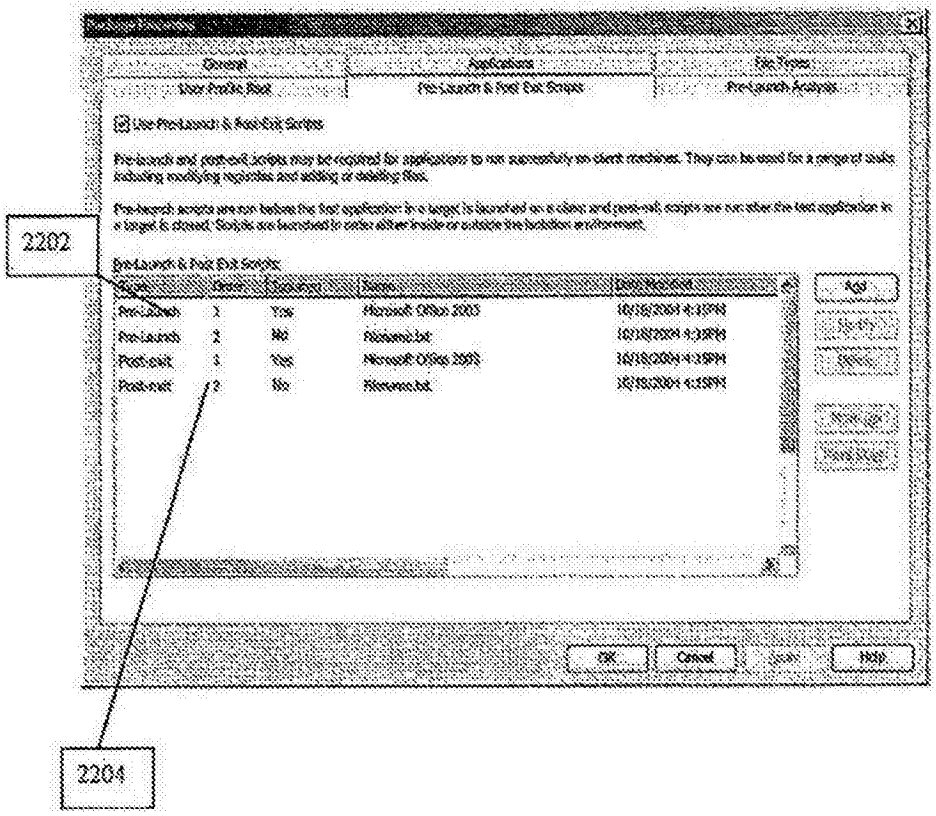

2202
2204

图22

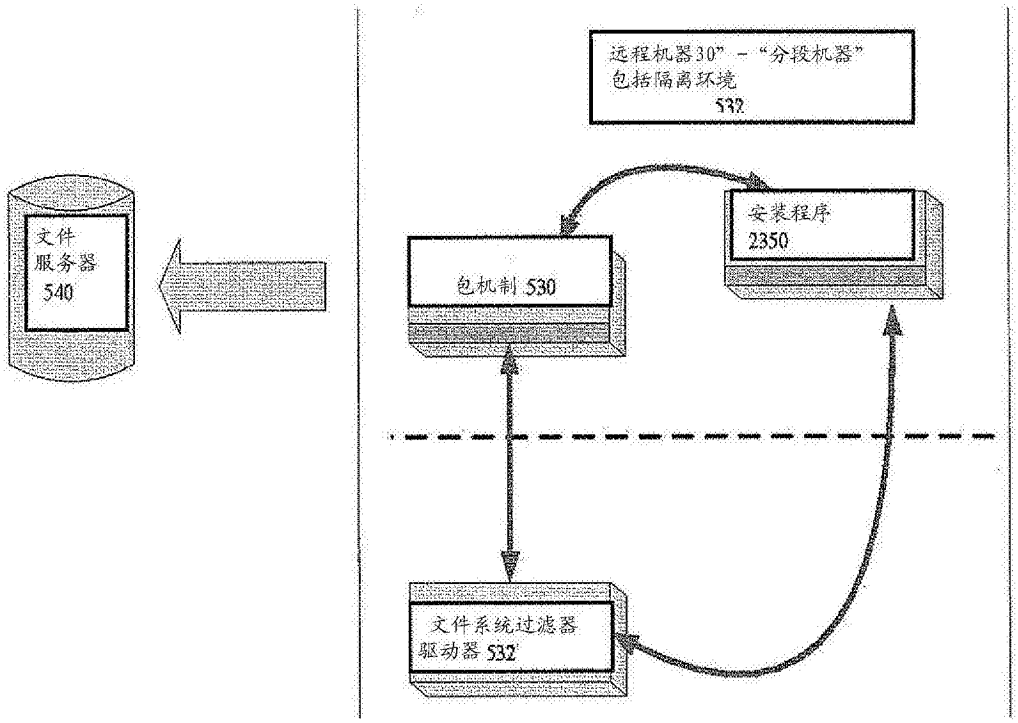

远程机器30" - "分段机器"
包括隔离环境
532
文件
服务器
540
包机制 530
安装程序
2350
文件系统过滤器
驱动器 532'

图23

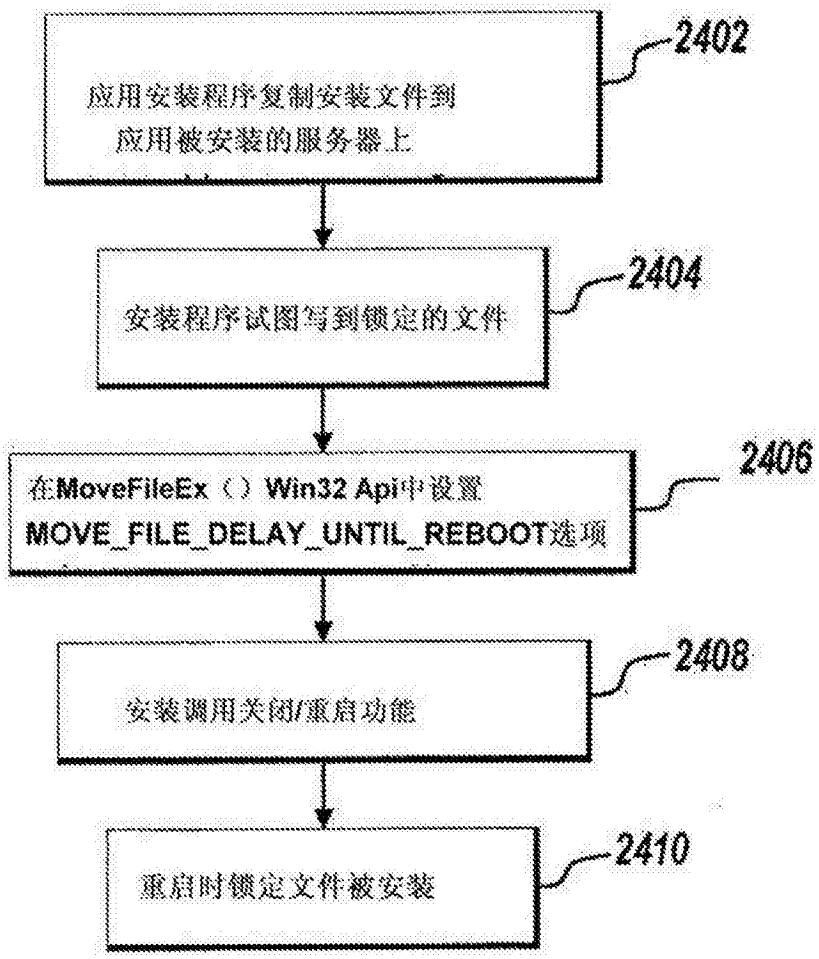

应用安装程序复制安装文件到
应用被安装的服务器上
2402
安装程序试图写到锁定的文件
2404
在MoveFileEx（）Win32 Api中设置
MOVE_FILE_DELAY_UNTIL_REBOOT选项
2406
安装调用关闭/重启功能
2408
重启时锁定文件被安装
2410

图24

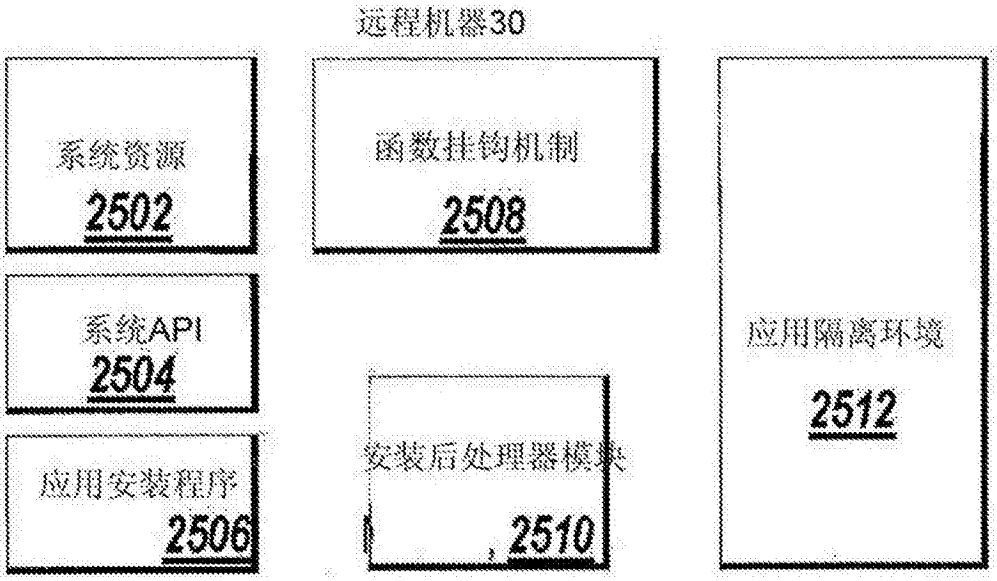

远程机器30
系统资源
2502
函数挂钩机制
2508
系统API
2504
应用隔离环境
2512
应用安装程序
2506
安装后处理器模块
2510

图25

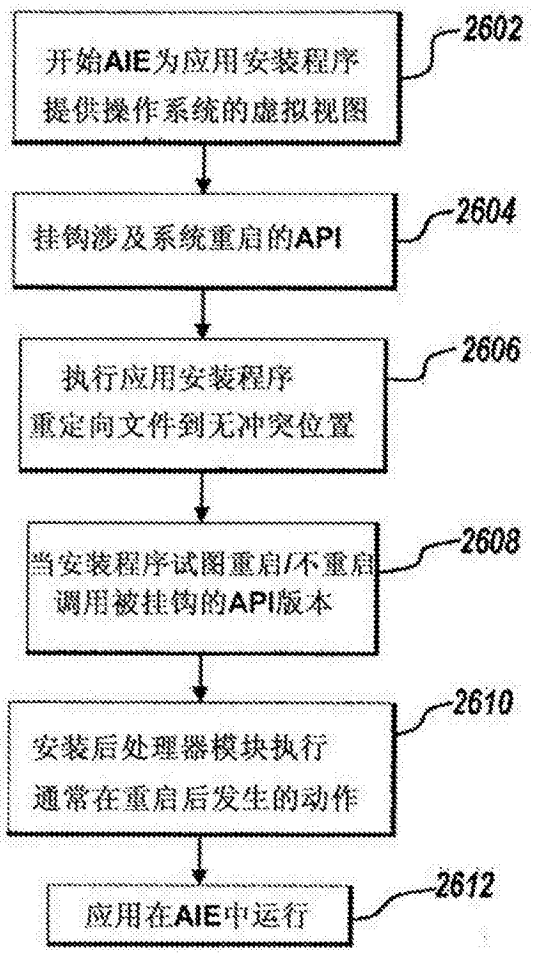

开始AIE为应用安装程序
提供操作系统的虚拟视图
2602
挂钩涉及系统重启的API
2604
执行应用安装程序
重定向文件到无冲突位置
2606
当安装程序试图重启/不重启
调用被挂钩的API版本
2608
安装后处理器模块执行
通常在重启后发生的动作
2610
应用在AIE中运行
2612

图26

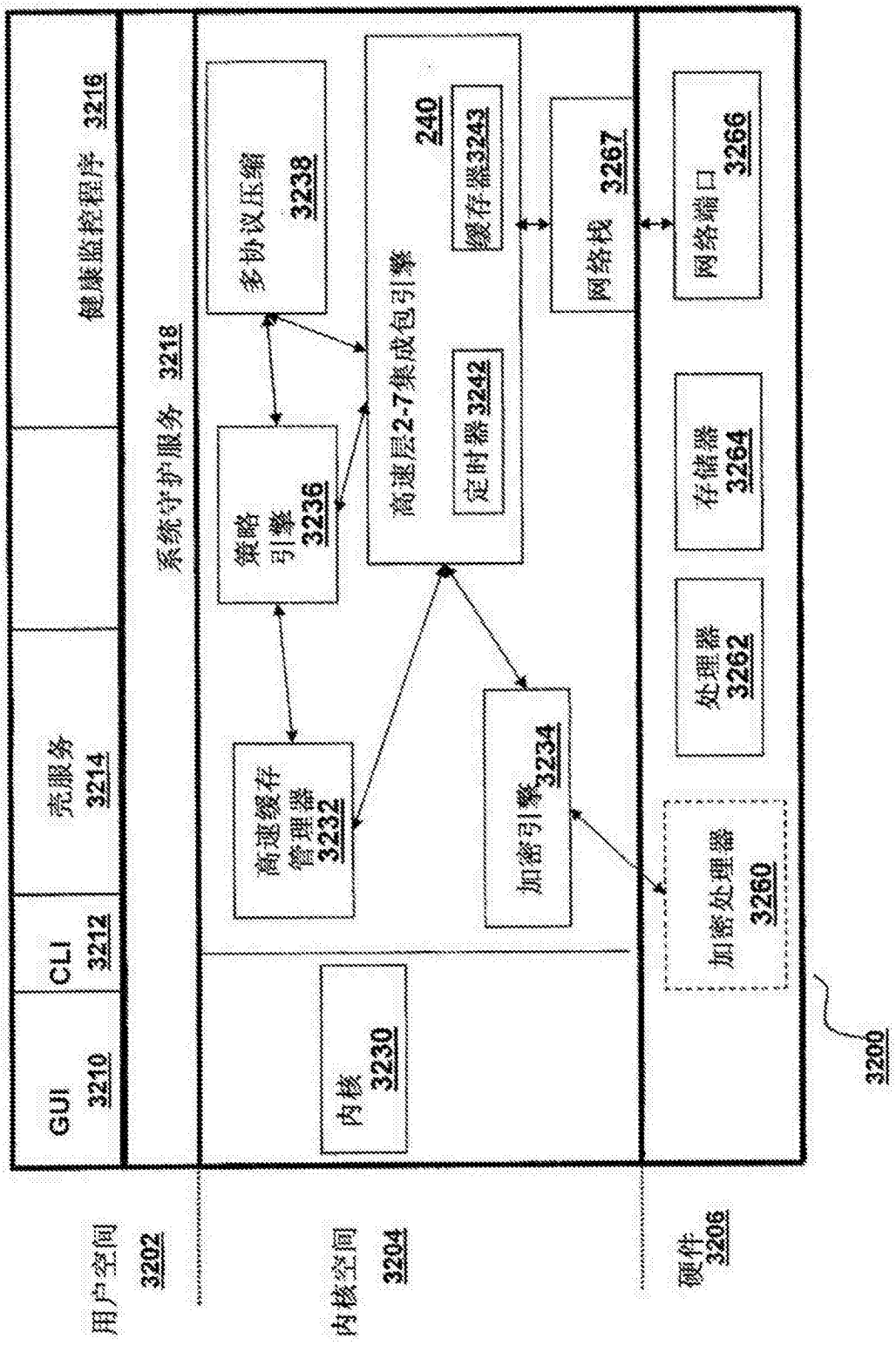

健康监控程序 3216
GUI 3210
CLI 3212
壳服务 3214
系统守护服务 3218
多协议压缩 3238
策略引擎 3236
高速缓存管理器 3232
高速层2-7集成包引擎 240
缓存器3243
定时器 3242
网络栈 3267
网络端口 3266
存储器 3264
处理器 3262
加密引擎3234
加密处理器 3260
内核 3230
3200
用户空间 3202
内核空间 3204
硬件 3206

图27

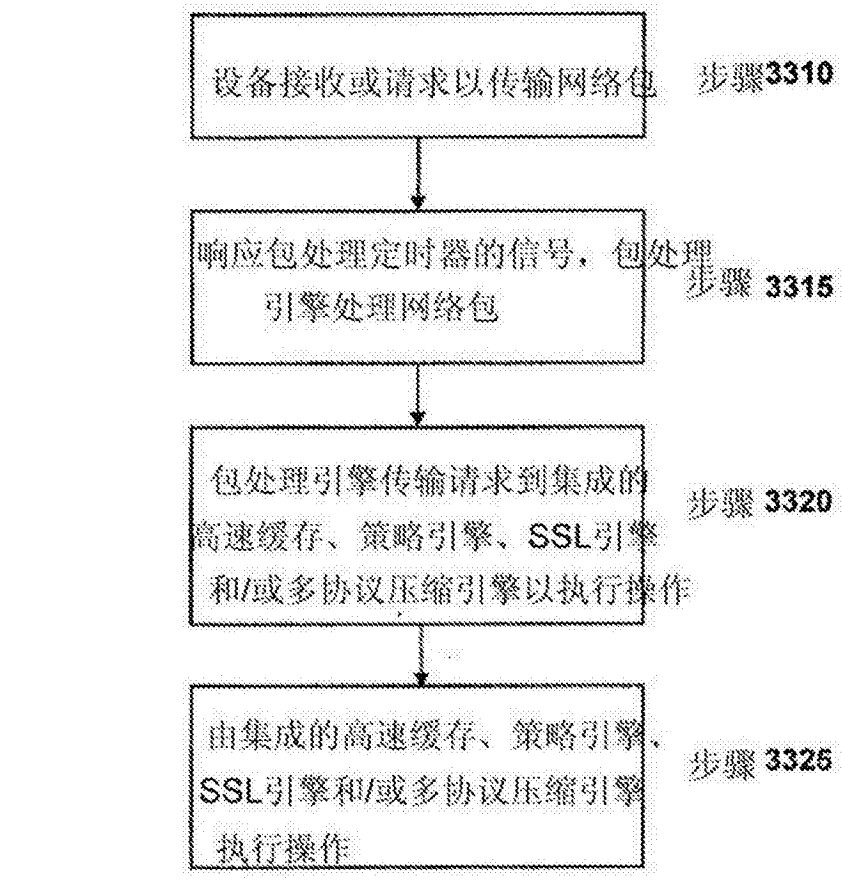

设备接收或请求以传输网络包
步骤3310
响应包处理定时器的信号，包处理引擎处理网络包
步骤 3315
包处理引擎传输请求到集成的高速缓存、策略引擎、SSL引擎和/或多协议压缩引擎以执行操作
步骤 3320
由集成的高速缓存、策略引擎、SSL引擎和/或多协议压缩引擎执行操作
步骤 3325
3300

图28A

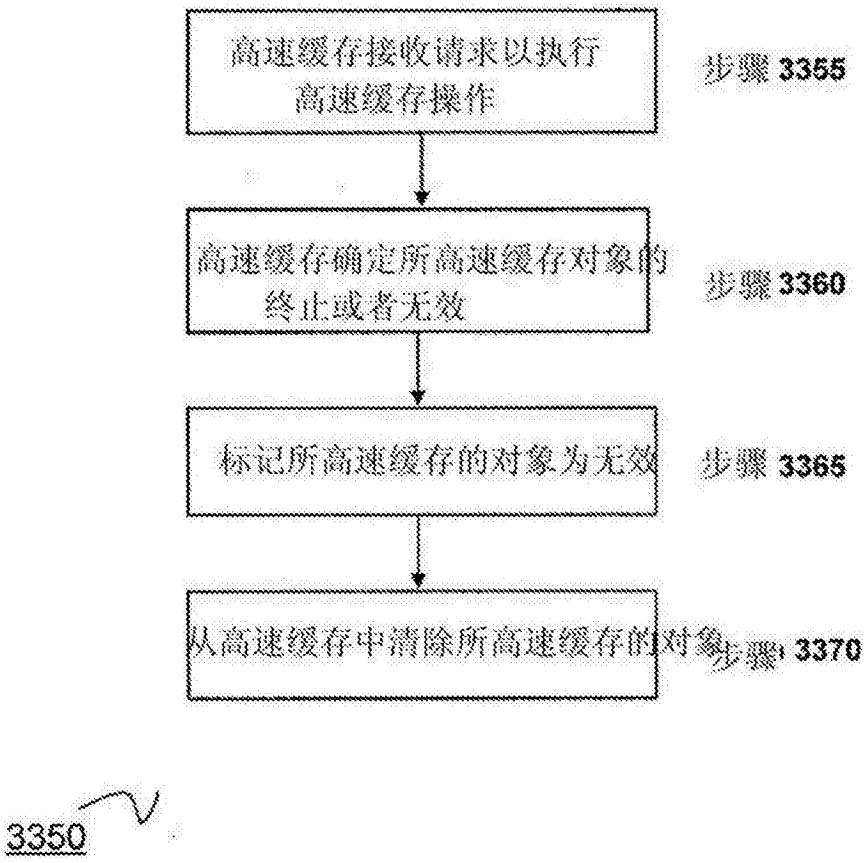

高速缓存接收请求以执行高速缓存操作
步骤 3355
高速缓存确定所高速缓存对象的终止或者无效
步骤 3360
标记所高速缓存的对象为无效
步骤 3365
从高速缓存中清除所高速缓存的对象
步骤 3370
3350

图28B

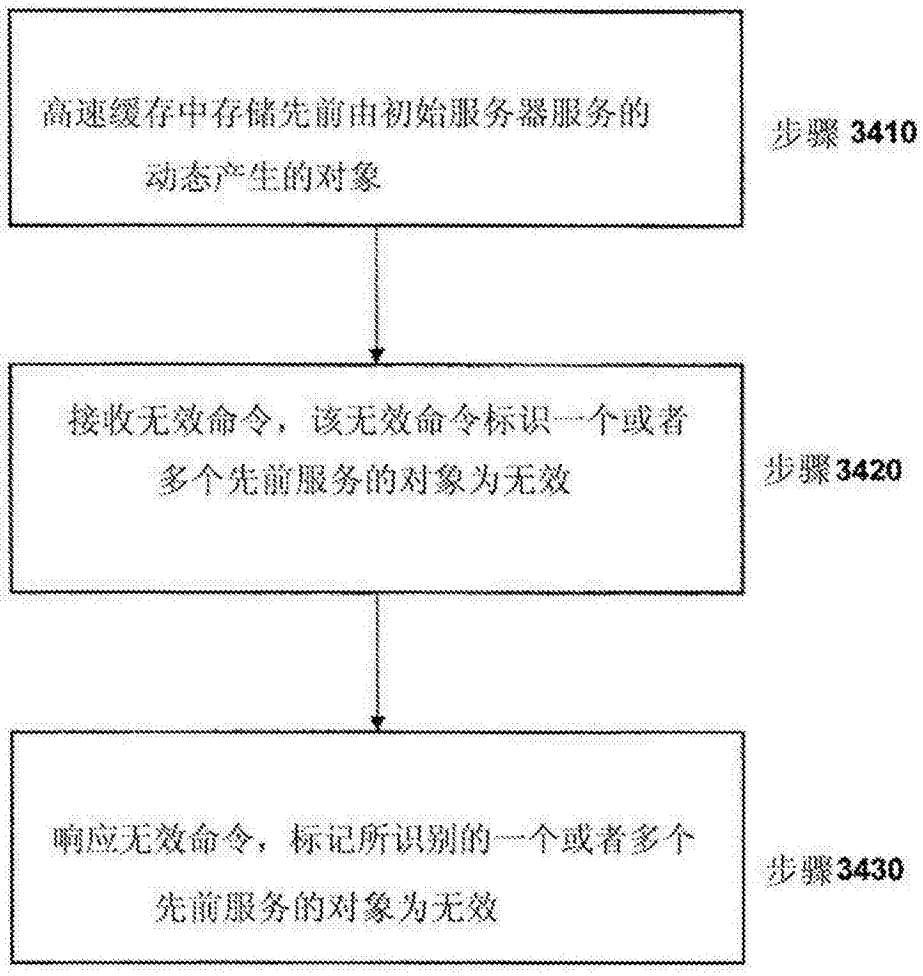

高速缓存中存储先前由初始服务器服务的动态产生的对象
步骤 3410
接收无效命令，该无效命令标识一个或者多个先前服务的对象为无效
步骤3420
响应无效命令，标记所识别的一个或者多个先前服务的对象为无效
步骤3430

图29A


先前被服务的来自初始服务器的动态产生对象被存储在高速缓存中
步骤 3410
在高速缓存中形成先前从初始服务器服务的对象的组
步骤 3412
维持组的记录
步骤 3414
接收无效命令以无效组
步骤 3422
标记组为无效
步骤 3432

图29B

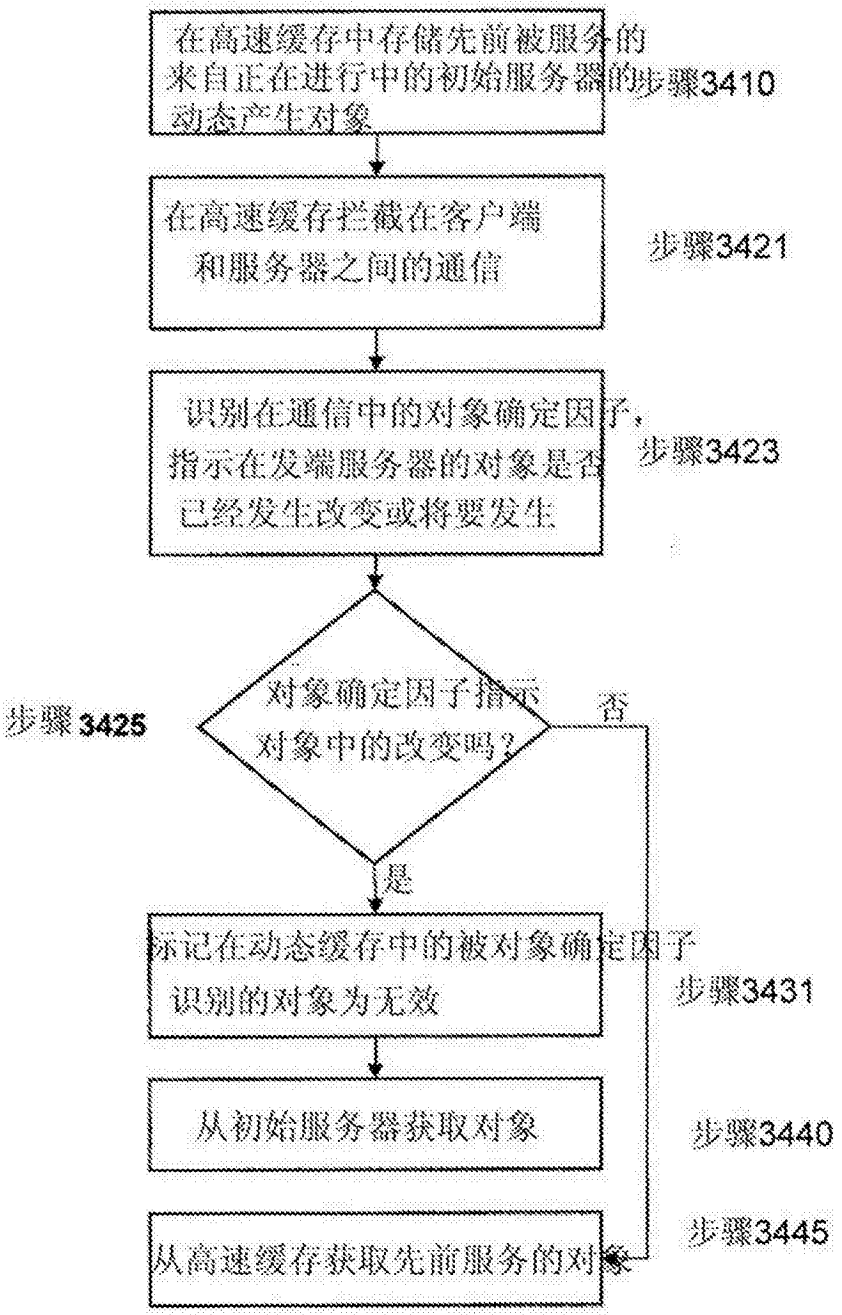

在高速缓存中存储先前被服务的
来自正在进行中的初始服务器的
动态产生对象
步骤3410
在高速缓存拦截在客户端
和服务器之间的通信
步骤3421
识别在通信中的对象确定因子，
指示在发端服务器的对象是否
已经发生改变或将要发生
步骤3423
对象确定因子指示
对象中的改变吗？
步骤3425
否
是
标记在动态缓存中的被对象确定因子
识别的对象为无效
步骤3431
从初始服务器获取对象
步骤3440
从高速缓存获取先前服务的对象
步骤3445

图29C

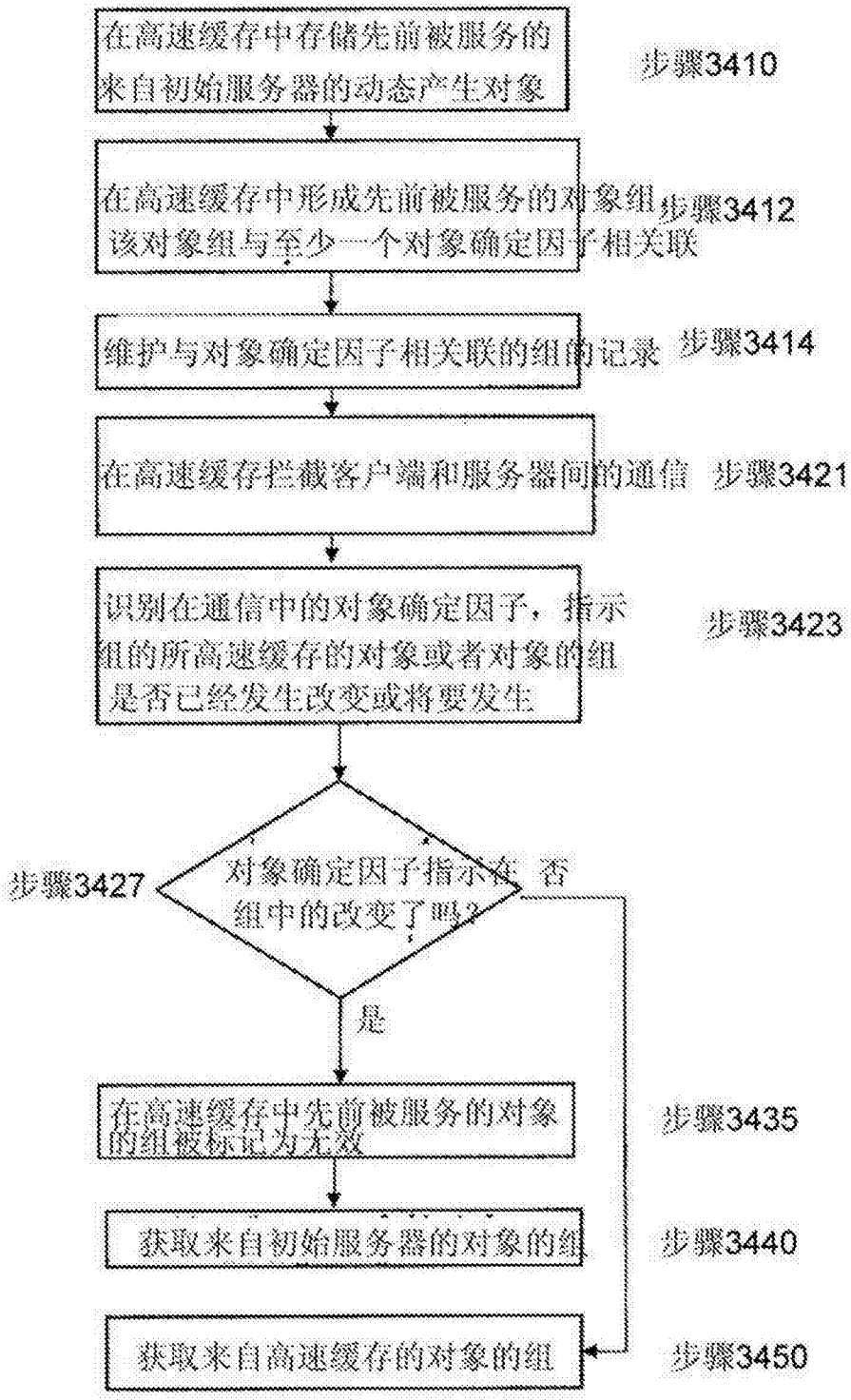

在高速缓存中存储先前被服务的来自初始服务器的动态产生对象
步骤3410
在高速缓存中形成先前被服务的对象组，该对象组与至少一个对象确定因子相关联
步骤3412
维护与对象确定因子相关联的组的记录
步骤3414
在高速缓存拦截客户端和服务器间的通信
步骤3421
识别在通信中的对象确定因子，指示组的所高速缓存的对象或者对象的组是否已经发生改变或将要发生
步骤3423
步骤3427
对象确定因子指示在组中的改变了吗？
否
是
在高速缓存中先前被服务的对象的组被标记为无效
步骤3435
获取来自初始服务器的对象的组
步骤3440
获取来自高速缓存的对象的组
步骤3450

图29D

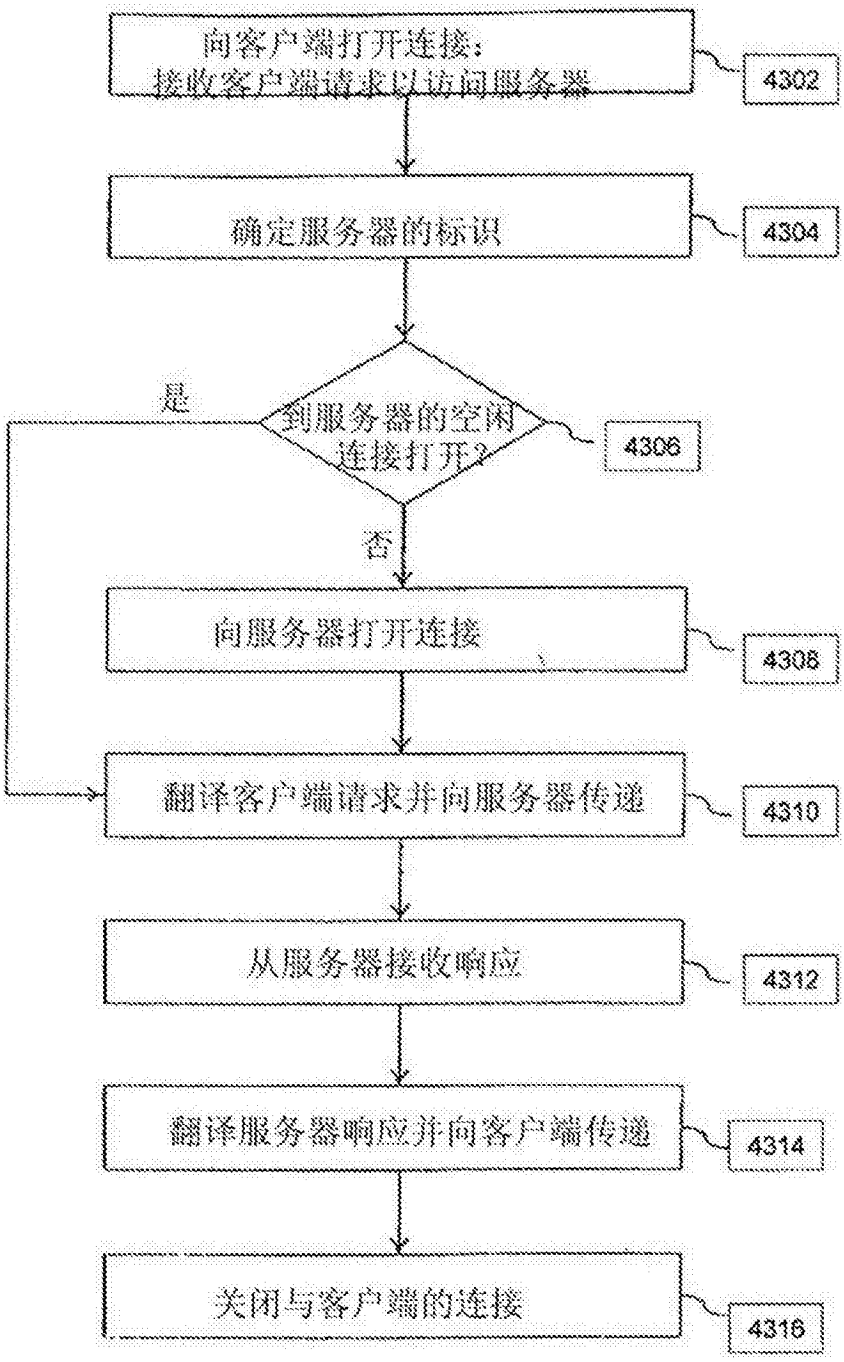

向客户端打开连接：
接收客户端请求以访问服务器
4302
确定服务器的标识
4304
是
到服务器的空闲
连接打开？
4306
否
向服务器打开连接
4308
翻译客户端请求并向服务器传递
4310
从服务器接收响应
4312
翻译服务器响应并向客户端传递
4314
关闭与客户端的连接
4316

图30

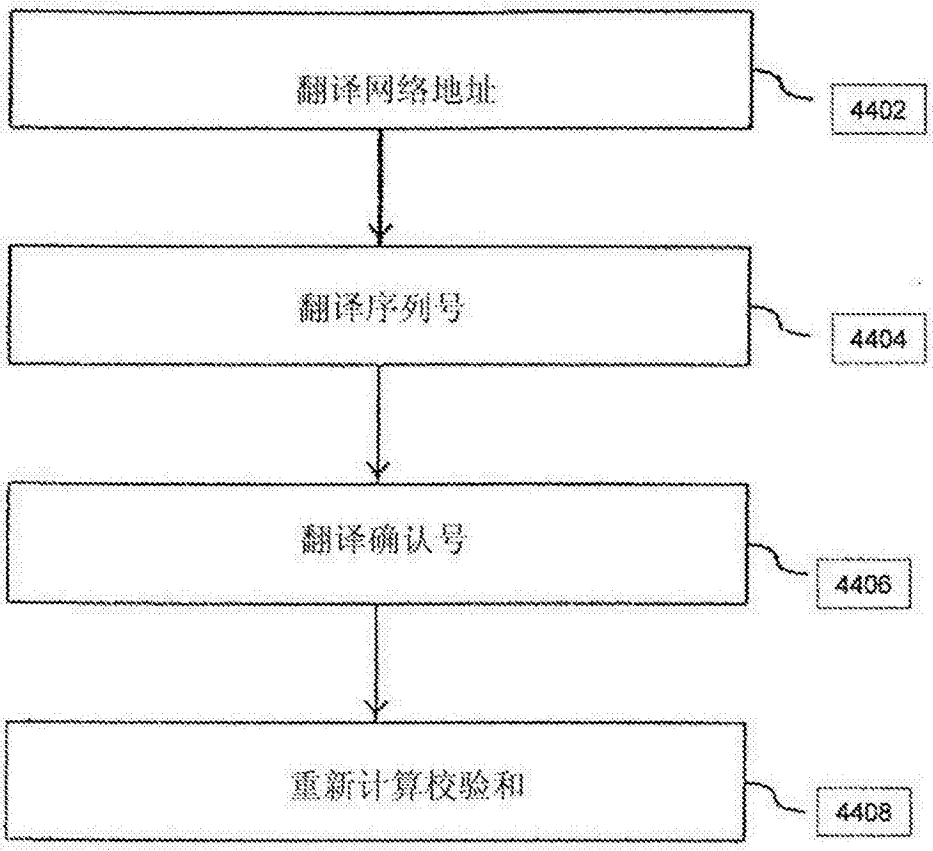

翻译网络地址
4402
翻译序列号
4404
翻译确认号
4406
重新计算校验和
4408

图31

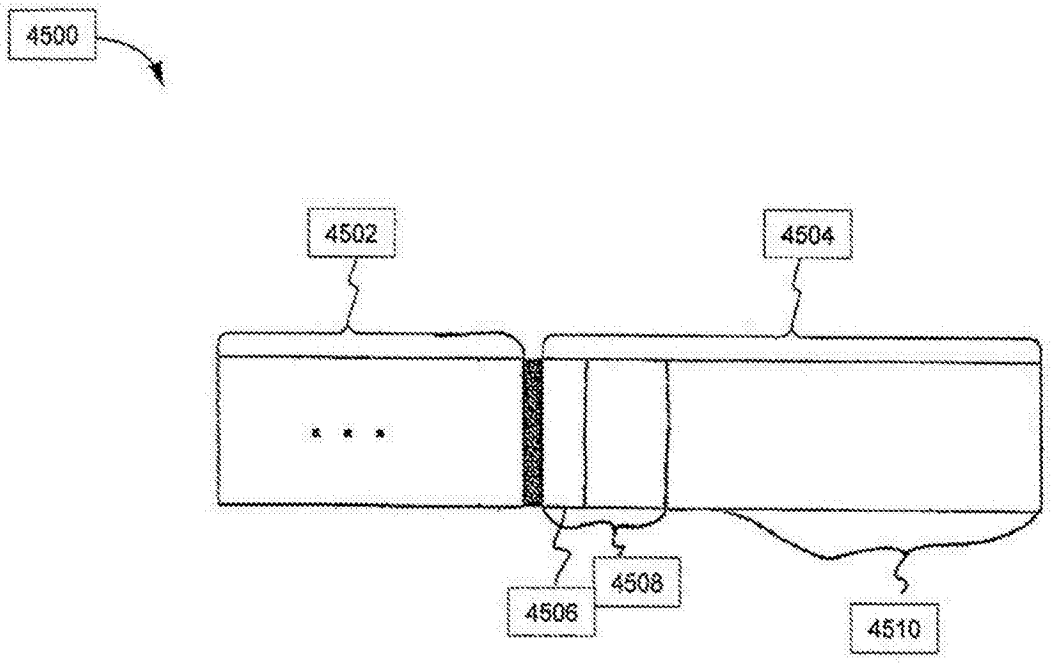

4500
4502
4504
4506
4508
4510

图32

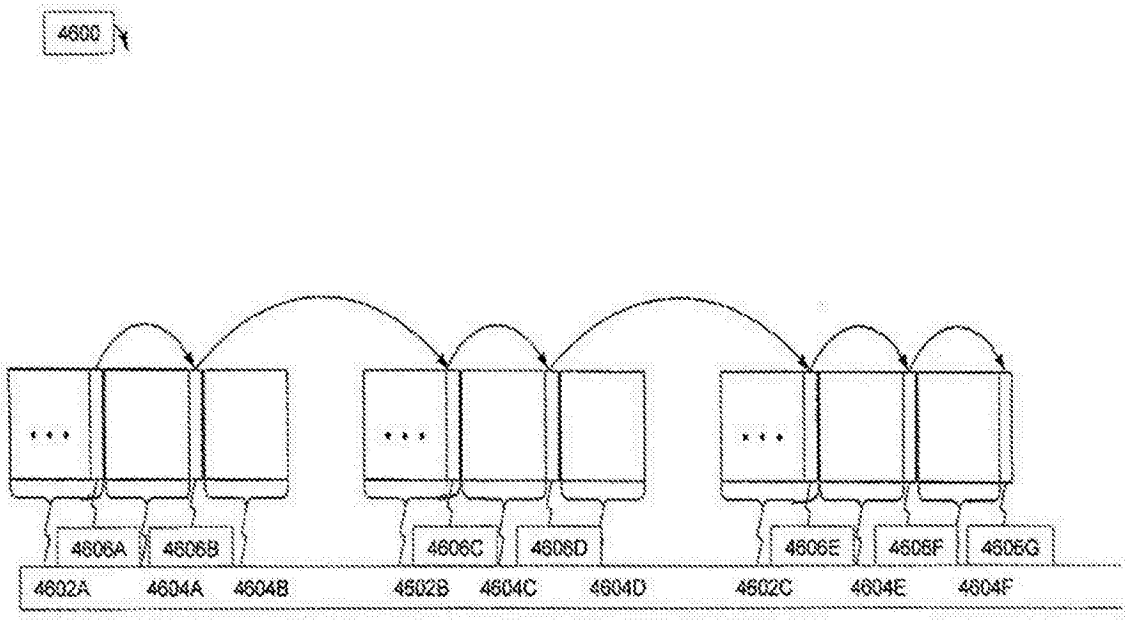

4600
4606A
4606B
4606C
4606D
4606E
4606F
4606G
4602A
4604A
4604B
4602B
4604C
4604D
4602C
4604E
4604F

图33

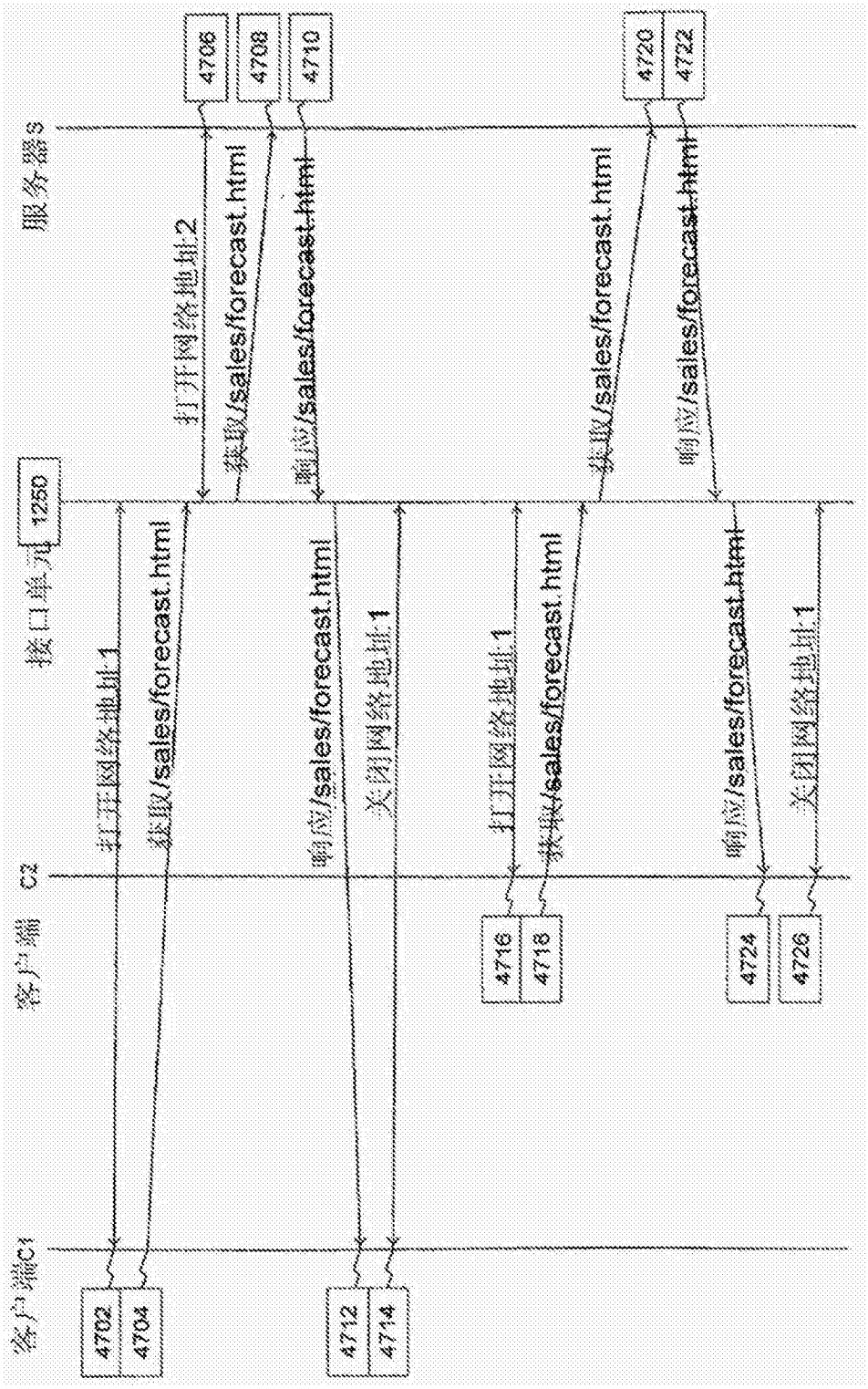

客户端C1
客户端 C2
接口单元 1250
服务器S
4702
4704
打开网络地址1
获取/sales/forecast.html
4706
打开网络地址2
4708
获取/sales/forecast.html
4710
响应/sales/forecast.html
4712
4714
响应/sales/forecast.html
关闭网络地址1
4716
4718
打开网络地址1
获取/sales/forecast.html
4720
4722
获取/sales/forecast.html
响应/sales/forecast.html
4724
4726
响应/sales/forecast.html
关闭网络地址1

图34

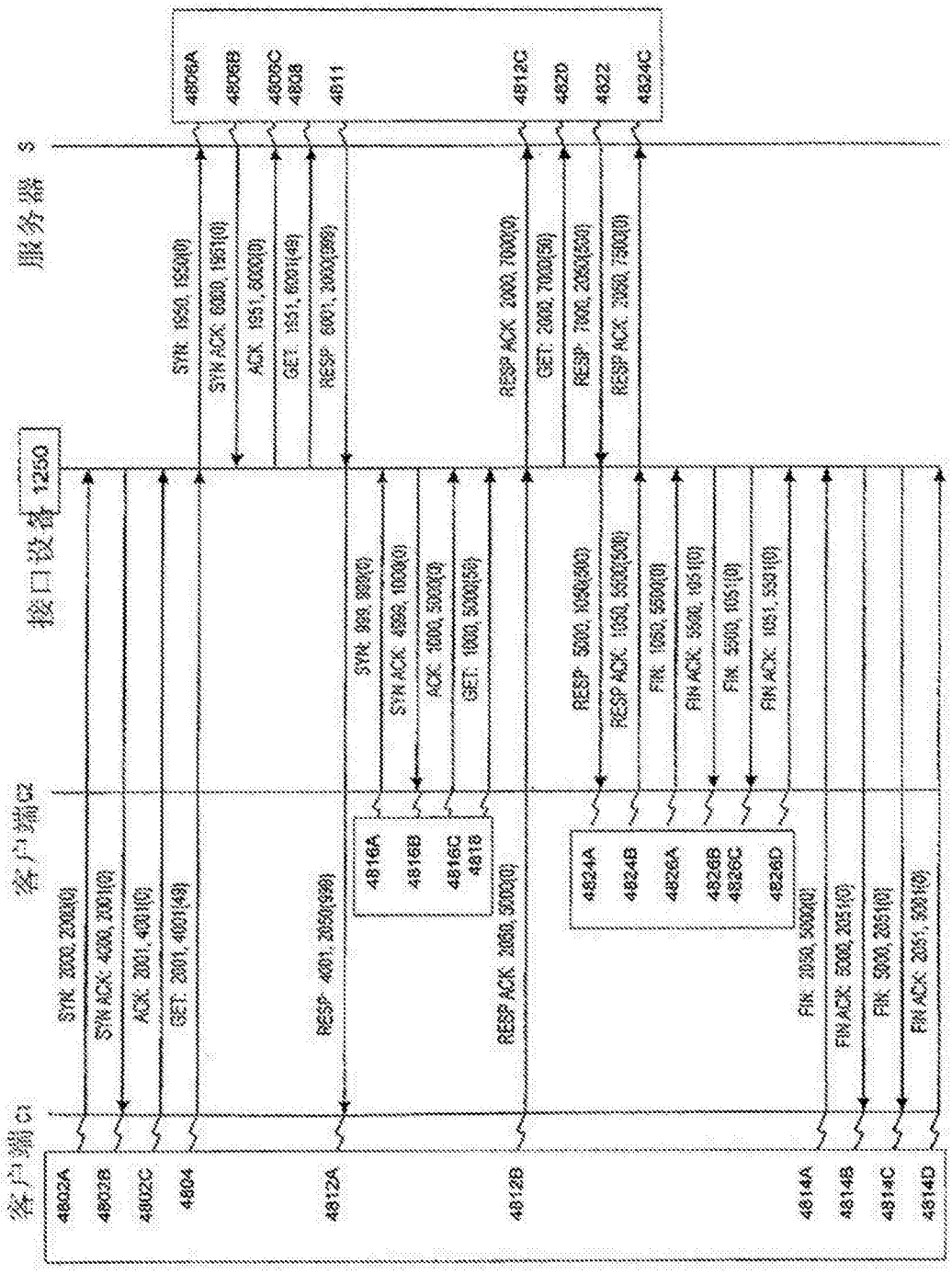

客户端C
客户端D
接口设备 1250
服务器 S
SYN 2000, 2000(0)
SYN ACK 4000, 2001(0)
ACK 2001, 4001(0)
GET 2001, 4001(49)
RESP 4001, 2050(999)
SYN 999, 999(0)
SYN ACK 4999, 1000(0)
ACK 1000, 5000(0)
GET 1000, 5000(56)
RESP ACK 2050, 5000(0)
RESP 5000, 10000(999)
RESP ACK 1050, 5000(500)
FIN 1050, 5000(0)
FIN ACK 5000, 1051(0)
FIN 5000, 1051(0)
FIN ACK 1051, 5001(0)
FIN 2050, 5000(0)
FIN ACK 5000, 2051(0)
FIN 5000, 2051(0)
FIN ACK 2051, 5001(0)
SYN 1950, 1950(0)
SYN ACK 6000, 1951(0)
ACK 1951, 6000(0)
GET 1951, 6001(49)
RESP 6001, 9000(999)
RESP ACK 2000, 7000(0)
GET 2000, 7000(56)
RESP 7000, 20000(500)
RESP ACK 2000, 7300(0)
4802A
4803B
4802C
4804
4812A
4812B
4814A
4814B
4814C
4814D
4816A
4816B
4816C
4818
4824A
4824B
4826A
4826B
4826C
4826D
4806A
4806B
4806C
4808
4811
4813C
4820
4822
4824C

图35

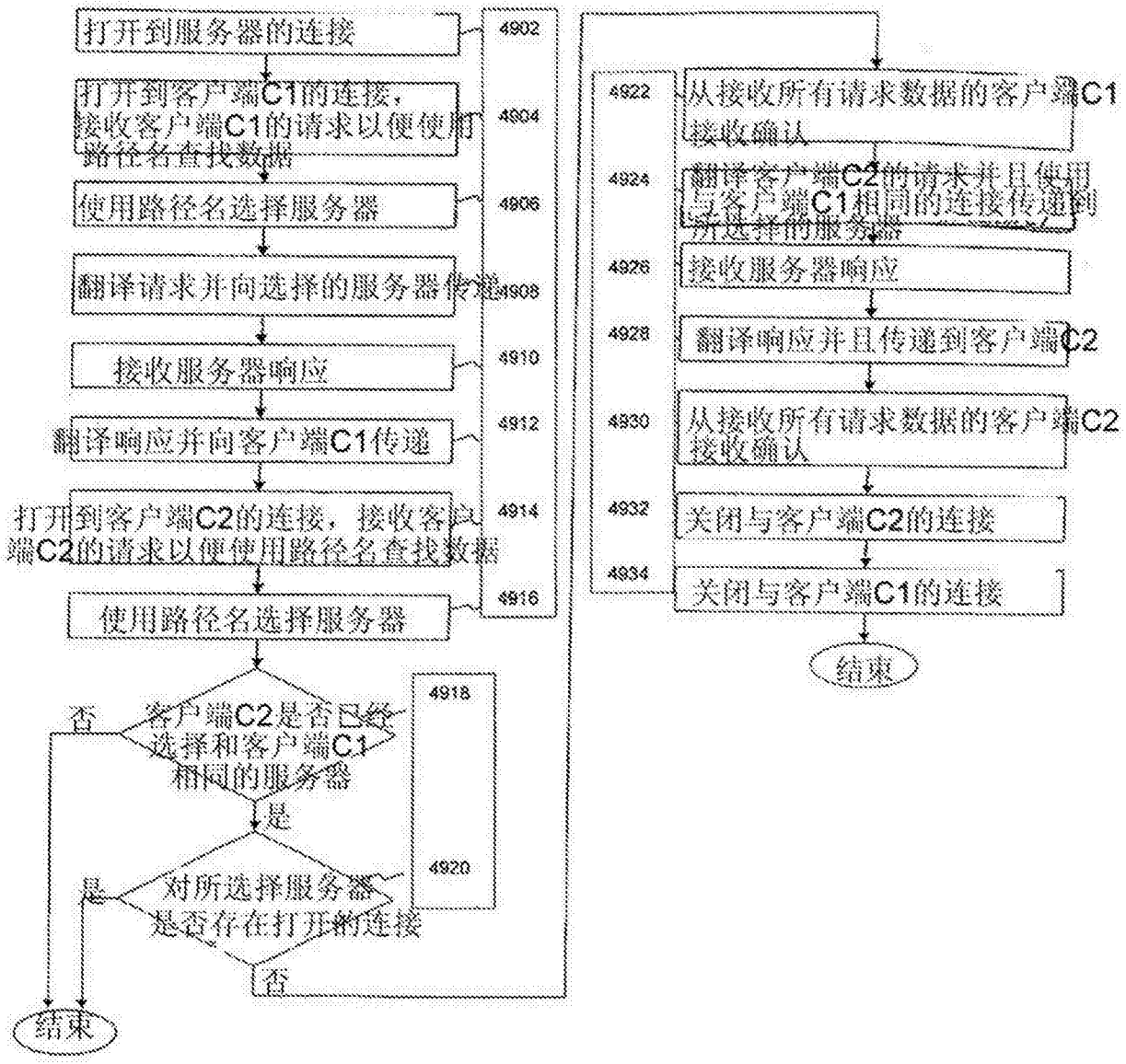

打开到服务器的连接
4902
打开到客户端C1的连接，接收客户端C1的请求以便使用路径名查找数据
4904
使用路径名选择服务器
4906
翻译请求并向选择的服务器传递
4908
接收服务器响应
4910
翻译响应并向客户端C1传递
4912
打开到客户端C2的连接，接收客户端C2的请求以便使用路径名查找数据
4914
使用路径名选择服务器
4916
客户端C2是否已经选择和客户端C1相同的服务器
4918
否
是
对所选择服务器是否存在打开的连接
4920
是
否
结束
从接收所有请求数据的客户端C1接收确认
4922
翻译客户端C2的请求并且使用与客户端C1相同的连接传递到所选择的服务器
4924
接收服务器响应
4926
翻译响应并且传递到客户端C2
4928
从接收所有请求数据的客户端C2接收确认
4930
关闭与客户端C2的连接
4932
关闭与客户端C1的连接
4934
结束

图36

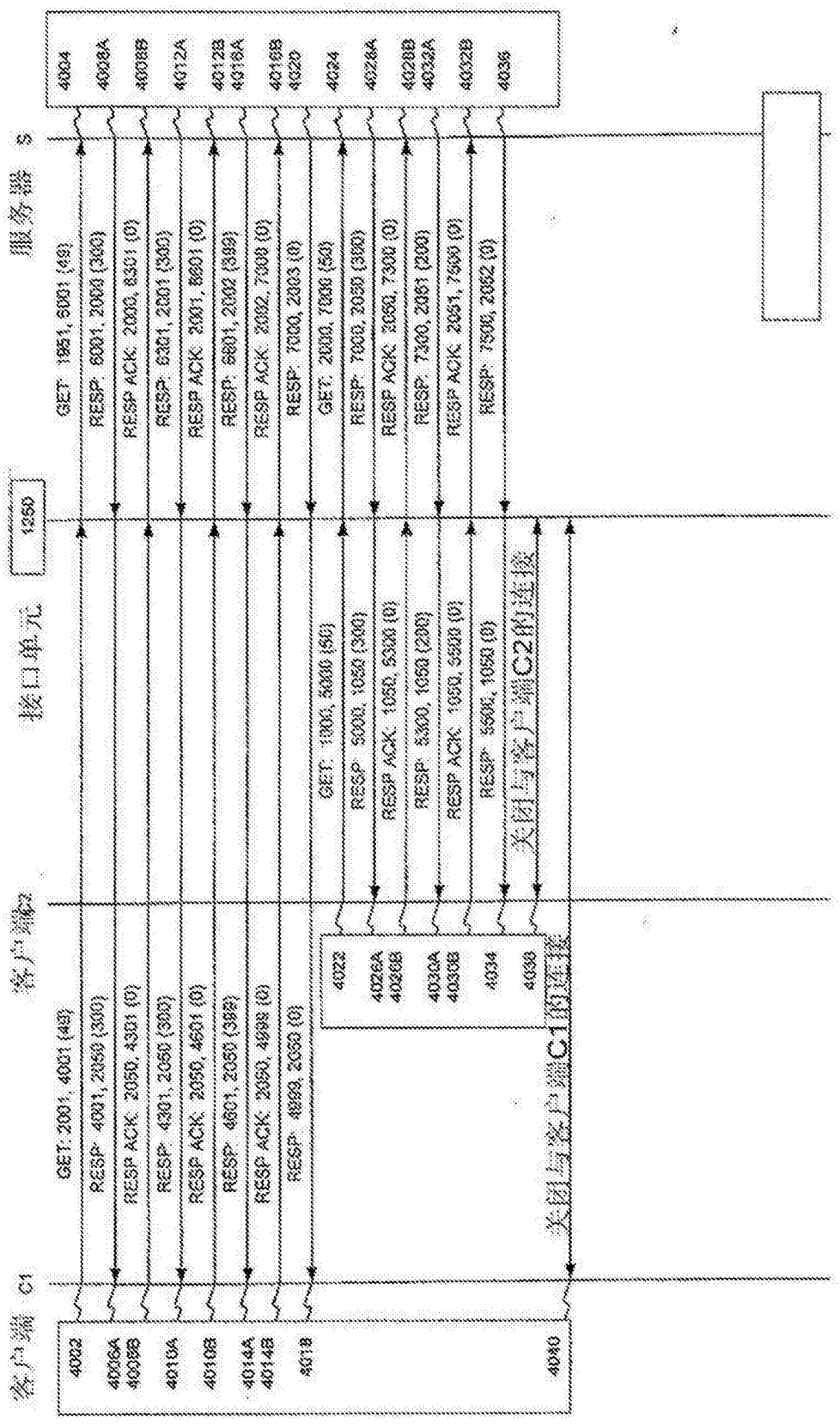

客户端 C1
客户端2
接口单元
1350
服务器 S
GET: 2001, 4001 (48)
RESP: 4001, 2050 (300)
RESP ACK: 2050, 4301 (0)
RESP: 4301, 2050 (300)
RESP ACK: 2050, 4601 (0)
RESP: 4601, 2050 (300)
RESP ACK: 2050, 4900 (0)
RESP: 4900, 2050 (0)
GET: 1951, 6001 (48)
RESP: 6001, 2060 (300)
RESP ACK: 2060, 6301 (0)
RESP: 6301, 2201 (300)
RESP ACK: 2201, 6601 (0)
RESP: 6601, 2062 (300)
RESP ACK: 2062, 7000 (0)
RESP: 7000, 2203 (0)
GET: 1300, 3000 (50)
RESP: 5000, 1050 (300)
RESP ACK: 1050, 5300 (0)
RESP: 5300, 1050 (200)
RESP ACK: 1050, 5500 (0)
RESP: 5500, 1050 (0)
关闭与客户端C2的连接
关闭与客户端C1的连接

图37

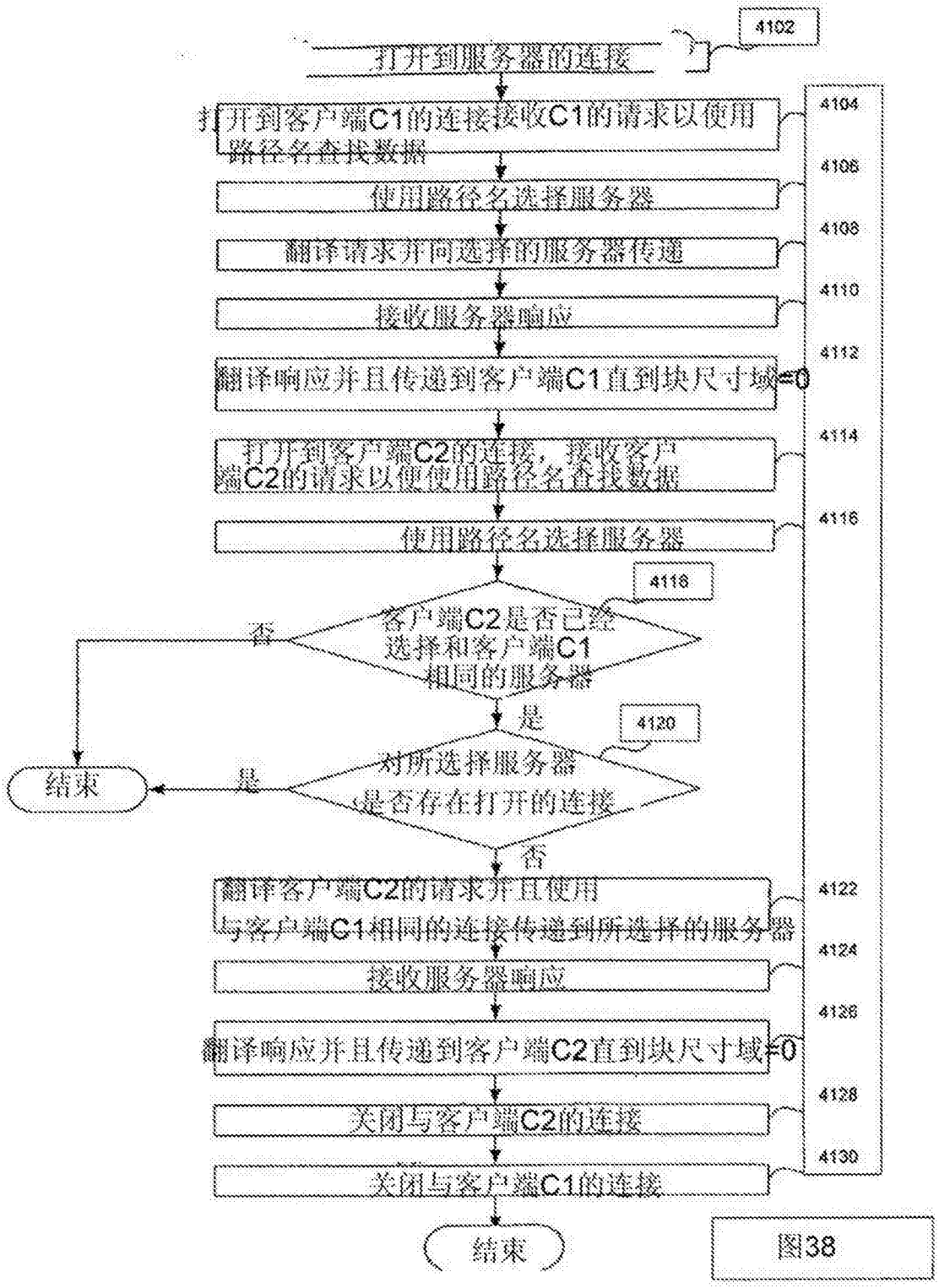

4102
打开到服务器的连接
4104
打开到客户端C1的连接接收C1的请求以使用路径名查找数据
4106
使用路径名选择服务器
4108
翻译请求并向选择的服务器传递
4110
接收服务器响应
4112
翻译响应并且传递到客户端C1直到块尺寸域=0
4114
打开到客户端C2的连接，接收客户端C2的请求以便使用路径名查找数据
4116
使用路径名选择服务器
4118
客户端C2是否已经选择和客户端C1相同的服务器
否
是
4120
对所选择服务器是否存在打开的连接
是
结束
否
4122
翻译客户端C2的请求并且使用与客户端C1相同的连接传递到所选择的服务器
4124
接收服务器响应
4126
翻译响应并且传递到客户端C2直到块尺寸域=0
4128
关闭与客户端C2的连接
4130
关闭与客户端C1的连接
结束
图38

图38

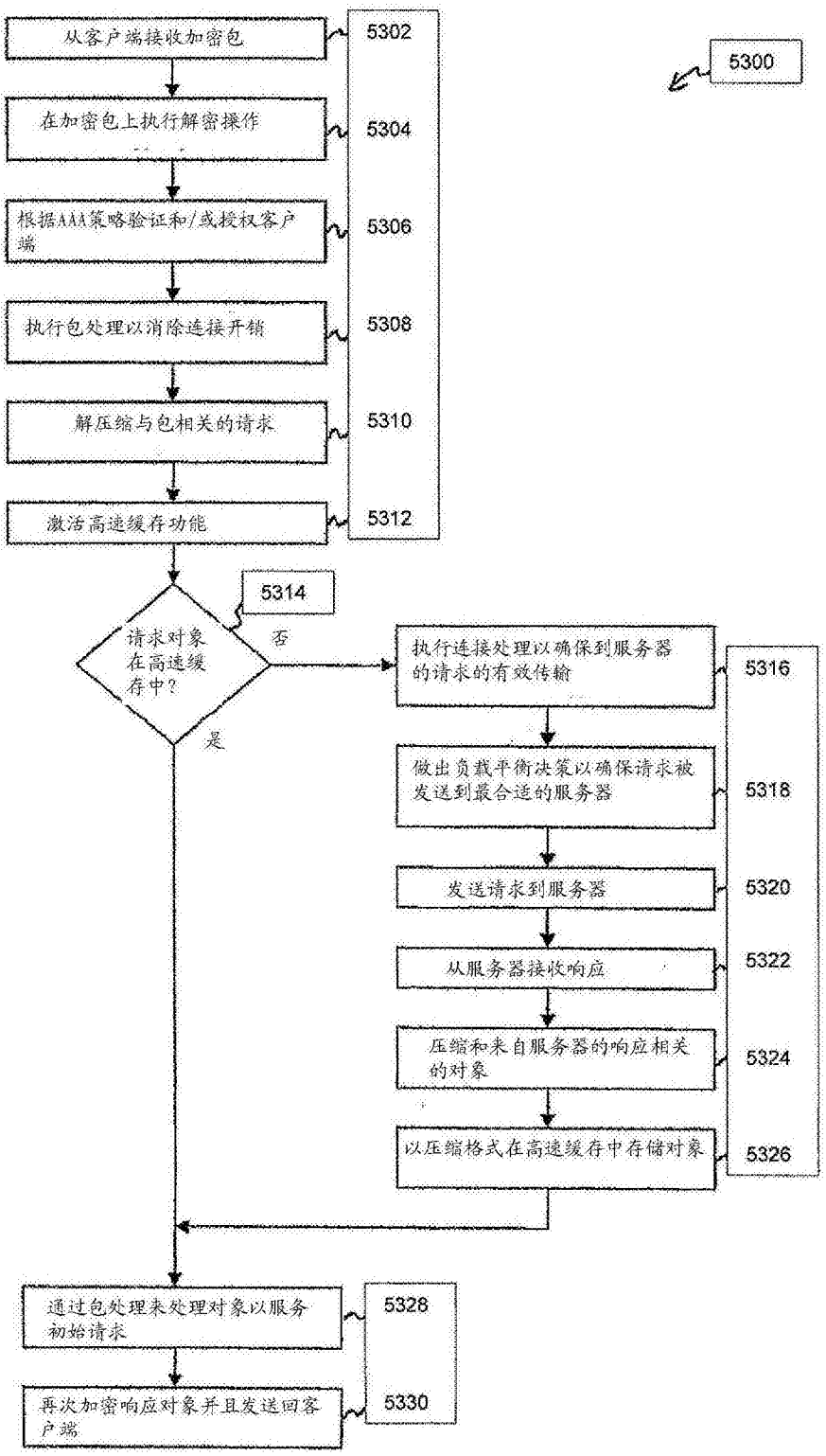

5300
从客户端接收加密包
5302
在加密包上执行解密操作
5304
根据AAA策略验证和/或授权客户端
5306
执行包处理以消除连接开销
5308
解压缩与包相关的请求
5310
激活高速缓存功能
5312
5314
请求对象在高速缓存中?
否
是
执行连接处理以确保到服务器的请求的有效传输
5316
做出负载平衡决策以确保请求被发送到最合适的服务器
5318
发送请求到服务器
5320
从服务器接收响应
5322
压缩和来自服务器的响应相关的对象
5324
以压缩格式在高速缓存中存储对象
5326
通过包处理来处理对象以服务初始请求
5328
再次加密响应对象并且发送回客户端
5330

图39

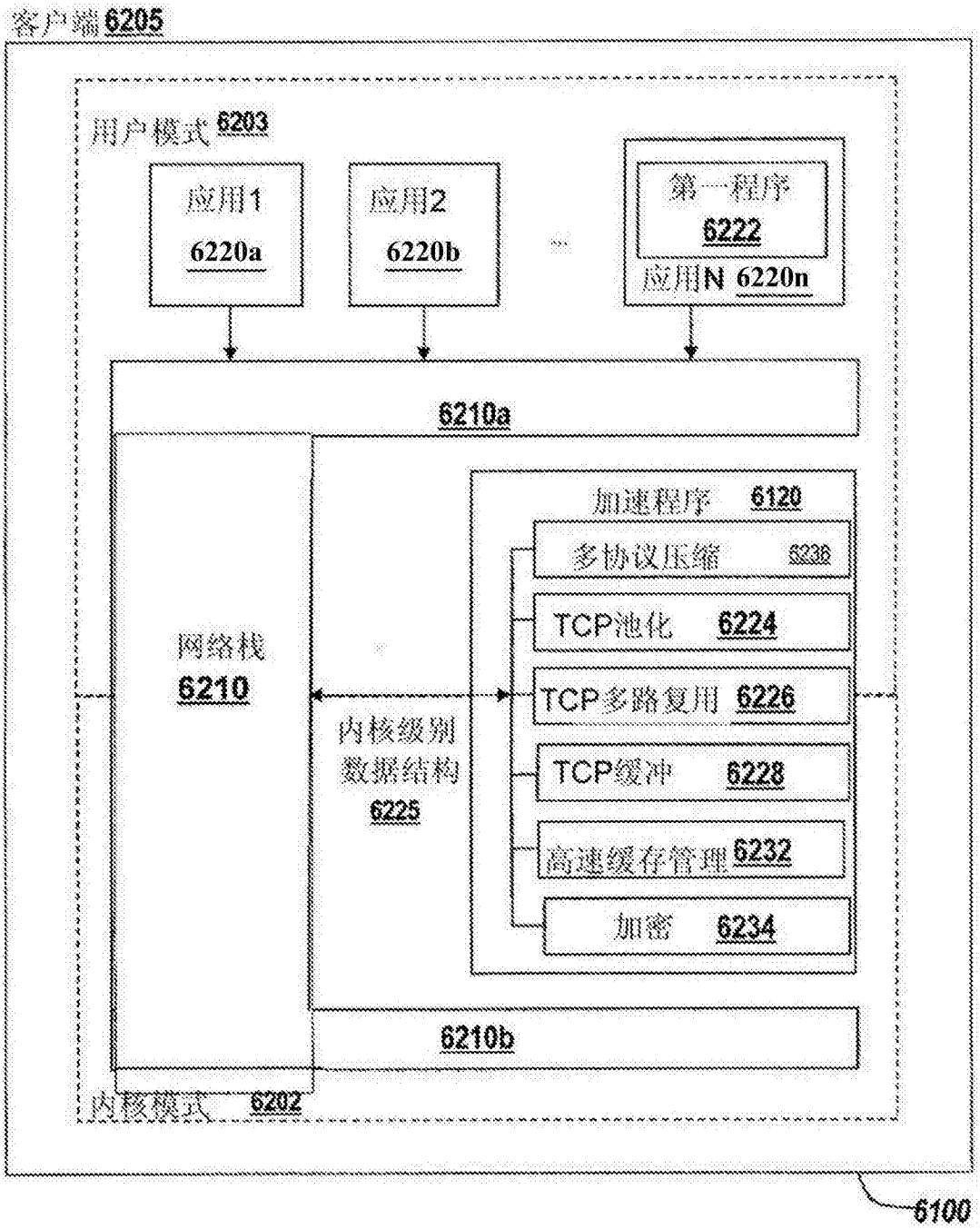

客户端 6205
用户模式 6203
应用1 6220a
应用2 6220b
...
第一程序 6222
应用N 6220n
6210a
加速程序 6120
多协议压缩 6230
TCP池化 6224
TCP多路复用 6226
TCP缓冲 6228
高速缓存管理 6232
加密 6234
网络栈 6210
内核级别数据结构 6225
6210b
内核模式 6202
6100

图40A

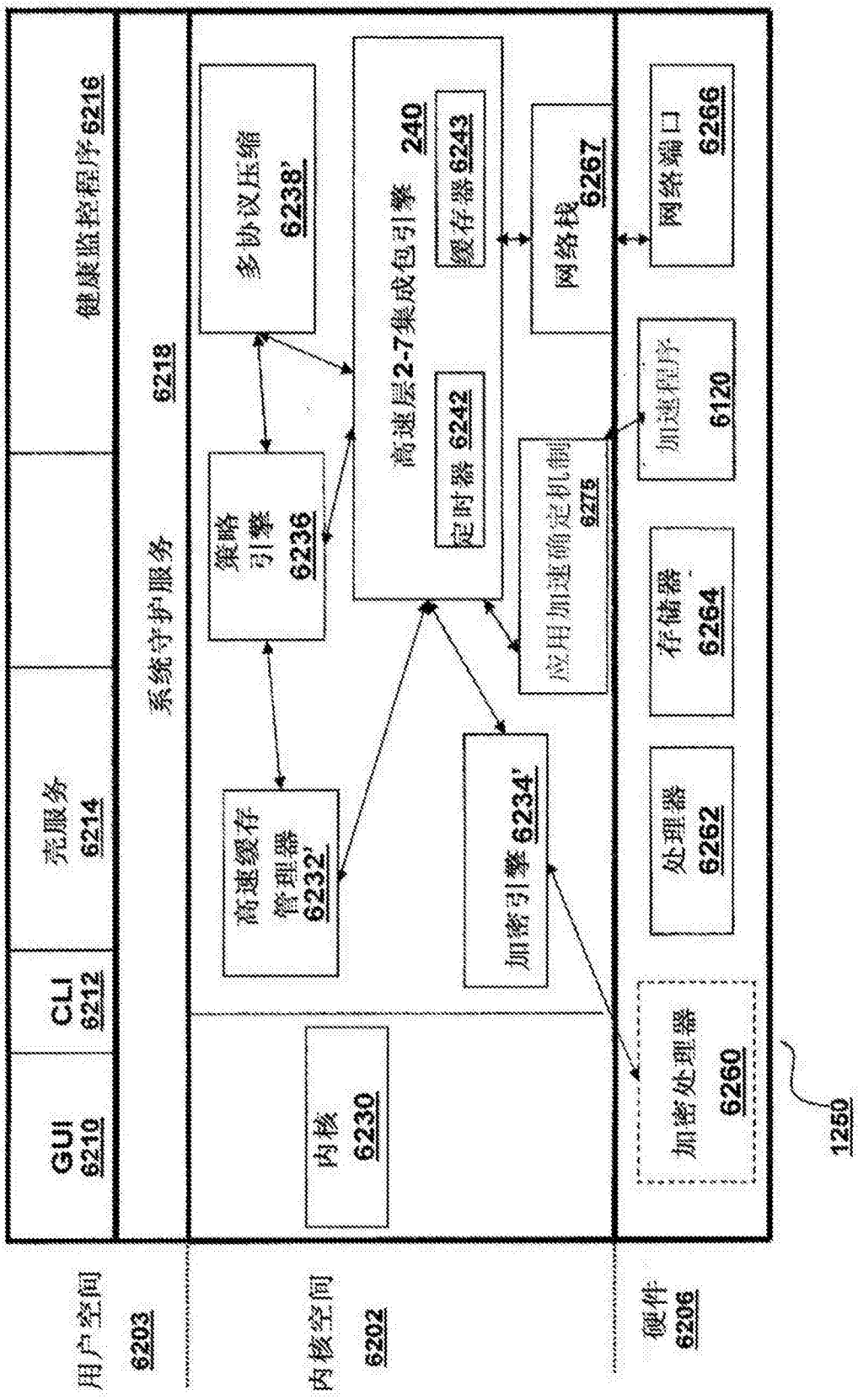

健康监控程序 6216
壳服务 6214
CLI 6212
GUI 6210
系统守护服务 6218
多协议压缩 6238'
策略引擎 6236
高速缓存管理器 6232'
内核 6230
高速层2-7集成包引擎 240
缓存器 6243
定时器 6242
应用加速确定机制 6275
加密引擎 6234'
网络栈 6267
网络端口 6266
加速程序 6120
存储器 6264
处理器 6262
加密处理器 6260
1250
用户空间 6203
内核空间 6202
硬件 6206

图40B

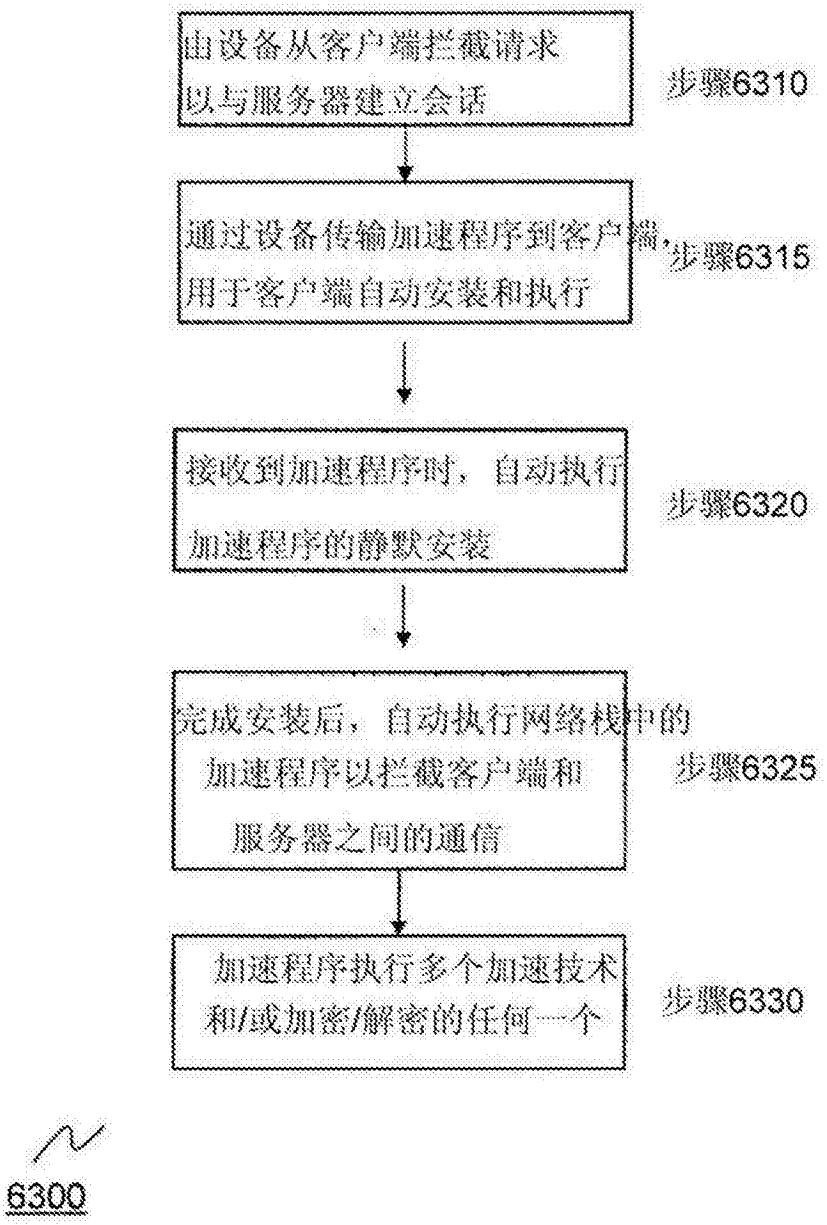

由设备从客户端拦截请求以与服务器建立会话
步骤6310
通过设备传输加速程序到客户端，用于客户端自动安装和执行
步骤6315
接收到加速程序时，自动执行加速程序的静默安装
步骤6320
完成安装后，自动执行网络栈中的加速程序以拦截客户端和服务器之间的通信
步骤6325
加速程序执行多个加速技术和/或加密/解密的任何一个
步骤6330
6300

图41A

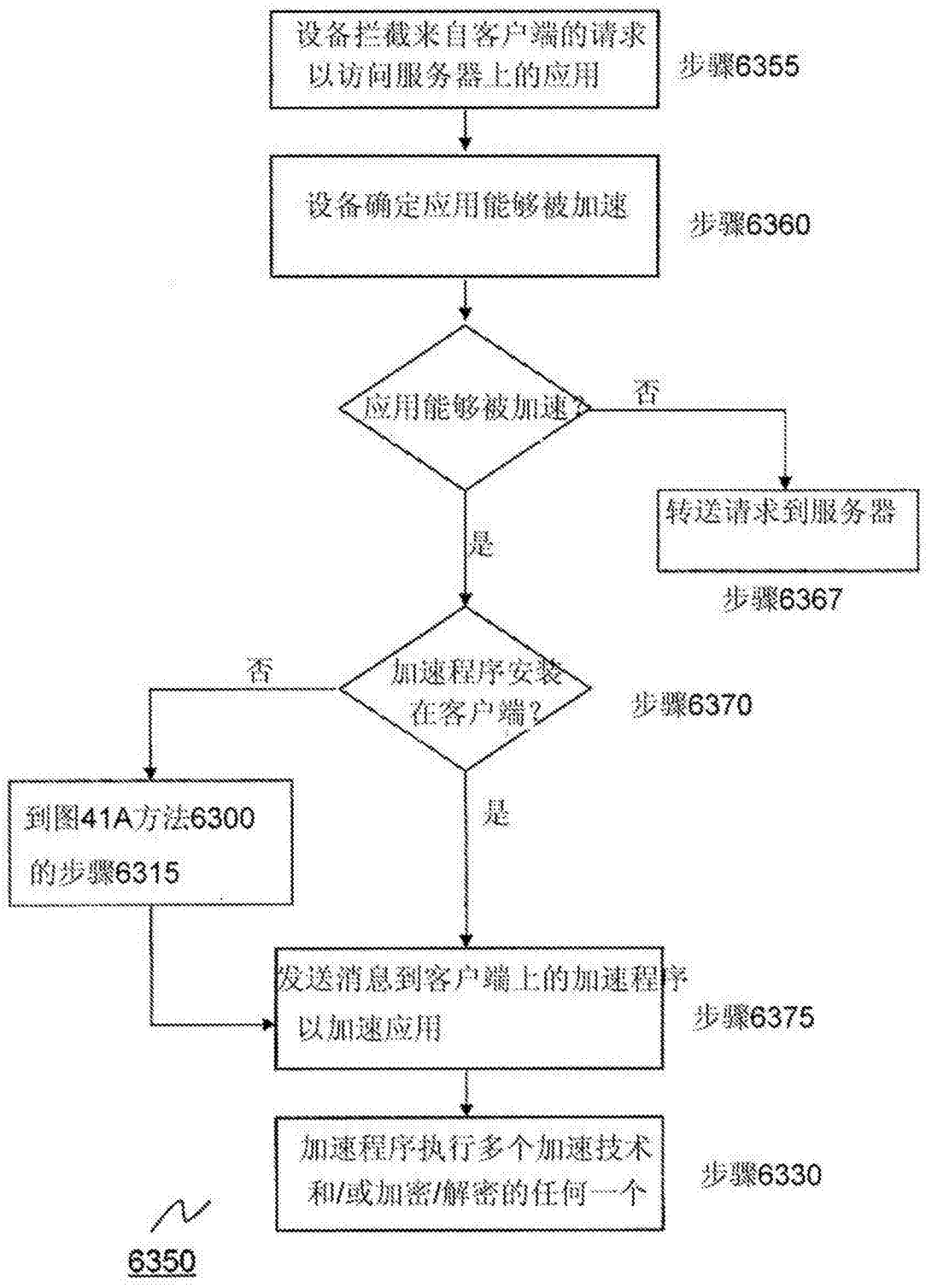

设备拦截来自客户端的请求以访问服务器上的应用
步骤6355
设备确定应用能够被加速
步骤6360
应用能够被加速?
否
转送请求到服务器
步骤6367
是
加速程序安装在客户端?
步骤6370
否
是
到图41A方法6300的步骤6315
发送消息到客户端上的加速程序以加速应用
步骤6375
加速程序执行多个加速技术和/或加密/解密的任何一个
步骤6330
6350

图41B

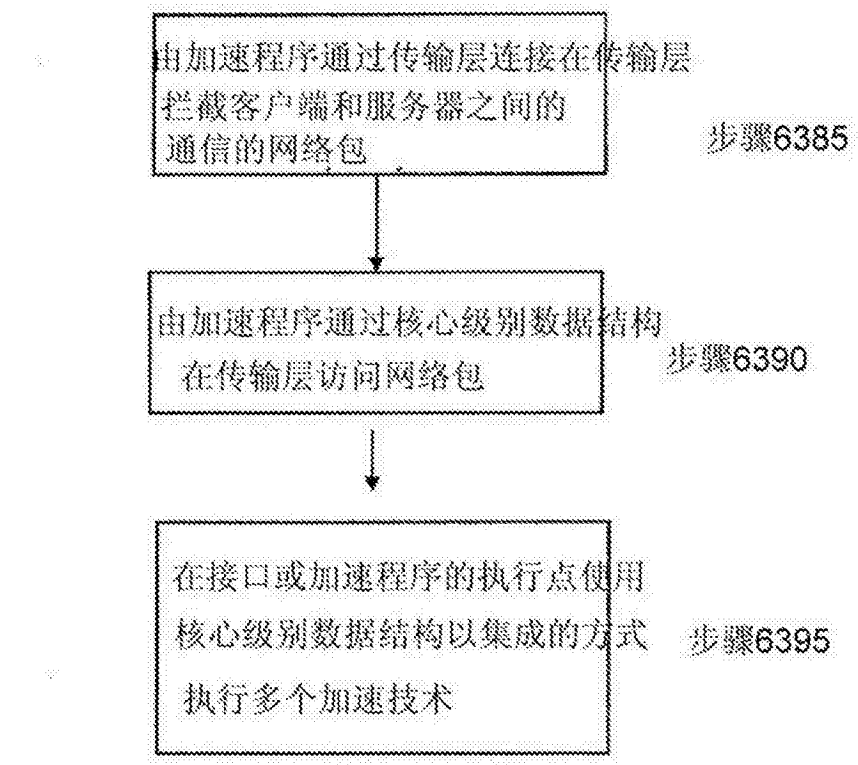

由加速程序通过传输层连接在传输层
拦截客户端和服务器之间的
通信的网络包
步骤6385
由加速程序通过核心级别数据结构
在传输层访问网络包
步骤6390
在接口或加速程序的执行点使用
核心级别数据结构以集成的方式
执行多个加速技术
步骤6395
6380

图41C

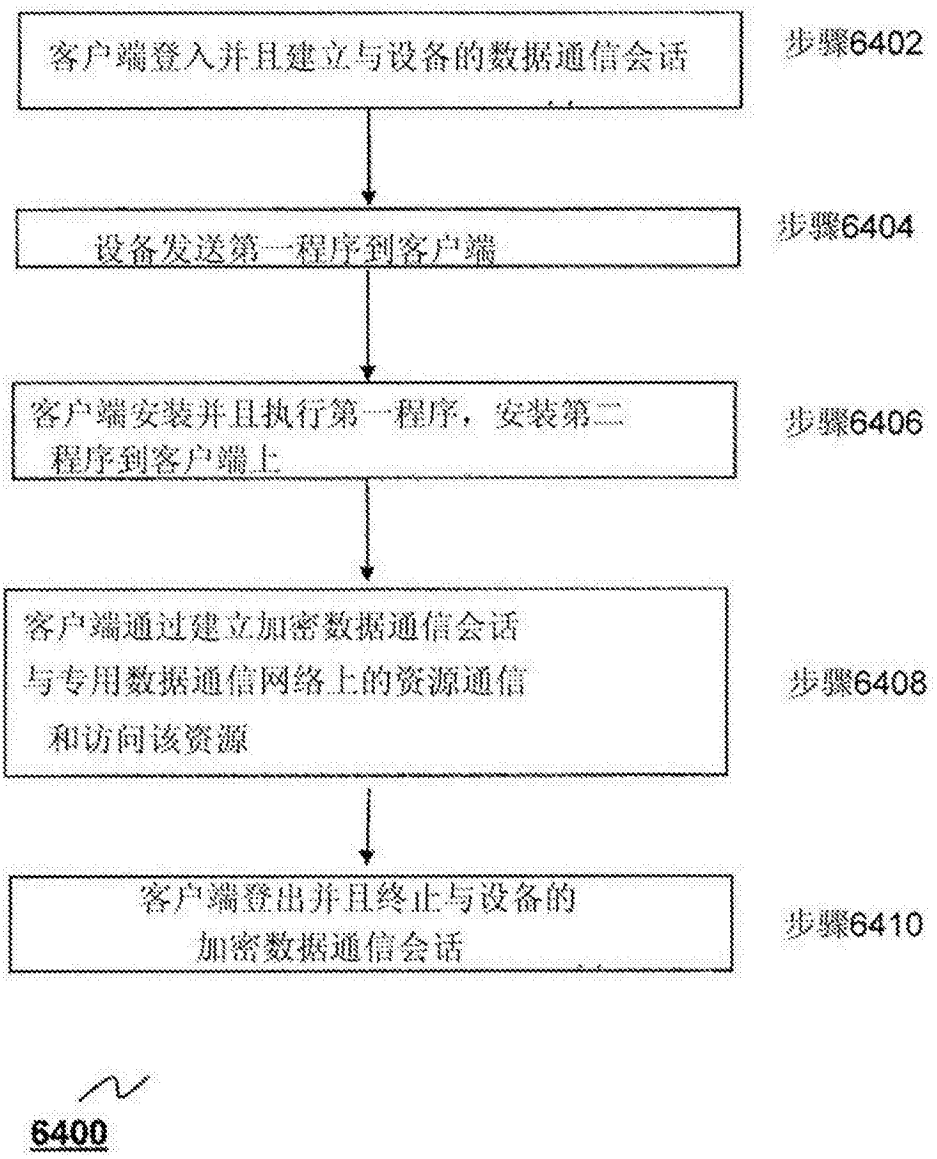

客户端登入并且建立与设备的数据通信会话
步骤6402
设备发送第一程序到客户端
步骤6404
客户端安装并且执行第一程序，安装第二程序到客户端上
步骤6406
客户端通过建立加密数据通信会话与专用数据通信网络上的资源通信和访问该资源
步骤6408
客户端登出并且终止与设备的加密数据通信会话
步骤6410
6400

图42A


客户端应用为客户端操作系统的传输层堆栈做出新的连接或解析域名
步骤6452
调用第二程序的功能
步骤6454
第二程序拦截从客户端指定到专用数据通信网络的资源的通信，并将该通信再路由到第一程序
步骤6456
第一程序通过建立的通信会话终止（或代理）连接隔离有效负载并封装用于分发的有效负载
步骤6458
第一程序通过预先建立的加密通信会话向专用数据通信网络中的设备通过公共数据通信网络发送拦截的通信
步骤6460
设备解密从第一程序接收的通信，并向合适的目标资源转发该解密的通信
步骤6462
目标资源处理解密的通信
步骤6464
目标资源向设备发送响应通信（如果有的话）
步骤6466
设备加密响应通信并通过预先建立的加密通信会话经由公共数据通信网络向客户端中的第一程序发送
步骤6468
第一程序解密响应通信并通过第二程序向合适的客户端应用转发解密的通信
步骤6470
6450

图42B

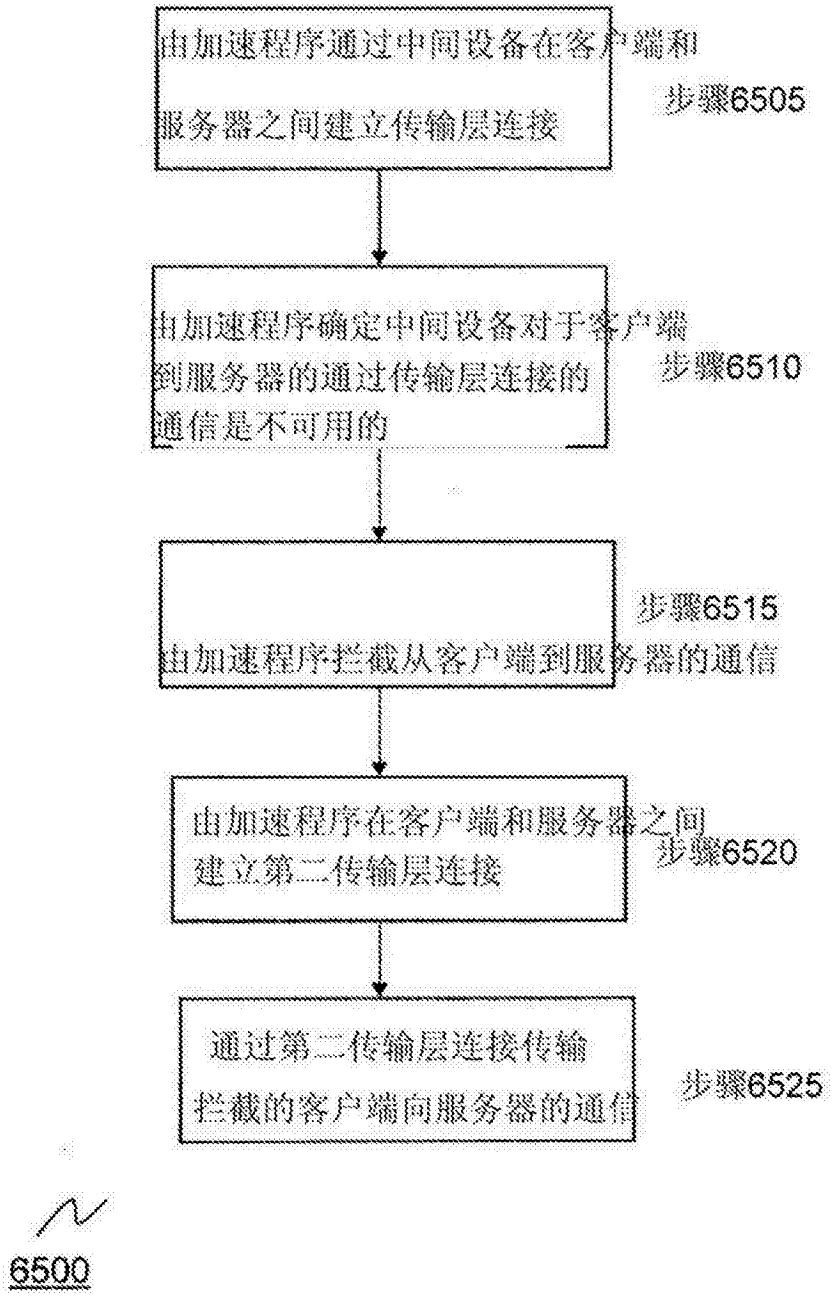

由加速程序通过中间设备在客户端和服务器之间建立传输层连接
步骤6505
由加速程序确定中间设备对于客户端到服务器的通过传输层连接的通信是不可用的
步骤6510
步骤6515
由加速程序拦截从客户端到服务器的通信
由加速程序在客户端和服务器之间建立第二传输层连接
步骤6520
通过第二传输层连接传输拦截的客户端向服务器的通信
步骤6525
6500

图43

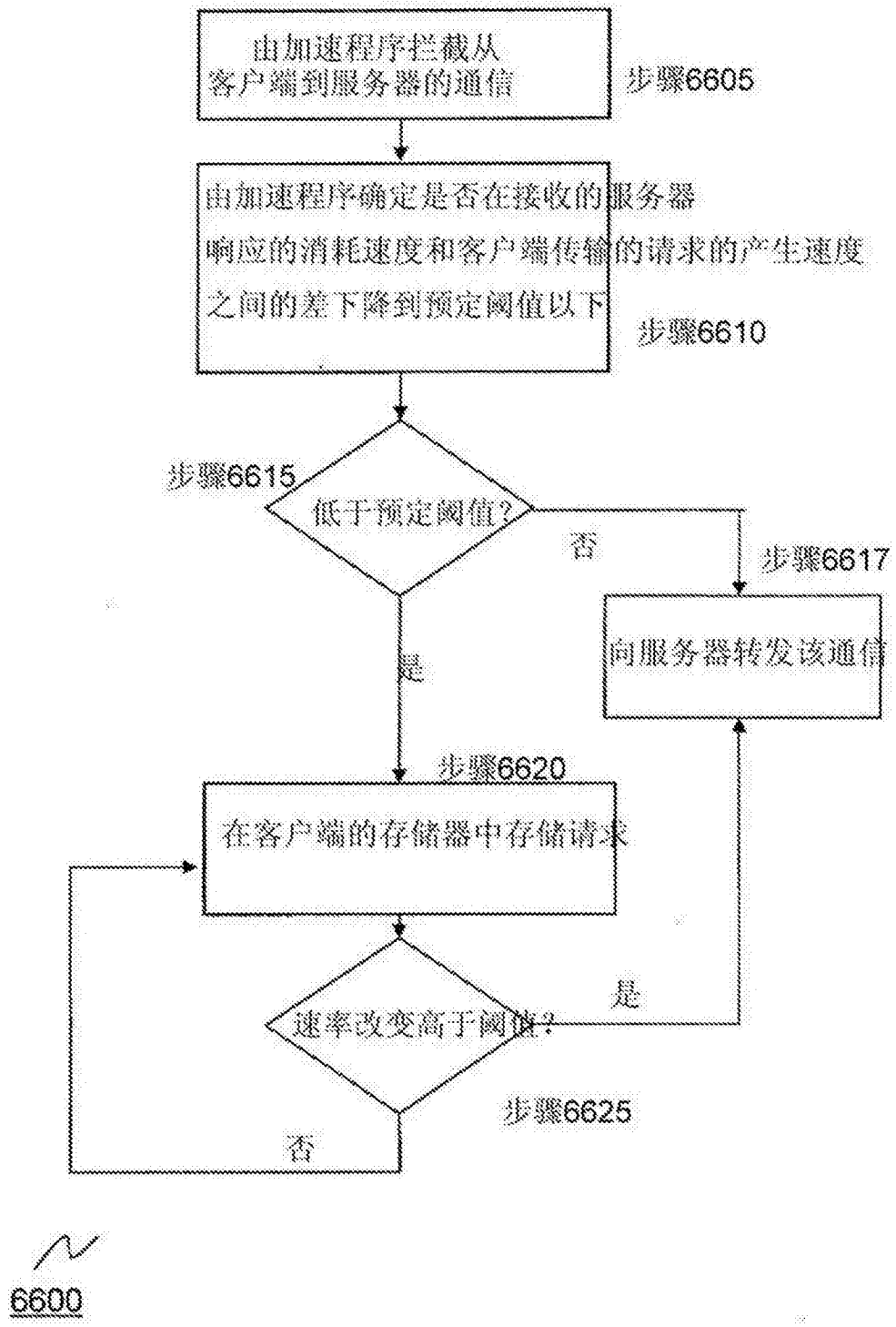

由加速程序拦截从客户端到服务器的通信
步骤6605
由加速程序确定是否在接收的服务器响应的消耗速度和客户端传输的请求的产生速度之间的差下降到预定阈值以下
步骤6610
步骤6615
低于预定阈值?
否
步骤6617
向服务器转发该通信
是
步骤6620
在客户端的存储器中存储请求
速率改变高于阈值?
是
步骤6625
否
6600

图44

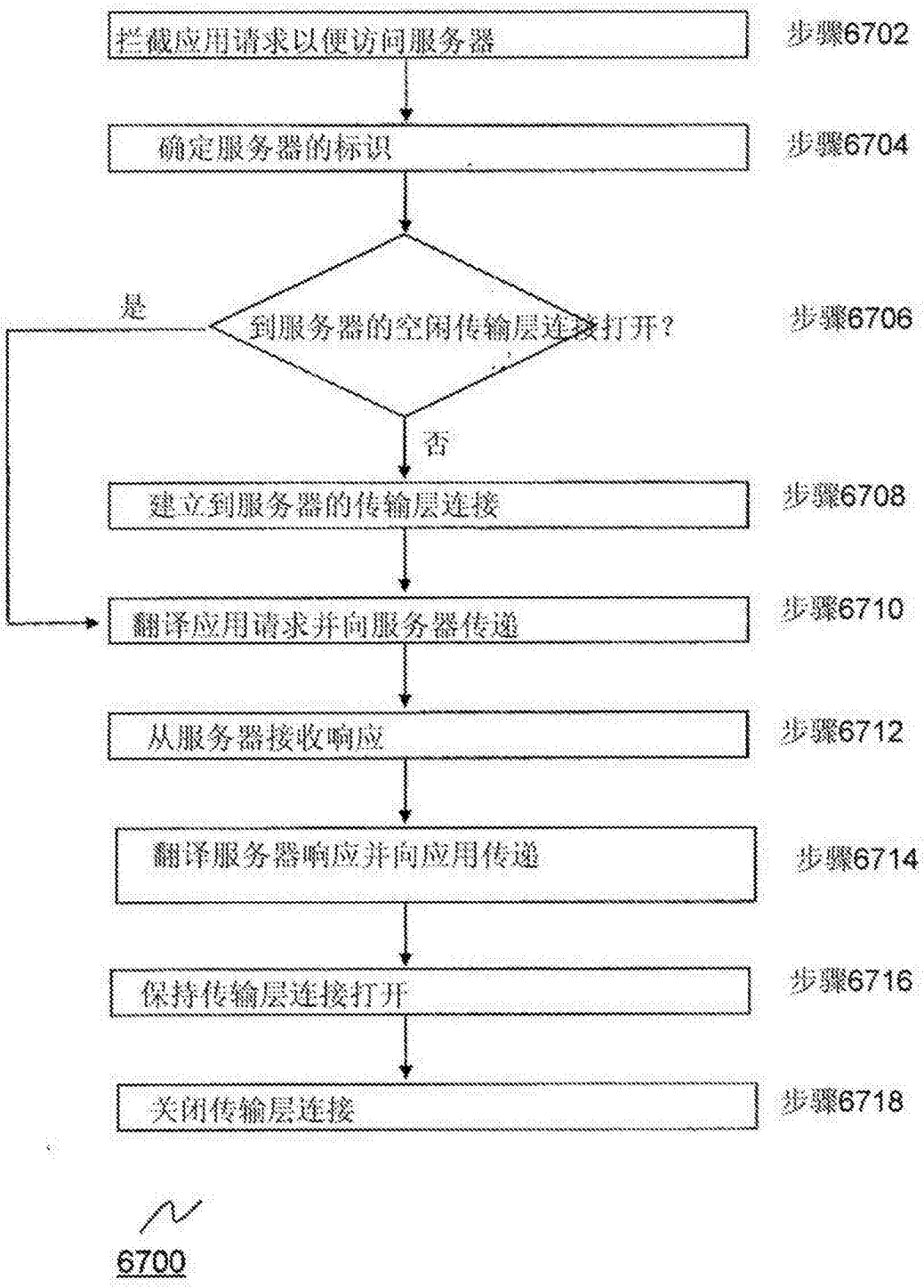

拦截应用请求以便访问服务器
步骤6702
确定服务器的标识
步骤6704
是
到服务器的空闲传输层连接打开？
步骤6706
否
建立到服务器的传输层连接
步骤6708
翻译应用请求并向服务器传递
步骤6710
从服务器接收响应
步骤6712
翻译服务器响应并向应用传递
步骤6714
保持传输层连接打开
步骤6716
关闭传输层连接
步骤6718
6700

图45A

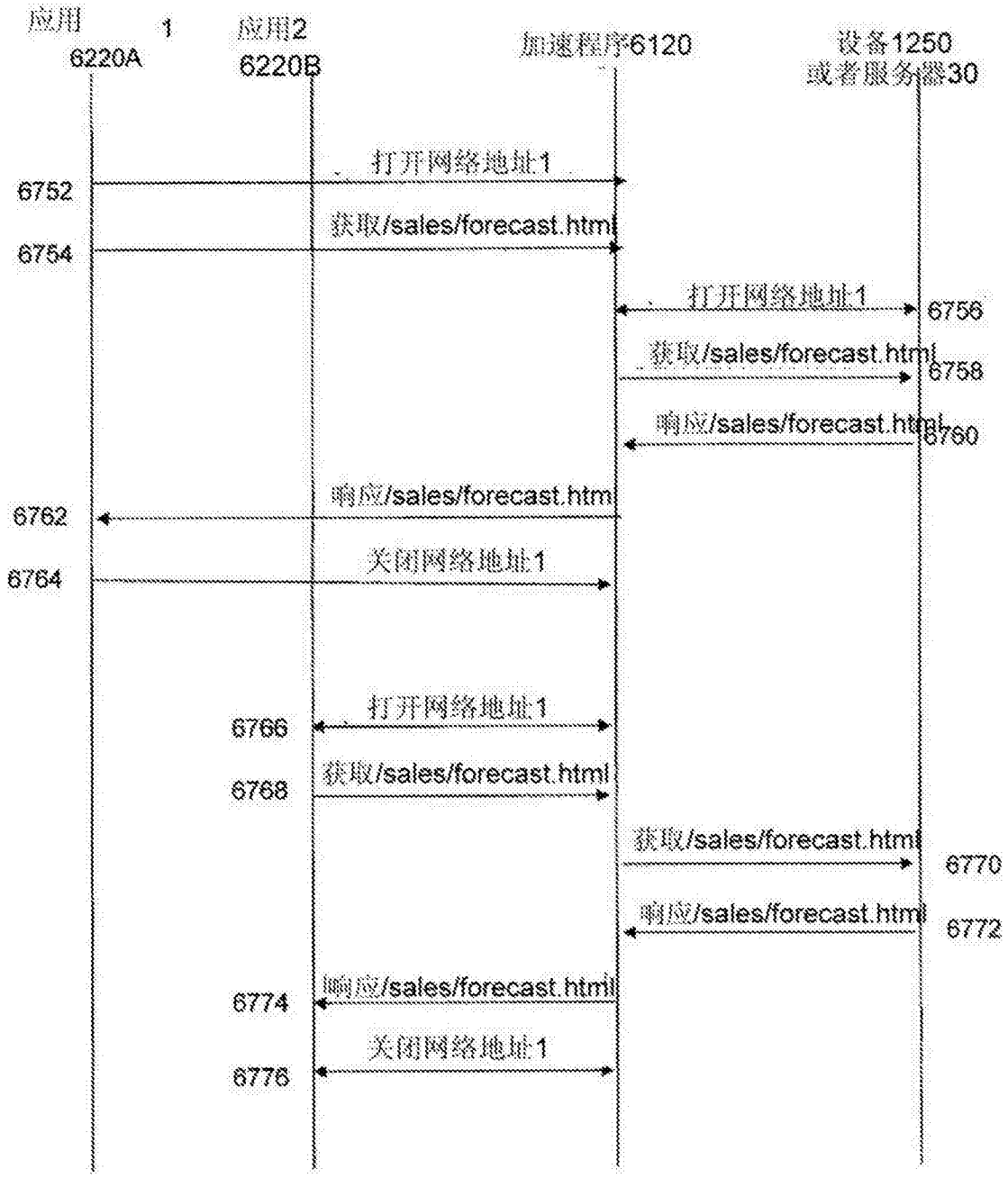

应用 1
6220A
应用2
6220B
加速程序6120
设备1250
或者服务器30
打开网络地址1
6752
获取/sales/forecast.html
6754
打开网络地址1
6756
获取/sales/forecast.html
6758
响应/sales/forecast.html
6760
响应/sales/forecast.html
6762
关闭网络地址1
6764
打开网络地址1
6766
获取/sales/forecast.html
6768
获取/sales/forecast.html
6770
响应/sales/forecast.html
6772
响应/sales/forecast.html
6774
关闭网络地址1
6776

图45B

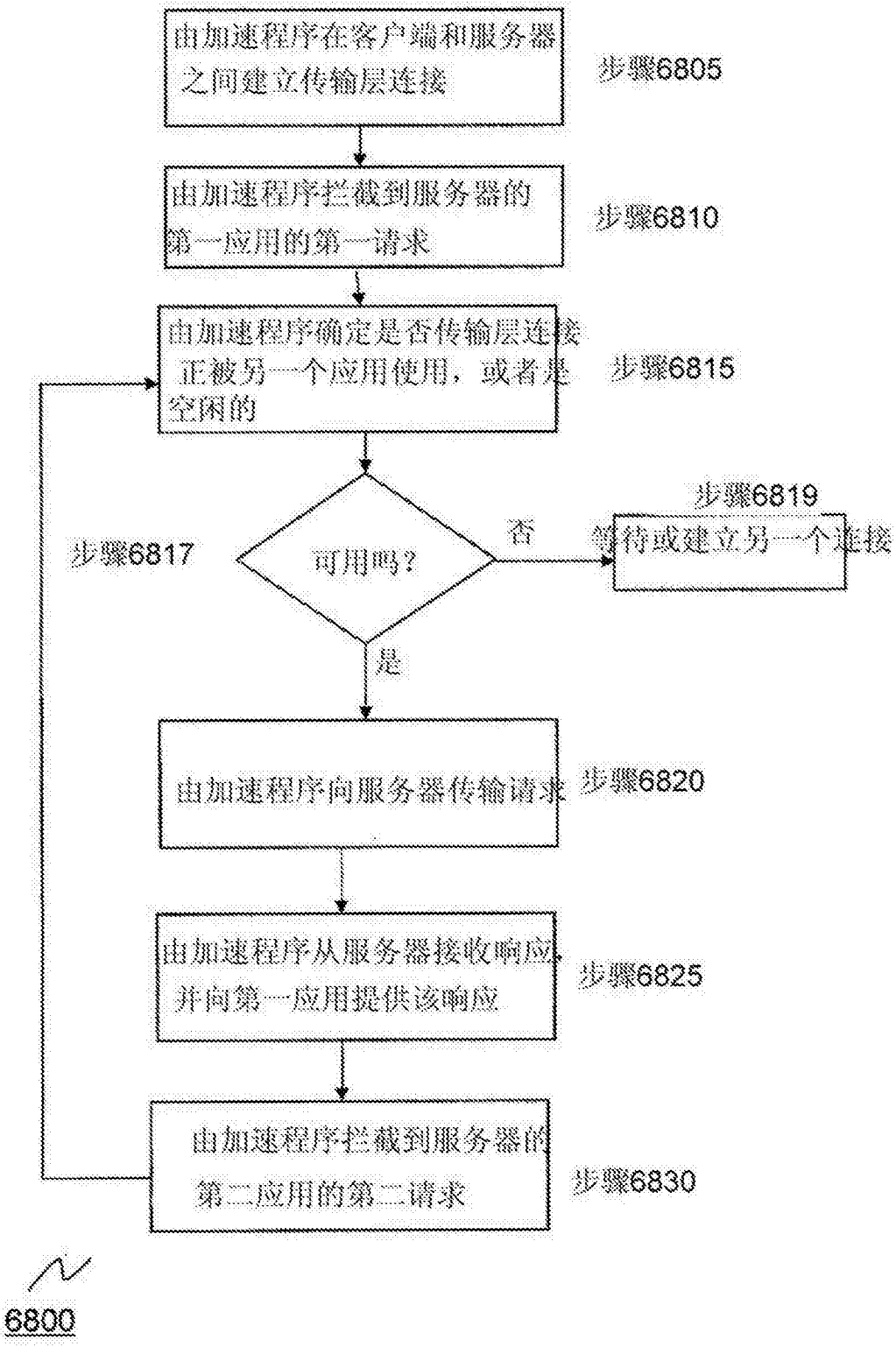

由加速程序在客户端和服务器之间建立传输层连接
步骤6805
由加速程序拦截到服务器的第一应用的第一请求
步骤6810
由加速程序确定是否传输层连接正被另一个应用使用，或者是空闲的
步骤6815
步骤6817
可用吗？
否
步骤6819
等待或建立另一个连接
是
由加速程序向服务器传输请求
步骤6820
由加速程序从服务器接收响应，并向第一应用提供该响应
步骤6825
由加速程序拦截到服务器的第二应用的第二请求
步骤6830
6800

图46

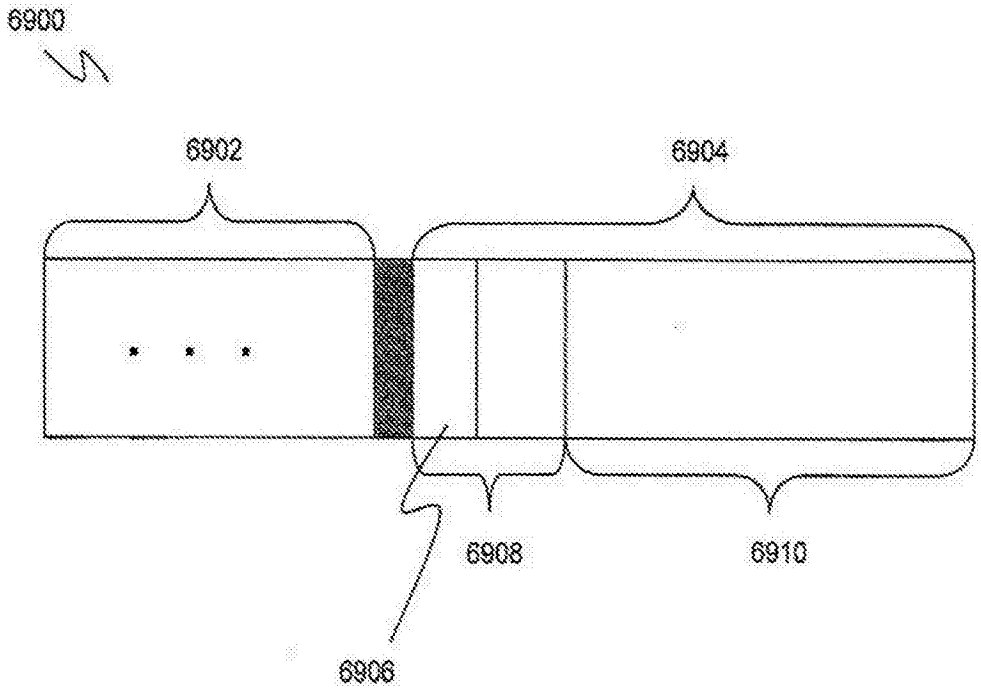


图47

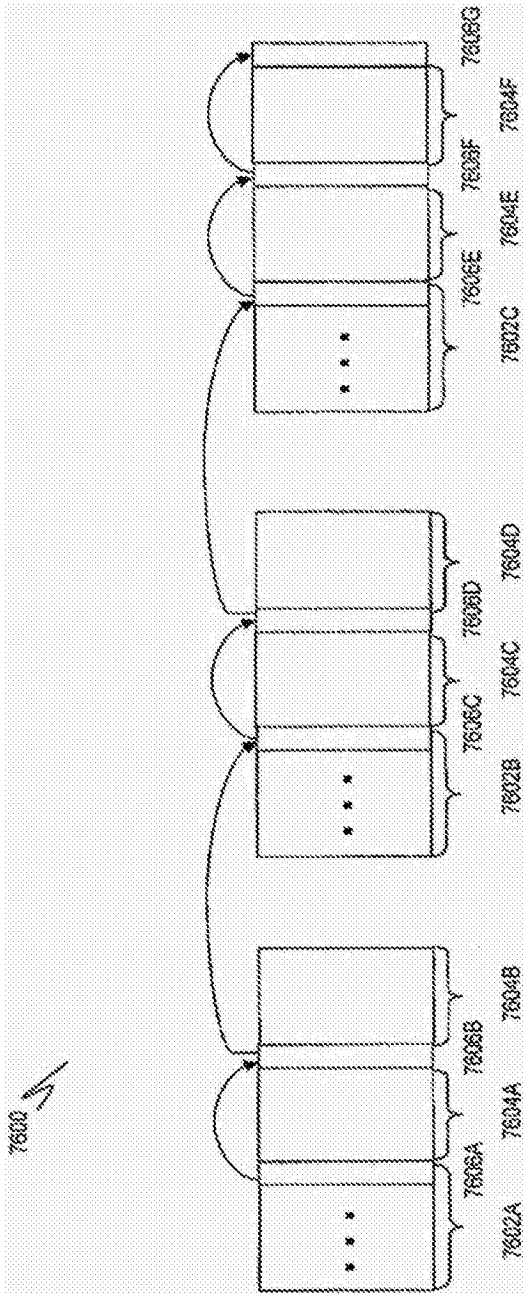

7600
7602A
7606A
7604A
7606B
7604B
7602B
7606C
7604C
7606D
7604D
7602C
7606E
7604E
7606F
7604F

图48

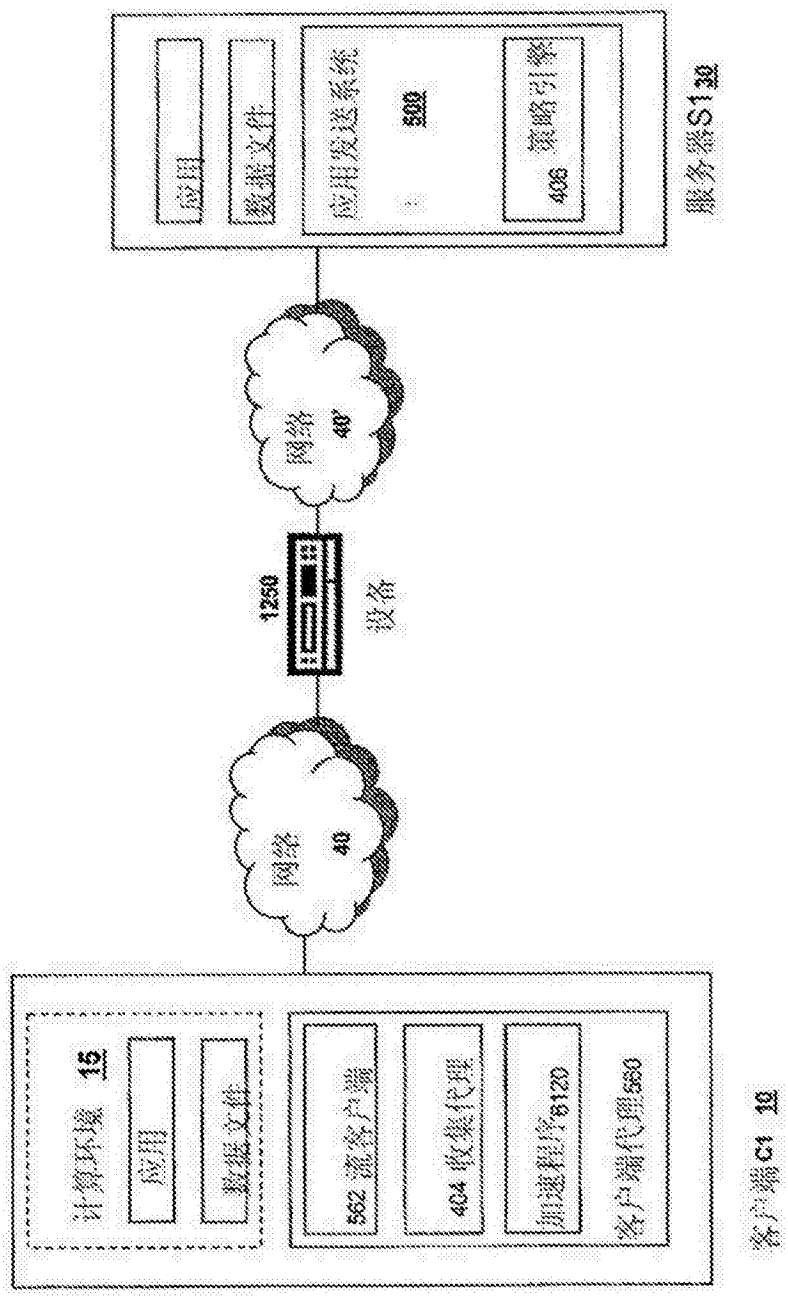

服务器S1 30
应用
数据文件
应用发送系统
502
策略引擎
406
网络
40'
1250
设备
网络
40
客户端C1 10
计算环境 15
应用
数据文件
562 流客户端
404 收集代理
加速程序 120
客户端代理 560

图49A

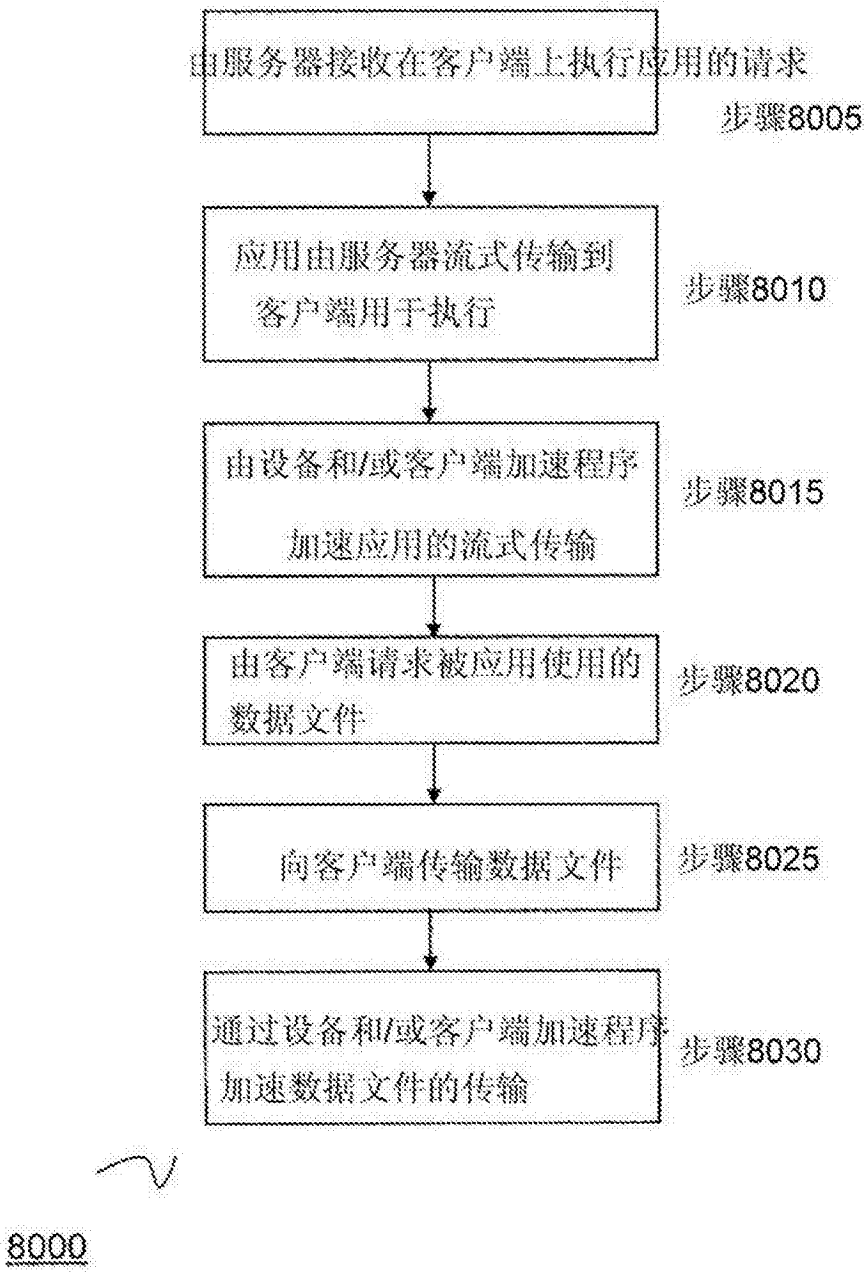

由服务器接收在客户端上执行应用的请求
步骤8005
应用由服务器流式传输到客户端用于执行
步骤8010
由设备和/或客户端加速程序加速应用的流式传输
步骤8015
由客户端请求被应用使用的数据文件
步骤8020
向客户端传输数据文件
步骤8025
通过设备和/或客户端加速程序加速数据文件的传输
步骤8030
8000

图49B