



(12)发明专利

(10)授权公告号 CN 105593813 B

(45)授权公告日 2019.09.24

(21)申请号 201480054050.6

(22)申请日 2014.09.25

(65)同一申请的已公布的文献号
申请公布号 CN 105593813 A

(43)申请公布日 2016.05.18

(30)优先权数据

61/884,743 2013.09.30 US
61/905,128 2013.11.15 US
61/905,111 2013.11.15 US
61/905,114 2013.11.15 US
61/905,116 2013.11.15 US
61/905,129 2013.11.15 US
61/905,105 2013.11.15 US
61/905,101 2013.11.15 US
61/905,119 2013.11.15 US
61/905,243 2013.11.17 US
61/905,247 2013.11.17 US
14/231,869 2014.04.01 US

(85)PCT国际申请进入国家阶段日
2016.03.30

(86)PCT国际申请的申请数据
PCT/US2014/057324 2014.09.25

(87)PCT国际申请的公布数据
W02015/048206 EN 2015.04.02

(73)专利权人 微软技术许可有限责任公司
地址 美国华盛顿州

(72)发明人 B·奥兰尼科

G·C·J·A·祖尼诺 W·Y·莫耶
A·比尔克 M·乔希
B·亚历山大 T·考克斯
I·埃齐鲁 J-S·古皮

(74)专利代理机构 上海专利商标事务所有限公司 31100

代理人 杨洁

(51)Int.Cl.

G06F 8/34(2018.01)
G06F 9/451(2018.01)
G06F 9/48(2006.01)
G06F 3/0481(2013.01)
G06F 3/0485(2013.01)
G06F 16/242(2019.01)

(56)对比文件

US 2013/0191880 A1,2013.07.25,
US 2005/0188349 A1,2005.08.25,
US 2013/0191880 A1,2013.07.25,
CN 101004685 A,2007.07.25,
CN 101499004 A,2009.08.05,
CN 1696896 A,2005.11.16,
US 2012/0030591 A1,2012.02.02,

审查员 黄超

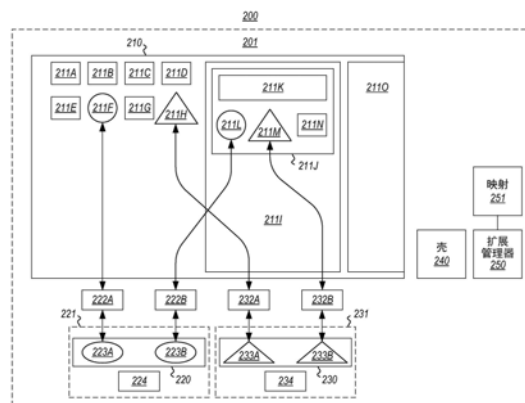
(54)发明名称

用于可视化从受限环境容器提供的数据的
呈现解释器

(57)摘要

在此描述的至少一些实施例涉及呈现解释器的用户,该呈现解释器从在受限环境容器(诸如IFrame)内执行的代码接收数据对象。受限环境容器阻止代码不利地影响用户界面系统整体上的性能和一致性,并且还阻止代码不利地影响其它受限环境容器中运行的代码的性能。

权利要求书1页 说明书6页 附图3页



1. 一种实例化和/或操作呈现解释器的方法,所述方法包括:

在受限环境容器内运行针对扩展应用的不可信代码以便提供其中运行所述不可信代码的受限环境;

使用针对所述扩展应用的应用数据生成多个数据对象;

使用为每个数据对象提供的呈现解释器来处理所述多个数据对象;并

将来自每个呈现解释器的经处理的数据对象投射到用户界面系统的对应的单个动态界面元素,其中每个受限环境容器被分摊到单个扩展应用,而每个呈现解释器被分摊到所述用户界面系统的单个动态界面元素,与将所述受限环境容器分摊到单个动态界面元素相对应。

2. 如权利要求1所述的方法,其特征在于,所述受限环境容器如果被重新配置就适于显示内容。

3. 如权利要求1所述的方法,其特征在于,所述受限环境容器是IFrame,且所述多个数据对象是查看模型。

4. 如权利要求1所述的方法,其特征在于,所述多个数据对象的副本在所述受限环境容器内,且所述多个数据对象的经同步的副本在用户界面系统环境内。

5. 如权利要求1所述的方法,其特征在于,所述代码是作为应用中扩展所述用户界面系统的特征的部分来运行的代码,所述单个动态界面元素由所述用户界面系统展示在用户界面上,但由作为所述应用的部分来运行的代码来填充。

6. 如权利要求1所述的方法,其特征在于,所述用户界面系统包括多个受限环境容器,每个受限环境容器专用于运行来自扩展所述用户界面系统的特征的对应应用的代码。

7. 如权利要求6所述的方法,其特征在于,进一步结构化包括实例化和/或操作以下:

容器管理组件,当要被来自还不存在针对其的受限环境容器的特定应用的代码填充的用户界面元素要被显示时,所述容器管理组件实例化新的受限环境容器。

8. 如权利要求6所述的方法,其特征在于,进一步包括实例化和/或操作以下:

虚拟化组件,所述虚拟化组件在当用户界面元素不再被显示在显示器上但仍在所述用户界面中时,从计算系统释放用户界面元素的用户界面状态。

9. 如权利要求8所述的方法,其特征在于,进一步包括实例化和/或操作以下:

容器管理组件,当所述计算系统不再具有对应于由特定应用填充的任何用户界面元素的任何用户界面状态时,所述容器管理组件释放对应于所述特定应用的受限环境容器。

10. 一种其上操作以下的计算系统:

用于呈现具有多个用户界面元素的用户界面的用户界面系统;

扩展所述用户界面系统的特征的多个扩展应用;

多个受限环境容器,每个受限环境容器专用于运行对应于所述多个受限环境容器之一的不可信代码,以便为其中运行所述不可信代码的每个扩展应用提供受限环境,每个受限环境容器使用针对所述扩展应用的应用数据以生成多个数据对象,并且所述多个受限环境容器基于每个扩展应用一个受限环境容器来被分摊,与每个动态界面元素一个受限环境容器相对应;

多个呈现解释器,针对由所述受限环境容器处理的每个数据对象,每个呈现解释器被配置用于处理对应的数据对象并投射到所述用户界面系统的单个动态界面元素中。

用于可视化从受限环境容器提供的数据的呈现解释器

[0001] 背景

[0002] IFrame的主要目的是在较大用户界面的特定窗口内以该窗口看起来好像是该较大用户界面的集成的部分的方式来呈现内容(下文“不可信内容”)。内容是“不可信”的,因为它是由不可信代码(通常是脚本)生成的。IFrame提供隔离功能,其阻止脚本对IFrame外部的对象作出有害动作。毕竟,IFrame内所提供的脚本通常由第三方实体提供。

[0003] 每个窗口通常有一个IFrame。主应用内容通常被显示在一个IFrame中。此外,存在针对具有不可信内容的每个窗口的专用IFrame。在传统用户界面系统中,可能存在用于将不可信内容呈现在显示的各部分的一个或两个IFrame(不同于主应用内容的主IFrame)。例如,在浏览器应用中,主浏览器显示可能由一个IFrame呈现。在显示的另一角落可能存在小的股票更新窗口,其常规地更新几个指定股票的价格。那个小窗口的内容也由单独的IFrame驱动。因此,在传统方法中,IFrame与显示不可信内容的窗口之间存在一对一对应。

[0004] 简要概述

[0005] 在此描述的至少一些实施例涉及呈现解释器的用户,解释器从在受限环境容器(诸如IFrame)内执行的代码接收数据对象(诸如查看模型)。受限环境容器阻止代码不利地影响用户界面系统整体上的性能和一致性,并且还阻止代码不利地影响其它受限环境容器中运行的代码的性能。

[0006] 呈现解释器处理从受限环境容器中接收的数据对象,并将结果数据展示给受限环境容器外部的用户界面元素。结果数据被结构化使得用户界面元素将结果数据的至少部分可视化。代码随后被允许安全地将内容投射到用户界面元素中。

[0007] 在某些实施例中,单个受限环境容器可包含多个代码元素,每个代码元素负责将内容投射到对应的用户界面元素中。因此,不再存在内容窗口和受限环境容器之间的一对一映射。取而代之的,在其中用户界面应用由多个其它实体扩展的上下文中,可向给定应用扩展分派单个受限环境容器,而不管扩展填充了多少内容窗口。与那个扩展相关联的代码接着在对应的受限环境容器内执行。因此,动态内容的多得多的元素可被呈现在使用较小数量的受限环境容器的用户界面上。因此,用户界面系统可从其它提供者安全地获得显著的动态内容贡献。

[0008] 提供本发明内容以便以简化的形式介绍将在以下的具体实施方式中进一步描述的一些概念。本概述并非旨在标识出要求保护的主题的关键特征或必要特征,亦非旨在用作辅助确定要求保护的主题的范围。

[0009] 附图简述

[0010] 为了描述可获得本发明的上述和其它优点和特征的方式,将通过参考附图中示出的本发明的具体实施例来呈现以上简要描述的本发明的更具体描述。可以理解,这些附图只描绘了本发明的各典型实施例,并且因此不被认为是对其范围的限制,将通过使用附图并利用附加特征和细节来描述和解释本发明,在附图中:

[0011] 图1例示出其中可采用本文中所描述的原理的示例计算系统;

[0012] 图2示出了包括可用图1的计算系统来操作的用户界面系统的环境;以及

[0013] 图3示出了其中代码在IFrame内的数据上操作,并且其中数据在IFrame和用户界面元素之间传送的机制。

[0014] 详细描述

[0015] 在此描述的至少一些实施例涉及包括用户界面系统的计算系统,该用户界面系统用于呈现具有多个用户界面元素的用户界面。尽管用户界面系统将用户界面元素呈现为容器,但用户界面元素的至少一些可由扩展用户界面元素的特征的应用填充。例如,用户界面系统可以是web应用。扩展浏览器应用的特征的应用可以是对web应用的扩展。

[0016] 为了使得在各应用和用户界面系统之间以及各应用它们本身之间存在安全和可靠的隔离,每个应用具有专用受限环境容器(例如,未显示的iFrame)。每个用户界面元素还包括呈现解释器,每个被配置来处理来自运行在受限环境容器中的对应的代码的任何接收到的数据对象(诸如查看模型),并将结果数据展示给对应的用户界面元素。任何给定呈现解释器提供的结果数据被结构化,使得对应的用户界面元素可视化结果数据的至少一些,从而允许对应的代码将内容投射到对应的用户界面元素中。这样的结果数据的提供在下文中将也被称为“投射内容”。

[0017] 在所接收的数据对象是呈现解释器的情况中,查看模型的副本可被包含在对应的受限环境容器内,并且相同查看模型的另一经同步的副本可与用户界面系统执行环境一起被包含。随着对查看模型的一个副本作出改变,使用消息收发将它们传播到查看模型的另一副本。因此,运行在受限环境容器中的应用的代码可将可视化投射到合适的用户界面元素中,而不冒险损害用户界面系统或其它应用。此外,由于呈现解释器可实施关于内容如何被投射到用户界面元素中的规则,呈现解释器趋向于提供用户界面的一致的外观和感受。类似地,当对可视化作出改变时,那些改变可被传播回受限环境容器内的底层数据。

[0018] 将参考图1描述对计算设备的一些介绍性讨论。随后,将参考后续的附图来描述示例用户界面、方法和支持体系结构。

[0019] 计算系统现在越来越多地采取多种多样的形式。例如,计算系统可以是手持式设备、电器、膝上型计算机、台式计算机、大型机、分布式计算系统或甚至常规上不被认为是计算系统的设备。在本说明书以及权利要求书中,术语“计算系统”被广义地定义为包括任何设备或系统(或其组合),该设备或系统包含至少一个物理且有形的处理器以及其上能具有可由处理器执行的计算机可执行指令的物理且有形的存储器。存储器可以采取任何形式,并可以取决于计算系统的性质和形式。计算系统可以分布在网络环境中,并可包括多个组分计算系统。

[0020] 如图1所例示,在其最基本的配置中,计算系统100通常包括至少一个处理单元102和存储器104。存储器104可以是物理系统存储器,该物理系统存储器可以是易失性的、非易失性的、或两者的某种组合。术语“存储器”也可在此用来指示诸如物理存储介质这样的非易失性大容量存储器。如果计算系统是分布式的,则处理、存储器和/或存储能力也可以是分布式的。如本文中所使用的,术语可执行模块或可执行组件或可指可在计算系统上执行的软件对象、例程或方法。此处所描述的不同组件、模块、引擎以及服务可以实现为在计算系统上执行的对象或进程(例如,作为分开的线程)。

[0021] 在随后的描述中,参考由一个或多个计算系统执行的动作描述了各实施例。如果这样的动作是以软件实现的,则执行动作的相关联计算系统的一个或多个处理器响应于已

经执行了计算机可执行指令来引导计算系统的操作。例如,这样的计算机可执行指令可以在形成计算机程序产品的一个或多个计算机可读介质上实现。这样的操作的示例涉及对数据的操纵。计算机可执行指令(以及被操纵的数据)可以存储在计算系统100的存储器104中。计算系统100还可包含允许计算系统100例如通过网络110与其他消息处理器通信的通信信道108。

[0022] 计算系统100还包括在其上可以呈现诸如在此所述的用户界面之类的用户界面的显示器112。这样的用户界面可以在渲染之前就在计算机硬件或其他由计算机表现的形式来生成。这样的用户界面的呈现和/或渲染可以由具有处理单元102的计算系统100来执行,所述处理单元执行在一个或多个计算机可读介质上具体化的一个或多个计算机可执行指令这样的计算机可读介质可以形成计算机程序产品的全部或部分。

[0023] 本文中描述的各实施例可包括或利用专用或通用计算机,该专用或通用计算机包括诸如例如一个或多个处理器和系统存储器等计算机硬件,如以下更详细讨论的本文中描述的各实施例还包括用于承载或存储计算机可执行指令和/或数据结构的物理和其他计算机可读介质。这样的计算机可读介质可以是可由通用或专用计算机系统访问的任何可用介质。存储计算机可执行指令的计算机可读介质是物理存储介质。承载计算机可执行指令的计算机可读介质是传输介质。由此,作为示例而非限制,本发明的各实施例可包括至少两种显著不同的计算机可读介质:计算机存储介质和传输介质。

[0024] 计算机存储介质包括RAM、ROM、EEPROM、CD-ROM或其他光盘存储、磁盘存储或其他磁存储设备、或者可用于存储以计算机可执行指令或数据结构的形式的期望程序代码手段且可由通用或专用计算机访问的任何其他有形介质。

[0025] “网络”被定义为使得电子数据能够在计算机系统和/或模块和/或其它电子设备之间传输的一个或多个数据链路。当信息通过网络或另一个通信连接(硬连线、无线、或者硬连线或无线的组合)传输或提供给计算机时,该计算机将该连接适当地视为传输介质。传输介质可以包括可用于携带计算机可执行指令或数据结构形式的期望程序代码装置并可被通用或专用计算机访问的网络和/或数据链路。上述的组合应当也被包括在计算机可读介质的范围内。

[0026] 此外,在到达各种计算机系统组件之后,计算机可执行指令或数据结构形式的程序代码资料可从传输介质自动传输到计算机存储介质(或反之亦然)。例如,通过网络或数据链路接收到的计算机可执行指令或数据结构可以在网络接口模块(例如,“NIC”)内的RAM中被缓冲,然后最终被传输至计算机系统RAM和/或计算机系统处的较不易失性的计算机存储介质。因而,应当理解,计算机存储介质可被包括在还利用(或甚至主要利用)传输介质的计算机系统组件中。

[0027] 计算机可执行指令例如包括,当在处理器处执行时使通用计算机、专用计算机、或专用处理设备执行某一功能或某组功能的指令和数据。计算机可执行指令可以是例如二进制代码、诸如汇编语言之类的中间格式指令、或甚至源代码。尽管用结构特征和/或方法动作专用的语言描述了本主题,但可以理解,所附权利要求书中定义的主题不必限于上述特征或动作。相反,上述特征和动作是作为实现权利要求的示例形式而公开的。

[0028] 本领域的技术人员将理解,本发明可以在具有许多类型的计算机系统配置的网络计算环境中实践,这些计算机系统配置包括个人计算机、台式计算机、膝上型计算机、消息

处理器、手持式设备、多处理器系统、基于微处理器的或可编程消费电子设备、网络PC、小型计算机、大型计算机、移动电话、PDA、寻呼机、路由器、交换机等等。本发明也可在其中通过网络链接(或者通过硬连线数据链路、无线数据链路,或者通过硬连线和无线数据链路的组合)的本地和远程计算机系统两者都执行任务的分布式系统环境中实施。在分布式系统环境中,程序模块可以位于本地和远程存储器存储设备二者中。

[0029] 图2示出了包括可用图1的计算系统100来操作的环境200。环境200包括呈现用户界面210的用户界面系统201。在此描述的原理可使用具有多用户界面元素的用户界面来操作,不管是谁的用户界面元素或如何分发。作为示例的目的,用户界面210具有多个用户界面元素211A到211O。

[0030] 存在扩展用户界面系统(下文也称为“扩展应用”)的特征且在用户界面系统内操作的一个或多个应用。然而,每个扩展应用的代码运行在专用于为那个扩展应用运行代码的专用受限环境容器内。这阻止任何给定扩展应用损害或扰乱用户界面系统的一致性,或损坏另一应用的性能。例如,在图2中存在运行于对应的专用受限环境容器221和231内的两个扩展应用220和230。

[0031] 尽管不需要,受限环境容器221可以是通常被提供以使用其中执行的代码来显示动态内容的容器。例如,每个受限环境容器可以是,例如,IFrame。然而,IFrame被配置使得不呈现任何可显示的内容,而仅运行其中的代码,从而利用了IFrame的隔离特性。

[0032] 典型地,在动态web页面中,当IFrame内的不可信代码被执行以生成那个内容时,每个用户界面元素存在一个IFrame。然而,当这样的用户界面元素(及对应的IFrame)的数量上升时,这是不可缩放的,因为系统的存储器资源变得被压垮。这对于允许其它不可信应用填充用户界面元素的用户界面而言尤为真实。

[0033] 根据在此描述的原理,IFrame被分摊为每个扩展应用一个,与每个动态用户界面元素一个相对。因此,由于任何给定应用扩展可填充远多于具有动态内容的单个动态界面元素,这显著减少了用户界面所需的IFrame的数量,因此促进了具有包含动态内容的大量用户界面元素的用户界面。

[0034] 例如,在图2的情况中,存在被示为具有由不可信扩展应用的执行驱动的内容的四个用户界面元素,包括用户界面元素211F、211H、211L和211M。使用传统方法,这将需要四个IFrame。然而,这个数量已经被减少到图2中的两个IFrame(对应于受限环境容器221和231)。IFrame使用中的百分比降低在具有仅由少量扩展应用填充的许多用户界面元素的复杂用户界面中可更加有意义。

[0035] 用户界面系统201管理给定扩展应用和该应用填充的用户界面元素之间的映射251。例如,在图2中,映射251将把扩展应用220的标识映射到用户界面元素211F和211L的标识。映射251还将把扩展应用230的标识映射到用户界面元素211H和211M的标识。

[0036] 容器管理组件(或“扩展管理器组件”)250具有对映射的访问权,并且负责创建并移除受限环境容器。当要被来自还不存在针对其的受限环境容器的特定扩展应用的代码填充的用户界面元素要被显示时,容器管理组件创建新的受限环境容器。此外,容器管理组件将对应的扩展应用加载到新创建的受限环境组件中。另一方面,当计算系统不再具有对应于由特定应用填充的任何用户界面元素的任何用户界面状态时,容器管理组件250释放(即,移除)对应于特定应用的受限环境容器。壳240负责添加并移除用户界面元素并通知容

器管理组件250这些事件。

[0037] 针对管理足以仅呈现屏幕上的那些用户界面元素的少量受限环境容器的目的(并且也许略多以便于更平滑的滚动或其它导航)、虚拟化组件(其可能是图2的壳240的部分)在当用户界面元素不再被显示在显示上时的至少一些情况下,从计算系统释放用户界面元素的用户界面状态。这将受限环境容器的使用非常显著地降低到可由用户界面系统显式地管理的可预测的数量。

[0038] 用户界面系统负责以它们可变得被填充的方式展示用户界面元素。例如,壳240确定新用户界面元素何时要被呈现,并获取该用户界面元素的模板(例如,标记)。壳240接着调用合适的扩展应用来填充用户界面元素。填充通过向对应呈现解释器提供一个或多个数据对象(例如,查看模型)的对应代码来发生。呈现解释器接着将来自查看模型的结果数据发送到对应用户界面元素,其中它可被用于填充用户界面元素。

[0039] 应用220包括代码223A和223B,其负责填充各用户界面元素211F和211L,并可基于应用数据224进行操作。应用230包括代码233A和233B,其负责填充各用户界面元素211H和211M,并可基于应用数据234进行操作。在一个实施例中,代码的每个所示项可以是单个控件、或可能是多个控件、或甚至被分层地结构化或依赖于其它控件的输出的复杂控件。控件可能已经被杀毒并被批准由用户界面系统使用。例如,对于用户界面系统提供的固有控件,系统提供方确保安全操作。另一方面,扩展应用提供由壳240杀毒的标记模板。当对应的应用被加载到各受限环境容器中之时或之后,扩展管理器250可负责将代码加载到合适的应用中。

[0040] 如前所述,这个代码的执行的结果不被直接提供到用户界面系统201的执行环境中。相反,每个代码具有对应的呈现解释器,其限制可由代码执行的操作和数据。这允许用户界面系统201和其它扩展应用被保护以防扩展应用的非故意有害结果,或甚至欺诈扩展应用。

[0041] 当代码要将内容提供给各用户界面元素时,代码执行并将一个或多个结果数据对象提供给它相应的呈现解释器。例如,在图2中,代码223A、223B、233A和233B具有对应呈现解释器222A、222B、232A和232B。作为一个例子,每个呈现解释器可以是查看模型。

[0042] 每个呈现解释器被配置来处理来自运行在受限环境容器中的对应的代码的任何接收到的数据对象,并将结果数据展示给其对应的用户界面元素。结果数据被结构化,使得对应的用户界面元素可视化结果数据的至少一些,从而允许对应的代码将内容投射到对应的用户界面元素中。

[0043] 在其中IFrame被用作受限环境容器且所接收到的一个或多个数据对象是查看模型的实施例中,可能存在查看模型的两个副本,一个操作在IFrame内而一个开放地操作在用户界面系统环境内。图3示出了机制300,其中代码311对IFrame310内的数据312进行操作并提供也被包含在IFrame310内的作为查看模型313A的结果对象。还存在操作在用户界面系统执行环境(附带地,可能也是IFrame)内的查看模型的副本313B。呈现解释器访问的是那个副本313B以从而填充对应于代码311的对应用户界面元素321。

[0044] 查看模型的副本313A和313B通过发送消息到消息收发组件320或从消息收发组件320接收消息来保持同步。例如,当代码311在IFrame310内提供查看模型313A的改变,消息收发组件320检测到该改变,并可能发消息给操作在IFrame310外的查看模型323B。呈现解

释器可接着访问查看模型313B以将内容投射到用户界面元素321中。相反,如果对用户界面元素321的状态作出改变,则那个改变反映在IFrame310外的查看模型313B内。查看模型313B接着发送消息给消息收发组件320。消息收发组件320发消息给经更改的对象内的IFrame310内部的查看模型313A。代码311接着访问查看模型313A,且应用数据312被更新。

[0045] 因此,在此描述的各实施例允许用户界面系统用若干应用扩展的贡献来丰富,而那些应用扩展没有不利地影响用户界面系统中的一致性,并且应用扩展没有损害用户界面系统的功能或相互损害。

[0046] 本发明可具体化为其它具体形式而不背离其本质特征。所描述的实施例在所有方面都应被认为仅是说明性而非限制性的。从而,本发明的范围由所附权利要求书而非前述描述指示。落入权利要求书的等效方案的含义和范围内的所有改变应被权利要求书的范围所涵盖。

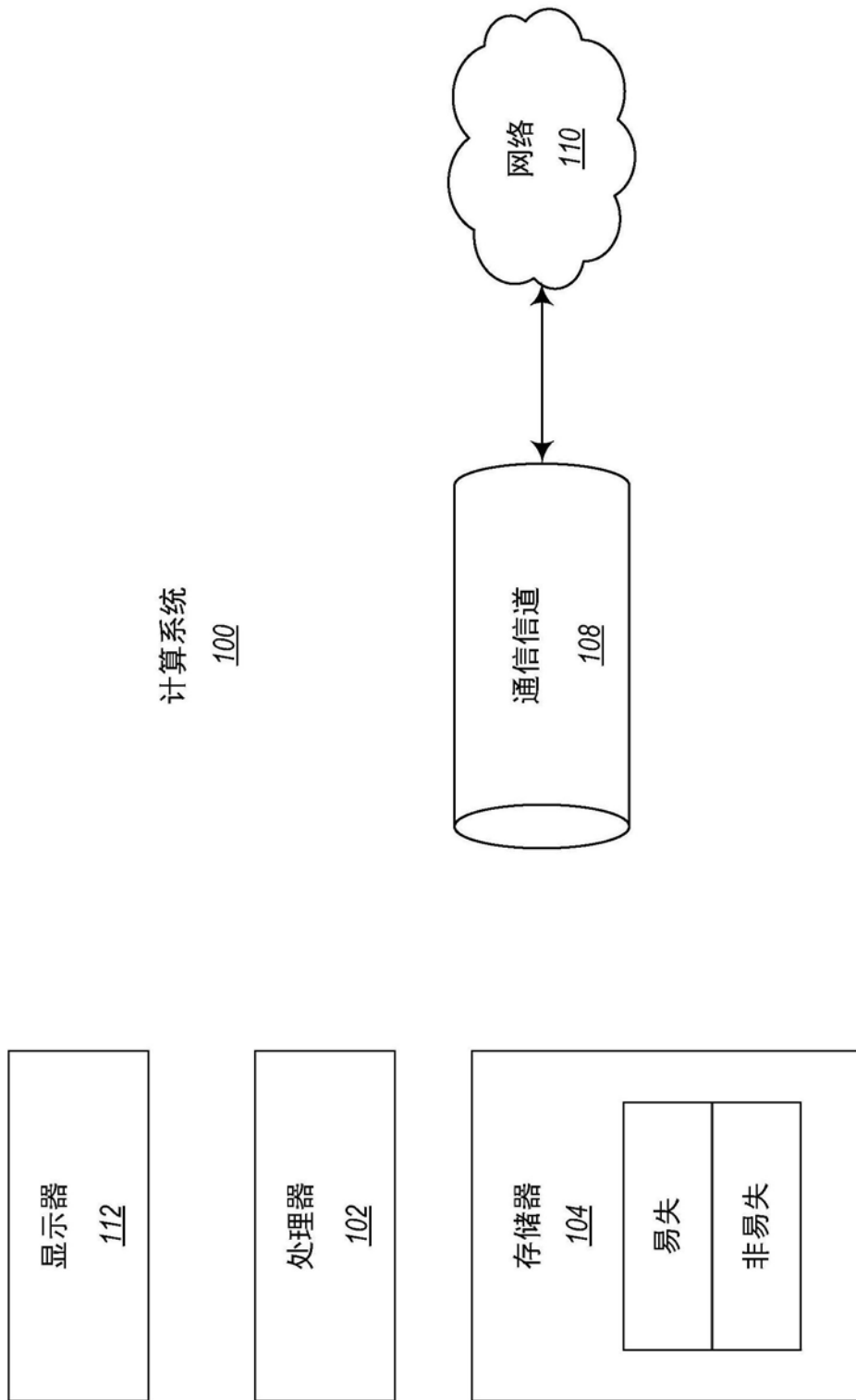


图1

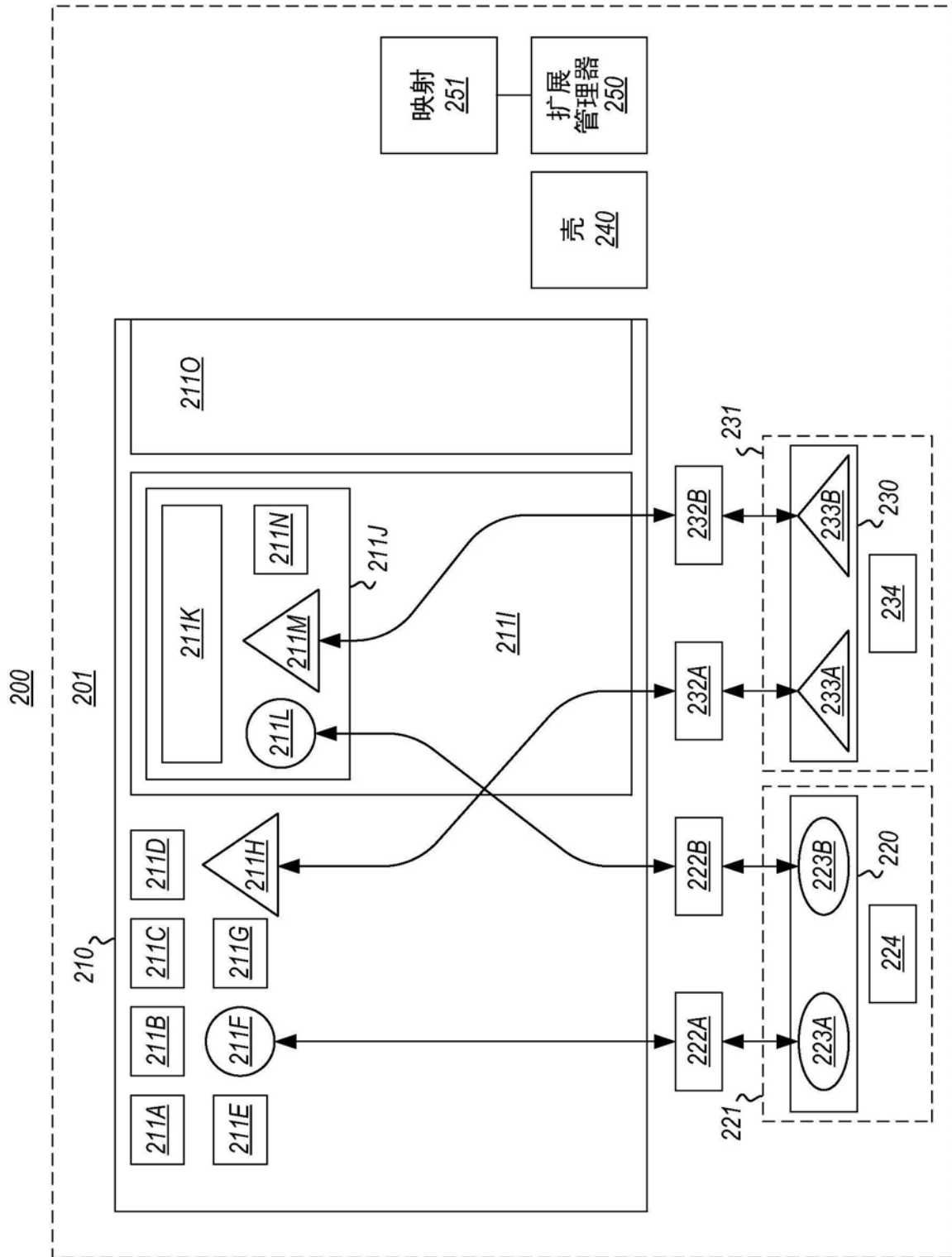


图2

300

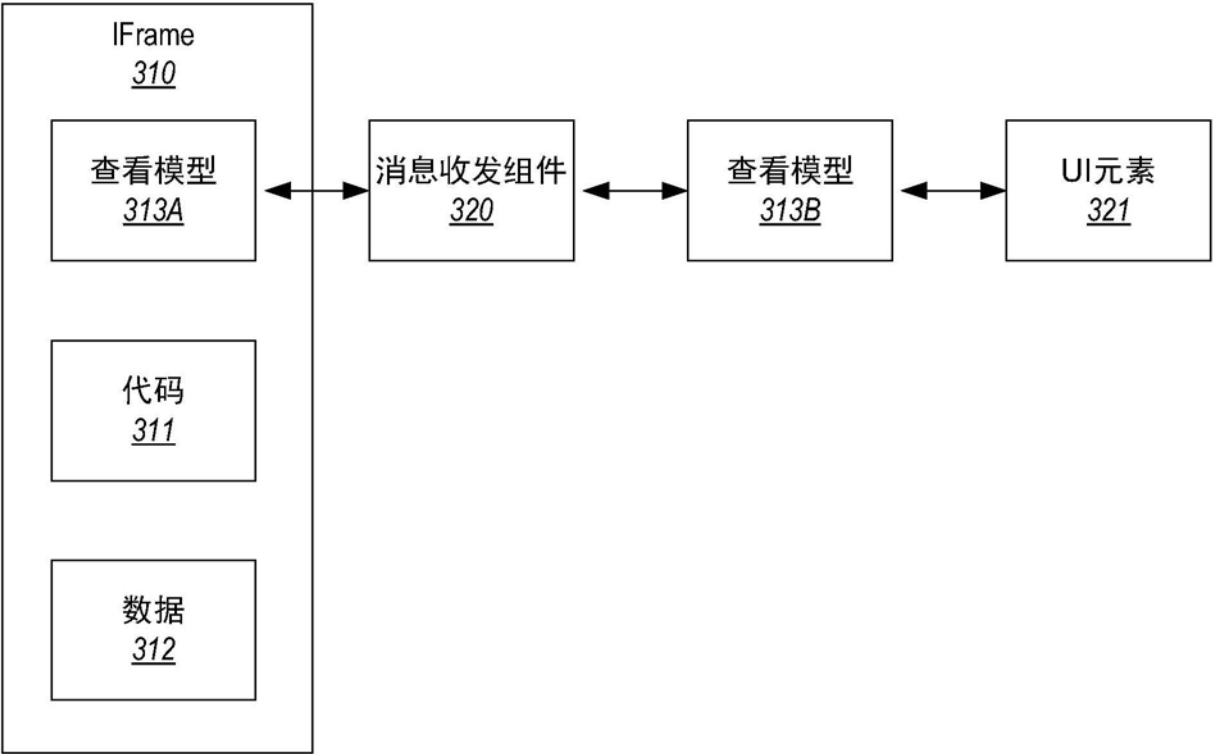


图3