(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 102854819 B
(45)授权公告日 2016.03.09

(21)申请号 201210297339.3

(22)申请日 2005.05.04

(30)优先权数据
60/567,980 2004.05.04 US

(62)分案原申请数据
200580014528.3 2005.05.04

(73)专利权人 费舍－柔斯芒特系统股份有限公司
地址 美国德克萨斯州

(72)发明人 肯·J·贝欧格特
斯蒂芬·吉尔伯特
马克·J·尼克松
迈克尔·J·卢卡斯

(74)专利代理机构 北京德琦知识产权代理有限公司 11018
代理人 康泉 宋志强

(51)Int.Cl.
G05B 19/042(2006.01)
G05B 19/418(2006.01)
G06F 9/44(2006.01)
H04L 29/08(2006.01)

(56)对比文件
US 2003/0191803 A1,2003.10.09,

审查员 贾奇峰

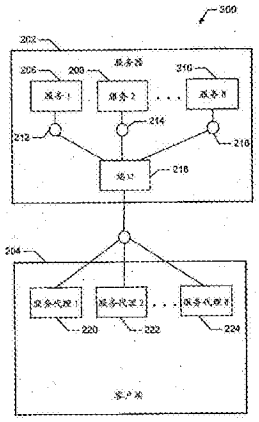
权利要求书1页 说明书15页 附图16页

(54)发明名称

用于过程控制系统的面向服务的架构

(57)摘要

本发明公开了一种用于过程控制系统的面向服务的架构。在一个示例中，用于在过程控制系统中的客户端过程和服务器过程之间传递过程控制信息的方法，建立包含多个过程控制服务的服务器过程，所述多个过程控制服务中的每个过程控制服务均有相应的服务界面。该示例性方法还建立具有用于多个服务中的每个服务的代理的客户端过程，该客户端过程与所述服务建立通信连接。另外，该示例性方法还向客户端过程提供与服务界面相关联的端口信息，使得客户端过程和服务器过程之间能够进行过程控制信息的传递。



CN 102854819 B

1. 一种在过程控制系统中注册过程控制服务的方法，包括：

在所述过程控制系统的发现服务器中建立一发现服务；

在所述过程控制系统的运行期服务器中建立一运行期服务；

使用所述运行期服务器的所述运行期服务产生发现代理，其中所述发现代理包括一端口号和一标识符占位符；

使用所述运行期服务将所述端口号通过所述发现代理传送到所述发现服务；

在将所述端口号传送到所述发现服务之后，使用所述运行期服务产生与所述过程控制服务相关联的标识符；和

通过所述发现代理向所述发现服务传送来自所述运行期服务器的所述标识符，在所述过程控制系统中注册所述过程控制服务。

2. 如权利要求 1 所述的方法，其中，所述过程控制服务包括会话服务、运行期服务、数据库服务、版本服务、历史服务、报警和事件服务或 OPC 数据服务中的至少一个。

3. 如权利要求 1 所述的方法，其中，所述标识符包括统一资源标识符。

用于过程控制系统的面向服务的架构

[0001] 本申请是申请日为2005年5月4日、申请号为200580014528.3(PCT/US2005/015394)且名称为“用于过程控制系统的面向服务的架构”的发明的分案申请。

[0002] 相关申请

[0003] 本申请为美国临时申请No.60/567,980的正规递交申请,并要求该申请的优先权,该申请名称为“用于再现、监控过程控制系统并与之交互的图形用户界面(Graphical User Interface for Representing,Monitoring,and Interacting with Process Control Systems)”,提交于2004年5月4日,特意将其全部内容按引用并入本文作为参考。本申请的相关申请还有,序列号为10/625,481的美国专利申请,名称为“图形显示元件、过程模块和控制模块在加工厂中的集成(Integration of Graphic Display Elements,Process Modules and Control Modules in Process Plants)”,提交于2003年7月21日,并于2004年8月5日公开,美国公开号为2004/0153804;接下来还有,序列号为10/278,469的美国专利申请的部分继续申请,该申请的名称为“加工厂中的智能过程模块和对象(Smart Process Modules and Objects in Process Plants)”,提交于2002年10月22日,并公开于2004年4月22日,美国公开号为2004/0075689,上述申请全部内容在此引用作为参考。本申请还与下面申请相关:序列号为10/368,151的美国专利申请,名称为“加工厂配置系统中的模块类对象(Module Class Objects in a Process Plant Configuration System)”,提交于2003年2月18日,并公开于2004年10月7日,美国公开号为2004/0199925,该申请全部内容在此引用作为参考。本发明还与下列专利申请相关,他们均在同一天被作为本申请的国际(PCT)申请提交,其全部内容在此引用作为参考:“过程环境中的关联图形显示(Associated Graphic Displays in Process Environment)”(代理备案号:06005/41111);“用于过程控制系统的用户可配置报警和报警趋势(User Configurable Alarms and Alarm Trending for Process Control Systems)”(代理备案号:06005/41112);“加工厂的过程模块和专家系统集成(Integration of Process Modules and Expert Systems in Process Plants)”(代理备案号:06005/41113);“在集成化环境中具有用户化过程图形层的加工厂用户界面系统(A Process Plant UserInterface System Having Customized Process Graphic Display Layers in an Integrated Environment)”(06005/41114);“过程环境中的脚本化图形(Scripted Graphics in a Process Environment)”(代理备案号:06005/41115);“过程配置和过程环境中的图形集成(Graphics Integration into a Process Configuration and control Environment)”(代理备案号:06005/41116);“过程环境中具有多个可视功能的图形元素(Graphic Element with Multiple Visualizations in a Process Environment)”(代理备案号:06005/41117);“在加工厂中配置图形显示元件和过程模块的系统(System for Configuring Graphic Display Elements and Process Modules in Process Plants)”(代理备案号:06005/41118);“用于统一的过程控制系统界面的图形显示配置框架(Graphic Display Configuration Framework for Unified Process Control System Interface)”(代理备案号:06005/41124);“加工厂用户界面中的基于标记语言的动态过程图形(代理备案号:06005/41127)”;“用于修改过程控制数据的

方法和装置(Methods and Apparatus for Modifying Process Control Data)”(代理备案号:06005/591622 和 20040/59-11622);“用于访问过程控制数据的方法和装置(Method and Apparatus for Accessing Process Control Data)”(代理备案号:06005/591623 和 20040/59-11623);以及“用于过程控制系统的集成化图形运行期界面”(代理备案号:06005/591628 和 20040/59-11628)。

技术领域

[0004] 本发明总的来说涉及过程控制系统,更具体而言,涉及一种用于过程控制系统的面向服务的架构。

背景技术

[0005] 过程控制系统(比如那些化学,石油或其它过程中用到的)一般包括一个或多个过程控制器和输入/输出(I/O)设备,它们通过模拟、数字或模/数相结合的总线与至少一个主机或操作员工作站以及与一个或多个现场设备通信地连接。现场设备可以是阀门、阀门定位器、开关以及变送器(如温度、压力和流速传感器),用在过程中执行诸如打开或关闭阀门以及测量过程参数等功能。过程控制器接收的信号用来表示现场设备生成的过程测量结果和/或关于现场设备的其它信息,并使用该信息执行一控制例行程序,然后产生控制信号,并通过总线或其它通信线路将该控制信号发送给现场设备以控制对过程的操作。按照这种方法,过程控制器可以利用现场设备通过总线和/或别的与现场设备相通信连接的通信链路来执行并协调控制策略。

[0006] 可以将来自现场设备和控制器的信息应用于由操作员工作站(如基于处理器的系统)执行的一个或多个应用程序(如软件例行程序、程序等等),以使操作员能够根据需要执行有关过程的功能,例如查看当前过程状态,对过程进行评估,以及修改过程操作等等。许多过程控制系统也包括一个或多个应用程序站。一般地,这些应用程序站由个人计算机、工作站或其它类似设备来实现,它们通过局域网(local area network,LAN)与控制器、操作员工作站以及过程控制系统内的其它系统进行通信地连接。每个应用程序站可以执行一个或多个软件应用程序。这些软件应用程序可以在过程控制系统内执行活动管理功能、维护管理功能、虚拟控制功能、诊断功能、实时监测功能、安全相关功能、配置功能等等功能。

[0007] 用于实现许多过程控制系统的各种应用程序的软件元素(如程序)一般都非常依赖使用面向对象的编程技术和架构。这种面向对象的编程技术和架构基于分层分布的软件对象,其中更高一层(或者更复杂的)软件对象由一个或多个较低一层的对象构造而成并继承了其属性。在这些面向对象的程序设计构造内使用大量的继承使得代码(即软件指令,程序等等)高度共享,这就趋向于大大减小用来实现控制系统所需的代码或程序总量。

[0008] 虽然现有的面向对象的程序设计构造能够有利地减小用来实现相对复杂的过程控制系统所需的代码或软件总量,但是与这些构造相关的高度继承或代码的高度共享导致了组成控制系统的各种例行程序或组件之间的高度依赖性(例如,数据依赖性)。结果,对这些各种软件例行程序的独立开发和/或修订变得非常困难或者说不可能。例如,在一些使用上述面向对象的构造的过程控制系统中,过程控制系统应用程序(如过程监测应用程序)、数据库服务以及运行期服务被紧密绑定(即彼此高度依赖)。尤其是,客户端应用程序

可以建立在一套通用的代表数据库中数据的数据组件。这样，通用数据组件中的任何改变都要求重建客户端应用程序。结果，这些公知的面向对象的过程控制软件架构固有的数据依赖性使得对数据库组件和运行期软件组件的独立修改或开发变得非常困难或不可能，特别是当过程控制系统的不同软件组件在不同的地方（即开发地点和开发中心）开发的情况。为了解决这些公知的面向对象的架构固有的数据依赖性问题，过程控制软件开发者必须严密协调数据库、运行期以及系统软件组件的开发，以使得这些组件能以统一的方式创建和发布。

发明内容

[0009] 在一个示例中，一种用于在过程控制系统中的客户端过程和服务器过程之间传递过程控制信息的方法和装置创建包括多个过程控制服务的服务器过程，每个过程控制服务都有对应的服务界面。该示例性方法和装置还建立一客户端过程，该客户端过程具有用于多个服务中的每个服务的代理，并与所述服务之间建立通信连接。另外，该示例性方法和装置向该客户端过程提供与所述服务界面相关联的端点信息（如端口信息、URL、URI，等等），以使得过程控制信息能够在该客户端过程和服务器过程之间传递。

[0010] 在另一个例子中，一种将信息绑定至运行期（runtime）应用程序的方法将与运行期应用程序相关联的数据上下文提供给一资源管理器，并通过该资源管理器将来自该数据上下文的至少一个属性与变量、脚本或数据源引用中的至少一个相关联。另外，该方法将所述至少一个属性绑定至变量、脚本或数据源引用中的所述至少一个。

附图说明

[0011] 图 1 为本文描述的使用面向服务的架构的示例性过程控制系统的框图。

[0012] 图 2 为描述可以用于图 1 的示例性过程控制系统的面向服务的架构的框图。

[0013] 图 3 是描述由图 2 的示例性架构提供的发现服务器和运行期服务器之间关系的图。

[0014] 图 4 为描述一种方法的框图，在该方法中，图 3 的发现服务及其主控服务器与图 3 的数据访问服务及其主控服务器协同工作。

[0015] 图 5 为一示例性过程，图 3 中发现服务通过该过程可以定位一服务。

[0016] 图 6 为描述一会话服务的示例性实现的图。

[0017] 图 7 为图 3 中示例性的运行期服务器提供的更为详细的运行期服务的图示。

[0018] 图 8 为可以由图 6 中的数据库服务器提供的更为详细的示例性数据服务的图示。

[0019] 图 9 为一示例性方法的示意图，在该方法中，版本控制服务器、版本服务器以及图 6 中的数据库服务器可以协同工作。

[0020] 图 10 为可以提供多个历史服务的更为详细的示例性历史服务器的图示。

[0021] 图 11 为描述一示例性历史扫描服务器以及方法的图，在该方法中，该历史扫描服务器可以和图 7 中的示例性运行期服务以及图 10 的示例性历史服务协同工作。

[0022] 图 12 描述一示例性报警和事件服务器，该服务器可用于实现本文描述的示例性面向服务的架构。

[0023] 图 13 描述一示例性 OPC 数据服务器，该服务器可用于实现本文描述的示例性面向

服务的架构。

[0024] 图 14 描述一种方法的示例，在该方法中，本文描述的示例性面向服务的架构可以用于将与客户端应用程序相关联的运行期过程链接或绑定至数据服务或数据源。

[0025] 图 15 描述一种方法的更为详细的示例，在该方法中，可以将用户界面显示图形链接或绑定至数据服务或数据源。

[0026] 图 16 描述一示例性的处理器系统，该系统可以用于实现本文描述的装置和方法。

具体实施方式

[0027] 总的来说，本文描述的示例性装置、方法和制品可用在过程控制系统中，以提供组成过程控制系统的各种软件元素（如程序、应用程序、服务等等）之间的高度独立性。具体来说，本文描述的示例利用了软件和硬件架构，该架构提供了多个松散耦合的（即数据基本上独立的）核心服务，多个独立的过程控制相关应用程序根据该核心服务进行分层。每个核心服务提供可以被独立测试或修订的明确的外部界面（比如模式、输入服务和 / 或从服务输出的参数、数据格式等等）。类似地，由于应用程序的数据基本上独立的特点，所以能够以高度独立的方法（比如，使不同开发团队和不同地方间的协作最小或很小）创建和开发过程控制相关的应用程序。换句话说，一个地方的软件开发人员在开发用于整个过程控制系统的特定应用程序时，不必紧密地参与或被告知由另一个软件开发团队在另一个物理位置对另一个应用程序进行的有关开发。在该方法中，本文描述的面向服务的架构的例子最小化或消除了不同开发团队花费大量时间（比如通过会议或其它形式的交流）来协调各自的开发成果的需要，因此很大程度上减少了各自的开发成果所需要的开发时间和成本。

[0028] 与许多公知的注重源码或软件重用的面向对象的实现或架构相反，本发明的示例性面向服务的过程控制架构强调数据服务和 / 或客户端应用程序之间的相对松散的耦合（比如，若有的话，相对小的数据依赖性）。另外，应用公知的面向对象的实现，过程控制软件应用程序或程序本质上由从共享类库中得到的软件元素组成或编成。相反，本文描述的面向服务的架构的例子使用由相对独立的服务组成的应用程序。具体地说，与本文描述的示例性面向服务的架构相关联的相对独立的服务不再共享类信息，而是共享模式（对结构来说）以及契约（对动作来说）。本文描述的每个相对独立的服务（如数据服务）公开一契约（比如使用 XML 等可扩展标记语言），该契约描述了该服务所能发送和接收的消息的结构。

[0029] 更进一步，公知的面向对象的过程控制软件实现认为软件元素或组件（如服务）和开发人员（如在不同开发地点的软件开发人员）之间的通信是便宜（比如站在有效处理的立场上）和不明显的。相反，本文描述的示例性面向服务的架构认为这样的通信是昂贵和明显的。因此，本文描述的示例性面向服务的架构被配置为用来减少抽象行为的次数和复杂度，其中这些抽象行为必须是共享的交叉服务（即与组成过程控制软件的核心服务的各种软件组件相关联的交叉服务边界）。

[0030] 公知的面向对象的实现以一种单一的方式（即一组相关的应用程序的公共发布）开展应用程序的使用，与之进一步相反的是，本文描述的示例性面向服务的架构以一种进行的、更为离散的方式开展应用程序的使用。例如，使用本发明提供的面向服务架构的例子，对客户端应用程序的修订可以独立于被其使用的底层数据服务，这样使整个系统和应用程序总的状态随着时间的推移可以变为混合。例如，在某些情况下，对服务的适当使用可

以在利用该服务的应用程序被使用之前进行。另外，应用程序和服务的充分的独立性使得服务和 / 或应用程序能够随着时间的推移以不同的速度独立发展(即以独立的方式开发和修改)。然而，为了保证在本文描述的示例性面向服务的架构或框架中使用的相对高度分离的服务和 / 或应用程序的相互操作中有着更大程度的完整性，每个服务可以被配置为对每个从别的服务或应用程序接收的消息执行一确认过程。

[0031] 如接下来进行更为详细的描述的，所述示例性面向服务的架构可以方便地用于将与运行期应用程序相关联的参数关联于、绑定于，或者更为一般地说，通信地或可操作地连接于变量、可执行脚本和 / 或上面提到的数据服务。例如，用于描述与一个过程控制系统用户界面应用程序相关联的图形显示元件的相关参数可以与一运行期数据服务通信地和可操作地连接。在这个例子中，连接于所述参数的数据的改变能够从数据服务自动地传送至显示应用程序，比如，这样能够接着更新图形显示来反映数据的改变。在一个实现中，由微软公司(Microsoft Corporation)提供的公知的 Avalon 和 Indigo 软件平台可以用来将与显示元件相关的属性绑定至变量、脚本和 / 或运行期数据服务。然而，也能够选用替代或者附加 Avalon 和 / 或 Indigo 的任意别的合适的软件平台。

[0032] 下面进行更为详细的描述，本文描述的示例性面向服务的架构包括数据服务层，用来提供过程控制相关数据给数据源层。如下所述，数据源层从一个应用程序内部为数据服务层提供相对一般的访问。位于数据源层的资源管理器管理变量、脚本和 / 或引用，使其包含在与一个或多个运行期应用程序、显示、图形元素，参数等等相关联的数据源中。比如，本文描述的资源管理器有利地使得与图形显示元件相关联的参数能够在运行时被绑定至一个或多个变量或数据服务(例如通过服务器提供的过程控制数据源)，而不是对参数和这样的变量或数据源之间的耦合进行静态编码。

[0033] 更具体来说，通过本文描述的示例性装置、方法以及制品，显示应用程序可以向资源管理器为与它的图形元素相关联的各种属性请求数值或者别的数据或信息。接下来，资源管理器可以在运行时被配置为向显示应用程序使用的属性提供(例如映射、关系或其它方式的关联)数据源。结果，显示应用程序不必被静态编码来定义它的图形元素和变量或数据源之间的关系，其中该数据源被绑定至与这些图形元素相关联的参数值。取而代之，资源管理器能够被配置为在运行时更加灵活和动态地定义这些关系。

[0034] 图 1 是使用本文描述的示例性面向服务的架构的示例性过程控制系统 10 的框图。如图 1 所示，过程控制系统 10 包括 :控制器 16、操作员站 18、运行应用程序站 20 和备用应用程序站 22，它们都可以通过总线或局域网(Local Area Network, LAN)24 进行通信连接，该局域网一般称为应用程序控制网络(Application Control Network, ACN)。操作员站 18 和应用程序站 20 和 22 可以用一个或多个工作站或任何别的适合的计算机系统或处理单元来实现。例如，应用程序站 20 和 22 能够用与下面图 16 中所示的示例性处理器系统 1602 类似的单处理器个人计算机、单或多处理器工作站等等来实现。另外，LAN 24 可以使用任何所需的通信介质和协议。例如，LAN 24 可以基于硬件的或无线的以太网通信方案，这是公知技术，所以在这里不再详述。然而，本领域普通技术人员很容易理解，任何别的合适的通信介质或协议都可以使用。进一步，虽然示出了单个 LAN，但更多个 LAN 和适用于应用程序站 20 和 22 的通信硬件可以用于为操作员站 18、应用程序站 20 和 22 和控制器 16 之间提供一条冗余的通信通路。

[0035] 控制器 16 可以与多个智能现场设备 26、28 和 30 通过数字数据总线 32 和输入 / 输出(I/O)设备 34 相连接。智能现场设备 26-30 可以是 Fieldbus 兼容(compliant)阀、执行器、传感器等等,其中,智能现场设备 26-30 使用公知的 Fieldbus 协议通过数据总线 32 进行通信。当然别的类型的智能现场设备和通信协议也能作为替代。例如,智能现场设备 26-30 能够由 Profibus 或 HART 兼容设备来代替,所述 Profibus 或 HART 兼容设备使用公知的 Profibus 或 HART 通信协议通过数据总线 32 进行通信。另外的 I/O 设备(与 I/O 设备 34 类似或相同的设备)可以与控制器 16 相连接使得别的成套的智能现场设备(可以是 Fieldbus 设备、HART 设备等等)能够与控制器 16 进行通信。

[0036] 除了智能现场设备 26-30 之外,一个或多个非智能现场设备 36 和 38 可以与控制器 16 进行通信连接。例如,所述非智能现场设备 36 和 38 可以是传统的 4-20 毫安(mA)或 0-10 伏直流(VDC)设备,它们可以通过各自的硬件链接 40 和 42 与控制器 16 通信。

[0037] 例如,控制器 16 可以是由爱默生过程管理有限责任合伙(Emerson Process Management,LLLP)销售的 DeltaVTM 控制器。然而,也可以用别的控制器代替。进一步说,虽然图 1 中仅示出了一个控制器,但别的任何所需类型或多种类型相结合的控制器都可以与 LAN 24 相连接。控制器 16 可以执行一个或多个与过程控制系统 10 相关联的过程控制例行程序。这样的过程控制例行程序由一个系统工程师或别的系统操作员使用操作员站 18 生成,并可被下载至控制器 16 进行实例化。

[0038] 如图 1 所图示的,示例性过程控制系统 10 也可以包括远程操作员站 44,该操作员站 44 通过通信链路 46 和 LAN 48 与应用程序站 20 和 22 进行通信连接。远程操作员站 44 可以是地理上的远程位置,在这种情况下通信链路 46 优选是,但并不必须是无线通信链路、基于互联网的或别的分组交换通信网络、电话线路(如数字用户专用线路)或以上的任意组合。

[0039] 如图 1 中的例子所描述的,运行应用程序站 20 和备用应用程序站 22 通过 LAN24 以及冗余链路 50 相互通信连接。该冗余链路 50 可以是运行应用程序站 20 和备用应用程序站 22 之间分离的、专用的(即不共享的)通信链路。例如,该冗余链路 50 可以使用专用的以太网链路(如相互连接的运行应用程序站 20 和备用应用程序站 22 中的专用以太网卡)来实现。然而,在别的例子中,所述冗余链路 50 可以使用 LAN 24 或冗余 LAN (未示出)来实现,两者不必是专用的,它们之一与应用程序站 20 和 22 通信地连接。

[0040] 一般说来,应用程序站 20 和 22 通过冗余链路 50 连续地,例外地,或周期性地交换信息(如应对参数值的改变,应用程序站配置也随之改变等)来创建和维持冗余上下文。该冗余上下文使得运行应用程序站 20 和备用应用程序站 22 可以进行无缝的或者说无波动的控制移交或转换。例如,冗余上下文使得从运行应用程序站 20 到备用应用程序站 22 的控制移交或转换能够响应运行应用程序站 20 中出现的硬件或软件的故障或者响应来自系统操作员或用户或者过程控制系统 10 的客户端应用程序的指令而进行。

[0041] 图 2 描述了可用于图 1 中的示例性过程控制系统 10 的示例性面向服务的架构或结构 200。如图 2 所示,示例性面向服务的架构或结构 200 包括服务器 202 和客户端 204。服务器 202 包括多个或集成的服务 206、208 和 210,其中的一些或全部可以执行相关的功能。服务 206、208 和 210 提供各自的界面(例如一套或多套的公开的参数)212、214 和 216,所述界面能够通过通信端口 218 与客户端 204 通信。服务界面 212、214 和 216 实际上是充

分上位的，也即充分独立于模式（schema）（即数据格式、协议等等），所述格式用于包含在配置和运行期数据库中与图 1 中示例性过程控制系统 10 相关联的数据。结果，如果增加新的服务能力（如功能）到一个或多个服务 206、208 和 210 中，则仅需要修改（如更新）服务界面 212、214 和 216。这样，就不必要改变服务界面 212、214 和 216，除非是增加使用新公开界面的数据对象用于过程控制系统 10（图 1）。

[0042] 例如，服务器 202 可以作为在基于处理器的系统，比如图 1 的示例性系统 10 中示出的一个或多个应用程序 20 和 22 和 / 或操作员站 18 和 44 中执行的软件来实现。当然，服务器 202 也可以使用其它任何与所述示例性过程控制系统 10（图 1）相连接的基于处理器的系统或工作站来实现。

[0043] 客户端 204 包括多个服务界面代理 220、222 和 224，所述各个代理与服务 206、208 和 210 中的一个相对应。客户端 204 使用服务界面代理的数目可以少于服务器 202 提供的服务的数目。换句话说，客户端 204 优选仅为其要求接入的服务产生代理。这样，根据与服务器 202 提供的服务 206、208 和 210 中的一个或多个进行访问或交互的需要，客户端 204 可以产生一个或多个代理。

[0044] 与服务器 202 类似，例如，客户端 204 可以作为在基于处理器的系统，比如一个或多个应用程序 20 和 22 和 / 或一个或多个操作员站 18 和 44 中执行的软件来实现。在一个示例性实现中，客户端 204 可以利用网络浏览器框架（如因特网浏览器）或其同类来访问由服务器 202 提供的服务 206、208 和 210 中的一个或多个。然而，任何别的所需软件架构可以用于代替或补充这种网络浏览器框架。一般来说，客户端 204 可以指示例性过程控制系统 10（图 1）内的任何所需的应用程序。这样，例如，客户端 204 可以为配置应用程序、维护应用程序、监测应用程序、过程控制应用程序和 / 或任意其它应用程序或应用程序的组合。如下面图 14 和 15 中更加详细的描述，客户端 204（即客户端应用程序）可以包括显示功能性（比如图形用户界面功能性），这使得一个或多个系统操作员、工程师和 / 或任意其它用户能够在配置操作或运行时等等期间查看和 / 或改变过程控制数据。

[0045] 虽然图 2 中的示例性架构 200 描述单个服务器与单个客户端之间的通信，但其它服务器和客户端也可以视需要使用。例如，在某些实现中，客户端 204 可以与多于一个的服务器中的服务进行通信、相互操作和 / 或访问。同样，在这些实现或别的实现中，示例性服务器 202（或其它单个的服务器）可以与多个客户端通信和相互操作。另外，应该认识到，客户端 204 可以和另一系统通信，并且为了通信的目的可以作为服务器使用。同样，在服务器 202 与除了客户端 204 之外的系统通信的情况下，它可以作为客户端使用来进行通信。进一步说，虽然服务 206、208 和 210 被描述为在服务器 202 中实现，但代替地，这些服务 206、208 和 210 中的一个或多个可以在客户端 204 中实现。

[0046] 这样，利用图 2 中示例性面向服务的架构 200，服务 206、208 和 210 彼此充分地分离（比如在数据相关性方面），并与利用（比如调用）服务 206、208 和 210 的应用程序充分地分离（比如在数据依赖性方面）。这种分离有利地使与每个服务 206、208 和 210 相关联的软件能够被独立地修改或修订，并且可以不必修改或修订被客户端 204 使用的用来访问服务 206、208 和 210 的应用程序而发布用于现场使用。同样，与客户端 204 相关联的应用程序可以被独立地修改或修订而不必要修改或修订服务 206、208 和 210，只要与客户端 204 相关联的应用程序支持或兼容各个服务 206、208 和 210 的界面 212、214 和 216。这样，代替通过在

生成与应用程序和 / 或服务 206、208 和 210 相关的软件时固定这样的关系(即生成数据依赖性),来静态定义与客户端 204 相关联的应用程序以及服务 206、208 和 210 中的一个多个之间的关系,图 2 中的示例性架构 200 允许这样的关系在运行时动态地建立。

[0047] 下面将进行更加详细的描述,所述面向服务的架构 200 可以提供多种核心服务和功能。具体来说,示例性架构 200 可以在示例性过程控制系统 10 (图 1)为注册和定位服务提供发现服务。示例性架构 200 可以提供 :会话服务,用来实现登录和退出一个或多个与客户端 204 相关联的应用程序 ;运行期服务,用来提供对本地和远程运行期信息的访问 ;数据库服务,用来访问各种过程控制数据库 ;以及版本控制服务,用来存储和检索与用于示例性过程控制系统 10 (图 1) 内的服务和 / 或应用程序相关的版本信息。另外,示例性架构 200 可以提供历史服务,用于查询历史信息 ;历史扫描服务,其要求与运行期服务和历史服务交互作用 ;报警和事件服务 ;用于完成对报警信息的读取 ;以及 OLE 过程控制(OPC) 数据服务。下面将更详细地描述上述每个示例性服务。然而,应该理解的是,如果需要,可以使用作为本发明明确描述的补充或替代的其它服务。进一步,应该认识到,优选地但并不是必须地,服务 206、208 和 210 一般与过程控制数据收集和处理功能相关联,系统用户(比如,系统操作员、工程师等等) 并不直接与该功能交互。相反,这样的服务(比如,服务 206、208 和 210) 一般提供它们的数据给(或接收数据于) 用户与之交互的应用程序(比如运行期图形用户界面)。进一步,这样的服务(如服务 206、208 和 210) 一般提供对多个客户端应用程序有用的信息或数据。例如,一个提供对过程控制数据库进行访问的服务可以对多个客户端应用程序,比如提供与过程控制操作、维护和历史等等相关的信息的图形用户界面是有用的。通过分开服务类功能(如数据库访问功能) 与客户端相关功能(如用户界面功能),可以以修订,修改等方式处理图形用户界面及其同类,而没有必要修改与用户界面交换数据的服务。同样,可以修订或修改所述服务和 / 或增加新的服务,而不必修改提供用户界面功能性的应用程序。

[0048] 图 3 是表示由图 2 中的示例性架构 200 提供的示例性发现服务 300 与运行期服务器 302 之间的关系。总的来说,示例性发现服务 300 提供服务注册功能 304、服务定位功能 306 和服务发布功能 308。

[0049] 如图 3 所示,发现服务 300,以致注册功能 304、定位功能 306 和发布功能 308 与运行期服务器 302 提供的数据访问服务 310 协同工作。数据访问服务 310 是下面将在图 7 中讨论的运行期服务器 302 提供的多个服务中的一个,它提供一界面来完成对这些服务相关信息的查询,所述服务可以安装在过程控制系统(如图 1 中的示例性系统 10)内的每个节点(如每个服务器) 上。

[0050] 使用统一资源标识符(Uniform Resource Identifier,URI)可以定位和 / 或注册服务。这样的 URI 可以采用包括服务器标识符字段、端口字段、路径标识符字段和 / 或查询字符串的地址形式。然而,任何别的所需 URI 格式可以用来达到同样或类似的结果。一些示例性 URI 格式(即身份角色、http 地址和定制 http 地址) 如下所示。

[0051] 身份角色

[0052]

```
Soap://www.contoso.com:ppp/whatsnew.aspx?date=today
        |          |   |          |-可选的查询字符串
        |          |   |-路径标识符
        |          |-端口
        |-服务器标识符
```

[0053] http 传输地址

[0054]

```
Soap.http://www.contoso.com:ppp/whatsnew.aspx?date=today
        |    |          |   |          |-可选的查询字符串
        |    |          |   |-路径标识符
        |    |          |-端口
        |    |-服务器标识符
        |-方案标识符
```

[0055]

[0056] 定制传输地址

[0057]

```
Soap.starburn://hostname:pppp/whatsnew.aspx?date=today
        |       |       |   |          |-可选的查询字符串
        |       |       |   |-路径标识符(我们的服务名)
        |       |       |-端口
        |       |-服务器标识符
        |-方案标识符
```

[0058] 图 4 为描述了一种方法的流程图，该方法中，示例性发现服务器 300（图 3）及其主控发现服务器 312 与数据访问服务 310 及其主控运行期服务器 302 协作或相互合作。总的来说，图 4 中描述的过程使得能够通过发现服务器 312 注册一个或多个运行期服务（如图 7 中的示例性运行期服务 700）。再详见图 4，当一工作站（比如操作员站、应用程序站等等）引导装入（块 402）时，注册过程 400 开始。然后，所述工作站实例化或启动发现服务器（比如图 3 中的发现服务器 312）（块 404），发现服务器接着启动发现服务（比如图 3 中的发现服务 300）。一旦发现服务器（其主控发现服务）启动发现服务，就公布之前与发现服务一起注册的 URI。

[0059] 在块 406 处启动发现服务后，工作站启动运行期服务器（块 408）。之后，运行期服务器通过产生一 URI 占位符，并将一端口关联至（或分配一端口给）URI 来产生一发现代理（块 410）。然后将在块 410 中分配的端口号传递给所述发现服务（块 412）

[0060] 在块 412 处，在将端口号发给发现服务后，运行期服务器使用在块 410 处分配的端口生成一要注册的 URI（块 414），然后向运行期服务公布要注册的 URI（块 416）。然后过程 400 通过一发现代理注册在块 416 处公布给发现服务的 URI（块 418）。

[0061] 图 5 为一示例性过程 500，发现服务可以通过该过程定位一服务（比如为想要的服

务确定 URI)。当客户端应用程序启动时,所述示例性过程 500 开始(块 502)。客户端应用程序接着为要确定的 URI 生成一占位符(块 504),并通过一发现代理调用发现服务内的定位功能(传送要定位的服务的名称)(块 506)。要定位的服务的 URI 作为块 506 处操作的结果返回至客户端应用程序,接着通过一服务代理启动已定位的服务(块 508)。

[0062] 图 6 为描述会话服务 600 的示例性实现的示意图。总的来说,示例性会话服务 600 装载入每个客户端应用程序,并跟踪与应用程序的调用应用程序和用户相关的特权。这样,此处所述的每个服务服务器包括一会话服务,用于协调对服务器提供的服务进行的访问,并确认与用户 / 客户端应用程序组合相关联的安全证书,其中这种用户 / 客户端应用程序组合要求访问由服务器提供的一个或多个服务。

[0063] 在操作中,客户端应用程序使用一特定的 URI 与服务器连接并请求会话服务界面。所述会话服务界面提供登录和注销功能。当调用登录功能时,基于请求的客户端应用程序和用户身份来进行安全确认,如果安全确认通过,则返回一会话句柄,并且该请求的客户端应用程序被连接于一运行期会话。如果客户端应用程序请求连接至由该服务器提供的另一(例如附加的)服务,则客户端应用程序将该会话句柄传递至该服务器,然后该服务器使用所述会话句柄在内部增加另一个服务给与该客户端应用程序相关联的会话。当调用注销功能时,所有没有被客户端应用程序显式关上的服务将被关上。

[0064] 现在再对图 6 进行详细描述,示例性会话服务实现 600 包括客户端过程 602、数据库过程 604 以及运行期过程 606。客户端过程 602 可以包括一个或多个应用程序,其在比如应用程序站(如图 1 所述的示例性应用程序站 20)、操作员站(比如图 1 中的操作员站 18)和 / 或任何别的能够执行机器可读指令、代码或软件的系统上运行。可以通过数据库服务器 608 全部执行或部分执行数据库过程 604,可以通过运行期服务器 302 (图 3) 全部执行或部分执行运行期过程 606。

[0065] 如图 6 所示,示例性会话服务 600 的客户端过程 602 包括多个客户端应用程序 610 和 612,它们具有各自的会话服务代理 614 和 616,会话服务代理 614 和 616 与数据库服务器 608 内的会话服务实例 618 和 620 相对应。另外,与客户端过程 602 相关联的每个示例性应用程序 610 和 612 请求数据库访问,因此它们包括各自的代理 622 和 624,所述代理 622 和 624 与数据库服务器 608 提供的数据访问服务实例 626 和 628 相关联。示例性客户端过程 602 也提供与下载服务 632 相关的代理 630。所述下载服务 632 可以由数据库服务器 608 提供。数据库服务器 608 进一步包括会话管理器 634,用于跟踪和 / 或管理数据库服务器 608 内部的有效会话(如会话 618 和 620)。

[0066] 数据库服务器 608 内部的会话 618 和 620 以及下载服务 632 通过各自的代理 642、644 和 646 与运行期服务器 302 中各自的实例 636、638 和 640 可操作地相关联。机器终端会话 648 管理会话 636 和 638,持有会话对象 650,并与安全服务 652 相互操作来为会话对象 650 执行上述的安全确认操作。

[0067] 这样,从图 6 中的例子实现可以看出,每个客户端应用程序可以与由一个或多服务器提供的一个或多个服务相互操作。更具体地说,在客户端应用程序请求访问过程控制数据库时,它与提供被请求的数据库访问服务的服务器建立会话(比如通信链路)。例如,在图 6 的示例性实现中,应用程序 610 在数据库服务器 608 中建立一会话(比如通过会话服务 618),并在数据库服务器 618 中使用数据访问服务 626。以这种方式,客户端应用程序可以

使用多个松散耦合的(或基本上数据独立的)服务来实现,其中每个服务可以被主控于不同的服务器上。客户端应用程序接着可以通过每个主控服务器中一个或更多个代理以及一个或更多个服务会话对象与每个应用程序需要的服务进行通信。

[0068] 图 7 更为详细地描述了运行期服务器 302 (图 3)提供的运行期服务 700。总的来说,运行期服务 700 提供访问本地和远程运行期过程控制信息。如图 7 所示,运行期服务 700 可以包括数据访问服务 310、上载服务 704、Fieldbus 试运行(commissioning)服务 706、运行期会话服务 708、下载服务 710、试运行服务 712、子系统服务 714 以及下载收听者服务 716。

[0069] 数据访问服务 310 提供的功能能够在运行期数据库(比如图 1 的示例性过程控制系统 10 内包含过程控制参数值的运行期数据库)中读出和写入参数。另外,数据访问服务 310 能够列举运行期数据库内成套的项目(比如被试运行的控制器节点)。

[0070] 上传服务 704、下载服务 710 以及下载收听者服务 716 能够向运行期数据库上传或下载数据,Filedbus 试运行服务 706 和试运行服务 712 使得现场设备能够在运行期数据库内进行试运行,而运行期服务 708 以上述方式结合图 6 中的例子实现与机器终端会话 720 和服务器会话服务 718 协作执行安全功能。

[0071] 图 8 为示例性数据库服务 800 的更为详细的描述性视图,其中数据库服务 800 可以由数据服务器 608 提供。如图 8 所示,数据库服务 800 可以包括数据访问服务 802、下载服务 804、上载服务 806、Fieldbus 试运行服务 808、输入服务 810、输出服务 812、试运行服务 814、数据库管理服务 816 以及版本控制服务 820。数据访问服务 802 提供查询和更新数据库的功能。数据访问服务 802 也提供一种命令功能,该功能允许由要执行的一客户端应用程序向数据库服务器 608 中的数据库下发的命令。每条命令可以包括多个要应用于数据库的协商和更新操作。另外,每条命令可以在单个数据库处理中执行。所述命令在应用程序中组成字符串并进行传递,接着在数据库服务器 608 中作为脚本而被解释(比如被编译并执行)。优选地但并不是必须地,所述指令由参数表示以便公用脚本可用于所有类似的操作。数据库服务器 608 可以缓存并预编译脚本以加快执行速度。在一个特定例子中,如果需要版本控制动作(即版本控制服务),则由客户端应用程序送给数据库服务器 608 的更新指令可以触发一应答信号给客户端界面。执行该命令并同时返回到调用例行程序之后,任何可用的更新都可以在数据库服务器 608 内部的数据库中完成。

[0072] 下载服务器 804 可以作为一个支持数据库服务器 608 内排序的界面来实现。下载服务器 804 界面提供的功能有 :在下载列表中添加项、检验并执行上载,通过这些功能来验依赖性,确认下载,并发送下载项给目标系统。下载服务 804 可以被配置为通过与运行期数据访问服务 702 (图 7)相互操作(比如图 6 所示)来执行上载。下载服务 804 也可以与运行期试运行服务 712 (图 7)进行相互操作来保证设备被试运行,并且可以与运行期下载服务 710 (图 7)相互操作来传送下载脚本。

[0073] 输入服务 810 也可以作为数据库服务器 608 中可排序的界面来实现。输入服务 810 可以提供功能来指定一输入文件,指定输入选择,并输入被请求的数据。输入服务 810 的界面也可以为服务器 608 提供一应答,使得调用客户端能够读取输入文件并报告输入过程。

[0074] 输出服务 812 也可以作为可排序的界面来实现。输出服务 812 可以包括能使输出

服务 812 向输出列表添加项并执行输出的界面。另外，输出服务 812 可以在调用客户端应用程序中为服务器提供一应答界面，使得客户端应用程序能够写入输出文件并报告输出过程。

[0075] 数据库管理服务 816 提供数据库的生成、删除、备份和恢复等功能。版本控制服务 820 提供能够进行版本管理（例如，不同的服务版本）的功能。

[0076] 图 9 为描述了一示例性方法的示意图，在该方法中，版本控制服务器 900、版本服务器 902 以及数据库服务器 608 可以相互操作。如图 9 中所描述的，版本控制服务器 900 包括多个版本控制服务 904。该版本控制服务 904 可以包括管理服务 906 和版本服务 908。

[0077] 图 10 为可以提供多个历史服务 1002 的示例性历史服务器 1000 的图解示意图。历史服务 1002 可以包括数据访问服务 1004、输入服务 1006、数据写入服务 1008、输出服务 1010、管理服务 1012 以及审计服务 1014。总的来说，历史服务 1002 使客户端应用程序能够访问和查询历史的过程控制信息。具体来说，数据访问服务 1004 能够实现在指定的时间周期内对历史信息进行读取，写入包含时间信息（比如时间戳）和注释的历史信息，并更新包含时间信息和注释的历史信息。输入服务 1006 允许在一指定文件中输入信息。数据写入服务 1008 可以被调用从而向历史服务器数据库写入数据。输出服务 1010 可以被调用来输出一文件。管理服务 1012 提供能够管理历史信息的功能，比如，生成、删除和移动历史数据档案文件等等。

[0078] 图 11 为描述示例性历史扫描服务器 1100 和方法的图，在该方法中，历史扫描服务器 1100 可以和运行期服务 702（图 7）以及历史服务 1002（图 10）相互操作。如图 11 所示，历史扫描服务器 1100 自己不能提供任何服务，而是需要利用运行期服务 702 和历史服务 1002。

[0079] 图 12 和 13 分别描述了报警和事件服务器 1200 和 OPC 数据服务器 1300。报警和事件服务器 1200 提供多个服务 1202，包括数据访问服务 1204、管理服务 1206 以及配置服务 1208。所述 OPC 数据服务器 1300 提供多个服务 1302，包括数据访问服务 1304、管理服务 1306 以及配置服务 1308。

[0080] 图 14 描述了一种方法的例子，在该方法中，本文描述的示例性面向服务的架构可用于链接和绑定与客户端应用程序相关联的用户界面显示图形至变量、脚本和 / 或数据服务或数据源。如图 14 所描述的，一个或多个数据服务 1400，比如历史数据服务 1402、运行期服务 1404 和 / 或任何别的数据服务 1406，其中一些或所有的数据服务可以在与上面讨论的图 2、图 3 以及图 7-13 相关的示例性服务类似或相同的方法中实现，这些数据服务可以被链接、通信地连接或者可操作地连接至一个或多个运行期过程 1408。运行期过程 1408 可以包括任何想要的运行期过程或应用程序。然而，在图 14 的例子中，运行期过程 1408 包括图形用户界面或显示应用程序 1410，使得用户能够查看和 / 或修改过程控制数据，比如过程控制参数值、过程相关历史信息、趋势信息、维护信息或者报警或提醒信息等等。

[0081] 图 14 还描述了使用数据绑定操作或过程 1412 和数据源操作或过程 1414 来实现运行期过程 1408 和数据服务 1400 之间的通信连接或链接。总的来说，运行期过程 1408、数据绑定过程 1412 以及数据源过程 1414 可以由比如客户端 204（图 2）来执行，而数据服务 1400 可以由比如服务器 202（图 2）来执行。具体来说，与客户端 204（图 2）相关联的过程，包括运行期过程 1408、数据绑定过程 1412 和数据源过程 1414 在内，也可以利用微软公

司提供的 Avalon 操作系统框架来实现。Avalon 框架为公知的，在这里仅就 Avalon 框架的某些方面来讨论以方便对本文所述的示例性装置、方法和制品的理解。进一步说，应该理解的是，虽然微软公司提供的 Avalon 框架仅为一个实现本发明例子特别有用的框架，但也可以使用任何别的合适的软件框架来取代 Avalon 框架或者与 Avalon 框架一起使用。

[0082] 下面再详细描述图 14，数据绑定相关过程 1412 包括资源管理器 1416，它产生和跟踪变量、脚本和数据源引用，如下面将进行更加详细地描述的。每个显示（例如显示 1410）由它自己的资源管理器实例（例如资源管理器 1416）进行管理。如图 14 所示，示例性显示 1410 包括图形元素 1418 和相关的属性 1420，它们与一个或多个变量 1422、脚本（即可执行的软件脚本）1424 和数据源引用 1426 相关联，数据源引用 1426 接下来通过上下文 1430 的一个或多个绑定或绑定对象 1428 被连接、链接或者绑定至数据服务 1400 中的一个或多个。绑定对象 1428 可以包括数据源（如数据服务 1400 中的一个或多个）内的特定数据项的位置的字符串描述，这样，绑定对象 1428 对于一数据源来说是特定的。另外，所述字符串描述可以包含一别名引用，它可以在运行期间被改变。如图所示，上下文 1430 也可以与运行期数据源 1431 和数据源管理器 1432 相连接，这将在下面讨论。

[0083] 如图 14 所描述的，资源管理器 1416 从显示 1410 收到一数据上下文。总的来说，数据上下文由显示 1410 表示元素 1418 时使用的属性 1420 组成。资源管理器 1416 被配置为将与数据上下文相关联的属性 1420 映射、关系或者关联至变量 1422、脚本 1424 和引用 1426 中适当的变量、脚本和引用。另外，资源管理器 1416 可以进一步被配置为通过绑定 1428、上下文 1430 和数据源管理器 1432 将属性 1420 映射、关系或关联至一个或多个数据服务 1400。比如配置工程师或别的系统用户能够在系统配置操作期间建立由资源管理器 1416 使用的映射，以在显示 1410 的属性 1420 与变量 1422、脚本 1424、引用 1426 和 / 或数据服务 1400 之间建立关系。这样，与许多公知的过程控制系统显示或用户界面相反的是，显示 1410 的运行期特征，更一般地说，运行期过程 1408，不能由组成与运行期过程 1408 相关联的应用程序的软件来静态地定义。相反，资源管理器 1416 被配置为在运行时建立的显示 1410 的属性 1420 与变量 1422、脚本 1424、引用 1426 和数据源 1400 中的一个或多个之间的关系。

[0084] 在操作中，资源管理器 1416 可以接收与变量、数值、文本名称等的改变相关的信息。在某些情况下，数据的变化可以通过提供给资源管理器 1416 的数据上下文中的改变来传递。具体来说，在上述情况下，用户发起的对一个或多个属性 1420 的改变可以调用一个或多个脚本 1424 来改变与变量 1422 和 / 或引用 1426 相关的一个或多个数值。在所述改变与存储在一个或多个数据服务 1400 的数据相关的情况下，对由资源管理器 1416 管理的对象的改变可以通过绑定 1428 和上下文对象 1430 自动传送至一个或多个合适的数据服务 1400。在别的情况下，数据的改变可以在一个或多个数据服务 1400 中发生（比如作为过程变量值改变的结果），而且通过由资源管理器 1416 提供的映射功能性，所述改变可以通过上下文 1430 和绑定 1428 自动传送至合适的属性 1420。

[0085] 数据源管理器 1432 产生和管理上下文对象（如上下文对象 1430），它用于维护与一组与一数据源相关联的绑定对象（如绑定对象 1428）相关的客户端状态信息。在与图形用户界面相关联的显示的情况下，在上下文对象与显示之间有着一一对应的关系。换句话说，每个显示有一个上下文，并且每个上下文仅包含与该显示相关联的绑定。这样在图 14

的例子中，上下文对象 1430 唯一地与显示 1410 相对应。

[0086] 上下文对象也提供数据更新机制，其中之一基于事件并且响应每个或各个事件发起更新，其中另一个实际上用于核对并且收集多个数据改变事件，通过（如根据一计时器）响应计时器来发送改变的绑定列表给数据源来周期性地发起更新。当然，如果有相当大量的事件要处理，独自响应每个事件可能会削弱运行期过程 1408 和 / 或整个过程控制系统（如图 1 中的过程控制系统 10）的性能（比如滞后响应）。这样，周期地响应脏绑定列表（如对于已被收集的绑定，数据已因其改变，但还没有被传递至运行期过程）可以提高整体性能。上下文对象 1430 和数据服务 1400 之间的连接可以使用比如微软公司的 Indigo 框架来实现。

[0087] 虽然与图 14 中的示例相关联的改变信息的流程被描述为在显示 1400 发起并通过资源管理器 1416、绑定 1428、数据源管理器 1432 以及上下文 1430 被自动传播至一个或多个数据服务 1400，但数据改变信息能够可选地或者附加地从一个或多个数据服务 1400 自动传播至显示 1410。

[0088] 图 15 描述了一种方法的例子，在该方法中，用户界面显示图形可以被链接或绑定至数据服务或数据源。如图 15 所图示的，显示（比如应用程序或别的提供显示的运行期实体）1502 包括显示元件或图形 1504，其接下来与参数或显示内容 1506 相关联。显示 1502 可以是与一个或多个过程控制相关的应用程序相关联的用户界面，并且，这样的话，图形 1504 可以是过程控制相关图形，比如用来容纳过程控制数值的文本框，过程控制设备比如泵和阀门等的图形表示。显示内容 1506 可以包含文本信息（比如，在显示图形 1504 为一文本框的情况）、代表过程条件（如红色可以指示一参数值达到或超过预定界限和 / 或需要采取纠正措施）的颜色（比如多种填充色中的一种）、图形元素（比如风机或泵叶轮）的转速、马达或别的振动或移动等设备的三维视图和 / 或任何别的内容。

[0089] 显示 1502 可以通信地和可操作地与资源管理器实例 1508 相连接，该资源管理器实例 1508 接下来还通信地和可操作地与过程上下文 1510 和过程绑定 1512 相连接。过程上下文 1510 可以被配置来提供给过程绑定 1512 被核对的数据改变通知或事件驱动数据改变通知。过程绑定 1512 可以参考数据源或数据服务中固定存储位置中的数据和 / 或可以参考数据在数据源或数据服务中的别名。过程绑定 1512 通过绑定对象 1514 与显示内容 1506 通信地或可操作地相连接。

[0090] 在操作中，资源管理器 1508 具有由显示 1502 用来再现显示图形 1504 和显示内容 1506 的数据上下文或参数。资源管理器 1508 将至少一个与显示 1502 相关联的属性映射、关系或关联至由过程上下文 1510 提供的数据，该过程上下文 1510 为存储在数据服务（比如图 14 所示的一个数据服务 1400）内的数据提供数据着陆位置。换句话说，过程上下文 1510 可以用一个上下文对象来实现，比如，上下文对象 1430 （图 14），这样就具有将来自数据服务或别的数据源的信息与资源管理器 1508 以及最终的显示内容 1506 从而显示 1502 进行可操作地和通信地相连接的功能。过程上下文 1510 可以被配置为以被核对的方式（比如周期性地发送一组改变的数据信息）或响应每个数据改变事件来向过程绑定 1512 发送数据改变信息。过程上下文 1510 被配置为接收来自资源管理器 1508 的信息，以通知过程上下文 1510 资源管理器 1508 用于请求改变通知的数据。这样，过程上下文 1510 可以被配置为仅提供与由资源管理器 1508 具有的显示 1502 的数据上下文相关联的数据相关的数据改变通

知。无论如何，当过程绑定 1512 接收到过程上下文 1510 发来的有关一个或多个数据值已改变的通知时，过程绑定 1512（通过绑定 1514）绑定新的值至显示内容 1506 并接着提供给资源管理器 1508 一改变通知。资源管理器 1508 在接到来自过程绑定 1512 的改变通知后，调用显示 1502 上的属性改变方法，从而使得显示 1502 更新以并入新绑定的数据。

[0091] 本文描述的功能块或操作可以用任何期望的软件、固件和硬件的组合来实现。例如，一个或多个微处理器、微控制器、应用程序专用集成电路（ASIC）等可以访问存储在机器或处理器的可存取存储介质上的指令或数据来执行本文描述的方法和装置。存储介质可以包括设备和 / 或介质的任意组合，比如：固态存储介质，包括随机存取存储器（RAM），只读存储器（ROM），电可擦除可编程只读存储器（EEPROM）等，光存储介质，磁存储介质等。另外，作为替代或补充，对于处理器或别的设备或者那些通过因特网、电话线路、卫星通信等来执行软件的设备来说，用于实现这些功能块的软件可以被传送给他们，或由他们来访问。

[0092] 图 16 描述了一个处理器系统 1602 的例子，它可用于实现本文描述的方法或装置。该示例性过程控制系统 1602 可以是比如服务器、个人计算机或别的类型的计算设备。

[0093] 例如，处理器 1600 可以用一个或多个奔腾系列、Itanium 系列或 XScale 系列的 Intel 微处理器来实现。当然，别的系列或公司的别的处理器也是适用的。处理器 1600 通过总线 1608 与包含易失性存储器 1604 和非易失性存储器 1606 在内的主存储器进行通信。易失性存储器 1604 可以由同步动态随机存取存储器（SDRAM）、动态随机存取存储器（DRAM）、RAMBUS 动态随机存取存储器（RDRAM）和 / 或别的类型的随机存取存储设备实现。易失性存储器 1606 可由闪存和 / 或别的所需类型的非易失性存储设备来实现。典型地，对存储器 1604 的存取可由一存储控制器（未示出）以惯用的方式来控制。

[0094] 系统 1602 还包括一接口电路 1610。该接口电路 1610 可以由任何类型的公知接口标准来实现，以便比如使得系统 1602 能够通过图 1 中的一个或多个链路 24、32、40、42、46 和 48 进行通信。

[0095] 系统 1602 还包括一个或多个用于存储软件和 / 或数据的海量存储设备 1618。这样的海量存储设备的例子包括：软磁盘机、硬磁盘机、光盘驱动器和 DVD（digital versatile disk）驱动器。

[0096] 虽然本文描述了某些方法或装置以及制品，但本专利覆盖的范围并不仅限于此。相反，本专利覆盖所有无论是从字面上还是从等同原则上完全落在本发明所附权利要求的范围内的方法、装置以及制品。

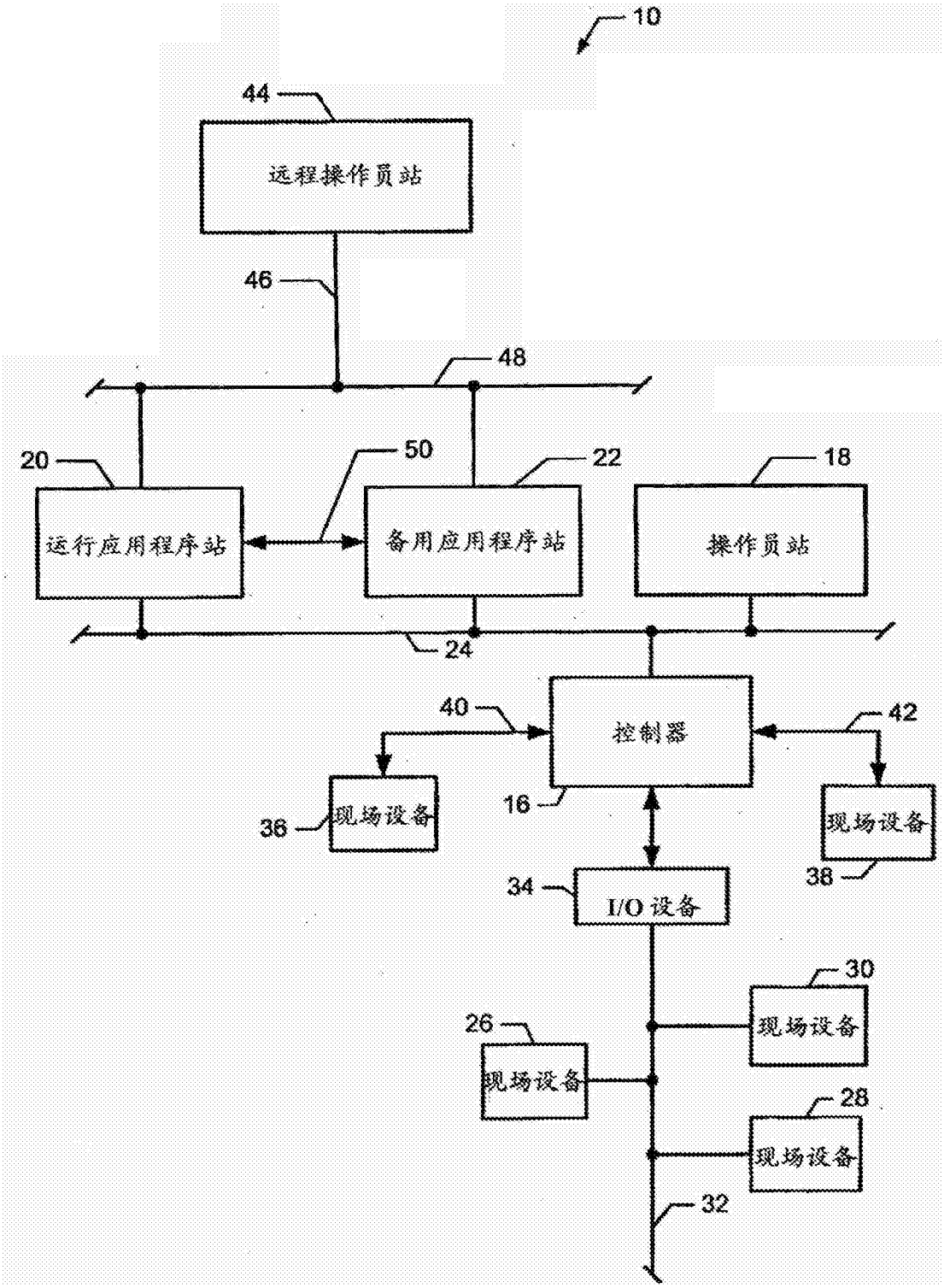

10
44
远程操作员站
46
48
20
50
22
18
运行应用程序站
备用应用程序站
操作员站
24
40
控制器
42
36
现场设备
16
现场设备
38
34
I/O 设备
30
现场设备
26
现场设备
28
现场设备
32

图 1

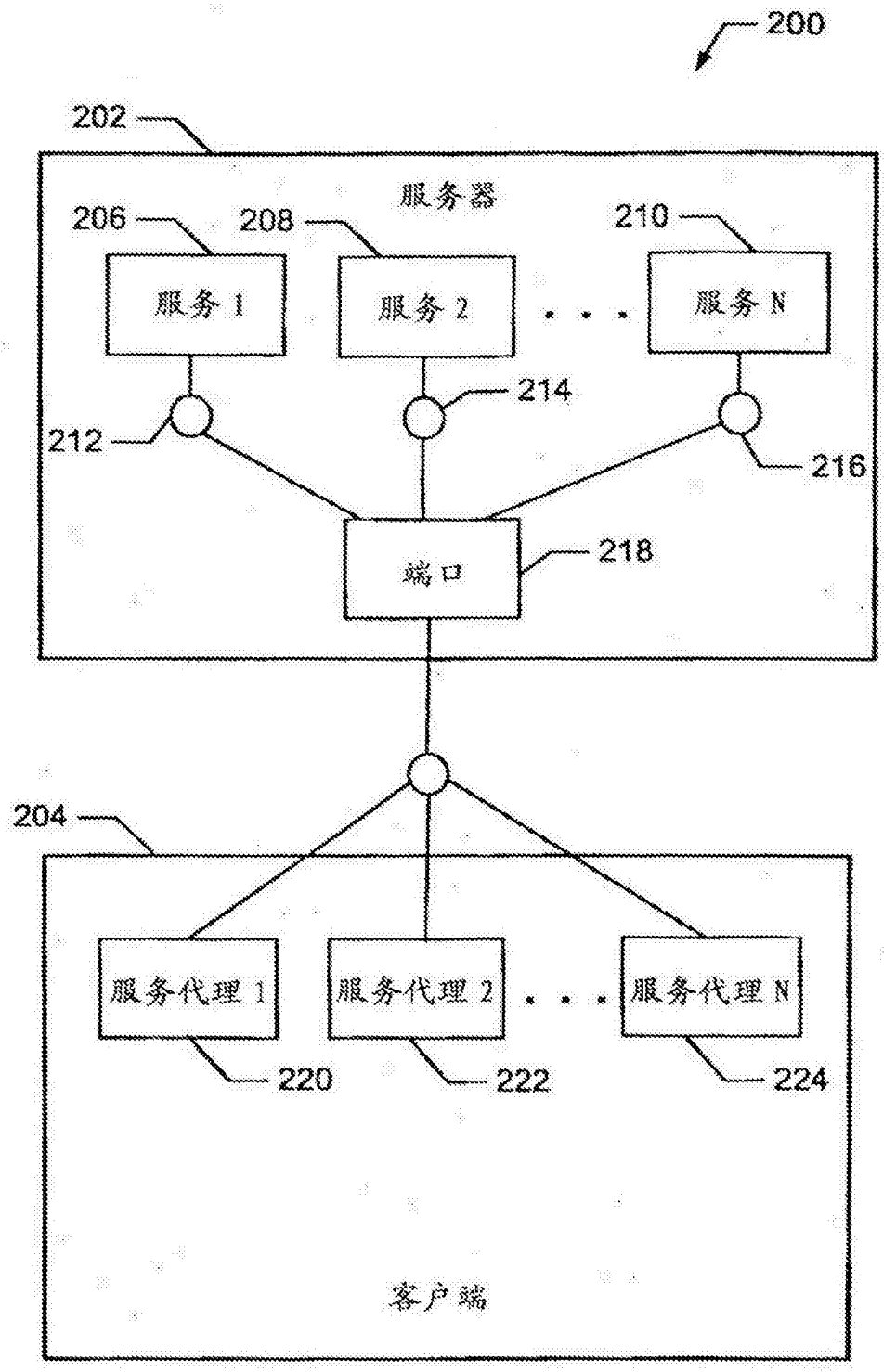
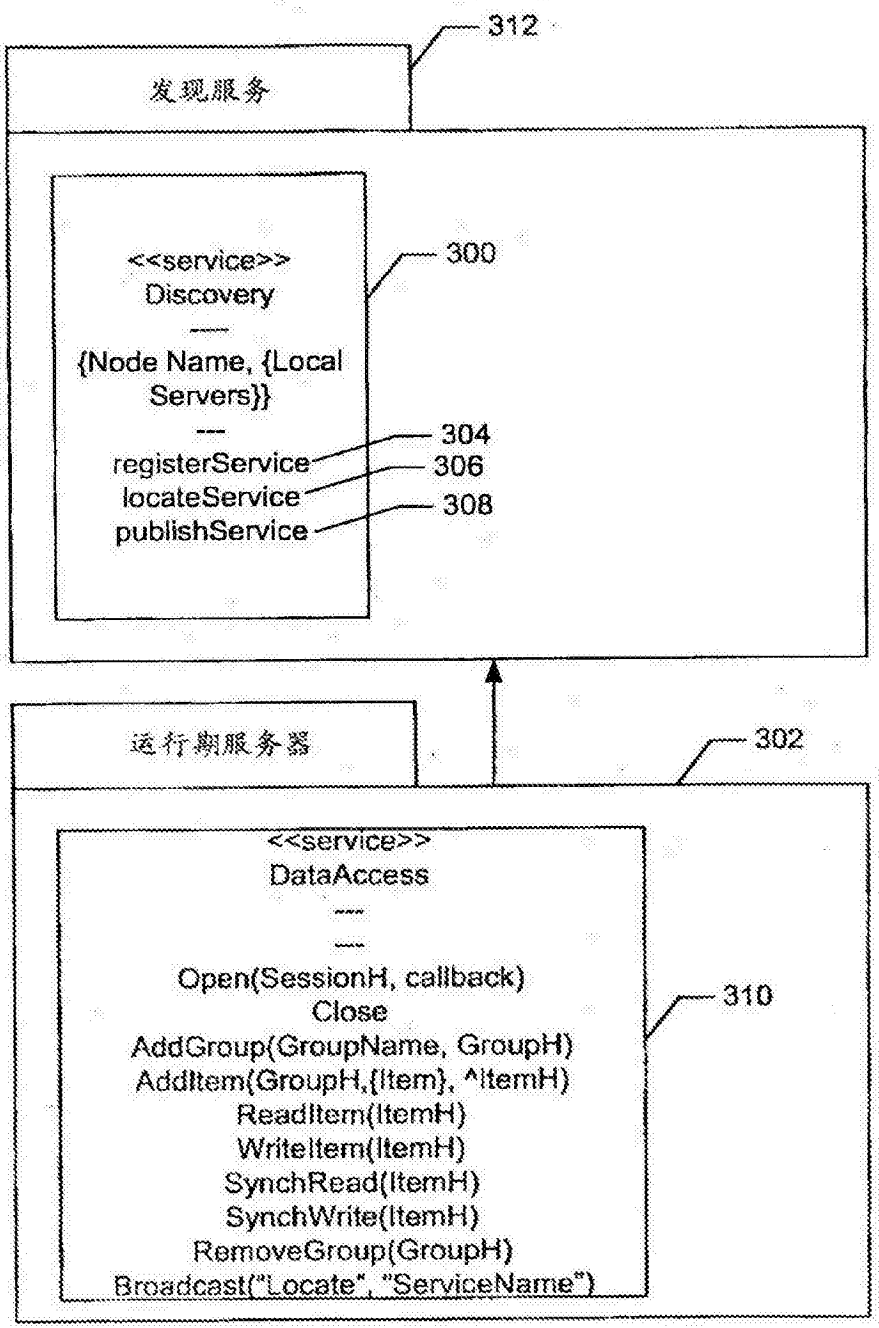

200
202
服务器
206
208
210
服务 1
服务 2
服务 N
214
212
216
端口
218
204
服务代理 1
服务代理 2
服务代理 N
220
222
224
客户端

图 2


312
发现服务
<<service>>
Discovery

{Node Name, {Local Servers}}
---
registerService
locateService
publishService
300
304
306
308
运行期服务器
302
<<service>>
DataAccess

Open(SessionH, callback)
Close
AddGroup(GroupName, GroupH)
AddItem(GroupH,{Item}, ^ItemH)
ReadItem(ItemH)
WriteItem(ItemH)
SynchRead(ItemH)
SynchWrite(ItemH)
RemoveGroup(GroupH)
Broadcast("Locate", "ServiceName")
310

图 3

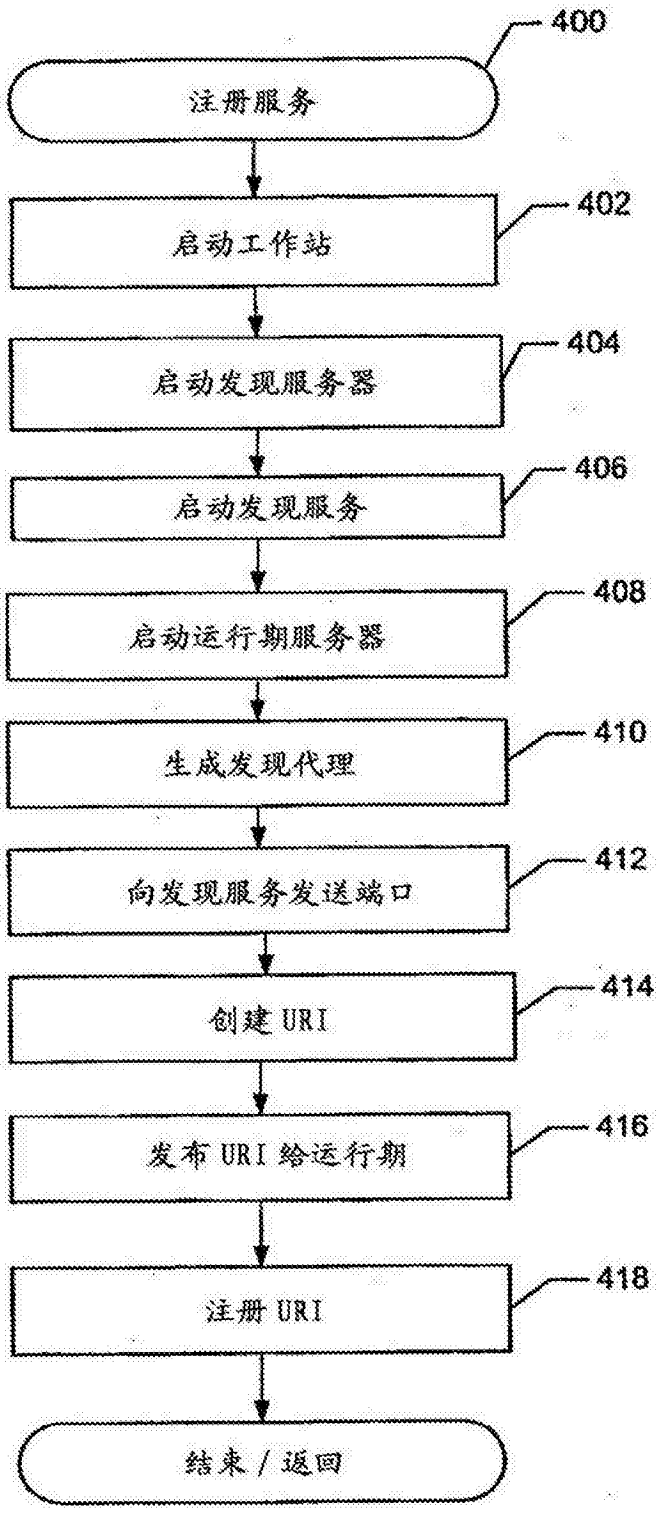

400
注册服务
402
启动工作站
404
启动发现服务器
406
启动发现服务
408
启动运行期服务器
410
生成发现代理
412
向发现服务发送端口
414
创建 URI
416
发布 URI 给运行期
418
注册 URI
结束 / 返回

图 4

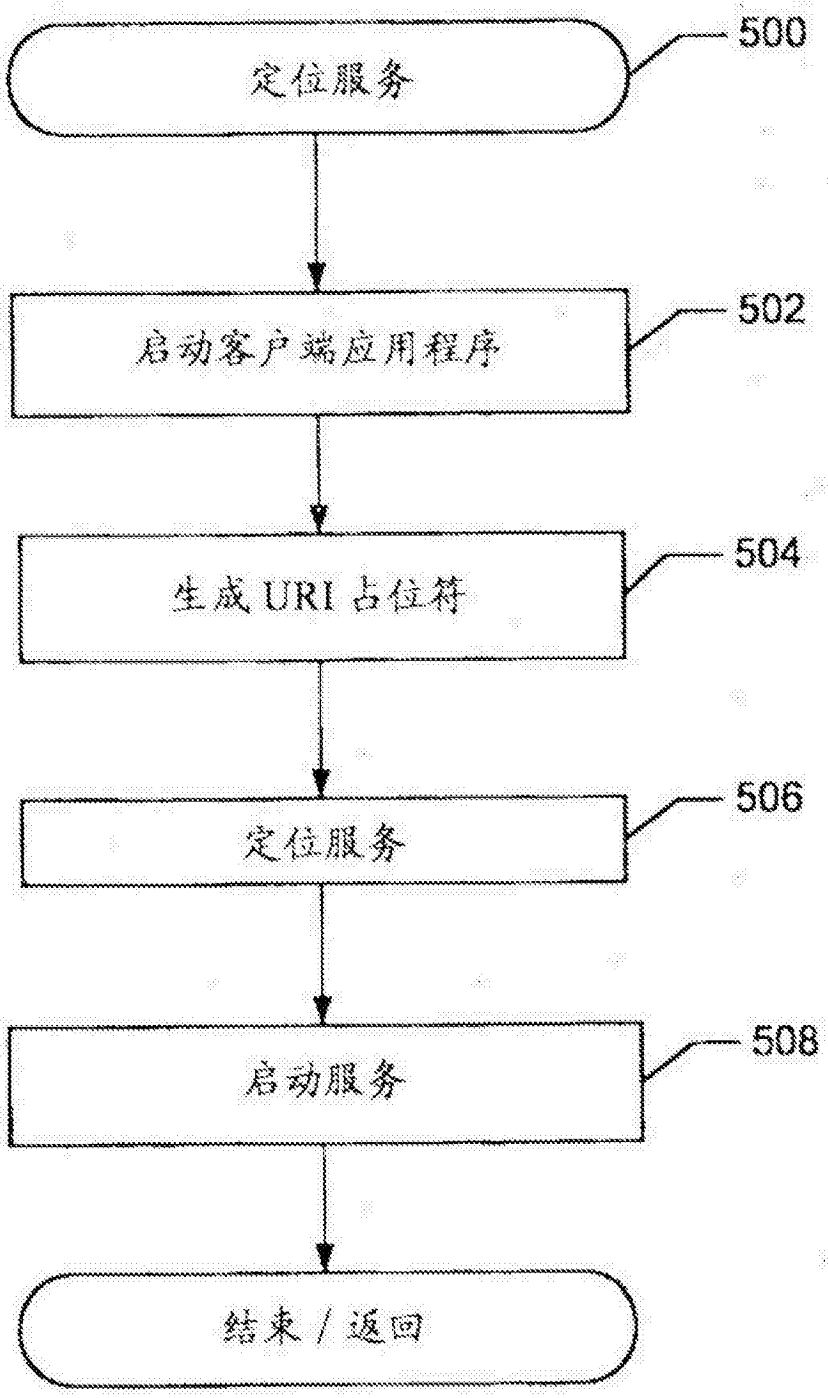


图 5

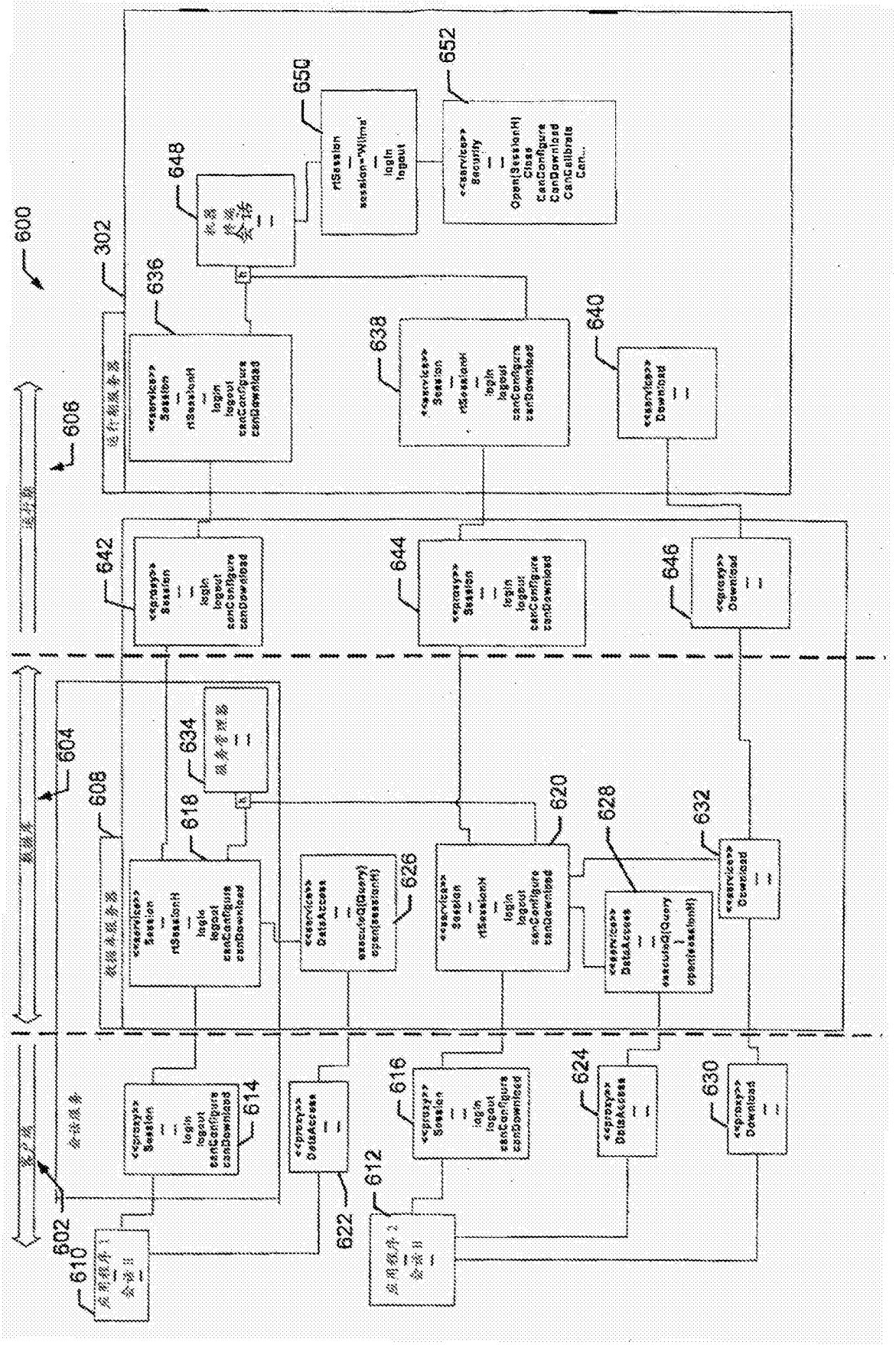

600
602
604
606
608
610
612
614
616
618
620
622
624
626
628
630
632
634
636
638
640
642
644
646
648
650
652
302

图 6

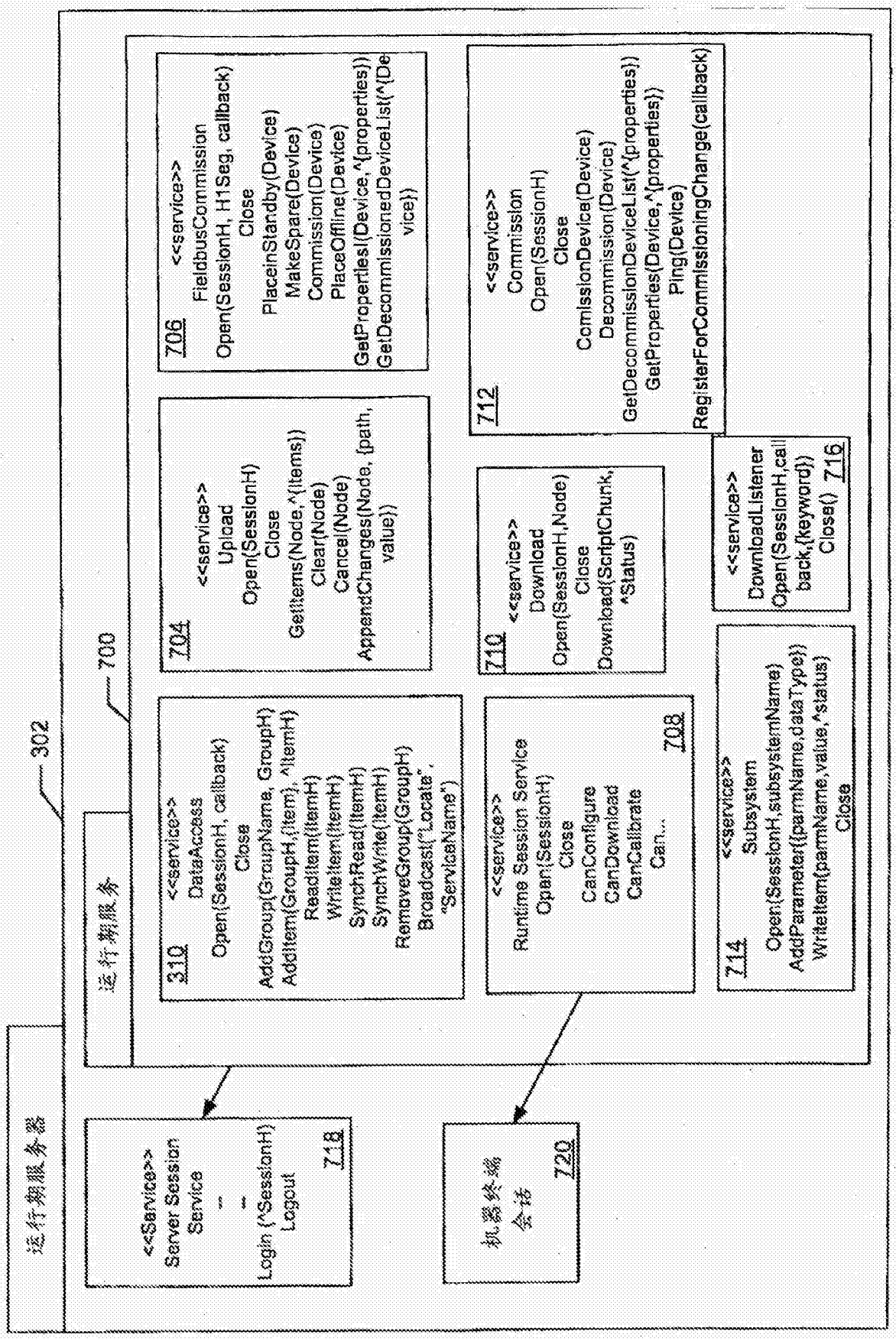

运行期服务器
302
700
运行期服务
<<Service>>
Server Session
Service
--
--
Login (^SessionH)
Logout
718
机器终端
会话
720
310
<<service>>
DataAccess
Open(SessionH, callback)
Close
AddGroup(GroupName, GroupH)
AddItem(GroupH,{Item}, ^ItemH)
ReadItem(ItemH)
WriteItem(ItemH)
SynchRead(ItemH)
SynchWrite(ItemH)
RemoveGroup(GroupH)
Broadcast("Locate",
"ServiceName")
704
<<service>>
Upload
Open(SessionH)
Close
GetItems(Node,^{Items})
Clear(Node)
Cancel(Node)
AppendChanges(Node, {path,
value})
706
<<service>>
FieldbusCommission
Open(SessionH, H1Seg, callback)
Close
PlaceInStandby(Device)
MakeSpare(Device)
Commission(Device)
PlaceOffline(Device)
GetPropertiesI(Device,^{properties})
GetDecommissionedDeviceList(^{De
vice})
708
<<service>>
Runtime Session Service
Open(SessionH)
Close
CanConfigure
CanDownload
CanCalibrate
Can...
710
<<service>>
Download
Open(SessionH,Node)
Close
Download(ScriptChunk,
^Status)
712
<<service>>
Commission
Open(SessionH)
Close
ComissionDevice(Device)
DecommissionDevice(Device)
GetDecommissionDeviceList(^{properties})
GetProperties(Device,^{properties})
Ping(Device)
RegisterForCommissioningChange(callback)
714
<<service>>
Subsystem
Open(SessionH,subsystemName)
AddParameter({parmName,dataType})
WriteItem(parmName,value,^status)
Close
716
<<service>>
DownloadListener
Open(SessionH,call
back,{keyword})
Close()

图 7

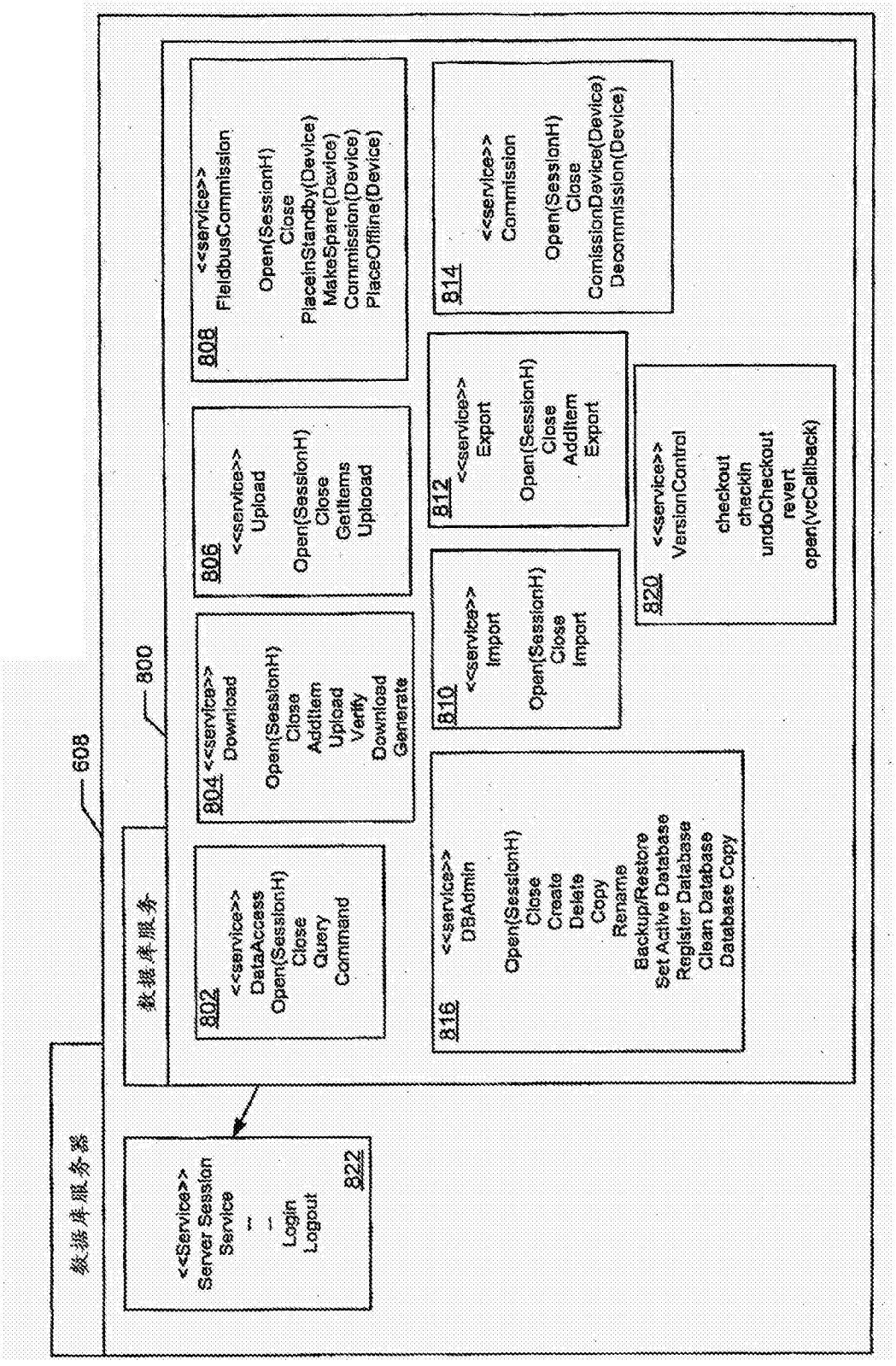

603
数据库服务器
800
数据库服务
802
<<service>>
DataAccess
Open(SessionH)
Close
Query
Command
804
<<service>>
Download
Open(SessionH)
Close
AddItem
Upload
Verify
Download
Generate
806
<<service>>
Upload
Open(SessionH)
Close
GetItems
Upload
808
<<service>>
FieldbusCommission
Open(SessionH)
Close
PlaceinStandby(Device)
MakeSpare(Device)
Commission(Device)
PlaceOffline(Device)
810
<<service>>
Import
Open(SessionH)
Close
Import
812
<<service>>
Export
Open(SessionH)
Close
AddItem
Export
814
<<service>>
Commission
Open(SessionH)
Close
ComissionDevice(Device)
DecommissionDevice)
816
<<service>>
DBAdmin
Open(SessionH)
Close
Create
Delete
Copy
Rename
Backup/Restore
Set Active Database
Register Database
Clean Database
Database Copy
820
<<service>>
VersionControl
checkout
checkin
undoCheckout
revert
open(vcCallback)
<<Service>>
Server Session
Service
Login
Logout
822

图 8

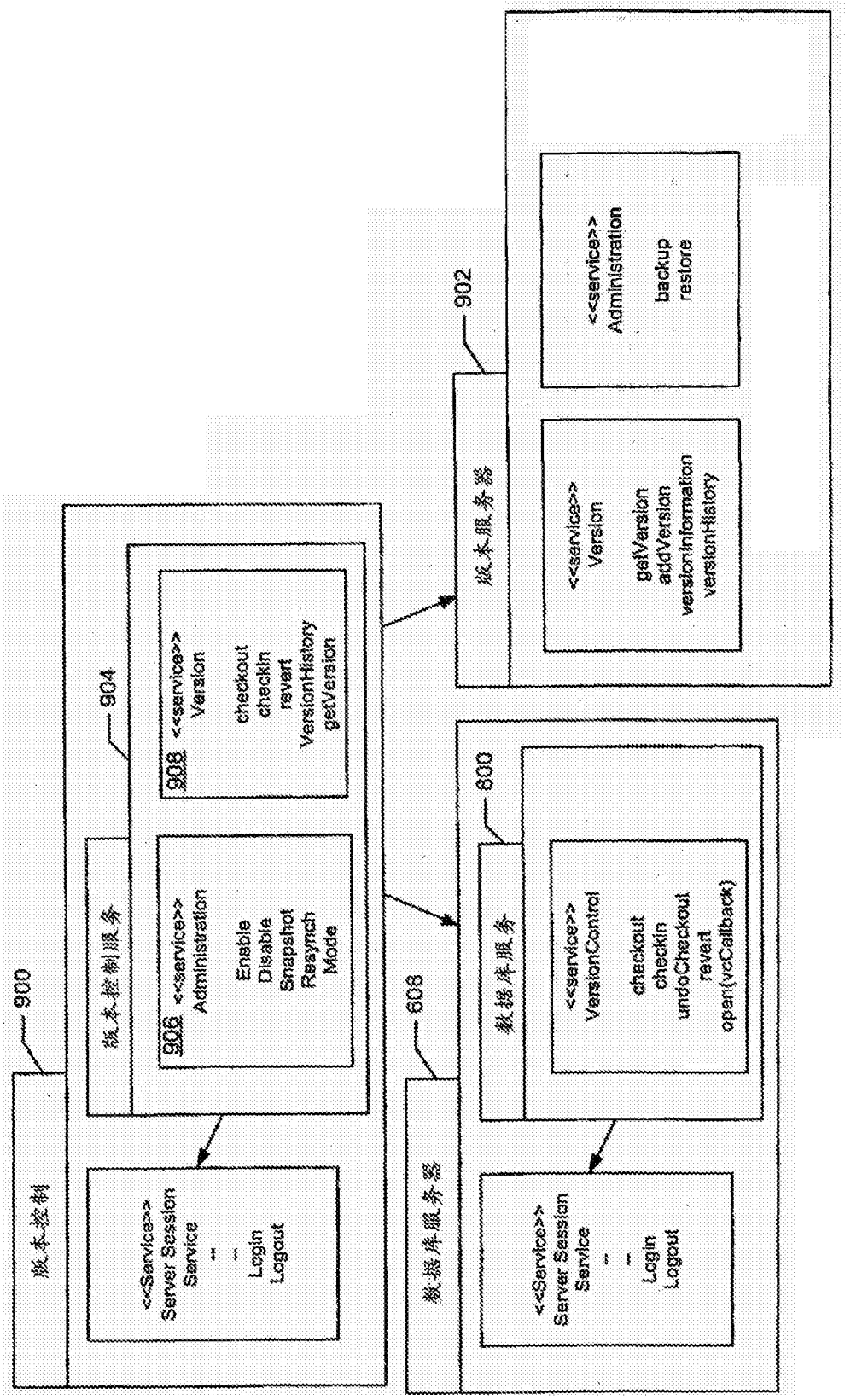

900
版本控制
<<Service>>
Server Session
Service
--
--
Login
Logout
版本控制服务
904
906
<<service>>
Administration
Enable
Disable
Snapshot
Resynch
Mode
908
<<service>>
Version
checkout
checkin
revert
VersionHistory
getVersion
902
版本服务器
<<service>>
Version
getVersion
addVersion
versionInformation
versionHistory
<<service>>
Administration
backup
restore
608
数据库服务器
800
数据库服务
<<service>>
VersionControl
checkout
checkin
undoCheckout
revert
open(vcCallback)
<<Service>>
Server Session
Service
--
--
Login
Logout

图 9

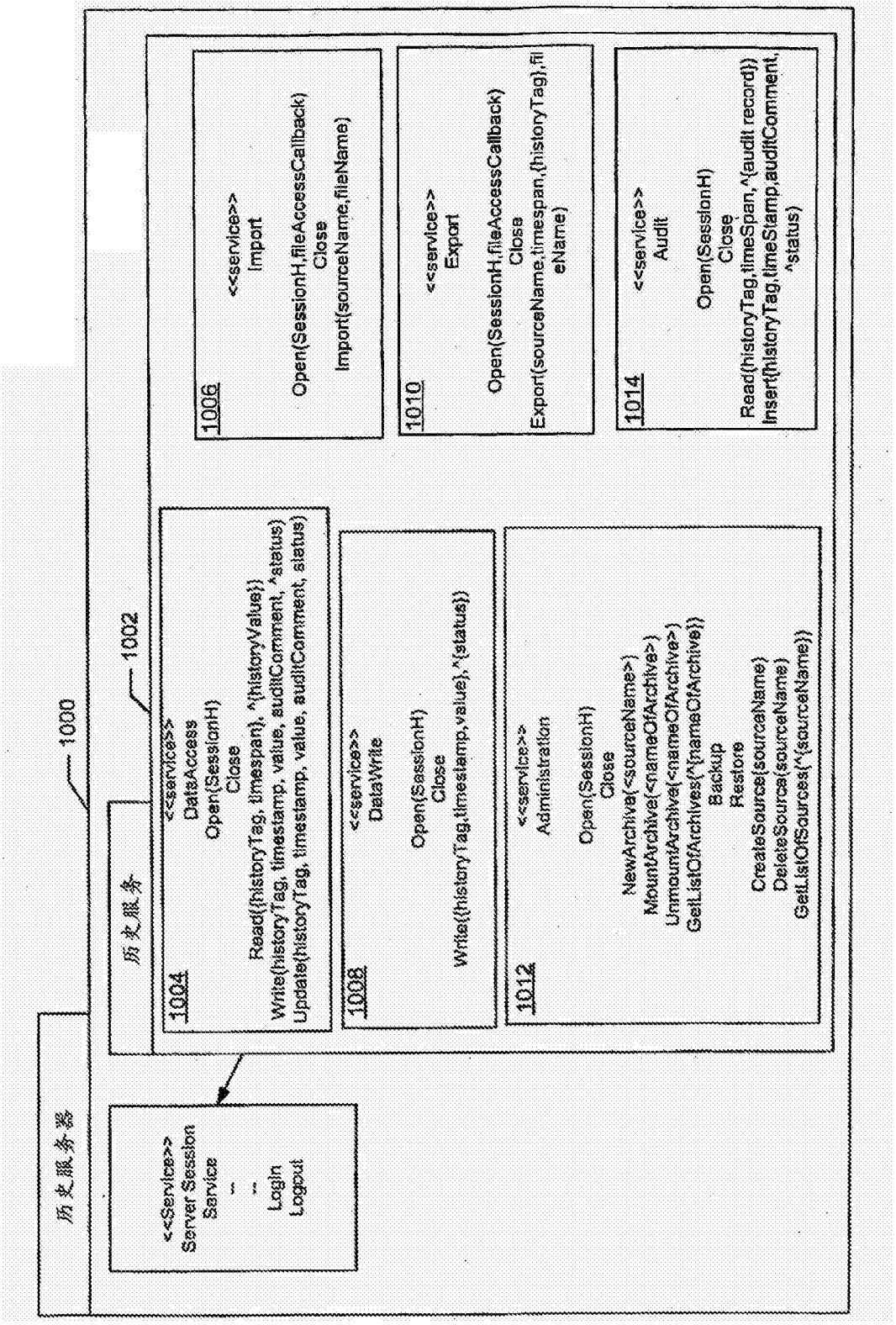

1000
历史服务器
<<Service>>
Server Session
Service
--
--
Login
Logout
1002
历史服务
1004
<<service>>
DataAccess
Open(SessionH)
Close
Read({historyTag, timespan}, ^{historyValue})
Write(historyTag, timestamp, value, auditComment, ^status)
Update(historyTag, timestamp, value, auditComment, status)
1008
<<service>>
DataWrite
Open(SessionH)
Close
Write({historyTag,timestamp,value},^{status})
1012
<<service>>
Administration
Open(SessionH)
Close
NewArchive(<sourceName>)
MountArchive(<nameOfArchive>)
UnmountArchive(<nameOfArchive>)
GetListOfArchives(^{nameOfArchive})
Backup
Restore
CreateSource(sourceName)
DeleteSource(sourceName)
GetListOfSources(^{sourceName})
1006
<<service>>
Import
Open(SessionH,fileAccessCallback)
Close
Import(sourceName,fileName)
1010
<<service>>
Export
Open(SessionH,fileAccessCallback)
Close
Export(sourceName,timespan,{historyTag},fileName)
1014
<<service>>
Audit
Open(SessionH)
Close
Read(historyTag,timeSpan, ^{audit record})
Insert(historyTag,timeStamp,auditComment,
^status)

图 10

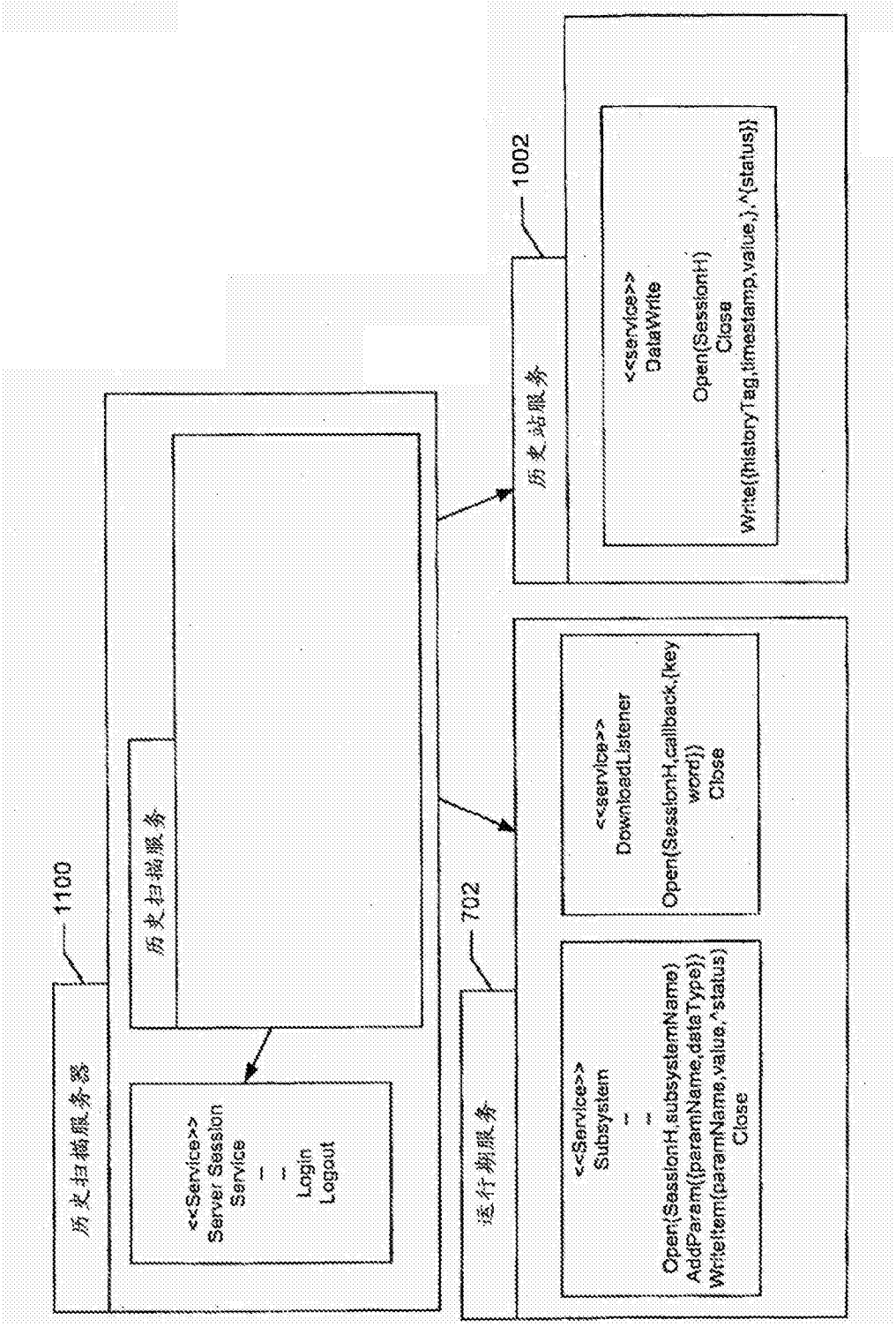

1100
历史扫描服务器
历史扫描服务
<<Service>>
Server Session
Service
~
~
Login
Logout
1002
历史站服务
<<service>>
DataWrite
Open(SessionH)
Close
Write(historyTag,timestamp,value,),^{status}}
702
远行期服务
<<service>>
DownloadListener
Open(SessionH,callback,{key
word})
Close
<<Service>>
Subsystem
~
~
Open(SessionH,subsystemName)
AddParam({paramName,dateType})
WriteItem{paramName,value,^status}
Close

图 11

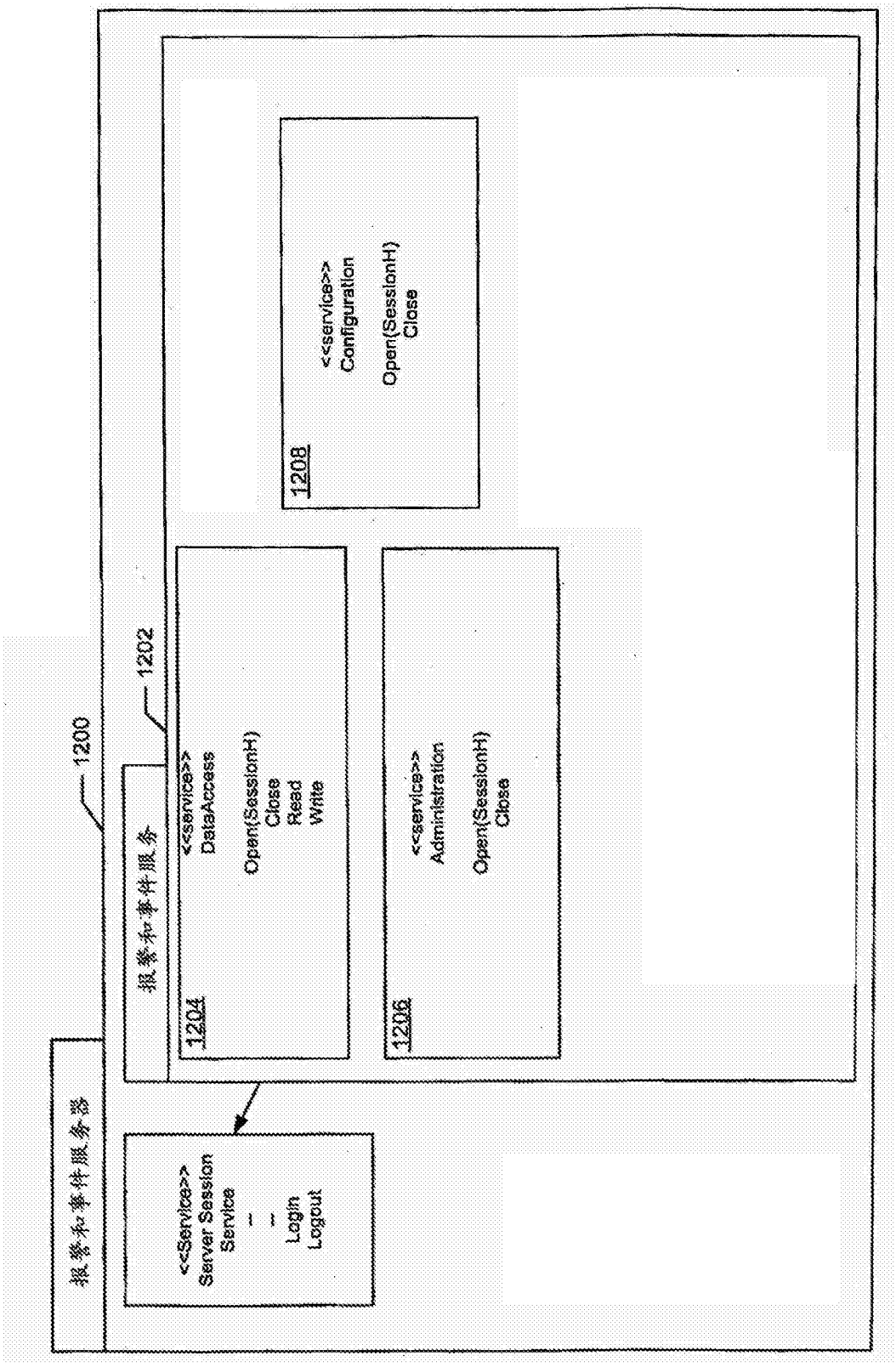

1200
报警和事件服务器
1202
报警和事件服务
1204
<<service>>
DataAccess
Open(SessionH)
Close
Read
Write
1206
<<service>>
Administration
Open(SessionH)
Close
1208
<<service>>
Configuration
Open(SessionH)
Close
<<Service>>
Server Session
Service
Login
Logout

图 12

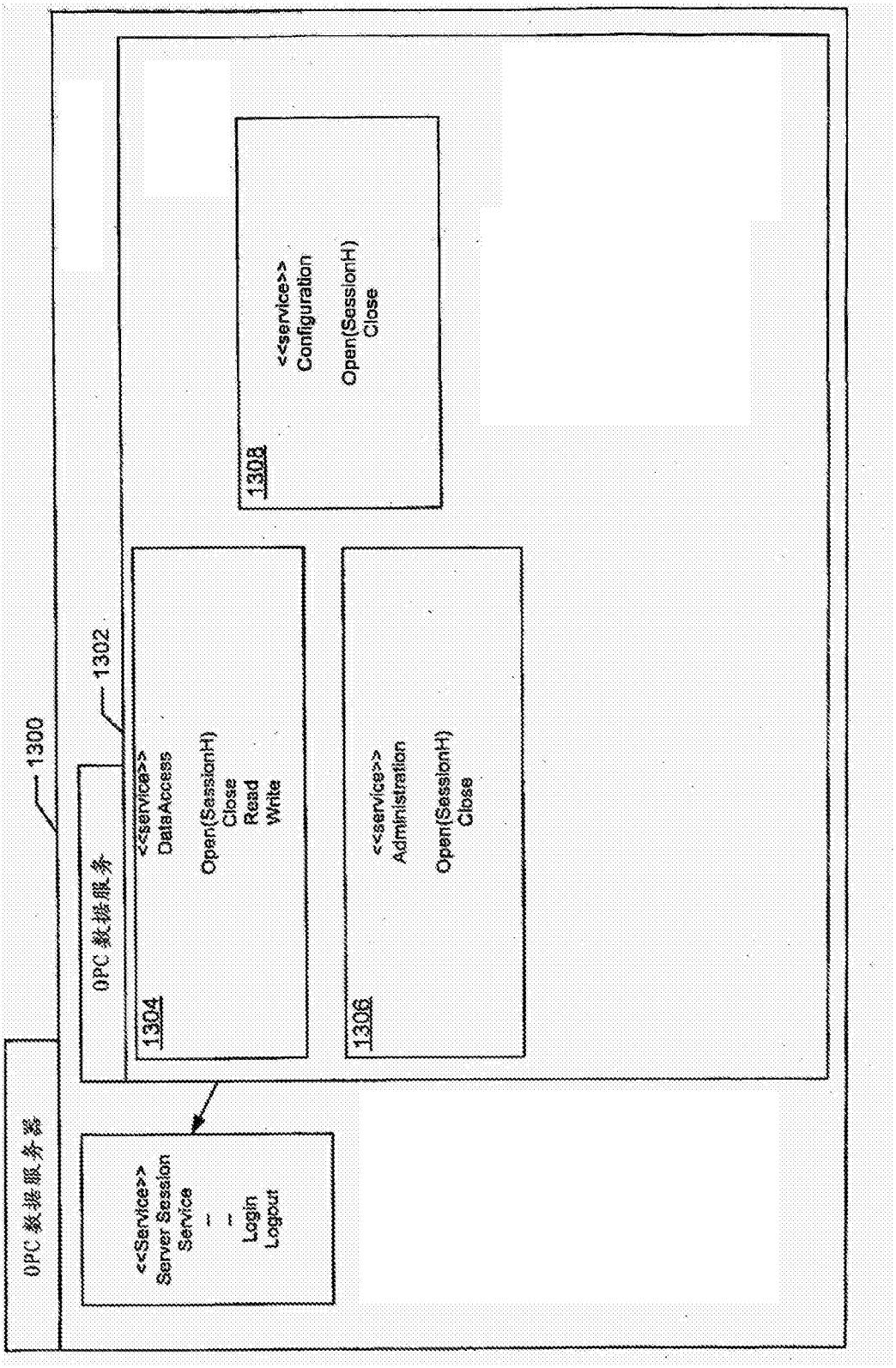

1300
OPC 数据服务器
1302
OPC 数据服务
<<service>>
DataAccess
Open(SessionH)
Close
Read
Write
1304
<<service>>
Administration
Open(SessionH)
Close
1306
<<service>>
Configuration
Open(SessionH)
Close
1308
<<Service>>
Server Session
Service
~
~
Login
Logout

图 13

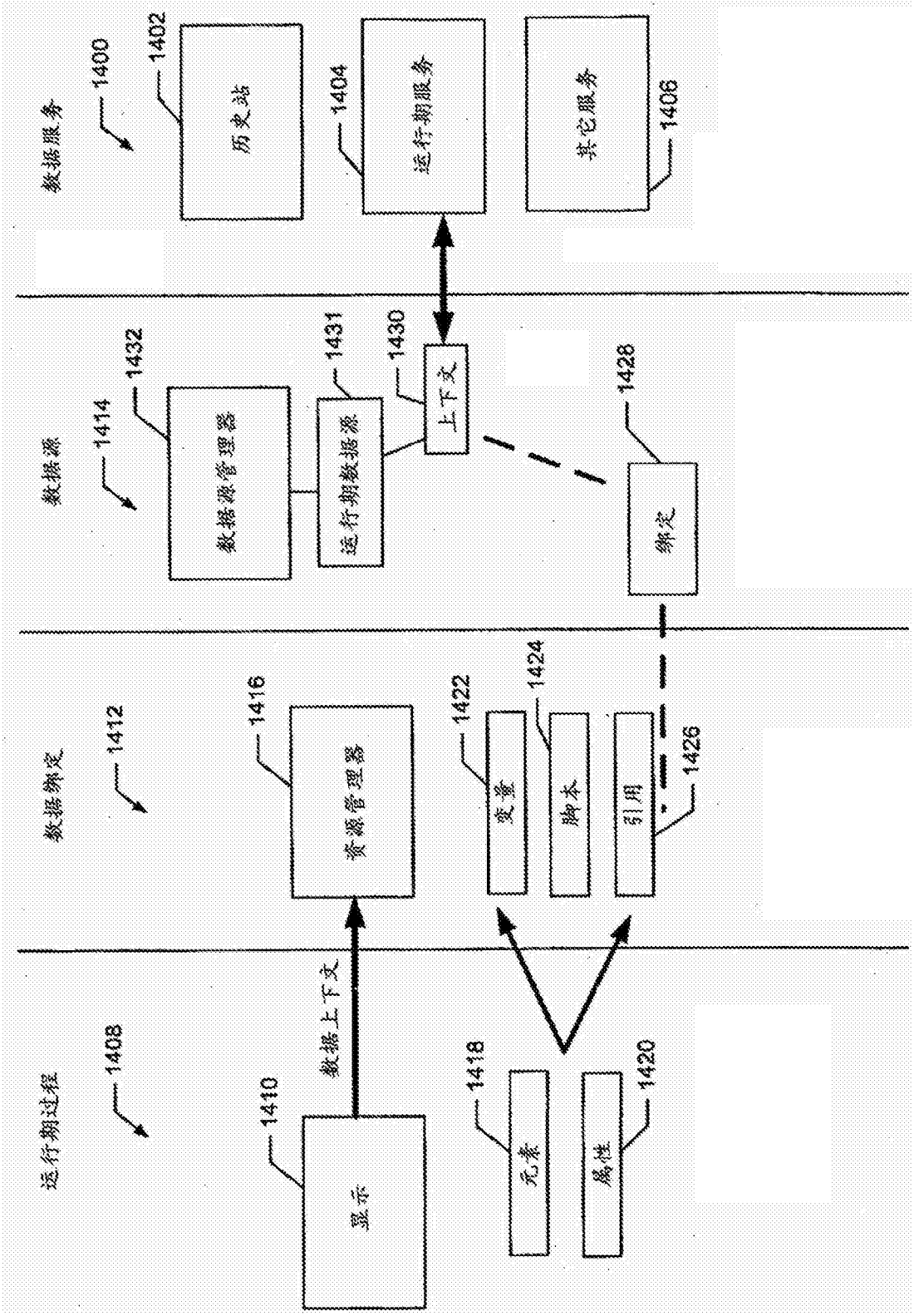

数据服务
1400
1402
历史站
1404
运行期服务
其它服务
1406
数据源
1414
1432
数据源管理器
1431
运行期数据源
1430
上下文
1428
绑定
数据绑定
1412
1416
资源管理器
1422
变量
1424
脚本
引用
1426
运行期过程
1408
数据上下文
1410
显示
1418
元素
属性
1420

图 14

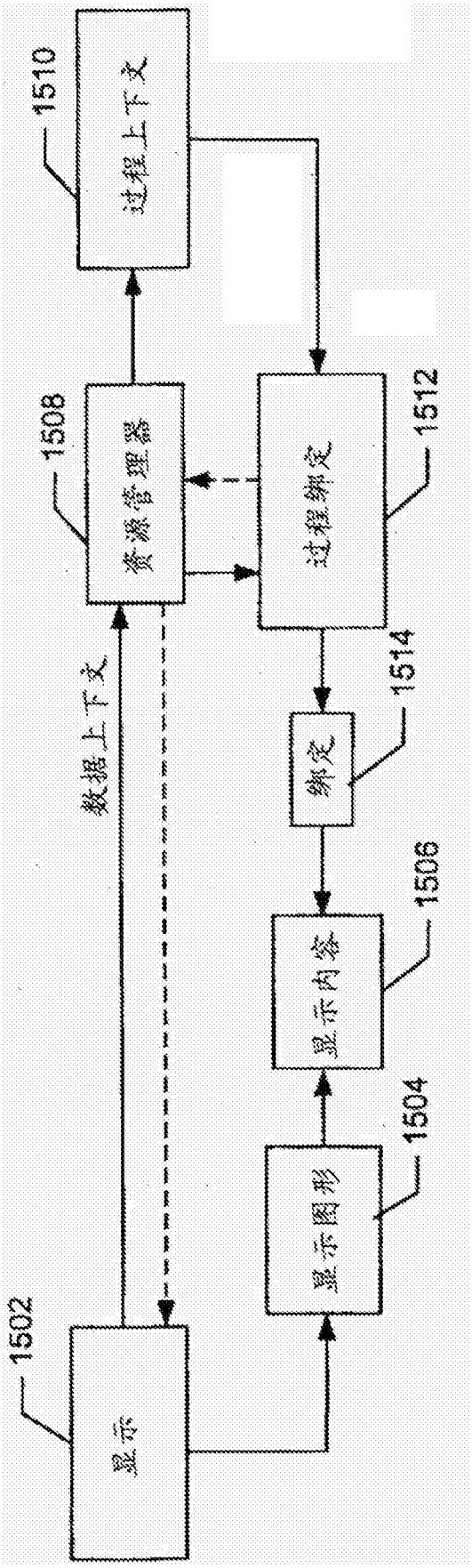

1510
过程上下文
1508
资源管理器
1512
过程绑定
数据上下文
1514
绑定
1506
显示内容
1504
显示图形
1502
显示

图 15

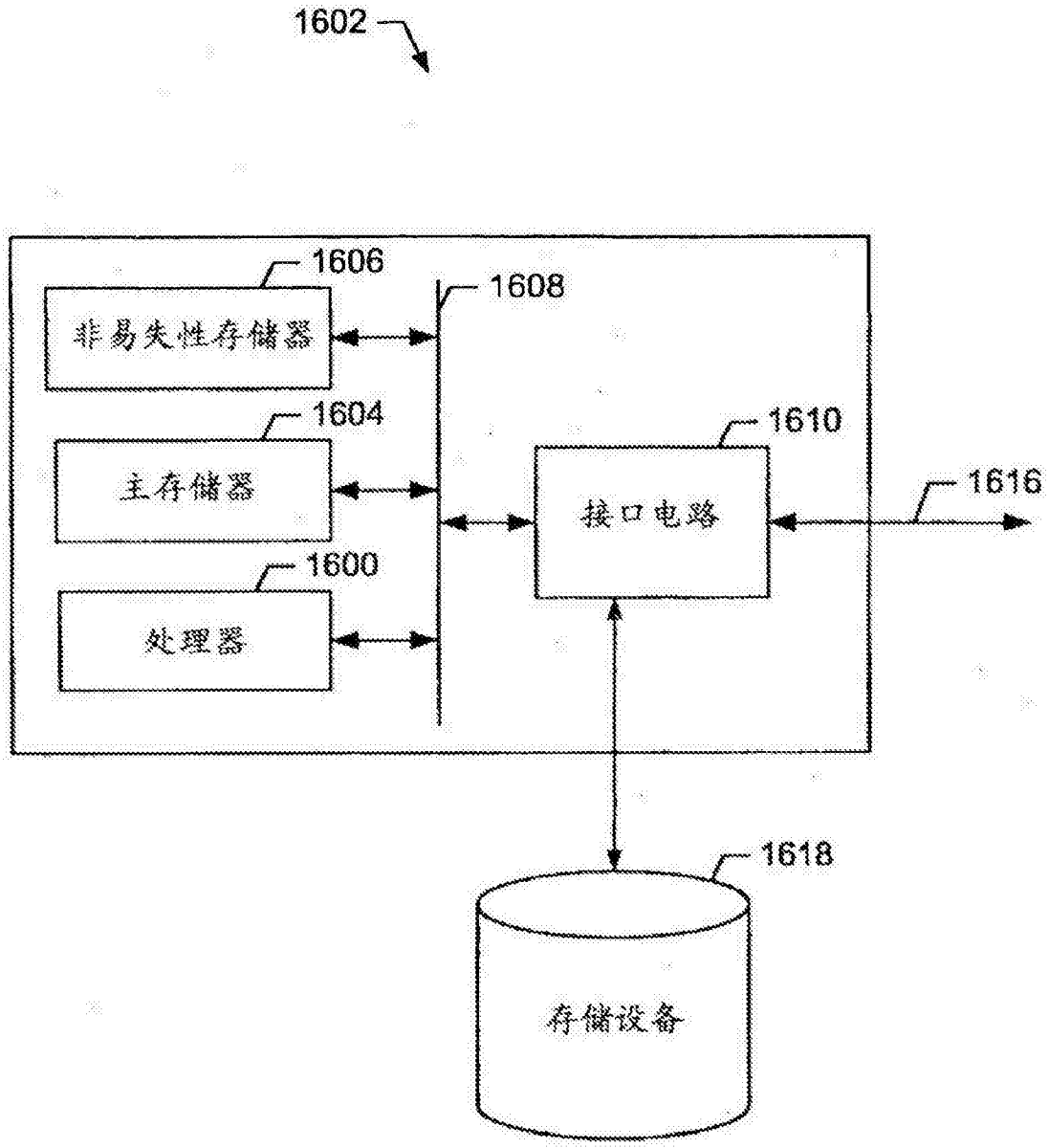

1602
1606
非易失性存储器
1608
1604
主存储器
1610
接口电路
1616
1600
处理器
1618
存储设备

图 16