[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 17/30 (2006.01)
G06F 9/44 (2006.01)

[12] 发明专利说明书

专利号 ZL 03811171.3

[45] 授权公告日 2008 年 4 月 16 日

[11] 授权公告号 CN 100382076C

[22] 申请日 2003.4.15 [21] 申请号 03811171.3

[30] 优先权

[32] 2002.6.28 [33] US [31] 10/184,255

[86] 国际申请 PCT/GB2003/001620 2003.4.15

[87] 国际公布 WO2004/003787 英 2004.1.8

[85] 进入国家阶段日期 2004.11.16

[73] 专利权人 国际商业机器公司

地址 美国纽约

[72] 发明人 克雷格·亨利·贝克

斯图尔特·厄尔勒·尼克拉斯

韦恩·埃勒莫·维克纳尔

[56] 参考文献

CN1177150A 1998.3.25

US2002/0013779A1 2002.1.31

CN1212401A 1999.3.31

IDL Compiler. Jeff Parsons. http://www.cs.wustl.edu/~doc/RandD/TAO/Status/idl_compiler.html. 2001

The Web services (r) evolution, Part 4Web Services Description Language(WSDL). Graham Glass. ftp://www6.software.ibm.com/software/developer/library/ws.peer4.pdf. 2001

审查员 吉张媛

[74] 专利代理机构 中国国际贸易促进委员会专利商标事务所

代理人 李德山

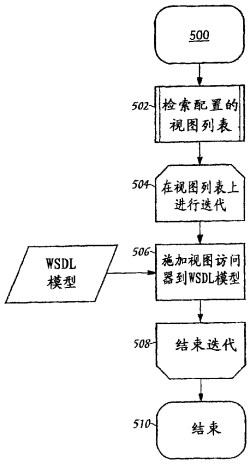
权利要求书 2 页 说明书 10 页 附图 12 页

[54] 发明名称

使用多个编程语言显示和执行 WEB 服务的方法和系统

[57] 摘要

Web 服务可以通过产生描述 Web 服务的数据模型来显示。视图访问器可被施加到产生的数据模型，其中视图访问器对应于预选的域。与预选的域相关联的 Web 服务的视图可响应视图访问器的施加被显示。视图访问器可包括用于封装要在数据模型的部件上执行的操作的访问器模式。



1、一种显示 Web 服务的方法，包括:

产生（204）描述 Web 服务的数据模型;

施加（208）视图访问器到数据模型，其中视图访问器对应于预选的域；以及

响应施加视图访问器的步骤显示(210)与预选的域相关联的 Web 服务的视图，其中视图访问器包括用于封装要在数据模型的部件上执行的操作的访问器模式;

响应显示视图的步骤接收（704）用户输入样本数据；以及

使用该样本数据执行（710）Web 服务。

2、根据权利要求 1 的方法，还包括迭代（208）施加视图访问器到数据模型并对于每个对应于预选的一组域中的一个域的视图显示（210）Web 服务的视图的步骤。

3、根据权利要求 1 的方法，其中数据模型包括数据结构，该数据结构包括对应于 Web 服务的操作的至少一个节点（408），和对应于操作的节点的至少一个子节点（410，412，414），该子节点对应于 Web 服务的输入和输出之一。

4、根据权利要求 3 的方法，其中施加视图访问器的步骤还包括:

传递（558）至少一个节点和至少一个子节点到视图访问器中；以及

填充（560）对应于预选的域的视图以产生表示传递进的节点的对应的视图。

5、一种显示 Web 服务的数据处理系统，包括:

用于产生（204）描述 Web 服务的数据模型的电路;

用于施加（208）视图访问器到数据模型的电路，其中视图访问器对应于预选的域；以及

用于响应施加视图访问器的步骤显示（210）与预选的域相关联的 Web 服务的视图的电路,其中视图访问器包括用于封装要在数据模

型的部件上执行的操作的访问器模式;

用于响应显示视图的步骤接收（704）用户输入样本数据的电路;以及

用于使用视图和样本数据执行（710）调用 Web 服务的 Web 页面的电路。

6、根据权利要求 5 的系统，还包括用于迭代（208）施加视图访问器到数据模型和对于每个对应于预选的一组域中的一个域的视图显示（210）Web 服务的视图的电路。

7、根据权利要求 5 的系统，其中数据模型包括数据结构，该数据结构包括对应于 Web 服务的操作的至少一个节点（408），和对应于操作的节点的至少一个子节点（410，412，414），该子节点对应于 Web 服务的输入和输出之一。

8、根据权利要求 7 的系统，其中用于施加视图访问器的电路还包括:

用于传递（558）至少一个节点和至少一个子节点到视图访问器中的电路; 以及

用于填充（560）对应于预选的域的视图以产生表示传递进的节点的对应的视图的电路。

使用多个编程语言显示和执行 Web 服务的方法和系统

技术领域

本发明大体上涉及数据处理系统，特别是涉及在分布式数据处理系统中访问和浏览 Web 服务。

背景技术

网络化数据处理系统的出现，特别是称为因特网的网络之网，激发了分布式数据处理服务的引入。在这种系统中，通常经由一个或多个网络远程地连接到服务提供商的客户机访问数据处理服务，该服务在远程数据处理系统上实现，且该系统把数据处理行为的结果返回到客户机。使用万维网（WWW）表示的服务并用其图形用户接口（GUI）定位来提供到该分布式数据处理服务的接口已变得很普遍。

通常，在这种分布式处理系统中，客户机发送请求到服务器。该请求可包括可能是到请求的特定的服务的输入的一个或多个参数。

在服务器端，系统构建 Web 页以返回响应到请求客户机。服务器访问包含定义 Web 页的代码的服务器页。嵌入代码中、并用于产生页面，即 HTML 脚本的，是服务器可执行来产生请求的数据处理服务以产生必要的 HTML 脚本在客户机机器显示结果的代码。

运行在客户机机器上的 Web 浏览器是能够解释 HTML 并在连接到客户机机器的、如 CRT 显示器的常规显示器上显示页面的应用程序。市场上能买到的 Web 浏览器包括 Netscape Navigator®、Mozilla，Internet Explorer®、iCab 和 Opera。以这种方式实现分布式计算服务的技术可包括动态服务网页（ASP）和 Java™ 服务网页（JSP）。另外，这种服务可经由与环境无关的过程间通信应用程序接口（API）访问服务器端的应用软件以执行一些或全部请求的任务，这些接口如

DCOM（分布式组件对象模型）、CORBA（公用对象请求代理体系）或远程方法调用（RMI）。为了响应浏览器对页面的执行，应用软件产生动态数据并把数据返回到客户机，然后客户机依照定义页面的代码显示数据。另外，如下面进一步描述的那样，服务器端应用程序不需要位于与页面服务器相同的硬件上，而是可被配置在可能远离客户机和页面服务器的其他的硬件上。

越来越多的 XML 适应性系统的部署引起并不局限于如 DCOM、RMI 或 CORBA 的特定于对象模型的协议的分布式数据处理技术的开发。称为可扩展标记语言的 XML 是用于描述结构化数据的基于标签的标记语言。不象 HTML，XML 标签不是预定义的。XML 是元标记语言。XML 包括一种机制——XML 模式和数据类型定义（DTD）来传达关于文件的结构和数据类型的信息。XML 的规范由万维网联盟（W3C）颁布。XML（和它的派生物）使其能够使用标准因特网协议来访问分布式数据处理服务。这种分布式应用到应用的数据处理实现统称为 Web 服务。可用于描述 Web 服务的 XML 派生物是 Web 服务定义语言（WSDL）。WSDL 文件定义特定的 Web 服务接受和产生的消息。

虽然 WSDL 是一种用于定义 Web 服务的丰富的语言，但是由 WSDL 描述抽象和评估 Web 服务接口对可能使用 Web 服务的 Web 页面开发者来说是一个挑战。Web 开发者有许多他们可用的途径来开发 Web 页面。即，对任何 Web 编程问题，有许多用户可选择的编程解决方案的域（domain）。例如，一些开发者可能更喜欢书写 Java 服务网页（JSP）标签来传递 Web 内容。其他人可能更喜欢用 Java™ 书写并使用 Java™ servelets 来开发内容。还有一些人可能使用 Javascript。这些类型的每种类型的 Web 开发者在分析应用程序中可能使用的 Web 服务时将有不同的编程视图（view）和不同的要求。Web 服务的“粗糙的”WSDL 描述可能不能有效地满足这些开发者的要求。因此，本领域需要评估 Web 服务的机制，籍此 WSDL 信息在与开发者的优选的编程领域一致的角度或者说视图上传送。另外，还

需要使用 Web 服务的该视图执行对 Web 服务的样本调用、并执行对服务的样本调用的机制。

发明内容

上述问题可在一些实施例中通过评估 Web 服务接口和执行对服务的样本调用而至少部分地得到解决。本发明的一种实施例中，显示 Web 服务的方法可包括产生 Web 服务的描述的数据模型的步骤。该方法还可包括把视图访问器施加到产生的数据模型的步骤，其中视图访问器对应于预选的域。该方法还可包括响应视图访问器的施加显示与预选的域相关联的 Web 服务的视图的步骤。视图访问器可包括用于封装在数据模型的部件上执行的操作的访问器模式。

附图说明

本发明将仅通过举例并参照附图加以描述，其中：

图 1 示出了可与本发明结合使用的提供 Web 服务的网络体系结构；

图 2 以流程图形式示出了依照本发明的一种实施例的用于提供 Web 服务的多个视图的一种方法；

图 3 以流程图形式更具体地示出了图 2 的方法的一部分；

图 4 示出了依照本发明原理的 WSDL 页面的示例性的抽象数据模型；

图 5.1 和 5.2 以流程图形式更具体地示出了图 2 的方法的另一部分；

图 6.1－6.3 示出了依照本发明原理的 Web 服务页面的视图的示例性的图形用户接口(GUI)视图；

图 7 以流程图形式示出了依照本发明原理执行 Web 服务的方法；

图 8 以方框图形式示出了依照本发明的一种实施例的用于为访问 Web 服务产生文件组件的数据处理系统；以及

图 9 以方框图形式示出了依照本发明的一种实施例的用于经由标

签库为访问 Web 服务产生文件组件以在 Web 页面中产生动态数据的数据处理系统。

具体实施方式

参照图 1，示出了依照本发明原理的可用于访问 Web 服务的分布式数据处理系统体系结构 100。（体系结构 100 可被理解为分布式数据处理系统体系结构的逻辑视图。换句话说，Web 服务的服务器 114 可视作逻辑上不同的组件，或者在物理上部署在与页面服务器 108 相同的硬件上，或者在物理上部署在与页面服务器 108 不同的硬件上。）目前，Web 服务是经由遵循例如 SOAP 的 XML 消息来访问的。客户机浏览器 104 发起请求 106，该请求要经由如图 1 所示的网络如因特网 110 传送到由图 1 中的页面服务器 108 示出的目标 web 服务器。

页面服务器 108 通过在响应 112 中返回请求的页面以响应请求。请求的页面可包括要动态产生的数据。该动态数据可在客户机机器上本地产生以响应执行定义响应 112 中返回的页面的脚本的客户机浏览器 104。另外，动态数据可由远程过程产生。这还可响应在页面服务器 108 经由一个请求在响应 112 中返回到如 web 服务的服务器 114 的远程服务器的页面中的代码。执行在响应 112 中接收的页面后，页面中对应的代码产生要到 web 服务的服务器 114 的服务请求 116。Web 服务的服务器 114 可执行 web 服务应用程序 118 作为响应，该应用程序产生动态数据。数据在服务响应 120 中返回到页面服务器 108。数据可作为 XML 数据经由 SOAP 消息返回。SOAP 是由 W3C（万维网联盟）颁布的建议标准。（SOAP 1.2 的起草的规范可见简单对象访问协议 1.2，http://www.w3.org/TR/SOAP12.）（SOAP 是简单对象访问协议的首字母缩写；从版本 12 起就不再是缩写。）页面服务器 108 把数据合并入页面并在响应 112 中把该页面发送到客户机浏览器。

要请求 web 服务产生动态数据，在响应 112 中发送到客户机浏览器 104 的页面必须具有访问 web 服务的适当的代码。例如，请求可嵌入 SOAP 消息中。但是，如前所述，web 服务通常在 WSDL 文件中定

义。产生 web 服务的包括 SOAP 视图的多个视图的方法，将与图 3－5 结合讨论。（在特定的域的情形下，视图可指 web 服务 WSDL 的视图。除 SOAP 外，域可包括 Java、JSP 和 Visual Basic。

这里提供的流程图并不一定指示在本发明的实施例中执行的操作的顺序。用这些流程图揭示的步骤可被并行执行。该流程图指示了那些可被执行以生成可用于产生和显示 Web 服务的多个视图的操作的考虑事项。还应注意，显示的顺序是示例性的，并不一定意味着步骤必须按所示的顺序执行。

现在参照图 2，该图以流程图形式示出了依照本发明的一种实施例提供 web 服务的多个视图的方法 200。在步骤 202，接收到定义 web 服务的 WSDL 文件的 URI(统一资源标识符)。例如，URI 可从运行在浏览 web 服务以在正在开发的 web 页面中使用 web 服务的 web 页面开发者的客户机机器上的 web 浏览器接收。在步骤 204 创建如 WSDL 文件中定义的 web 服务的数据模型。步骤 204 将结合图 3 做更具体的描述。

在步骤 206，使用来自步骤 204 的数据模型创建视图。视图的产生将结合图 6 进一步描述。

在步骤 208，步骤 206 中创建的视图上的迭代被起动。对每个视图，视图在步骤 210 中显示，视图上的迭代在步骤 212 终止。过程 200 在步骤 214 结束。

现在参照图 3，该图更具体地示出了图 2 的步骤 204。在步骤 302，步骤 202 中接收的 URI 被检索，在步骤 304 确定对应于 URI 的 WSDL 文件是否在高速缓冲存储器中。如本领域普通技术人员所认识到的，WSDL 数据模型可与高速缓存 Web 页面类似的方式高速缓存以减小带宽需求。这样，先前产生的 WSDL 文件的数据模型可更快地返回并保存了处理带宽。如果 WSDL 数据模型不在高速缓冲存储器中，在步骤 306WSDL 文件被解析并创建其数据模型。

WSDL 页面的数据模型可以是页面的树状表示。参照图 4，显示了简化的 WSDL 页面的示例性数据模型 400。树包括多个节点 402－

414。节点 402 识别了特定的 Web 服务。节点 404、406 和 408 表示被服务执行的操作。操作 408 要有两个输入，输入 410 和输入 412，以及产生输出 414。

回到图 3，在步骤 308，数据模型被添加到高速缓冲存储器。

相反地，如果 WSDL 页面在高速缓冲存储器中，先前高速缓冲存储的 WSDL 数据模型在步骤 310 被检索。在步骤 312，数据模型被输入到图 2 的步骤 206，步骤 204 终止。

现在参照图 5.1 和 5.2，图 5.1 和图 5.2 更具体地示出了图 3 的步骤 306。在步骤 502，配置的视图的列表被检索。配置的视图可指访问器模式被实现的视图，访问器模式将结合步骤 506 进一步讨论。因此，例如，该配置的视图的列表可包括 Java 视图、JSP 视图、Visual Basic 视图、SOAP 视图和 粗糙 WSDL 视图。

在步骤 504，一个在视图列表上的迭代环被输入。在步骤 506，视图访问器模式为列表中的每个视图被施加于 WSDL 数据模型。如上文描述的，WSDL 数据模型可表示为树状数据结构，其中它的节点对应于的 web 服务操作，和到其的各输入和输出。访问器模式是封装在数据模型的部件上执行的操作的机制。（访问器模式在面向对象的编程领域被称为访问器设计模式，或可供选择地，这里为了简单称为“访问器”）。换句话说，视图访问器是封装 WSDL 数据模型的节点上的操作的对象，该操作特别是提供 WSDL 节点的视图的操作。因此，例如，Java 视图访问器可提供 WSDL 数据模型到 Java 片断。因此，调用这个在如图 4 的节点 404 的节点上表示 Web 服务操作的视图访问器，可提供该节点到 HTML 文本域的 Java 代码片断，该片断表示了特定的操作如何被调用。类似地，标签库视图访问器可把节点作为表示操作如何被调用的标签库提供。因此，视图访问器抽象关于 WSDL 数据模型（它本身是 Web 服务的 WSDL 描述的表示）中的节点的信息并把它嵌入到对应的视图中。注意，依照关于访问器模式的面向对象的编程原理，视图访问器可以是抽象的母访问器类的具体的子类。（例如， 见 Erich Gamma, Richard Helm, Ralph Johnson and John

Vlissides, Design Patterns, Elements of Reusable Object-Oriented Software(Addison Wesley, 1995)，第 5 章，331－344 页）

步骤 506 可被跨整个 WSDL 数据模型树的访问器执行。在每个节点，节点“接受”发送消息到视图访问器的视图访问器。换句话说，节点调用视图访问器，把它自己传递到其中，视图访问器执行它的方法以增加它产生的特定的视图。这样，因为节点本身重复调用视图访问器，且节点自动地传递到视图访问器，该视图访问器可执行适当的多态方法以产生对应于数据结构的调用节点的视图，即 WSDL 文件的树状数据模型。这些操作在图 5.2 的步骤 552－562 中示出。如图 5 所示， WSDL 数据模型从图 4 的步骤 412 输入到步骤 506。 步骤 506 对在步骤 504 输入的迭代环的列表中的每个视图进行重复。迭代环在步骤 508 终止，在步骤 510 处步骤 306 结束。

图 6.1－6.3 示出了示例性的 GUI 窗口 602a－c，分别显示了提供天气数据的 Web 服务的 JSP 标签库视图、Java™、以及 SOAP 消息视图。(JSP 标签库在名称为“Systems and Methods For Accessing Web Services Using A Tag Library”、 序号为 10/185,796 的同一申请人拥有的未决的美国专利申请（AUS9-2002-0330US1）中被讨论。例如，该天气服务可用于检索天气数据以在 Web 页面中显示当前气温和特定的地点的天气预报。该服务的 WSDL 文件可规定，当浏览器发送请求以产生数据，调用“GetTemperature”方法，需要的输入是邮递区号或其他的地点标识符。相应地，在窗口 602a，操作 604 被规定为标签库中对“GetTemperature”的调用。类似地，输入类型 608 被规定为“zipcode”。这样，服务将要为其提供天气数据的地点通过设置邮递区号被规定。如下结合图 7 的描述，开发者可进行样本调用以测试服务，值 610 已被用户设置为邮递区号“78758”。如下面讨论的，用户可通过“点击”按钮 612 来执行服务。类似地，Java 视图，窗口 602b 显示“GetTemperature”方法 614，以自变量“zipcode”616 作为输入视图。以同样的方式，SOAP 视图，窗口 602c 包括“GetTemperature”操作 618，和 SOAP 消息正文 622 中的“zipcode”输入 620。

现在参照图 7，该图以流程图形式示出了执行对 web 服务的样本调用的过程 700。过程 700 可与依照图 3 的方法产生的视图结合使用。在步骤 702，响应用户输入接收视图选择。在步骤 704，接收用户样本数据。该样本数据可分别输入到视图窗口，如图 6.1－6.3 中示出的视图窗口 602A－602C 中的一个。在步骤 706，执行请求被接收。用户可通过激活“按钮”或相似的用户输入设备来发起执行请求。在步骤 708，在与步骤 702 中选定的特定的视图相关联的域内对应的代码被插入到 web 服务页面。在步骤 710，用户的客户机执行页面，例如，通过如图 1 的浏览器 104 的浏览器。Web 页面的执行以在上文结合图 1 描述的方式产生对 web 服务的请求。该请求包括客户机提供的在步骤 704 接收的示例性的输入数据。为响应样本数据，如图 1 的 web 服务应用程序 118 的 web 服务应用程序产生基于样本数据的结果，并把如 SOAP 消息的包含结果的消息返回到客户机显示器 1。

图 8 描绘了一种实施本发明的典型的硬件环境，它示出了依照本发明的数据处理系统 800 的一种示例性的硬件配置。例如，图 1 的客户机 104 可依照数据处理系统 800 实现。系统 800 包括中央处理单元（CPU）810，如常规微处理器，和多个经由系统总线 812 互连的其他的单元。数据处理系统 800 包括随机存储器（RAM）814、只读存储器（ROM）816 和用于连接如磁盘机 820 到总线 812 的外围设备的输入/输出（I/O）适配器 818、用于连接键盘 824、鼠标 826 和/或如触摸屏设备(未显示)的到总线 812 的其他用户接口设备的接口适配器 822。系统 800 还包括用于连接数据处理系统 800 到能使数据处理系统与其它系统通信的数据处理网络的通信适配器 834，和用于连接总线 812 到显示设备 838 的显示适配器 836。CPU810 可包括其他在这里未显示的电路，这些电路可能包括在微处理器中常见的电路，例如，执行单元、总线接口单元、运算逻辑单元等等。CPU810 还可位于单个集成电路上。

显示器 838 通过显示适配器 836 连接到系统总线 812。如此，用户能够通过键盘 824、轨迹球 832 或鼠标 826 输入到系统以及经由扬

声器 828 和显示器 838 接收来自系统的输出。

本发明的优选实现包括作为程序化执行这里描述的一种方法或成套方法的计算机系统的实现，和作为计算机程序产品。根据计算机系统实现，执行该方法或成套方法的指令集位于如上面的描述而配置的一个或多个计算机系统的随机存储器 814 中。这些指令集结合执行它们的系统组件可产生多个 Web 服务视图，并执行对 Web 服务的样本调用。直至被计算机系统要求，指令集可作为计算机程序产品存储在另一个计算机存储器中，例如，硬盘驱动器 820（它可包括可擦除存储器，如在硬盘驱动器 820 中偶然使用的光盘或软盘）。另外，计算机程序产品还能存储在其他计算机，并在需要时被通过网络或如因特网的外部网络传送到用户的工作站。本领域技术人员将理解，指令集的物理存储物理地改变了存储介质，所以介质承载了计算机可读信息。该改变可以是电的、磁的、化学的、生物的、或其他一些物理变化。当方便以指令、符号、字符等术语描述本发明时，读者应记住这些所有的和相似的术语应和适当的物理部件相关联。

注意，本发明可能描述如比较、确认、选择、识别的术语，或其他能与人力操作员相关联的术语。但是，对形成至少一种实施例的一部分的至少多个这里描述的操作，都不需要人力操作员的动作。描述的操作大部分是处理电信号以产生其他电信号的机器操作。

图 9 示出了依照主题发明的数据处理系统 900 的示例性的硬件配置。例如，图 1 的页面服务器 108 和 Web 应用程序服务器 114 可依照数据处理系统 900 加以实现。系统 900 包括中央处理单元（CPU）910，如常规微处理器，和许多经由系统总线 912 相互连接的其他的单元。数据处理系统 900 包括随机存储器（RAM）914、只读存储器（ROM）916 和用于连接外围设备的输入/输出（I/O）

适配器 918，如磁盘机 920 到总线 912。系统 900 还包括用于连接数据处理系统 900 和数据处理网络，使数据处理系统能够和其他系统通信的通信适配器 934。CPU910 可包括其他在这里未显示的电路，该电路将包括在微处理器中常见的电路，例如，执行单元、总线接口

单元、运算逻辑单元等等。CPU910 还可位于单个集成电路上。

本发明的优选实现方案包括作为被编程以执行这里描述的一种方法或多种方法的计算机系统的实现方案，和作为计算机程序产品的实现方案。根据计算机系统的实现方案，执行该一种方法或多种方法的指令集位于如上所述总体上配置的一个或多个计算机系统的随机存储器 914 中。这些指令集结合执行它们的系统组件可执行对 Web 服务的样本调用。指令集可作为计算机程序产品存储在另一个计算机存储器，例如硬盘驱动器 920（它可包括可拆卸存储器，如在硬盘驱动器 920 中可能会使用的光盘或软盘），直至被计算机系统要求为止。另外，计算机程序产品还可存储在其他计算机，并在需要时通过网络或如因特网的外部网络传送到用户的工作站。本领域技术人员将理解，指令集的物理存储物理上改变了存储于其上的介质，以使介质承载计算机可读信息。该改变可以是电的、磁的、化学的、生物的、或其他一些物理变化。虽然用指令、符号、字符等术语描述本发明是很方便的，读者应记住所有这些术语和类似的术语应和适当的物理部件相关联。

图1

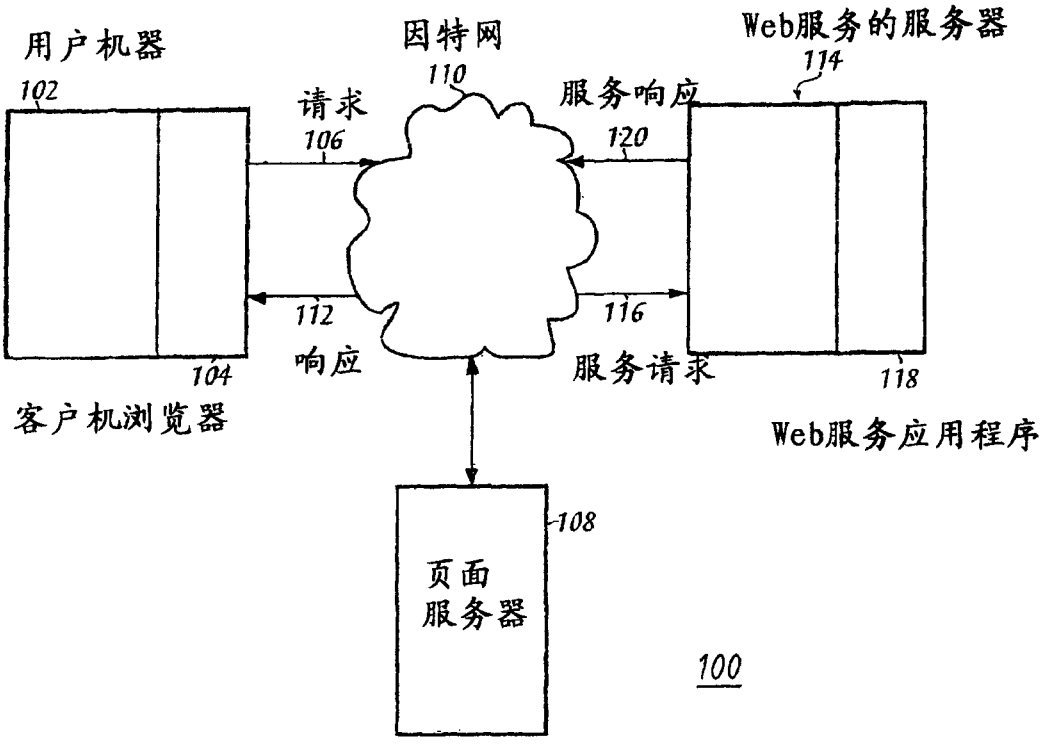

用户机器
102
因特网
110
Web服务的服务器
114
请求
106
服务响应
120
112
响应
116
服务请求
104
客户机浏览器
118
Web服务应用程序
108
页面
服务器
100

图2

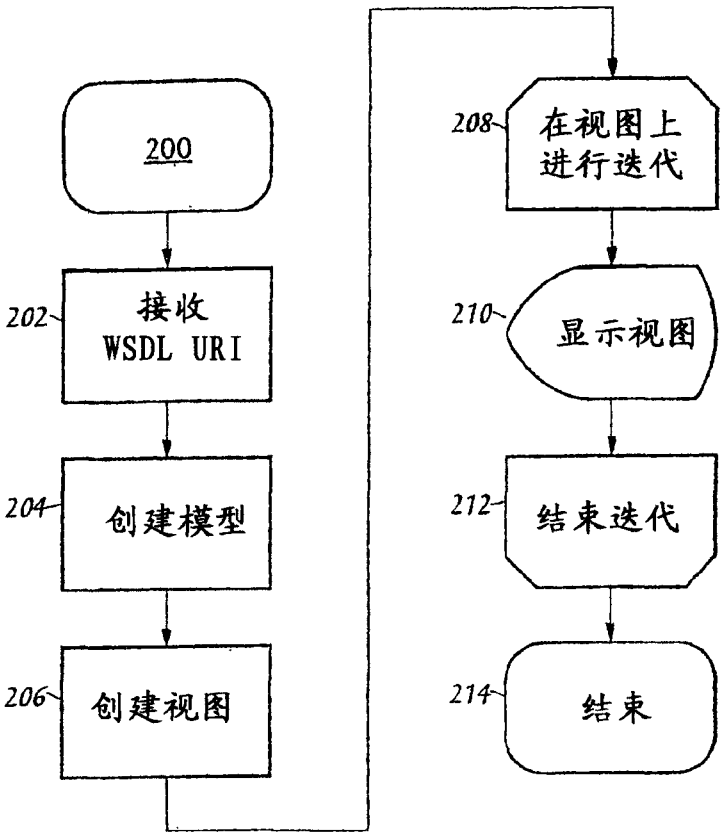

200
202
接收
WSDL URI
204
创建模型
206
创建视图
208
在视图上
进行迭代
210
显示视图
212
结束迭代
214
结束

图 3

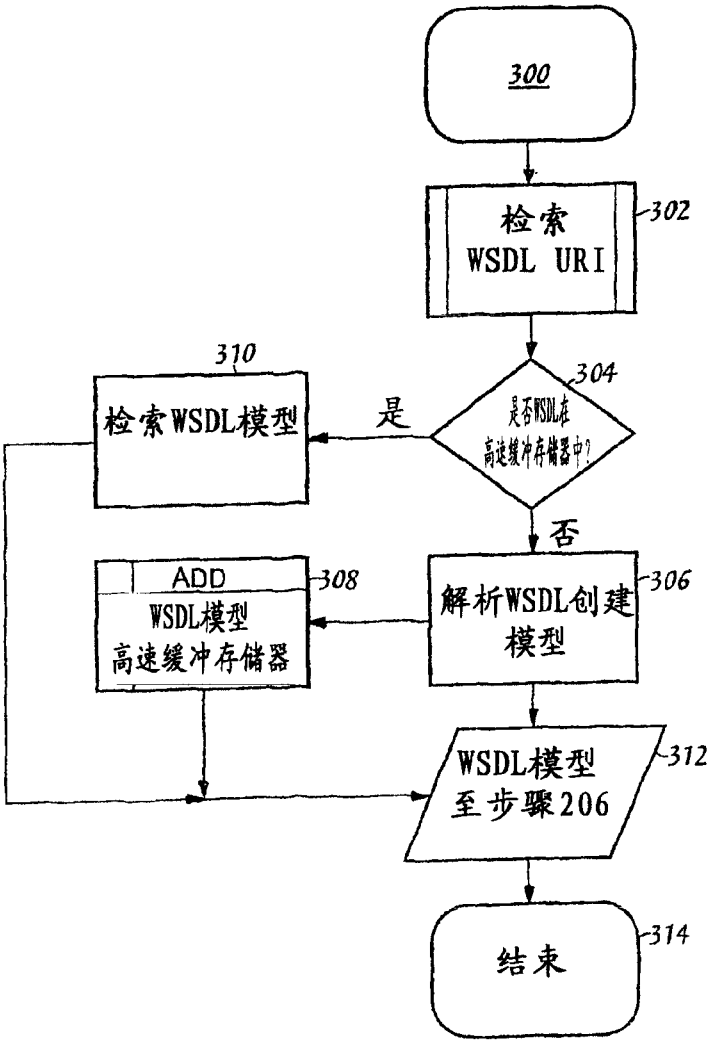

300
检索
WSDL URI
302
310
检索WSDL模型
是
304
是否WSDL在
高速缓冲存储器中?
否
ADD
WSDL模型
高速缓冲存储器
308
解析WSDL创建
模型
306
WSDL模型
至步骤206
312
结束
314

图4

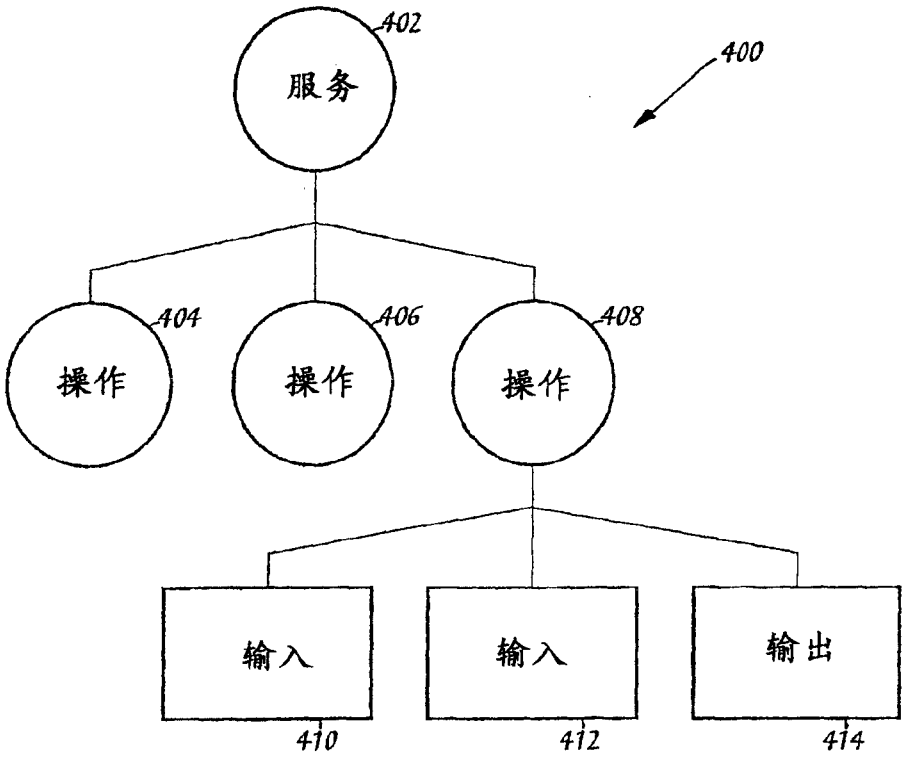


图5.1

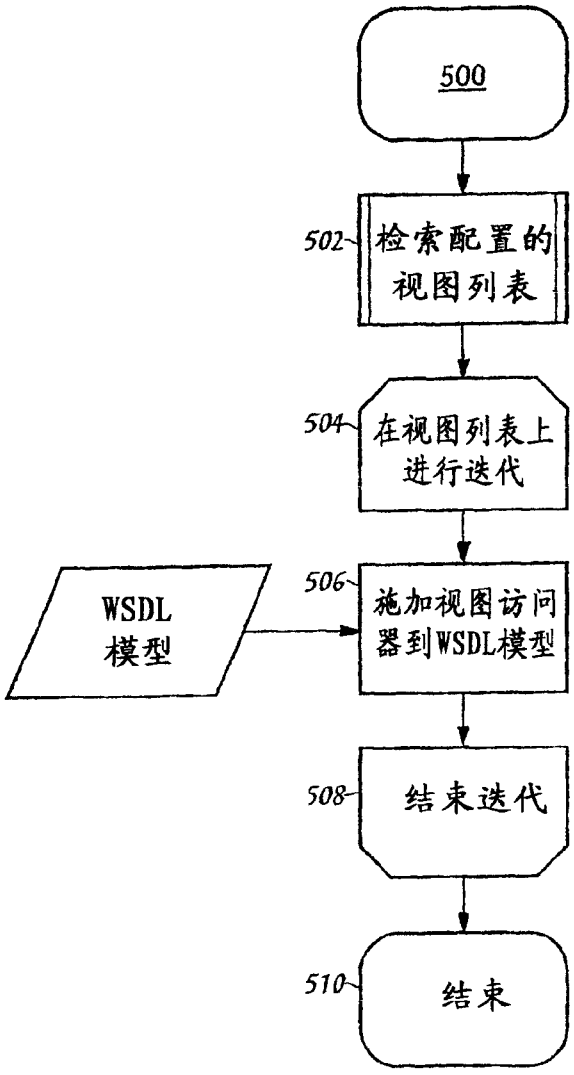

500
502
检索配置的视图列表
504
在视图列表上进行迭代
WSDL模型
506
施加视图访问器到WSDL模型
508
结束迭代
510
结束

图5.2

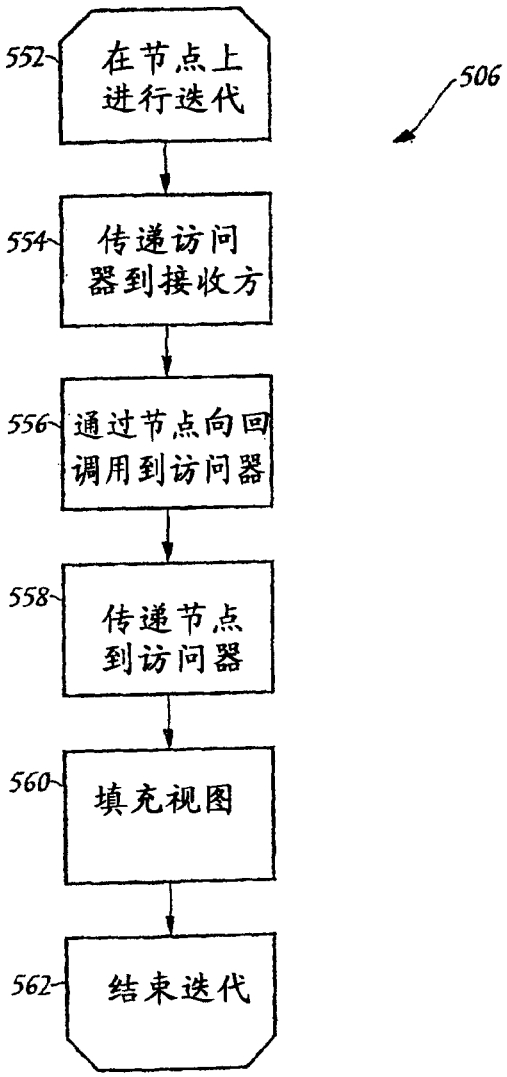

552
在节点上进行迭代
506
554
传递访问器到接收方
556
通过节点向回调用到访问器
558
传递节点到访问器
560
填充视图
562
结束迭代

图6.1

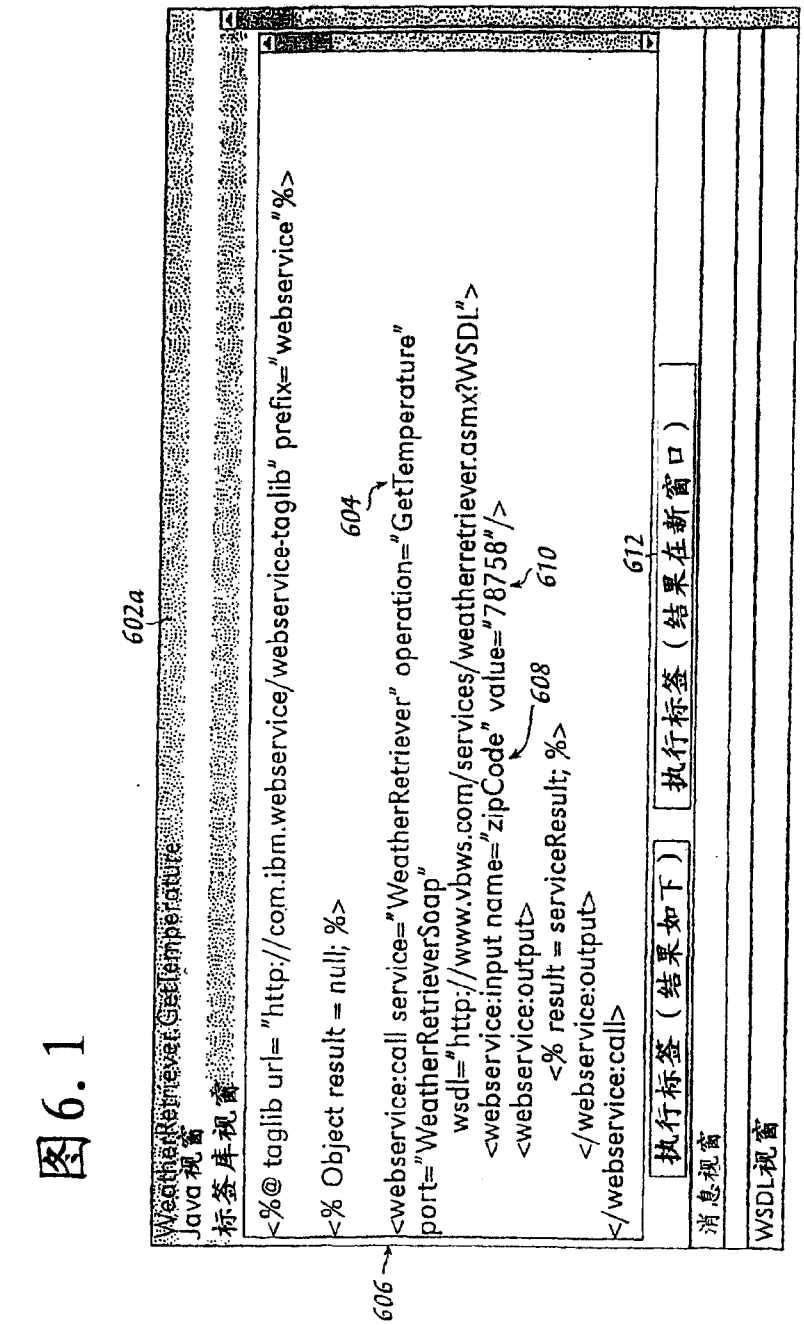

WeatherRetriever.GetTemperature
Java 视窗
标签库视窗
602a
<%@ taglib url= "http://com.ibm.webservice/webservice-taglib" prefix="webservice"%>
<% Object result = null; %>
<webservice:call service="WeatherRetriever" operation="GetTemperature"
port="WeatherRetrieverSoap"
wsdl="http://www.vbws.com/services/weatherretriever.asmx?WSDL">
<webservice:input name="zipCode" value="78758"/>
<webservice:output>
<% result = serviceResult; %>
</webservice:output>
</webservice:call>
606
604
608
610
612
执行标签（结果如下）
执行标签（结果在新窗口）
消息视窗
WSDL视窗

图6.2

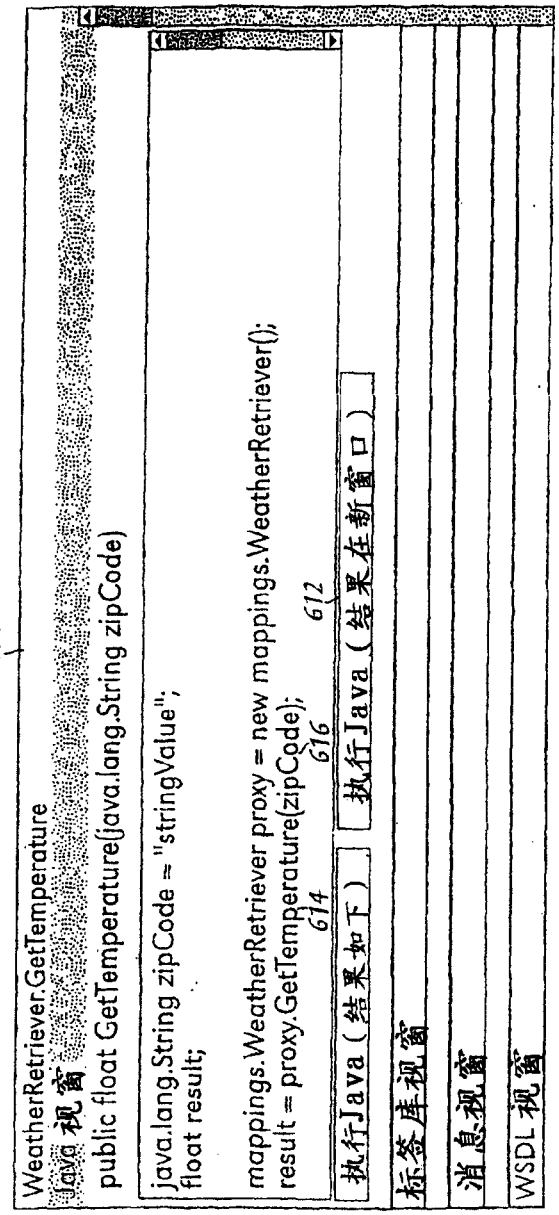

602B
WeatherRetriever.GetTemperature
Java 视窗
public float GetTemperature(java.lang.String zipCode)
java.lang.String zipCode = "stringValue";
float result;
mappings.WeatherRetriever proxy = new mappings.WeatherRetriever();
result = proxy.GetTemperature(zipCode);
614
612
616
执行Java（结果如下）
执行Java（结果在新窗口）
标签库视窗
消息视窗
WSDL 视窗

图6.3

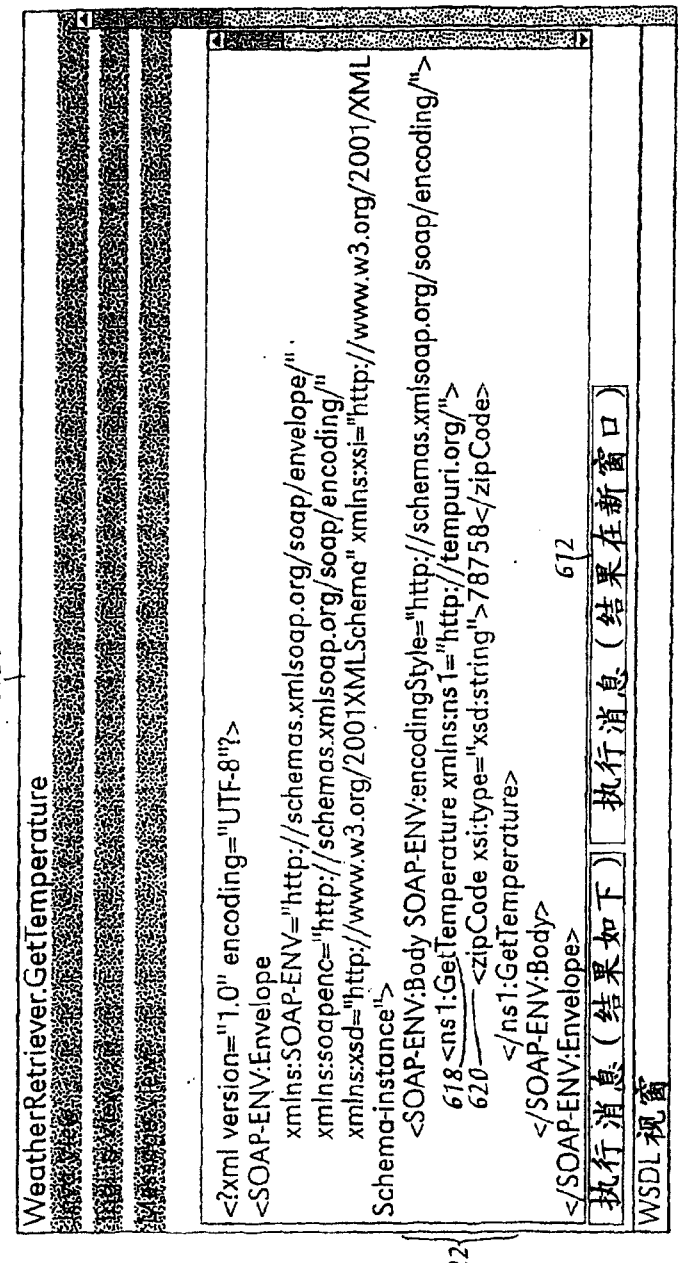

602C
WeatherRetriever.GetTemperature
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001XMLSchema" xmlns:xsi="http://www.w3.org/2001/XML
Schema-instance">
<SOAP-ENV:Body SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
618 <ns1:GetTemperature xmlns:ns1="http://tempuri.org/">
620 <zipCode xsi:type="xsd:string">78758</zipCode>
</ns1:GetTemperature>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
612
622
执行消息（结果如下）
执行消息（结果在新窗口）
WSDL视窗

图7

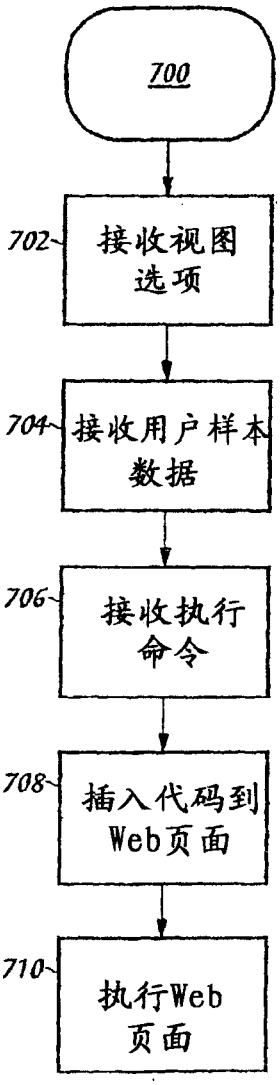
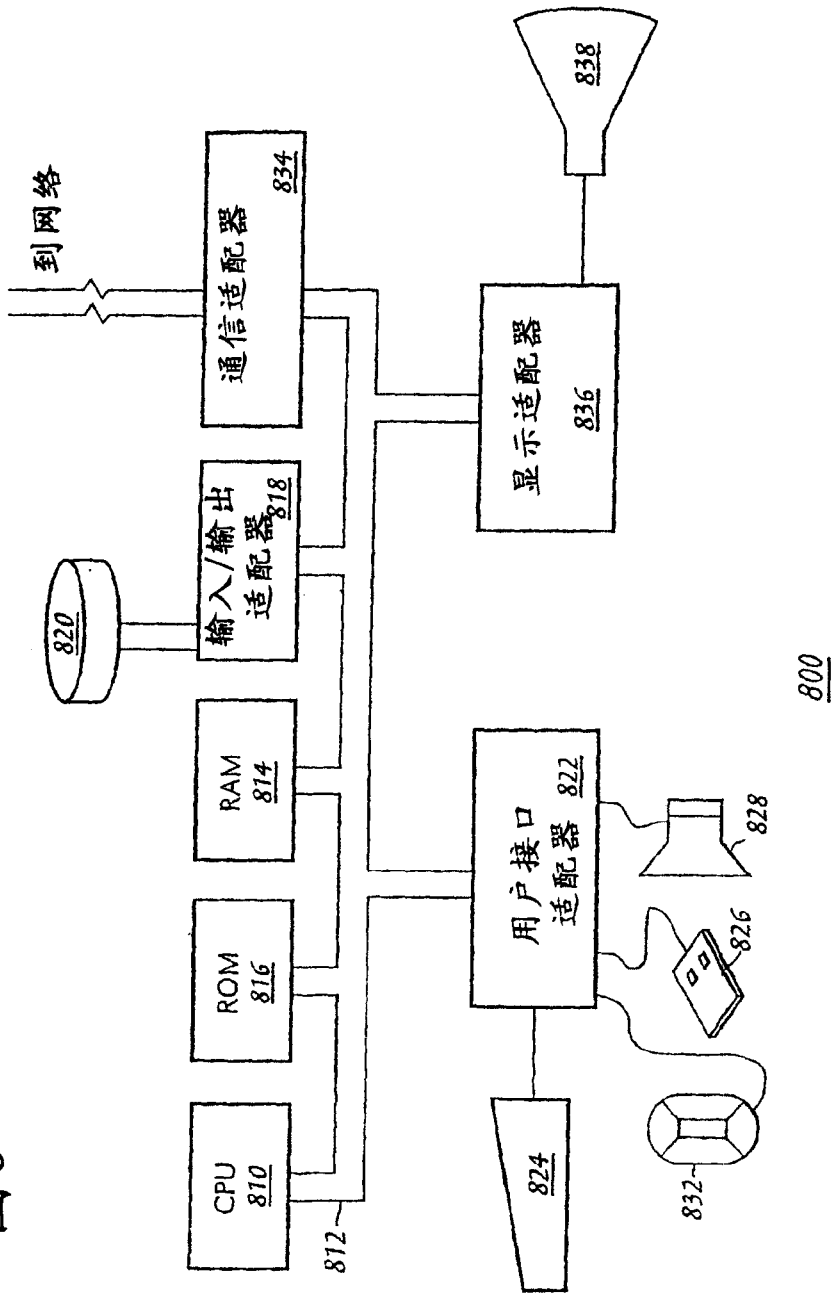

700
702
接收视图选项
704
接收用户样本数据
706
接收执行命令
708
插入代码到Web页面
710
执行Web页面

图8


到网络
通信适配器
834
838
显示适配器
836
输入/输出
适配器
818
820
RAM
814
ROM
816
CPU
810
812
用户接口
适配器
822
828
826
824
832
800

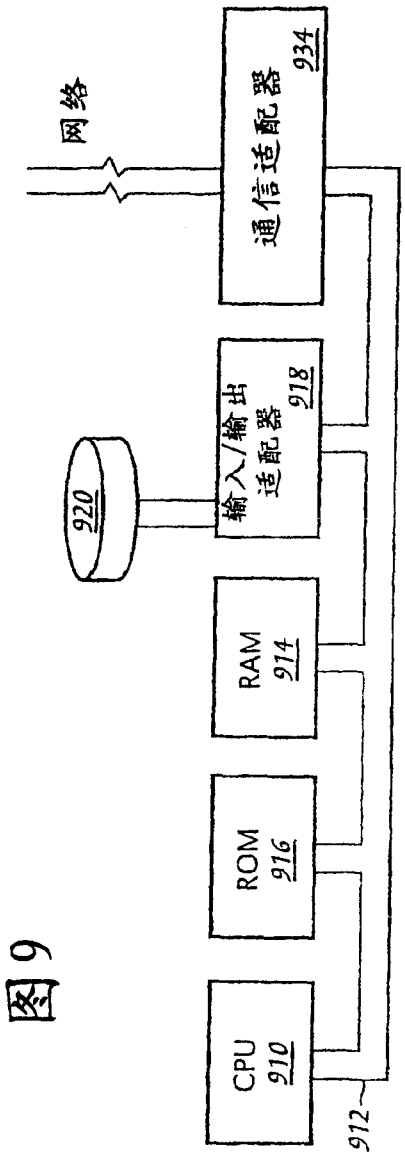


图9