[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
G09G 5/373
G06F 9/445

[12] 发明专利申请公开说明书

[21] 申请号 03808862.2

[43] 公开日 2005 年 10 月 19 日

[11] 公开号 CN 1685395A

[22] 申请日 2003.4.17 [21] 申请号 03808862.2
[30] 优先权
[32] 2002. 4.19 [33] SG [31] 200202352 - 1
[86] 国际申请 PCT/SG2003/000089 2003.4.17
[87] 国际公布 WO2003/090200 英 2003.10.30
[85] 进入国家阶段日期 2004.10.19
[71] 申请人 雷迪克斯私人有限公司
地址 新加坡新加坡市
[72] 发明人 钱德拉塞克·拉特阿克里沙南
韩恭珑

[74] 专利代理机构 北京东方亿思知识产权代理有限责任公司
代理人 王 怡

权利要求书 6 页 说明书 22 页 附图 20 页

[54] 发明名称 用于使用多个应用的系统和方法

[57] 摘要

本发明公开了一种用于服务器的系统，用于将数据以流的形式发送到用户机器，其中所述数据作为数据流被发送到用户机器以显示在用户机器上，所述数据的分辨率由用户机器显示数据的能力决定。 对于用户机器，所述数据可以被用来操作应用，其中所述应用在服务器上执行，并且用户机器操作和显示应用所需的数据被发送到用户机器。 用户机器具有显示设备，以流的形式发送的数据根据显示设备的分辨率要求而发送，所述显示设备充当服务器的显示设备。 使用了一种操作系统，以使服务器能够执行在显示设备上进行显示的应用，该操作系统以单操作层体系结构存在于用户机器中。 所述数据可以是将显示在用户机器的显示设备上的 HTML 媒体文件，该文件被服务器从媒体格式转换到在服务器和用户机器之间达成一致的通用媒体格式，和/或在被发送到用户机器之前进行尺寸调整。 所述应用在被保护的环境中执行，在该环境中实现访问控制以将所述应用的访问限制到系统的至少一个受限区域中。 服务器可以将附带了用于在用户机器上自动安装的指令的安装发送到用户机器，所述指令在被发送到用户机器之前与所述安装封装在一起，以便使用户机器一旦接收到所述安装，就可以解包所述安装和所述指令，执行所述指令，并启动在用户机器上的安装。

ISSN1008-4274

1. 一种用于服务器的系统，用于将数据以流的形式发送到用户机器，其中所述数据被作为流发送到所述用户机器以用于在所述用户机器上进行显示和操纵，所述数据所具有的分辨率由所述用户机器显示所述数据的能力决定。

2. 一种用于服务器的系统，用于使用户机器能够操作和操纵应用，其中所述应用在所述服务器上被执行，并且所述用户机器对所述应用进行操作和显示所需的数据被作为数据流发送到所述用户机器，以用于在所述用户机器上显示，所述数据所具有的分辨率由所述用户机器操作和显示所述应用的能力决定。

3. 一种应用服务提供者操作系统，其中在服务器上执行应用，所述服务器以流的形式发送用于在用户机器的显示设备上进行显示和操纵的数据，所述数据是根据所述显示设备的分辨率要求来发送的，所述显示设备充当所述服务器的显示设备。

4. 一种软件系统，用于使服务器能够执行用于在用户机器的显示设备上进行显示和操纵的应用，所述软件系统以单操作层体系结构存在于所述用户机器中。

5. 如权利要求 4 所述的软件系统，其中所述软件系统包括用于在所述用户机器上进行操作的平台；所述平台包括作为操作系统而操作的平台引擎。

6. 如权利要求 5 所述的软件系统，其中所述操作系统是用于安全性、驱动程序支持、功率管理、引导加载程序和文件系统。

7. 如权利要求 4 所述的软件系统，其中所述单操作层体系结构被用在所述服务器中。

8. 一种用于服务器的系统，用于将数据下载到用户机器，从而使媒体文件能够被显示在所述用户机器的显示设备上，所述媒体文件被所述服务器从媒体格式转换到在所述服务器和所述用户机器之间达成一致的通用媒体格式。

9. 如权利要求 8 所述的系统，其中使用了不变的压缩率，以允许所述数据的实时流发送。

10. 一种用于服务器的系统，用于将数据下载到用户机器，从而使得可以在所述用户机器的显示设备上显示 HTML 文件，所述服务器包括 HTML 尺寸调整服务器，用于在将所述文件发送给所述用户机器之前调整所述文件的尺寸。

11. 如权利要求 10 所述的系统，其中在所述 HTML 文件中的任何图像都被进行尺寸调整，以使其能够被完整地显示在所述显示设备上。

12. 如权利要求 10 或 11 所述的系统，其中传递所述 HTML 文件并在所述服务器上修改所述 HTML 文件的代码，用于使所述 HTML 文件能够显示在所述显示设备上。

13. 如权利要求 3 所述的系统，其中多个应用在所述服务器上被执行，在所述服务器上执行的所有应用都在单一操作系统之下执行，使得将显示和所述数据以流的形式发送到所述显示设备，而无需所述多个应用启动其本地操作系统，也不需要所述多个应用的本地操作系统。

14. 一种用于服务器的系统，用于使用户机器能够操作在服务器上执行的应用，其中所述应用在被保护的环境中被执行，在该环境中实现了访问控制，从而将所述应用的访问限制到所述系统的至少一个受限区域中。

15. 如权利要求 14 所述的系统，其中所述应用在执行前被复制到所述被保护的环境中。

16. 一种用于服务器的系统，该服务器提供了到用户机器的安装，由所述服务器发送到所述用户机器的所述安装附带了用于在所述用户机器上进行自动安装的指令，在被发送到所述用户机器之前，所述指令被与所述安装封装在一起，使得所述用户机器一旦接收到所述安装，就可以解包所述安装和所述指令，执行所述指令，并且启动在所述用户机器上的所述安装。

17. 如权利要求 16 所述的系统，其中所述安装是设备驱动程序。

18. 如权利要求 17 所述的系统，其中所述设备驱动程序文件被复制到系统文件位置，并且所述系统的设置被更新。

19. 如权利要求 16 到 18 中的任意一个所述的系统，其中为在所述用户机器上使用的设备驱动程序安装保留了一份记录，使得被较频繁使用的设备驱动程序被保存在所述服务器的存储器中。

20. 如权利要求 19 所述的系统，其中所述存储器是只读存储器。

21. 如权利要求 16 所述的系统，其中所述安装是对在所述用户机器上操作的操作系统的更新。

22. 如权利要求 16 到 21 中的任意一个所述的系统，其中在所述安装中的新文件被复制到所述服务器。

23. 如权利要求 2 所述的系统，其中所述用户机器包括显示设备，该显示设备为所述服务器充当显示设备。

24. 如权利要求 23 所述的系统，其中多个应用在所述服务器上被执行，在所述服务器上执行的所有应用都在单一操作系统之下执行，使得将显示和所述数据以流的形式发送到所述显示设备，而无需所述多个应用启动其本地操作系统，也不需要所述多个应用的本地操作系统。

25. 如权利要求 2、23 或 24 中的任意一个所述的系统，其中在所述用户机器中，系统以单操作层体系结构来操作软件。

26. 如权利要求 25 所述的系统，其中所述软件包括用于在所述用户机器上进行操作的平台；所述平台包括作为操作系统而操作的平台引擎。

27. 如权利要求 26 所述的系统，其中所述操作系统是用于安全性、驱动程序支持、功率管理、引导加载程序和文件系统。

28. 如权利要求 25 所述的系统，其中所述单操作层体系结构被用在所述服务器中。

29. 如权利要求 1 所述的系统，其中所述数据是将要显示在所述用户机器的显示设备上的媒体数据，HTML 媒体文件被所述服务器从媒体格式转换到在所述服务器和所述用户机器之间达成一致的通用媒体格式。

30. 如权利要求 29 所述的系统，其中使用了不变的压缩率，以允许所述数据的实时流发送。

31. 如权利要求 1、29 或 30 中的任意一个所述的系统，其中所述服务器包括 HTML 尺寸调整服务器，用于在将 HTML 文件发送给所述用户机

器之前调整所述 HTML 文件的尺寸。

32. 如权利要求 31 所述的系统，其中在所述 HTML 文件中的任何图像都被进行尺寸调整，以使其能够被完整地显示在所述显示设备上。

33. 如权利要求 31 或 32 所述的系统，其中传递所述 HTML 文件并在所述服务器上修改所述 HTML 文件的代码，用于使所述 HTML 媒体文件能够显示在所述显示设备上。

34. 如权利要求 2 或权利要求 23 到 28 中的任意一个所述的系统，其中多个应用在所述服务器上被执行，在所述服务器上执行的所有应用都在单一操作系统之下执行，使得将显示以流的形式发送到所述显示设备，而无需所述多个应用启动其本地操作系统。

35. 如权利要求 2 或权利要求 23 到 28 或权利要求 34 中的任意一个所述的系统，其中所述应用在被保护的环境中被执行，在该环境中实现了访问控制，从而将所述应用的访问限制到所述系统的至少一个受限区域中。

36. 如权利要求 34 所述的系统，其中所述应用在执行前被复制到所述被保护的环境中。

37. 如权利要求 3 或 13 所述的系统，其中所述系统包括用于在所述用户机器上进行操作的平台；所述平台包括作为操作系统而操作的平台引擎。

38. 如权利要求 37 所述的系统，其中所述操作系统以单操作层体系结构存在于所述用户机器中。

39. 如权利要求 38 所述的系统，其中所述操作系统是用于安全性、驱动程序支持、功率管理、引导加载程序和文件系统。

40. 如权利要求 38 所述的系统，其中所述单操作层体系结构被用在所述服务器中。

41. 如权利要求 3、13 或 37 到 40 中的任意一个所述的系统，其中所述应用在被保护的环境中被执行，在该环境中实现了访问控制，从而将所述应用的访问限制到所述系统的至少一个受限区域中。

42. 如权利要求 41 所述的系统，其中所述应用在执行前被复制到所述被保护的环境中。

43. 如权利要求 8 或 9 所述的系统，其中所述数据是将要被显示在所述用户机器的显示设备上的 HTML 文件，所述服务器包括 HTML 尺寸调整服务器，用于在将所述文件发送给所述用户机器之前调整所述文件的尺寸。

44. 如权利要求 43 所述的系统，其中在所述 HTML 文件中的任何图像都被进行尺寸调整，以使其能够被完整地显示在所述显示设备上。

45. 如权利要求 43 或 44 所述的系统，其中传递所述 HTML 文件并在所述服务器上修改所述 HTML 文件的代码，用于使所述 HTML 文件能够显示在所述显示设备上。

46. 如权利要求 1、8、9、29 到 33 或 43 到 45 中的任意一个所述的系统，其中多个应用在所述服务器上被执行，在所述服务器上执行的所有应用都在单一操作系统之下执行，使得将所述数据以流的形式发送到所述显示设备，而无需所述多个应用启动其本地操作系统。

47. 如权利要求 4 到 7、25 到 28，或 38 到 40 中的任意一个所述的系统，其中所述单操作层体系结构包括用于提供软件接口的引擎执行器。

48. 如权利要求 4 到 7、25 到 28、38 到 40，或 47 中的任意一个所述的系统，其中所述单操作层体系结构包括用于提供本地硬件支持的引擎监听器。

49. 如权利要求 4 到 7、25 到 28、38 到 40、47 或 48 中的任意一个所述的系统，其中所述单操作层体系结构不具有软件层。

50. 如权利要求 4 到 7、25 到 28、38 到 40，或 47 到 49 中的任意一个所述的系统，其中应用编程接口被翻译为命令。

51. 如权利要求 4 到 7、25 到 28、38 到 40，或 47 到 50 中的任意一个所述的系统，其中所述用户机器能够启动、执行、操纵、监控和停止在所述服务器上的应用。

52. 如权利要求 4 到 7、25 到 28、38 到 40，或 47 到 51 中的任意一个所述的系统，其中所述平台识别被预先规划的硬件，并且将不会对未经授权的硬件起作用。

53. 如权利要求 10 到 12、31 到 33，或 43 到 47 中的任意一个所述的

系统，其中通过向所述文件中的任何不具有宽度标签和高度标签的对象添加宽度标签和高度标签来进行尺寸调整，并且修改所述宽度标签和高度标签中的值，从而使得可以根据所述显示设备的分辨率要求而将所述对象显示在所述显示设备上。

54. 如权利要求 53 所述的系统，其中用所述宽度标签值除以 800，并乘以所请求的分辨率的宽度。

55. 如权利要求 53 或 54 所述的系统，其中用所述高度标签值除以 600，并乘以所请求的分辨率的高度。

56. 如权利要求 3、13、37 到 42 或 53 到 55 中的任意一个所述的系统，其中由所述用户机器将所述分辨率要求提供给所述服务器。

57. 如权利要求 8、9、29 到 33，或 43 到 47 中的任意一个所述的系统，其中所述通用媒体格式是预定的。

58. 如权利要求 57 所述的系统，其中所述通用媒体格式是一种流格式，并且具有不变的压缩率。

59. 如权利要求 57 或 58 所述的系统，其中到所述通用媒体类型的转换是通过代码转换进行的。

60. 如权利要求 57 或 58 所述的系统，其中到所述通用媒体类型的转换是通过首先将 HTML 媒体文件解码和解压缩到原始数据而进行的。

61. 一种在处理器上可操作的软件布置，所述软件布置包括计算机程序，该计算机程序配置所述处理器，用于实现如权利要求 1 到 60 中的任意一个所述的系统中的一个或多个系统。

62. 一种计算机系统，该计算机系统包括一个或多个装置，用于实现相应的如权利要求 1 到 60 中的任意一个所述的系统中的一个或多个系统。

用于使用多个应用的系统和方法

技术领域

本发明涉及用于使用多个应用的系统和方法，并且本发明尤其，但不专门地涉及在移动计算环境中所使用的这种系统和方法。

定义

遍及本说明书，提及机器（machine）应被理解为提及了包括个人计算机（“PC”）、桌上型计算机、膝上型计算机、笔记本计算机、个人数字助理（“PDA”）和移动/蜂窝/手持电话（“mobile phone”）。

背景技术

大多数的计算机应用都被希望能够在个人计算机或桌上型计算机上使用。因此它们通常都包括非常大的文件，并且是 RAM 密集型的。为了处理到用户机器的一般通信，需要该机器具有已安装的若干应用，并且可能需要同时打开这些应用，所述的到用户机器的一般通信例如是通过电子邮件，或用于访问万维网（World Wide Web），或任意形式的媒体重放。对于在多数 PC 中可获得的硬盘和 RAM，这种需求通常不是问题。而对于很多笔记本，并且尤其对于 PDA 和移动电话，这是一个重大的问题。因此，它们通常使用这些应用的缩小版本，或不能使用全部的相关应用。这样做会导致很多麻烦，特别当使用移动电话或 PDA 与 PC 或类似机器进行通信时尤为如此。这还意味着 PDA 和移动电话的使用被局限于它们能够运行的那些应用。在单一操作系统内与为其他操作环境而编写的应用协同工作，这种能力并非任何类型的机器都能具备。应用是专为一种环境而编写的，无法实现互操作性。

现有技术的考虑

移动计算环境缺乏一种能真正增强该产业能力的解决方案。还不存在能够发挥移动硬件的潜力的软件，尤其当与桌上型硬件相比较时，移动硬

件的潜力受到极大的限制。桌上型计算环境的操作系统已经被精简（strip down），并被嵌入到移动设备中。在此过程中，它继承了该操作系统的缺陷。一些当前提供的嵌入式系统包括 Win CE、Palm OS、Symbian 和 Java OS。这些已精简操作系统之间的实际不同仅仅在于它们来自不同的供应商。

它们都模仿相似的体系结构，具有不同的接口。这些嵌入式操作系统都模仿桌上型计算，具有从桌上型环境访问的多个应用，并在不同的层上执行这些应用。在个人计算环境中采用了与 Windows 和 Mach OS 所推广的一模一样的模型和体系结构。由于在个人计算环境中可以获得大量的处理功率和复杂的硬件，例如硬盘驱动器、高存储模块、强大的视频卡和极好的声卡，因此该模型非常适合于个人计算环境。在移动设备的嵌入式空间中不存在这样的硬件能力。在该空间中，硬件受到很大限制。低处理能力、小存储容量、有限的存储模块都是在嵌入式空间中一般会发现的问题。尽管使用了复杂的硬件，但是操作系统的未经删节的版本也没有为用户提供最优的性能和体验。用户遭遇到崩溃（crash）、运行延迟，还需要处理不同的格式，这些格式要求来自不同供应商的不同播放器。

这就导致精简版本的性能等级无法接受。因此，无法实现其赋予用户“活动式”（on the go）计算能力的预期目的。

使用多个嵌入到移动设备中的分层的成熟（full-fledged）的操作系统作为缩小版本是一种折衷的办法。

因此，本发明的主要目的在于提供一种用于在用户机器，尤其是诸如 PDA 或移动电话的移动机器上使用多个成熟应用的系统和方法，其中，在服务器上，而不是用户机器上执行所述应用。在用户机器上也可以操纵所述应用。

本发明的另一个目的在于提供一种这样的系统，在该系统中，凭借用户机器的显示设备充当服务器的显示设备，以允许在用户设备上进行本地操纵（manipulation）。

本发明的最后一个目的在于提供一种这样的系统，在该系统中，根据用户机器的显示设备的分辨率，以一种自动化的方式，利用服务器发送用

于显示在该显示设备上的数据。

发明内容

考虑到上述及其他目的，本发明提供了一种用于服务器的系统，用于将数据以流的形式发送到用户机器，其中数据被作为流发送到用户机器以用于在用户机器上进行显示和操纵，所述数据所具有的分辨率由所述用户机器显示数据的能力决定。数据包括多个字节的代码。

在另一种形式中，本发明提供了一种用于服务器的系统，用于使用户机器能够操作应用，其中所述应用在服务器上被执行，并且用户机器对所述应用进行操作和显示所需的数据被作为数据流发送到所述用户机器，以用于在所述用户机器上显示，所述数据所具有的分辨率由所述用户机器操作和显示所述应用的能力决定。

本发明还提供了一种应用服务提供者操作系统，其中在服务器上执行应用，所述服务器以流的形式发送用于在用户机器的显示设备上显示的数据，所述数据是根据显示设备的分辨率要求来发送的，所述显示设备充当所述服务器的显示设备。在服务器上可以执行多个和各种应用，在服务器上执行的所有应用都在单一操作系统之下执行，使得可以将数据流发送到所述显示设备，而无需所述多个应用启动其本地操作系统，也不需要存在其本地操作系统。

在另一种形式中，本发明提供了一种软件系统，用于使服务器能够执行在用户机器的显示设备上进行显示和控制的应用，所述软件系统以单操作层体系结构存在于用户机器中。所述软件系统包括用于在用户机器上进行操作的平台；所述平台包括作为操作系统而操作的平台引擎。优选地，所述平台是用于安全性、驱动程序支持、功率管理、引导加载程序和文件系统。所述单操作层体系结构也可以被用在服务器中。

本发明还提供了一种用于服务器的系统，用于将数据以流的形式发送到用户机器，从而使任何媒体文件都能够显示在用户机器的显示设备上，所述媒体文件被服务器从媒体格式转换到在服务器和用户机器之间达成一致的通用媒体格式。通过转换，保证了不变的压缩率，从而允许实时地以

流的形式发送数据。

本发明还提供了一种用于服务器的系统，用于将数据以流的形式发送到用户机器，从而使 HTML 文件能够显示在用户机器的显示设备上，所述服务器包括 HTML 尺寸调整服务器，用于在将 HTML 文件发送给用户机器之前调整该文件的尺寸。可以对 HTML 文件中的任何图像进行尺寸调整，以使其能够被完整地显示在显示设备上。在服务器上对 HTML 文件进行解析和代码修改，从而使该 HTML 文件可以显示在显示设备上。这个过程是以一种自动化的方式执行的。

在另一种形式中，本发明还提供了一种用于服务器的系统，用于使用户机器能够操作在服务器上执行的应用，其中所述应用在被保护的环境中被执行，在该环境中实现了访问控制，从而将应用的访问限制到所述系统的至少一个受限区域中。所述应用在执行前可以被复制到所述被保护的环境中。

在最后一种形式中，本发明提供了一种用于服务器的系统，该服务器提供了到用户机器的安装，由服务器发送到用户机器的所述安装附带了用于在用户机器上进行自动安装的指令，在被发送到用户机器之前，所述指令被与安装封装在一起，使得用户机器一旦接收到所述安装，就可以解包所述安装和所述指令，执行所述指令，并且启动在用户机器上的安装。所述安装可以是设备的驱动程序，在这种情况下，设备驱动程序文件被复制到系统文件位置，并且系统设置被更新。可以为在用户机器上使用的设备驱动程序安装保留一份记录，使得被较频繁使用的设备驱动程序被保存在服务器的存储器中。所述存储器可以是只读存储器。

所述安装可以是对在用户机器上操作的操作系统的更新，在这种情况下，在安装中的新文件被复制到服务器中。

本发明还扩展到本发明的所有形式的所有可能的组合。

附图说明

为了能够全面地理解本发明，并且能够将本发明容易地投入实际应用，这里将参考说明性的附图，以非限制性示例的方式描述本发明的一个

优选实施例，在附图中：

图 1 是关于要在用户机器上使用的平台的总流程图；

图 2 是关于图 1 中的“用户认证”的流程图；

图 3 是关于图 1 中的“浏览 HTML 文件”的流程图；

图 4 是关于图 1 中的“启动多媒体插件”的流程图；

图 5 是关于图 1 中的“启动远端应用”的流程图；

图 6 是关于图 1 中的“系统更新”的流程图；

图 7 是关于平台对输入的响应的总流程图；

图 8 是关于图 7 中的“用户输入”的流程图；

图 9 是关于图 7 中的“系统输入”的流程图；

图 10 是关于图 7 中的“服务器输入”的流程图；

图 11 是关于服务器操作的总流程图（第一部分）；

图 12 是关于服务器操作的总流程图（最后部分）；

图 13 是关于图 11 和 12 中的“用户认证”的流程图；

图 14 是关于图 11 和 12 中的“HTML 文件尺寸调整和缓存”的流程图；

图 15 是关于图 11 和 12 中的“媒体流发送”的流程图；

图 16 是关于启动应用的总流程图（第一部分）；

图 17 是关于启动应用的总流程图（最后部分）；

图 18 是关于图 11 和 12 中的“设备驱动程序请求”的流程图；

图 19 是关于图 11 和 12 中的“操作系统更新”的流程图；

图 20 是关于平台的系统体系结构图；

图 21 是关于服务器引擎的系统体系结构图；以及

图 22 是关于应用容宿（hosting）的图。

具体实施方式

虽然后面的描述是与移动计算环境（移动电话、PDA 等等）相关，但是本发明可以应用于能通过电信网络与服务器进行通信的任何机器。

在以下的描述中，将使用这样的标号，即标号的前两位数字指代该对

象所在的附图。例如，0605 指示图 6 中的对象 5，而 0700 指示图 7。

首先参考图 20，该平台以单操作层体系结构（“SOLA”）构建，并且具有引擎执行器 2001 和引擎监听器 2020。引擎执行器 2001 提供了软件接口 2002，该接口包括因特网接口和远端应用接口。引擎监听器提供了本地的硬件支持接口。

该平台只具有其引擎。平台引擎所提供的外壳（shell）由该平台引擎在本地提供。平台引擎提供了 HTML 和多媒体输出，作为本地的 I/O 支持。这里没有提供 API，因此消除了应用层。这样也减少了第三方的应用能够进行直接的硬件访问的机会。这样，任何的本地访问不得不被平台引擎屏蔽掉。应用不是在平台引擎之上运行的。相反，要创建会话来处理应用，并且应用在远端的服务器上运行。

由于没有应用呼叫任何 API，因此不存在软件层。相反，从远端应用所呼叫的 API 被翻译为在服务器引擎和平台引擎之间共享的命令。这就是为什么没有应用运行在平台引擎之上的原因。在任何时候，平台引擎只接受来自于用户和服务器的指令。另外，允许网络内容呼叫能够被用于硬件处理任务的命令。这种呼叫与 API 呼叫不同，这是因为在每次呼叫所述命令时，都要对命令进行解析，因此不存在由网络内容和远端应用进行的直接的硬件资源访问。

由于应用是运行在服务器上的，所以它们不需要是现有 OS 的精简版本。平台引擎从服务器引擎接收所需的数据。然后将数据显示在用户机器的显示设备（屏幕等等）上。在移动机器上，支持 HTML 4.0 网络内容的能力意味着利用该平台带给用户完全的网络体验。当第一次连接到服务器时，平台向服务器提供用户机器的详细信息（包括其显示设备的分辨率），以便使发送给用户机器的所有数据都将能够由平台进行处理，并能够被显示在用户机器的显示设备上。

通用多媒体格式 2004 允许用户索取任意文件格式的任意富媒体（rich media）内容，并且具有重放内容的已转换版本的能力；这消除了对若干插件的需要，所述插件利用了机器的大量存储器（RAM）。通用多媒体格式 2004 给出了压缩的一致性，使得媒体文件的压缩尽管来自不同的开发

者，也将保持一致性。这允许在所有时间内有效地使用带宽。

由于应用的执行不是在本地端进行的，因此应用的启动、操纵和多远端任务意味着更快的应用处理过程。相反，监控和操纵是在本地进行的。由于在本地端的 OS 不需要存储器分页或程序转换，所以从一种应用到另一种应用的转换相对较快。

外部的外设支持和自动安装意味着用户不需要知道所需的驱动程序和安装指令。相反，由平台来请求所需的驱动程序，并且自动地安装所需的驱动程序。这样，由于服务器知道平台可以支持什么样的驱动程序，并且将提供用于自动安装的必要细节，所以系统驱动程序可以具有很高的完整性。平台可以为外设的使用执行驱动程序的自动安装。

平台引擎的引擎执行器定义了软件能力，而引擎监听器定义了硬件能力。

本地 HTML 解析器 2003 意味着平台引擎具有在本地解析 HTML 的能力。与其他依赖于浏览器应用进行解析并提供 HTML 输出的 OS 不同，平台引擎在本地解析并提供 HTML 输出。这提供了一种 HTML 内容的标准化输出，并且允许在所有移动设备之间保持格式的一致性。

本地 HTML 4.0 解析器 2003 所使用的标准是基于 NCSA Mosaic（TM）和 W3C 标准。但是，本地 HTML 浏览器模块无需外部播放器或插件，就可以读取所有的富媒体内容。给定 URL，浏览器将通过向服务器发送用户机器的显示设备的解析屏幕尺寸和色彩深度说明来请求调整原始 HTML 内容的尺寸。基于已调整过尺寸的 HTML 文件，浏览器查找任何附带了富媒体的内容。如果有，则浏览器将启动通用多媒体插件 2004，该插件将以通用的格式从服务器中取回多媒体内容。

由于该插件在 HTML 解析器内，并且在需要时被使用，所以鉴于该功能的本质，这里使用了术语“插件”。但是，它不是在浏览器或应用内所安装的标准插件。一经要求，它就通过向服务器发送在用户机器处的分辨率，请求调整源媒体文件的尺寸，并将源媒体文件格式转换到通用的媒体格式。该通用的媒体格式是一种标准的媒体格式，并且优选地，该格式具有高压缩率，以允许实时流。由于该格式对过程没有影响，所以它可以是

由平台和服务器两者预先确定的任意格式。

由于只取回一种格式，所以通过这样做可以消除对若干插件的需要。这给出了在媒体格式、分辨率控制和压缩率上的一致性。

平台引擎不允许应用在其之上运行。因此，平台引擎提供了基本的 I/O 接口，包括键盘、扬声器输出和 USB 支持。但是，平台引擎允许创建应用的会话，以便使平台引擎可以控制在服务器上所执行的应用。这是一种被并入 SOLA 中的网络计算技术，以便可以实现存储器使用最优化和快速多任务（multitasking）。通常，远端应用接口允许从用户机器启动、执行、操纵、监控和停止在服务器处的应用。

给定应用的 URL 地址，远端应用接口将随着远端地启动应用的请求一起，发送用户机器的屏幕分辨率。作为回复，远端应用接口将接收到 GUI 指令，用于在应用被成功地启动和执行时，在用户机器的屏幕上显示 GUI 组件。一旦接收到 GUI 指令，远端应用接口就将创建 GUI 组件，并且允许在本地无需服务器的干预而执行组件的交互作用。只有当需要诸如排序或保存文件一类的过程时，远端应用接口才会发送呼叫，该呼叫指示在服务器处需要执行哪部分程序。服务器向远端应用接口发回 GUI 指令，以更新其察看端口。当应用在服务器处被处理时，远端应用接口还可以检查该应用是否呼叫任何命令或硬件请求。服务器引擎和平台引擎两者都能重新定向到属于另一引擎的其他引擎命令。

平台引擎具有内置的硬件接口，并且由于这是嵌入式的环境，因此扩展不是必需的。而且，允许选择硬件扩展支持会损害 OS 的安全性。平台引擎识别预先编程的（pre-programmed）硬件，并且不会对未经授权的第三方硬件起作用。通过这样做，可以防止黑客利用第三方硬件来访问用户机器。

本地硬件支持接口是用于平台引擎的引擎监听器。它的主要任务是监听硬件操作，并进行检测。引擎监听器被编程为“监听”固定的硬件种类。由于不同的硬件使用不同的驱动程序，所以引擎监听器是一个“驱动程序层”，其允许标准化的硬件访问。

这里只准许两种硬件访问方式。第一种是通过呼叫命令。由于每次命

令被呼叫时，都要对其进行解析和解释，所以应用和网络内容都不能直接地访问硬件资源。第二种是通过呼叫传统的 OS API。由于 API 的一个子集被转换为命令，所以以这种方式也可以允许硬件访问。但是，由于只有 API 的一个子集可以被转换，所以利用传统的 API 呼叫方式的硬件访问并不总是被允许。引用了存储器区段的地址和分配、直接端口引用、二进制码的本地传递、钩子（hook）和中断编程的 API 呼叫都是不安全的 API 呼叫，它们将不能被转换为命令。平台引擎支持更高级 API 呼叫的转换，所述更高级 API 呼叫例如是打开 CD-ROM 驱动架、调整音量、打印文档等等。传统的 API 到命令的转换是在服务器处进行的。

由于平台引擎提供了本地的集成硬件支持，所以任意无线支持硬件将被检测并自动安装。这是在服务器的帮助下进行的。服务器帮助平台引擎识别用于被检测到的设备的安装指令。这样，则不再需要用户的干预。

如图 21 所示，服务器以与平台类似的单层结构进行操作。但是，由于服务器操作系统更加复杂，且不是在嵌入式的环境中，因此存在对服务器的扩展。在服务器引擎 2122 之下列出的所有组件都是单独的服务器，只是它们被编程为对从服务器引擎所呼叫的命令进行“监听”。服务器引擎还要协调和重新定向到各个服务器的操作。

服务器还要对平台做出反应，以及对平台进行监听。因此，服务器引擎是在“前线”（front line）中。服务器引擎对其所有服务都不提供本地支持，而是为其所有服务充当接口。服务中的每一种都容宿于它们各自的服务器中。这些服务器可以是多个服务器的服务器组（server suite）中的一部分，或者也可以来自外部方。例如，服务器组的存储数据库服务器可以由“Oracle”数据库服务器来替代。但是，必须保留来自服务器组的存储数据库服务器，因为在其上安装扩展，以使服务器能连接到第三方的数据库服务器。服务器组中的所有服务器彼此间相互作用，因此它们都被集成到服务器引擎。唯一的例外是在需要第三方服务器时。这种情况下，如上所述，在各个服务器上安装扩展，以连接到外部服务器。这适用于电子邮件 2124、存储数据库 2123、管理 2125、因特网网关 2126 和用户认证服务器 2127。

在服务器侧的存储消除了对数据进行同步的需求，因为数据是可以实时得到的。通过在服务器上具有该处理，使得移动机器可以启动为带有大RAM、大硬盘和高速处理器以及高总线速度的 PC 所建立的应用。此外，操作的分离减少了在服务器与用户机器之间的数据流量。

由于服务器处理的是会话数据，而不是资源密集型的应用数据，因此多会话（multi-sessions）意味着服务器可以比传统的应用容宿服务器支持更多的用户。另外，由于应用是在服务器上执行的，没有在用户机器上运行的应用，而通常病毒将其自身依附于应用的指定部分，因此无法直接在用户机器处进行病毒攻击。由于用户不得不在服务器处经过因特网网关，因此由于服务器可以通过负载均衡技术和防火墙安全性来进行跟踪和监控，所以可以以一种更有效的方式来处理安全问题。

服务器利用 MSISDN 信息来识别用户中的每一个，该信息是唯一且安全的。利用这种跟踪系统，可能可以提供支付识别记账系统。这还允许服务器授权用户从任意网络的接入，只要用户具有相同的 MSISDN。

HTML 尺寸调整和转换服务器 2128 是在服务器组中的多个服务器之一。由于其从因特网上取回网络内容并调整内容的尺寸，因而它是数据处理服务器。该服务器基于预先编制的脚本进行工作，该脚本是关于调整 HTML 文件尺寸的多个指令。

一旦 HTML 文件的 URL 和用户机器的分辨率被发送到服务器，该服务器则检查，以判断所需文件是否已经在其缓存中。在缓存中没有的文件将从因特网上获得。一旦从因特网上取回文件，则服务器除了在日志中存储其操作外，还将把 WIDTH（宽度）和 HEIGHT（高度）标签添加到在 HTML 文件中不具有这些标签的对象中去。然后其修改 WIDTH 和 HEIGHT 标签中的值，以便在用户机器的显示设备上可以以所要求的分辨率察看这些 HTML 文件。具体做法是用原始的 WIDTH 除以 800，再乘以所要求分辨率的宽度。在大多数情况下，除非由用户请求，否则不修改 HEIGHT 标签。如果用户请求修改 HEIGHT，则用 HEIGHT 的值除以 600，再乘以所要求分辨率的高度。HTML 文件中的文本可以或者通过修改 FONT（字体）标签中的 SIZE（尺寸）值，即将 SIZE 值除以 4，或者

通过添加级联式表（cascading-style sheet）来调整尺寸，所述级联式表不考虑所有的 FONT 标签，并使用字体尺寸 1。由于 IMG 标签的 WIDTH 和 HEIGHT 值被改变，所以向用户发送原始的图形文件不会像发送较大文件一样过分地利用带宽。因此，如果在 HTML 文件中附带了较小尺寸的原始图形文件，则给图形文件的尺寸将被调整，以便使该文件更小。图形文件尺寸的调整依赖于其格式。该格式可以是 JPEG 压缩或 GIF 压缩。图形文件的尺寸被调整到可以显示的分辨率。不必要的大文件不被发送。当服务器调整由平台 0100 所支持的表、嵌入对象、表单、输入和其他标签的尺寸时，要对 0HTML 代码做进一步修改。这不是代码转换，因为没有图像被丢弃，并且 HTML 文件的布局将与原始设计完全相同。任意调整尺寸的工作都将被缓存，以用于以后的再次使用。

在服务器上调整 HTML 的尺寸并进行转换的优点在于，可以采用预处理器来确保用户将接收到适合其机器屏幕的 HTML 内容。另一个优点在于它允许图像在发送给用户前被压缩，将图像调整到更小的分辨率，以与用户机器的显示设备的分辨率相匹配。

媒体转换和流服务器 2129 既是一个转换服务器，也是一个流服务器，但是实际的处理可能是在分开的机器上进行的，以给出更好的性能。转换本质上是数据处理密集型的，而流是带宽和会话监控密集型的。

有两种将媒体格式转换为通用媒体格式的方法。通用媒体格式是一种工业标准格式，为了消除对用于播放多种格式媒体的插件的需要，服务器和平台 0100 已经达成一致，即使用该通用媒体格式作为双方彼此共享的唯一格式。用于选择通用媒体格式的标准是这样的，即该格式必需是一种流格式，并且该格式应该具有不变的压缩率，以实现实时流。

第一种方法是使用现有的代码转换技术，以实现从一种媒体格式到另一种的转换。由于有些格式不使用非对称压缩和编码，因此代码转换技术并不是对所有格式转换都有用。因此，如果第一种方法不可行，则不得不采用第二种方法。第二种方法要花费更多的时间和处理功率。该方法包含将原始的媒体格式解码和解压缩成原始数据（raw data），例如位图帧和 PCM 波形。此外，由于每种媒体格式具有其自己的解码和编码算法，因此

转换的目的并不是对媒体文件进行更好的压缩和编码。更确切地说，转换是为了给出不变的压缩率，以及为了消除在平台 0100 上为了能够察看多种格式的媒体而具有的多个插件。

还执行尺寸调整，以便使服务器向平台 0100 发送较小的文件。所执行的任何转换和尺寸调整都会被缓存，以用于以后的再次使用。在转换和尺寸调整之后，服务器将文件以流的形式发送到平台 0100。

应用容宿和执行服务器是服务器的容宿和执行环境。由于平台 0100 没有提供本地的容宿和执行环境，因此利用服务器的应用容宿和执行服务器来启动和执行应用。这部分内容将在后面的图 16 和图 17 中做进一步的描述。

电子邮件服务器是一个同服务器引擎一起工作的标准的电子邮件服务器。它还允许为扩展编写脚本及对扩展编程，以连接到其他的电子邮件服务器。这允许服务器与其他的电子邮件服务器兼容。

电子邮件服务器的一般操作与标准服务器没有不同。但是，当要发送和接收电子邮件时，电子邮件服务器将检查用户概况，以判断该用户是否已经预订了第三方电子邮件服务器。如果是，则电子邮件服务器将首先使用为第三方电子邮件服务器而编写的扩展脚本，使用用户的用户 ID 和密码登录，并且发送或收取电子邮件。如果接收到一封电子邮件，则将该邮件存储在用户的电子邮件账户中。然后退出用户的第三方电子邮件服务器账户。

电子邮件服务器不仅提供了电子邮件访问，还提供了兼容性扩展，使得用户可以登录到不同的服务器。

存储数据库是一个容宿服务器，该服务器利用数据库来分配定额，并且执行到用户空间的访问控制。它的主要任务是向平台 0100 的用户提供磁盘访问。

优选地，分配定额和执行访问控制的数据库具有五个字段：用户 ID、定额限制、密码、私人访问目录和公共访问目录。私人和公共访问目录每个都具有不同的访问权限，所述访问权限以不同的方式提供访问。私人访问目录是一个“只供察看”的目录，因此用户可以打开受版权保护的文

件，但不能将它们转发到第三方或其他目录。当用户已经从内容提供者那里预订或购买了内容时，内容提供者可以将受版权保护的媒体文件写入私人访问目录中。公共访问目录是一般的磁盘空间，在其中用户可以创建、复制、删除和打开文件。它不具有访问限制，并且用户可以打开要使用目录中所存储的文件的应用。

硬件安装 2130 和 OS 更新 2131 服务器是一个数据库服务器，其中存储了所有的硬件驱动程序和 OS 更新。一旦接收到来自平台 0100 的要求，服务器就将搜索合适的硬件驱动程序或 OS 更新包，并且将其发送给用户。所发送的这种包对平台 0100 是唯一的，因为它携带了脚本，该脚本指导平台 0100 如何安装 OS 更新的驱动程序，以便使在该过程中不需要用户的介入。这部分内容还将在后面的图 18 和 19 中进行描述。

用户认证服务器在图 13 中进行了描述。

因特网网关是一个标准的网关服务器或代理服务器。它可以具有不同的队列系统和安全防火墙的实现方法。管理服务器是使用日志、配置面板和监控机制的中心存储库。网络的管理员可以：警告新的 OS 更新；检查用户是否需要在数据库中没有找到的任何硬件驱动程序；监控存储数据库；配置应用服务器；创建用户账户；产生记账和使用报告；设置安全策略；服务器更新；以及修改所有服务器的配置设定。

参考图 1，示出了平台 0100。这是驻留于用户机器中的操作系统。它与服务器连接并交互，以用于登录、请求网络内容富媒体、应用和系统更新。“电话特征”指的是诸如蜂窝电话呼叫、SMS、地址本、联系人、日历等等一般的电话功能。

引擎是命令解释器，并且调用引擎 0130 来进行所有必需的本地处理和网络控制。引擎 0130 处理所有的操作，并且只有在存在触发时才会作出反应。然后它分支到引擎系统所提供的不同服务。由引擎所提供的服务包括处理因特网连接、重放网络内容富媒体、启动远端应用、系统设置和外设支持。该引擎融入了传统操作系统的分层结构，提供了操作完整性、执行速度和有效的存储器控制。

平台触发 0131 是在用户端调用的触发。该触发包括：通过触摸屏、

键盘、命令或系统输入而进行的用户输入。

平台 0100 引导并启动。在初始化电话特征之后，平台 0100 将被启动。然后平台 0100 呈递（render）和打开用户接口。平台 0100 将开始监听平台触发。当存在平台触发时，将执行平台触发检测过程，以检测被调用的触发是什么类型。如果是“登录到服务器”的触发 0132，则将启动用户认证过程 0200。如果是“请求 HTML”0133，则将启动察看 HTML 过程 0300。如果该触发是发送内容富媒体流的请求 0134，则将启动“启动多媒体插件”过程 0400。如果该触发是要求执行应用 0135，则将启动“启动远端应用”过程 0500。如果是系统更新 0136，则将启动系统更新过程 0600。如果该触发是由本地硬件输入 0137 所调用，则将处理硬件触发 0138。平台 0100 将继续循环和检测平台触发，直到该平台被关闭。

如图 2 所示，一旦平台 0100 启动，并且用户登录，则调用用户认证 0200。该过程包括从 SIM 卡模块中获得 MSISDN、用于 PKI 的用户 ID 和密码，以及将这些信息提交给服务器 0201。用户 ID 和密码将从用户机器的快闪 ROM 中获得。这两者都会被发送到服务器，以用于认证 0204。然后将启动服务器用户认证过程 1300。一旦建立了认证过程 1300，并且用户是有效的 0205，则打开用户与服务器之间的 SSH 连接 0206。由于最初的认证使用了 SIM 卡，所以采取了增强的安全措施。认证和安全隧道比传统的 SSL 连接更加安全。此外，通过使用要使用用户 ID 和密码的第二层登录，在由电信服务提供商所提供的服务上提供了加密和压缩。如果不是有效的用户 0207，则用户将不被接纳，并且平台 0100 将在用户机器的显示设备上显示“请向运营商要求服务”。

图 3 中示出了察看 HTML 文件的过程。虽然打开 HTML 文件的过程与传统的 HTML 浏览器类似，但是该 HTML 浏览器模块无需外部播放器或外部插件就可以读取全部的内容富媒体。给定 URL 0301，浏览器从服务器 0302 请求 HTML 内容。一旦接收到 HTML 内容，则服务器调整其尺寸并将其缓存 1400。然后浏览器检查是否基于已调整尺寸的 HTML 文件附带了任意的内容富媒体 0303。如果存在内容富媒体，则浏览器启动其多媒体插件 0400。这是一个内部插件，该插件以一种通用格式从服务器取回

多媒体内容。这部分内容将在图 4 中示出。

多媒体插件 0400 是通用插件，该插件取回通用格式的视频和音频内容。这消除了对于多种媒体格式的对多个插件的需求。服务器将所有传统的媒体格式转换为通用媒体格式。该插件本质上是一种流媒体插件。它使用公知的、传统的流技术和工业压缩技术。其优点在于，由于该插件只以一种格式取回多媒体内容，因此它消除了对若干插件的需求。

在媒体内容的 URL 地址 0401 被传递到服务器流媒体处理 0402 之后，将从服务器开始媒体流数据的转换 1500。视频和音频重放的察看端口将被更新 0403。在文件已经完成了以流的形式发送 0404 之前，将会继续从服务器获得流数据。同时，还要检查用户是否触发了会改变重放流顺序的任意重放设置 0405。如果发生改变，则将会因此更新视频和音频重放的察看端口。

图 5 示出了启动诸如桌面应用的远端应用的过程，所述桌面应用例如是“Windows”办公生产率（office productivity）软件或标准的桌面应用。这些应用都是为诸如“Windows”、“Linux”、Java 和“Palm”OS 的 OS 而为桌上型 PC 环境本地编写的传统桌面应用。

给定应用的 URL 地址 0501，平台 0100 将其屏幕分辨率连同启动远端应用的请求一起发送到服务器 0502。作为回复，一旦该应用被成功地启动并执行 1600/1700，则平台 0100 将接收到 GUI 指令，从而在屏幕上显示 GUI 组件 0503。这与屏幕转储（screen dumping）或远端桌面技术不同，在屏幕转储或远端桌面技术中，用户不知道所接收到的是什么。一旦接收到 GUI 指令，平台 0100 就将创建 GUI 组件，并允许在本地无需服务器的介入而执行这些组件之间的交互 0504。当要求诸如排序或保存文件一类的过程时，平台 0100 再次求助于服务器。在所有其他的时间，平台 0100 都是在本地操作的。平台 0100 还检查是否将执行任何平台触发。如果是，则平台将允许该触发被调用，并且过程继续进行。

如果输入过程要求显示更新 0505，则接口操作将被发送到服务器 0506。服务器将返回反馈信息 0570。如果反馈信息是平台触发 0508，则将调用平台触发。然后将取回已输出的应用显示，并且将重新启动用户接

口输入检测。

在图 6 的系统更新中，“设备标题和制造商名称”指的是在初始化过程 0601 期间，从用户机器发送到服务器的外设的说明和制造商信息。这类信息一般例如是设备 A 模型名称、打印机 USB、制造商 B、版本 1.2。

OS 更新 0602 指的是被发布给用户的更新补丁或修正程序。这类更新一般是由平台 0100 自动接受的，并且执行自动更新过程。当执行这种操作时，发生 OS 更新触发 0603。

一旦检测到触发，系统更新模块就将进行检测，以判断该触发是 OS 更新，还是外部设备。该过程是通过检测触发源来进行的。OS 更新只能来自服务器，而设备检测只能发生在平台 0100 侧。由于所安装的外设可以被使用多次，因此将驱动程序文件存储在用户机器的 ROM 中。因此，在下一次检测到该设备时，通过从用户机器的 ROM 中取回必需的驱动程序，就可以立即激活该设备。在最初的安装期间，平台 0100 将服务器必需的所有详细信息发送到该服务器，所述信息包括设备标题和制造商名称。如果检测到设备，则将对在 ROM 中是否存在该设备的驱动程序进行检查 0604。如果是，则停止并重新启动适当的服务，以激活该设备 0605。如果设备的驱动程序不在 ROM 中，则将启动设备驱动程序请求过程 1800。如果没有找到设备的驱动程序，则返回一个报告，所返回的报告警告“没找到驱动程序”。如果找到了驱动程序 0606，则该驱动程序被下载 0607，解包 0608，并且将执行包内的自动安装脚本 0609。设备的驱动程序文件将被复制到系统文件位置，并且将更新系统设置。由于对 ROM 能够支持的设备驱动程序的数量有所限制，所以最旧的设备驱动程序，或最少使用的设备驱动程序将被宣布作废，并且从 ROM 中删除。在完成更新之后，将停止并重新启动适当的服务，以激活该设备。

OS 更新是以类似的方式执行的。给定一个被调用的 OS 更新触发，平台与服务器协商，以核实这是否是所要求的更新。如果这是 OS 更新，则将平台的 OS 设置发送到服务器，以用于同步。在同步之后，将启动 OS 更新请求。如果不存在更新，用户将收到“无更新”0611。如果存在更新，则用户将接收到 OS 更新或升级包 0610。接收到的包被解包，并且将

执行包内的自动安装脚本 0612。该脚本将指导平台 0100 将任何新文件复制到系统中，并且重新编译必需的文件 0613。

在图 7 中，输入触发指的是由用户 0701、平台 0702 和服务器 0703 所产生的输入。

用户指的是使用机器的人。典型的输入 0800 是来自使用键盘、鼠标、指示器（pointer）或提交 URL 的指令。系统指的是用户机器。来自设备自身的唯一可能输入 0900 是当机器检测到外部的外设并将其报告给平台 0100 时出现的。

服务器指的是位于远端区域的服务器，该服务器经由网络被连接到电信服务提供者或通信基础设施，所述电信服务提供者或通信基础设施具有到用户机器的连接。这里存在两种来自服务器 1000 的可能触发/输入。第一种是 OS 更新警告，由此服务器警告平台 0100 出现新的 OS 更新，指导平台 0100 下载该修正程序/更新并安装之。第二种来自服务器的可能触发是来自服务器的命令。这将发生在用户当前使用的应用呼叫命令时，并且该命令在服务器处被处理。但是，如果特定的命令不能在服务器处被处理（例如，打印功能），那么服务器将通过指示平台 0100 执行重新定向功能来重新定向该命令功能。

当输入是来自用户时（图 8），该输入可以被分类为两种，即服务请求或登录。这两种输入都必须由平台 0100 来处理。在平台完成其处理之后，要对是否需要服务器处理做出决定。如果不需要服务器做出进一步的处理 0801，则处理结束 0802。否则由服务器接管处理过程 1100/1200，直到该请求被处理。

对于系统输入（图 9），唯一的可能输入是检测到新的硬件设备。平台 0100 首先执行所有必要的预处理工作，然后确定需要从服务器获取的任意驱动程序 0901。如果需要驱动程序，那么服务器将检查其是否具有所需的驱动程序 0902。如果没有找到合适的驱动程序，那么服务器将产生一个报告，声明用户需要什么驱动程序 0903。如果驱动程序被找到，则一旦在系统上安装了该驱动程序，它就将被平台激活 0904。一旦安装了驱动程序，则用户将能够操作该新安装的硬件设备。

在图 10 中示出了对服务器输入 1100/1200 的响应。在服务器已经至少执行了少量的处理之后，首先形成服务器输入 1100/1200。该步骤将警告平台存在 OS 升级/更新 1001 或来自服务器侧的呼叫命令 1002，从而指导平台 0100 进行诸如打印或扫描功能的本地处理。这两种输入都被馈送到平台 0100，以用于由平台 0100 进行处理。

在图 11 和 12 中示出了服务器的操作。服务器包括驻留在其中的软件。服务器通过网络被连接到电信服务提供者的基础设施。基本的组件包括数据库、电子邮件、代理、缓存、应用和媒体服务器。服务器引擎提供在服务器处的通信和操作指令。服务器中的组件使用命令来彼此会话，并且只有当发布这样的命令时，才会执行操作。服务器中的多个组件中的每一个都具有执行引擎，而主服务器拥有服务器引擎，该服务器引擎是命令解释器。

当服务器启动 1101 时，其初始化数据库、电子邮件、代理、缓存、应用、OS 更新和媒体服务器 1102，以确保它们被校准，并且一起工作。然后，服务器引擎被加载 1103，并且开始监听用户登录 1104。当用户登录 1105 时，将启动服务器认证过程 1300。将登录的细节存储到日志文件中 1106，并且建立与用户之间的安全外壳（SSH）连接 1107。然后进入检查是否存在用户请求的过程 1108。如果没有用户请求，则将检查用户是否被断开 1109。如果用户被断开，则将存储退出的细节 1110。如果用户仍旧在连接中，则将继续检查。当接收到用户请求时，将调用服务器请求检测过程 1111。用户的行为被存储在日志中 1112。如果用户的请求是关于 HTML 内容 1201，则将启动调整尺寸和 HTML 缓存过程 1400。如果该请求是富媒体内容 1202，则将启动服务器流媒体过程 1500。如果用户的请求是关于应用 1203，则将启动启动应用过程 1700。如果用户的请求是设备驱动程序 1204，则将启动设备驱动程序请求过程 1800。如果用户的请求是 OS 更新或升级 1205，则将启动 OS 升级请求过程 1900。否则，该请求被忽略和/或被返回到平台，并且服务器将继续检查用户的请求。

在图 13 中示出了用户认证期间的服务器操作。这里，认证网关指的是一台单独的服务器，该服务器处理电信服务提供者处的用户认证。当服

务器接收到 MSISDN、用户 ID 和用户密码时 1301，对用户数据库进行搜索，以找到匹配项 1302。如果不存在匹配，则拒绝该连接 1303。虽然用户通过电信服务提供者的网络进行登录，这意味着它们已经通过了一个认证过程，但是服务器将再次利用电信服务认证网关进行检查 1304，以判断该请求是否来自黑名单中的用户，是否已经超出其提供者的定额，或者是否是一个没有预定该服务的无效用户。如果电信认证网关的回复指出该用户无效 1305、超出定额 1306 或被记于黑名单中 1307，则拒绝该连接。否则，将证实用户为有效用户 1308。

如图 14 所示，一旦 HTML 文件的 URL 和用户机器的显示器的分辨率被发送到服务器 1401，则该服务器将检查所需文件是否已经在缓存中 1402。不在缓存中的文件将从因特网上获得 1403。一旦从因特网上取回文件，则服务器将该操作存储到日志中 1404，并且添加或修改 HTML 文件中的某些调整文本尺寸的值 1405。然后将图形文件的尺寸调整到在用户机器上可显示的分辨率 1406，因此不发送不必要的大文件。对 HTML 代码的进一步修改 1407 包括 WIDTH 和 HEIGHT 的值，并且服务器调整平台所支持的表、嵌入材料、表单、输入和其他标签的尺寸。这不是代码转换。没有图像被丢弃，并且 HTML 文件的布局将与原始文件相同 1408。任何调整尺寸的工作都会被缓存 1409，以用于以后的再次使用。

对于服务器流媒体（图 15），给定 URL 和请求的分辨率 1501，服务器将获取缓存中没有的必需的媒体文件 1502，并将文件的格式转换为通用文件格式。该过程包括利用文件的扩展名（文件名的最后三个字母）来检测媒体 1503，以及随后将文件格式转换为通用文件格式 1504。这样就确保了向平台输出正确的格式。转换技术包括以下两种，即或者是代码转换，或者将原始文件格式解码到输出数据中，再将该数据重新编码为通用格式。将使用哪种技术依赖于媒体的原始格式。尺寸调整也会被执行 1505，以便使服务器将较小的文件传送至平台。所执行的任何转换和尺寸调整都会被自动地缓存，以用于以后的再次使用 1506。在转换和尺寸调整之后，服务器将该媒体文件以流的形式向平台 0100 发送 1507。

在图 16、17 和 22 中示出了在平台 0100 上启动应用的过程。这里，被

保护的环境 2202 指的是向应用发出的访问控制。由于应用可能会覆写其他用户的数据，因此不允许应用写入服务器中磁盘空间的所有部分。因此，访问控制是用于将应用的访问限制到系统的受限区域中。访问控制还可以从用户的存储区域（数据库）中镜像出文件列表，并且将该列表馈送到应用 2201。当应用请求文件时，从数据库中取回该文件，并将该文件复制到被保护的环境中，以用于处理。当会话被关闭，或应用停止时，将文件写回数据库。这再次确保了数据将不会恶化。但是，在应用失败并且发生重启动的情况下，所包含的会话在其超时之前仍旧保存着“临时”文件。这意味着没有数据丢失。

一旦请求被馈送到服务器 1601，该服务器就检查应用是否在服务器域内 1602。这是一个重要的检查，因为服务器只能执行可信的应用 1603/1604。服务器还要检查用户是否已经预订了该应用 1605。在可以启动执行之前，服务器将创建一个被保护的环境 1606，这是服务器一般不执行的操作。在被保护的环境内，服务器将启动 API 服务器，该 API 服务器将该应用所依赖的本地 OS 的 API 映射到服务器 API 1607。该过程是实时进行的。服务器具有一组 API，这些 API 遵从于其他 OS 的所有本地 API 1608。在这种情况下的映射指的是重新定向。现在不是执行本地的 API，而是执行服务器的一组 API 1609。这给予服务器对应用更大的控制能力 1610。由服务器所覆盖的操作包括向平台 0100 输出显示 1611。结果，GUI 指令不是在服务器侧执行，而是被输出到用户侧。过程指令是在服务器侧执行的。由于 GUI 指令是通过 API 被呼叫的，所以服务器检测这些 API，并将其输出到平台 0100。其他操作包括重新定向硬件请求和 MXI 命令，这些不能由服务器来完成。由于硬件请求是通过 API 被呼叫的 1701，因此服务器检测这些 API 1702，并且将它们重新定向回平台 1703，以用于处理。不能在服务器端被处理，但是是在服务器上被呼叫的 MXI 命令被定向到平台，并在平台上被执行 1704。该应用将持续运行，并处理用户请求，直到会话结束，或用户停止该应用。

在映射能够发生之前，必须将应用加载到被保护的执行环境 2202 中，在该环境中，通过截取功能呼叫来捕捉其他 OS 的本地 API 呼叫。被

保护的执行环境首先识别其支持的所有其他 OS 的本地 API 呼叫。当检测到对本地 OS 库的 API 功能呼叫时，被保护的执行环境将呼叫其 API 集合，该集合等同于正在被应用呼叫的 API 集合。由于所有的 API 都被映射到服务器 API，因此现在可以在服务器上本地地运行应用。

由于包括 GUI 和 I/O API 的所有 API 都被映射到服务器 API，因此服务器具有对应用更大的控制能力。这允许服务器来决定什么操作可以被输出到平台，什么操作可以在本地处理。在执行应用时，被保护的环境将输出服务器将不处理的 API 呼叫。这种呼叫包括 GUI 和 I/O 呼叫。GUI API 呼叫被直接输出到平台，并且平台将处理必需的 GUI 组件。这里使用的术语“输出”意味着 GUI API 被重新定向，并且在平台端被呼叫。I/O 呼叫由呼叫命令来重新定向，所述呼叫命令通过平台引擎由平台解析，用于执行 I/O 操作。

被保护的环境的优点在于，由于在每个被保护的环境中的访问控制限制，从而该环境减少了病毒扩散的机会。

应用容宿和执行服务器并入了单个应用的多会话启动技术。由于在服务器的存储器中只要求加载应用的一个拷贝，因此减少了存储器的使用和对服务器的影响。该技术是一项在大多数运行 UNIX 的主机系统中使用的标准的多会话技术。

在图 18 中示出了设备驱动程序请求。“包”（package）指的是一个包含多个压缩文件的文件。包通常包括必需的核心内容（在此情况下是驱动程序），并且还包括由平台 0100 所执行的安装指令。

一旦设备标题和制造商名称被馈送到服务器 1801，服务器则执行搜索 1802。如果存在匹配 1803，则出于参考的目的，服务器将细节存储到日志中，并且将适当的包发送给用户 1805。由于该包是在服务器端准备好的，并且以这样的方式被打包，使得一旦平台 0100 接收到该包，就将对该包进行解包，并且运行包内所包括的安装指令 1806。如果不存在匹配，则在日志文件中创建日志记录，该日志记录给出了所需驱动程序的细节 1804。然后，将通过电子邮件向管理员发送所需设备驱动程序的细节的拷贝，并且向用户发送报告，以告知用户没有找到驱动程序 1808。

在图 19 中示出了对操作系统的更新。这里的包具有与图 18 中的包相似的意义，只是这里的包包含 OS 更新文件，而不是驱动程序。典型的 OS 更新包包括平台所需的任意附加文件，以及将由平台所执行的安装指令。

服务器将检查在用户的平台上已经安装了哪些包 1901。然后服务器编译所需包的列表 1902。如果该列表不为空，则服务器将继续检查是否存在任何的相关性或其他包 1903。通常第一轮就可以覆盖所有相关性，但是出于核实的目的，还要进行第二轮检查，以便在平台安装该包时不会出现问题 1904。在每次相关性检查之后，如果在已编译的列表中添加任意的包，就需要执行一轮额外的相关性检查。这是为了判断被添加的包是否具有任何相关性。一旦所有的包都得到确认，则服务器将发送这些包 1905，这些包是顺序发送的 1906，以确保最先请求的包是最先由平台 0100 接收到并安装的包。

本发明还被扩展到在处理器上可执行的软件布置，该软件布置包括对处理器进行配置，以执行上述功能中的一个或多个功能的计算机程序。本发明还被扩展到计算机系统，该计算机系统包含一个或多个装置，用于执行相应的上述一个或多个功能。

虽然在前面的说明中已经描述了本发明的一个优选实施例，但是本技术领域的技术人员将会理解，不脱离本发明，可以对操作、设计或结构的细节做出很多变化和修改。

本发明可以扩展到被单独公开的，或在所有可能的排列与组合中被公开的所有特征。

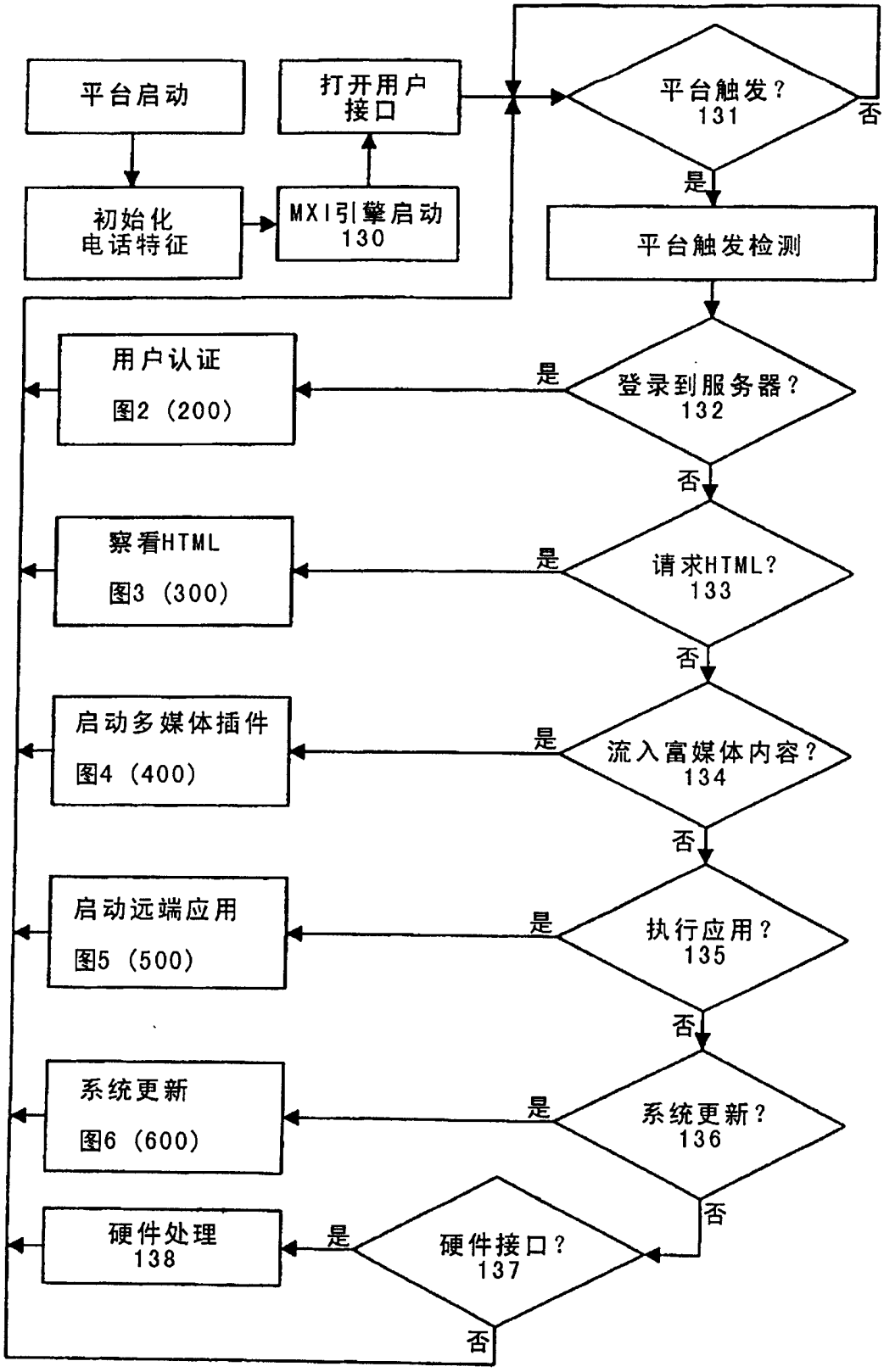
平台启动
打开用户接口
平台触发？131
否
是
初始化电话特征
MXI引擎启动130
平台触发检测
用户认证
图2（200）
是
登录到服务器？132
否
察看HTML
图3（300）
是
请求HTML？133
否
启动多媒体插件
图4（400）
是
流入富媒体内容？134
否
启动远端应用
图5（500）
是
执行应用？135
否
系统更新
图6（600）
是
系统更新？136
否
硬件处理138
是
硬件接口？137
否

图1

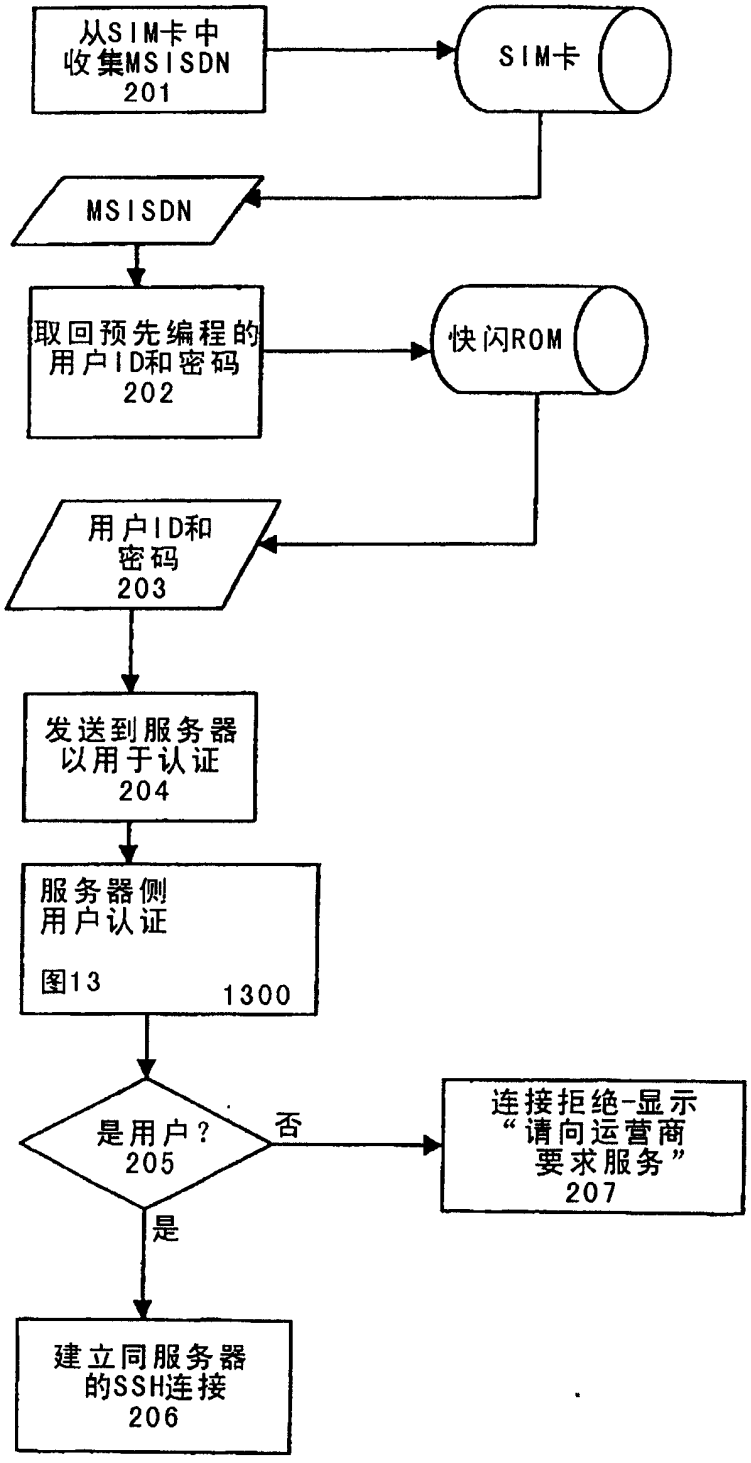
从SIM卡中
收集MSISDN
201
SIM卡
MSISDN
取回预先编程的
用户ID和密码
202
快闪ROM
用户ID和
密码
203
发送到服务器
以用于认证
204
服务器侧
用户认证
图13
1300
是用户？
205
否
连接拒绝-显示
“请向运营商
要求服务”
207
是
建立同服务器
的SSH连接
206

图2

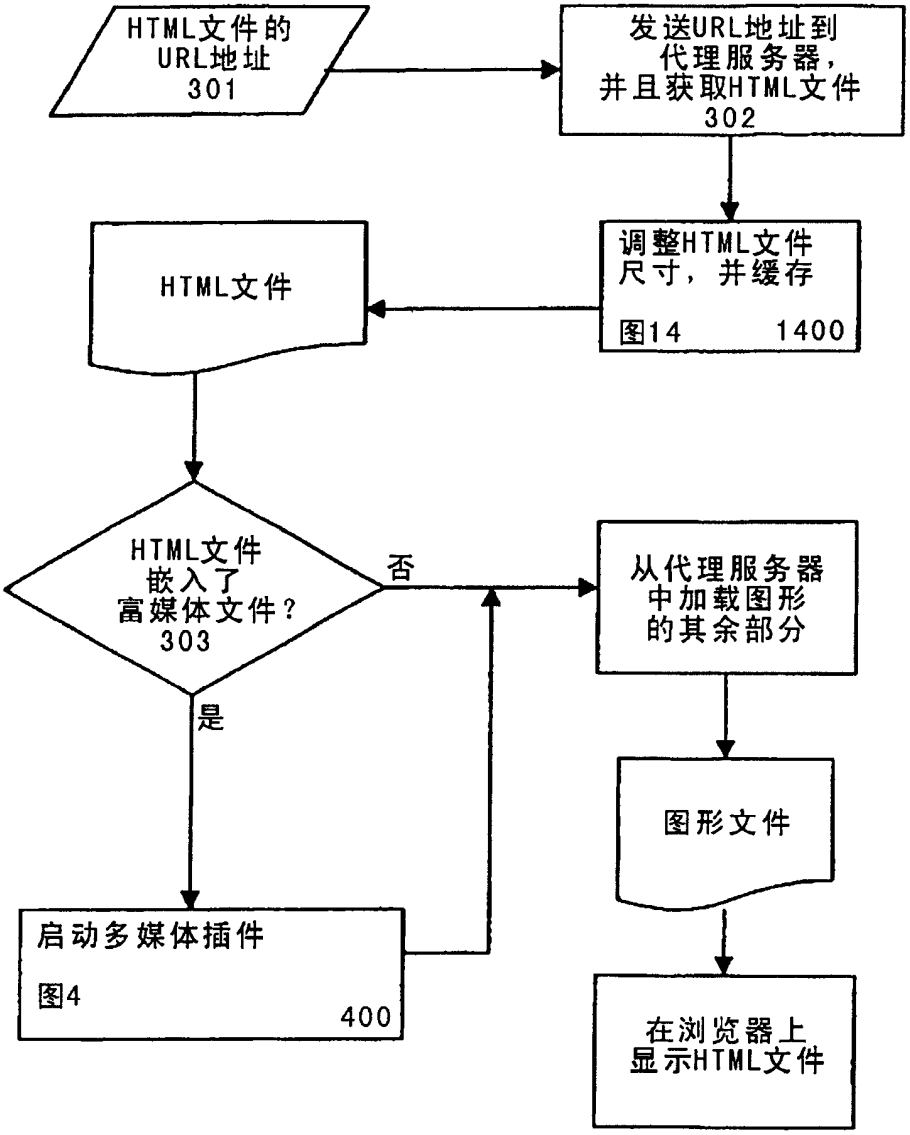
HTML文件的
URL地址
301
发送URL地址到
代理服务器，
并且获取HTML文件
302
调整HTML文件
尺寸，并缓存
图14 1400
HTML文件
HTML文件
嵌入了
富媒体文件？
303
否
是
从代理服务器
中加载图形
的其余部分
图形文件
启动多媒体插件
图4
400
在浏览器上
显示HTML文件

图3

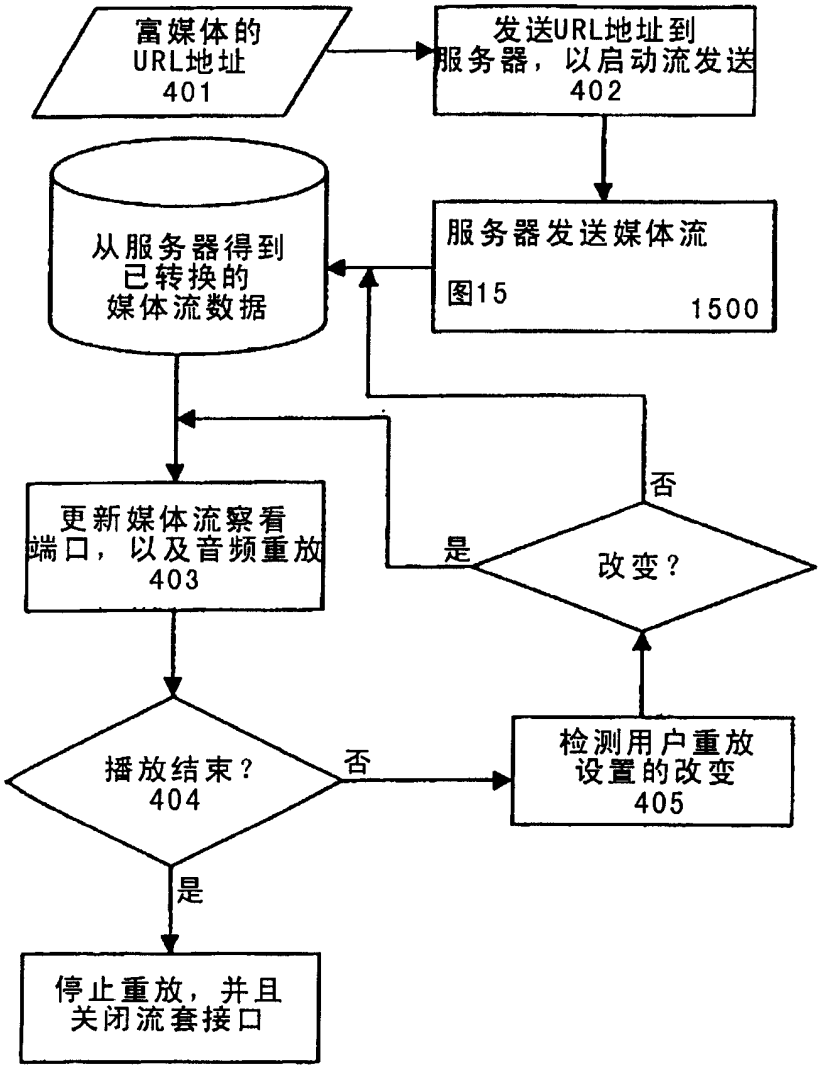
富媒体的
URL地址
401
发送URL地址到
服务器，以启动流发送
402
从服务器得到
已转换的
媒体流数据
服务器发送媒体流
图15
1500
更新媒体流察看
端口，以及音频重放
403
否
是
改变？
播放结束？
404
否
检测用户重放
设置的改变
405
是
停止重放，并且
关闭流套接口

图4

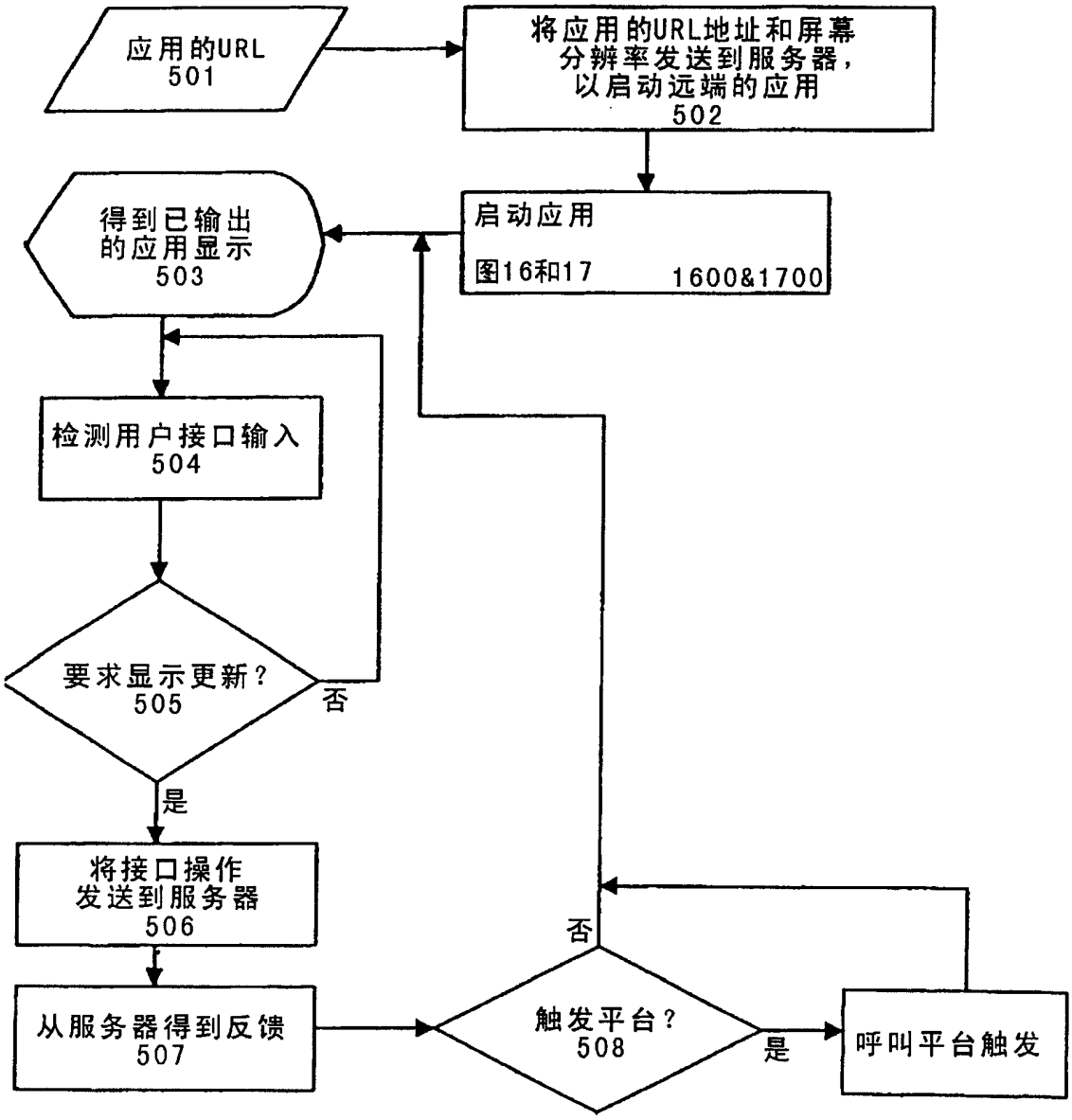
应用的URL
501
将应用的URL地址和屏幕
分辨率发送到服务器，
以启动远端的应用
502
得到已输出
的应用显示
503
启动应用
图16和17 1600&1700
检测用户接口输入
504
要求显示更新？
505
否
是
将接口操作
发送到服务器
506
从服务器得到反馈
507
触发平台？
508
否
是
呼叫平台触发

图5

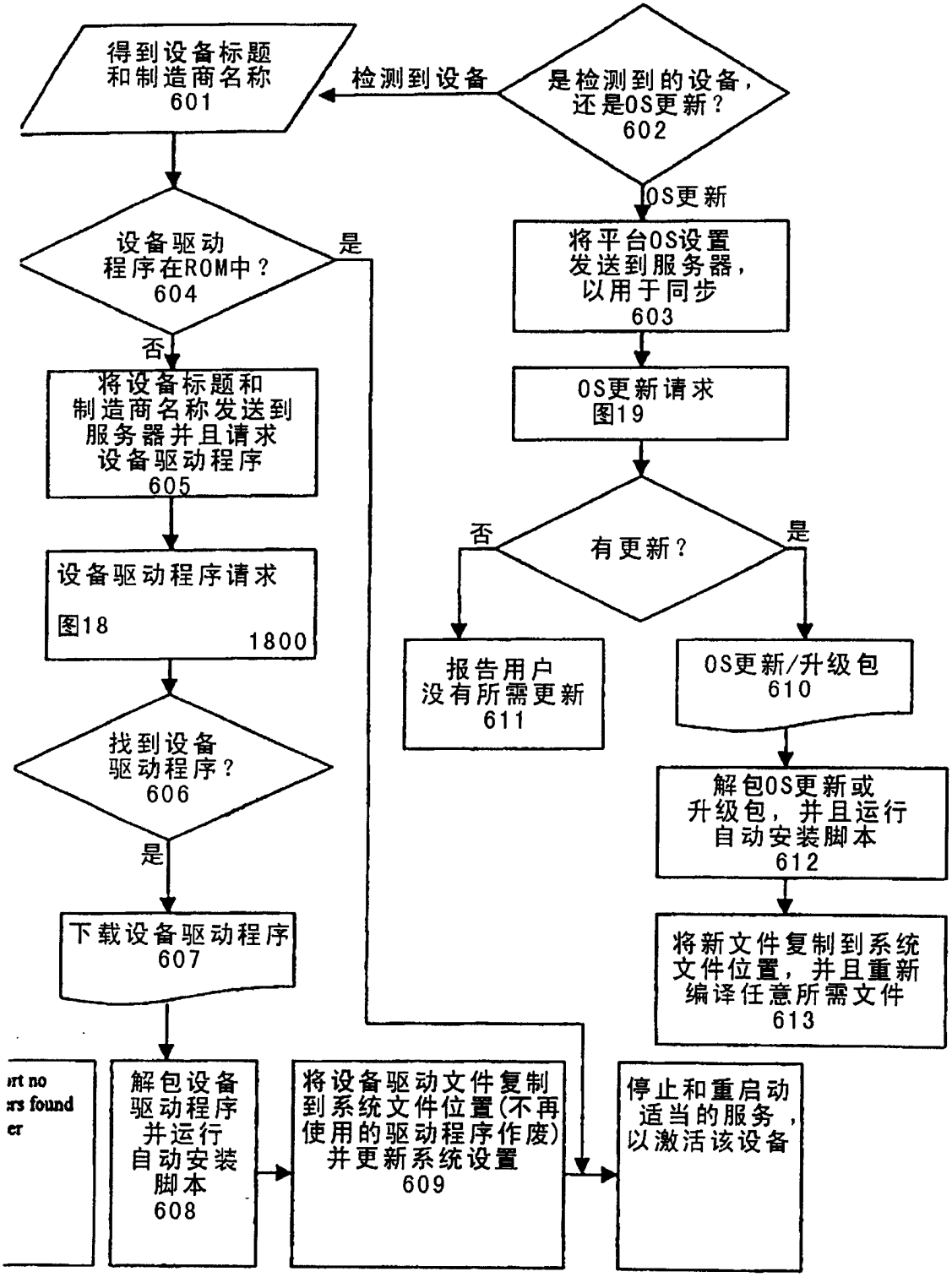
得到设备标题
和制造商名称
601
检测到设备
是检测到的设备，
还是OS更新？
602
OS更新
设备驱动
程序在ROM中？
604
是
否
将平台OS设置
发送到服务器，
以用于同步
603
将设备标题和
制造商名称发送到
服务器并且请求
设备驱动程序
605
OS更新请求
图19
设备驱动程序请求
图18
1800
否
有更新？
是
报告用户
没有所需更新
611
OS更新/升级包
610
找到设备
驱动程序？
606
是
解包OS更新或
升级包，并且运行
自动安装脚本
612
下载设备驱动程序
607
将新文件复制到系统
文件位置，并且重新
编译任意所需文件
613
解包设备
驱动程序
并运行
自动安装
脚本
608
将设备驱动文件复制
到系统文件位置(不再
使用的驱动程序作废)
并更新系统设置
609
停止和重启动
适当的服务，
以激活该设备

图6

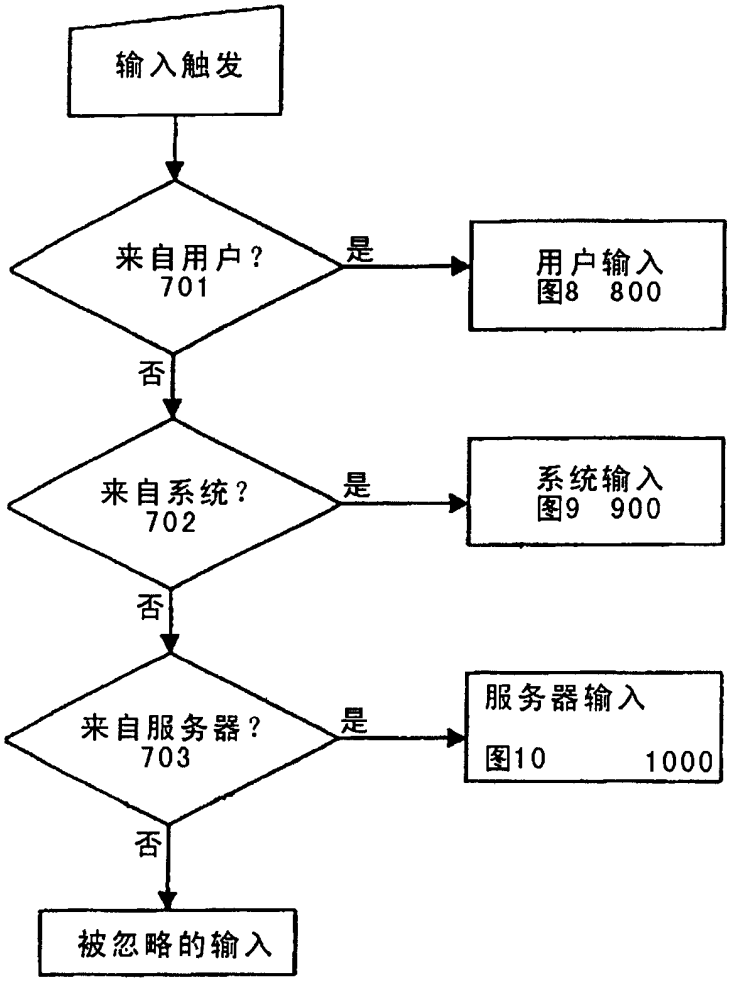
输入触发
来自用户？
701
是
用户输入
图8 800
否
来自系统？
702
是
系统输入
图9 900
否
来自服务器？
703
是
服务器输入
图10 1000
否
被忽略的输入

图7

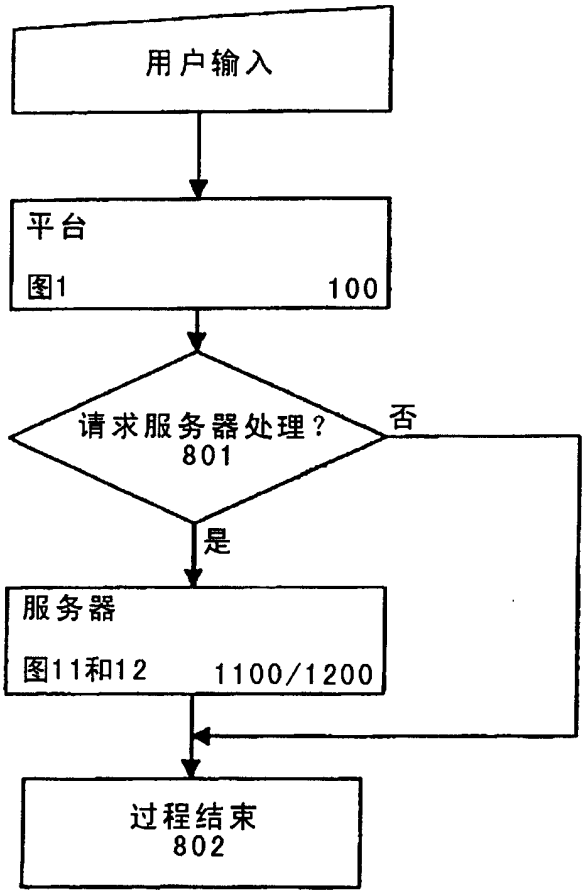

图8

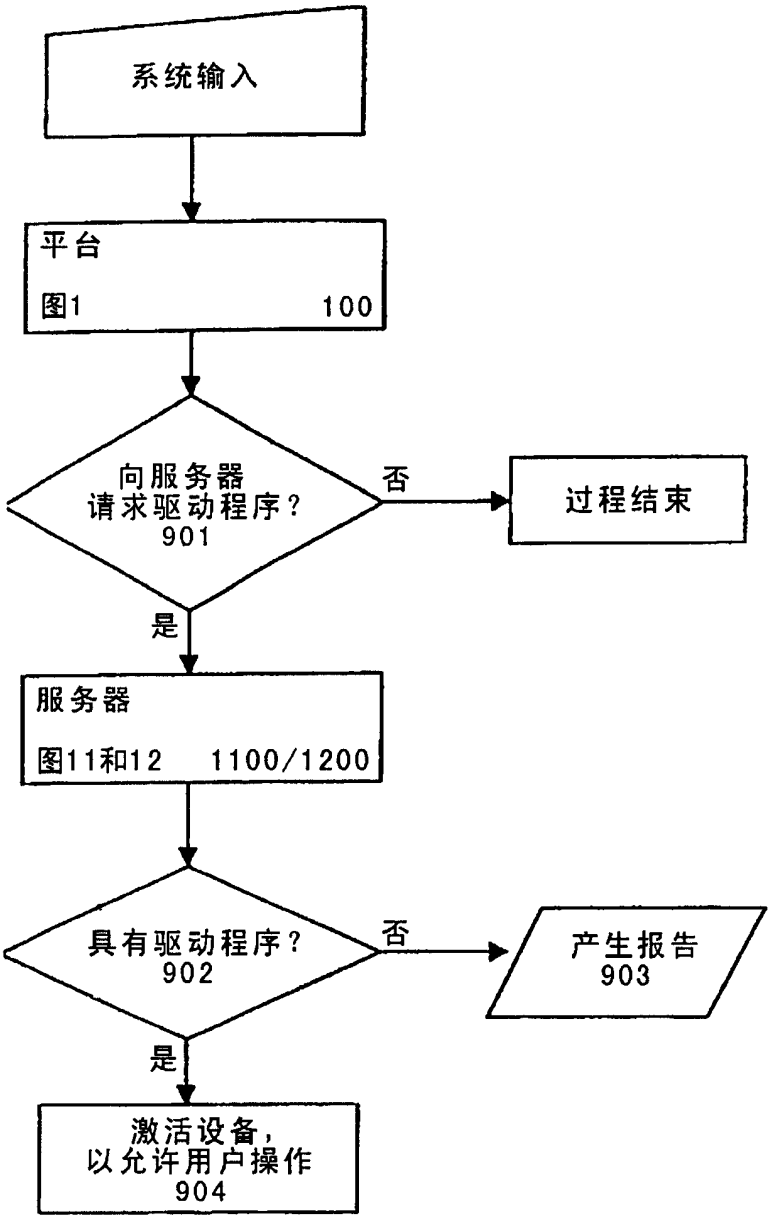
系统输入
平台
图1 100
向服务器
请求驱动程序？
901
否
过程结束
是
服务器
图11和12 1100/1200
具有驱动程序？
902
否
产生报告
903
是
激活设备，
以允许用户操作
904

图9

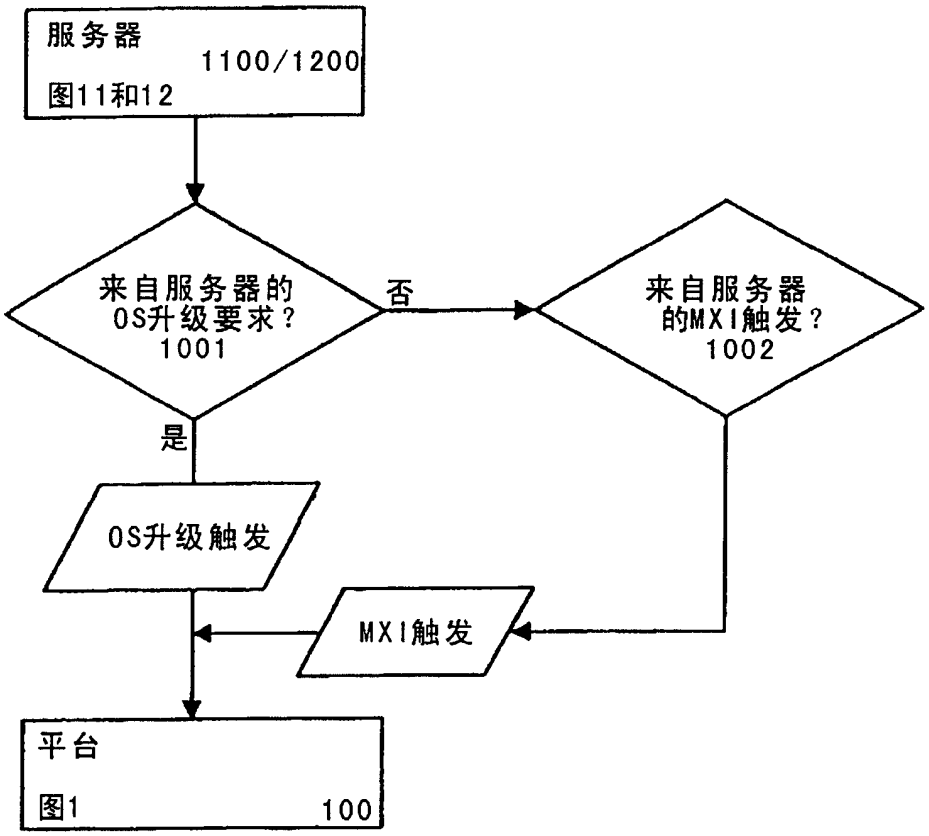
服务器
1100/1200
图11和12
来自服务器的
OS升级要求？
1001
否
来自服务器
的MXI触发？
1002
是
OS升级触发
MXI触发
平台
图1
100

图10

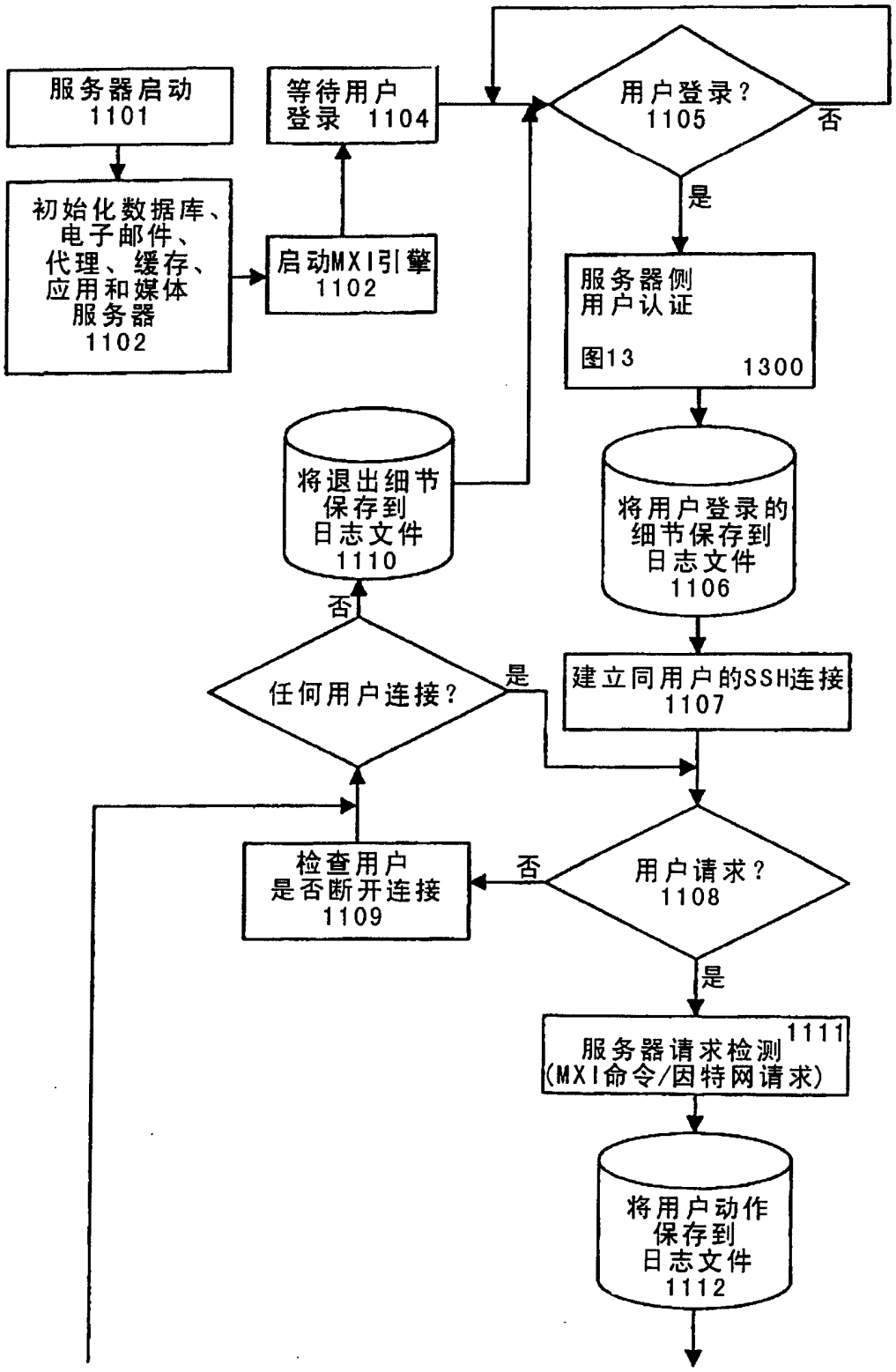
服务器启动
1101
初始化数据库、
电子邮件、
代理、缓存、
应用和媒体
服务器
1102
启动MXI引擎
1102
等待用户
登录 1104
用户登录？
1105
否
是
服务器侧
用户认证
图13
1300
将用户登录的
细节保存到
日志文件
1106
建立同用户的SSH连接
1107
用户请求？
1108
否
是
服务器请求检测
(MXI命令/因特网请求)
1111
将用户动作
保存到
日志文件
1112
检查用户
是否断开连接
1109
任何用户连接？
否
是
将退出细节
保存到
日志文件
1110

图11

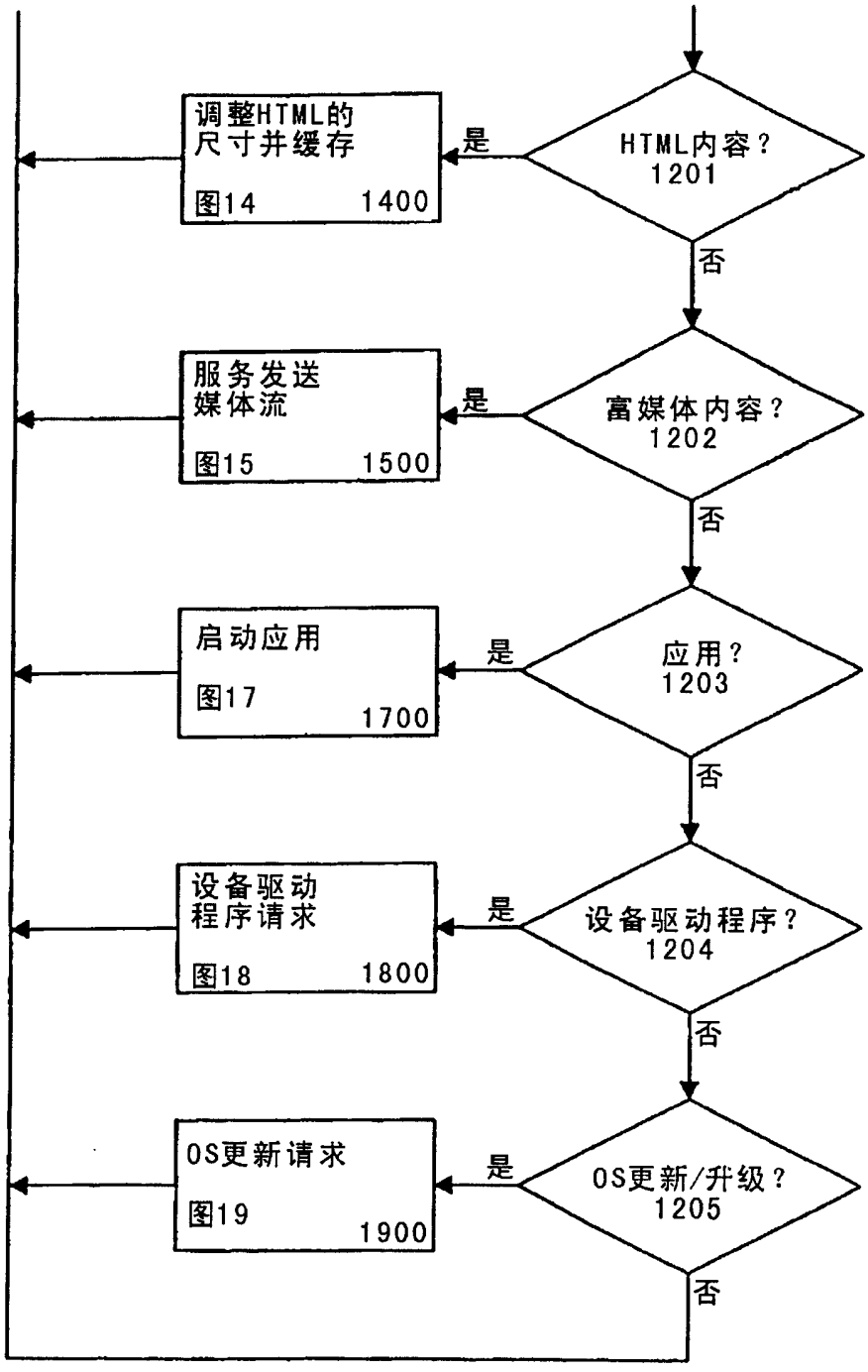
HTML内容？
1201
是
调整HTML的
尺寸并缓存
图14 1400
否
富媒体内容？
1202
是
服务发送
媒体流
图15 1500
否
应用？
1203
是
启动应用
图17 1700
否
设备驱动程序？
1204
是
设备驱动
程序请求
图18 1800
否
OS更新/升级？
1205
是
OS更新请求
图19 1900
否

图12

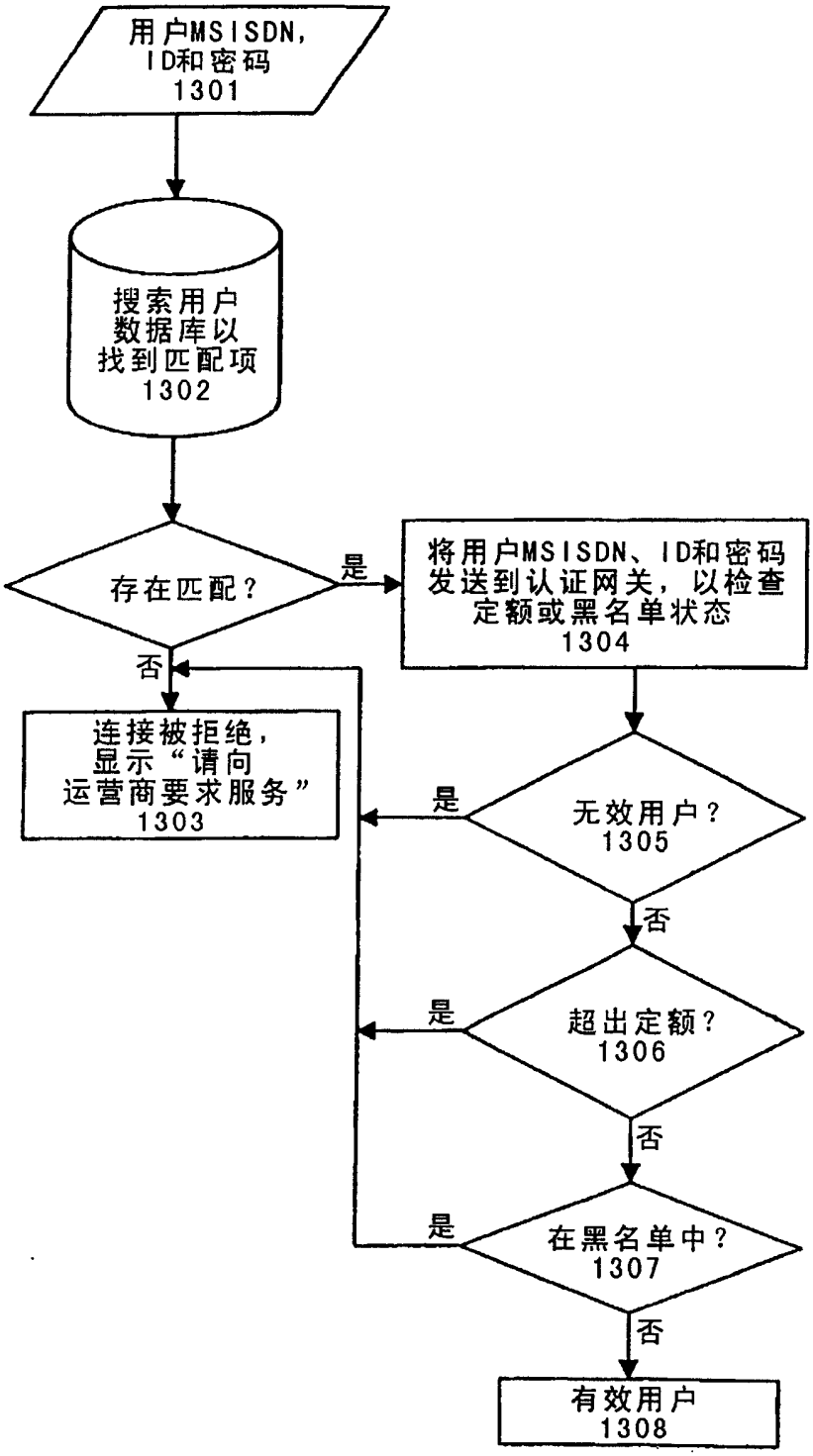
用户MSISDN，
ID和密码
1301
搜索用户
数据库以
找到匹配项
1302
存在匹配？
是
否
将用户MSISDN、ID和密码
发送到认证网关，以检查
定额或黑名单状态
1304
连接被拒绝，
显示“请向
运营商要求服务”
1303
无效用户？
1305
是
否
超出定额？
1306
是
否
在黑名单中？
1307
是
否
有效用户
1308

图13

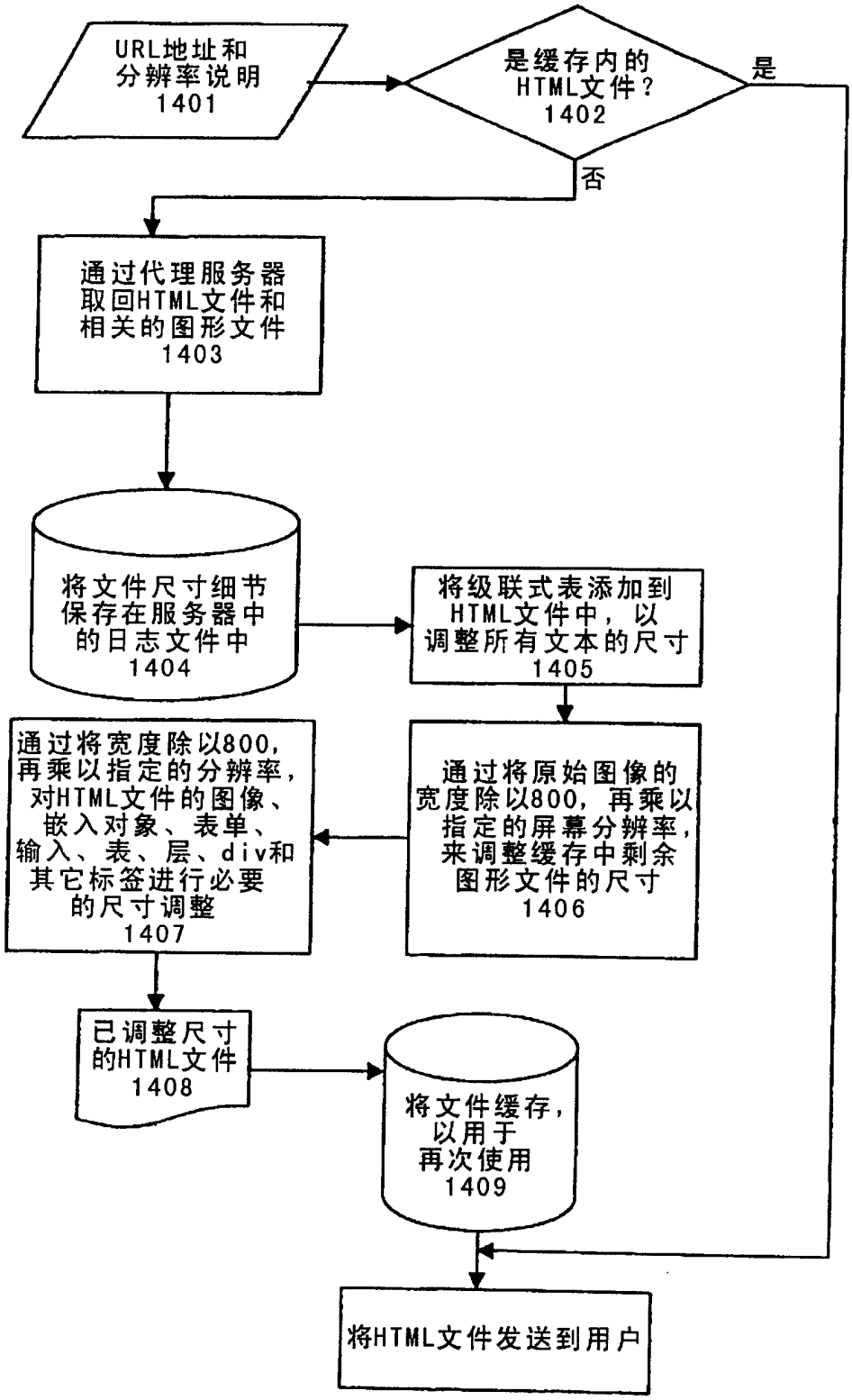
URL地址和
分辨率说明
1401
是缓存内的
HTML文件？
1402
是
否
通过代理服务器
取回HTML文件和
相关的图形文件
1403
将文件尺寸细节
保存在服务器中
的日志文件中
1404
将级联式表添加到
HTML文件中，以
调整所有文本的尺寸
1405
通过将宽度除以800，
再乘以指定的分辨率，
对HTML文件的图像、
嵌入对象、表单、
输入、表、层、div和
其它标签进行必要
的尺寸调整
1407
通过将原始图像的
宽度除以800，再乘以
指定的屏幕分辨率，
来调整缓存中剩余
图形文件的尺寸
1406
已调整尺寸
的HTML文件
1408
将文件缓存，
以用于
再次使用
1409
将HTML文件发送到用户

图14

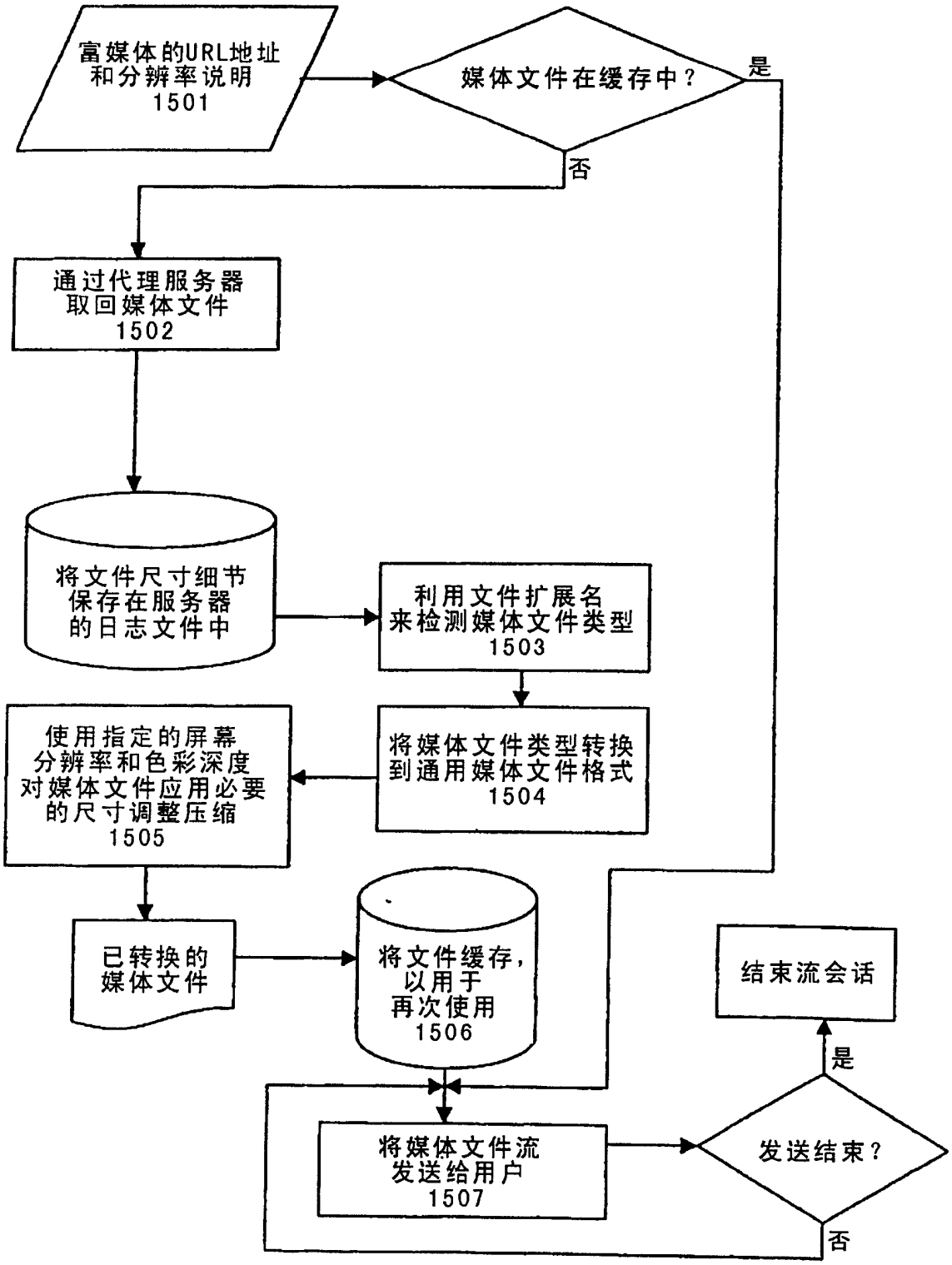
富媒体的URL地址
和分辨率说明
1501
媒体文件在缓存中？
是
否
通过代理服务器
取回媒体文件
1502
将文件尺寸细节
保存在服务器
的日志文件中
利用文件扩展名
来检测媒体文件类型
1503
将媒体文件类型转换
到通用媒体文件格式
1504
使用指定的屏幕
分辨率和色彩深度
对媒体文件应用必要
的尺寸调整压缩
1505
已转换的
媒体文件
将文件缓存，
以用于
再次使用
1506
结束流会话
是
将媒体文件流
发送给用户
1507
发送结束？
否

图15

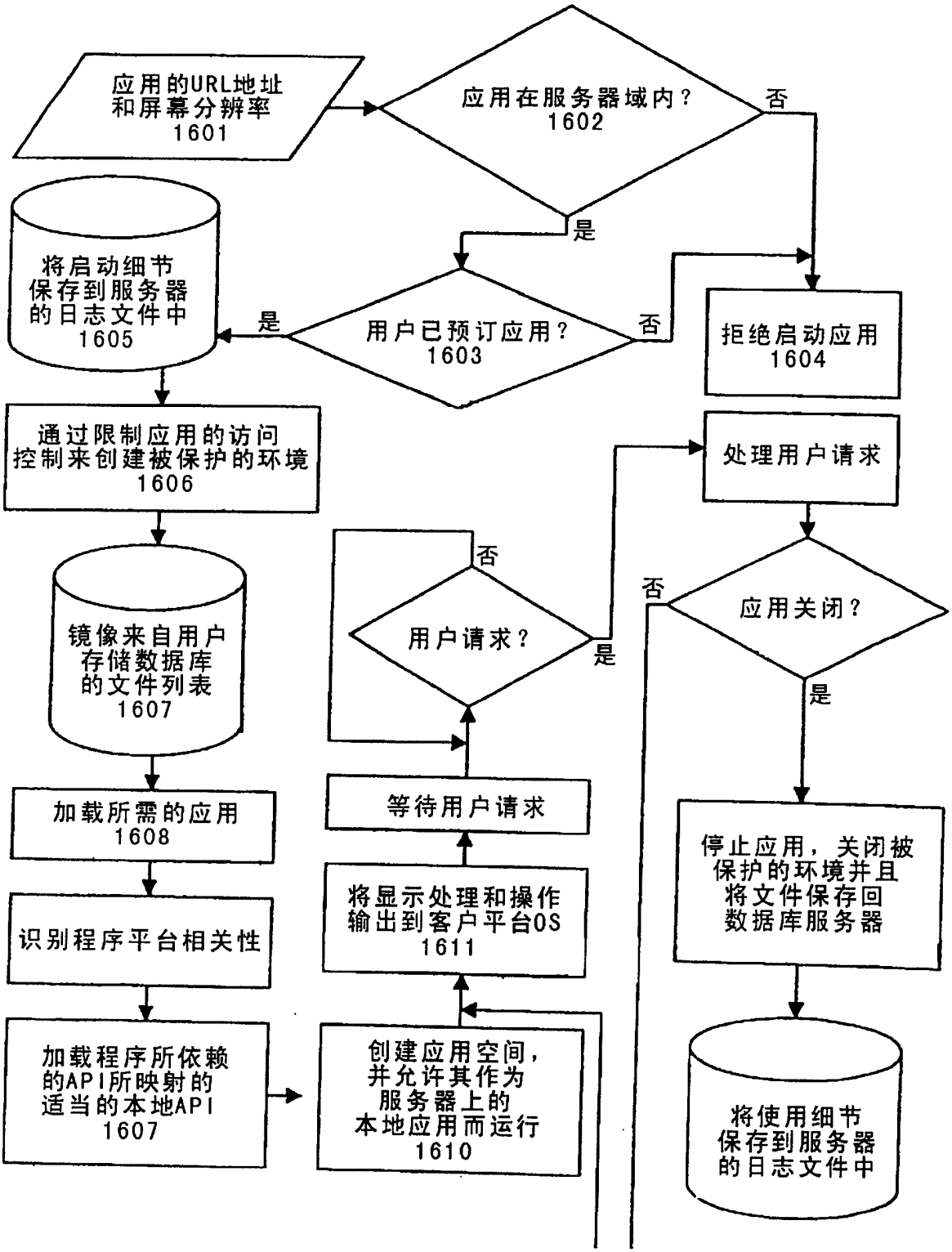
应用的URL地址
和屏幕分辨率
1601
应用在服务器域内？
1602
否
是
将启动细节
保存到服务器
的日志文件中
1605
是
用户已预订应用？
1603
否
拒绝启动应用
1604
通过限制应用的访问
控制来创建被保护的环境
1606
处理用户请求
镜像来自用户
存储数据库
的文件列表
1607
否
用户请求？
是
否
应用关闭？
是
加载所需的应用
1608
等待用户请求
停止应用，关闭被
保护的环境并且
将文件保存回
数据库服务器
识别程序平台相关性
将显示处理和操作
输出到客户平台OS
1611
加载程序所依赖
的API所映射的
适当的本地API
1607
创建应用空间，
并允许其作为
服务器上的
本地应用而运行
1610
将使用细节
保存到服务器
的日志文件中

图16

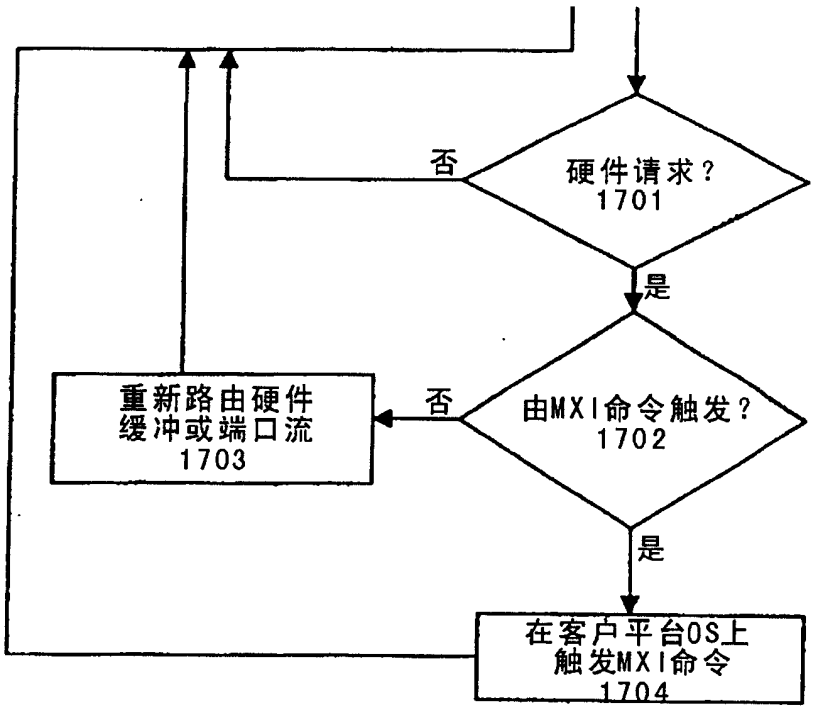
硬件请求？
1701
否
是
由MXI命令触发？
1702
否
是
重新路由硬件
缓冲或端口流
1703
在客户平台OS上
触发MXI命令
1704

图17

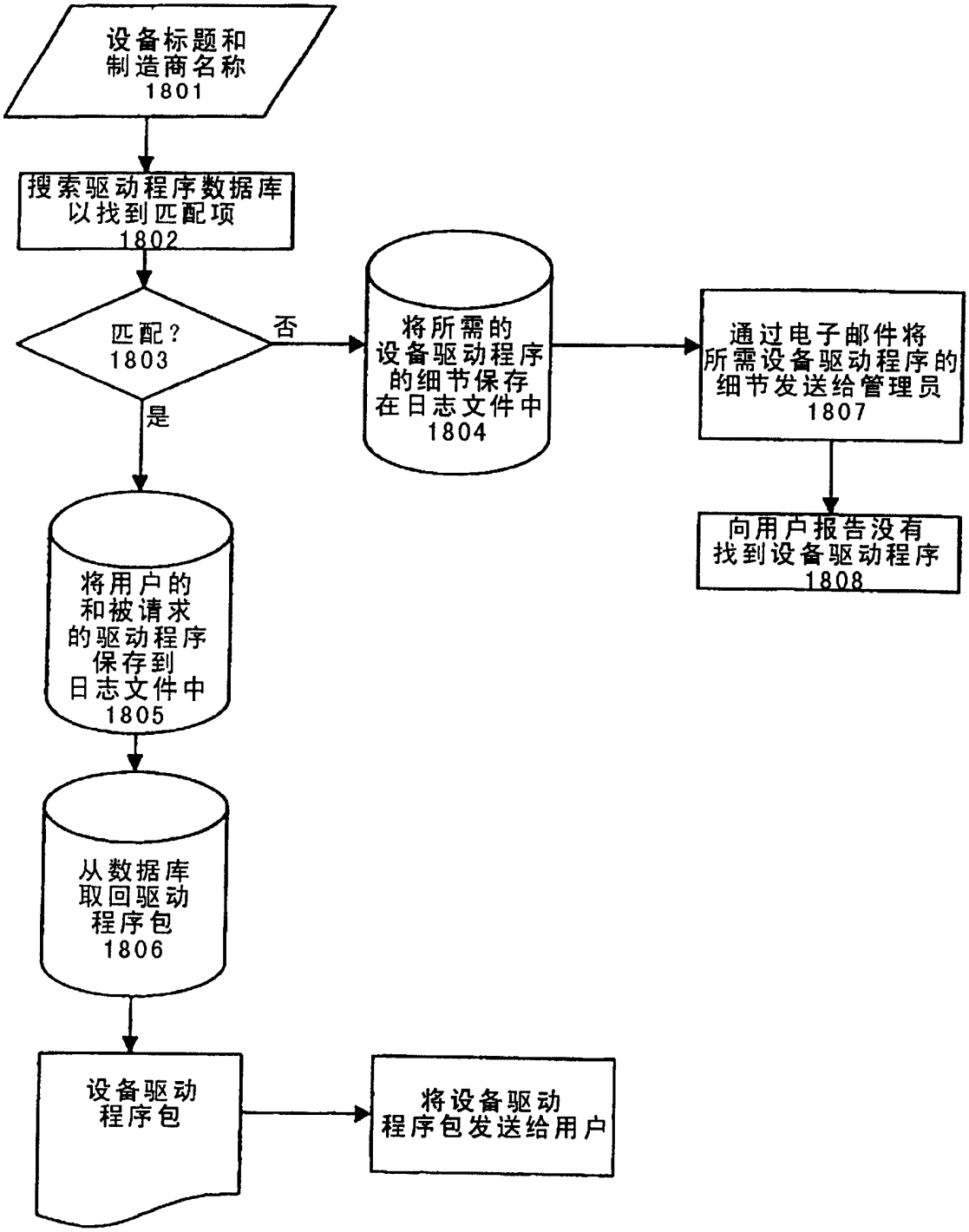
设备标题和
制造商名称
1801
搜索驱动程序数据库
以找到匹配项
1802
匹配？
1803
否
是
将所需的
设备驱动程序
的细节保存
在日志文件中
1804
通过电子邮件将
所需设备驱动程序的
细节发送给管理员
1807
向用户报告没有
找到设备驱动程序
1808
将用户的
和被请求
的驱动程序
保存到
日志文件中
1805
从数据库
取回驱动
程序包
1806
设备驱动
程序包
将设备驱动
程序包发送给用户

图18

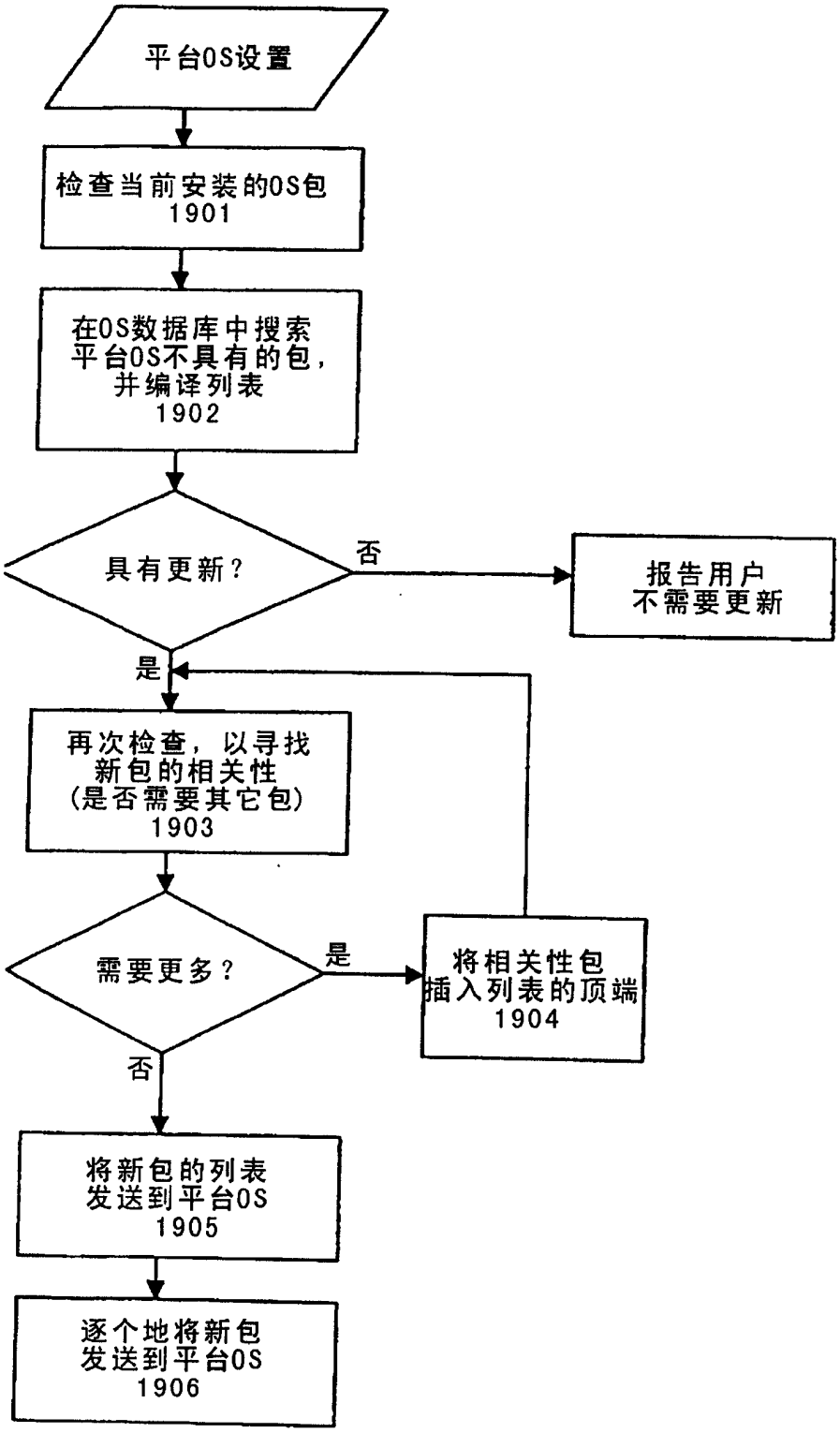
平台OS设置
检查当前安装的OS包
1901
在OS数据库中搜索
平台OS不具有的包，
并编译列表
1902
具有更新？
否
报告用户
不需要更新
是
再次检查，以寻找
新包的相关性
(是否需要其它包)
1903
需要更多？
是
将相关性包
插入列表的顶端
1904
否
将新包的列表
发送到平台OS
1905
逐个地将新包
发送到平台OS
1906

图19

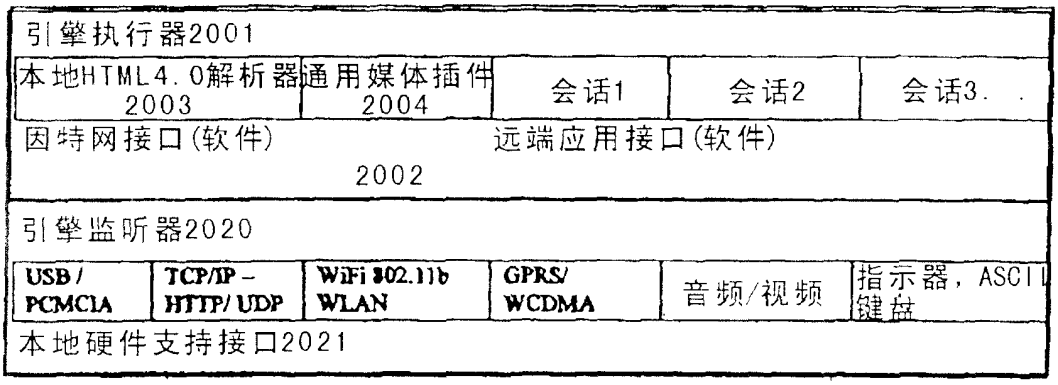
引擎执行器2001
本地HTML4.0解析器 2003
通用媒体插件 2004
会话1
会话2
会话3. .
因特网接口(软件)
远端应用接口(软件)
2002
引擎监听器2020
USB/ PCMCIA
TCP/IP – HTTP/ UDP
WiFi 802.11b WLAN
GPRS/ WCDMA
音频/视频
指示器，ASCII 键盘
本地硬件支持接口2021

图20

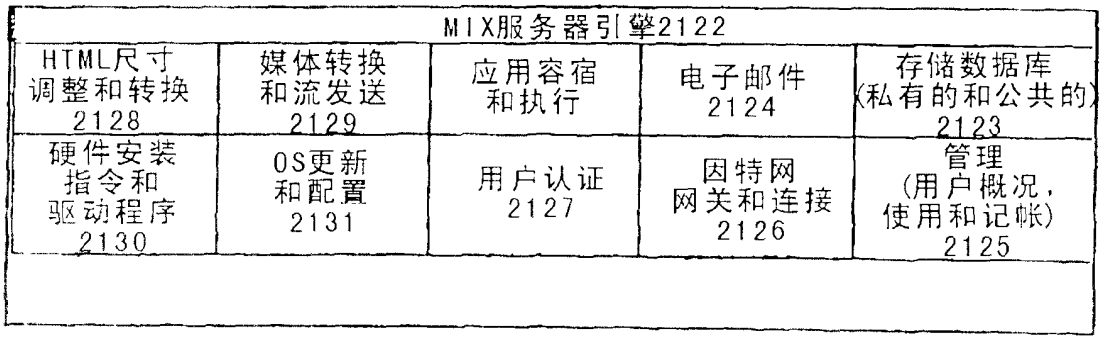
MIX服务器引擎2122
HTML尺寸调整和转换 2128
媒体转换和流发送 2129
应用容宿和执行
电子邮件 2124
存储数据库(私有的和公共的) 2123
硬件安装指令和驱动程序 2130
OS更新和配置 2131
用户认证 2127
因特网网关和连接 2126
管理(用户概况，使用和记帐) 2125

图21

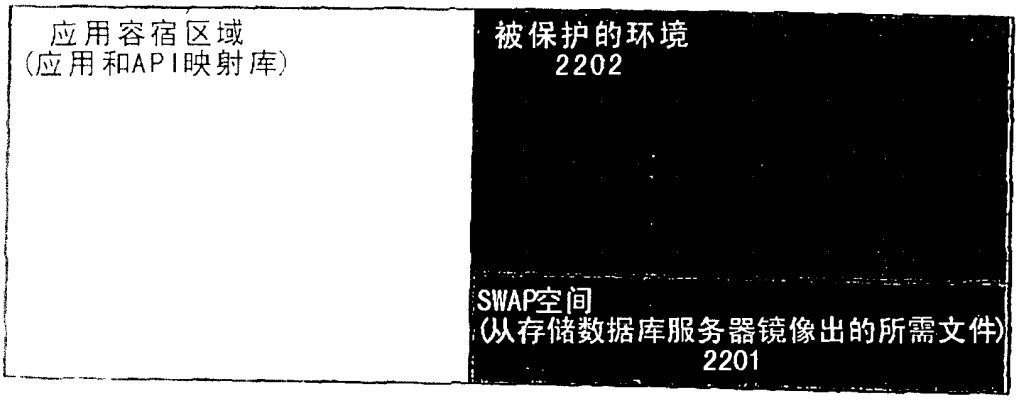
应用容宿区域
(应用和API映射库)
被保护的环境
2202
SWAP空间
(从存储数据库服务器镜像出的所需文件)
2201

图22