(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 107924411 B
(45)授权公告日 2023.04.21

(21)申请号 201680047005.7
(22)申请日 2016.05.20
(65)同一申请的已公布的文献号
申请公布号 CN 107924411 A
(43)申请公布日 2018.04.17
(30)优先权数据
62/205,282 2015.08.14 US
15/054,755 2016.02.26 US
(85)PCT国际申请进入国家阶段日
2018.02.09
(86)PCT国际申请的申请数据
PCT/US2016/033422 2016.05.20
(87)PCT国际申请的公布数据
WO2017/030615 EN 2017.02.23
(73)专利权人 甲骨文国际公司
地址 美国加利福尼亚
(72)发明人 C·D·斯特劳布 P·刘 P·苏伯拉玛尼姆
(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
专利代理师 李晓芳
(51)Int.Cl.
G06F 9/451(2018.01)
(56)对比文件
US 2010293080 A1,2010.11.18
JP 2014142755 A,2014.08.07
US 2014040789 A1,2014.02.06
CN 103944784 A,2014.07.23
CN 102957748 A,2013.03.06
US 2005081105 A1,2005.04.14
US 8812546 B1,2014.08.19
US 2014019419 A1,2014.01.16
US 2010257230 A1,2010.10.07
审查员 王彩勤

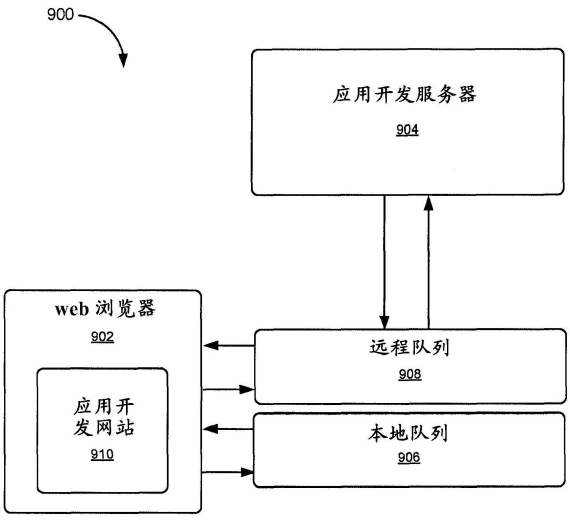
权利要求书3页 说明书43页 附图15页

(54)发明名称

一种事务系统中UI状态的恢复的方法和系统

(57)摘要

系统恢复用户界面("UI")状态。系统接收由与UI交互的用户执行的动作,并且基于动作来确定事务,其中事务被配置为修改与UI对应的模型。系统存储UI的第一UI状态和模型的第一模型状态,然后提交事务。系统随后基于第一用户交互来确定撤销事务。系统然后将UI恢复到第一UI状态并且将模型恢复到第一模型状态。在一个实施例中,第一模型状态在撤销事务之前被恢复,而第一UI状态在撤消事务之后被恢复。



CN 107924411 B

1.一种基于模型-视图-控制器MVC框架的自动保存方法,所述MVC框架包括处理与数据源的交互并运行业务逻辑的模型层、处理应用用户界面UI的视图层、以及管理应用流并充当模型层和视图层之间的接口的控制器,所述自动保存方法包括:

由客户端设备的web浏览器接收由与对应于服务器的网站交互的用户执行的用户动作,其中所述服务器托管UI,其中所述用户动作由所述用户在与所述UI交互时执行,并且其中所述UI包括具有多个第一UI组件的第一UI区域和具有多个第二UI组件的第二UI区域,所述UI同时显示所述第一UI区域和所述第二UI区域;

确定与所述用户动作对应的变更记录,并且基于所述用户动作来确定事务,其中所述事务被配置为修改与所述UI对应的模型,并且被配置为变更所述第一UI区域和所述第二UI区域中的至少一个的可视表示,所述变更被配置为不被记录在所述MVC框架的模型层的模型中,其中所述UI包括所述MVC框架的视图层;

在提交所述事务之前,存储所述UI的第一UI状态和所述模型的第一模型状态,其中所述第一UI状态包括相互独立地存储的所述第一UI区域的状态和所述第二UI区域的状态,并且在被渲染时包括各UI区域的可视表示的状态;

提交所述事务,以修改所述模型并变更所述第一UI区域和所述第二UI区域中的至少一个的可视表示;

基于第一用户交互来确定撤销所述事务;以及

将所述UI恢复到第一UI状态并且将所述模型恢复到第一模型状态,其中所述第一UI区域的状态和所述第二UI区域的状态被相互独立地恢复。

2.如权利要求1所述的方法,其中在撤销所述事务之前恢复第一模型状态,其中在撤销所述事务之后恢复第一UI状态。

3.如权利要求1所述的方法,其中所述UI的第二UI状态和所述模型的第二模型状态在所述事务被提交之后并且在第一用户交互之前被存储。

4.如权利要求3所述的方法,还包括:

基于第二用户交互来确定重做所述事务;以及

将所述UI恢复到第二UI状态,并且将所述模型恢复到第二模型状态。

5.如权利要求4所述的方法,其中在重做所述事务之前恢复第二模型状态,其中在重做所述事务之后恢复第二UI状态。

6.如权利要求1所述的方法,其中所述网站是应用开发网站,并且所述服务器是应用开发服务器。

7.如权利要求1所述的方法,其中所述事务构成多个相关的模型变更,并且所述模型包括页面模式。

8.如权利要求1所述的方法,其中,如果在所述事务的执行期间没有发生错误,那么所述事务被提交。

9.如权利要求1所述的方法,其中所述第一UI区域包括主内容区域,并且所述第二UI区域包括侧边栏,其中所述事务包括所述主内容区域中的变更,其中所述变更引起所述侧边栏中的对应变更。

10.如权利要求1所述的方法,还包括:

将所述变更记录在第一队列中排队,以提交对本地模型的对应变更,

其中第一队列是有序的持久性队列，所述有序的持久性队列维护用于在与所述网站交互中执行撤销和重做操作的变更记录的历史。

11.如权利要求10所述的方法，还包括：

将所述变更记录在与所述服务器通信的第二队列中排队，以在所述服务器处持久化所述变更记录，

其中第二队列是有序队列，在所述有序队列中一次一个地处理变更记录，并且只有当第二队列中的先前变更记录已经被成功地记录在所述服务器上时，每个变更记录才被发送到所述服务器。

12.如权利要求11所述的方法，其中所述变更记录在被成功地传送到所述服务器之后，从第二队列中被移除。

13.如权利要求11所述的方法，其中第二队列通过请求/响应机制与所述服务器通信，在所述请求/响应机制中，第二队列中的每个变更记录被发送到所述服务器，所述服务器又发送回指示所述变更记录是否已在所述服务器处被持久化的对应响应。

14.如权利要求1所述的方法，其中所述变更记录反映由所述网站中的所述用户动作引起的模型变更，并且被添加在构成相关的模型变更的事务中。

15.如权利要求11所述的方法，其中，当所述事务被提交时，所述事务中的变更记录被放在第一队列中，其中如果在所述事务的执行期间没有发生错误，那么所述事务被提交。

16.如权利要求15所述的方法，其中，当所述事务被记录时，所述事务中的变更记录被放在第二队列中，其中当所述事务的变更被应用于对应的客户端侧模型，并且任何必要的用户界面被更新以反映对应的变更时，所述事务被记录。

17.一种其上存储有指令的计算机可读介质，所述指令当由处理器执行时，使所述处理器执行根据权利要求1-16中任一项所述的方法的步骤。

18.一种基于模型-视图-控制器MVC框架的自动保存系统，所述MVC框架包括处理与数据源的交互并运行业务逻辑的模型层、处理应用用户界面UI的视图层、以及管理应用流并充当模型层和视图层之间的接口的控制器，所述自动保存系统包括：

接收模块，所述接收模块通过客户端设备的web浏览器接收由与对应于服务器的网站交互的用户执行的用户动作，其中所述服务器托管UI，其中所述用户动作由所述用户在与所述UI交互时执行，并且其中所述UI包括具有多个第一UI组件的第一UI区域和具有多个第二UI组件的第二UI区域，所述UI同时显示所述第一UI区域和所述第二UI区域；

确定模块，所述确定模块确定与所述用户动作对应的变更记录，并且基于所述用户动作来确定事务，其中所述事务被配置为修改与所述UI对应的模型，并且被配置为变更所述第一UI区域和所述第二UI区域中的至少一个的可视表示，所述变更被配置为不被记录在所述MVC框架的模型层的模型中，其中所述UI包括所述MVC框架的视图层；

存储模块，在所述事务被提交之前，所述存储模块存储所述UI的第一UI状态和所述模型的第一模型状态，其中所述第一UI状态包括相互独立地存储的所述第一UI区域的状态和所述第二UI区域的状态，并且在被渲染时包括各UI区域的可视表示的状态；

提交模块，所述提交模块提交所述事务，以修改所述模型并变更所述第一UI区域和所述第二UI区域中的至少一个的可视表示；

确定模块，所述确定模块基于第一用户交互来确定撤销所述事务；以及

恢复模块,所述恢复模块将所述UI恢复到第一UI状态并且将所述模型恢复到第一模型状态,其中所述第一UI区域的状态和所述第二UI区域的状态被相互独立地恢复。

19.一种用于自动保存的设备,包括:

处理器;和

其上存储有指令的计算机可读介质,所述指令当由所述处理器执行时,使所述处理器执行根据权利要求1-16中任一项所述的方法的步骤。

20.一种包括部件的装置,该部件用于执行根据权利要求1-16中任一项所述的方法的步骤。

一种事务系统中UI状态的恢复的方法和系统

[0001] 对相关申请的交叉引用

[0002] 本申请要求于2015年8月14日提交的、标题为“RESTORATION OF UI STATE IN TRANSACTIONAL SYSTEMS”的美国临时申请No.62/205,282的优先权，该美国临时申请的公开内容通过引用并入本文。

技术领域

[0003] 一个实施例一般而言涉及提供用户界面(“UI”)恢复功能的系统，并且具体地涉及提供UI恢复功能的事务系统。

背景技术

[0004] 一般而言，无处不在的移动服务和无线连接驱动了针对各种个人和商业需要而对于移动设备应用(通常称为“app”)的需求。这种需求又导致期望简化和加快移动应用的开发和修改、同时还允许复杂的应用特征并确保业务安全不被危及的移动应用开发平台/手段。

发明内容

[0005] 一个实施例是恢复用户界面(“UI”)状态的系统。该系统接收由与UI交互的用户执行的动作，并且基于该动作确定事务，其中该事务被配置为修改与UI对应的模型。该系统存储UI的第一UI状态和模型的第一模型状态，并且然后提交事务。该系统随后基于第一用户交互确定撤销事务。该系统然后将UI恢复到第一UI状态并且将模型恢复到第一模型状态。在一个实施例中，第一模型状态在撤销事务之前被恢复，而第一UI状态在撤消事务之后被恢复。

附图说明

[0006] 图1是根据本发明的实施例的、用于开发使用移动云服务的应用的系统的框图。

[0007] 图2是根据本发明的一些实施例的、用于便于移动计算设备和企业计算机系统之间的通信的计算环境的框图。

[0008] 图3A示出了根据本发明的实施例的移动应用主界面(springboard)；图3B和图3C示出了根据本发明的实施例的移动应用用户界面(“UI”)。

[0009] 图4是根据本发明的实施例的移动应用框架运行时架构的框图。

[0010] 图5是根据本发明的实施例的、用于在移动云基础设施中开发移动应用的系统的框图。

[0011] 图6是根据本发明的实施例的、用于构建移动应用的系统中的网络组件的框图。

[0012] 图7是根据本发明的实施例的移动安全套件组件的框图。

[0013] 图8是根据本发明的实施例的移动应用开发的流程图。

[0014] 图9是根据本发明的实施例的用于web应用开发的系统的框图。

[0015] 图10是根据本发明的实施例的自动保存功能的流程图。

[0016] 图11示出了根据一个实施例的示例用户界面(“UI”)。

[0017] 图12示出了根据一些实施例的示例提交、撤消和重做流程。

[0018] 图13是根据本发明的实施例的UI恢复功能的流程图。

具体实施方式

[0019] 应用是指软件程序,该软件程序在执行时执行具体的期望任务。一般而言,在包含一个或多个操作系统(“OS”)、虚拟机(例如,支持Java™编程语言)、设备驱动器等的运行时环境中执行若干应用。开发人员常常使用应用开发框架(“ADF”)(ADF本身是应用)来实现/开发期望的应用。ADF提供了在开发应用时可以直接/间接被使用的预定义的代码/数据模块集合。ADF还可以提供诸如集成开发环境(“IDE”)、代码生成器、调试器等之类的工具。一般而言,ADF通过提供可重用的组件来简化应用开发,这些可重用的组件可以由应用开发人员使用以通过例如选择组件执行期望的任务和定义所选组件的外观、行为和交互来定义用户界面(“UI”)和应用逻辑。一些ADF(诸如来自Oracle公司的“Oracle ADF”)基于模型-视图-控制器(“MVC”)设计模式,该MVC设计模式促进松散耦合以及较容易的应用开发和维护。

[0020] 一般而言,许多公司已经表示需要允许它们的雇员利用来自场外位置的移动设备访问安全企业应用,使得在路上的(on-the-go)雇员可以访问存储在企业计算机系统上的信息。利用这样的能力,销售人员可以在路上工作,服务技术人员可以在客户现场查找部件,雇员可以在家工作,等等。一些公司还想要允许终端客户访问位于企业计算机系统中的数据。这种访问可以通过改善客户体验和降低成本来使公司与竞争者区分开。例如,通过实现这种访问,商店可以允许客户在任何方便的时候针对商品来远程搜索商店库存并且购物,由此改善客户体验并降低对销售人员、操作员和其他工作人员的需求。

[0021] 不同的企业应用供应商传统上通过提供专门门户结合公司拥有的安全移动设备或定制的移动应用来满足这种需要。但是,随着当前各种可用个人移动设备的爆炸,这些传统的解决方案迅速变得过时,因为供应商根本跟不上变得可用的所有最新OS和硬件。

[0022] 此外,依赖于应用所使用的数据的类型和/或应用类型,应用可能需要与不同的企业计算机系统连接和同步。这些企业计算机系统可以由不同的后端计算机系统支持,这些不同的后端计算机系统还可以基于应用类型和数据类型而变化。但是,不同的后端企业系统可以使用不同的通信协议和机制来向设备传送数据,由此使运行各种应用的移动计算设备遇到与支持企业计算机系统的不同后端计算机系统进行通信的挑战。

[0023] 此外,安全性可能成为允许访问企业的内部计算机系统中的关注点。移动计算设备和企业计算机系统之间支持的通信协议的差异可以进一步使用于移动计算设备和企业计算机系统之间的通信的安全访问管理复杂化。例如,可以实现不同的机制来确保应用的认证以访问具有专有安全协议的特定企业计算机系统。一些已知的系统已经尝试通过将现成的消费者移动设备与公司的后端企业系统连接来解决这个问题。这些设备可以被配置为具有通过专用于与企业后端计算机系统通信的特殊门户来连接到企业网络的应用或OS。但是,移动设备的制造商、应用开发人员和企业可以受益于用于开发应用和将移动设备连接到企业后端计算机系统的更灵活和健壮的技术。

[0024] 与已知系统相比,本发明的实施例提供了用于在“云”服务中的快速的业务用户友

好的移动应用整合的基于声明性浏览器的客户端应用开发工具。在一个实施例中，云服务是来自Oracle公司的“移动云服务”(“MCS”)。实施例允许利用对于后端服务使用云服务的预定义模板来构建移动应用，使得可以在应用开发期间向开发人员呈现服务定义，以允许UI设计和后端服务之间的快速连接。

[0025] MCS

[0026] 在使用MCS的实施例中，MCS便于移动计算设备和企业计算机系统之间经由云计算机系统的通信。MCS在移动设备和公司的企业网络之间使用基于第三方云的接口。基于云的接口集中了用于各种企业计算机系统的安全适配器，并且将不同的协议变换(translate)为标准化的表征状态转移(“REST”)架构。公司可以使用本发明的实施例来利用MCS上的可用工具创建它们自己的定制移动应用，并且这些应用可以以原生(native)形式被下载到移动用户设备上。一旦应用被安装，它就可以访问MCS的基于云的接口，以通过由MCS提供的安全适配器到达各种企业计算机系统。

[0027] 对于使用MCS的实施例中的应用开发，MCS在移动后端即服务(“MBaaS”，也称为“BaaS”)模型下提供后端服务。MBaaS允许Web和移动应用开发人员将他们的应用链接到由后端应用暴露的API和后端云存储装置，同时还提供用户管理、推送通知、与社交网络服务的集成，等等。通过使用依据MBaaS模型在MCS中提供的后端服务，实施例提供了被配置为用于由不熟悉编码的非技术用户进行移动应用开发的声明性的基于Web的UI。

[0028] 在一个实施例中，当用户开始开发新应用时，向导被启动，并且请求用户给出新应用的名称和描述。然后，请求用户通过从可以为第一页预先播种(pre-seed)UI的预定义模板集合(例如，选项卡、底部选项卡、分页等)中进行选择来设计应用的第一页。然后，通过指定模板中的细节来完成UI，同时预览被自动更新以显示变化。在完成UI设计时，用户可以使用面板(palette)来浏览通过MCS对于移动应用可用的可用服务和数据源的目录(例如，服务目录)。对于目录的被添加到UI的每个项目，向用户呈现属性列表，并且用户可以利用一个或多个手势(例如，拖放等)将属性绑定到UI元素。用户可以重复特征定义和数据绑定的过程，以创建移动应用。其它UI组件(诸如地图、图形等)还可以添加到UI。当应用准备好进行测试时，用户可以发布应用，使得创建对应的二进制文件(构建用于iOS、Android或任何其它移动设备OS的本机可执行文件)，并且快速响应(“QR”)码随后被生成并被提供给用户。如果用户通过移动设备扫描QR码，则应用通过空中(over the air)安装到移动设备上。

[0029] 实施例在ADF中使用预构建的组件。这些组件提供数据交互、数据可视化和封装的浏览器侧操作，并且简化丰富客户端应用开发。ADF还可以实现诸如Apache Cordova插件之类的插件，以访问设备特征，诸如相机、全球定位系统(“GPS”)、联系人等。

[0030] 在一个实施例中，当ADF接收到构建用于移动设备的应用的请求时，它确定已利用工具包被预编译的一个或多个已开发的应用的部分，并且修改与这些现有应用相关联的声明性信息。然后实施例通过打包代表用于期望的操作系统(“OS”，诸如iOS、Android等)的所请求的应用的二进制工件(artifact)基于现有应用的一个或多个二进制工件和修改的声明性信息来构建所请求的应用。然后，ADF编译所请求的应用，以生成一个或多个二进制工件和定义文件集合。在终端用户开发中，工件是由终端用户在不需要了解编程语言的情况下创建的应用或复杂数据对象。

[0031] 移动安全

[0032] 一些实施例使用由诸如来自Oracle公司的“Oracle移动安全套件”(“OMSS”)之类的移动安全套件提供的安全服务。OMSS是移动设备和移动应用安全解决方案,OMSS提供以雇员为中心的综合企业移动管理(“EMM”)解决方案和以消费者为中心的移动和社交服务。EMM通过无缝地绑定到现有用户身份以及利用用于移动接入的企业后端身份管理基础设施的高级特征来提供移动设备管理(“MDM”)、移动应用管理(“MAM”)、移动内容管理(“MCM”)和移动身份策略。遵照公司需求的安全策略可以被定义以实施完整的设备锁定(通常用于公司拥有的设备)和/或将个人应用与安全的“容器化的”公司应用和数据分离(用于自带设备(“BYOD”)情况)。移动和社交服务提供软件开发工具包(“SDK”),从而允许公司开发人员保护用于iOS和Android设备的定制企业应用、消除(bridge)移动设备、社交网络和企业后端身份管理基础设施之间的差距(gap)。

[0033] OMSS为了应用和内容安全而将安全容器交付到移动设备,以分离、保护和擦除公司应用和数据。移动设备和企业内联网资源之间的所有通信都通过仅可以由移动设备的经审查的(或“容器化的”)应用使用的经认证的传输层安全(“TLS”)/安全套接字层(“SSL”)隧道(“AppTunnel”)。AppTunnel在位于公司非军事区(“DMZ”)的移动安全访问服务器处终止。这个服务器向移动设备提供安全的内联网访问并且仅终止来自安全容器的AppTunnel,由此减少流氓应用的风险以及对设备级VPN的需要。

[0034] 通过利用由ADF提供的内容,实施例提供了基于浏览器的应用开发,其不需要编码并且容易映射到业务服务。实施例还允许内联(inline)(例如,当应用被开发时)预览应用,以及从浏览器编辑、测试和发布应用。相应地,实施例被配置为供商业用户(例如,非技术用户)使用,而不是被配置为由专业开发人员使用的IDE(诸如来自Oracle公司的“Jdeveloper”)。

[0035] 服务目录

[0036] 为了支持使用MCS的本发明的实施例,MCS提供对API目录(诸如来自Oracle公司的“Oracle API目录”(“OAC”))的访问。OAC提供对组织中的可用API的可见性,以使得这些API可以被重用于应用开发。OAC包括用于API资产的简单元模型、利用API填充OAC的自动化、以及让用户搜索用于API的OAC并且理解API的细节以评估API在其应用中的适合性的能力。OAC包括在OAC中创建API资产的采集器(harvester)。在一些实施例中,在项目构建时执行采集。采集器自检(introspect)部署的服务并且创建表示在项目中发现的服务的API资产,其中在项目中发现的服务诸如面向服务的架构(“SOA”)服务和服务总线代理、基于Web服务描述语言(“WSDL”)的Web服务以及基于Web应用描述语言(“WADL”)的REST服务。创建的资产被收集在OAC中。

[0037] 在采集器创建API资产之后,管理者(curator)利用简单的编辑器来编辑API资产,以提供附加的元数据来促进对API的发现和理解。管理者可以改变OAC中的API资产的名称、向OAC中的API资产添加描述、标记关键字或添加文档引用。这种元数据简化了由用户对每个API资产的发现和理解。在编辑API元数据后,管理者通过使API对OAC中的用户可见来发布该API。已发布的资产在OAC控制台中并经由Oracle JDeveloper Oracle Enterprise Repository(企业储存库)插件可用。用户可以搜索OAC,以发现API以及审查由管理者提供的元数据,以了解有关API的更多信息。

[0038] 向每个OAC用户指派角色,该角色确定哪些OAC特征和内容对每个用户可用。OAC中

存在预定义的角色，包括开发人员、管理者和管理员(admin)。具有开发人员角色的用户具有针对已发布的API来搜索OAC、检查API元数据以更好地了解API、声明对API的兴趣以及提交对API的评分和评论的能力。除了开发人员角色可用的能力外，具有管理者角色的用户还可以运行采集器以便在OAC中创建新的API资产、编辑API以更新它们的元数据以及发布它们。除了管理者和开发人员可用的能力外，具有管理员角色的用户还可以访问OAC中的管理页面，以通过编辑系统设置、创建新用户、创建新部门、管理会话以及利用导入/导出工具来管理OAC的基础设施。管理员还可以配置OAC所包括的安全特征。

[0039] 在一些实施例中，应用可以作为本机应用或被托管应用而被开发和部署到移动设备。对于本机应用部署，在设备上安装完整的应用。对于被托管应用的开发，用户需要从“应用商店”下载托管应用，其中这种托管应用“托管”将作为“特征”被安装到托管应用上的被托管应用。这个实施例可以允许从服务器更新正在运行的托管应用，使得声明性元数据可以被发送到设备并且覆盖在现有应用之上，以更新应用以便针对这种新的元数据来运行。

[0040] 图1是用于通过利用允许使用MCS 122作为后端服务的预定义模板来开发应用的系统环境100的框图。在应用开发期间可以向用户呈现服务定义，从而允许UI设计和后端服务之间的快速连接。

[0041] 在所示实施例中，系统环境100包括向一个或多个客户端计算设备104、106和108提供云服务的云基础设施系统102。客户端计算设备104、106和108可以由用户用来与云基础设施系统102交互。客户端计算设备104、106和108可以被配置为操作诸如Web浏览器、专有客户端应用(例如，Oracle Forms)或某个其它应用之类的客户端应用，该客户端应用可以由客户端计算设备的用户用来与云基础设施系统102交互，以使用由云基础设施系统102提供的服务。

[0042] 云基础设施系统102可以具有除了所绘出的组件之外的其它组件。另外，图1中所示的实施例仅仅是可以结合本发明的实施例的云基础设施系统的一个示例。在一些其它实施例中，云基础设施系统102可以具有比图1中示出的更多或更少的组件、可以组合两个或更多个组件或者可以具有不同的组件配置或布置。

[0043] 客户端计算设备104、106和108可以是运行软件(诸如Microsoft Windows **Mobile®**)和/或各种移动OS(诸如iOS、Windows Phone、Android、BlackBerry 10、Palm OS等)并且启用互联网、电子邮件、短消息服务(“SMS”)、**BlackBerry®**或其它通信协议的便携式手持设备(例如，**iPhone®**、蜂窝电话、**iPad®**、计算平板、个人数字助理(“PDA”)或可穿戴设备(例如，Google **Glass®**头戴式显示器)。客户端计算设备104、106和108可以是通用个人计算机，通用个人计算机包括例如运行各种版本的Microsoft **Windows®**、Apple **Macintosh®**和/或Linux OS的个人计算机和/或膝上型计算机。客户端计算设备104、106和108可以是运行各种商用的**UNIX®**或类UNIX OS(包括但不限于各种GNU/Linux OS，诸如Google Chrome OS)中的任何OS的工作站计算机。可替代地或附加地，客户端计算设备104、106和108可以是能够经(一个或多个)网络110进行通信的任何其它电子设备，诸如瘦客户端计算机、启用互联网的游戏系统(例如，具有或不具有**Kinect®**姿势输入设备的Microsoft Xbox游戏控制台)和/或个人消息传送设备。

[0044] 虽然示出了具有三个客户端计算设备的示例性系统环境100，但是可以支持任何数量的客户端计算设备。其它设备(诸如具有传感器的设备等)可以与云基础设施系统102交互。

[0045] (一个或多个)网络110可以便于客户端104、106和108与云基础设施系统102之间的通信和数据交换。(一个或多个)网络110可以是本领域技术人员熟悉的可以利用各种商用协议中的任何商用协议来支持数据通信的任何类型的网络，其中各种商用协议包括但不限于传输控制协议/互联网协议(“TCP/IP”)、系统网络架构(“SNA”)、互联网分组交换(“IPX”)、AppleTalk等。仅仅作为示例，(一个或多个)网络110可以是局域网(“LAN”)，诸如基于以太网、令牌环等的LAN。(一个或多个)网络110可以是广域网和互联网。它可以包括虚拟网络，该虚拟网络包括但不限于虚拟专用网络(“VPN”)、内联网、外联网、公共交换电话网络(“PSTN”)、红外网络、无线网络(例如，依据电气和电子协会(“IEEE”)802.11协议套件、**Bluetooth®**和/或任何其它无线协议中的任何协议操作的网络)；和/或这些网络和/或其它网络的任何组合。

[0046] 云基础设施系统102可以包括一个或多个计算机和/或服务器。这些计算机系统或服务器可以包括一个或多个通用计算机、专用服务器计算机(包括例如个人计算机(“PC”)服务器、**UNIX®**服务器、中型服务器、大型机计算机、机架安装式服务器等)、服务器场、服务器集群或任何其它适当的布置和/或组合。在各种实施例中，与云基础设施系统102相关联的一个或多个计算机系统或服务器可以适于运行在前述公开内容中描述的一个或多个服务或软件应用。例如，与云基础设施系统102相关联的一个或多个计算机系统或服务器可以对应于用于执行根据本公开的实施例的本文描述的处理的服务器。

[0047] 与云基础设施系统102相关联的一个或多个计算机系统或服务器可以运行包括上面讨论的OS中的任何OS以及任何商用服务器OS的OS。与云基础设施系统102相关联的一个或多个计算机系统或服务器还可以运行各种附加的服务器应用和/或中间层应用中的任何服务器应用和/或中间层应用，这些附加的服务器应用和/或中间层应用包括超文本传输协议(“HTTP”)服务器、文件传输协议(“FTP”)服务器、公共网关接口(“CGI”)服务器、**JAVA®**服务器、数据库服务器等。

[0048] 在某些实施例中，由云基础设施系统102提供的服务可以包括按需对云基础设施系统102的用户可用的许多服务，诸如在线数据存储和备份解决方案、基于Web的电子邮件服务、托管的办公套件和文档协作服务、数据库处理、受管理的技术支持服务等。由云基础设施系统102提供的服务可以动态缩放，以满足其用户的需要。由云基础设施系统102提供的服务的具体实例化在本文中被称为“服务实例”。一般而言，从云服务提供商的系统经由通信网络(诸如互联网)对用户可用的任何服务被称为“云服务”。通常，在公共云环境中，构成云服务提供商的系统的服务器和系统不同于客户自己的内部部署的(on-premises)服务器和系统。例如，云服务提供商的系统可以托管应用，并且用户可以经由诸如互联网之类的通信网络按需订购和使用应用。

[0049] 在一些示例中，由云基础设施102实例化的服务实例可以包括对存储装置、被托管的数据库、被托管的Web服务器、软件应用的受保护的计算机网络访问，或由云供应商提供给用户的其它服务，或者如以其它方式在本领域中已知的服务。例如，由云基础设施102实

例化的服务实例可以包括通过互联网对云上的远程存储装置的受密码保护的访问。作为另一个示例，由云基础设施102实例化的服务实例可以包括供联网开发人员私人使用的基于Web服务的被托管的关系数据库和脚本语言中间件引擎。作为另一个示例，由云基础设施102实例化的服务实例可以包括对在云供应商的网站上托管的电子邮件软件应用的访问。

[0050] 在某些实施例中，云基础设施系统102可以包括以自助服务、基于订阅、弹性可扩展、可靠、高度可用和安全的方式交付给客户的一套应用、中间件、开发服务和数据库服务供应物。如在云基础设施服务102中体现的这种云基础设施系统的示例是来自Oracle公司的“Oracle公共云”。

[0051] 云基础设施系统102可以经由不同的部署模型提供云服务。例如，可以在公共云模型下提供服务，在该公共云模型中云基础设施系统102由销售云服务的组织拥有(例如，由Oracle公司拥有)，并且服务对一般公众或不同行业企业可用。作为另一个示例，可以在私有云模型下提供服务，在该私有云模型中云基础设施系统102仅针对单个组织操作，并且可以为该组织内的一个或多个实体提供服务。云服务还可以在社区云模型下被提供，在该社区云模型中云基础设施系统102和由云基础设施系统102提供的服务由相关社区中的若干组织共享。云服务还可以在混合云模型下被提供，该混合云模型是两个或更多个不同模型的组合。

[0052] 在一些实施例中，由云基础设施系统102提供的服务可以包括在软件即服务(“SaaS”)类别、平台即服务(“PaaS”)类别、基础设施即服务(“IaaS”)类别、MBaaS类别或其它服务类别(包括混合服务)下提供的一个或多个服务。在一些实施例中，由云基础设施系统102提供的服务可以包括但不限于应用服务、平台服务、基础设施服务、后端服务等。在一些示例中，应用服务可以由云基础设施系统102经由SaaS平台提供。SaaS平台可以被配置为提供落入SaaS类别下的云服务。例如，SaaS平台可以提供在集成的开发和部署平台上构建和交付按需应用套件的能力。SaaS平台可以管理和控制用于提供SaaS服务的底层软件和基础设施。通过利用由SaaS平台提供的服务，客户可以利用在云基础设施系统上执行的应用。客户可以获取应用服务，而无需客户购买分开的许可和支持。可以提供各种不同的SaaS服务。示例包括但不限于为大型组织提供用于销售业绩管理、企业集成和业务灵活性的解决方案的服务。

[0053] 在一些实施例中，平台服务可以由云基础设施系统102经由PaaS平台提供。PaaS平台可以被配置为提供落入PaaS类别下的云服务。平台服务的示例可以包括但不限于使得组织(诸如Oracle)能够在共享的公共架构上整合现有应用的服务，以及利用由平台提供的共享服务来构建新应用的能力。PaaS平台可以管理和控制用于提供PaaS服务的底层软件和基础设施。客户可以获取由云基础设施系统102提供的PaaS服务，而无需客户购买分开的许可和支持。平台服务的示例包括但不限于来自Oracle公司的“Oracle Java云服务”(Oracle Java Cloud Service，JCS)、来自Oracle公司的“Oracle数据库云服务”(Oracle Database Cloud Service，DBCS)以及其它服务。

[0054] 通过利用由PaaS平台提供的服务，客户可以采用由云基础设施系统102支持的编程语言和工具，并且还可以控制所部署的服务。在一些实施例中，由云基础设施系统102提供的平台服务可以包括数据库云服务、中间件云服务(例如，Oracle Fusion Middleware服务)和Java云服务。在一个实施例中，数据库云服务可以支持共享服务部署模型，这些共享

服务部署模型使得组织能够汇集数据库资源并且以数据库云的形式向客户提供数据库即服务。在云基础设施系统中，中间件云服务可以为客户提供开发和部署各种业务应用的平台，而Java云服务可以为客户提供部署Java应用的平台。

[0055] 各种不同的基础设施服务可以由云基础设施系统102中的IaaS平台提供。基础设施服务便于利用由SaaS平台和PaaS平台提供的服务的客户对底层计算资源(诸如存储装置、网络和其它基本计算资源)的管理和控制。

[0056] 在某些实施例中，云基础设施系统102可以在云基础设施系统中提供对云服务(例如，SaaS、PaaS、IaaS和MBaaS服务)的综合管理。在一个实施例中，云管理功能可以包括用于供应、管理和跟踪由云基础设施系统102接收的客户订阅的能力等。在各种实施例中，云基础设施系统102可以适于自动供应、管理和跟踪客户对由云基础设施系统102提供的服务的订阅。客户经由订阅订单可以订购由云基础设施系统102提供的一个或多个服务。然后云基础设施系统102执行处理，以提供客户的订阅订单中的服务。

[0057] 在一个实施例中，云管理功能可以由一个或多个模块(诸如订单管理和监视模块112)提供。这些模块可以包括一个或多个计算机和/或服务器或者利用一个或多个计算机和/或服务器被提供，该一个或多个计算机和/或服务器可以是通用计算机、专用服务器计算机、服务器场、服务器集群或任何其它适当的布置和/或组合。

[0058] 在示例性操作中，利用客户端计算设备104、106或108的客户可以通过请求由云基础设施系统102提供的一个或多个服务来与云基础设施系统102交互。客户可以利用各种手段向云基础设施系统102发布服务请求134。服务请求134可以包括对由云基础设施系统102提供的一个或多个服务的订阅下订单、访问由云基础设施系统102提供的一个或多个服务等。在某些实施例中，客户可以访问云UI 132、134、136并且经由这些UI下订阅订单。由云基础设施系统102响应于客户下订单而接收的订单信息可以包括识别客户和客户打算订阅的、由云基础设施系统102提供的一个或多个服务的信息。在客户已经下订单之后，经由云UI 132、134和/或136接收订单信息。

[0059] 在这个示例中，订单管理和监视模块112将从客户接收的信息发送到订单数据库，以使客户下的订单被存储。订单数据库可以是由云基础设施系统102操作并且与其它系统元件结合操作的若干数据库中的一个数据库。订单管理和监视模块112可以将包括存储在订单数据库中的订单信息的全部或部分的信息转发到订单管理模块。在一些实例中，订单管理模块可以被配置为执行与订单相关的计费和记帐功能，诸如验证订单，以及在通过验证后，预订订单。

[0060] 在某些实施例中，云基础设施系统100可以包括身份管理模块114。身份管理模块114可以被配置为提供身份服务，诸如云基础设施系统102中的访问管理和授权服务。在一些实施例中，身份管理模块114可以控制关于希望利用由云基础设施系统102提供的服务的客户的信息。这种信息可以包括认证这些客户的身份的信息和描述这些客户被授权相对于各种系统资源(例如，文件、目录、应用、通信端口、存储器段等)执行哪些动作的信息。身份管理模块114还可以包括对关于每个客户的描述性信息以及关于可以如何和由谁来访问和修改该描述性信息的管理。

[0061] 在某些实施例中，云基础设施系统102还可以包括用于提供被用来向云基础设施系统102的客户提供各种服务的资源的基础设施资源116。在一个实施例中，基础设施资源

116可以包括诸如服务器、存储装置和网络资源之类的硬件的预先集成和优化的组合，以执行由PaaS平台和SaaS平台提供的服务。

[0062] 在一些实施例中，云基础设施系统102中的资源可以由多个用户共享并且根据需求动态地重新分配。此外，资源可以被分配给不同时区中的用户。例如，云基础设施系统102可以使第一时区中的第一组用户能够利用云基础设施系统的资源指定的小时数，并且然后使相同的资源能够被重新分配给位于不同时区中的另一组用户，从而最大化资源的利用率。

[0063] 在某些实施例中，可以提供由云基础设施系统102的不同组件或模块以及由云基础设施系统102提供的服务共享的多个内部共享服务118。这些内部共享服务118可以包括但不限于安全和身份服务、集成服务、企业储存库服务、企业管理器服务、病毒扫描和白名单服务、高可用性、备份和恢复服务、用于启用云支持的服务、电子邮件服务、通知服务、文件传输服务等。

[0064] 在某些实施例中，可以提供由云基础设施系统102的不同组件或模块以及由云基础设施系统102提供的服务共享的多个外部共享服务120。这些外部共享服务120可以包括但不限于安全和身份服务、集成服务、企业储存库服务、企业管理器服务、病毒扫描和白名单服务、高可用性、备份和恢复服务、用于启用云支持的服务、电子邮件服务、通知服务、文件传输服务等。

[0065] 在各种实施例中，外部共享服务120可以包括向(一个或多个)企业计算机系统126提供访问、数据转换、自动化等的一个或多个组件。对(一个或多个)企业计算机系统126的访问可以由云基础设施系统102的不同组件或模块以及由云基础设施系统102所提供的服务共享。在一些实施例中，对(一个或多个)企业计算机系统126的访问可以由云基础设施系统102所提供的、受限于一个或多个订户的服务实例共享。

[0066] 在进一步的实施例中，外部共享服务120可以包括由云基础设施系统102的不同组件或模块以及由云基础设施系统102所提供的服务共享的外部应用编程接口(“API”)服务128。这些外部API服务128可以包括但不限于由其它第三方服务或实体提供的API。

[0067] 各种不同的移动云服务可以由云基础设施系统102中的MCS 122提供。根据本发明的一些实施例，MCS 122便于移动计算设备和企业计算机系统(例如，企业计算机系统126)之间的通信。MCS 122可以包括用来存储企业数据和认证信息的一个或多个存储器存储设备(“本地存储装置”)。企业数据可以从企业计算机系统126或从客户端计算设备104、106或108接收，或者可以包括由云基础设施系统102转换的企业数据，或者是这二者的组合。认证信息可以从身份管理系统114接收和/或由云基础设施系统102生成。在一些实施例中，认证信息可以包括指示用户的关于对服务的请求的安全认证的信息。

[0068] 企业计算机系统(诸如企业计算机系统126)可以在与云基础设施系统102不同的地理位置(例如，远程地理位置)处物理地位于云基础设施系统102的防火墙之外。在一些实施例中，企业计算机系统126可以包括一个或多个不同的计算机或服务器。在一些实施例中，企业计算机系统126可以是单个计算机系统的一部分。

[0069] 在某些实施例中，企业计算机系统126可以利用一个或多个不同的协议与云基础设施系统102通信。企业计算机系统126中的每一个企业计算机系统可以利用不同的通信协议与云基础设施系统102通信。企业计算机系统126可以支持相同或不同的安全协议。在一

些实施例中,MCS 122可以包括代理系统,以处理与企业计算机系统126的通信。

[0070] 协议可以包括通信协议,诸如SPeeDY(“SPDY”)。协议可以包括诸如基于HTTP的协议之类的应用协议。在一些实施例中,企业计算机系统126可以利用诸如REST或简单对象访问协议(“SOAP”)之类的通信协议与云基础设施系统102通信。例如,REST协议可以支持包括统一资源标识符(“URI”)或统一资源定位符(“URL”)的格式。针对利用REST协议的通信而被格式化的企业数据可以容易地转换成诸如JavaScript对象符号(“JSON”)、逗号分隔值(“CSV”)和简易信息聚合(“RSS”)之类的数据格式。企业计算机系统126和云基础设施系统102可以利用诸如远程过程调用(“RPC”)(例如,扩展标记语言(“XML”)RPC)之类的其它协议来通信。

[0071] 在一些实施例中,MCS 122可以包括被配置为支持与由云基础设施服务102提供的一个或多个服务的通信的适配器接口,该一个或多个服务中的一些服务可以支持用于通信的不同协议或技术。在一些实施例中,MCS 122可以包括被配置为支持与企业计算机系统126的通信的适配器接口,这些企业计算机系统中的一些企业计算机系统可以支持用于通信的不同协议或技术。MCS 122可以包括一个或多个适配器,该一个或多个适配器中的每个适配器可以被配置为根据通信协议、企业计算机系统的类型、应用的类型、服务的类型或其组合进行通信。适配器支持的通信协议可以特定于服务或企业计算机系统126中的一个或多个企业计算机系统。

[0072] 在某些实施例中,客户端计算设备104、106和108可以各自实现可以提供用于与MCS 122通信的特定UI的应用。特定UI可以被配置为利用特定通信协议进行通信。在一些实施例中,特定UI可以包括可以被调用以便与MCS 122通信的可调用接口、函数、例程、方法和/或操作。特定UI可以为了与由云基础设施服务102提供的服务或与企业计算机系统126通信和/或请求服务而接受企业数据作为输入参数。在一些实施例中,通过MCS 122的通信可以为了利用定制的通信协议进行通信而被转换。在一些实施例中,特定UI可以对应于应用中的定制客户端。

[0073] MCS 122可以包括一个或多个可调用接口,例如API。与MCS 122相关联的可调用接口可以使得移动计算设备上的应用能够向MCS 122传送请求。与MCS 122相关联的可调用接口可以支持公共或标准接口,该公共或标准接口可以允许根据标准化协议、架构风格和/或格式(例如,REST协议)从应用接收请求(包括请求的参数)。与MCS 122相关联的可调用接口可以是计算设备104、106或108中的任何计算设备的用户可配置的。与MCS 122相关联的可调用接口可以根据通信协议接收对服务的请求。设备应用开发人员可以针对他们的定制应用连接到MCS 122。在一些实施例中,与MCS 122相关联的可调用接口可以由开发应用的同一个人配置,使得该人可以实现定制应用,以与MCS 122通信。

[0074] 与MCS 122相关联的可调用接口还可以使企业计算机系统126能够根据标准化的协议或格式与MCS 122通信。与应用开发人员类似,管理企业计算机系统的人可以实现被配置为经由一个或多个可调用接口与MCS 122通信的代码(例如,代理系统)。与MCS 122相关联的可调用接口可以基于计算设备的类型、企业计算机系统的类型、应用、代理系统、服务、协议或其它标准来实现。在一些实施例中,与MCS 122相关联的可调用接口可以支持对服务的请求,这些服务包括认证、压缩、加密、使用游标的分页、基于客户端的调节(throttling)、不可否认性、日志记录和度量收集。在一些实施例中,与MCS 122相关联的可

调用接口可以被实现为用于定制的业务相关服务，诸如认证、策略实施、响应的高速缓存、对MCS 122的调用的调节、在异步和同步模式之间的变换、对底层服务的调用的日志记录或其组合。在一些实施例中，与MCS 122相关联的可调用接口可以使用户能够加载用于由云基础设施系统102实现的定制代码。定制代码可以为云基础设施系统102实现与MCS 122相关联的一个或多个可调用接口，该一个或多个可调用接口可以使用户能够访问定制服务或其它企业计算机系统。

[0075] 与MCS 122相关联的协议变换器可以处理消息，以确定用于消息的通信协议和/或以将消息转换为用于目的地的通信协议。与MCS 122相关联的协议变换器可以转换从客户端计算设备104、106或108接收的请求。该请求可以从由客户端计算设备104、106或108支持的通信协议的格式转换为由企业计算机系统126或云基础设施服务102提供的服务所支持的通信协议。与MCS 122相关联的协议变换器可以转换从由企业计算机系统126或云基础设施服务102提供的服务接收的响应。响应可以从由企业计算机系统126或云基础设施服务102提供的服务支持的通信协议的格式转换成由客户端计算设备104、106或108支持的通信协议的格式。

[0076] 与MCS 122相关联的安全服务可以管理对于从客户端计算设备104、106或108中的任何客户端计算设备接收的请求的安全认证。与MCS 122相关联的安全服务可以保护客户进程和企业数据的完整性。为了防止系统或数据被破坏，当从客户端计算设备104、106或108接收到请求时，安全认证可以发生。安全认证可以在分派请求以供云基础设施系统102处理之前被执行。为用户确定的安全认证可以使得与移动计算设备相关联的用户能够具有经由MCS 122请求服务的授权。安全认证可以减少用户对经由MCS 122请求的不同请求和/或服务进行认证的工作。与MCS 122相关联的安全服务可以被实现为被配置为执行认证请求的安全性的各种操作的一个或多个功能块或模块。

[0077] 与MCS 122相关联的认证服务可以管理用于从客户端计算设备104、106或108接收的请求的安全认证。与MCS 122相关联的认证服务可以确定用于与向MCS 122发送请求的计算设备相关联的用户的安全认证。安全认证可以基于时间段来确定，该时间段可以绑定到应用的操作(例如，启动应用)、请求、计算设备、企业计算机系统、与请求相关的其它标准或其组合。安全认证可以针对以下各项中的任何一个进行验证和准许，诸如单独的请求、一个或多个企业计算机系统、特定服务、服务类型、用户、计算设备、用于确定安全认证的其它标准或其组合。在一些实施例中，云基础设施系统102可以存储从企业计算机系统或支持企业计算机系统的认证系统接收的用户的认证信息。云基础设施系统102可以通过执行查找功能以确定与请求相关联的用户的身份是否具有做出这种请求的授权来确定认证。所存储的认证信息可以包括用户可以被授权访问的信息，诸如请求的类型、功能、企业计算机系统、企业数据等。在一些实施例中，基础设施系统102可以发起与发出请求的计算设备的通信，以确定认证。

[0078] 在一些实施例中，安全认证可以基于与请求服务的用户关联的角色来确定。角色可以与请求访问MCS 122的用户关联。在一些实施例中，用户可以作为MCS 122的订户或租户来请求服务，该用户可以被准许访问由MCS 122提供的资源和/或服务。认证可以对应于用户对MCS 122的订阅，使得用户可以被授权作为订户经由MCS 122请求服务。在一些实施例中，订阅可以限于由MCS 122提供的特定资源集合。安全认证可以基于MCS 122的用户可

访问的资源和/或服务。在一些实施例中，可以在执行期间向请求供应被称为“运行时环境”的模板。运行时环境可以与为请求、用户或设备分配的资源关联。

[0079] 在一些实施例中，与MCS 122相关联的认证服务可以请求身份管理系统确定用于用户的安全认证。身份管理系统可以由云基础设施系统102(例如，作为身份管理114)实现或由云基础设施系统102外部的另一计算机系统实现。身份管理114可以基于用户的角色或者用户对访问MCS 122的订阅来确定用户的安全认证。可以针对企业计算机系统、企业计算机系统提供的服务、企业计算机系统的功能或特征、用于控制对企业计算机系统的访问的其它标准或其组合来向角色或订阅指派特权和/或资格。

[0080] ADF

[0081] 可以在云基础设施系统102中提供各种不同的ADF 124。ADF 124提供基础设施代码，以实现基于敏捷的SOA的应用。ADF 124还通过一个或多个开发工具(例如，“Oracle JDeveloper 11g”开发工具)提供开发的可视的和声明性的方法。由ADF 124提供的一个或多个框架可以实现MVC设计模式。这种框架提供集成的解决方案，该集成的解决方案利用对以下领域的解决方案来覆盖MVC架构的所有层，这些领域诸如对象/关系映射、数据持久性、可重用控制器层、丰富的Web UI框架、数据到UI的绑定、安全性和定制。延伸到核心的基于Web的MVC方法之外，这种框架还与Oracle SOA和WebCenter Portal框架集成，从而简化了完整复合应用的创建。

[0082] 在某些实施例中，ADF 124使得开发敏捷应用变得容易，其中这些敏捷应用通过将服务接口耦合到由云基础设施系统102提供的内置业务服务来将数据作为服务暴露。业务服务实现细节的这种分离在ADF 124中经由元数据执行。这种元数据驱动的架构的使用使应用开发人员能够专注于业务逻辑和用户体验，而不是服务被如何访问的细节。在某些实施例中，ADF 124在模型层中的元数据中存储服务的实现细节。这使开发人员能够在不修改UI的情况下交换服务，从而使应用极其敏捷。此外，创建UI的开发人员不需要操心业务服务访问细节。相反，开发人员可以专注于开发应用接口和交互逻辑。创建用户体验可以像将期望的业务服务拖放到可视页面设计器上并且指示哪种类型的组件应当表示该数据一样简单。

[0083] 在各种实施例中，开发人员与ADF 124交互，以创建形成企业应用的模块。企业应用可以在云基础设施系统102的上下文内执行。在各种实施例中，开发人员与ADF 124交互，以创建形成移动应用的模块。移动应用可以在云基础设施系统102的上下文内执行。正如通过阅读本文提供的公开内容将对相关领域的技术人员明显的，下面描述的本发明的特征可以利用编程语言和应用开发框架的任何期望的组合实现。

[0084] 在一个示例中，由ADF 124提供的一个或多个框架可以被体现为Oracle ADF。相应地，ADF 124中的框架可以基于MVC设计模式。MVC应用被分为：1)处理与数据源的交互并运行业务逻辑的模型层、2)处理应用UI的视图层、以及3)管理应用流并充当模型层和视图层之间的接口的控制器。将应用分为这三个层简化了跨应用对组件的维护和重用。每个层与其它层的独立性导致松散耦合的SOA。

[0085] 在各种实施例中，ADF 124提供允许开发人员以多层的形式创建应用的工具和资源，每层包含根据预定义的规范实现期望的逻辑的代码模块/文件。因此，在一个实施例中，ADF 124使应用能够被开发为四层：包含提供应用的UI的代码模块/文件的视图层、包含控

制应用的流的代码模块的控制器层、包含为底层数据提供抽象层的数据/代码模块的模型层、以及包含提供对来自各种源的数据的访问以及处理业务逻辑的代码模块的业务服务层。

[0086] 在某些实施例中,ADF 124让开发人员选择在实现这些层中的每个层时他们偏好使用的技术。企业JavaBean(“EJB”)、Web服务、JavaBeans、JPA/EclipseLink/TopLink对象以及许多其它事物都可以被用作ADF 124的业务服务。视图层可以包括利用Java Server Faces(“JSF”)、Desktop Swing应用和Microsoft Office前端实现的基于Web的界面以及用于移动设备的界面。

[0087] 在一方面,视图层表示正在被开发的应用的UI。视图层可以包括桌面视图、移动视图和基于浏览器的视图,这些视图中的每个视图提供UI的全部或一部分,并且以对应于视图类型的各种方式可访问。例如,网页可以由应用响应于接收到包含对应URL的客户端请求而发送。然后,网页可以由与发出请求的客户端系统关联的显示单元(未示出)上的浏览器显示,由此使发出请求的客户端系统的用户能够与企业应用交互。ADF 124支持对业务服务的多信道访问,从而允许业务服务的重用和从Web客户端、基于客户端-服务器Swing桌面的应用、Microsoft Excel电子表格、诸如智能电话之类的移动设备等进行访问。

[0088] 可以利用超文本标记语言(“HTML”)、Java服务器页面(“JSP”)和JSF中的一个或多个来实现形成视图层(诸如网页)的代码文件/模块。可替代地,UI可以利用诸如Swing和/或XML之类的Java组件来实现。如进一步指出的,UI可以利用用户对桌面应用(诸如Microsoft的Word和Excel)的经验和熟悉。

[0089] 如上面所指出的,在这些层中的每一层中提供相关的用户开发的代码/数据模块。但是,每一层通常包含由ADF 124提供的其它预定义的代码/数据模块。这些预定义的模块中的一些预定义的模块可以在开发期间使用,例如,用作用于开发网页的模板,用于在所开发的代码中包括期望的功能,等等。其它预定义模块(诸如URL重写模块)可以与开发的应用一起部署,并且可以在企业应用执行期间向用户提供附加的功能(将所请求的URL映射到内部名称)。

[0090] 控制器层包含控制应用的流的代码模块/文件。每个控制器对象包含根据在视图层中呈现信息的期望方式实现的软件指令和/或数据。期望方式可以包括当用户点击/选择另一个网页中的链接时要显示的特定网页、当执行期间发生错误时要显示的页面、指示要存储/检索的特定数据,等等。

[0091] 在一方面,控制器层管理应用的流并处理用户输入。例如,当在页面上点击“搜索”按钮时,控制器确定要执行的动作(进行搜索)和导航到的位置(结果页面)。在JDeveloper中对于基于Web的应用存在两个控制器选项:标准JSF控制器或扩展JSF控制器功能的ADF控制器。无论使用哪种控制器,应用流通常都是通过在图表上布置页面和导航规则来设计的。应用的流可以被分解为较小的、可重用的任务流;包括非可视组件,诸如流中的方法调用和决策点;并且创建在单个包含页面(containing page)的区域内运行的“页面片段”流。

[0092] 形成控制器层的代码模块/文件常常被实现为接收客户端请求并作为对应的响应发送期望的网页的Java小服务程序。控制器对象还可以例如被实现为Apache Jakarta Struts控制器,或根据JSF标准实现。

[0093] 模型层包含将各种业务服务连接到在其它层中的使用它们的对象(诸如连接到上

面讨论的控制器对象或直接连接到桌面应用)的数据/代码模块。模型层的每个抽象数据对象提供可以被用来访问在底层业务服务层中执行的任何类型的业务服务的对应接口。数据对象可以从客户端抽象服务的业务服务实现细节和/或向视图组件暴露数据控制方法/属性,从而提供视图层与数据层的分离。

[0094] 在一方面,模型层包括利用元数据文件来定义接口的两个组件(数据控制组件和数据绑定组件)。数据控制组件从客户端抽象业务服务实现细节。数据绑定组件向UI组件暴露数据控制方法和属性,从而提供视图与模型的干净分离。由于模型层的元数据架构,当将任何类型的业务服务层实现绑定到视图层和控制器层时,开发人员获得相同的开发体验。

[0095] 在某些实施例中,ADF 124强调贯穿整个开发过程使用声明性编程范式,以允许用户专注于应用创建的逻辑,而不必参与实现细节。在高级别,用于Fusion Web应用的开发过程通常涉及创建应用工作空间。通过利用向导,由开发人员选择的技术所需的库和配置被自动添加,并且应用被结构化为具有包和目录的项目。

[0096] 通过对数据库对象进行建模,可以创建任何数据库的在线数据库或离线副本,定义被编辑,以及模式被更新。通过利用统一建模语言("UML")建模器,然后可以为应用创建用例。还可以设计应用控制和导航。图表绘制器(diagrammer)可以被用来可视地确定应用控制和导航的流。然后,可以自动创建描述流的底层XML文件。资源库可以被用来允许开发人员通过简单地将导入的库拖放到应用中来查看和使用这些导入的库。可以从数据库表利用向导或对话框创建实体对象。视图对象从这些实体对象被创建以供应用中的页面使用。可以实现验证规则和其它类型的业务逻辑。

[0097] 在这个示例中,业务服务层管理与数据持久化层的交互。它提供诸如数据持久性、对象/关系映射、事务管理和业务逻辑执行之类的服务。业务服务层可以以以下选项中的任何选项实现:作为简单的Java类、EJB、Web服务、JPA对象和Oracle ADF业务组件。此外,数据可以直接从文件(XML或CSV)以及REST中被消费。因此,每个业务服务管理与对应的数据持久化层的交互,并且还提供诸如对象/关系映射、事务管理、业务逻辑执行等之类的服务。业务服务层可以利用简单的Java类、企业Java Beans、Web服务等当中的一个或多个来实现。

[0098] 业务组件表示使用例如来自Oracle公司的"Oracle ADF业务组件"实现的、用来提供与数据库、Web服务、遗留系统、应用服务器等的交互的业务服务。在一个实施例中,业务服务层的业务组件包含合作以提供业务服务实现的应用模块、视图/查询对象和实体对象的混合。应用模块可以是UI客户端为了与应用/事务数据一起工作而与其通信的事务组件/代码模块。应用模块可以提供可更新的数据模型以及与用户事务相关的过程/函数(通常被称为服务方法)。

[0099] 实体对象可以表示数据库表中的对应行并且简化对存储在对应行中的数据的操纵(更新、删除等)。实体对象常常封装用于对应行的业务逻辑,以确保一致地实施期望的业务规则。实体对象还可以与其它实体对象关联,以反映存储在底层数据库中的行之间存在的关系。

[0100] 图2示出了根据本发明的一些实施例的、用于促进移动计算设备和企业计算机系统之间的通信的计算环境200的框图。为了说明的目的,本文提供了各种示例来描述用于使移动计算设备(例如,计算设备202)与一个或多个企业计算机系统(诸如云企业计算机系统240(例如,"serviceprovider.com")和内部部署的企业计算机系统250)通信的技术。这些

通信可以是为了交换或传送企业数据、请求由企业计算机系统提供的服务、传送消息或其组合。

[0101] 消息可以包括服务调用消息、结果消息、请求消息、在内部传送的其它消息、在计算设备和企业计算机系统之间传送的其它消息或其组合。消息可以包括消息类型(例如,来自一组共享类型常量的类型值)、相关性id(例如,用来将这个消息与一个或多个其它消息相关的id)、支持基于优先级的消息队列的优先级信息、超时、支持消息数据隔离的敏感度指示、消息源(例如,发送者的统一资源标识符)、消息目的地(例如,唯一地识别目的地的统一资源标识符)、请求上下文(例如,来自分派器的请求信息)和/或消息有效载荷,该有效载荷可以依赖于正在被发送的消息的类型(诸如参数数据和结果数据)而具有不同的属性。

[0102] 如本文所描述的企业数据可以包括从企业计算机系统接收的数据、发送到企业计算机系统的数据、由企业计算机系统处理的数据或其组合。企业数据可以是与用于消费者应用和/或服务的数据可区分的。在一些实施例中,例如,企业数据可以基于企业数据的应用或使用而改变,而用于消费者应用的数据(例如,消费者数据)可以在使用中保持不变。在某些实施例中,企业数据可以包括指示用于存储、使用和/或管理企业数据的标准的规则或者可以与这些规则关联。例如,企业数据可以与指示用于存储、使用和/或管理企业数据的一个或多个策略的策略信息关联。在某些实施例中,策略信息可以被包括在企业数据中。在某些实施例中,企业数据可以包括由在企业计算机系统中执行的应用或服务处理、存储、使用或传送的数据。例如,企业数据可以包括业务数据(例如,业务对象),诸如来自企业应用的JSON格式化数据、结构化数据(例如,键值对)、非结构化数据(例如,由应用处理或使用的内部数据、JSON格式的数据、社交帖子、会话流、活动馈送,等等)、二进制大对象(“BLOB”)、文档、系统文件夹(例如,沙箱环境中的应用相关的文件夹)、使用REST技术的数据(在本文被称为“RESTful数据”)(例如,通过REST端点变得可用的同步数据)、系统数据、配置数据、同步数据或其组合。在一些实施例中,企业数据可以包括REST格式化的企业数据。REST格式化的企业数据可以包括RESTful数据。REST格式化的数据可以包括根据由企业计算机系统实现的REST技术而被格式化的数据。配置或同步数据可以包括用于企业数据的同步的数据,诸如版本、历史、集成数据,等等。企业数据中的文档可以包括XML文件、可视资产、配置文件、媒体资产,等等。BLOB可以包括在数据库管理系统中被存储为单个实体的二进制数据的集合,诸如图像、多媒体对象、或可执行代码、或者本领域中已知的其它对象。

[0103] 企业计算机系统可以包括被配置为针对实体或企业操作的各种计算系统。例如,企业计算机系统可以包括一个或多个计算机系统,诸如企业服务器计算机(例如,后端服务器计算机),以处理对服务的请求。企业计算机系统可以包括可以利用企业数据进行处理和/或操作的应用和/或服务。例如,企业计算机系统250可以提供用于管理或操作企业的一个或多个服务和/或应用。服务可以包括但不限于客户关系管理(“CRM”)、人力资本管理(“HCM”)、人力资源(“HR”)管理、供应链管理、企业通信、电子邮件通信、业务服务、其它企业管理服务或应用或其组合。企业计算机系统250可以包括专用于提供一个或多个服务的一个或多个计算机系统。在一些实施例中,提供服务的每个不同计算机系统可以位于企业内部,或者可以远离企业定位。在一些实施例中,支持不同服务的多个不同计算机系统可以位于单个地理位置,诸如企业内部。在图2所示的示例中,内部部署的企业计算机系统250可以包括HR系统254和CRM系统256,这两者都可以位于企业内部。在一些实施例中,企业计算机

系统250可以包括或实现代理系统252，以促进或处理云计算机系统210和一个或多个企业系统之间的通信。下面进一步详细描述企业计算机系统，诸如云企业计算机系统240和内部部署的企业计算机系统250。

[0104] 计算机环境200可以包括被实现以作为安全的中间计算环境操作的MCS 212，MCS 212可以促进计算设备202和一个或多个企业计算机系统之间的通信，因为计算设备202可能没有被配置为与这种企业计算机系统通信。例如，一些企业计算机系统可能由遗留的或后端计算机系统支持。这种系统可以被配置为利用不同的通信和/或安全协议来操作。由这种企业计算机系统支持的协议可以与由移动计算设备支持的协议不同。MCS 212可以支持与不同类型的移动计算设备通信。由此，MCS 212可以实现促进企业计算机系统和移动计算设备之间的通信的技术，以使得虽然它们的通信不兼容(诸如存在格式或通信协议之间的差异)，但是它们也能够彼此通信。例如，MCS 212可以变换移动计算设备和企业计算机系统之间的通信协议。

[0105] 云计算机系统210可以支持MCS 212。云计算机系统210可以利用硬件、软件、固件或其组合来实现。例如，云计算机系统210可以包括一个或多个计算设备，诸如服务器计算机。云计算机系统210可以包括一个或多个存储器存储设备和一个或多个处理器。存储器存储设备可以是(一个或多个)处理器可访问的并且可以包括存储在其上的指令，这些指令当被(一个或多个)处理器执行时，使(一个或多个)处理器实现本文公开的一个或多个操作。在一些实施例中，存储器存储设备可以作为本地存储装置(例如，高速缓存)操作。云计算机系统210可以包括不同种类的操作系统。存储器存储设备可以是(一个或多个)处理器可访问的并且可以包括存储在其上的指令，这些指令当被(一个或多个)处理器执行时，使(一个或多个)处理器实现本文公开的一个或多个操作、方法或过程。存储器存储装置可以作为本地存储装置操作。本地存储装置可以利用诸如存储器存储设备或其它计算机可读存储介质之类的任何类型的持久性存储设备来实现。在一些实施例中，本地存储装置可以包括或实现一个或多个数据库(例如，文档数据库、关系数据库或其它类型的数据库)、一个或多个文件存储库、一个或多个文件系统或其组合。本地存储装置可以存储企业数据。

[0106] 在某些实施例中，云计算机系统210可以包括一个或多个数据存储库，诸如元数据储存库224、诊断存储库226和分析存储库228。数据存储库224、226、228可以由云计算机系统210中的任何组件访问。

[0107] 元数据储存库224可以存储与MCS 212关联的所有元数据。这个信息可以包括各自具有其自己关于可用性和性能的需求的运行时数据和设计时数据。MCS 212的租户或订户可以具有任何数量的应用。每个应用可以被版本化并且可以具有相关联的零个或更多个版本化的资源API和这些资源API约定(contract)的零个或更多个版本化的服务实现。这些实体是运行时用来将虚拟请求(mAPI)映射到具体服务实现(服务)的东西。这种映射向移动开发人员提供了当她设计和建立其应用时，不必知道实际的实现服务的优势。并且不需要她针对每个服务错误修复重新发布新的应用。元数据储存库224可以存储可以由计算设备(例如，计算设备202)调用的一个或多个可调用接口。可调用接口可以是应用的用户(例如，开发人员)可定制的，以促进与MCS 212的通信。元数据储存库224可以存储对应于可调用接口的一种或多种配置的元数据。元数据储存库224可以被配置为存储用于实现可调用接口的元数据。可调用接口可以被实现为在用于通信的一种格式、协议或架构风格与用于通信的

另一种格式、协议或架构风格之间进行变换。元数据储存库224可以是由通过认证的用户经由外部网络可修改的。

[0108] 诊断存储库226可以存储关于在MCS 212中发生的处理的诊断信息。诊断存储库226可以存储经由MCS 212传送的消息和日志信息。分析存储库228可以存储在系统中在处理期间捕获的日志记录和分析数据。

[0109] 云计算机系统210可以代表MCS 212利用它的计算资源来实现定制代码216(例如，操作、应用、方法、函数、例程等)的执行。计算资源可以被分配以用于针对作为MCS 212的订户或租户而关联的特定用户来使用。资源可以针对用户、设备、应用或与订户相关的其它标准来分配。依赖于移动计算设备试图与企业计算机系统通信的需求，MCS 212可以被缩小或放大。MCS 212可以被配置为使得它是弹性的，以处理高于移动计算设备和企业计算机系统之间的正常流量的突发和临时周期。在一些实施例中，MCS 212可以包括支持可扩展性的元件，使得组件可以被添加或代替以满足通信中的需求。

[0110] 计算设备202可以与MCS 212通信(例如，发送请求消息)，以请求由企业计算机系统提供的服务。计算设备202(例如，移动计算设备)可以利用硬件、固件、软件或其组合来实现。计算设备202可以经由MCS 212与企业计算机系统240、250通信。计算设备202可以包括或可以被实现为端点设备、PDA、平板计算机、膝上型计算机、移动计算设备、桌上型计算机、可穿戴计算机、寻呼机等。计算设备202可以包括一个或多个存储器存储设备和一个或多个处理器。计算设备202可以包括不同种类的操作系统。存储器存储设备可以是(一个或多个)处理器可访问的并且可以包括存储在其上的指令，这些指令当由(一个或多个)处理器执行时，使得(一个或多个)处理器实现本文公开的一个或多个操作、方法或过程。存储器存储装置可以作为本地存储装置操作。本地存储装置可以利用诸如存储器存储设备或其它计算机可读存储介质之类的任何类型的持久性存储设备实现。在一些实施例中，本地存储装置可以包括或实现一个或多个数据库(例如，文档数据库、关系数据库或其它类型的数据库)、一个或多个文件存储库、一个或多个文件系统或其组合。本地存储装置可以存储企业数据。

[0111] 在各种实施例中，计算设备202可以被配置为执行和操作一个或多个应用，诸如web浏览器、客户端应用、专有客户端应用，等等。应用可以包括针对由企业计算机系统提供的企业数据和/或服务被配置的特定应用。客户端应用可以经由一个或多个网络被访问或操作。应用可以包括用于操作应用的图形UI(“GUI”)。

[0112] 计算设备202可以利用无线通信经由一个或多个通信网络与MCS 212进行通信。通信网络的示例可以包括移动网络、无线网络、蜂窝网络、LAN、广域网(“WAN”)、其它无线通信网络或其组合。在某些实施例中，计算设备202可以利用定制通信协议(例如，定制协议)建立与MCS 212的通信连接。通信连接可以通过云计算机系统210与MCS 212建立。定制协议可以是基于HTTP的协议。通过利用定制通信协议，计算设备202可以在任何计算设备平台上操作，以与云计算机系统210进行通信。

[0113] 计算设备202可以通过一个或多个可调用接口(例如API)与云计算机系统210进行通信。可调用接口可以在计算设备202上实现。可以为定制应用实现使得这些应用能够与MCS 212通信的可调用接口。在一些实施例中，可调用接口可以针对MCS 212被开发。可调用接口可以使得应用能够与MCS 212进行通信，而不必适应协议(例如，通信或开发协议)和/或架构风格或格式的差异。

[0114] MCS 212可以由一个或多个防火墙204、230保护，以便提供处理请求和执行定制代码216的安全的环境。计算设备202和MCS 212之间的通信可以由外部通信防火墙204分离。防火墙204可以与云计算机系统210连接，以促进对MCS 212的安全访问。防火墙204可以允许云计算机系统210和计算设备(例如，计算设备202)之间的消息传送。这些消息(例如，HTTP消息或REST消息)可以符合可以由可调用接口支持的通信协议(例如，HTTP或REST)。在另一个示例中，云计算机系统210和计算设备202之间的消息可以符合诸如SPDY之类的通信协议。MCS 212可以管理防火墙230，以保护云计算机系统210和企业计算机系统240、250之间的通信。防火墙230可以允许云计算机系统210和计算设备(例如，计算设备202)之间的消息传送。这些消息(例如，SPDY消息、HTTP消息或REST消息)可以符合通信协议(例如，SPDY、HTTP或REST)。计算设备202和企业计算机系统240、250之间的通信可以是经由MCS 212的双向的。

[0115] 因为与计算设备202和企业计算机系统240、250的通信可以经由不安全的公共网络发生，所以防火墙204、230为去往和来自MCS 212的通信提供添加的保护层。防火墙204、230可以使MCS 212能够将它的内部网络与连接计算设备202和企业计算机系统240、250的外部网络区分开。防火墙204、230虽然被示为两个不同的防火墙，但是在一些实施例中可以被实现为封装MCS 212的单个防火墙。

[0116] 云计算机系统210还可以通过与企业计算机系统通信而作为中间计算环境操作，其中这些企业计算机系统中的一些企业计算机系统可以具有不同的通信协议。这些通信协议可以是定制的或特定于与云计算机系统210通信的应用或服务。另外，云计算机系统210可以与企业计算机系统通信，以便根据企业计算机系统所支持的格式来提供企业服务和/或交换企业数据。云计算机系统210可以维护企业数据的本地存储装置(例如，本地高速缓存)并且可以使用本地存储装置来管理移动计算设备和企业计算机系统240、250之间的企业数据的同步。

[0117] 计算设备202可以与MCS 212进行通信(例如，发送请求消息)，以请求由企业计算机系统提供的服务。通过防火墙204接收的请求可以首先被安全服务232处理。安全服务232可以管理用于与请求关联的用户的安全认证。因此，云计算机系统可以提供包括提供本文描述的安全机制的技术优点，该安全机制可以保护客户通信和企业数据的完整性。云计算机系统的技术优点可以包括防止或减少受损的通信和/或数据被损害，认证可以在最开始进行，从而将访问仅限制到具有所需凭证的用户。云计算机系统的技术优点可以包括服务和服务调用流被结构化，以使得当请求进来时，它们只能访问它们被授权的服务。通过将授权与系统处理的其余部分解耦，另一个技术优点可以包括授权“可以由谁来完成什么”的任务被委派给专门供应的安全子系统(例如，身份管理系统)，该安全子系统可以被扩展以支持特定公司客户所需的任何附加的定制安全措施。在一些实施例中，可以为请求、会话、用户、设备、与用户相关的其它标准或其组合确定安全认证。安全认证可以为接收到的每个请求执行。在一些实施例中，安全服务232可以基于请求的先前验证来确定认证。安全认证可以为用户或设备确定，以使得对不同企业计算机系统240、250的请求可以基于安全性的单次验证来认证。

[0118] 本发明的其它技术优点可以包括云计算机系统使计算设备能够与各种企业计算机系统通信，这些企业计算机系统中的一些企业计算机系统可以不同地实现。例如，计算设

备202、云计算机系统210和企业计算机系统250可以位于彼此物理地分离的不同地理位置。因此，计算设备202可以与企业计算机系统250通信，而不管它们的位置。技术优点可以包括云计算机系统使计算设备能够向可以支持一个或多个不同的安全协议的企业计算机系统传送对服务的请求。在一些情况下，企业计算机系统可以由不能够容易地适应于不同安全协议的后端系统支持。在一些情况下，应用的开发人员可能期望能够实现应用，以能够在不知道这些安全协议的情况下请求服务。企业计算机系统的用户(例如，管理员或架构师)可能同样期望能够接收请求，而无需适应不同类型的应用、安全协议和标准。技术优点可以使得能够通过实现如本文描述的云计算机系统来满足这些期望，该云计算机系统可以处理安全认证，使得请求可以满足正被请求的不同企业计算机系统的安全措施。

[0119] 在一些实施例中，安全服务232可以确定用于所请求的企业计算机系统的安全协议，并且相应地根据这种安全协议生成安全令牌。安全令牌可以与请求一起被传递到企业计算机系统，以使得该企业计算机系统能够基于生成的安全令牌来验证认证。企业计算机系统可以支持不同的安全协议。安全协议可以是标准，安全性通过该标准来确定。可以基于由安全服务232生成的安全令牌来验证安全性。安全服务232可以确定用于针对请求被识别的企业计算机系统的安全协议。在一些实施例中，企业计算机系统250可以具有代理系统252，代理系统252可以根据由MCS 212支持的定制或特定安全协议而被配置或实现。照此，MCS 212可以根据这种定制的安全协议生成安全令牌。

[0120] 云计算机系统210可以包括一个或多个负载均衡器系统206、208，实现一个或多个负载均衡器系统206、208和/或与一个或多个负载均衡器系统206、208通信。在确定安全认证时，云计算机系统210可以请求负载均衡器系统206、208中的任何一个来检查它接收到的请求以及检测请求针对哪个服务。MCS 212可以被配置为具有负载均衡器206、208并且利用开始时的资源被更新，以使得当请求进来时，负载均衡器206、208可以跨不同的资源平衡所请求的负载。

[0121] 云计算机系统210可以包括分派器218，分派器218可以处理请求并将它们分派到适当的服务。请求可以在分派时被路由到适当的服务。在一些实施例中，服务本身可以在MCS 212中或在企业计算机系统中将一个内部请求路由到另一个内部服务。在一些实施例中，分派器218可以解析请求，以便基于请求的URI和/或URL中识别出的目的地的位置(例如，地址)来确定请求的目的地。分派器218可以解析请求及其报头，以提取以下信息中的一个或多个：租户标识符、服务标识符、应用名称、应用版本、请求资源、操作和参数，等等。分派器218可以使用解析出的信息来执行元数据储存库224中的查找。分派器218可以检索对应的应用元数据。分派器218可以基于所请求的资源和元数据中的映射来确定目标服务。虽然最初是非常基本的映射，但是元数据可以被增强以提供更复杂的、基于规则的分派。分派器218可以执行任何特定于分派器的日志记录、度量收集，等等。然后，分派器218可以根据应用元数据执行初始授权。分派器218可以格式化入站请求和任何其它必要的信息，并且把该消息放在路由总线220上以供进一步处理。分派器218可以将请求放在队列上并且等待对应的响应。分派器218可以处理从路由总线220接收到的响应并将响应返回到计算设备202。

[0122] 除了处理对外部请求的分派，分派器218还可以在分派内部请求时起作用。这些内部请求可以以复合服务或对服务的定制代码调用的形式进入。在这两种情况下，调用者都可以使用如在应用内定义的逻辑服务名称。分派器218可以使用当前执行上下文来确定该

应用并且使用该逻辑名称来确定要调用的适当服务。

[0123] 云计算机系统210可以包括路由总线220，以管理向路由总线220注册的消息到目的地的递送。路由总线220可以作为用于管理云服务212中的通信的中央系统操作。通过路由总线220传送的数据可以被处理，以捕获和存储数据。路由总线220可以提供框架，以使得附加的集中式服务(附加的授权、调试，等等)可以容易地根据需要被插入。由路由总线220捕获的数据可以被存储在诊断存储库226和/或分析存储库228中。

[0124] 路由总线220可以将消息路由到一个或多个目的地。在一些实施例中，消息可以包括执行定制代码216的请求。在这种实施例中，路由总线220可以请求234调用定制代码216。在一些实施例中，路由总线220可以将请求传递到由请求中的信息识别出的目的地企业计算机系统。路由总线220可以请求236适配器接口222执行变换(如果需要的话)，以将请求传递到企业计算机系统，例如企业计算机系统240或企业计算机系统250。

[0125] 在某些实施例中，云计算机系统210可以包括或实现适配器接口222，以便将消息变换或转换成由接收企业计算机系统支持的协议。适配器接口222可以与企业计算机系统240、250中的每一个建立单独的通信连接。云计算机系统210可以被配置为经由一个或多个网络(未示出)与企业计算机系统240、250进行通信。通信网络的示例可以包括互联网、移动网络、公共网络、无线网络、蜂窝网络、LAN、WAN、其它通信网络或其组合。在某些实施例中，通信连接可以是利用高速通信干线促进的高速通信连接。与企业计算机系统240、250的通信可以穿过防火墙230，这确保与外部网络的通信被保护，以防止经由这些通信对MCS 212的未授权访问。

[0126] 在一些实施例中，云计算机系统210可以促进向计算设备202的用户的通知。云计算机系统210可以包括支持与用户的有状态交互的警报管理服务，例如以基于用户偏好通过一个或多个信道递送警报、等待响应以及基于响应采取行动。对于在一个信道上发送的警报的响应可以通过另一个信道被接收，服务需要能够处理该响应。平台可以具有用于流行交互模式的内置状态模型，并且是利用新的状态模型可扩展的。一些警报信道可以包括单向或者双向的已知的通信资源。示例包括SMS、**Twitter®**、推送通知以及Google Cloud **Messaging®**。

[0127] 在一些实施例中，云计算机系统210可以使计算设备能够访问和/或请求一个或多个服务，诸如对象存储服务、数据库服务、访问web服务、社交服务、资源服务或其组合。

[0128] 云计算机系统210可以提供可以为BLOB提供存储设施的对象存储服务。存储的基本单元可以是文本，并且具有读和写操作。还可以提供用于JSON对象的基本查询设施。

[0129] 云计算机系统210可以提供数据库服务，以允许到托管的数据库的连接，以用于执行查询或写入。所需的参数化可以需要用于数据库的完整连接串、SQL串或者所存储的要执行的过程、任何参数和可能的凭证。必要的信息可以在运行时被提供或者在应用元数据中被预先配置。

[0130] 云计算机系统210可以提供对web服务(诸如SOAP web服务)的访问。云计算机系统210可以提供对REST服务的访问，诸如到任意REST资源的连接。

[0131] 云计算机系统210可以提供对社交服务的访问，社交服务可以提供与流行的社交站点(诸如**Facebook®**、**Twitter®**等)中的许多社交站点的基本整合。这些服务可以允许利用来自这些站点的用户凭证的第三方认证以及对它们的服务的访问。示例包括发送

推文或更新您的状态。

[0132] 云计算机系统210可以提供公共云服务，以使用户能够简化和优化通信。例如，服务开发人员可以使用MCS 212的通用web服务来向利用云计算机系统210的云服务托管的资源对话。

[0133] 云计算机系统(诸如本文描述的云计算机系统)可以使移动计算设备能够与企业计算机系统通信，而不管计算资源的差异。云计算机系统可以配备有较多的资源和到企业计算机系统的较快、较可靠的连接，以频繁地进行通信来接收企业数据。云计算机系统可以管理和协调来自企业计算机系统的对服务的请求。通过将请求变换成由消息的接收者支持的协议，云计算机系统减少了开发人员配置应用以与不同类型的后端计算机系统通信的负担。企业能够维护它们的后端系统，而不必适应对于移动设备支持的通信协议的进步或变更。不同的企业计算机系统可以基于所处理的请求和所提供的服务的类型而支持不同的安全协议。通过以集中的方式管理用于访问不同企业计算机系统的安全认证，企业计算机系统不需要适应安全协议的差异。通过认证云计算机系统的用户，处理请求可以变得更高效，因为可以不用在每种情况下都执行认证。

[0134] 在一些实施例中，应用可以在诸如来自Oracle公司的Oracle移动应用框架(“MAF”)之类的MAF下被部署，其中MAF提供内置的安全性以控制对应用的访问并且确保敏感数据的加密。MAF是使用HTML5和级联样式表(“CSS”)(以在Web视图中渲染UI)、Java(用于应用业务逻辑)和Apache Cordova(以访问诸如GPS活动和电子邮件之类的设备特征)的混合移动架构。因为MAF使用这些跨平台技术，所以可以为Android和iOS设备二者构建相同的应用，而不必使用任何特定于平台的工具。在应用被部署到设备之后，它像利用诸如Objective C或Android SDK之类的特定于平台的工具创建的应用那样工作。另外，MAF允许为智能电话或为平板电脑构建相同的应用，由此允许在相同应用中重用业务逻辑并且以各种类型的设备、屏幕尺寸和能力为目标。

[0135] 图3A示出了示例移动应用主界面300，该示例移动应用主界面300包括被称为“WorkBetter”的MAF应用302，MAF应用302被部署为“重”应用(例如，以与从应用商店获得的正常的iPhone“app”相同的方式位于移动设备中的移动应用)。MAF应用可以包括作为应用特征被添加的一个或多个嵌入式应用。这种添加的应用特征被表示为在主应用的主界面或导航栏内的图标。应用特征本质上是这种移动应用的构建块。集成到MAF应用中的每个应用特征执行一组特定任务。应用特征可以被分组在一起，以补充彼此的功能。例如，提供客户联系人的应用特征可以与用于产品库存的应用特征配对。因为每个应用特征具有它自己的类加载器和web视图，所以应用特征是彼此独立的，因此单个MAF应用可以由若干不同开发团队创建的应用特征组装。应用特征还可以在其它MAF应用中重用。MAF应用本身可以被重用为另一个应用的基础，从而允许独立的软件供应商(“ISV”)创建可以由特定客户配置的应用。

[0136] 除了在设备上本地运行的混合移动应用之外，依赖于移动应用的要求和可用资源，应用特征可以被实现为以下移动应用类型中的任何移动应用类型：

[0137] • 对于在服务器上托管的移动web应用，虽然代码可以在平台之间移植，但是对设备特征和本地存储装置的访问可以是受限的，因为这些应用由设备的浏览器管理。

[0138] • 本机应用在Xcode中或者通过Android SDK创作，并且因此在为这两个平台服务

方面是受限的。代码的重用同样是受限的。

[0139] MAF支持认证和访问控制，以用于在开发人员可以指定适当的登录服务器(例如，运行具有基本认证的“Oracle身份管理”和/或“Oracle WebLogic”的服务器、支持OAuth协议的服务器等)的应用中在特征级别的细化的安全性。在运行时，向用户呈现登录屏幕，并且适当的令牌是可访问的，以用于进一步的Web服务调用。利用MAF，开发人员可以构建满足具有不同特权的用户的需求(例如，基于用户角色或特权显示/隐藏组件)的单个UI。

[0140] MAF利用SSL/TLS(HTTP安全(“HTTPS”))、设备上的加密来实施通信加密，以在加密密钥存储库中保持凭证，以用于在支持离线认证以及通过利用SQLite加密扩展的SQLite数据库加密时的验证。加密用于利用MAF构建的应用的SQLite数据库可以在开发应用时经由配置选项执行。在一些实施例中，MAF支持应用的离线和在线操作模式，使得自包含应用可以在连接模式和断开模式下在移动设备上运行。对于数据访问/存储，这种应用可以利用本地加密的SQLite数据库。可以构建应用，使得对数据的初始访问从远程服务器通过Web服务执行，然后数据被存储在本地SQLite数据库中以供离线访问。当连接再次可用时，数据可以被复制并且被同步到服务器。MAF还支持用户认证凭证的本地存储，以实现对受保护的应用的离线认证/授权。

[0141] 图3B和图3C示出了根据本发明的实施例的HR移动应用UI 304。UI 304可以在打开主界面(诸如图3A的移动应用主界面300)上的图标时被提供。在图3B中，UI 304包括关于雇员的各种HR相关信息，诸如图片、头衔、联系信息、社交网络信息、绩效/评级信息、薪酬信息、管理者、技能、位置等。图3C指示可以从其获得UI 304中的信息的各种源，诸如位于本地或云中的服务。例如，基本雇员信息可以从诸如PeopleSoft系统、应用和产品(“SAP”)等之类的内部核心HR服务306获得，而位置信息从诸如Google之类的地图服务308获得。类似地，绩效信息可以从诸如TALEO之类的人才管理云服务310获得，并且社交网络信息312(例如，Twitter、Facebook、LinkedIn等)可以从web获得。在一个实施例中，来自这些各种源的信息在被下发到移动设备202(参见图2)上的应用之前通过MCS 212(参见图2)被传输。

[0142] 图4是根据实施例的MAF运行时架构400的框图。运行时架构400包括部署到移动设备404的“瘦”设备本机容器402。运行时架构400表示将呈现与模型层和控制器逻辑分离的MVC开发方法。设备本机容器402允许MAF应用通过与本地SQLite数据库406(经由SQLite 408)、移动设备服务426(经由Apache Cordova 410的Cordova API)以及诸如配置服务器444、服务器生成的HTML 430、推送服务448和web服务440之类的服务器侧资源412交互而充当不同平台(例如，iOS、Android等等)上的本机应用。

[0143] 设备服务426是设备404本机的服务和特征(诸如相机、GPS、电子邮件等等)。配置服务器444是基于Web分布式创作和版本控制(“WebDav”)并且托管由应用配置服务使用的配置文件的服务器。WebDav在例如互联网工程任务组(“IETF”)请求注解(“RFC”)中定义。配置服务器444作为参考实现被递送。在Java 2平台企业版(“J2EE”)服务器上托管的任何常见WebDav服务可以用于此目的。服务器生成的HTML 430包括在远程服务器上托管并且用于基于浏览器的应用特征的Web内容。推送服务448可以包括例如作为向MAF应用发送通知事件的通知提供者的Apple推送通知服务(Apple Push Notification Services，“APN”)和Google云消息传送(Google Cloud Messaging，“GCM”)推送服务。Web服务440是例如远程托管的基于SOAP的Web服务。

[0144] 设备本机容器402包括使用移动设备的web引擎来显示和处理基于web的内容的Web视图416。在MAF应用中,Web视图416通过将应用标记渲染为HTML 5来交付UI。可以通过实现以下内容类型中的任何内容类型来为移动应用特征创建UI:MAF应用移动XML(“AMX”)视图420、控制器422、本地HTML 424或服务器HTML 428,其中MAF AMX视图420、控制器422和本地HTML 424提供HTML5和JavaScript呈现418。从各种内容类型实现的应用特征可以在相同的移动应用内共存,并且还可以彼此交互。

[0145] 其内容被实现为MAF AMX视图420的应用驻留在设备404上并提供最可靠的设备本机用户体验,类似于以特定于该设备的平台的语言创作的应用。MAF提供了使用户能够从适合移动设备的外形因子的组件来以声明的方式创建UI的一组代码编辑器。这些组件可以用于创建页面布局(例如,列表视图)以及输入组件(例如,输入字段)。当用户开发MAF AMX视图420时,他们可以利用数据控制组件,这些数据控制组件使得用户能够以声明的方式创建数据绑定的UI组件并且访问Web服务以及移动设备的服务(例如,相机、GPS或电子邮件)。在运行时,Web视图416中的JavaScript引擎将MAF AMX视图定义渲染为HTML5和JavaScript。

[0146] 对于其内容被实现为控制器422的应用,控制器422管理移动应用中的页面之间的流。控制器422使用户能够将应用的流分解为较小的可重用任务流并且包括非可视组件(诸如方法调用和决策点)。在图4的实施例中,控制器422被包括在MAF AMX视图420中并且由MAF AMX视图420调用,以例如转变页面和/或激活动作。但是,在替代实施例中,控制器422可以被实现为MAF AMX视图420的对等体。

[0147] 对于其内容被实现为本地HTML 424的应用,HTML页面作为MAF应用的一部分在设备上运行。本地HTML文件可以通过Apache Cordova 410和JavaScript API访问设备本机的特征和服务。

[0148] 对于其内容被实现为服务器HTML 428的应用,UI从可以在应用特征的Web视图416内打开的服务器生成的网页(服务器生成的HTML 430)递送。在MAF的上下文内,这种内容类型被称为远程URL。用于这些基于浏览器的应用的资源不驻留在设备404上。相反,UI、页面流逻辑和业务逻辑从远程服务器递送。

[0149] 当这些远程托管的Web应用之一被允许在Web视图416内打开时,它可以使用Cordova JavaScript API来访问任何指定的设备本机特征或服务,诸如相机或GPS能力。当利用远程URL内容实现应用时,用户可以利用已经针对移动使用被优化的现有的基于浏览器的应用,或者使用已经专门针对特定类型的移动设备编写的应用。对于可以在台式机或平板电脑上的浏览器内运行的应用,用户可以利用通过基于丰富客户端的组件(诸如由来自Oracle公司的“Oracle ADF Faces”提供的组件)创建的应用来实现远程URL内容。对于专门针对移动电话的应用,远程URL内容可以从利用MAF创建的网页递送。利用MAF创作的应用不仅可以在各种智能电话上渲染,而且它们还可以通过利用Apache Trinidad JSF组件和动态选择的样式表构建的UI优雅地降级到功能手机上可用的减少的能力。因为内容是远程提供的,所以只要服务器连接保持活动,应用就是可用的。

[0150] 设备本机容器402还包括提供JavaScript API的Apache Cordova 410,JavaScript API将设备的本机特征和服务集成到移动应用中。虽然用户可以从Java代码(或者在将MAF移动应用实现为本地HTML 424时利用JavaScript)以编程方式访问这些API,但是用户可以在创建MAF AMX页面时声明性地添加设备集成,因为MAF将这些API打包为数

据控制组件。

[0151] 设备本机容器402还包括Java虚拟机(“JVM”)432。Java为MAF应用提供Java运行时环境。JVM 432在设备本机代码中实现,并且作为本机应用二进制文件的一部分嵌入(或编译)到MAF应用的每个实例中。JVM 432基于Java平台微型版(“Java ME”)连接设备配置(“CDC”)规范。在运行时架构400中,JVM 432包括业务逻辑434、模型436和Java数据库连接(“JDBC”)438。Java在MAF应用中实现业务逻辑434。受管理的Bean(“MBean”)是可以被创建以扩展MAF的能力(诸如提供用于处理从服务器返回的数据的附加的业务逻辑)的Java类。MBean由嵌入式Java支持执行,并且符合Java ME CDC规范。模型436包括连接业务逻辑组件和UI的绑定层。此外,绑定层提供执行逻辑,以调用web服务440,诸如远程托管的基于SOAP的web服务。这些服务通过Java层(JVM 432)访问。在MAF AMX中创作的应用特征通过数据控制组件来访问基于SOAP的数据服务。JDBC 438是使模型层能够通过创建、读取、更新和删除(“CRUD”)操作来访问加密的SQLite数据库406中的数据的API。

[0152] 设备本机容器402还包括应用配置442,应用配置442指的是允许应用配置被下载和刷新的服务,诸如用于配置服务器444的远程URL连接或web服务的URL端点。应用配置服务从服务器侧的基于WebDav的服务下载配置信息。

[0153] 设备本机容器402还包括提供凭证管理、单点登录(“SSO”)和访问控制的模块446。MAF通过“Oracle访问管理移动和社交”(Oracle Access Management Mobile and Social,“OAMMS”)身份管理器(“IDM”)SDK来处理用户认证和凭证管理。MAF应用执行离线认证,这意味着当用户在连接时登录到应用时,MAF在设备404上本地维护用户名和密码,从而即使到认证服务器的连接变得不可用,也仍然允许用户继续访问应用。MAF加密本地存储的用户信息以及存储在本地SQLite数据库406中的数据。在针对登录服务器进行认证之后,用户可以访问由该连接保护的所有应用特征。MAF还通过应用用户角色和特权来限制对应用特征(或应用特征的特定功能)的访问而支持访问控制的概念。对于远程服务的Web内容,MAF使用白名单来确保只有预期的URI可以在应用特征的Web视图416内打开(并且访问设备特征)。

[0154] 设备本机容器402还经由推送处理机414来实现推送通知,推送处理机414与包括在服务器侧资源412中的推送服务448进行通信并且使得MAF应用能够从通知服务器(诸如iOS或Android通知服务器)接收事件。Java层(JVM 432)处理通知处理。

[0155] 在运行时架构400中,设备本机容器402与加密的SQLite数据库406交互,加密的SQLite数据库406是保护本地存储的数据并且由模型层使用JDBC 438调用的嵌入式SQLite数据库。MAF应用生成这个加密的SQLite数据库406,该数据库是轻量级的跨平台关系数据库。因为数据库406被加密,所以如果设备丢失或被盗,则它保护数据。只有输入正确用户名和密码的用户才能访问这个数据库中的数据。

[0156] 图5是根据本发明的实施例的、用于在移动云基础设施中开发移动应用的系统500的框图。在系统500中,用户可以使用用户设备528经由基于web的工具在云基础设施506中开发和构建应用。在一个实施例中,应用可以通过空中(over the air)被下载到移动设备526上,从而消除对应用商店的需要。本机应用与在MCS 502中创建的后端504对话。在一个实施例中,图4的MAF运行时架构400可以被用来向移动设备526递送应用。在一个实施例中,应用的声明性语法通过空中被部署到移动设备526上,并且声明性语法在移动设备526上由图4的MAF运行时架构400解释。

[0157] 云基础设施506包括提供管理UI 516的MCS 502,可以通过管理UI 516来执行应用开发。MCS 502还包括生产环境512和测试环境514,在生产环境512和测试环境514中分别可以开发和测试移动应用。这些环境通过经由连接器与对应的后端504对话来提供生产/测试功能。首先在测试环境514中开发应用。一旦被发布,应用就移动到生产环境512。

[0158] 在一个实施例中,通过利用用户设备528通过安全524与MCS管理UI 516(也称为门户)进行通信来开发移动应用。MCS管理UI 516包括应用开发服务器518,应用开发服务器518可以经由MCS管理UI 516对接。在MCS管理UI 516中开发的应用可以通过与生产环境512和/或测试环境514通信而在用户设备528的浏览器上或在移动设备526上运行。在一个实施例中,当应用被部署在移动设备526上,移动设备526与测试环境514通信。但是,如果应用在移动设备526上被更新,则这些更新通过MCS管理UI 516执行。

[0159] 在系统500中开发的应用可以被构建为轻应用或重应用。重应用是完整的应用,诸如从应用商店下载的应用。轻应用是作为添加的特征被部署到已经部署的完整应用(即,托管应用)(诸如Oracle应用)的应用。托管应用充当保持轻应用的容器。重应用和轻应用二者都可以由安全容器进一步容器化,如本文参考图7所描述的。

[0160] 图6是根据本发明的实施例的、用于构建移动应用的系统600中的网络组件的框图。在系统600中,用户602(以下也记载为第一设备602)与MCS网站(在图6的示例实施例中被表示为“https://mcs-tenant-a.cloud.oracle.com”)交互以发起构建请求,并且设备604(以下也记载为第二设备设备604)与MCS网站通信以执行本机应用的无线安装(over the air install)。一般而言,无线安装包括下载诸如特性列表文件(具有扩展名“.plist”的“p-list”文件)之类的文件,该文件描述应用和从其下载对应的应用存档文件(具有“.ipa”扩展名并存储应用的文件)的位置,以及然后从该位置下载应用存档文件。

[0161] 第一设备602和第二设备604通过经由公共Oracle HTTP服务器(“OHS”)606与服务器610的MCS门户VM 612通信来与MCS网站交互。公共OHS 606是面向公众的HTTP服务器,它将流量指引到位于防火墙608后面的MCS门户VM 612。公共OHS 606实现WebGate,WebGate是用于Oracle访问管理器(“OAM”)的Web服务器插件,以拦截HTTP请求并将它们转发到对应的访问服务器以进行认证和授权。因此,公共OHS 606认证第一设备602的用户,将用户凭证传递给MCS门户VM 612,并终止与第一设备602的SSL连接。在图6的示例实施例中,第一设备602和第二设备604利用用于https的端口“443”来访问在“https://mcs-tenant-a.cloud.oracle.com”处的公共OHS 606。

[0162] MCS门户VM 612是标准的WebLogic服务器(“WLS”)应用,它的数据由租户模式服务614中的单个租户模式支持,并且它对应的应用开发客户端是使用Oracle快速启动企业工具包(“JET”)框架编写的。WebLogic服务器是Oracle公司开发的Java EE应用服务器。数据库模式是对象(例如,表、视图、所存储的过程等等)的容器,以将这些对象进行逻辑分组。

[0163] MCS门户VM 612是单个租户,并且其安全性经由Oracle Web服务管理器(“OWSM”,在本文参考图7描述)提供。因而,MCS门户VM 612在可信区中运行WLS。MCS门户VM 612处理第一设备602的请求,并且具有到租户模式服务614的连接。MCS门户VM 612还经由负载均衡器616连接到构建服务器场618(例如Mac迷你场)。在图6的实施例中,MCS门户VM 612使用开放端口80(或等价物)以用于到公共OHS 606/来自公共OHS 606、到负载均衡器616以及来自服务器场618中的各个服务器的HTTP通信。

[0164] 租户模式服务614与MCS门户VM 612交互并存储用于租户的应用数据、企业签名证书和供应简档。负载均衡器616将场任务路由到服务器场618中的服务器。路由可以最初以循环(round robin)方式执行。在图6的实施例中,负载均衡器616是来自F5公司的BIG-IP装置(appliance),BIG-IP装置使用开放端口80(或等价物)并且提供冗余。服务器场618包括处理构建作业(job)的多个服务器(例如,20个服务器)。它连接到用于存储应用二进制文件(例如,5TB)的文件管理器(未示出)。在一个实施例中,服务器场618的连接经由在服务器上本地运行的本地Tomcat实例来处理,并且构建工具和过程由本机OSX调用来处理。

[0165] 构建应用

[0166] 在一个实施例中,一旦第一设备602的用户已经创建应用并且希望产生本机二进制文件,用户就在MCS网站(例如,在“https://mcs-tenant-a.cloud.oracle.com/max/build”)处发起构建POST请求。POST是HTTP协议支持的、用于请求Web服务器接受并存储包含在请求消息的主体中的数据的请求方法。构建POST请求的有效载荷包括应用的应用标识符(“ID”)。公共OHS 606接收请求,终止SSL,针对OAM对用户进行认证和授权(假设用户已登录),将用户身份放入请求的HTTP报头中,并且将请求经过防火墙608转发到MCS门户VM 612的WLS服务器(例如,在“http://mcs-tenant-a.internal/max/build”运行的WLS服务器)。

[0167] MCS门户VM 612接收该请求,授权用户针对所请求的应用的特权,并且向租户模式服务614发送对应用数据、租户企业证书、加密的证书密码和租户供应简档的查询。一旦租户模式服务614返回所请求的项,MCS门户VM 612就在(存储在租户模式服务614处的)构建作业的表中创建新的条目,以记录构建尝试并且捕获对应的新构建记录的主键。MCS门户VM 612还创建针对在负载均衡器616后面(例如,在“http://max-mini-farm.internal/build/initiate”处)构建服务器场618的新的POST请求,从而将对应的参数(应用数据、签名证书和密码以及供应简档)传入请求的主体以及作业完成的回调URL中,其中回调URL在构建作业的表中编码对应构建记录的主键。以下功能提供了包括对应参数的构建POST请求有效载荷的示例:

[0168] applicationData:(app data)

[0169] signingCertificate:(cert)*

[0170] signingPassword:(password)*

[0171] provisioningProfile:(profile)

[0172] callbackUrl:http://mcs-tenant-

[0173] a.internal:3000/maxbuild/complete?jobId＝(BuildJobId)**

[0174] 在这个示例中,根据这个实施例,证书和密码由第一设备602的用户排他地为构建移动应用而创建(即,根据这个实施例,证书和密码不与除了构建移动应用以外的服务共享),并且端口3000不能被公开访问。

[0175] 负载均衡器616维护服务器场618中的健康服务器的列表。在一个实施例中,这经由在特定时间间隔(例如,每隔几分钟)执行健全性检查的健康检查来完成。在接收到构建作业请求时,负载均衡器616从列表中的健康服务器池中选择服务器,并将构建作业请求路由到该服务器(例如,将作业路由到“http://mac-mini1.internal/build/initiate”)。在一个实施例中,选择服务器根据用于构建具有同等复杂度的作业的循环过程。

[0176] 在一个实施例中,Tomcat web服务器在服务器场618中的选择的服务器上运行。

Tomcat web服务器接收构建作业请求,并且起动在异步小服务程序上运行的外部处理以防止输入/输出阻塞请求线程池。当处理完成时,Tomcat Web服务器创建对于请求有效载荷中的回调URL的POST请求。以下功能为这个新请求提供了示例有效载荷:

[0177] result:(如果成功,那么是success,等)

[0178] binaryKey:(jobId)

[0179] 如果事件已经成功,那么MCS门户VM 612接收新的请求,并用来自有效载荷的二进制键来更新构建作业的表中的对应记录。它还(例如,经由在Oracle业务智能企业版("OBIEE")11g推送上轮询或具有计划中的OBIEE 12c的异步小服务程序)通知客户端(即,第一设备602)构建作业已经完成并且产生具有用于下载应用的编码链接(例如,"https://mcs-tenant-a.cloud.oracle.com/max/native-application/(binaryKey)")的QR码。

[0180] 安装应用

[0181] 在一个实施例中,一旦第二设备604的用户扫描第二设备604上的QR码,就发起"无线"安装。扫描QR码打开QR码中编码的URL(例如,"https://mcs-tenant-a.cloud.oracle.com/max/native-application/(binaryKey)")。公共OHS 606接收该请求,终止SSL,针对OAM对用户进行认证和授权(假设用户已登录),将用户身份放入请求的HTTP报头中,并且将请求经过防火墙608转发到MCS门户VM 612的WLS服务器(例如,该WLS服务器在"http://mcs-tenant-a.internal/max/build"处运行)。

[0182] MCS门户VM 612接收请求,向用户授权针对所请求的应用的特权,确定发出请求的设备(第二设备604)的用户-代理(在这个上下文中为设备的OS框架,例如iOS相对于(vs)Android),识别第二设备604的平台(例如,iOS),并通过指向公共OHS 606来将该请求转发到对应的URL(例如,"https://mcs-tenant-a.cloud.oracle.com/max/native-application/plist/(binaryKey)"),该公共OHS 606又将该请求转发到MCS门户VM 612,以进行授权(如在本文描述的构建过程期间执行的那样进行授权,以确保允许用户下载应用)。MCS门户VM 612接收所转发的请求并且生成包括用于对应平台(例如,iPhone)的应用信息的特性列表文件(例如,iOS"p-list"文件)以及到二进制文件的链接(例如,"https://mcs-tenant-a.cloud.oracle.com/max/native-application/ios/(binaryKey)")。

[0183] 第二设备604然后提示用户他们是否想要安装应用。假设是,那么第二设备604通过指向公共OHS 606来跟随到二进制文件的链接(例如,"https://mcs-tenant-a.cloud.oracle.com/max/native-application/ios/(binaryKey)"),公共OHS 606又将请求转发到MCS门户VM 612,以进行授权(如在本文所述的构建过程期间执行的那样进行授权,以确保允许用户下载应用)。MCS门户VM 612接收该请求并且生成用于在负载均衡器616后面(例如,在"http://max-mini-farm.internal/download/ios/(binaryKey)"处)构建服务器场618的新构建作业请求。负载均衡器616(例如,经由循环过程)从健康服务器池中选择构建服务器场618中的服务器,并将构建作业请求路由到该服务器(例如,路由到"http://mac-mini1.internal/download/ios/(binaryKey)")。所选服务器上的应用服务器(例如,Tomcat)接收请求、确定对应的内容是否存在,并从网络(例如,从"Filer:/filer_mnt/generated_binaries/(binaryKey)/result.ipa")流传输二进制文件。负载均衡器616将被流传输的响应返回给MCS门户VM 612,MCS门户VM 612接收该响应并将该响应复制到它的请求的到第二设备604的输出流中。最后,第二设备604接收二进制文件并执行安装。

[0184] 图7是使用由移动安全套件700(诸如OMSS)提供的安全服务的实施例中的移动安全套件组件的框图。OMSS组件跨公司DMZ 740和企业内联网(或公司网络750)分布。在OMSS下,安全容器706(诸如来自Oracle公司的“Oracle移动安全容器”)被安装在移动设备702上,并且被配置为保持“容器化的”应用708(例如,已经被安全地链接到它们的特定容器的应用)。移动设备702还可以包括在安全容器706外部保持的其它个人应用704。

[0185] 安全容器706包括安全web浏览器712、文件管理器(未示出)、文档编辑器(未示出)和可选的安全移动邮件管理器710。安全移动邮件管理器710包括个人信息管理(“PIM”)应用,诸如经由“Microsoft Exchange ActiveSync”(“EAS”)协议与公司邮件服务器同步的邮件客户端、日历、联系人、任务和便笺。许多应用(诸如“Oracle Business Intelligence(业务智能)”(“BI”)、“Oracle Fusion Tap”、“Oracle Social Network(社交网络)”、“Oracle Enterprise Manager Cloud Control(企业管理器云控制)”、“Oracle WebCenter Spaces”等以及广泛的第三方企业应用)可以用安全容器706被容器化。在移动设备702上的容器化应用708内空闲(at rest)的所有数据都被加密。加密数据存储装置包括数据库、文件存储库、高速缓存和用户偏好。安全容器706使用诸如应用隧道(“AppTunnel”)714(如美国专利8,332,464中所描述的,该美国专利的完整公开内容并入本文)之类的安全信道来与公司DMZ 740后面的公司网络750通信。在一个实施例中,利用联邦信息处理标准(“FIPS”)批准的算法使用TLS/SSL来加密通过应用隧道714输送的数据。

[0186] 在一个实施例中,当Web浏览器或其它客户端程序对安全访问服务器(诸如来自Oracle公司的“Oracle移动安全访问服务器”(“MSAS”))进行未经认证的请求时,安全访问服务器将利用重定向到适当的安全容器进行响应。安全容器使用密钥层次结构来保护数据。所有密钥都是由从未被存储的用户凭证得到的。密钥层次结构涉及多个密钥,以支持数据的不同敏感性。例如,唯一密钥用于用户的认证证书,该唯一密钥允许在非常短的时间段内开放。浏览器高速缓存使用不同的密钥,该不同的密钥必须对于整个会话保持解密。主安全容器分发并管理用于用户的安全企业工作空间中的完整应用集合的密钥。

[0187] 与常规的移动虚拟专用网(“VPN”)解决方案相比,安全容器706具有至少三个显著的益处:设备信任相对于网关、安全容器密码相对于设备密码、以及安全容器AppTunnel相对于设备级VPN。OMSS将网络的Kerberos认证信任直接扩展到用户的设备,而不是在位于DMZ中的网关服务器处停止。OMSS比实现由VPN提供商提供的“受约束的委托”明显更高效和安全。受约束的委托解决方案不仅不太安全,而且设置和维护更麻烦。另外,当处理消费者设备和BYOD程序时,可用性和安全性之间的权衡被放大。公司IT需要强大的密码来保护BYOD设备上的公司数据。相反,用户想要简单的密码,或者最好根本没有设备密码,以使得他们可以容易地访问社交网络和其它消费者应用。要求设备密码对用户来说是令人沮丧的,因为他们经常将设备用于不需要企业认证的非企业用途。实施例通过仅需要密码以访问公司应用程序来提供在处理BYOD程序时在安全性和可用性之间的必要平衡。

[0188] 此外,设备级VPN在用户的设备和企业的网络之间提供可信的、安全隧道。但是,设备级VPN解决方案更适合诸如笔记本电脑之类的企业拥有的并且安全的端点设备,而不适合消费者移动设备。一旦移动设备VPN隧道向网络开放,设备上的任何应用就都可以访问这个安全隧道,从而造成严重的安全漏洞。但是,利用实施例,从移动设备702到公司网络750的连接仅存在于安全容器706和企业服务器之间。

[0189] 在移动安全套件700中,MSAS 716通常部署在公司DMZ 740中,并且为了高可用性和可扩展性,多个服务器实例可以被部署在负载均衡器后面。MSAS 716提供服务器和容器化的应用708之间的隧道连接。MSAS 716安排(broker)认证(强认证利用到“Oracle访问管理器”(“OAM”)722的HTTPS连接或者到Kerberos域控制器718的Kerberos连接)、授权、审计并且实现对其目的地(公司网络750中的资源,诸如web应用和web服务724)的SSO并且将请求代理到其目的地。MSAS 716充当由安全容器706和容器化应用708发起的隧道连接的终止端点。

[0190] MSAS 716支持来自Oracle公司的“Oracle API网关”(“OAG”)和来自Oracle公司的OWSM以向组织的REST API基础设施添加安全性、威胁保护和调节策略。SSO通过OAuth、OAM令牌、Kerberos和NT LAN管理器(“NTLM”)来支持。SAML通过与SAML身份提供者(诸如Oracle、CA或Ping身份)的Kerberos或OAM 722集成来支持。MSAS 716与OAM平台集成,并且支持对于向由OAM、OAG和OWSM保护的后端资源的SSO检索OAM和OAuth令牌。MSAS 716还通过对受PIN保护的Microsoft Active Directory(活动目录)执行公钥基础设施(“PKI”)认证来支持“虚拟智能卡”认证。数字证书在安全容器应用内被供应并且仅在成功的PIN验证后才能访问。MSAS与OAM的集成允许上下文感知的、基于风险的增强的(step-up)认证。

[0191] 移动安全套件700还实现了OWSM,OWSM是SOA套件的组件并且解决基于web服务的SOA安全性和管理。SOA基础设施的目的是允许消费者调用由提供者暴露的服务。OWSM为这种服务基础设施的策略管理和安全性提供解决方案。它通过由来自Oracle公司的“Oracle企业管理器”提供的集中式管理界面提供策略的可见性和控制。OWSM允许公司(1)集中式地定义和存储应用于构成SOA基础设施的多个Web服务的声明性策略,(2)通过可配置的代理来本地实施安全和管理策略,以及(3)监视运行时安全事件,诸如失败的认证或授权。它还通过允许在无需中断正在运行的业务过程的情况下实时地实施策略变更来提供响应安全威胁和安全违反的业务敏捷性。

[0192] 移动安全套件700还实现公司网络750内的“Oracle移动安全管理器”(“MSM”)720。MSM 720是在Oracle Linux或Red Hat企业Linux上运行的“WebLogic”受管理服务器。MSM 720与公司网络750中的Microsoft交换服务器(Exchange Server)728集成,以提供对企业电子邮件服务的访问。MSM 720还与LDAP服务器732集成,以供应用户、指派和管理用于移动设备管理和用于访问安全容器706的策略、管理应用目录、控制设备的远程锁定或擦除以及保护工作空间应用(擦除安全容器706移除用于工作空间应用的所有数据和配置)以及为安全容器设置访问控制策略。通过将策略模板与用户和用户组关联来将策略指派给用户。可用的策略控制包括设备限制、认证(认证频率、失败的尝试阈值、用于PKI的PIN强度)、目录(应用、URL、文件共享)、容器/应用(受损的平台、位置服务、离线状态、不活动持续时间、数据泄露预防(“DLP”))、时间访问(如果在时间窗口之外,那么锁定)、地理访问(如果在地理围栏(城市、州、国家)之外,那么锁定)、设备(特定于白名单的设备型号、指定最低OS级别)、浏览器(禁用地址栏、禁用下载)、文件浏览器(允许/禁止、禁用下载、指定文件服务器URL)、个人信息管理器(“PIM”、邮件服务器URL)、供应(邀请模板、PKI细节),等等。如果用户在多个组中并且具有多个策略,那么遵循具体规则来解析策略组合。

[0193] MSM 720维护EMM策略,然后EMM策略被关联到目录中的一个或多个用户组。MSM 720不执行任何用户或组管理,而是直接(无同步)利用来自目录存储库的这些身份和组。

MSM 720经HTTPS使用APNS和CGN来向设备发送通知。MSM 720还将WebDAV前端暴露给内部的启用CIFS/SMB的文件系统730或“Microsoft SharePoint服务器”，并且使得能够从客户端浏览内联网文件共享。

[0194] 随着越来越多的组织在社交网络上建立存在，IT部门需要对社交身份的支持，社交身份依赖于比企业身份更轻量级的安全标准，但是所述更轻量级的安全标准更好地适于社交网络的要求。例如，一些网站可能要求用户提供从Facebook或Google获得的访问令牌，以便对他们的服务进行认证。因而，移动安全套件700还实现OAMMS，OAMMS包括与现有后端身份管理基础设施对接的服务器。该服务器充当被支持的移动客户端应用和后端身份服务之间的中介。这将客户端应用与后端基础设施解耦，以使得后端基础设施可以被修改，而不必更新移动客户端程序。OAMMS包括以下功能：

[0195] • 利用OAuth标准的委托的授权。

[0196] • 移动服务将基于浏览器的(HTML5)和本机的移动应用连接到企业身份管理基础设施(通常是“Oracle访问管理平台”)。

[0197] • 互联网身份服务，该互联网身份服务让OAMMS在与流行的、基于云的身份认证和授权服务(诸如Google、Yahoo、Facebook、Twitter或LinkedIn)交互时被用作依赖方。通过部署OAMMS，向用户提供多个登录选项，而无需为每个身份提供者单独地实现访问功能。用户简档服务为用户自助服务功能(诸如自注册、简档维护、密码管理和帐户删除)、LDAP CRUD操作提供REST接口(客户使用相同的REST接口为应用构建图形UI)。用户简档服务还作为OAuth资源可用。

[0198] • 访问管理集成服务，用于通过由代理SDK提供的运行时REST接口来利用OAM 720。

[0199] 图8是根据本发明的实施例的、用于移动应用开发的流程图。在一个实施例中，图8(以及下面描述的图10)的流程图的功能由存储在存储器或其它计算机可读或有形介质中的软件实现并且由处理器执行。在其它实施例中，功能可以由硬件(例如，通过使用专用集成电路(“ASIC”)、可编程门阵列(“PGA”)、现场可编程门阵列(“FPGA”)，等等)执行，或者由硬件和软件的任何组合执行。基于云的移动应用开发的示例在于2015年6月29日提交的、标题为“CLOUD BASED EDITOR FOR GENERATION OF INTERPRETED ARTIFACTS FOR MOBILE RUNTIME”的美国临时申请No.62/186,080(代理人案号:88325-924721(165701US)，客户参考编号:ORA150600-US-PSP)中提供，该美国临时申请的公开内容通过引用结合于此。

[0200] 在810，生成应用定义向导。如本文使用的应用定义向导表示在利用一个或多个预定义的云可访问服务的移动应用的定义过程期间引导用户的一个或多个UI的集合。应用定义向导可以实现各自与应用定义过程的一部分关联的一个或多个工作流。在一个实施例中，应用定义向导可以提示或以其它方式引导用户指定应用默认内容，诸如应用标识符前缀、默认图标、启动屏幕、默认应用/特征模板、设置企业供应简档/密钥库等。

[0201] 在某些实施例中，应用定义向导可以提示或以其它方式引导用户指定应用名称、形状因子(诸如电话或平板设备)、导航类型(例如，空意味着单个特征或UI、作为主界面、导航栏(“NavBar”)、Spring/Nav组合等)以及任何应用偏好。

[0202] 在820，接收应用定义。如本文所讨论的，应用定义可以包括为了至少创建最低功能的移动应用所需的任何信息。在830，基于应用定义来生成移动应用。在一个实施例中，移

动应用在目标设备的模拟器中表示，并且可以包括在被解释时充当经编译的移动应用的一组定义。

[0203] 在840，生成特征选择向导。如本文所使用的特征选择向导表示在利用一个或多个预定义的云可访问服务的移动应用的开发过程期间引导用户的一个或多个UI的集合。特征选择向导可以实现各自与应用开发过程的一部分关联的一个或多个工作流。在一个实施例中，特征选择向导可以提示或以其它方式引导用户指定可以与移动应用一起使用的特征、UI模块、业务对象等。

[0204] 在某些实施例中，特征选择向导可以提示或以其它方式引导用户指定移动应用的第一屏幕的组件。组件可以从组件目录中选择。

[0205] 在某些实施例中，特征选择向导可以提示或以其它方式引导用户指定移动应用的其它屏幕的组件。这些其它屏幕可以形成一个或多个UI模块的部分。在某些实施例中，特征选择向导可以提示或以其它方式引导用户指定移动应用的一个或多个UI模块。UI模块表示可以相对于移动应用执行的处理器、任务或流程。UI模块可以从提供UI元素和页面流的内聚集合的UI模块目录或模板集合中选择。UI模块的一些示例是批准工作流、工作者任务、数据条目任务、报告构建者等。模板提供UI元素集合的预先设置的布置/绑定，使得用户仅需要配置这些UI元素以及绑定模板，而不必布置和绑定单独的UI元素。在一个实施例中，用户可以将他们自己的模板贡献给另一个用户可用的模板集合。用户可以配置或以其它方式指定表示UI模块的一系列页面。对于每个页面，可以像以前一样向用户呈现布局模板集合。每个布局模板可以具有若干方面，诸如选择辅助模板。

[0206] 在一些实施例中，特征选择向导可以提示或以其它方式引导用户指定移动应用的附加特征，诸如先前定义的业务对象。用户可以指定后端服务、API或连接器的什么资源要被每个组件、屏幕、UI模块等的UI元素使用或以其它方式与每个组件、屏幕、UI模块等的UI元素关联。

[0207] 在850，接收特征定义，并且在860生成数据绑定向导。本文使用的数据绑定向导表示在利用一个或多个预定义的云可访问服务的移动应用的数据绑定过程期间引导用户的现有UI的UI元素或一个或多个UI的集合。数据绑定向导可以实现一个或多个工作流，每个工作流与应用开发过程的一部分关联。在一个实施例中，数据绑定向导可以提示或以其它方式引导用户指定特征、屏幕、UI模块等如何被绑定到可以与移动应用一起使用的业务对象、服务、API等。在某些实施例中，数据绑定向导可以提示或以其它方式引导用户指定移动应用的业务对象。可以从移动应用可用的目录或服务、API等的集合中选择业务对象。

[0208] 在870，接收数据绑定定义。在各种实施例中，步骤840-870可以串行或并行执行。可以对移动应用的单独的元素或一组元素执行840-870中的单独的步骤。如图所示，用户可以重复特征定义和数据绑定的过程，以创建移动应用。

[0209] 在880，部署移动应用。用户可以利用部署在目标设备上的测试应用或作为部署在目标设备上的本机应用来测试应用。

[0210] 事务的自动保存

[0211] 当前，在大多数Web应用中，不存在由用户执行的显式的保存动作。相反，应用代表用户自动保存其内容。因而，应用需要确定何时特定保存边界发生，并且然后相应地执行自动保存。保存边界是被配置为触发应用的自动保存的条件(例如，时间实例/间隔、用户动

作、应用变量值、应用/系统事件，等等)。

[0212] 一些已知的系统基于简单的基于时间的控制功能来执行自动保存(例如，在设定的时间段之后自动保存)，但是在需要更多逻辑保存边界的撤销/重做动作的情况下不是理想的。一些已知的系统使用基于动作的边界，其中当用户执行动作时触发自动保存。这些系统可以响应于撤销/重做操作而提供自动保存功能，但是导致确定何时执行自动保存以及如何协调绑定到相同用户动作的模型变更的增加的复杂性。例如，当用户将屏幕上的一些文本绑定到后端服务时，对UI定义文件和绑定定义文件进行相应的变更，并且这些文件中的每一个文件都通过它们相应的模型进行修改。因而，提供由撤消/重做动作触发的自动保存需要协调这些模型变更，这导致增加的复杂性。模型是“真实的来源”，它是对实际数据的抽象，诸如包括描述屏幕上的UI组件的声明性语法的页面模式。

[0213] 另外，就性能和数据完整性而言，客户端需要向服务器确认是否成功执行了保存操作，以便客户端知道它是否需要重试保存操作。一些已知的系统在允许进一步的用户输入之前等待服务器响应。但是，如果与服务器的连接慢，那么这种等待会对于用户阻塞UI并且严重损害性能。一些已知的系统不等待服务器并且不阻塞客户端UI。但是，如果服务器发生故障，那么客户端常常无法恢复，因为服务器和客户端之间的状态不再匹配。

[0214] 相反，实施例实现将用户动作与客户端上的具体模型变更相关的事务系统，并且在确保数据完整性的同时提供基于对应动作的保存边界。此外，如果在尝试保存动作时发生服务器错误，那么实施例不需要等待服务器对保存操作进行响应(并且因此不阻塞客户端UI)，同时还允许自我修复。

[0215] 一个实施例定义由影响客户端侧模型的用户动作触发的客户端侧事务。事务是由单个用户动作造成的一系列相关变更。例如，对于需要对UI定义文件和绑定定义文件进行改变的、将屏幕上的一些文本绑定到后端服务的用户动作，作为针对这些文件中的每一个文件应用相应的模型变更的代替，一个实施例捕获涵盖对这两个模型的变更的事务，并且使用该事务来支持自动保存功能。

[0216] 在一个实施例中，当模型被修改时，将变更记录添加到当前事务。在用户动作完成之后，事务被提交并且修改被记录在变更记录中。模型的示例是页面模式，页面模式是描述当前屏幕上的UI组件的声明性语法。这种模型可以被实现为操纵定义屏幕的可扩展标记语言(“XML”)代码的应用编程接口(“API”)。用于这种模型的示例模型变更是：

[0217] some_component.setAttribute(“style”,“color:red”)

[0218] 其中“some_component”是页面模型中的UI组件并且变更是修改样式属性，以使它成为红色。用于这个模型的示例事务是一系列模型变更(例如，上面提到的对页面模式和对应绑定的变更)，这可以通过由用户点击按钮而造成。对于这个模型可以被配置为触发自动保存动作的用户动作的示例是用户手势，诸如按下按钮、输入一些文本、拖放UI的一些块，等等。例如，当完成用于开发应用的“新应用向导”时，用户可以修改该应用和它的屏幕模型，并且当用户点击“完成”按钮以完成开发应用时可以触发自动保存。

[0219] 一个实施例定义保存边界，以将本地内容自动保存到服务器的远程持久性存储库。持久性存储库是长期存储数据的存储装置(例如，文件系统、数据库等等)。这个实施例实现在两个分离的队列上的两个分离的生命周期。变更记录首先被应用于本地队列上的本地生命周期，以将变更提交到本地模型。这个生命周期发生得非常快，并且允许客户端UI以

接近瞬时的方式反映这些变更。变更记录还在远程生命周期下运行，在远程生命周期中，变更记录在远程队列中被排队，以便被发送到服务器。远程队列是管理被存储到服务器的远程持久性存储库中的内容的持久性队列。在远程队列上排队的变更记录将按照它们被创建的次序被“同步”发送到服务器。即，在一个实施例中，一次一个地处理远程队列上的变更记录，并且只有当先前的变更记录已经被成功记录在服务器上时，才将每个变更记录发送到服务器。这确保在服务器处维持变更的次序。

[0220] 当将变更保存到服务器的保存请求失败时，实施例知道远程队列处的它们需要重新发送被排队的保存请求的点。因而，实施例重试远程队列中失败的记录，并且然后继续处理后续记录。如果服务器失败持续存在，那么实施例转变到离线模式并继续在远程生命周期上记录变更记录。当与服务器的连接已经恢复或服务器不稳定性已经被校正时，这些记录然后将以正确的次序保存在服务器处。

[0221] 通过允许几乎瞬时的UI更新，实施例在不牺牲数据完整性的情况下改进客户端UI性能。例如，由用户执行的动作(包括复杂的动作，诸如创建复杂的工件)对于用户来说是立即完成的。另外，由于服务器保存是同步的，因此实施例可以处理服务器中断，并且确保数据不会丢失并且内容将具有与服务器的最终一致性。

[0222] 在一个实施例中，事务在用户动作被触发时被创建并且在动作完成时被提交或者被回滚。当系统完成处理动作的结果时，该动作完成。在这两个事件(用户动作和用户动作的完成)之间，响应于用户动作可以发生模型变更。例如，当用户动作是按压按钮时，系统对这种按压做出响应，并且当系统完成按压的处理时，动作完成。如果按钮被按压以便例如向页面的某个部分添加新的属性，但是添加该属性被认为是非法的，那么事务被取消。另一方面，如果属性实际上是合法的，那么事务被提交。

[0223] 作为另一个示例，当用户尝试通过“拖放”功能将按钮添加到屏幕时，如果用户可以成功地将按钮放到屏幕，那么事务完成。但是，如果用户尝试执行不可接受的动作(诸如在一个按钮内添加另一个按钮)，那么页面模型检测到这个动作是不可接受的并通知系统。这造成事务失败并且事务的所有相关活动(所有模型变更)都被回滚(还原)。在一些实施例中，可以响应于由用户手势发起的单个模型变更来进行附加的模型变更。例如，当用户在可接受的位置放下按钮时，系统可以(例如，默认地)自动向该按钮添加文本作为分离的/附加的模型变更。

[0224] 一个实施例实现如在事务数据库中提供的事务功能。事务数据库是数据库管理系统(“DBMS”)，在DBMS中，如果对数据库的写入事务没有正确完成(例如，由于电源或连接丢失)，那么这些事务可以被回滚。一些关系DBMS支持包括各自在数据库中读取和/或写入信息的一个或多个数据操纵语句和查询的事务。事务还可以被实现为嵌套事务，嵌套事务包括在其内的起动新事务(即，子事务)的语句。如果在事务的执行期间没有发生错误，那么事务被提交。事务提交操作应用事务内的所有数据操纵并将结果持久化(persist)到对应的模型。如果在事务期间发生错误或者用户指定了回滚操作，那么事务内的数据操纵不被持久化到模型。部分事务不能提交给模型，因为这将使模型处于不一致的状态。

[0225] 在一个实施例中，每个模型被配置为记录它对事务的变更，包括执行变更和撤销该变更的最小指令集。这种最小指令集可以用于实现撤销/重做动作。例如，对于由XML支持的页面模型，调用“some_component.setAttribute(...)”立即修改XML，而“[set

attribute“foo”on element“X”]”可以是可以用于实现撤消/重做动作的单个指令。

[0226] 在一个实施例中,当事务被提交时,它被发送到本地生命周期,在本地生命周期中事务按它们完成的次序被记录。当事务的变更被应用到对应的客户端模型并且任何必要的UI都被更新以反映这些变更时,该事务被认为“被记录”。一旦事务在本地被记录,它的变更记录就被添加到远程生命周期的远程持久性队列中。远程生命周期是需要被保存到服务器的事务变更记录的队列。这些记录一次一个地被处理,并指示内容如何在服务器上被持久化。如果将变更记录持久化到服务器完成,那么从远程持久性队列中移除该变更记录。但是,如果服务器保存失败或者在一段时间(例如,在尝试持久化操作之后的20秒)之后没有接收到响应,那么确定持久化操作已失败。在这种情况下,一个实施例使用指数退避算法来重试保存动作。例如,一个实施例试图在1秒之后尝试保存动作的第一次重试,并且如果第一次重试失败,那么等待2秒以尝试第二次重试,并且如果第二次重试失败,那么等待4秒以尝试第三次重试,依此类推。如果故障持续存在,那么一个实施例转变到离线模式,并且以更长的时间间隔(例如,每分钟)进行检查,以确定服务器连接是否已经恢复。这可以通过重试在持久性队列的头部的事务的保存来执行。

[0227] 实施例适用于任何基于web的应用(诸如云环境中的基于web的应用)。例如,当基于云的IDE中的基于web的应用的用户将对象拖放到UI中的设计画布上时,多个分离但逻辑上相关的文件被更新,并且对这些文件的变更被视为单个逻辑事务,以允许将组合动作的撤消/重做作为单个原子事务。在提供IDE的分布式系统中,作为特定事务的一部分的对象中的一些对象可以远离用户。因而,实施例确保对这些远程更新的协调从用户的角度来看是无阻塞的,并且发送到远程系统的变更在排序方面被协调,以使得可以重放失败的事务。

[0228] 一个实施例实现管理撤销历史列表的“TransactionManager”JavaScript模块,该撤销历史列表包括可以被撤消或重做的已提交的事务的历史。TransactionManager(事务管理器)是单体(即,仅创建这个JavaScript模块的一个实例)并且向调用者提供“startTransaction”JavaScript方法,使得它们可以开始记录变更。在任何时候,调用者都可以使用“getCurrentTransaction()”JavaScript方法来检索当前事务。一旦完成所有变更,“TransactionManager”就调用“commitTransaction()”JavaScript方法,以将变更记录保存到撤消历史中。如果在事务的生命期期间出于某种原因而存在错误,或者如果提交操作失败,那么“TransactionManager”调用移除被记录的变更的“rollbackTransaction()”JavaScript方法。为了撤销最新近的事务,可以调用“undo()”JavaScript方法。调用“undo()”通过撤销每个事务来重复回溯事务的历史。在任何时候,都可以调用“redo”JavaScript方法来重做最新近的撤销事务。“redo”可以被调用与“undo”被调用的次数一样多的次数。每个“undo”/“redo”触发类型为undo/redo的“TransactionEvent”JavaScript方法。不应当在事务内调用“undo”和“redo”JavaScript方法。

[0229] 一个实施例实现“ChangeRecord”JavaScript模块,作为用于构成事务的一部分的变更记录的基本JavaScript模块。每个事务可以包括一个或多个变更记录,并且所有记录都以原子方式被处理(即,都发生或者都不发生)。如果一个变更记录回滚,那么同一事务中的所有变更记录随后都被回滚。变更记录具有两个生命周期:本地和远程。

[0230] 本地生命周期包括在本地客户端上发生的非阻塞操作。非阻塞操作是在处理发生时不停止UI的操作,诸如不等待服务器响应来继续。存在着在客户端上发生的三个生命周

期JavaScript方法:“localCommit”、“localRevert”和“localReplay”。当最初提交变更记录时,“localCommit”被调用恰好一次。每当(例如,在撤消或回滚操作中)记录被还原时,就调用“localRevert”。每当(例如,在重做操作中)要求记录被重放时,调用“localReplay”。

[0231] 远程生命周期与本地生命周期并行发生,但是由于远程生命周期JavaScript方法是异步执行的(即,同时或并行),因此与本地生命周期JavaScript方法相比,它们可能被延迟。存在用于远程操作的两个JavaScript方法:“remotePersist”和“remoteRestore”。“remotePersist”将变更的结果持久化到远程服务器(即,将变更保存到数据库),而“remoteRestore”将(即,在变更之前的)原始状态恢复到远程服务器。

[0232] 在一个实施例中,变更记录由表示系统中的单个资源的唯一类型识别。资源可以是诸如屏幕、与屏幕相关的绑定、应用的导航流、应用元数据等之类的模型。类型还识别操作的JavaScript方法。在一个实施例中,使类型是非特定的,以便允许变更记录和监听者定义操作的约定(contract)。类型是包括变更的操作以及资源的ID的元组。操作还可以唯一地识别哪种类型的资源正在被修改。例如,在以下功能中,“resourceType”指示哪种类型的模型受到影响(即,与数据库中的表相关),并且“id”识别该资源的具体实例(即,该表中的行):

[0233] {

[0234] resourceType:page

[0235] id:pageId

[0236] }

[0237] 对于某些类型,持久化操作可以组合多个变更记录的远程生命周期。例如,如果两个持久化操作对于同一类型发生,假设第二个持久化操作将覆盖第一个持久化操作的数据,那么第一个持久化操作可以被跳过。在一个实施例中,单个变更记录影响单个资源,并且如果在同一个事务中修改了多个资源,那么使用多个变更记录。例如,当将屏幕上的一些文本绑定到后端服务的用户动作要求对UI定义文件和绑定定义文件进行变更时,一个实施例捕获涵盖对这两个模型的变更的事务并且将针对变更的相应变更记录添加到每个模型。

[0238] 一个实施例还提供了允许仅UI(UI-only)变更记录在本地生命周期中被添加的JavaScript方法。这种变更记录仅在本地生命周期中使用,以便在本地影响UI变更。仅要求仅UI变更记录的变更的示例是改变屏幕上的选项卡。这种变更不要求数据变更,并且因此无需将任何内容同步回服务器。对于附带发生的UI处理(例如,在事务之前/之后捕获UI状态),使用在本地生命周期事件产生(eventing)期间可用的事务元数据。例如,当用户切换屏幕上的选项卡时,如果用户执行撤销,那么实施例需要切换回选项卡。在仅UI事务中记录的元数据将捕获哪个选项卡处于活动状态,以便在撤消期间可以恢复该选项卡。这种元数据经由事务元数据(例如,“the_current_transaciton.setMetadata(CurrentTab,“some tab”)”)被存储。

[0239] “ChangeRecord::localCommit”JavaScript方法在本地提交事务。一些模型根据命令立即执行操作,因此没有任何内容要提交。在这些情况下,这个JavaScript方法可以用来通知模型何时提交完成。这个JavaScript方法可以可选地返回布尔值。在一些模式中,操作是试探性(tentative)执行的,并且可能没有实际变更被提交。在这些情况下,这个JavaScript方法返回“false”,以指示没有对底层模型进行变更。如果这个JavaScript方法

返回“false”,那么事务不会中止,但是这个特定的变更被事务忽略。其所有记录对于这个JavaScript方法都返回“false”的事务被静默处置,但是不回滚。仅UI的变更记录不受由这个JavaScript方法返回的内容的影响。

[0240] 当调用者修改模型上的特性时立即进行其变更的模型的示例是由XML支持的页面模型。对于这个模型,调用“some_component.setAttribute(...)”立即修改XML。由于模型在事务完成之前被修改,因此当事务完成时在提交操作期间没有工作要做。

[0241] 一些模型不立即修改它们的底层资源。一些模型在暂时性存储装置中捕获对该模型做出的变更,并且当在提交操作期间事务完成时,这些变更从该暂时性存储装置中读取并且被写入模型。其中变更没有必要的模型的示例是用户在同一事务中将按钮的文本从“a”改变为“b”、然后从“b”改变回“a”的情况。由于值“a”从事务的开始到事务被提交时没有变更,因此模型可以选择什么也不做,因为该操作的撤销将对模型没有任何影响。

[0242] “ChangeRecord::remoteCommit”JavaScript方法将变更持久化到远程服务器。当持久化操作完成时,这个生命周期JavaScript方法指示它已经完成。当持久化操作被拒绝时,这个生命周期JavaScript方法指示已经发生了错误,并且安排在将来重试该持久化操作。

[0243] 图9是一个实施例中的用于web应用开发的系统900的框图。用户通过使用运行与应用开发服务器904对应的应用开发网站910的web浏览器902来开发应用。由用户执行的动作通过请求/响应机制被反映在应用开发服务器904处,在该请求/响应机制中,每个变更被发送到应用开发服务器904,应用开发服务器904又发送回指示该变更是否已经在应用开发服务器904处被持久化的对应响应。

[0244] 在一些已知的系统中,除非从应用开发服务器904接收到指示上次进行的变更在应用开发服务器904处被持久化的响应,否则不允许用户继续与应用开发网站910交互以及进行进一步的变更。但是,当接收响应存在延迟时(由于例如应用开发服务器904正被加载、到应用开发服务器904的连接缓慢、在蜂窝/无线场景中与应用开发服务器904的连接较差,等等),这可能导致差的用户体验和整体性能。事实上,在一些已知系统中,用户可能需要在每次变更之后等待2-3秒才能够继续。当用户执行一系列动作并且用户然后打算撤消所有这些动作时,这个问题被放大,因为用户将不得不等待2-3秒来执行每个动作,然后等待另外的2-3秒来撤消每个动作,只是为了以没有变化而结束。

[0245] 相反,系统900通过实现本地队列906和远程队列908来在客户端侧(即,web浏览器902)维护事务处理系统。例如,当用户在应用开发网站910中做出变更“A”时,按事务(即,具有明确的起点和终点的工作的单元/块)来看待该变更。例如,“A”可以是在应用开发网站910上向屏幕添加表单。因而,每个离散操作都是它自己的事务。当事务被提交(即,事务已经进入系统并且用户已经决定执行对应的动作)时,“A”被放在本地队列906中。应用开发网站910一次一个地处理本地队列906的内容并执行对应的变更。例如,当“A”是本地队列906中最旧的未处理的项时,应用开发网站910从本地队列906中取出“A”并且在web浏览器902的屏幕上显示其效果。

[0246] 事务还在与本地队列906独立的远程队列908中排队。远程队列908也被排序并且其内容被一次一个地处理。例如,对于向Web浏览器902的屏幕添加表单的变更“A”,用户立即看到结果,而同时“A”可能仍然在远程队列908中等待以向应用开发服务器904传送。如果

应用开发服务器904慢(例如,如果需要花费5秒将变更上传到应用开发服务器904),那么作为等待“A”被成功发送到应用开发服务器904的代替,用户可以继续并且在本地执行其它变更“B”、“C”和“D”。在客户端侧,尽管服务器慢,但是变更“A”到变更“D”立即发生并且被记录。因而,本地队列906被非常快地处理。

[0247] 如果在应用开发服务器904处发生灾难性事件(例如,服务器停机),那么系统900转变到离线模式,并且尚未在应用开发服务器904处反映的变更停留在远程队列908中,以使得一旦应用开发服务器904复原,它们就可以向应用开发服务器904传送。即使Web浏览器902终止,远程队列908也保持这些变更。在应用开发服务器904再次可用之后,通过参考远程队列908,系统900识别哪个变更是将要向应用开发服务器904传送的下一个变更。远程队列908中的项仅在接收到来自应用开发服务器904的、指示该项已被成功存储在应用开发服务器904处的对应响应之后才被消除。由于项一次一个地被传送到应用开发服务器904,因此在每个实例,系统900知道哪些项已经被成功地传送到应用开发服务器904。

[0248] 在一个实施例中,当最新近的变更保存资源的整体时,可以从本地队列906和/或远程队列908中消除对该资源的先前变更。例如,在一系列变更“A”到“D”中,如果“B”和“D”修改相同的资源,并且“C”和“D”是独立的,那么可以从队列中消除“B”。这种资源的示例是web浏览器902上的UI,在该UI中将一个表单字段移动到另一个表单字段修改相同的资源。

[0249] 在一个实施例中,当处理本地队列906和/或远程队列908时,相应的指针被移动,而不是移动事务本身。在一个实施例中,在远程队列908中,撤销操作与其它事务一样被调度。但是,如果撤销事务被添加到远程队列908,而对应的原始事务仍在远程队列908中等待,那么这两个事务可以在远程队列908中被取消。

[0250] 在一个实施例中,从一开始(例如,到达应用开发网站910的第一时间)就在本地队列906中维护撤销历史。在这个实施例中,本地队列906是持久对象。

[0251] 图10是根据本发明实施例的自动保存功能的流程图。

[0252] 在1010,客户端设备的web浏览器接收由与对应于服务器的网站交互的用户执行的用户动作。网站可以是应用开发网站,并且服务器可以是应用开发服务器。

[0253] 在1020,确定与用户动作对应的变更记录,并且在1030,变更记录在第一队列中排队,以提交对本地模型的对应变更。在一个实施例中,第一队列是有序的持久性队列,该持久性队列维护用于在与网站交互中执行撤消和重做操作的变更记录的历史。

[0254] 在1040,变更记录在与服务器通信的第二队列中排队,以在服务器处持久化变更记录。在一个实施例中,第二队列是有序队列,在第二队列中一次一个地处理变更记录,并且只有当第二队列中的先前变更记录已经成功地记录在服务器上时,每个变更记录才被发送到服务器。变更记录在向服务器成功传送之后,从第二队列中被移除。第二队列通过请求/响应机制与服务器进行通信,在该请求/响应机制中,第二队列中的每个变更记录被发送到服务器,服务器进而发送回指示变更记录是否已经在服务器处被持久化的对应响应。

[0255] 在一个实施例中,变更记录反映在网站中由用户动作造成并被添加在构成相关模型变更的事务中的模型变更。当事务被提交时,事务中的变更记录被放在第一队列中。如果在事务的执行期间没有发生错误,那么该事务被提交。网站一次一个地处理第一队列的内容并执行对应的变更。当事务被记录时,事务中的变更记录被放在第二队列中。当事务的变更被应用于对应的客户端侧模型并且任何必要的用户界面被更新以反映对应的变更时,事

务被记录。

[0256] 一个实施例确定在时间阈值到期之后没有从服务器接收到对请求的响应，并转变到离线模式，同时保留第二队列的内容并继续在第二队列中排队新的变更记录。然后，实施例确定与服务器的通信被恢复，转变回在线模式，并且恢复在服务器处持久化第二队列的内容。即使当Web浏览器终止时，第二队列也保持其内容。

[0257] 一个实施例确定变更记录保存资源的整体并且将先前对资源的变更记录从第一队列和第二队列中移除。

[0258] 在一个实施例中，当撤消事务被添加到第二队列而对应的原始事务仍在第二队列中等待时，撤销事务和对应的原始事务二者都从第二队列中被取消。

[0259] 如所公开的，实施例提供了事务自动保存系统，该事务自动保存系统允许客户端侧变更是几乎瞬时的，同时还允许即使在服务器连接有问题或在客户端执行撤消/重做动作的情况下，也在远程服务器处持久化这些变更，而不会丢失数据或造成客户端与服务器之间的不一致。因而，由于不需要在将用户变更持久化到远程服务器的同时阻塞用户，所以实施例大大增强了性能并改进了用户体验。

[0260] UI状态的恢复

[0261] 当前，一些事务系统提供撤销和重做功能，以还原或重新应用模型变更。但是，虽然还期望恢复UI状态(即，用户正在看什么UI)，但是UI状态信息可以不是模型数据的一部分，并且因此恢复模型可能不会自动恢复UI状态。例如，用户可以在系统中的页面“X”上，对页面“X”进行模型变更，然后切换/导航到页面“Y”。如果用户对针对页面“X”的模型变更执行撤消，那么将视图切换回页面“X”将是非常期望的，因为这是用户进行模型变更的地方。但是，仅仅撤销对页面“X”的模型变更并不会自动将用户从页面“Y”带到页面“X”。随着系统变得更加复杂，优选地需要恢复的UI状态的量也增加。

[0262] 一些已知的系统要么根本没有解决这个问题(即，在撤销/重做时UI没有改变)，要么仅在一定程度上恢复UI状态。例如，在一些已知的系统中，当接收到撤消或重做动作时，用户被带回到某个主/默认UI，而不是特定于撤销/重做动作的详细UI。

[0263] 一些已知的系统试图检查变更，以确定向用户显示什么UI。但是，实现这个功能使系统复杂化并且难以维护。

[0264] 与已知系统相反，实施例提供了事务系统，在该事务系统中当事务被回滚或者随后被重新提交时(即，撤销/重做功能中的典型操作)，UI状态可以被恢复。一个实施例在提交事务之前和之后都存储UI状态和模型状态，并且当对事务执行撤消或重做操作时使用所存储的状态来恢复适当的UI状态和模型状态。因而，即使对于有大量UI状态需要恢复的复杂系统，实施例也使UI恢复在事务环境内可管理并且是自动的。

[0265] 在一个实施例中，在撤销动作时，用户看到在执行该动作之前他们正在查看的相同UI。一个实施例对于重做提供了类似功能。即，在重做已经被撤销的动作时，用户看到他们在执行该动作之后查看的相同UI。因而，通过防止用户丢失UI内的上下文，实施例显著地改进了用户体验。

[0266] 实施例是可扩展的，以允许UI的任何块参与恢复功能并且不要求UI理解事务系统的语义。即，UI不需要理解事务实际执行了什么模型变更、事务对系统意味着什么，或者事务一般而言如何工作(例如，提交操作、回滚操作等等是如何工作的)。

[0267] 在一个实施例中，当在浏览器上渲染网页时，网页的组件被初始化。即，当网页被渲染时，UI元素/角色(actor)首先被下载到浏览器，并且然后被执行。在这个实施例中，系统中的UI角色可以在初始化时向事务框架注册。注册还可以是动态的，使得在工作会话期间创建的UI可以向事务框架注册，以及然后在UI被移除时注销。

[0268] 在一个实施例中，当事务开始时，事务框架查询已注册的UI并检索它们的当前UI状态的快照。在一个实施例中，UI状态包括由事务导致的变更的语义含义。事务所涉及的UI角色知道如何恢复这种UI状态，因为这些角色起初创建了UI状态。以下是可以被包括在UI状态中的用来记录按钮的当前颜色为蓝色的示例功能：

[0269] {

[0270] “color”:“blue”

[0271] }

[0272] 然后事务提交并更新系统中的对应模型。一旦这些变更反映在UI中，就再次查询已注册的UI，并且取得它们的当前UI状态的另一个快照。在一个实施例中，快照不指示它们在事务内的什么生命周期中(例如，事务的生命周期状态是提交、撤消、还是重做等等)。例如，不管用户执行动作、撤消操作还是重做操作等，UI都取得它的当前快照。在一个实施例中，UI状态的“事务前”和“事务后”快照与对应的事务变更记录一起被存储。

[0273] 在一个实施例中，变更记录独立于对应的模型状态被存储，并且存储用于修改该模型状态的指令。假定模型状态在任何给定的时间点是正确的(例如，在任何给定的时间点匹配提供给用户的UI)。在一个实施例中，变更记录被存储在“撤消”堆栈中。每个变更记录包括可以执行对应动作或撤销该动作的指令集。因而，当用户撤消动作时，最后一个变更记录从撤消堆栈中被检索并且被调用以执行对应的撤销指令。

[0274] 在一个实施例中，当模型变更被应用于(例如，通过执行撤销操作)随后被回滚的事务时，向注册的UI提供当事务开始时取得的它们对应的UI快照，并且提示UI恢复它们对应的UI快照。类似地，当模型变更(例如，通过执行重做操作)在随后被重放的回滚事务中被恢复时，注册的UI被恢复到当提交事务时所取得的它们对应的UI快照。

[0275] 在一个实施例中，恢复UI的一个块(piece)的快照不影响UI的其它块。例如，当快照恢复时，一个UI块不会对另一个UI块产生事件(event)，这是因为在该另一个UI块自己的生命周期之外更新该另一个UI块可能引起不可预测的行为。因而，在一个实施例中，受影响的UI的每个块都独立于其它UI块来注册和管理其UI状态、不对其它UI块的事件作出反应，并且在该过程期间不向其它UI块发送任何事件。

[0276] 实施例适用于支持撤销/重做功能的任何系统或web应用。

[0277] 图11示出了根据一个实施例的示例UI 1100。UI 1100可以用于例如配置/设计移动应用或另一种类型的应用。UI 1100包括三个UI块：模板选择块1102、模板块1104和组件选择块1106。在一个示例中，模板选择块1102可以提供用于选择模板A 1110、模板B 1112或模板C 1114的选项。一旦在模板选择块1102中选择了模板选项，所选模板就显示在模板块1104中。在一个实施例中，模板选择块1102中的与模板块1104中的所选模板对应的UI组件可以在模板选择块1102中被强调。

[0278] 模板块1104可以由组件选择块1106中提供的组件进一步定义和/或修改。在组件选择块1106中提供的选项可以与模板块1104中示出的模板的选择相关。例如，组件选择块

1106可以提供用于向模板块1104添加按钮D 1116、按钮E 1118或按钮F 1120的选项。在一个示例中，在组件选择块1106中选择对应选项时，可以在模板1104中示出按钮1108。在一个实施例中，组件选择块1106中的与在模板块1104中添加的按钮对应的UI组件可以在组件选择块1106中被强调。

[0279] 在一个实施例中，虽然UI 1100中的UI块彼此相关，但是每个UI块的UI状态独立于其它UI块被存储，并且每个UI块的恢复不影响其它UI块的恢复或者对其它UI块的恢复产生事件。因而，可以独立地恢复每个UI块，并且UI块不接收来自其它UI块的事件或者向其它UI块发送事件。例如，如果模板选择块1102中被强调的UI组件与模板块1104中选择的模板对应，那么模板选择块1102的UI状态包括这种强调信息，但是它的恢复不对模板块1104产生事件以示出对应的模板。在这种场景中，模板块1104的UI状态被独立地保存和恢复，以示出适当的模板，由此避免需要在两个相关的UI块之间进行相互关联的事件产生(eventing)。类似的独立恢复功能针对模板块1104和组件选择块1106被实现，模板块1104和组件选择块1106保持相互关联的信息但是在恢复时不对彼此产生事件。因而，UI 1100的每个UI块独立于其它UI块存储它自己的状态，因此没有UI块需要等待另一个UI块以进行恢复。

[0280] 在一个实施例中，UI的所有块都可以保存它们相应的状态，但是只有受动作影响的UI的那些块才需要被保存/恢复。在一个实施例中，一些UI块可能不知道动作在做什么(即，动作的效果将是什么)，因此，即使每个可见UI块不受该动作的影响，它的状态也被保存/恢复。在一个实施例中，如果在执行动作时UI块不可见，那么这种UI块知道它不受该动作的影响，因为它在动作被执行的时间点甚至没有被显示。因而，在动作期间不可见的UI块的状态可以不需要被保存/恢复。

[0281] 以下提供对于具有主内容区域和侧边栏的页面保存和恢复UI的示例，其中侧边栏应当具有比主内容区域稍暗的背景色。该页面可以以主内容区域为“灰色”并且侧边栏为“深灰色”开始。主内容区域的背景色可以如下被改变为“红色”：

[0282] • 用户做出改变主内容区域的背景色的手势，并且开始对应的事务。

[0283] • 框架首先记录所有的UI状态。假设仅存在两个UI，这两个UI包括用于主内容区域的拾色器和用于侧边栏的拾色器，每个拾色器根据以下功能来记录它的当前状态：

[0284]
```
主内容拾色器:
{
    “背景色”: “灰色”
}
侧边栏拾色器:
{
    “背景色”: “深灰色”
}
```

[0285] • 主内容拾色器包括知道当它的颜色改变时，侧边栏拾色器也应当自动改变为主内容拾色器的新颜色的较暗版本的代码。因而，主内容拾色器告诉侧边栏拾色器使其颜色

变成“深红色”。

[0286] • 事务被提交并且模型状态变为：

[0287] 主内容颜色:红色

[0288] 侧边栏颜色:深红色

[0289] 如果用户随后撤销了这个事务,那么：

[0290] • 模型状态被还原为：

[0291] 主内容颜色:灰色

[0292] 侧边栏颜色:深灰色

[0293] 在这个时候,模型是正确的并且反映撤销动作,但是拾色器仍然显示错误的“红色”颜色。

[0294] • 告知UI块(即,拾色器)将它们的状态恢复到事务之前的状态,即：

[0295]
```
主内容拾色器:
{
    “背景色”： “灰色”
}
侧边栏拾色器:
{
    “背景色”： “深灰色”
}
```

[0296] 主内容拾色器然后将其状态恢复回“灰色”,并且侧边栏拾色器将其状态恢复回“深灰色”。在恢复这些状态时,改变主拾色器不再自动告诉侧边栏拾色器成为主内容拾色器的恢复颜色的较暗版本。即,不存在类似于正常UI变更的“事件产生”那样的“事件产生”,并且每个UI块自己管理其状态。在这个实施例中,UI块可以读取它们对应的模型,以知道它们的状态是什么。但是,跟踪UI块的UI状态还跟踪/存储可能没有记录在对应模型中的UI变更,诸如拾色器是否应当可见。

[0297] 图12示出了根据一些实施例的示例提交流1202、示例撤消流1204以及示例重做流1206。

[0298] 在提交流1202中,当事务发生时,在事务开始之前(即,在“INIT”状态下),在1208记录对应的模型状态和UI状态。这些模型状态和UI状态在图12中分别被表示为“INIT MODEL”和“INIT UI”。然后,在1210,提交前状态向外部监听者提供执行在提交状态之前的动作的机会。在1212,事务结束(即,事务在“COMMIT”状态提交),并再次记录对应的模型状态和UI状态。这种模型状态和UI状态在图12中分别被表示为“COMMIT MODEL”和“COMMIT UI”。

[0299] 在撤消流1204中,撤销(还原)操作试图恢复在事务之前的UI状态和模型状态(即,在1208捕获的“INIT MODEL”和“INIT UI”状态)。为此,在1214,“INIT MODEL”状态在还原操作之前在预还原状态处被恢复。在1216,执行还原操作。还原操作包括执行撤消动作。例如,

如果文本从“A”变为“B”，然后指示撤销动作，那么还原操作将“B”改变回“A”。在还原操作完成之后，在1218，“INIT UI”状态在还原后状态处被恢复。

[0300] 在重做流1206中，重做(重放)操作试图在事务提交之后(即在1212处捕获的“COMMIT MODEL”和“COMMIT UI”状态)恢复UI状态和模型状态。为此，在1220，在重做操作之前在预重放状态处恢复“COMMIT MODEL”状态。在1222，重放操作被执行。重放操作包括执行重做动作。例如，如果文本从“A”改变为“B”，然后撤消动作将“B”改回“A”，那么重做动作再次将“A”改回“B”。在重放操作完成之后，在1224，“COMMIT UI”状态在重放后状态处被恢复。

[0301] 图13是根据本发明的实施例的UI恢复功能的流程图。

[0302] 在1310，接收由与UI交互的用户执行的动作。用户动作的示例是用户手势，诸如按压按钮、输入一些文本、拖放UI的某个块，等等。在一个实施例中，动作由用户的客户端设备的web浏览器接收，并且用户与对应于托管UI的服务器的网站交互。在一个实施例中，网站是应用开发网站，并且服务器是应用开发服务器。

[0303] 在1312，基于动作来确定事务，其中事务被配置为修改与UI对应的模型。在一个实施例中，事务构成多个相关的模型变更。

[0304] 模型的示例是页面模式，页面模式是描述当前屏幕上的UI组件的声明性语法。这个模型可以被实现为操纵定义屏幕的XML代码的API。用于这个模型的示例模型改变为：

[0305] some_component.setAttribute(“style”,“color:red”)其中“some_component”是页面模型中的UI组件，并且变更是修改样式属性以使其成为红色。用于这个模型的示例事务是一系列模型变更(例如，上面提到的对页面模式以及对应的绑定的变更)，这可以是由用户点击按钮造成的。

[0306] 在1314，存储UI的第一UI状态和模型的第一模型状态。

[0307] 在1316，事务被提交。在一个实施例中，如果在事务的执行期间没有发生错误，那么事务被提交。

[0308] 在1318，基于第一用户交互来确定撤消事务。

[0309] 在1320，UI被恢复到第一UI状态，并且模型被恢复到第一模型状态。在一个实施例中，第一模型状态在撤消事务之前被恢复，并且第一UI状态在撤消事务之后被恢复。在一个实施例中，UI包括两个或更多个UI块，并且第一UI状态包括用于UI的每个块的UI状态。在一个实施例中，UI的每个块的UI状态独立于UI的其它UI块被存储和恢复。

[0310] 在一个实施例中，在事务提交之后并且在第一用户交互之前存储UI的第二UI状态和模型的第二模型状态。

[0311] 在一个实施例中，基于第二用户交互而进一步确定重做事务，然后UI被恢复到第二UI状态并且模型被恢复到第二模型状态。在一个实施例中，在重做事务之前恢复第二模型状态，并且在重做事务之后恢复第二UI状态。

[0312] 如所公开的，实施例提供了事务系统，该事务系统使用在提交事务之前/之后所存储的UI状态和模型状态的快照，并且当执行撤销/重做操作时恢复适当的UI状态和模型状态。在一个实施例中，在撤消动作后，向用户提供他们在执行该动作之前正在查看的相同UI。在一个实施例中，在重做动作后，向用户提供他们在执行该动作之后正在查看的相同UI。因而，实施例显著改进了用户体验并且改进了UI设计过程的效率。

[0313] 本文具体地示出和/或描述了若干实施例。但是，将理解的是，在不背离本发明的

精神和预期范围的情况下,所公开的实施例的修改和变型被上述教导涵盖并且在所附权利要求的范围内。

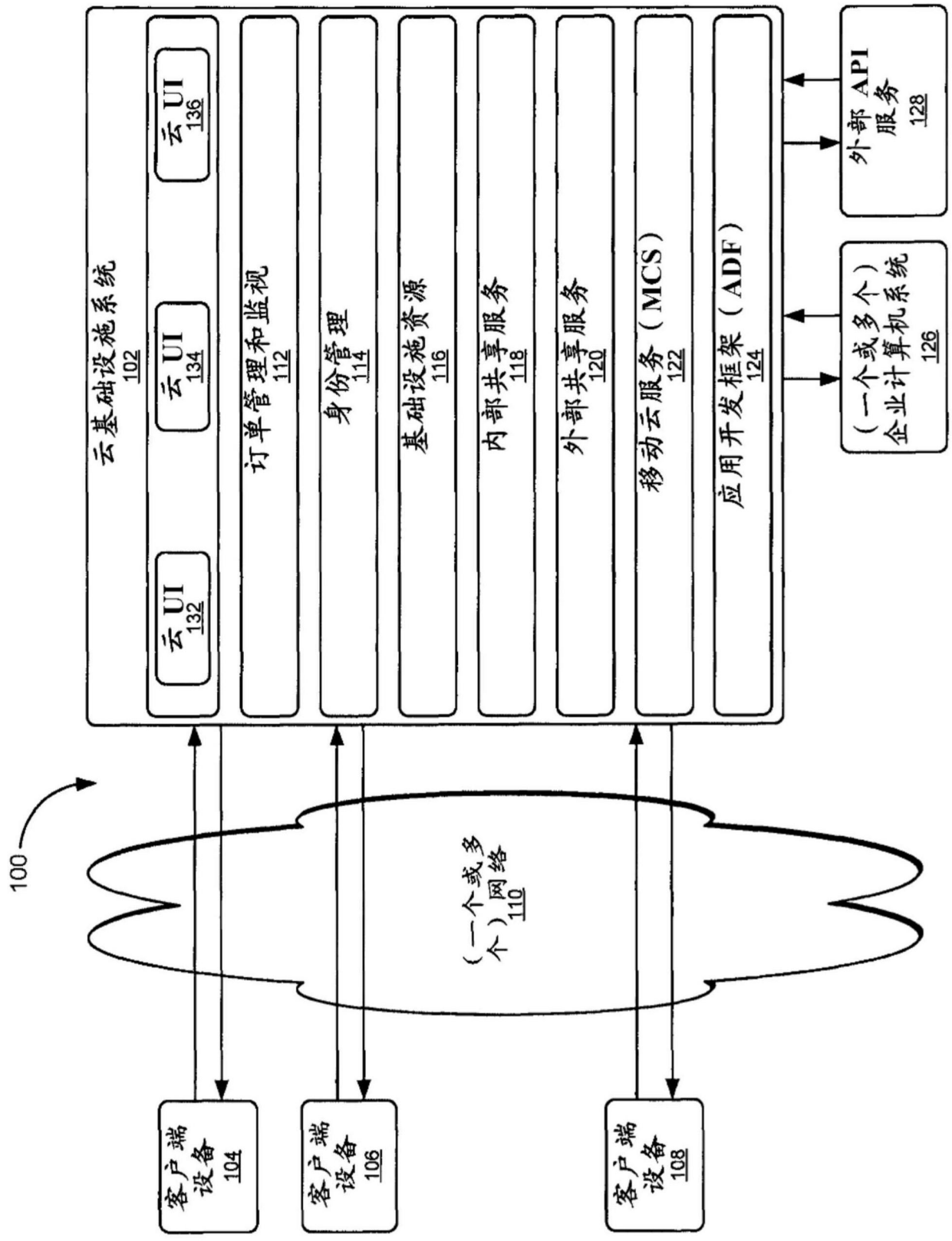

100
云基础设施系统
102
云 UI
132
云 UI
134
云 UI
136
订单管理和监视
112
身份管理
114
基础设施资源
116
内部共享服务
118
外部共享服务
120
移动云服务（MCS）
122
应用开发框架（ADF）
124
外部 API
服务
128
（一个或多个）
企业计算机系统
126
（一个或多个）网络
110
客户端
设备
104
客户端
设备
106
客户端
设备
108

图1

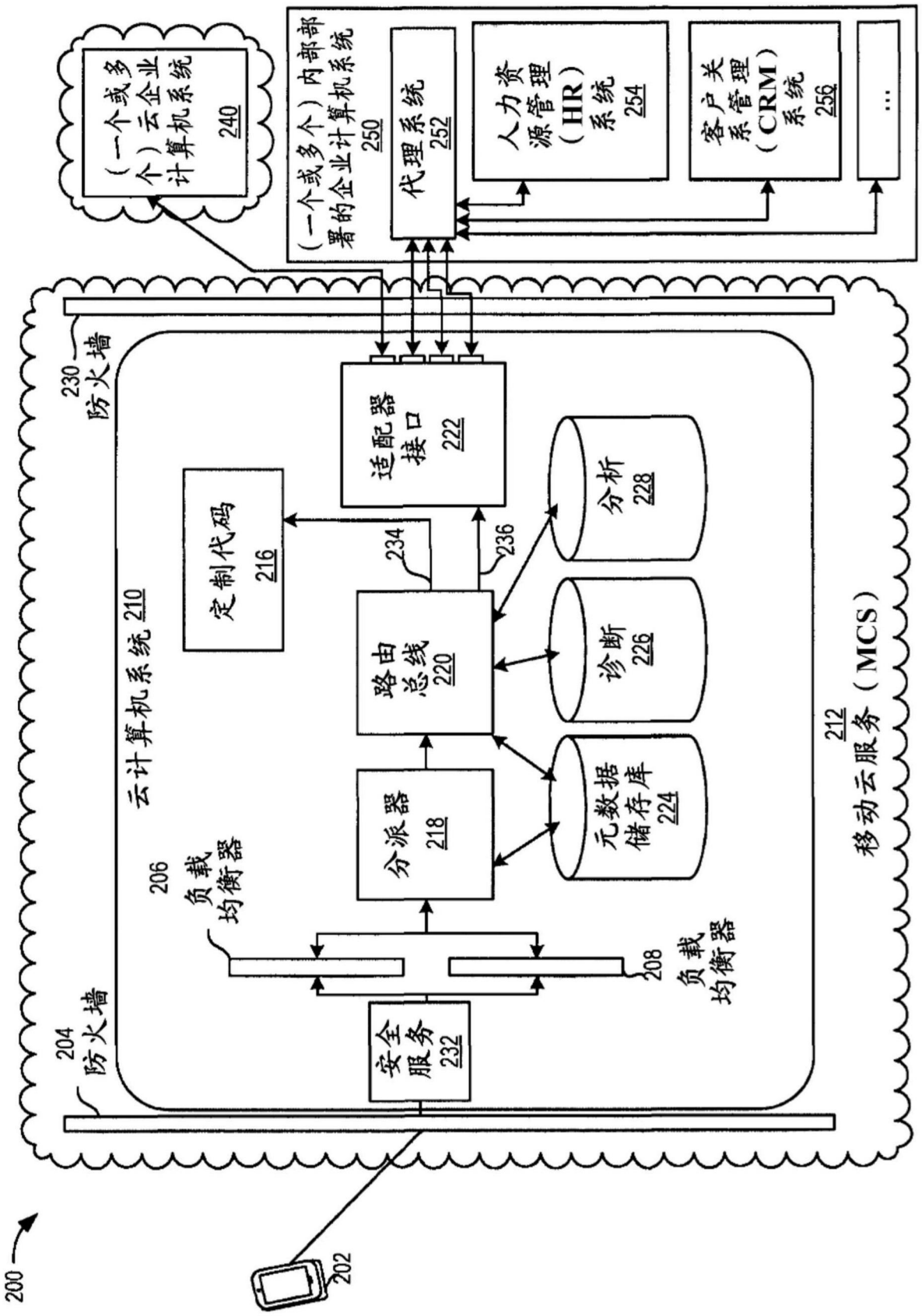

200
202
移动云服务（MCS）
212
云计算机系统 210
防火墙
230
防火墙
204
负载
均衡器
206
负载
均衡器
208
安全
服务
232
分派器
218
路由
总线
220
定制代码
216
适配器
接口
222
234
236
元数据
储存库
224
诊断
226
分析
228
（一个或多个）云企业
计算机系统
240
（一个或多个）内部部
署的企业计算机系统
250
代理系统
252
人力资
源管理
（HR）
系统
254
客户关
系管理
（CRM）
系统
256
...

图2

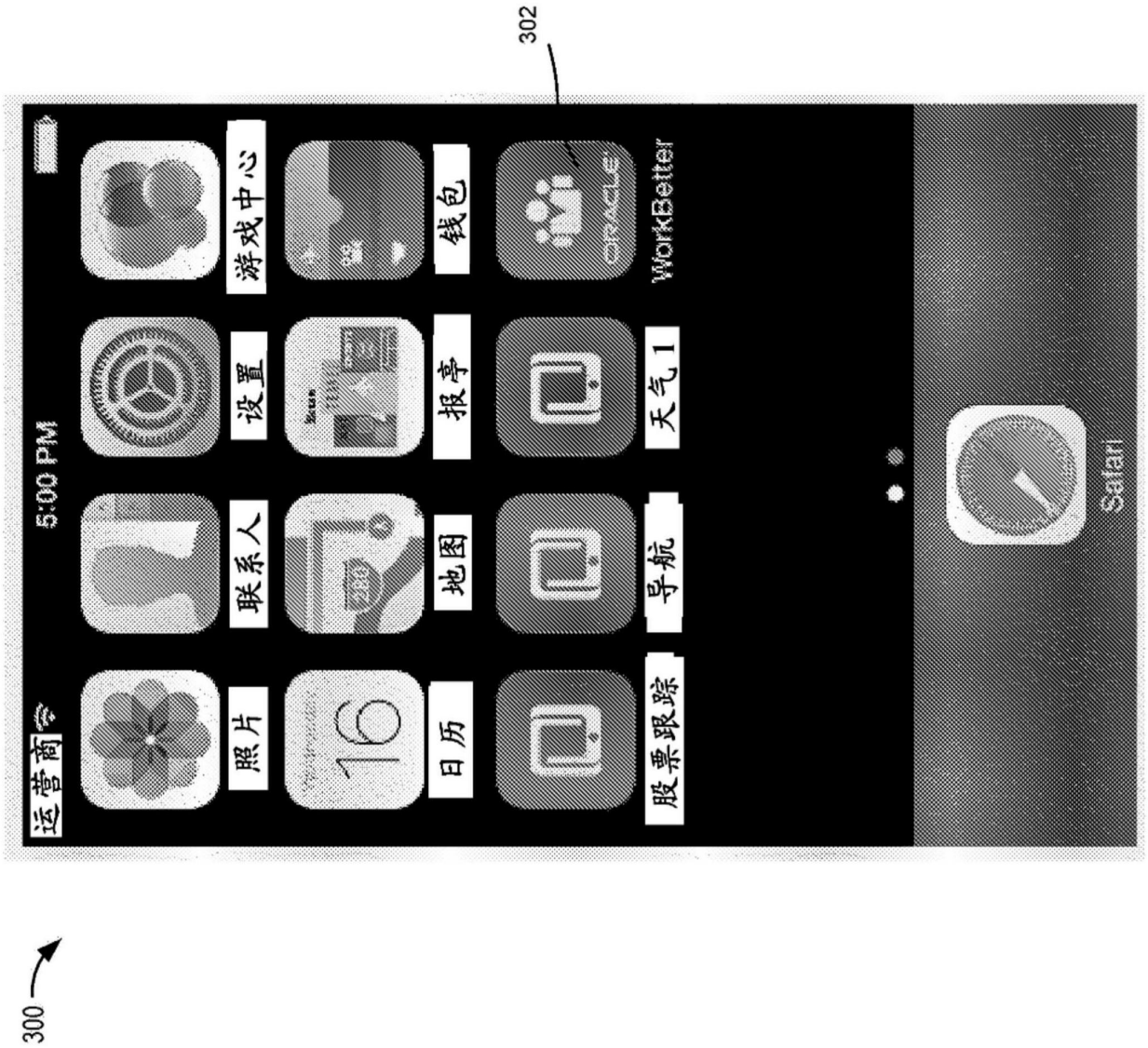

302
运营商
5:00 PM
照片
联系人
设置
游戏中心
日历
地图
报亭
钱包
股票跟踪
导航
天气 1
WorkBetter
ORACLE
Safari
300

图3A

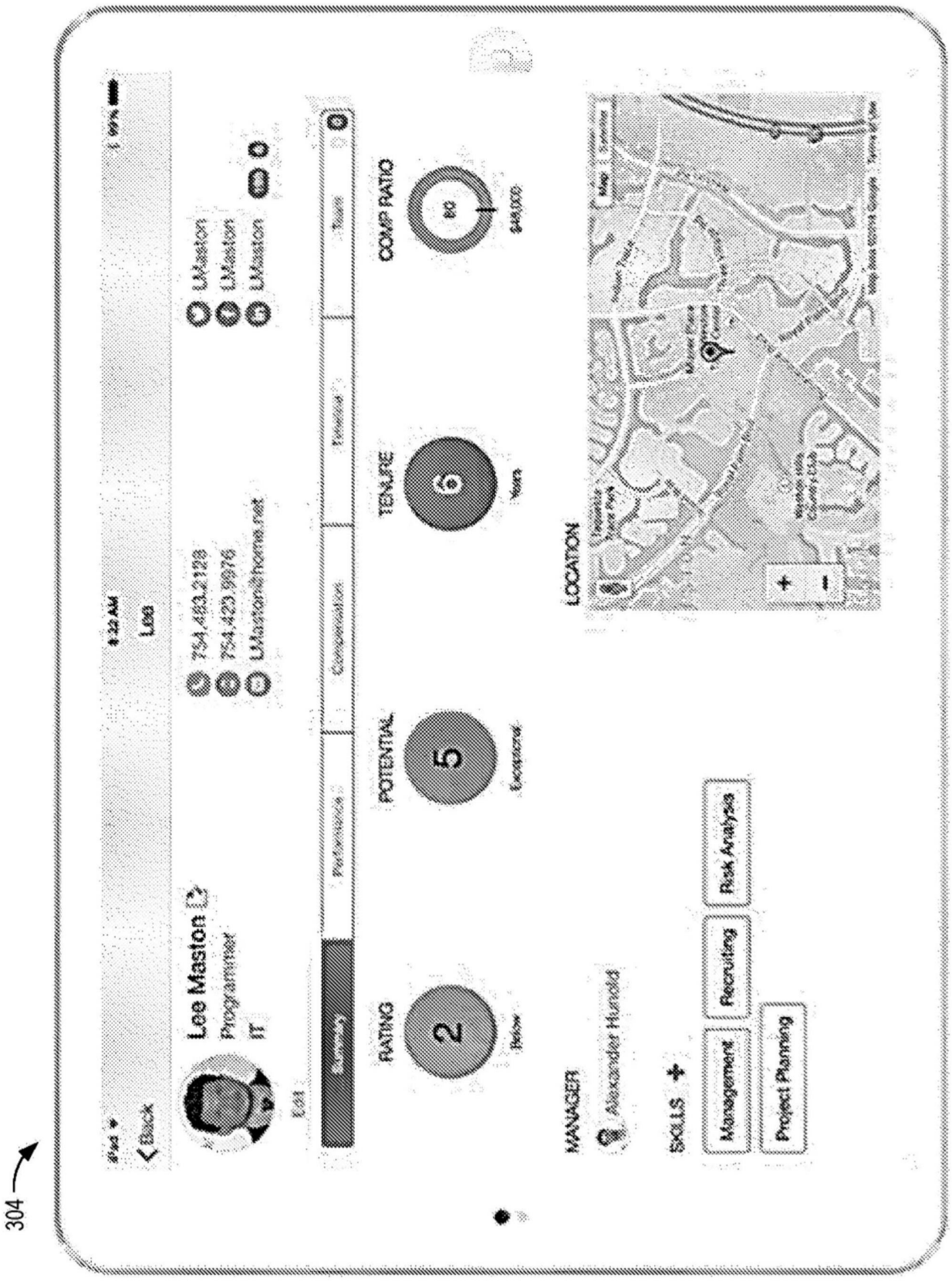

304
Back
Lee
Lee Maston
Programmer
IT
734.463.2128
734.420.9976
LMaston@home.net
LMaston
LMaston
LMaston
Summary
Performance
Compensation
Talent
Team
RATING
2
Below
POTENTIAL
5
Moderate
TENURE
6
Years
COMP RATIO
80
$48,000
MANAGER
Alexander Hunold
SKILLS
Management
Recruiting
Risk Analysis
Project Planning
LOCATION

图3B

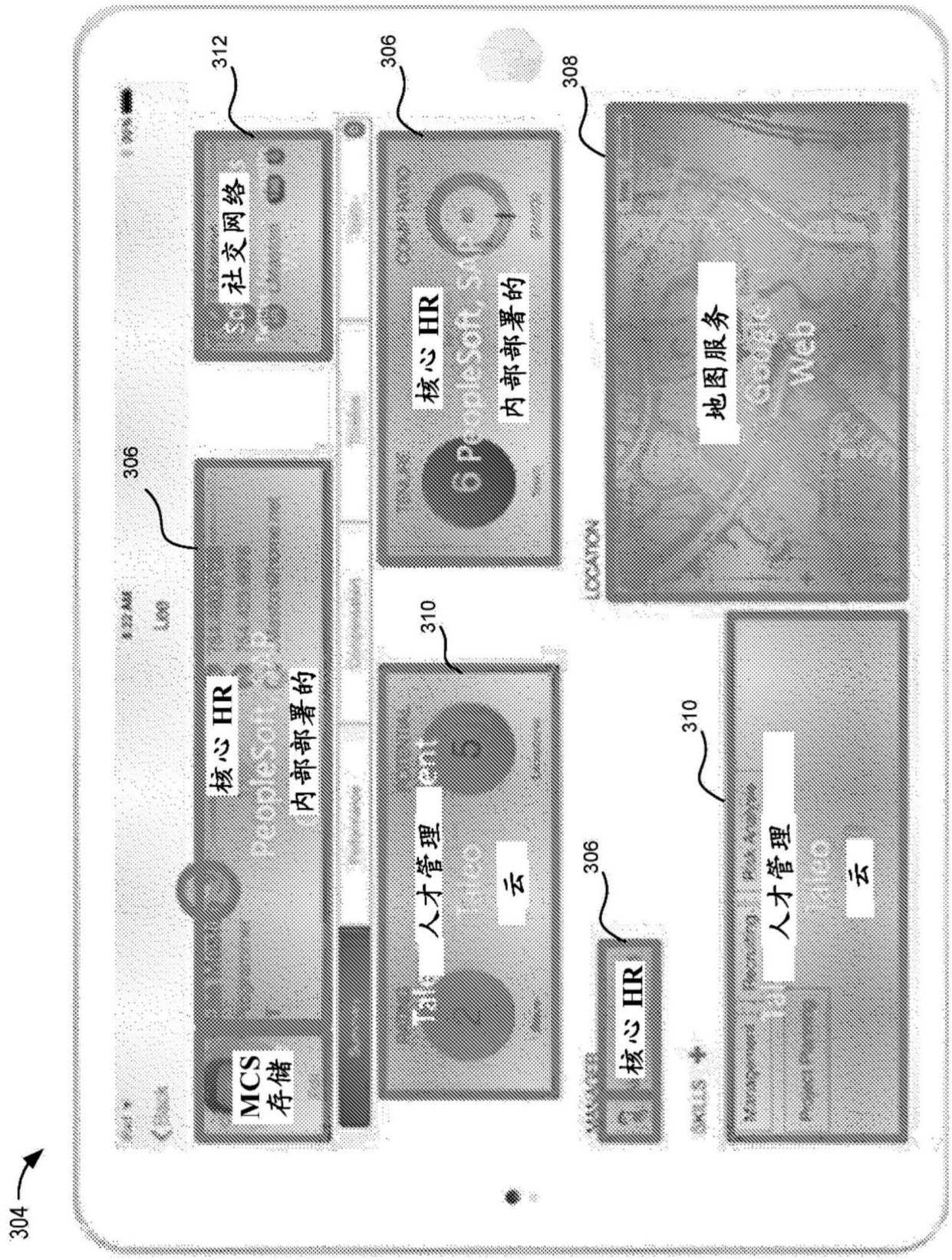

304
312
306
社交网络
核心 HR
内部部署的
MCS
存储
306
核心 HR
内部部署的
308
地图服务
310
人才管理
云
306
核心 HR
310
人才管理
云

图3C

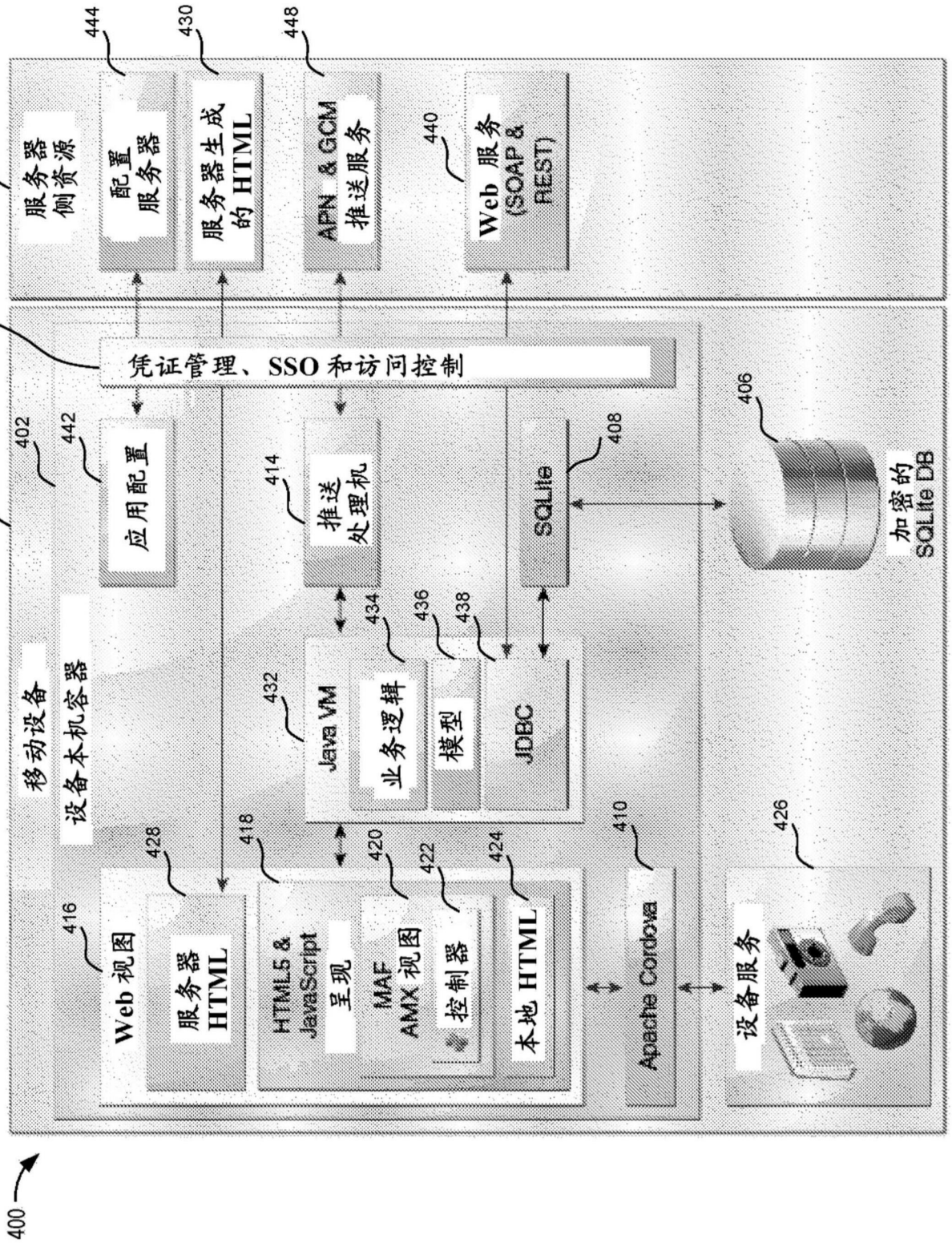

400
412
服务器侧资源
444
配置服务器
430
服务器生成的 HTML
448
APN & GCM 推送服务
440
Web 服务 (SOAP & REST)
446
凭证管理、SSO 和访问控制
404
402
移动设备
设备本机容器
442
应用配置
414
推送处理机
406
加密的 SQLite DB
408
SQLite
434
436
438
Java VM
业务逻辑
模型
JDBC
432
428
418
420
422
424
410
426
Web 视图
服务器 HTML
HTML5 & JavaScript 呈现
MAF AMX 视图
控制器
本地 HTML
Apache Cordova
设备服务
416

图4

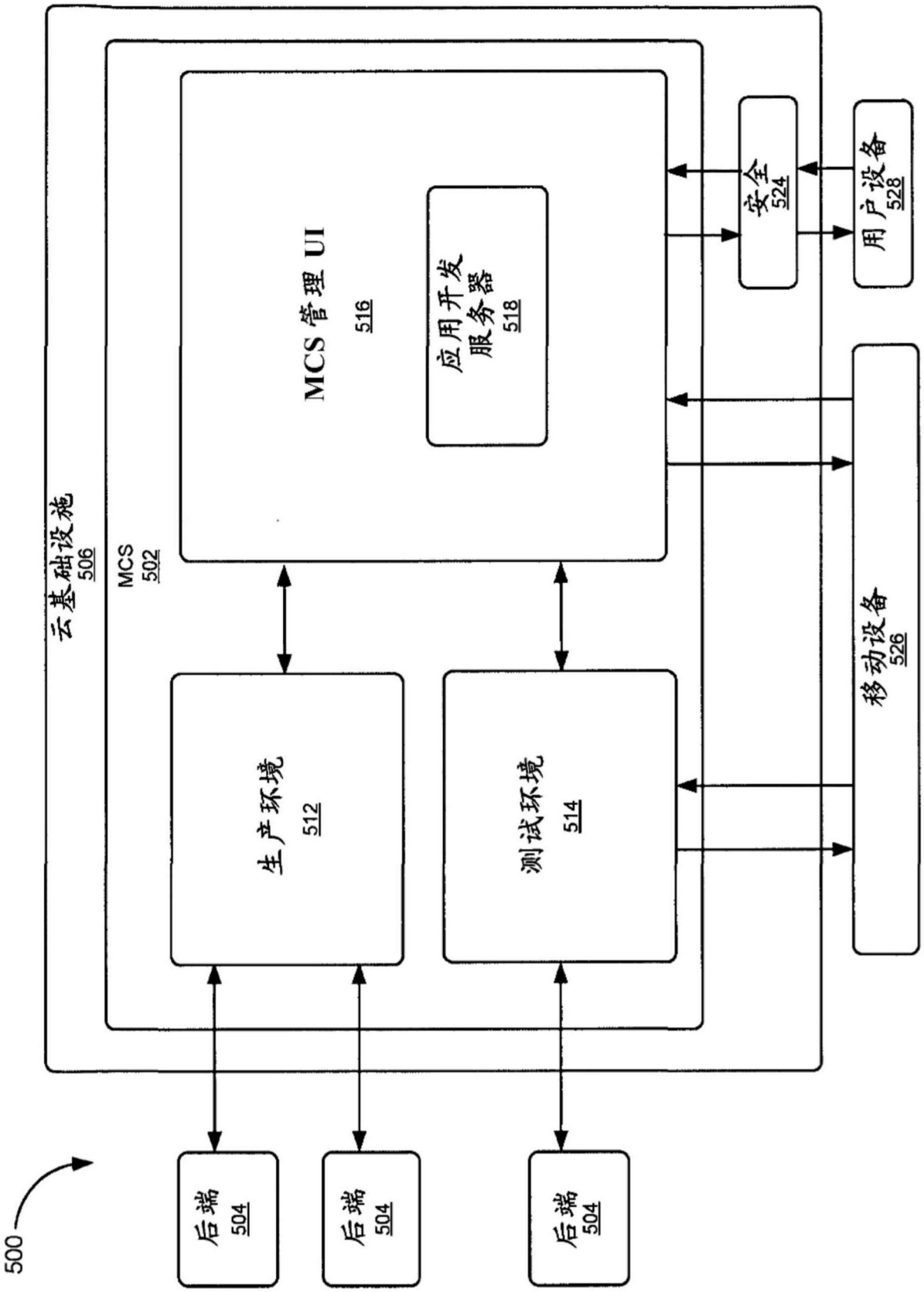

500
云基础设施
506
MCS
502
MCS 管理 UI
516
应用开发
服务器
518
生产环境
512
测试环境
514
安全
524
用户设备
528
移动设备
526
后端
504
后端
504
后端
504

图5

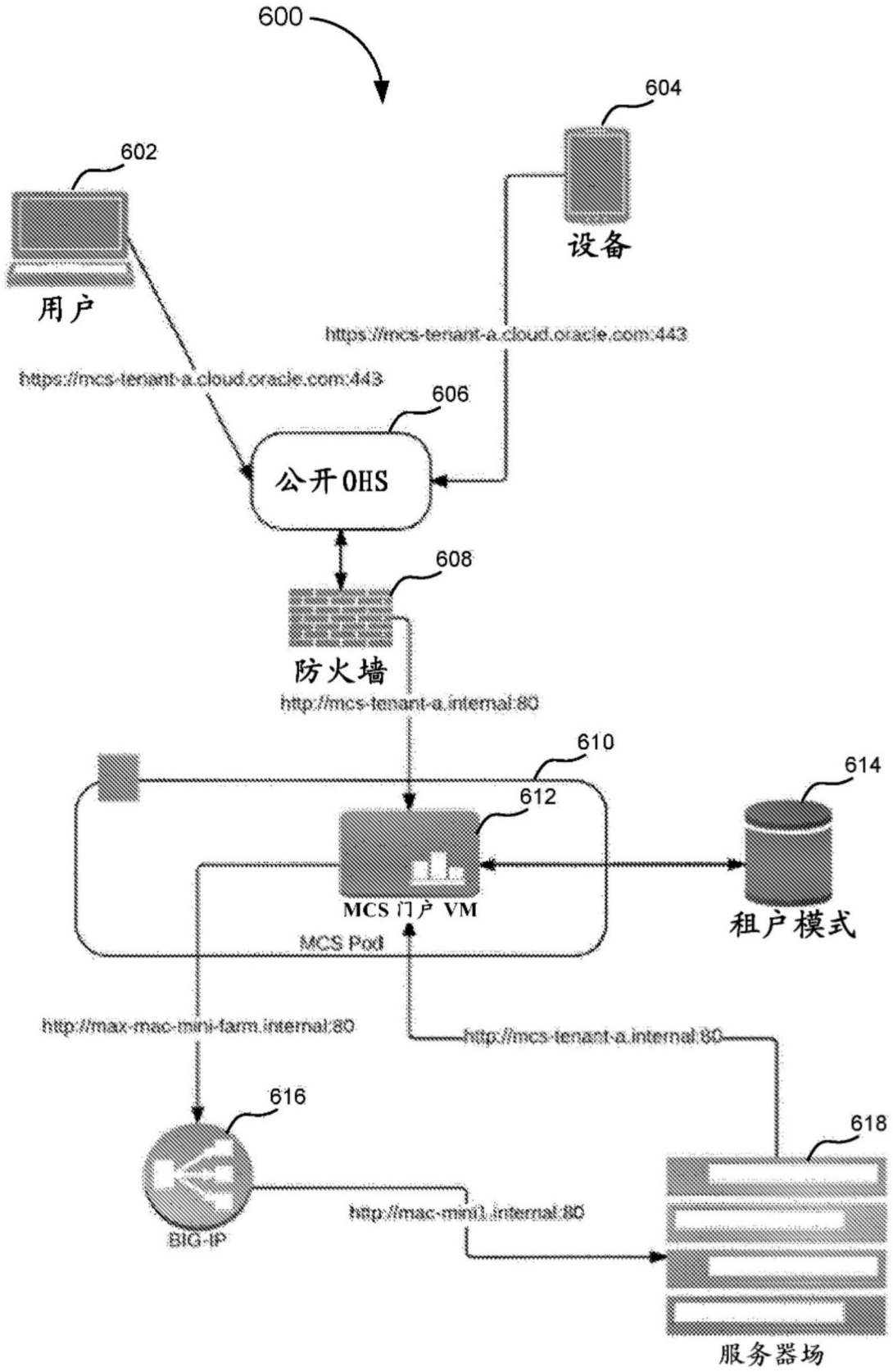

600
604
602
设备
用户
https://mcs-tenant-a.cloud.oracle.com:443
https://mcs-tenant-a.cloud.oracle.com:443
606
公开OHS
608
防火墙
http://mcs-tenant-a.internal:80
610
612
614
MCS 门户 VM
租户模式
MCS Pod
http://max-mac-mini-farm.internal:80
http://mcs-tenant-a.internal:80
616
618
BIG-IP
http://mac-mini1.internal:80
服务器场

图6

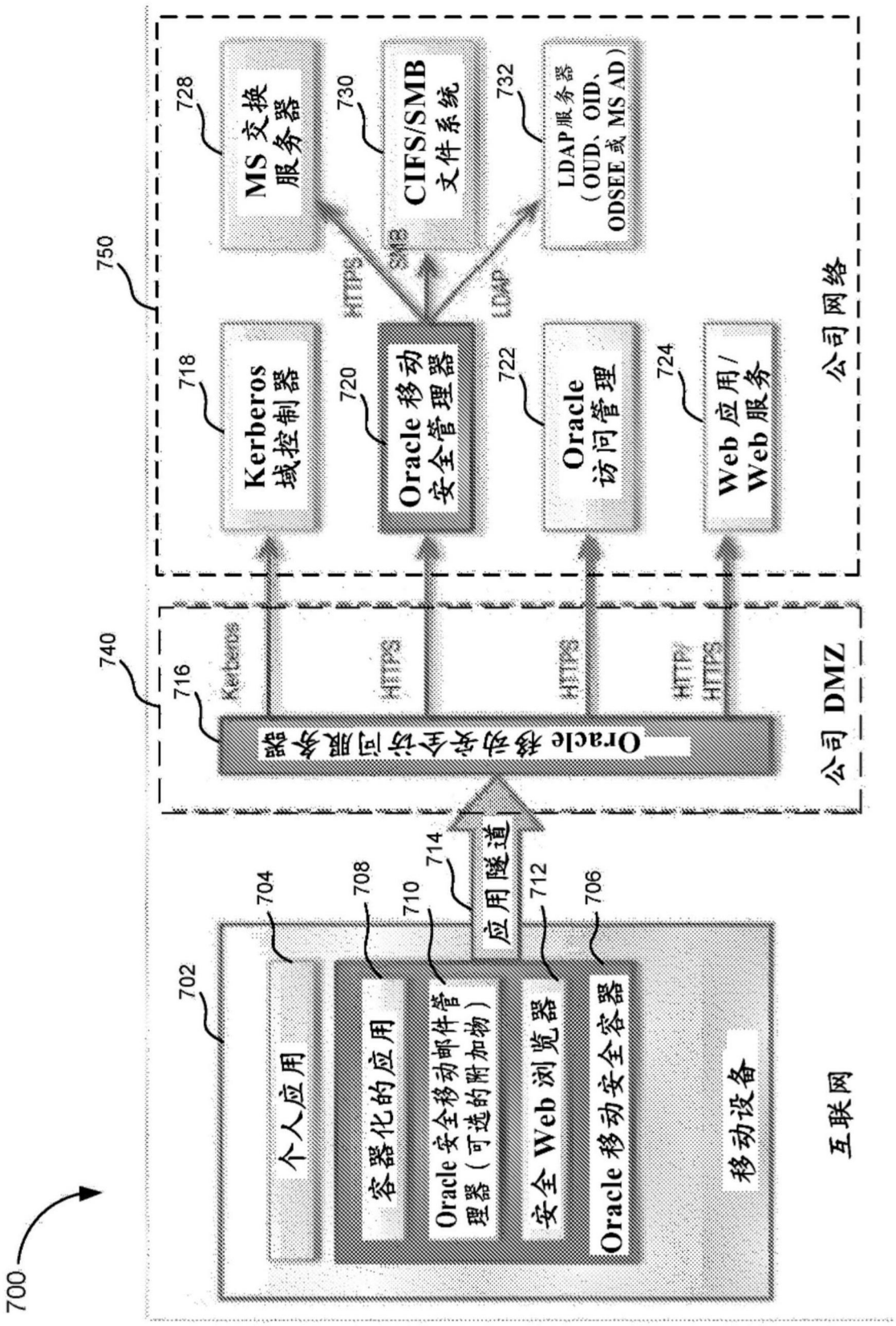

700
互联网
移动设备
个人应用
容器化的应用
Oracle 安全移动邮件管理器（可选的附加物）
安全 Web 浏览器
Oracle 移动安全容器
应用隧道
公司 DMZ
Oracle 移动应用网关
Kerberos
HTTPS
HTTP
HTTPS
公司网络
Kerberos 域控制器
Oracle 移动安全管理器
Oracle 访问管理
Web 应用/ Web 服务
MS 交换服务器
CIFS/SMB 文件系统
LDAP 服务器（OUD、OID、ODSEE 或 MS AD）
HTTPS
SMB
LDAP
702
704
706
708
710
712
714
716
718
720
722
724
728
730
732
740
750

图7

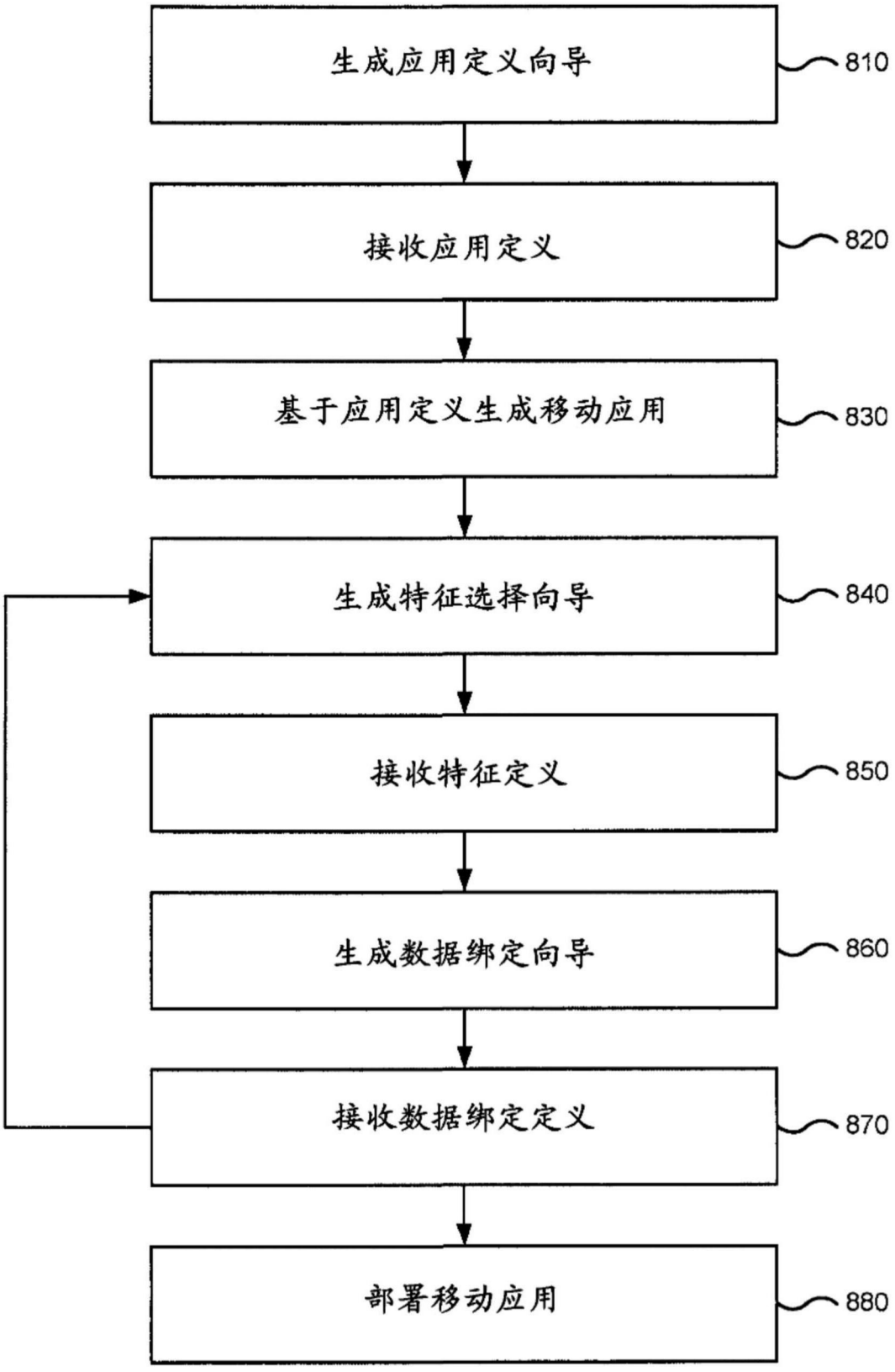

生成应用定义向导
810
接收应用定义
820
基于应用定义生成移动应用
830
生成特征选择向导
840
接收特征定义
850
生成数据绑定向导
860
接收数据绑定定义
870
部署移动应用
880

图8

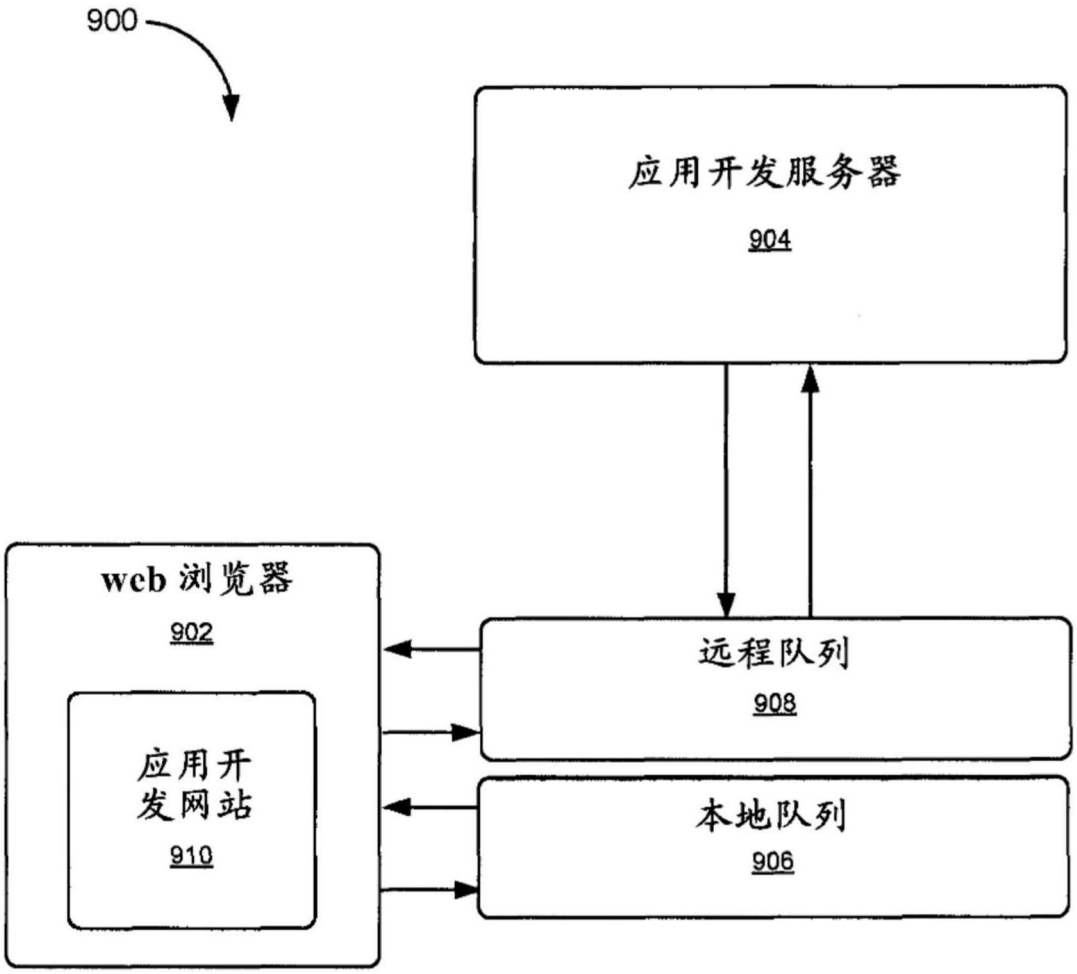


图9

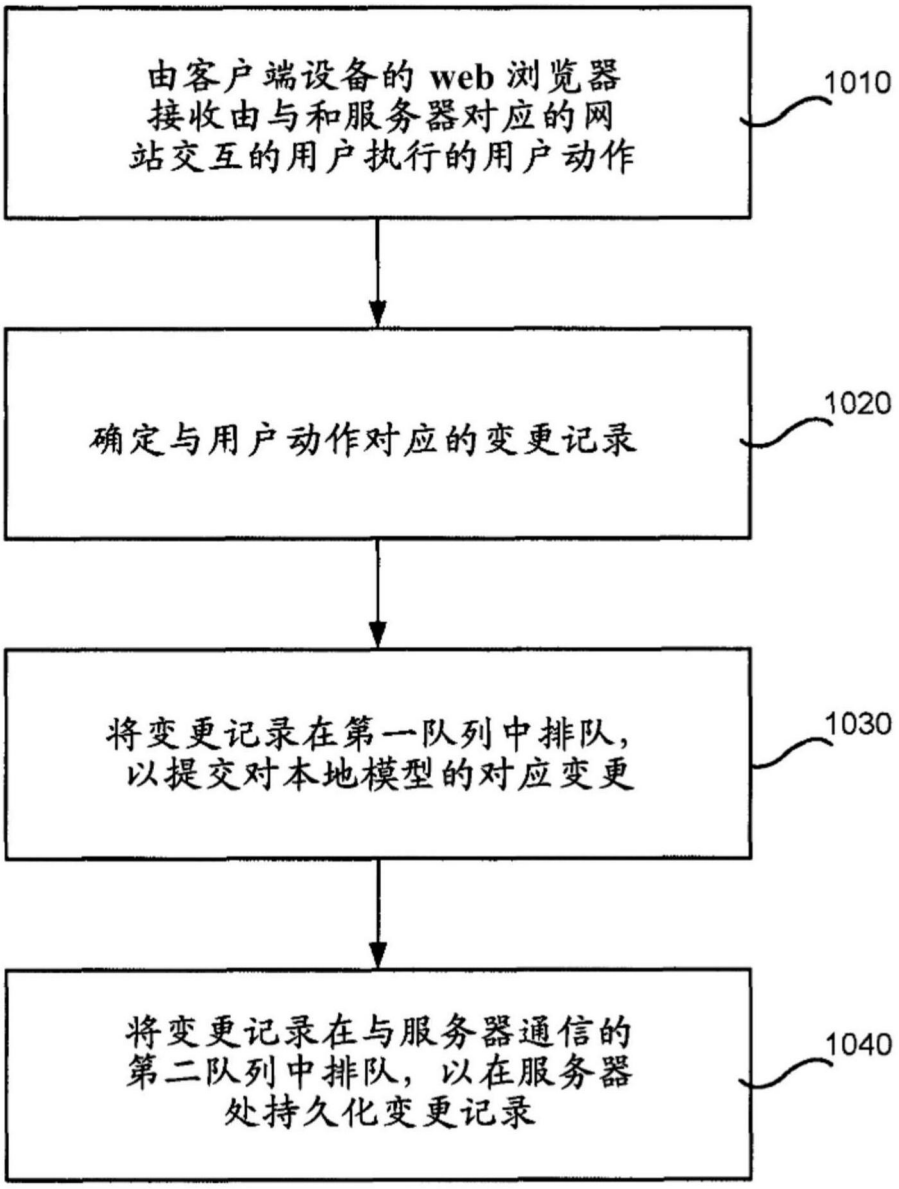

由客户端设备的 web 浏览器接收由与和服务器对应的网站交互的用户执行的用户动作
1010
确定与用户动作对应的变更记录
1020
将变更记录在第一队列中排队，以提交对本地模型的对应变更
1030
将变更记录在与服务器通信的第二队列中排队，以在服务器处持久化变更记录
1040

图10

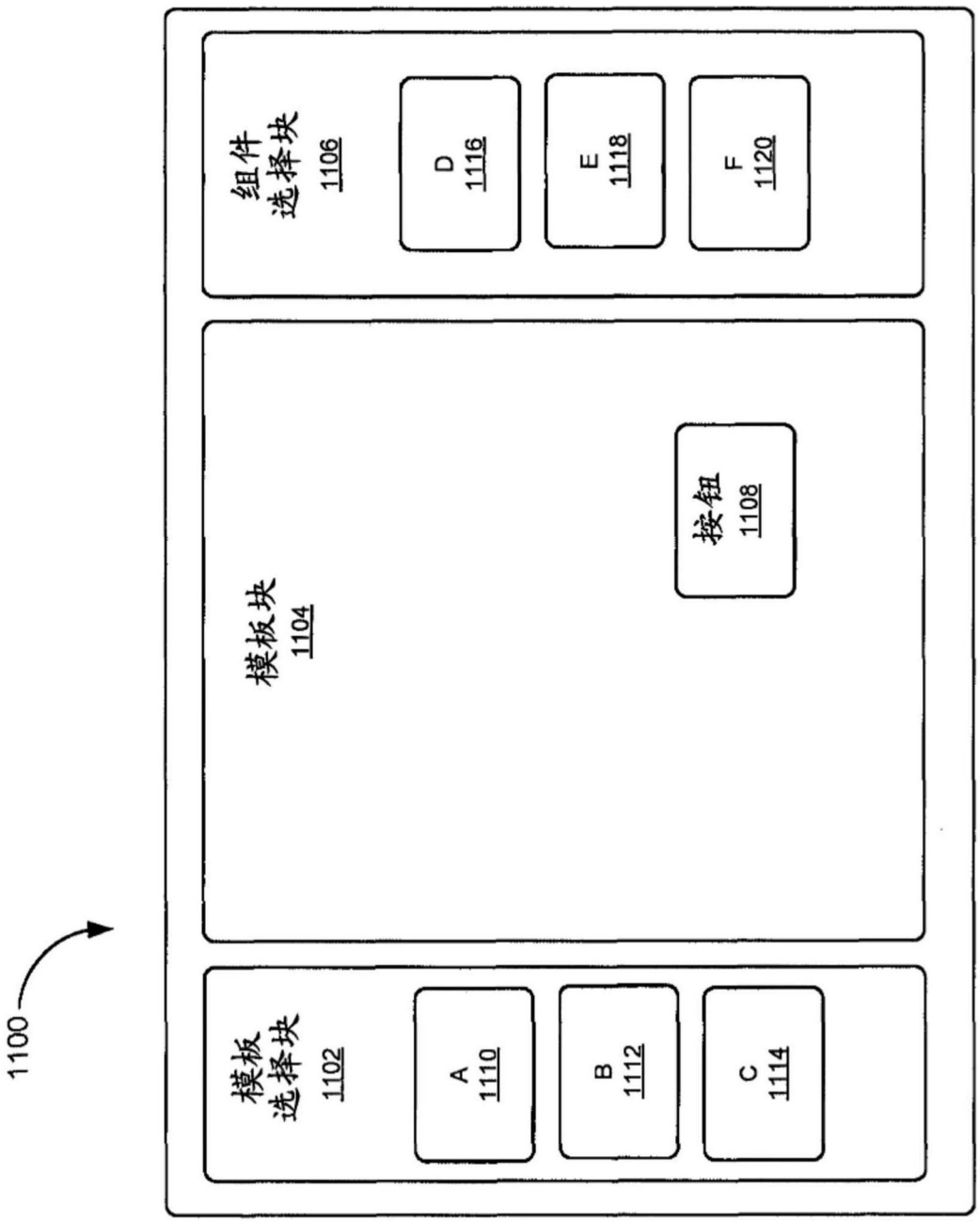

1100
模板选择块
1102
A
1110
B
1112
C
1114
模板块
1104
按钮
1108
组件选择块
1106
D
1116
E
1118
F
1120

图11

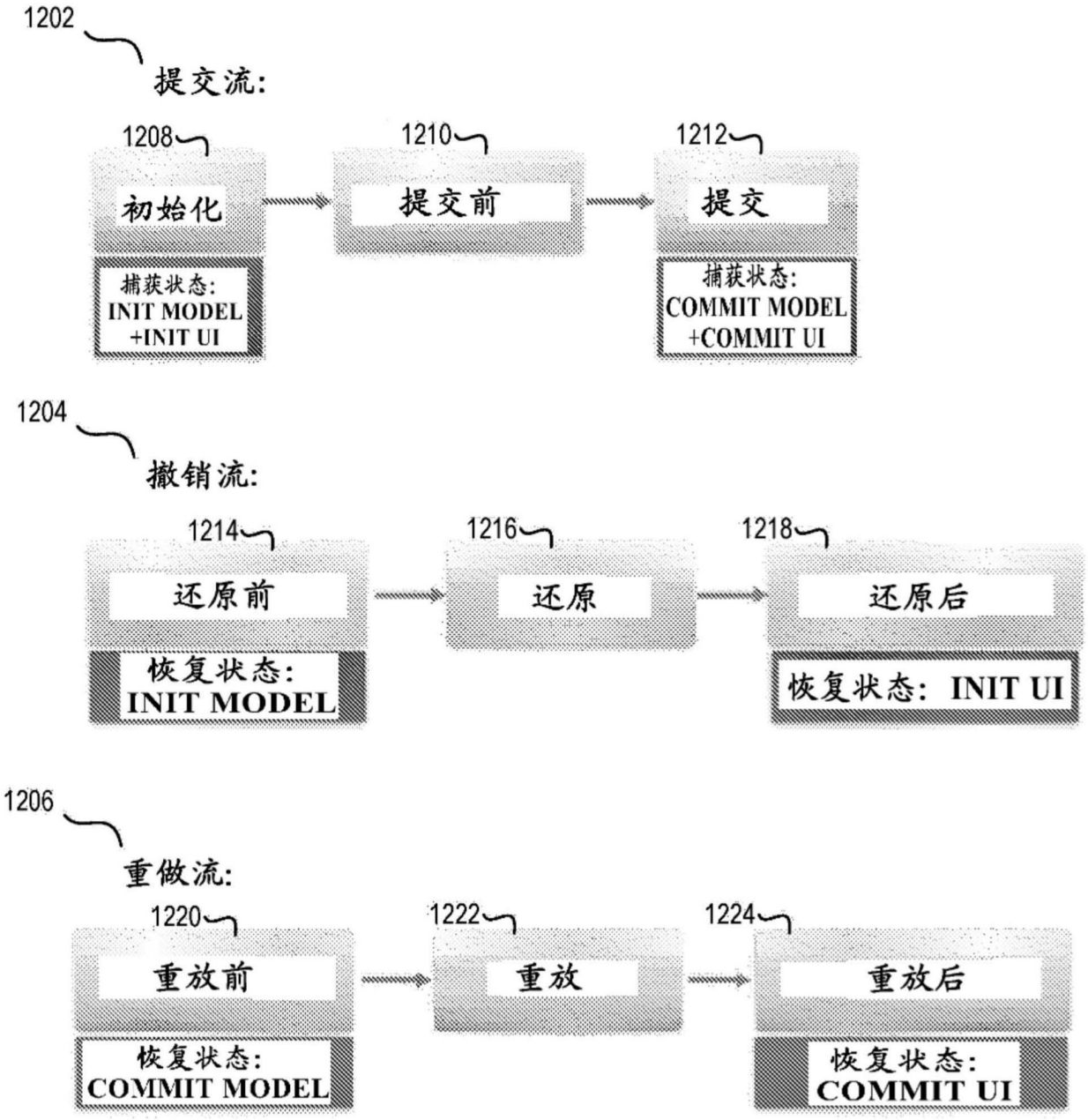

1202
提交流:
1208
初始化
捕获状态:
INIT MODEL
+INIT UI
1210
提交前
1212
提交
捕获状态:
COMMIT MODEL
+COMMIT UI
1204
撤销流:
1214
还原前
恢复状态:
INIT MODEL
1216
还原
1218
还原后
恢复状态: INIT UI
1206
重做流:
1220
重放前
恢复状态:
COMMIT MODEL
1222
重放
1224
重放后
恢复状态:
COMMIT UI

图12

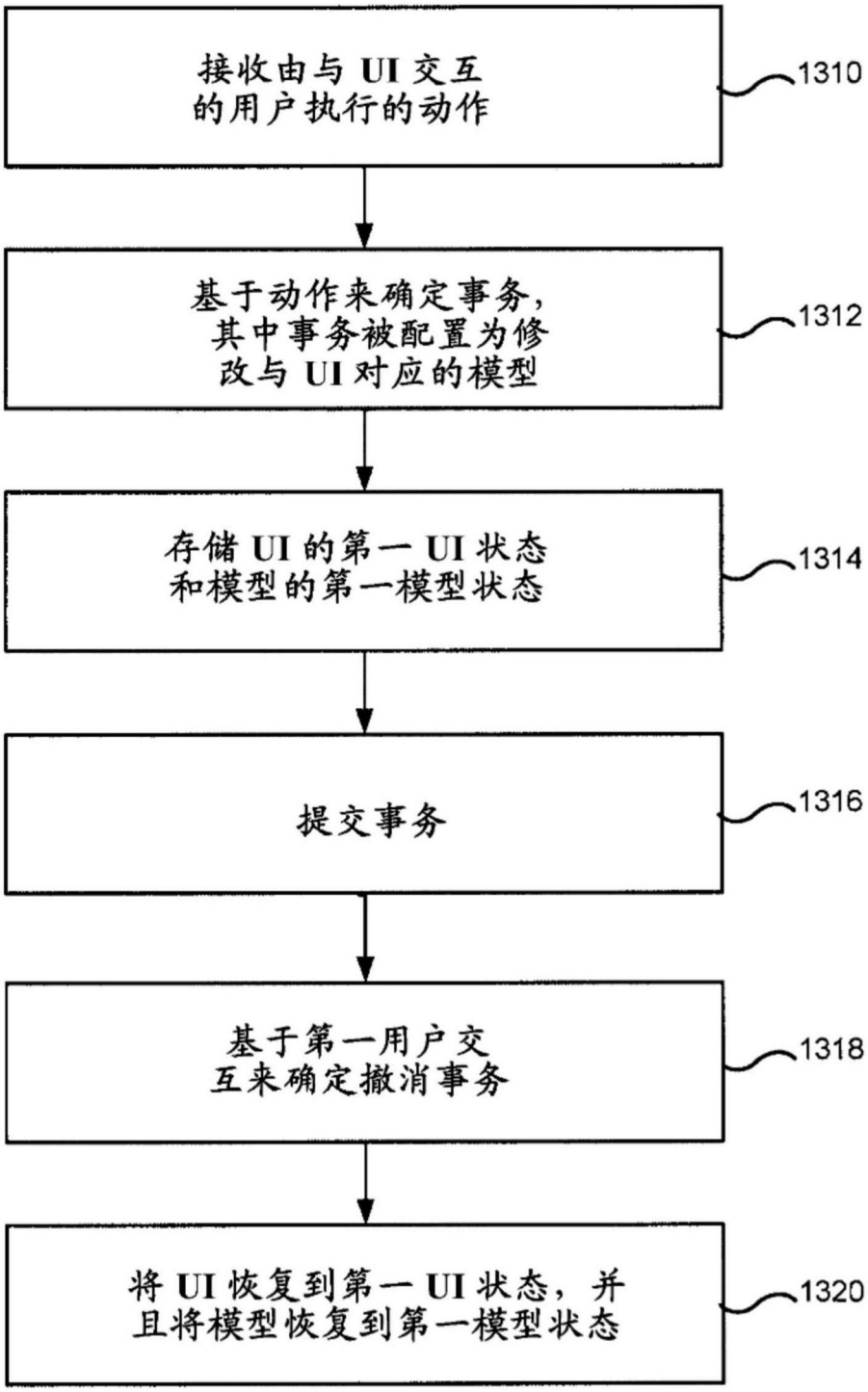

接收由与 UI 交互的用户执行的动作
1310
基于动作来确定事务，其中事务被配置为修改与 UI 对应的模型
1312
存储 UI 的第一 UI 状态和模型的第一模型状态
1314
提交事务
1316
基于第一用户交互来确定撤消事务
1318
将 UI 恢复到第一 UI 状态，并且将模型恢复到第一模型状态
1320

图13