[19] 中华人民共和国国家知识产权局

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

[12] 发明专利申请公布说明书

[21] 申请号 200610092851.9

[43] 公开日 2007 年 5 月 2 日

[11] 公开号 CN 1955922A

[22] 申请日 2006.6.16
[21] 申请号 200610092851.9
[30] 优先权
[32] 2005.10.27 [33] US [31] 11/260,576
[71] 申请人 国际商业机器公司
地址 美国纽约
[72] 发明人 尼克尔·古普塔
杰弗里·M.·阿切特曼
布赖恩·G.·瓦斯伯格
布赖恩·R.·摩尔

[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 颖

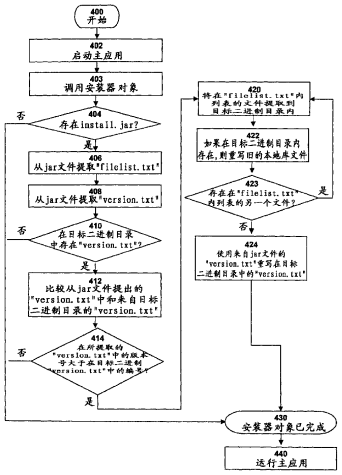
权利要求书 2 页 说明书 15 页 附图 4 页

[54] 发明名称

用于动态提供本地库及其相关性的方法和系统

[57] 摘要

提供了一种将本地代码库从服务器提供给用解释编程语言编写的应用以在远程计算机上执行的方法、系统和程序。 从解释编程语言应用的主类内调用安装器对象。 该安装对象从包括表示本地代码库的至少一个本地代码库文件名的服务器提取文件列表。 将用本地代码库文件名表示的本地代码库安装到远程计算机上的目标目录，随后在远程计算机上调用解释编程语言应用的主类。 还将附加本地代码库相关性安装到在远程计算机上的目标目录。



1. 一种从服务器向解释编程语言应用提供本地代码库以在远程计算机上执行的方法，包括:

从解释编程语言应用的主类中调用安装器对象;

提取包含至少一个本地代码库文件名的文件列表，其中该本地代码库文件名表示在远程计算机上的目标目录和本地代码库;

将用本地代码库文件名表示的本地代码库安装到目标目录; 和

在远程计算机上调用解释编程语言应用的主类。

2. 权利要求 1 的方法，其中该文件列表还包括表示本地代码库相关性的至少一个附加库文件名。

3. 权利要求 2 的方法，还包括:

将由附加库文件名表示的本地代码库相关性安装到远程计算机上的目标目录。

4. 权利要求 1 的方法，还包括:

从服务器提取最新版本文件，其中最新版本文件表示服务器上的文件列表的最新版本编号;

在远程计算机上的目标目录中定位当前版本文件，其中当前版本文件表示在远程计算机上的文件列表的当前版本编号; 和

比较当前版本文件和最新版本文件中的版本编号。

5. 权利要求 4 的方法，还包括:

用最新版本文件重写目标目录中的当前版本文件。

6. 权利要求 3 的方法，还包括:

用本地代码库文件和至少一个本地代码库相关性重写远程计算机上的至少一个较旧库文件。

7. 权利要求 1 的方法，其中解释编程语言应用是 Java 应用，并且目标目录是 Java 运行时环境（JRE）二进制目录。

8. 一种从服务器向解释编程语言应用提供本地代码库以在远程计算机上执行的系统，包括:

用于从解释编程语言应用的主类中调用安装器对象的装置;

用于提取包含至少一个本地代码库文件名的文件列表的装置，其中该本地代码库文件名表示在远程计算机上的目标目录和本地代码库;

用于将用本地代码库文件名表示的本地代码库安装到目标目录的装置; 和

用于在远程计算机上调用解释编程语言应用的主类的装置。

9. 权利要求 8 的系统，其中该文件列表还包括表示本地代码库相关性的至少一个附加库文件名，还包括:

用于将由附加库文件名表示的本地代码库相关性安装到远程计算机上的目标目录的装置。

10. 权利要求 8 的系统，还包括:

用于从服务器提取最新版本文件的装置，其中最新版本文件表示服务器上的文件列表的最新版本编号;

用于在远程计算机上的目标目录中定位当前版本文件的装置，其中当前版本文件表示在远程计算机上的文件列表的当前版本编号; 和

用于比较当前版本文件和最新版本文件中的版本编号的装置。

11. 权利要求 10 的系统，还包括:

用于用最新版本文件重写目标目录中的当前版本文件的装置。

12. 权利要求 9 的系统，还包括:

用于用本地代码库文件和至少一个本地代码库相关性重写远程计算机上的至少一个较旧库文件的装置。

13. 权利要求 8 的系统，其中解释编程语言应用是 Java 应用，并且目标目录是 Java 运行时环境（JRE）二进制目录。

14. 一种在计算机可读介质内的计算机程序产品，用于将本地代码库从服务器提供给解释编程语言应用以在远程计算机上执行，包括用于执行权利要求 1 - 7 中任一权利要求的方法的步骤的指令。

用于动态提供本地库及其相关性的方法和系统

技术领域

本发明涉及数据处理系统，并涉及下载、更新和运行在数据处理系统上以解释编程语言编写的应用的处理。具体而言，本发明提供一种用于将本地代码库及其相关性更新和安装到适当位置以由以诸如 Java、REXX、BASIC、SmallTalk、Python 和 Perl 等解释编程语言编写的应用使用的方法、程序和系统。

背景技术

以解释编程语言编写的程序可以由编程虚拟机执行。在解释语言中，并非生成本地机器代码，而是由编译器（complier）生成将由编程虚拟机使用的字节代码。这些字节代码提供执行应用必需的控制和数据。随后，为了实际地执行解释编程语言应用，解释器解释由编译器生成的已编译字节代码。理论上，可以解释任何编程语言；然而，术语“解释编程语言”通常表示通过由解释器执行而实施的语言。术语“解释编程语言”还表示不为其编写编译器的语言。

公知的解释编程语言是 Java，在此将其用作优选的例子。然而，对于本领域的技术人员来说，应当很容易理解诸如 REXX、BASIC、SmallTalk、Python、Perl 等其它解释编程语言在本发明的保护范围之内。

Java 是最初由 Sun Microsystems 开发的旨在生成可在没有修改的情况下在所有硬件平台上运行的应用的软件编程语言。因为这个原因，Java 应用已在万维网（WWW）上广泛使用。可以在超文本置标语言（HTML）文档内调用或单独启动 Java 应用。

如本领域的技术人员公知的，WWW 服务器可以包括用于存储和发送应用程序的设施，所述应用例如是用 Java 编写以在远程计算机上

执行的应用程序。

Java 虚拟机（JVM）是仅驻留在远程计算机上的存储器内的虚拟计算机组件。对于将要在不同类型的数据处理系统上执行的 Java 应用而言，编译器通常生成体系结构中性文件格式，以便所编译的代码可以在任一处理器上执行，只要该 JVM 可以由处理器访问。该体系结构中性文件格式包括由 Java 编译器生成和并非专用于给定计算机体系结构的字节代码指令。字节代码是由 Java 编译器生成和由 Java 解释器执行的机器无关代码。作为 JVM 一部分的 Java 解释器可选择地解码和解释称作“字节代码”的中间代码。这些字节代码指令被设计为容易地在任一平台上解释和容易地转换成本地机器代码。

Java Web Start（JWS）提供一种运行 Java 应用而不在远程计算机上安装除 JVM 和 JWS 之外的任何东西的机制。通常，只要在远程计算机上启动 Java 应用，JWS 就简单地从服务器下载 Java 应用 jar 文件。这利用了 Java 代码的机器独立性，以便当在远程计算机上下一次启动 Java 应用时，容易地更新该 Java 应用。

但是，Java 应用通常使用本地代码，例如，以 C 或 C++编程语言编写的对于特定平台或操作系统专用的代码。Java 应用使用 Java 本地接口（JNI）访问由这些本地方法提供的本地代码。通常，这些本地方法被作为动态加载的库（或 DLL）提供给 JNI。本地方法 DLL 通常是不可跨不同操作系统移动的。

当 Java 应用使用 JNI 并具有对依赖于其他 DLL 的 DLL 的相关性时，JWS 会在启动（launch）Java 应用时遇到一些局限性。例如，如果 Java 应用具有带有与其他库的相关性的任何 JNI 代码，则如果所述库不在 Java 运行时环境（JRE）的路径中，应用将不会开始。这适用于任何主 Java 应用以及 JWS 本身（例如，如果存在任何带有与其他库的相关性的 JNI 代码，则 JWS 也不开始）。仅当库不具有任何其他相关性或者相关性已经安装在 JVM 可以定位到它们的位置时，JWS 才可以加载。

即使其在高速缓存中可用，JWS 也不允许从 webstart 高速缓存

加载 JNI 库。例如，主 Java 应用具有在库 a.dll 中的 JNI 代码，库 a.dll 具有对库 b.dll 和 c.dll 的相关性。即使库 b.dll 和 c.dll 在与 a.dll 相同的 jar 文件中，JWS 也无法加载 b.dll 和 c.dll。为了使 JWS 工作，需要将库 b.dll 和 c.dll 安装在已经在 JRE 路径中的现有目录中。

在这种情况下，Java Web Start 启动应用，但是应用不能开始。为了避免这种情况，必须在远程计算机上运行诸如 InstallShield 之类的外部安装器程序。然后，远程计算机与服务器相连，并接收相关的库以用于安装。接下来，Java 应用重新开始。如果在应用下次启动时有对库的更新，那么还必须手动安装它们。

因此，将希望提供一种安装和更新并不需要外部安装器的本地代码库及其相关性的机制。还将希望提供一种安装和更新本地代码库及其相关性以用于以解释编程语言编写的应用的机制。

更具体地，将希望提供一种安装和更新并不需要外部安装器的 JNI 库及其相关性的机制。还将希望提供一种使用 JWS 在 JRE 路径内安装相关性的机制以及一种用于 JNI 容易地更新其相关性的机制。

本发明提供一种用于从服务器动态地安装和更新这种本地代码库及其相关性以克服上述目标的方法和系统。

发明内容

本发明的方法从服务器将本地（native）代码库提供给解释编程语言应用以在远程计算机上执行。在解释编程语言应用的主类内调用安装器对象。从包括指定本地代码库的至少一个库文件名的服务器提取文件列表。将用库文件名指示的本地代码库安装到在远程计算机上的目标二进制目录（bin directory），随后在远程计算机上调用该解释编程语言应用的主类。还将附加本地代码库相关性（dependency）安装到远程计算机上的目标二进制目录。该方法从服务器提取最新版本的文件，在远程计算机上的目标二进制目录内定位当前版本文件，并比较当前版本文件和最新版本文件内的版本编号以确定将哪些文件安装在远程计算机上。该方法还使用最新版本文件重写（overwrite）

在目标二进制目录内的当前版本文件，以及用本地代码库文件及其相关性的最新版本重写在远程计算机上存在的较旧的本地代码库文件。

根据下文对当前优选实施例的详细描述，结合附图阅读，本发明的上述和其它特征和优点将变得更加显而易见。详细描述和附图仅是说明本发明，而不是限制，本发明的范围由权利要求书及其等同范围定义。

附图说明

图 1 是可以实施本发明的数据处理系统网络的方框图；

图 2 是根据本发明优选实施例的可以实施为服务器的数据处理系统的方框图；

图 3 是图示在其中可以实施本发明的数据处理系统的方框图；和

图 4 图示根据当前发明的一种实施例的动态地将本地代码库提供给解释编程语言应用的方法。

具体实施方式

本发明实施一种数据处理系统和方法，用于管理在使用诸如 Java、REXX、BASIC、SmallTalk、Python、Perl 等解释编程语言开发的应用内所使用的本地代码库及其相关性的安装。在这种语言内，将指令编译成由虚拟机使用的字节代码，但是某些本地代码（尤其是库及其相关性）也被使用。

在下文的描述中，阐述许多具体细节以提供对本发明的完整理解。然而，对于本领域的技术人员来说，在没有这些具体细节的情况下也可以实施本发明将是显而易见的。在其它的例子中，以方框图的形式图示公知的设备以便不因为不必要的细节影响本发明。

图 1 是实施本发明的分布式数据处理系统的一种实施例的附图。分布式数据处理系统 100 包括网络 102，它是用于在分布式数据处理系统 100 内连接在一起的各种设备和计算机之间提供通信链路的介质。网络 102 可以包括诸如有线或光纤电缆的永久连接或通过电话连

接实现的临时连接。包含本发明的各种处理可以驻留在同一主机上或者在网络 102 上互连的不同机器上。（例如因特网、内部网、广域网（WAN）或局域网（LAN））。因而，受益于本发明的机器具有适当的连网硬件以建立到一台或多台其它机器（服务器和/或远程计算机）的连接。例如，连接到网络 102 的机器可以具有到网络的 TCP/IP 或 NETBIOS 连接，所述网络运行在令牌环或以太网适配器上。

服务器 104 连接到如上所述的网络 102，以及存储单元 106。服务器 104 向远程计算机 108、110、112 提供数据，例如 Java 应用、JNI、库和其它程序。这些数据可以存储在存储单元 106 上并由服务器 104 访问。

在本发明的图示实施例中，中央服务器 104 可以使用基于 Web 的软件系统，例如 Sun 微系统公司的 Java Web Start，它支持中央服务器 104 通过 WWW 连接分配和更新合适的 Java 应用、JNI 库及其相关性。

远程计算机 108、110 和 112 还连接到网络 102。远程计算机 108、110 和 112 例如可以是个人计算机或网络计算机，具有多种处理器和操作系统。

分布式数据处理系统 100 可以包括附加服务器、客户机和未图示的其它设备。图 1 将作为例子，而不作为对本发明处理的结构限制。

图 2 是可以实施为服务器，例如图 1 中的服务器 104 的数据处理系统的方框图。数据处理系统 200 可以是包括连接到系统总线 206 的多个处理器 202 和 204 的对称多处理器（SMP）系统。可选择地，可以使用单处理器系统。还连接到系统总线 206 的是存储器控制器/超高速缓存 208，它提供到本地存储器 209 的接口。I/O 总线桥 210 连接到系统总线 206，并提供到 I/O 总线 212 的接口。可以如图所示集成存储器控制器/超高速缓存 208 和 I/O 总线桥 210。

连接到 I/O 总线 212 的外围组件互连（PCI）总线桥 214 提供到 PCI 本地总线 216 的接口。多个调制解调器可以连接到 PCI 总线 216。典型的 PCI 总线实施方式将支持四个 PCI 扩展槽或内插连接器。可以

通过经内插板连接到 PCI 本地总线 216 的调制解调器 218 和网络适配器 220 提供图 1 中到网络计算机 108-112 的通信连接。

附加 PCI 总线桥 222 和 224 提供用于附加 PCI 总线 226 和 228 的接口，由此可以支持附加调制解调器或网络适配器。以这种方式，数据处理系统 200 允许到多个网络计算机的连接。存储器映射图形适配器 230 和硬盘 232 还可以直接地或间接地连接到如图所示的 I/O 总线 212。

本领域的普通技术人员将理解图 2 所示的硬件可以改变。例如，也可以附加地或者替代所图示的硬件使用其它外围设备，例如光盘驱动等。图 2 所示的数据处理系统例如可以是 IBM eServer pSeries，在纽约 Armonk 的国际商业机器公司的产品，运行 Active Interactive Executive（AIX）操作系统。图 2 并不意味着对本发明的体系结构限制。

在本发明的一种实施例中，在服务器 104 上将本地代码库和列出这些本地代码库的文件名的文件打包成 jar 文件，例如称作“install.jar”。包含版本号的另一个文件也打包在服务器 104 上的同一 jar 文件中。在远程计算机 108、110、112 上调用的安装器对象从服务器 104 提取包含本地代码库列表的文件。这些相同文件的版本在 install.jar 文件内。因此，install.jar 文件包含下述内容:

例如称作“version.txt”的文件，它提供 jar 文件的版本号;

例如称作“filelist.txt”的文件，它提供运行主要的解释编程语言应用需要的文件列表; 和

在“filelist.txt”内列出的所有文件。

这个安装器对象查阅名称列表，并将来自“install.jar”的每个文件提取到远程计算机上的所需二进制目录。

图 3 是可以实施为诸如计算机 108、110、112 等远程计算机的数据处理系统的方框图。可以在这样一个远程计算机上或者在通过计算机网络连接的一个或多个这样的计算机上执行更新或安装本地代码库及其相关性的一个或多个处理。在说明性的实施例中，在这样一个远

程计算机上，或者在通过计算机网络连接的一个或多个计算机上执行将由 Java 应用使用的 Java 本地接口库及其相关性的更新或安装。

除了其它组件之外，图 3 的数据处理系统 300 包括处理器 302、主存储器 304、操作系统 314、虚拟机解释器 316、web 启动应用 318 和一个或多个主应用程序 320 以及至少一个安装器对象 322。

操作系统 314 在处理器 302 上运行，并用于协调和提供在图 3 内的数据处理器系统 300 内各个组件的控制。操作系统可以是商业可用的操作系统，例如 Windows XP，它可以从微软公司购买。一个或多个编程系统可以与操作系统 314 共同运行，所述操作系统 314 例如是解释编程语言，包括但并不限制于 Java、REXX、BASIC、SmallTalk、Python、Perl、等等。诸如 Java 等面向对象的编程系统可以与操作系统共同运行，并提供从在数据处理系统 300 上执行的 Java 程序或应用对操作系统的调用，（“Java”是 Sun 微系统公司的商标）。操作系统的指令、面向对象编程系统和应用或程序位于诸如硬盘驱动器的存储设备上，并可以加载到主存储器 304 内以由处理器 302 执行。

虚拟机（VM）316 是抽象计算机器，包括指令集合并使用存储器 304 内的各个存储区域。将 VM 316 加载或存储在诸如主存储器 304 内以由处理器 302 执行。VM 316 是能够解释一个或多个解释编程语言以向操作系统 314 提供所需程序的数据或控制的任何适当虚拟机。

在说明性的实施例中，Sun 微系统 Java 2 平台，标准版本（J2SE）仿真在各种平台上的 VM。具体关于 Java 虚拟机的其它细节可以在 Java™ Virtual Machine Specification，Tim Lindholm 和 Frank Yellin，Addison Wesley（1997），ISBN 0-201-63452-X 中获得，其在此引用作为参考。也可以使用任何适当的 Java 虚拟机解释器。

Web 启动应用（WSA）318 是基于 Web 的软件系统，它支持服务器 104 通过如在图 4 内进一步描述的与远程计算机 108、110、112 的 WWW 连接分配和更新期望的解释编程语言应用、本地代码库和相关性。将 WSA 318 例如加载或存储在主存储器 304 内以由处理器 302 执行。使用 WSA 318 在远程计算机 108、110 和 112 上启动解释编程

语言应用。典型地，在专用平台或运行时环境内启动这些应用，所述平台或环境包含核心可执行、核心文件、支持文件和启动期望应用需要的任何其它数据。

在说明性的实施例中，WSA 318 是 Java Web Start，它是基于 Web 的软件系统，支持服务器 104 在一个或多个远程计算机上启动期望的 Java 应用，安装和更新 JNI 库及其相关性和通过 WWW 连接将执行 Java 应用需要的其它数据提供给远程计算机。例如，WSA 318 更新应用 jar 文件，或者根据本发明，将所需要的本地代码接口放置在目标目录内。WSA 318 是 VM 316 的一部分或者与之通信。

因而，在这个说明性的例子中，Java 运行时环境（JRE）包括 Java Web Start 应用、Java 虚拟机、核心可执行、核心文件和支持文件以建立标准 Java 平台。因此，JRE 通常在 Java Web Start 运行之前已经存在于远程计算机 108、110 和 112 上。JWS 318 用于更新应用 jar 文件。例如，当用户第一次访问 Java 应用时，JWS 软件将下载该应用需要的所有文件，并且如果所请求的版本本地不可用，则还下载 JRE。

主应用 320 是基于 Web 的软件系统，它支持服务器 104 通过 WWW 连接在远程计算机 108、110 和 112 上执行期望的应用功能和处理与这些应用功能相关的数据。典型地，在专用平台或运行时环境内启动主应用 320，所述平台或环境可以访问可执行、支持文件和运行期望应用 320 需要的任何其它数据。尽管图 3 仅图示一个主应用，但是本发明的方法适合于与多个期望的应用一起使用。主应用 320 是 VM 316 的一部分或与之通信，并可以由 WSA 318 启动。在说明性的实施例中，主应用 320 是任一期望 Java 应用，其中的许多应用在本技术领域中是公知的。

安装器对象 322 是用于执行如图 4 所示的本发明步骤的变量和相关方法的软件集合（bundle）。由远程计算机 108、110 和 112 或者更准确地由主应用 320 调用安装器对象。安装器对象例如能够确定某些文件的存在与否以确定在远程计算机上的这些文件的版本，确定它从

服务器下载的文件版本，并且如果它下载的版本较新或者某些文件根本不在远程计算机上，则重写这些文件。

例如安装器对象包括试图在远程计算机上的下载文件列表内发现 install.jar 文件的计算机程序代码。安装器对象还包括支持从 install.jar 文件提取特定本地代码库的计算机程序代码。安装器对象还包括能够将所提取的库放置在目标二进制目录内的计算机程序代码。安装器对象还包括从服务器调用本地代码库和将它们加载到远程计算机上的计算机程序代码。通常与用于主应用的其它 jar 文件一起打包安装器对象，并可以在应用自身的主类内调用该安装器对象。

在说明性的实施例中，Java Web Start 连接到用远程计算机 108、110、112 表示的服务器 104 上的 URL。虽然 JWS 运行主应用类 320，但是安装器对象 322 还由主应用类 320 调用，并且安装器对象如下文所述执行本发明的方法。

本领域的普通技术人员将理解图 3 中的硬件可以根据实施方式改变。附加地或者替代图 3 所示的硬件，也可以使用其它的内部硬件或外围设备，例如快速 ROM（或等同的非易失性存储器）或者光盘驱动器等。而且，可以将本发明的处理应用于多处理器数据处理系统。

因而，例如，远程计算机 108、110、112 是任一个人计算机或工作站平台，基于英特尔、PowerPC 或 RISC，并包括诸如 IBM OS/2、微软 Windows XP、微软 Windows NT 4.0、Unix、AIX 5SL 等操作系统。典型的计算机运行英特尔 x86 处理器、OS/2 Warp 第 3 版操作系统、JVM 版本 1.1.1 和 Java Web Start 1.04。可选择地，计算机运行 x86 处理器、Windows XP（或 Windows NT）操作系统，JVM 的版本是 1.4 和以上。通常，根据与其一起打包的 JVM 的版本，JWS 的版本也是 1.4 及以上。

图 3 并不是体系结构上的限制。例如，数据处理系统 300 还可以是笔记本计算机、手持式计算机或个人数字助理。数据处理系统 300 还可以是公用电话亭或 Web 设备。

图 4 是根据本发明动态地将本地代码库及其相关性提供给解释编

程语言应用的方法 400 的流程图。可由计算机可使用或计算机可读介质可访问的计算机程序产品实现本发明的方法，所述介质提供由计算机或任一指令执行系统使用或结合使用的程序代码，例如 Java Web Start。

本地代码库通常是由用本地代码（通常 C 或 C++）编写的解释编程语言应用使用的文件。这些本地代码库专用于特定平台，而不是独立于体系结构的。通常，动态地加载这些库，意味着不是在应用启动的过程中加载它们。也就是，直到应用需要时才加载它们。此外，这些库通常取决于其它库。也就是，可以加载第一库（例如 a.dll），随后，由于其被加载，可能需要一个或多个其它的库（例如 b.dll 和/或 c.dll 等）。

DLL 对于实施插件或模块特别有用，因为直到需要时才加载它们。这些库也可用于多种用途。例如，可插件鉴权模块（PAM）系统使用 DL 库以允许管理者配置和重新配置鉴权。DLL 可用于实施解释器，以有时将代码编译成机器代码，例如在实施刚好及时（just-in-time）（JIT）编译器或多用户地牢游戏（multi-user dungeon: MUD）时。

也可以使用包括 Linux、Solaris、AWT 和上述 Java 本地接口（JNI）的多种接口访问 DLL。尽管 JNI 用作说明性的例子，但本发明的方法也可以由任一等价接口使用以将任一期望本地代码库、任一动态加载库和任一期望相关性提供给给定解释编程语言应用。

如在方框 402 中看到的，当 web 启动应用 318 试图在远程计算机 108、110 和 112 上启动解释的编程语言时，方法 400 开始。通常，主应用的启动涉及远程计算机 108、110 和 112 连接到服务器 104。这例如在如上所述的典型网络连接上实现。在说明性的实施例中，Java Web Start 试图通过从远程计算机连接到服务器 104 启动主 Java 应用。

如在方框 403 中看到的，由远程计算机调用安装器对象。安装器对象 322 通常由通过 WSA 318 启动的主应用 320 调用。安装器对象 322 也可以由 WSA 318 调用。即使 filelist.txt 和 version.txt 文件不存在，也可以调用安装器对象。在这种情况下，它将不安装任何文件。

在说明性的实施例中，安装器对象由通过 Java Web Start（JWS）启动的 Java 应用调用。JWS 在远程计算机上启动。随后，JWS 从服务器获取在远程计算机上运行期望 Java 应用需要的所有 jar 文件。随后，JWS 在远程计算机上启动期望的 Java 应用。JWS 或主 Java 应用随后调用安装器对象。一旦 JWS 在远程计算机上运行，则它会下载需要从服务器下载的所有文件。典型地，JWS 需要下载以启动 Java 应用的文件由服务器指定。

在本发明中，所有的 jar 文件已经在远程计算机上。所以，安装器对象开始于通过试图在远程计算机上的下载文件列表内发现 install.jar 文件。如果安装器对象发现“install.jar”，则它将查阅其从 install.jar 文件提取 JNI 库和将它们放置在指定 JRE 二进制目录内的逻辑。安装器对象将从服务器下载 JNI 库，或者搜索在远程计算机上的现有库，并确保将它们放置在适当的 JRE 二进制目录内。随后，安装器从服务器调用 JNI 库，并将其加载到远程计算机上。在调用 JNI 代码以由主 Java 应用使用之前调用这个安装器类对象。将该安装类与用于 Java 主应用的其它 jar 文件一起打包，并可以从该应用自身的主类内调用。

以本技术领域内公知的方式，远程计算机使用在 Java 运行时环境内的 Java Web Start 调用 Java 安装器对象。Java 是致力于将数据定义为对象和可应用于这些对象的方法的面向对象的编程语言和环境。Java 安装器对象是根据本发明的软件组件，它已经被编译以动态地将 JNI 库及其相关性提供给主 Java 应用。Java 安装器对象例如能够确定某些文件的存在与否，确定在远程计算机上的这些文件的版本，确定它从服务器下载的文件的版本，并且如果它下载的版本较新或者某些文件在远程计算机上根本不存在则重写这些文件。

例如使用下述内容调用安装器对象：

```
public static void main(String [] args) {
Installer installer = new Installer ();
Installer.install();
```

Run the application ...

}

如在方框 404 中看到的，远程计算机确定“install.jar”是否存在在远程计算机上。“install.jar”（jar 文件）包含运行主应用必需的所有文件，包括实际的本地代码库以及列出本地代码库文件名的文件。包含版本号的另一个文件也打包在同一 jar 文件内。安装器对象提取包含本地代码库列表的文件。它搜索名称列表，并将每个文件提取到目标二进制目录。因而，“Install.jar”包含本地代码库和它的相关性。“Install.jar”还包含列出本地代码库和所有的相关文件的文件名的文件。将这个文件称作例如“install.txt”或“filelist.txt”。通常，这个列表每行包含一个文件的名称。“Install.jar”还包含具有版本串的文件。将这个文件称作例如“install.txt”或“version.txt”。还将该安装器类打包在同一 jar 文件内。

在说明性的实施例中，“Install.jar”（jar 文件）包含运行主 Java 应用的所有必须文件，包括实际的 JNI 库以及列出 JNI 库的名称的文件。包含版本编号的另一个文件也打包在同一 jar 文件内。安装器对象提取包含 JNI 库列表的文件。它查阅名称列表和将每个文件提取到指定的 JRE 二进制目录内。因而，“Install.jar”包含 JNI 代码及其相关性。“Install.jar”还包含 JNI 库文件列表及其所有相关性。安装器对象也打包在同一 jar 文件内。

如果“install.jar”不存在，则所有的必需文件已经可用于在远程计算机上的主应用。该方法前进到方框 430，其中安装器对象已经完成其工作，并由此到方框 440，其中现在启动主应用。现在，这个初始调用安装器对象的主应用继续执行其主要任务。在已经适当地修改这些库之后，这个主应用可以执行任一适当的功能，其中若干功能在本技术领域中是公知的。

在说明性的实施例中，这个主应用是 Java 应用，其主要任务是本技术领域中公知的任一适当的 Java 功能。

如在方框 406 中看到的，远程计算机从 jar 文件提取“filelist.txt”。

该文件“filelist.txt”是一个或多个本地代码库的文件名及其相关性的列表。

例如，“filelist.txt”包含文件列表，例如:

a.dll

b.dll

c.dll

其中 a.dll 是 JNI 库，b.dll 和 c.dll 是它的相关性。

在说明性的实施例中，“filelist.txt”列出用于特定 Java 应用的 JNI 库文件以及与这些库文件的所有相关性。

如在方框 408 中看到的，远程计算机从 jar 文件提取“version.txt”。文件“version.txt”包含单个号码，表示将要安装的文件的版本。每次修改本地代码库时递增文件“version.txt”。因而，“version.txt”一开始包含例如编号 1.1。在修改本地代码库之后，随后“version.txt”将包含编号 1.2。当再次在服务器上打包 install.jar 文件时，随着任一次改变本地代码库递增在“version.txt”内的版本编号。在说明性的实施例中，无论何时改变 JNI 库，均递增在“version.txt”内的版本编号。

如在方框 410 中看到的，远程计算机确定“version.txt”是否存在于其中解释编程语言应用将运行的运行时环境的目标目录中。在说明性的实施例中，目标目录是 JRE 二进制目录，它通常是在安装 JVM 的远程计算机文件系统上的位置。如果在目标目录内不存在“version.txt”，则尚未在远程计算机上安装所需要的动态加载本地代码库及其相关性。如果是这种情况，则该方法返回方框 404，并调用安装器对象以在远程计算机上安装所有适当的文件。如果存在“version.txt”，则该方法可以前进到方框 412。

如在方框 412 中看到的，远程计算机比较在目标二进制目录内在方框 410 上发现的文件“version.txt”的版本号与在方框 408 上提取的文件“version.txt”。在本发明的一种实施例中，这个版本号比较是在两个“version.txt”文件内包含的两个编号的字符串比较。

如在方框 414 中看到的，确定在方框 408 上提取的版本号是否大于在方框 410 上发现的编号。如果在方框 408 上提取的版本号不大于在方框 410 上发现的编号，则该方法前进到方框 430，其中已经完成安装器对象。

如果在方框 408 上提取的版本号的确大于在方框 410 上发现的编号，则该方法前进到方框 420，其中将在“filelist.txt”内列出的所有适当文件提取到目标目录。此时，该方法还包括加载表示目标目录的文件，通常表示主应用的缺省路径。根据本发明，该缺省路径指向本地代码库和任一独立库。在说明性的实施例中，将在“filelist.txt”内列出的第一个文件提取到 JRE 二进制目录。此外，主 Java 应用的缺省路径表示 JRE 二进制目录。这样，将在“filelist.txt”内列出的 JNI 库及其相关性移动到将由 Java Web Start 和因而 Java 主应用使用的合适路径。也就是，通过将 JNI 库写入在 JRE 的二进制目录内，该方法将 JNI 库放置在需要这些库的应用的“路径”内。

如在方框 422 中看到的，如果在目标目录内存在本地代码库的旧版本，则重写这些文件。在说明性的例子中，如果在 JRE 二进制目录内存在 JNI 库文件的旧版本，则重写这些文件。例如，如果是 Java 应用第一次在远程计算机上运行，则将提取在“filelist.txt”内列出的每个文件。如果并非第一次，则将仅重写在 jar 文件内具有较新版本的旧 JNI 库。

如在方框 423 中看到的，确定在“filelist.txt”内是否存在另一个（下一个）文件。如果列出这样一个文件，则它通常在 “filelist.txt” 前一文件之后的下一行上。如果列出另一个文件，则该方法返回 420。如果不存在列出的文件，则该方法前进到 424。

如在方框 424 中看到的，一旦已经将“filelist.txt”内的所有文件写入到目标目录，则使用来自 jar 文件的最新“version.txt”重写在目标目录内的文件“version.txt”。在说明性的实施例中，使用来自 JAR 文件的最新“version.txt”重写在 JRE 二进制目录内的“version.txt”。因而，JRE 二进制目标现在将具有表示在其目录内文件的最新版本的

“version.txt”。

该方法前进到方框 430，其中安装器对象已经完成其工作。

该方法随后前进到方框 440，其中主应用现在完全启动，并可以在没有中断的情况下运行其任务。在说明性的实施例中，启动 Java 应用（Java Web Start 或另一个 Java 应用）。一开始调用安装器对象的这个主 Java 应用现在继续执行其主要任务。在已经适当地修改库之后，这个主 Java 应用可以执行任一适当的 Java 功能，其中若干功能在现有技术中是公知的。

本发明可以采取完全硬件实施例、完全软件实施例或同时包含硬件和软件单元的实施例的形式。在优选实施例中，在软件内实现本发明，所述软件包括但是并不限制于固件、驻留软件或微代码等。此外，本发明可以采取可以从计算机可用或计算机可读介质访问的计算机程序产品的形式，所述介质提供由计算机或任一指令执行系统使用或结合使用的程序代码。为了这个说明书的目的，计算机可用或计算机可读介质可以是可包含、存储、传送、传播或传输由指令执行系统、仪器或设备使用或结合使用的程序的任一个设备。该介质可以是电子、磁、光、电磁、红外或半导体系统（或仪器或设备）或诸如载波的传输介质。计算机可读介质的例子包括半导体或固态存储器、磁带、可拆除计算机磁盘、随机访问存储器（RAM）、制度存储器（ROM）、刚性磁盘或光盘。

虽然当前将在此公开的本发明的实施例视为优选的，但是在不脱离本发明的精神和范围的情况下，可以对其进行各种改变和修改。本发明的范围将在权利要求书内表示，落入其含义和等价范围内的所有改变将包含在其中。

图1

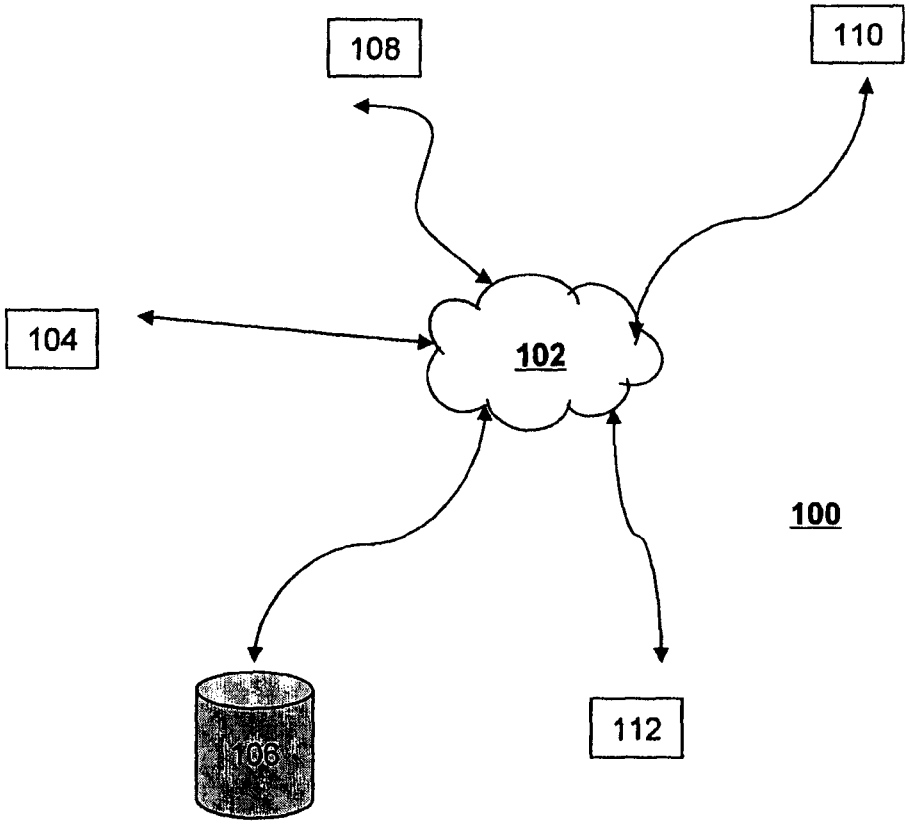


图 2

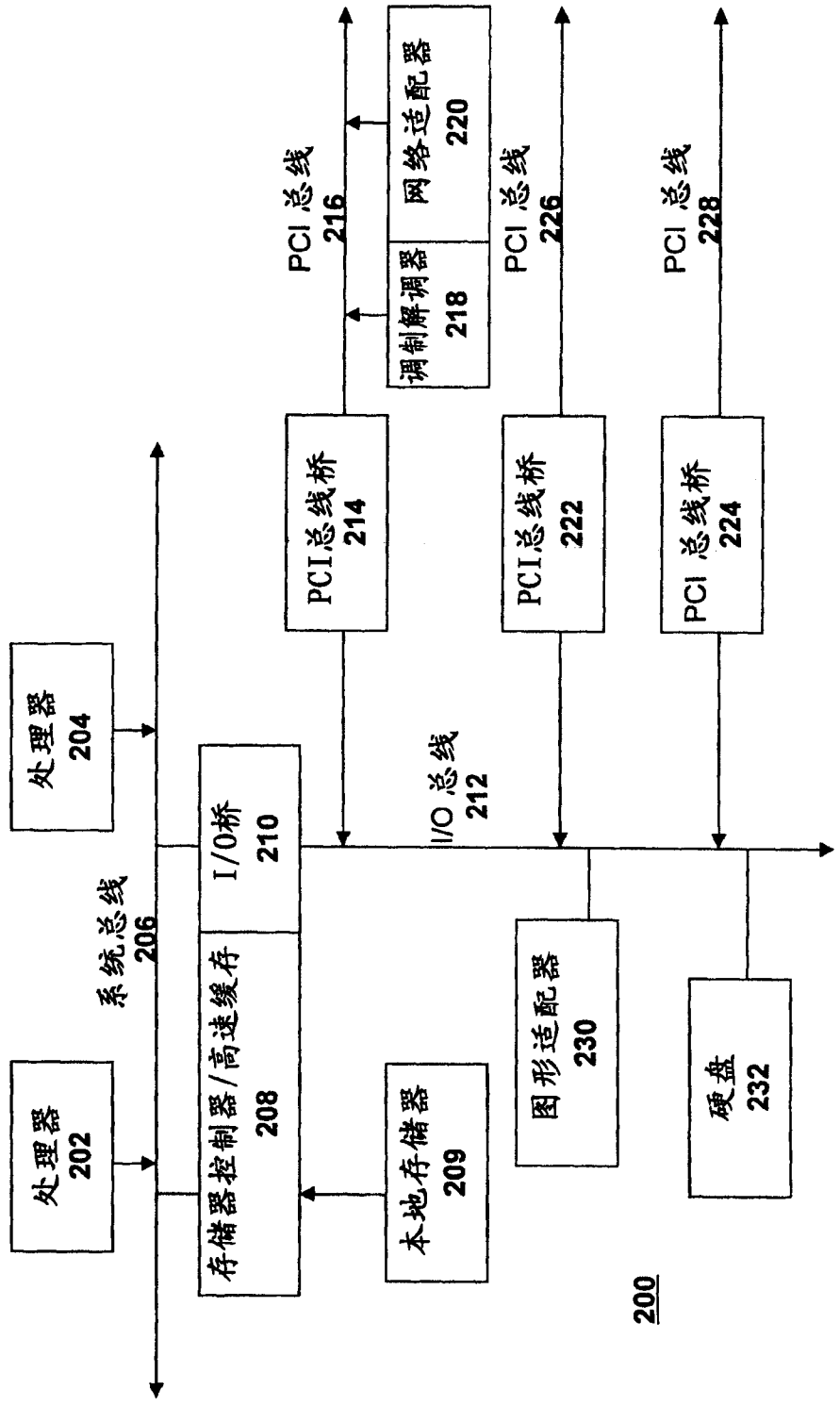

处理器
202
处理器
204
系统总线
206
存储器控制器/高速缓存
208
I/O桥
210
本地存储器
209
I/O 总线
212
PCI总线桥
214
PCI总线桥
222
PCI 总线桥
224
PCI 总线
216
PCI 总线
226
PCI 总线
228
调制解调器
218
网络适配器
220
图形适配器
230
硬盘
232
200

图3

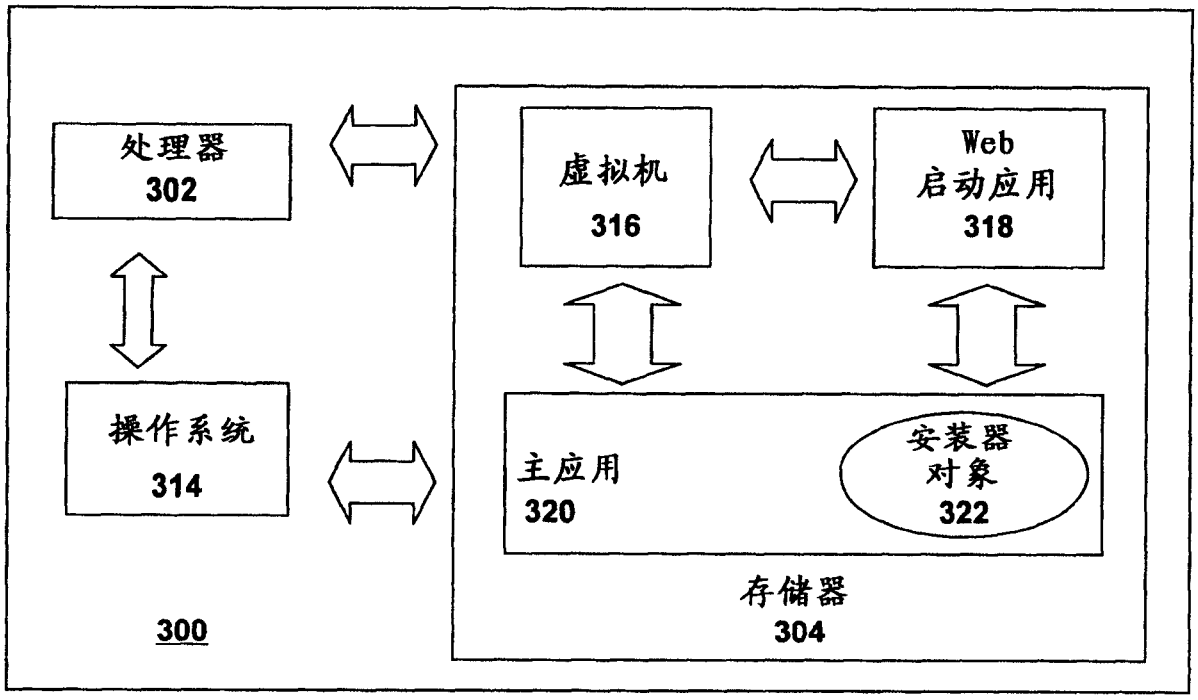
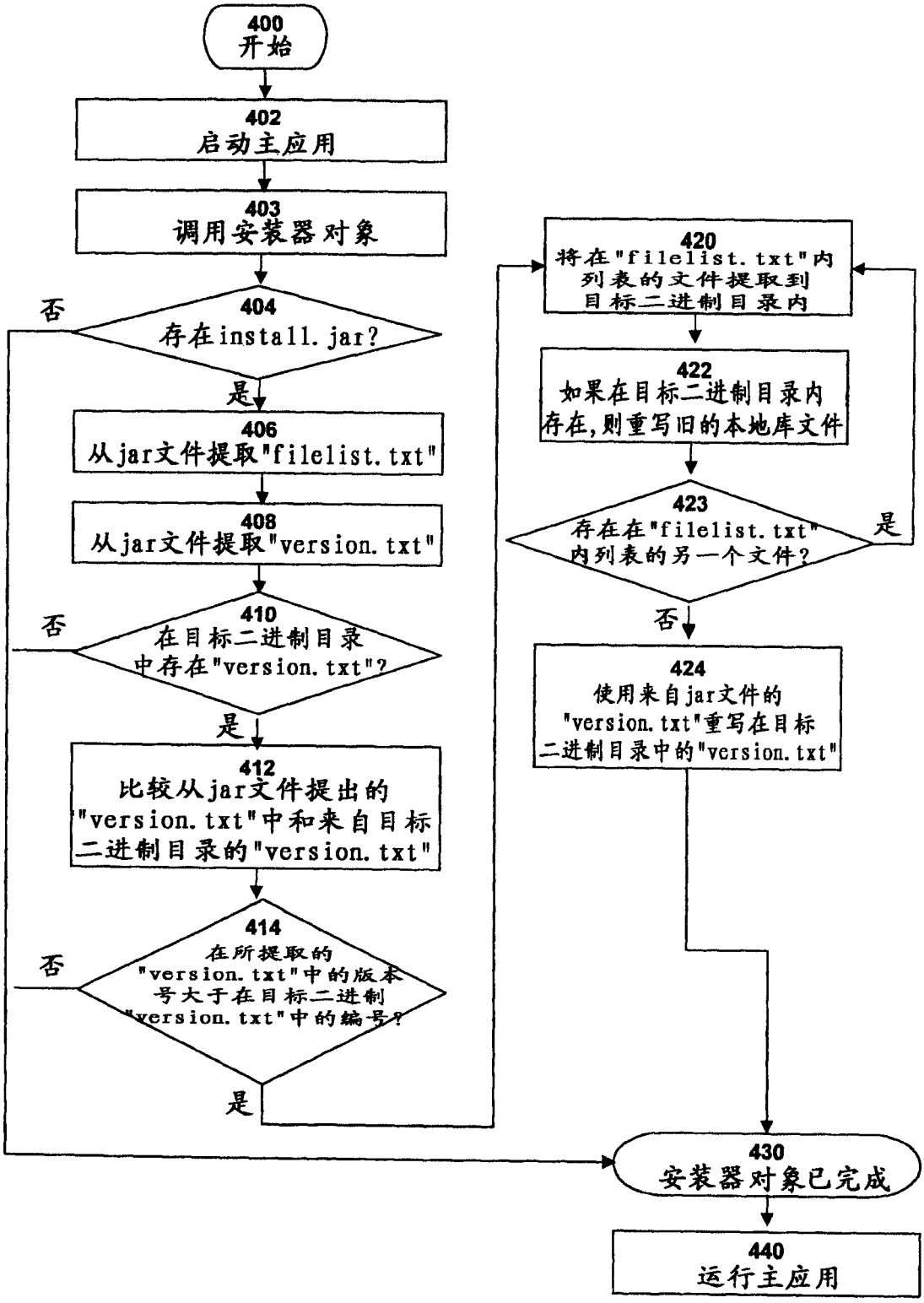

处理器
302
操作系统
314
300
虚拟机
316
Web
启动应用
318
主应用
320
安装器
对象
322
存储器
304

图4


400
开始
402
启动主应用
403
调用安装器对象
404
存在install.jar?
否
是
406
从jar文件提取"filelist.txt"
408
从jar文件提取"version.txt"
410
在目标二进制目录中存在"version.txt"?
否
是
412
比较从jar文件提出的"version.txt"中和来自目标二进制目录的"version.txt"
414
在所提取的"version.txt"中的版本号大于在目标二进制"version.txt"中的编号?
否
是
420
将在"filelist.txt"内列表的文件提取到目标二进制目录内
422
如果在目标二进制目录内存在,则重写旧的本地库文件
423
存在在"filelist.txt"内列表的另一个文件?
是
否
424
使用来自jar文件的"version.txt"重写在目标二进制目录中的"version.txt"
430
安装器对象已完成
440
运行主应用