(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 103562862 B
(45)授权公告日 2017.04.26

(21)申请号 201180071359.2
(22)申请日 2011.10.09
(65)同一申请的已公布的文献号
申请公布号 CN 103562862 A
(43)申请公布日 2014.02.05
(30)优先权数据
13/152133 2011.06.02 US
(85)PCT国际申请进入国家阶段日
2013.12.02
(86)PCT国际申请的申请数据
PCT/US2011/055531 2011.10.09
(87)PCT国际申请的公布数据
WO2012/166189 EN 2012.12.06
(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 L.E.布兰科 S.P.蒙卡尤 R.芬克
(74)专利代理机构 中国专利代理(香港)有限公司 72001
代理人 陈慧 汪扬
(51)Int.Cl.
G06F 9/44(2006.01)
G06F 3/14(2006.01)
(56)对比文件
US 7847755 B1,2010.12.07,
US 2005088436 A1,2005.04.28,
US 7168048 B1,2007.01.23,
CN 101258478 A,2008.09.03,
审查员 杨蕊

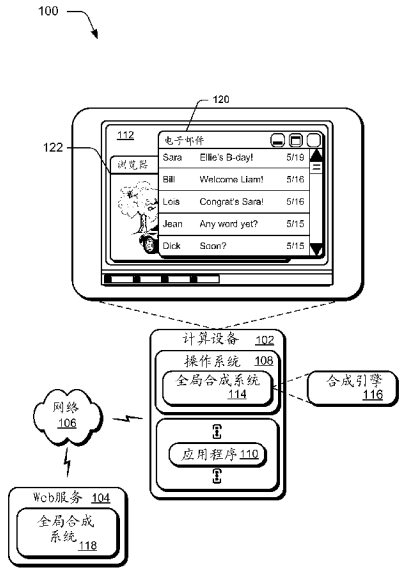
权利要求书1页 说明书11页 附图6页

(54)发明名称

全局合成系统

(57)摘要

描述了一种全局合成系统。在一个或多个实施方式中,全局合成系统可以被配置成执行用于多个应用程序的再现。例如,全局合成系统可以被配置成显露出应用程序可访问的一个或多个应用编程接口(API)。然后API可以被用来使得单合成引擎执行用于所述多个应用程序的再现。单个合成引擎的使用可以被用来支持多种不同的功能,从而通过知道所述应用程序中的每一个提供了什么元素以及这些项目如何与至显示设备的再现相关而执行高效的再现。



CN 103562862 B

1.一种包括至少部分地在硬件中实现的一个或多个模块的系统，所述一个或多个模块被配置成实现可经由到多个应用程序的多个应用编程接口访问的单个合成引擎，所述合成引擎被配置成接收描述来自所述多个应用程序的每一个的、要被所述合成引擎再现以便显示在显示设备上的元素的数据，并且进一步实现对象数据库模块，其特征在于，所述对象数据库模块和所述单个合成引擎驻留在计算设备的内核中，所述对象数据库模块将所述数据的至少一部分从所述多个应用编程接口传达至所述单个合成引擎，并且所述对象数据库模块执行验证以确定所述元素是否符合用于由所述单个合成引擎再现的标准。

2.如权利要求1所述的系统，其中所述元素包括与第一所述应用程序对应的第一窗口和与第二所述应用程序对应的第二窗口。

3.如权利要求1所述的系统，其中第一所述元素由第一所述应用程序提供并且第二所述元素由第二所述应用程序提供。

4.如权利要求3所述的系统，其中所述一个或多个模块被配置成：

确定第一所述元素在被再现时是否会被第二所述元素遮挡；并且

响应于第一所述元素会被遮挡的确定，跳过第一所述元素的再现。

5.如权利要求1所述的系统，其中所述数据的所述至少一部分描述所述元素的列表的开始和结束，并且所述对象数据库模块被配置成响应于从对应的所述应用程序接收到所述列表的结束的指示而向所述合成引擎提供所述元素的列表。

6.如权利要求1所述的系统，其中所述对象数据库模块被配置成跟踪：所述元素中的哪些元素对应于所述多个应用程序中的哪个应用程序。

7.如权利要求1所述的系统，其中所述对象数据库模块被配置成将描述已改变的元素的所述数据的所述至少一部分传达至所述合成引擎，并且不传达描述未改变的元素的所述数据的另一部分。

全局合成系统

技术领域

[0001] 本发明涉及全局合成技术，更具体地涉及一种被配置成执行对多个应用程序的再现的全局合成系统。

背景技术

[0002] 在计算设备上可以再现广泛而多样的元素，比如图标、窗口、动画等等。另外，典型地在计算设备上执行的应用程序的数量像每一个应用程序典型地提供的元素的数量一样持续增加，从而提供附加的功能和更丰富的用户体验。

[0003] 然而，元素的这种增加可能消耗计算设备的大量资源，比如处理器、存储器、图形硬件和其它资源。另外，这种消耗也可能对应用程序本身的执行有影响。

发明内容

[0004] 描述了一种全局合成系统。在一种或多种实现方式中，全局合成系统可以被配置成执行对多个应用程序的再现。例如，全局合成系统可以被配置成显露出应用程序可访问的一个或多个应用编程接口(API)。然后API可以被用来使得单个合成引擎执行对多个应用的再现。单个合成引擎的使用可以被用来通过知道每一个应用程序提供了什么元素以及这些项目如何与至显示设备的再现相关而执行高效的再现。

[0005] 本发明内容被提供以按简化的形式介绍将在以下具体实施方式中进一步描述的概念的选择。本发明内容不旨在标识要求保护的主题的关键特征或必要特征，也不旨在被用作确定要求保护的主题的范围的辅助。

附图说明

[0006] 参照附图描述具体实施方式。在附图中，附图标记的最左端的(多个)数字标识该附图标记初次出现的图。在说明书和附图中不同情况下相同附图标记的使用可以表示相似或相同的项目。

[0007] 图1是可操作来实现全局合成系统的示例实施方式中的环境的图示。

[0008] 图2示出了一个示例系统，其中全局合成系统被更详细地图示为包括图1的合成引擎以及用户模式库和对象数据库模块。

[0009] 图3描绘了被配置成全局合成树的图表的示例，该全局合成树可由合成引擎消耗以再现元素。

[0010] 图4是描绘示例实施方式中的过程的流程图，其中生成描述用于由单个合成引擎再现的元素的图表。

[0011] 图5图示了如包括参照图1描述的计算设备的示例系统。

[0012] 图6图示了示例设备的各种组件，该示例设备可以被实现为如参照图1、2和5描述的、实现本文描述的技术的实施例的任何类型的计算设备。

具体实施方式

[0013] 概述

[0014] 被计算设备用来再现元素的常规技术采用分布式系统，在该分布式系统中为每个应用程序分配对应的合成引擎。因为这一点，不同的合成引擎不知道其它合成引擎正在执行什么。这可能导致可能妨碍实现常规技术的计算设备的效率的冗余、对元素的不必要再现等等，并且因而可能会使这些常规技术不太适合被“单薄”计算设备所使用。

[0015] 本文描述了全局合成技术。在一个或多个窗口中，单个合成引擎可由多个不同的应用程序经由一个或多个API访问。因而，可以使合成引擎“知道”各种应用程序正在贡献什么以及那些元素如何相关。然后可以利用该知识来提高对应用程序的元素的再现的效率。在一个或多个实现方式中，合成引擎与属于那些应用程序的线程异步地运行，这允许应用程序促使内容在它们的窗口内被制成动画并且使用不同的再现技术对这样的内容进行栅格化。此外，来自系统中的每个应用程序的合成数据可以在单个图表(例如全局合成树谱)中被管理，这允许合成引擎执行诸如遮挡检测之类的全局优化以及以高效且安全的方式对来自多个应用程序的内容进行混合和匹配。结合以下附图可以找出这些和其它技术的进一步讨论。

[0016] 在以下讨论中，首先描述可以采用本文描述的技术的示例环境。然后描述可以在示例环境以及其它环境中执行的示例过程。结果，示例过程的执行不限于示例环境，并且示例环境不限于示例过程的执行。

[0017] 示例环境

[0018] 图1是可被操作以采用本文描述的技术的示例实现方式中的环境100的图示。图示的环境100包括经过网络106通信地耦合至web服务104的计算设备102。可以以多种方式配置计算设备102以及可以实现web服务104的计算设备。

[0019] 例如，计算设备可以被配置为能够在网络106上通信的计算机，比如台式计算机、移动站、娱乐电器、通信地耦合至显示设备的机顶盒、无线电话、游戏控制台等等。因而，计算设备102的范围可以是从具有大量存储器和处理器资源的全资源设备(例如，个人计算机、游戏控制台)到具有有限存储器和/或处理资源的低资源设备(例如，传统的机顶盒、手持游戏控制台)。此外，尽管示出了单个计算设备102，但计算设备102可以代表多个不同设备，例如商业上用以执行比如web服务104操作的多个服务器、遥控器和机顶盒组合、被配置成捕获手势的图像捕获设备和游戏控制台等等。

[0020] 尽管网络106被图示为因特网，但该网络可以采取广泛而多样的配置。例如，网络106可以包括广域网(WAN)、局域网(LAN)、无线网络、公共电话网络、内联网等。另外，尽管示出了单个网络106，但网络106可以被配置成包括多个网络。

[0021] 计算设备102进一步被图示为包括操作系统108。操作系统108被配置成抽取计算设备102的底层功能给可在计算设备102上执行的应用110。例如，操作系统108可以抽取计算设备102的处理、记忆、网络和/或显示功能，使得应用程序110可以在不知道“如何”实现该底层功能的情况下被写入。应用程序110例如可以向操作系统108提供数据以便由显示设备112再现和显示，而不必理解该再现会如何执行。

[0022] 操作系统108还可以表示多种其它功能，例如管理文件系统和可由计算设备102的

用户导航的用户界面。其一个示例被图示为显示在计算设备102的显示设备112上的桌面。

[0023] 操作系统108还被图示为包括全局合成系统114。全局合成系统114可以表示这样的系统：该系统包括被配置成允许应用程序110使用单个全局合成引擎116(以下也简称为合成引擎116)在显示设备112上绘制项目的直接合成组件。尽管被图示为操作系统108的一部分，但全局合成系统114可以以多种其它方式来实现，例如作为浏览器的一部分、作为孤立的模块等等。另外，全局合成系统114可以跨网络106分布，其一个示例被图示为在web服务104上包括全局合成系统118。

[0024] 用户体验(例如为应用程序110生成的用户界面)可以包括大量的可以相互交互的元素，比如窗口、动画(例如文本滚动)等。例如，第一窗口可以对应于电子邮件应用程序，而第二窗口122可以对应于如在显示设备112上图示的浏览器。因此，在任何一个特定的时间点可以使众多不同的再现组件参与。另外，这些不同的元素可以具有不同的刷新率，比如伴随有视频和静态文本的“自动收报机(ticker)”显示的动画。

[0025] 全局合成系统114可以被用来抽取该功能，使得不同的应用程序110可以卸载该再现并且因而不知道如何执行再现。例如，应用程序110可以提供描述要再现的元素、元素的放置以及元素如何彼此相关的数据。

[0026] 此外，全局合成系统114可以支持“独立的”动画。例如，应用程序110可以传达动画的声明描述，其描述了如何再现该动画。例如，该描述可以描述什么正在被制成动画、重绘发生的速率、动画开始的位置、动画移动的曲线、动画的结束位置、动画被再现的时间量等等。

[0027] 然后可以由全局合成系统114执行该再现并且在无进一步指令的情况下继续该再现。以这种方式，动画的再现不依赖于调用者(例如，应用程序110)，从而与应用程序110的通信的损失、应用程序110的不一致处理等等对动画的再现没有影响。从而，这可以被用来通过削减应用程序110为再现动画所进行的通信的数量来改进动画的“平滑度”和“流畅度”以及计算设备102(乃至以下进一步描述的网络106)的资源。

[0028] 按照惯例，由计算设备102执行的每一个应用程序110与对应的合成引擎交互以执行用于相应的应用的显示的处理和再现。因此，常规合成引擎经常“按进程”实现。因而，常规技术可以涉及在任一时间执行多个不同的合成引擎。此外，常规的合成引擎典型地“不知道”由其它的合成引擎所执行的再现。这可能导致计算设备102的资源的低效率使用，比如绘制窗口，即使它可能被另一窗口遮挡，这可能导致处理器、图形处理器、存储器和计算设备102的其它资源的不必要的使用。

[0029] 在一个或多个实现方式中，全局合成系统114遵循全局架构，使得多个应用程序110可以例如经过应用编程接口访问合成引擎116。例如，单个合成引擎116可以负责整个桌面和来自当前正在被执行的应用程序110的元素。因而，单个合成引擎116可以“知道”要再现的用于多个应用程序的不同元素，并且相应地做出反应。

[0030] 继续前一示例，合成引擎116可以知道应用程序的元素(例如窗口)将会被再现的用于另一应用的元素(例如另一窗口)遮挡。如图1所示，例如，应用程序可以具有要在与电子邮件应用程序对应的窗口120后面显示的窗口。以前，即使应用程序的对应窗口对于计算设备102的用户不可见，该窗口仍被再现。然而，合成引擎116可以使用本技术来跳过被遮挡窗口的再现，从而节约计算设备102的资源。

[0031] 全局合成系统114可以被用来提供广泛而多样的功能。如前面所描述的，全局合成系统114可以执行对要由多个应用程序再现的事物的全局分析并且高效地确定要绘制其中的哪些元素，比如在其中如前面描述的可以跳过元素的再现的遮挡的情况下。此外，资源可以被全局合成系统114集中(pool)起来，以便由多个应用程序110共享中间存储器，而不是如可被常规总和引擎消耗(既被引擎本身消耗也被由引擎正在绘制的事物消耗)的碎片化的存储器。

[0032] 全局合成系统114还可以支持安全技术。例如，全局合成系统114可以被用来绘制受保护的视频，比如加密的电影和对应的许可。对于常规的合成引擎，应用程序将内容合成，然后将内容传送至对应的合成引擎。这可能导致内容在无保护的情况下“明文(in the clear)传送”(例如，未压缩的视频帧的传送)，并且因而内容可能暴露给恶意方。

[0033] 然而，在本文描述的一个或多个实现方式中，内容可以被提供给全局合成系统114，并且可以在应用程序110不可直接接触的受保护区域内执行该内容。因而，对合成引擎116不进一步传达内容的信任可以被用来保护该内容。例如，可以依赖合成引擎来生成供再现用的像素，而不将这些像素暴露给由计算设备102所执行的应用程序。因而，该“单向流动”可以帮助确保错误的应用程序(例如，来自恶意实体)不接收不受保护的内容。

[0034] 另外，常规的合成引擎和对应的应用程序“拥有”显示设备112的屏幕的特定部分。因此，一些显示技术难以支持使用这些常规技术。一种这样的技术是对于多个应用程序的透明性，因为显示设备112的区域可能被一个或另一个常规的合成引擎所拥有，但不能被两者都拥有。例如，显示设备112的特定区域可以涉及来自至少两个不同应用程序的窗口和它们对应的常规合成引擎。按惯例，为了支持诸如透明性之类的技术，将窗口的每个实例绘制到存储器，然后将效果应用到资源密集的组合。

[0035] 然而，因为全局合成系统114可以意识到不同的窗口并且相应地做出反应，所以可以在不事先将窗口单独地绘制到存储器并且然后将视觉效果应用到窗口的情况下，实现希望的结果。例如，全局合成系统114可以利用描述要被再现的元素的单个层级树并且由此“知道”元素如何相关。因此，全局合成系统114可以在无需使用常规技术执行的中间步骤的情况下直接绘制到计算设备102的存储器。因此，这些技术可以被由于存储器和/或处理限制而不能采用常规合成引擎的“单薄”计算设备所采用。

[0036] 此外，常规的合成引擎一般异步地执行，以支持不同应用程序的不同再现速率。尽管这确实支持了这样的功能：由与一个合成相关联的应用程序所指定的再现不影响由另一应用所指定的再现，但这可能导致对计算设备102资源的低效率使用。

[0037] 另外，分配给不同的常规合成引擎的优先级可能对常规的合成引擎以及相关联的应用两者都造成错误。例如，常规的合成引擎可以被给予用于被计算设备102的处理器执行的相对较高的优先级。然而，可能发生这样的情况：其中合成引擎消耗大量的资源以便支持多个动画。因此，合成引擎以及应用程序自身在面临这样的资源密集型任务时可能没有足够的可用于如预期那样执行的资源。

[0038] 在一个或多个实现方式中，合成引擎116可以被分配高优先级，例如，相对于分配给由计算设备102执行的其它线程的优先级，可以将高线程优先级分配给合成引擎116。全局合成系统114然后可以对来自不同源的元素如何被合成引擎116再现的优先级进行管理，其被合成引擎116再现被给予高优先级以确保再现发生。例如，全局合成系统114可以管理

哪些元素被更新、该更新发生的频率(例如从60Hz切换到30Hz)等。因此,全局合成系统114可以帮助提升计算设备102的资源针对其它使用的可用性,所述其他使用比如提供用于再现的元素的应用程序110。

[0039] 图2图示了一个示例系统200,其中全局合成系统114更详细地图示为包括图1的合成引擎116以及用户模式库202和对象数据库模块204。用户模式库202支持多种API 206、208,它们被图示为被相应的应用进程210、212用来与对象数据库模块204交互。在该示例系统200中合成引擎116被图示为在它自身的进程214内执行。

[0040] 多个分布式的合成引擎的常规使用保护将引擎的执行与计算设备的应用程序彼此免受彼此之害。例如,如果第一应用程序失败,则耦合至第一应用程序的第一常规合成引擎可能也失败。然而,由于常规合成引擎缺乏彼此的"知识",因而耦合至不同的常规合成引擎的第二合成引擎被所述失败保护。

[0041] 在一个或多个实现方式中,全局合成系统114可以采用技术来保护免受由提供用于再现的元素的应用程序110对合成引擎116的状态的腐化。一种这样的技术是采用操作系统108的内核216内的合成引擎116。因此,合成引擎116可以"信任"也在内核216中执行的其它组件。

[0042] 以这种方式,全局合成系统114可以采用用户模式218与内核模式220之间的"信任边界",以使得由内核执行检查。在一个或多个实现方式中,可以依赖应用程序确定由诸如参数检查之类的应用程序提供的"正确性",以提高由全局合成系统114所进行的处理的效率。

[0043] 另一种这样的技术涉及跟踪,以使得全局合成系统114"知道"哪些数据(例如用于再现的元素)属于哪个应用程序。因此,一个应用程序的失败(例如崩溃)不影响另一应用程序的元素。另外,全局合成系统114在这种失败的情况下可以"清理"元素,例如在预定量的时间后从正在被再现的元素中移除这些元素。以这种方式,全局合成系统114可以允许应用程序优雅地失败并且不影响其它应用程序和对应的元素再现。

[0044] 如图示,在专用系统进程214上执行合成引擎116,该专用系统进程214不同于诸如计算设备102的其它应用进程210、212之类的用来执行其它代码的进程。此外,该进程214可以被分配高级别的信任和优先级。进程214例如从得到合成数据的观点来看可以被信任,并且因而可以用于受保护的数据,比如受保护的视频、以权限管理保护的电子邮件等。

[0045] 如图2所示,全局合成系统114可以使用3个部分来实现。第一部分被图示为合成引擎116,其代表执行再现(即"进行绘制")到显示设备112的功能。第二部分被图示为用户模式库202,其代表应用程序通过显露应用编程接口(API)206、208调用的实体。例如,用户模式库202可以充当比如通过使用动态链接库(DLL)从应用程序接收合成数据的"信箱投递口"。

[0046] 第三部分被图示为对象数据库模块204,其被图示为驻留在内核216中。对象数据库模块204代表负责在用户模式库202与合成引擎116之间移动数据的功能。

[0047] 对象数据库模块204还可以执行验证。例如,应用程序可以调用用户模式库202以创建诸如位图之类的元素。如果所请求的元素不符合由对象数据库模块204所施行的标准(例如,少于"N"个像素),则对象数据库模块204可以向调用用户模式库202的应用程序返回失败消息。因此,对象数据库模块204可以在内核216内操作以控制提供给合成引擎116什

么。对象数据库模块204可以施行多种其它策略。

[0048] 因此，合成引擎116可以依赖由对象数据库模块204提供的、符合对象数据库模块204实现的策略的数据。换句话说，合成引擎可以假设数据是有效的和正确的并且因此适合于再现。因此，可以由对象数据库模块204执行单次验证，然后由合成引擎116利用该验证来执行再现而不需要进一步验证。

[0049] 对象数据库模块204还可以代表这样的功能：向合成引擎116通知要被再现的数据何时已改变。例如，经由用户模式库实现的API可以被配置成消耗描述要再现什么的图表，在图3中示出了其一个示例。图表300可以包括要再现的元素的列表以及在显示设备112的何处再现元素的描述，其可以包括元素在动画的情况下如何移动。

[0050] 此外，可以使用子元素来形成要再现的元素，并且因而图表300可以采取层级结构。另外，图表300可以描述元素如何被再现，例如，绘制文本一次、再现动画等等。因而，图表300可以描述元素以及元素如何彼此相关。

[0051] 图表300表示可以被合成引擎116用来再现场景的两个对象集合、被合成在一起的位图以及限定合成那些位图所依据的空间关系的视觉件(visual)。在该模型中，位图是合成引擎116的“什么”而视觉件是合成引擎116的“如何”。那些对象被布置在树结构中并且被绑定至最顶级别或子窗口以供合成。在该图中，子视觉件4也被图示为指向DX位图。这描述了合成系统的允许开发者对于多个UI元素使用相同的内容的功能，这可以帮助节省计算设备102的资源，例如存储器。

[0052] 再次返回图2，一旦被合成引擎116接收，应用程序不需要进一步调用合成引擎116以保持再现元素。因而，合成引擎116与常规技术相比可以节省资源，所述常规技术可能涉及每秒进行60次调用以便以显示设备112的刷新率来再现动画的应用程序。因而，应用程序110可以调用用户模式库202的API以构建结构，并且调用API以构建要由合成引擎再现的元素。

[0053] 为了对由合成引擎116再现的内容进行改变，应用程序可以调用用户模式库202的另一应用编程接口以更新结构和/或元素。例如，应用程序可以经由更新API来提供数据，以提供要用于股票自动收报机动画的信息。

[0054] 在一个或多个实现方式中，还可以利用批处理技术来限定要使用帧再现哪些元素。如前面所描述的，全局合成系统114可以接收多种不同元素以用于以多种不同速率再现。相应地，全局合成系统114可以支持这样的结构：其中形成要被一起再现的元素的列表。因此，全局合成系统114可以实现用于元素的开始和结束的定义，其中在开始和结束之间接收到的元素直到接收到“结束”才被再现。因而，该帧可以支持一种“全有或全无”的办法再现用于特定帧的元素，并且确保元素当希望时被一起再现以供显示。

[0055] 例如，对象数据库模块204可以跟踪应用程序的两个不同状态。第一状态可以引用用于当前显示的元素。第二状态可以引用要在第一状态之后显示并且被改变的元素。因而，第二状态可以被用来建立用于由合成引擎116再现的元素的列表，一旦列表完成(例如，一旦从应用程序接收到列表完成的指示)则进行该再现。

[0056] 一旦完成，则可以将改变发送至合成引擎116。另外，从应用程序接收到列表完成的指示的定时可以被用来确定何时显示那些改变，例如哪一帧。因此，合成引擎116可以接收在一个或多个已完成的列表中描述的一批改变，但是未被指示为完成的列表不被传达。

该批处理于是可以定义由合成引擎116再现的帧。另外，这可以帮助限制如使用不支持这样的定义的常规技术可能发生的错误的视觉伪像的显示。应当显而易见，按需要可以将与帧对应的时间量设定用于多种不同的量。

[0057] 因而，对象数据库模块204可以记住先前由合成引擎116再现了什么(例如，元素和那些元素的属性)，并且知道将要再现什么。因此，对象数据库模块204通过比较该信息来确定哪些元素被改变。相应地，对象数据库模块204可以向合成引擎116传达描述该改变的信息，而不传达未被改变的信息。

[0058] 此外，帧的使用可以进一步提高效率。例如，应用程序可以传达描述对象要被移动某一距离并且返回原始位置的数据。对象数据库模块204可以确定该移动要在单个帧的时间段内发生。相应地，对象数据库模块204可以抑制将该数据传达至合成引擎116并相反地使对象保留在其先前状态。例如，该移动可以在用来刷新显示设备112的时间段内发生，并且因而不管怎样都不会被用户看到。

[0059] 以这种方式，对象数据库模块204可以丢弃要提供给合成引擎116用于再现的列表的构建中的中间状态。然后，该列表可以以多种方式被传达，比如由合成引擎116执行以实施改变的命令阵列。另外，该技术还可以被用来解决由应用程序发送的未绑定数据的情况，因为单个情况被报告给合成引擎。

[0060] 对象数据库模块204和合成引擎116还可以采用描述改变被实现的确认技术。例如，合成引擎116可以从对象数据库模块204接收描述要做出的改变的通信。当正在做出改变时，对象数据库模块204可以等待发送附加改变，直到接收到做出了先前的改变的确认为止。一旦对象数据库模块204接收到确认，则附加改变可以被传达至合成引擎。

[0061] 另外，该技术可以被用来提供“节流”并且因此进一步节省计算设备102的资源。例如，应用程序可以做出若干错过显示设备刷新速率的请求。通过使用帧并且对改变进行批处理，减少了否则将被消耗的资源的量。

[0062] 通过调用用户模式库202的API对要显示什么做出改变的应用程序可以被多线程化。相应地，在一个或多个实现方式中，对象数据库模块204可以采用技术以使得来自单个应用程序的多个线程的多个调用不使状态恶化。这可以通过应用程序自身将其线程锁定在一起来执行，例如一个线程可以被阻塞，同时应用程序的另一线程完成任务。

[0063] 然后可以将来自多个线程的改变存储在可以由操作系统108管理的队列中，例如通过可以用于变量的联锁访问，以使得不同的线程可以在做出“完整”改变后再将控制转交给另一线程。这可以支持以上描述的帧技术以便到达要被合成引擎处理的工作的原子单元。此外，对于未接收到更新的帧，可以暂停合成引擎的执行，直到下一帧为止，从而进一步节省计算设备102的资源。

[0064] 内核216还可以结合全局合成系统114来添加多种功能。例如，当应用程序失败(例如崩溃)时，内核216可以向合成引擎116通知该发生。合成引擎116然后可以执行与在应用程序通过正常的退出过程“自然地”停止执行的情况下将会使用的那些技术类似的技术。以这种方式，合成引擎116可以到达“清理”状态以使得与应用程序对应的元素被从计算设备102的显示设备112上的显示中移除。因此，这促进了合成引擎116的执行中的鲁棒性。

[0065] 合成引擎116的执行也可以通过实施用于用户模式库202的只写API来保护安全。以这种方式，合成引擎116可以生成像素，但却不将那些像素向回暴露给应用程序，从而保

护图像不受恶意方侵害。

[0066] 如前面所描述的，这些技术的实现方式还可以涉及计算设备102“之外”的设备，它们可以跨诸如web服务104之类的一个或多个实体分布。例如，这些技术可以被采用来通过元素和属性的分批(诸如图3的图表300之类的合成树)经由网络106的通信来支持终端服务、远程桌面环境等。因此，可以在别处生成图表300(例如，通过在web服务104上实现用户模式库202和/或对象数据库模块204)并且由合成引擎116通过计算设备102的执行来传送用于再现的合成树。以这种方式，即使在网络106不可靠的情况下也可以平滑地显示动画。结合下列过程可以找到这些和其它技术的进一步讨论。

[0067] 全局合成系统114还可以被配置成利用可从硬件(例如，图形卡)获得的功能来执行操作，这意味着可以使操作更快并且展现出更好的性能。其示例包括图形处理单元(GPU)的使用以通过混合两个重叠的半透明窗口、绘制几何修剪(圆角)等向合成元素(例如，UI块、整个窗口)添加透明度。当硬件(例如，GPU)不可用时，全局合成系统114仍可以通过“落回”软件再现来执行这些操作。

[0068] 总体上，本文描述的任何功能都可以使用软件、固件、硬件(例如固定逻辑电路)或这些实现方式的组合来实现。本文使用的术语“模块”、“功能”和“引擎”总体上表示软件、固件、硬件或其组合。在软件实现方式的情况下，模块、功能或引擎表示当在处理器(例如一个或多个CPU)上执行时执行指定任务的程序代码。程序代码可以存储在一个或多个计算机可读存储器设备中。以下描述的技术的特征是平台无关的，意指这些技术可以在具有多种处理器的多种商业计算平台上实现。

[0069] 例如，计算设备102还可以包括促使计算设备102的硬件执行操作的实体(例如软件)，例如处理器、功能块等。例如，计算设备102可以包括计算机可读介质，该计算机可读介质可以被配置成保持促使计算设备(且更特别地是计算设备102的硬件)执行操作的指令。因此，这些指令起作用来配置硬件以执行操作，并且以这种方式导致硬件的变换以执行功能。这些指令可以由计算机可读介质通过多种不同的配置提供给计算设备102。

[0070] 计算机可读介质的一种这样的配置是信号承载介质，并且因而被配置成将指令(例如作为载波)比如经由网络传输至计算设备的硬件。计算机可读介质还可以被配置为计算机可读存储介质，并且因而不是信号承载介质。计算机可读存储介质的示例包括随机存取存储器(RAM)、只读存储器(ROM)、光盘、闪存、硬盘存储器和其它可以使用磁、光和其它技术存储指令和其它数据的存储器设备。

[0071] 示例过程

[0072] 下面的讨论描述可以利用先前描述的系统和设备实现的全局合成系统技术。每一过程的各方面可以在硬件、固件、软件或其组合中实现。这些过程被示出为指定由一个或多个设备执行的操作的块的集合，并且不一定限于图示的按各块执行操作的次序。在以下讨论的部分中，将参照图1的环境100、图2的系统200和图3的图表300。

[0073] 图4描绘了示例实现方式中的过程400，其中生成描述用于由单个合成引擎再现的元素的图表。生成描述已被指定用于由一个或多个计算设备执行的多个应用再现的元素的图表(块402)。如图3所示，图表300可以包括要再现的元素的列表以及关于在显示设备112上的何处再现元素的描述，它可以包括在动画的情况下元素如何移动。此外，可以使用子元素来形成要再现的元素，并且因而图表300可以采取层级结构。另外，图表300可以描述元素

如何被再现。因而图表300可以描述元素以及元素如何彼此相关。多种其它图表也被考虑。

[0074] 该图表被传达以由单个合成引擎接收，以使得元素被再现(块404)。这可以以多种方式执行。例如，图表可以从web服务经由网络传达以由执行单个合成引擎的计算设备接收(块406)，例如从web服务104的全局合成系统118经由网络106到计算设备102的合成引擎116。在另一示例中，图表可以从对象数据库模块传达至单个合成引擎(块408)。对象数据库模块204例如可以基于从多个应用程序接收的数据生成图表，并且将图表传达至合成引擎以引用被更新的元素。多种其它示例也被考虑。

[0075] 示例系统和设备

[0076] 图5图示了包括如参照图1描述的计算设备102的示例系统500。示例系统500当在个人计算机(PC)、电视设备和/或移动设备上运行应用程序时实现用于无缝用户体验的普适环境。服务和应用程序当在利用应用程序、播放视频游戏、观看视频等等的同时从一个设备转换到下一个设备时，对于普通用户体验在全部三种环境中基本相似地运行。在图示的情况中，计算设备102被图示为实现以上描述的全局合成系统114的至少一部分。

[0077] 在示例系统500中，多个设备通过中央计算设备相互连接。中央计算设备可以是所述多个设备的本地设备，或者可以远离所述多个设备定位。在一个实施例中，中央计算设备可以是通过网络(因特网)或其它数据通信链路连接至多个设备的一个或多个服务器计算机的云。在一个实施例中，该互连架构实现了跨多个设备递送的功能，以向多个设备的用户提供共同的和无缝的体验。多个设备中的每一个可以具有不同的物理需求和能力，并且中央计算设备使用平台实现将既针对设备定制又对所有设备而言公共的体验递送至设备。在一个实施例中，创建目标设备的类并且针对设备的通用类定制体验。可以由设备的物理特征、使用类型或其它共同特性来定义设备的类别。

[0078] 在各种实现方式中，计算设备102可以采取比如用于计算机502、移动装置504和电视506使用的多种不同配置。这些配置中的每一种包括可以具有总体上不同的构造和能力的设备，并且因而计算设备102可以根据一种或多种不同的设备类来配置。例如，计算设备102可以被实现为包括个人计算机、桌面计算机、多屏幕计算机、膝上型计算机、上网本等的计算机502设备类。

[0079] 计算设备102还可以被实现为包括诸如移动电话、便携音乐播放器、便携游戏设备、平板计算机、多屏幕计算机等之类的移动设备的移动502设备类。计算设备102还可以被实现为包括具有或连接至在休闲的观看环境中的总体上更大的屏幕的设备的电视506设备类。这些设备包括电视机、机顶盒、游戏控制台等。本文描述的这些技术可以得到计算设备102的这些各种配置的支持并且不限于这些技术在本文描述的具体示例。

[0080] 云508包括和/或代表用于内容服务512的平台510。平台510将云508的硬件(例如服务器)和软件资源的底层功能抽出。平台被图示为支持全局合成系统118的至少一部分，比如用户模式库和/或对象数据库模块204。因此，平台可以帮助支持全局合成系统118的全部或部分的远程处理。

[0081] 平台510可以将资源和功能抽出以将计算设备102与其它计算设备连接。平台510还可以用来将资源的定标抽出以针对所遭遇的经由平台510实现的全局合成系统118的需求提供相应的规模水平。相应地，在互连设备实施例中，本文描述的功能的功能实现方式可以遍布系统500分布。例如，功能可以部分地在计算设备102上实现，以及经由将云508的功

能抽出的平台510来实现。

[0082] 图6示出了示例设备600的各种组件，该示例设备可以被实现为参照图1、2和5描述的、实现本文描述的技术的实施例的任何类型的计算设备。设备600包括实现设备数据604(例如，接收到的数据、正在接收的数据、计划用于广播的数据、数据的数据分组，等等)的有线和/或无线通信的通信设备602。设备数据604或其它设备内容可以包括设备的配置设定、存储在设备上的媒体内容和/或与设备的用户关联的信息。存储在设备600上的媒体内容可以包括任何类型的音频、视频和/或图像数据。设备600包括一个或多个数据输入606，经过数据输入606可以接收到任何类型的数据、媒体内容和/或输入，比如用户可选输入、消息、音乐、电视媒体内容、记录的视频内容和从任何内容和/或数据源接收到的任何其他类型的音频、视频和/或图像数据。

[0083] 设备600还包括通信接口608，该通信接口608可以被实现为串行和/或并行接口、无线接口、任何类型的网络接口、调制解调器中的任何一种或多种并且可以被实现为任何其它类型的通信接口。通信接口608提供其它电子、计算和通信设备与设备600传达数据所用的通信网络与设备600之间的连接和/或通信链路。

[0084] 设备600包括处理各种计算机可执行指令以控制设备600的操作并且实现本文描述的技术的实施例的一个或多个处理器610(例如，任何微处理器、控制器等)。可替代地或此外，设备600可以利用与通常标以612的处理和控制电路结合地实现的硬件、固件或固定逻辑电路中的任何一个或组合来实现。尽管未示出，但设备600可以包括耦合设备内的各种组件的系统总线或数据传送系统。系统总线可以包括诸如存储器总线或存储器控制器、外围总线、通用串行总线和/或利用多种总线架构中任一种的处理器或局部总线之类的不同总线结构的任何一种或组合。

[0085] 设备600还包括诸如一个或多个存储器组件之类的计算机可读介质614，其示例包括随机存取存储器(RAM)、非易失性存储器(例如，只读存储器(ROM)、闪存、EPROM、EEPROM等中的任何一个或多个)和盘存储设备。盘存储设备可以被实现为任何类型的磁或光存储设备，比如硬盘驱动器、可记录和/或可重写的光盘(CD)、任何类型的数字多用盘(DVD)等。设备600还可以包括大容量存储介质设备616。

[0086] 计算机可读介质614提供数据存储机制以存储设备数据604以及各种设备应用程序618和与设备600的操作方面相关的任何其它类型的信息和/或数据。例如，操作系统620可以利用计算机可读介质614被保持为计算机应用程序，并且在处理器610上执行。设备应用程序618可以包括设备管理器(例如，控制应用程序、软件应用程序、信号处理和控制模块、特定设备固有的代码、用于特定设备的硬件抽象层等)。设备应用程序618还包括实现本文描述的技术的实施例的任何系统组件或模块。在该示例中，设备应用程序618包括被示出为软件模块和/或计算机应用程序的接口应用程序622和输入/输出模块624。输入/输出模块624代表被用来提供与诸如触摸屏、轨迹板、照相机、话筒等之类的被配置成捕获输入的设备的接口的软件。可替代地或此外，接口应用程序622和输入/输出模块624可以被实现为硬件、软件、固件或其任何组合。此外，输入/输出模块624可以被配置成支持多个输入设备，比如分别捕获视频和音频输入的单独设备。

[0087] 设备600还包括向音频系统628提供音频数据和/或向显示系统630提供视频数据的音频和/或视频输入-输出系统626。音频系统628和/或显示系统630可以包括处理、显示

和/或以其他方式再现音频、视频和图像数据的任何设备。视频信号和音频信号可以从设备600经由RF(射频)链路、S-视频链路、合成视频链路、分量视频链路、DVI(数字视频接口)、模拟音频连接或其它类似的通信链路传达至音频设备和/或显示设备。在一种实施例中，音频系统628和/或显示系统630被实现为设备600的外部组件。可替代地，音频系统628和/或显示系统630被实现为示例设备600的集成组件。

[0088] 结论

[0089] 尽管以特定于结构特征和/或方法动作的语言描述了本发明，但应当理解，所附权利要求中限定的本发明不一定限于所描述的特定特征或动作。相反，这些特定特征和动作是作为实现要求保护的本发明的示例形式而公开的。

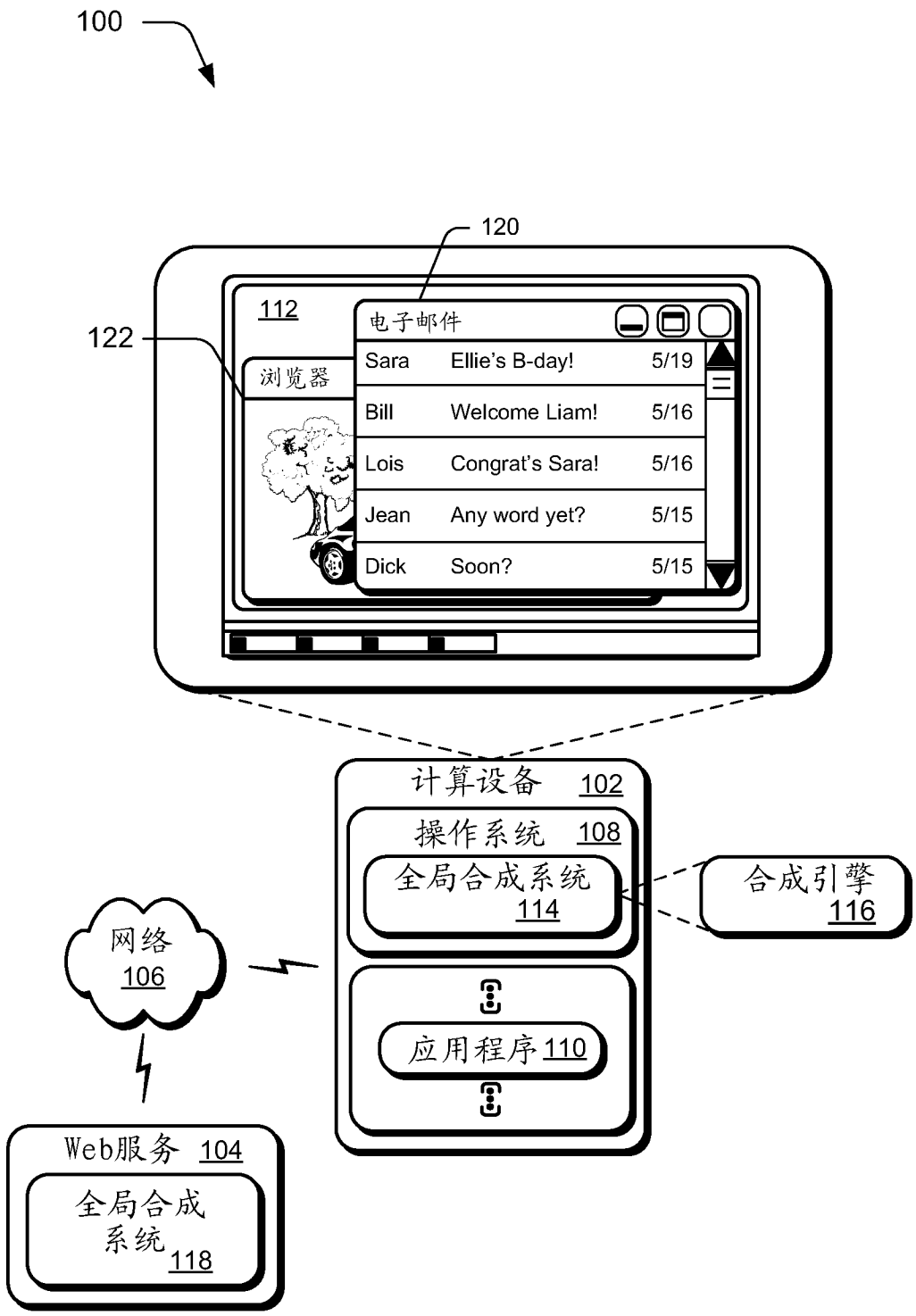

100
120
112
122
电子邮件
浏览器
Sara Ellie's B-day! 5/19
Bill Welcome Liam! 5/16
Lois Congrat's Sara! 5/16
Jean Any word yet? 5/15
Dick Soon? 5/15
计算设备 102
操作系统 108
全局合成系统 114
合成引擎 116
网络 106
应用程序 110
Web服务 104
全局合成系统 118

图 1

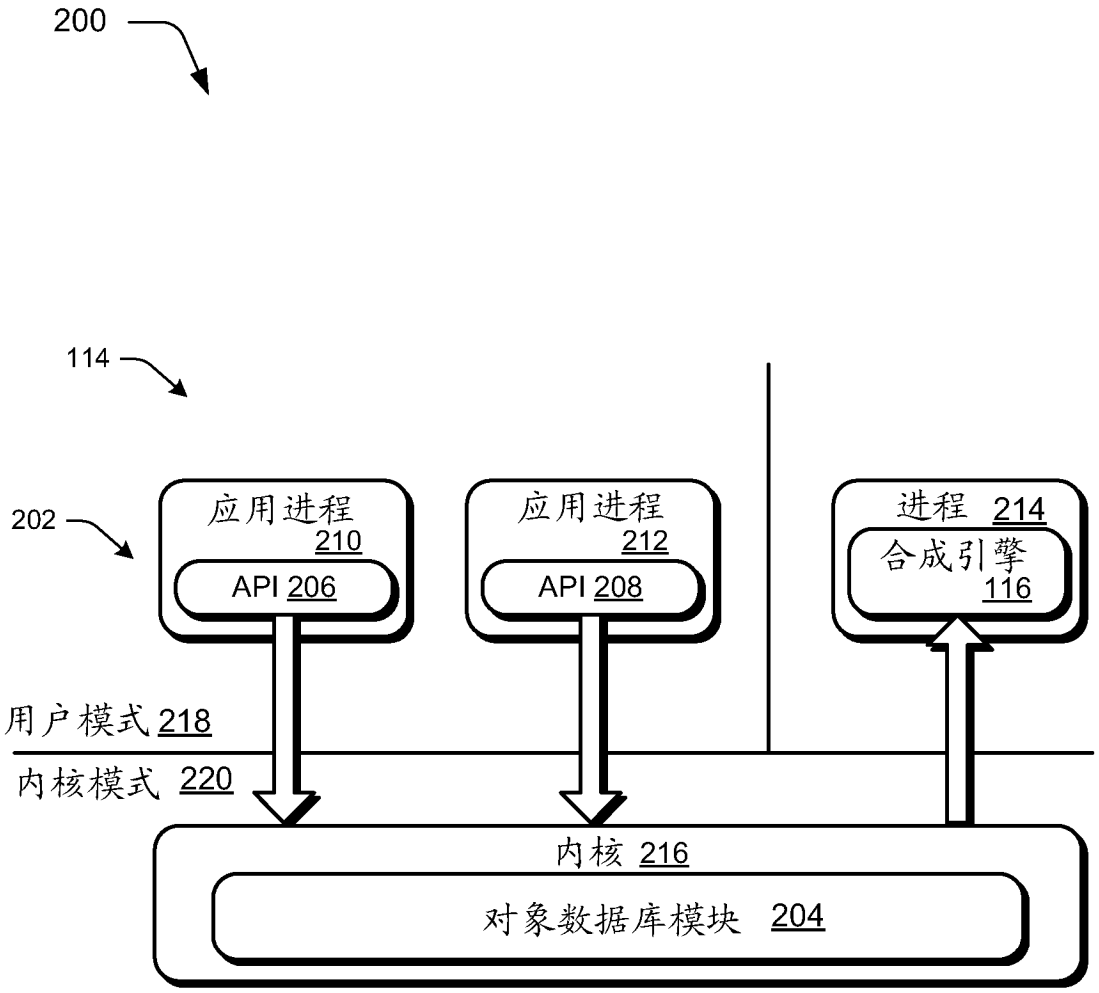

200
114
202
应用进程
210
API 206
应用进程
212
API 208
进程 214
合成引擎
116
用户模式 218
内核模式 220
内核 216
对象数据库模块 204

图 2

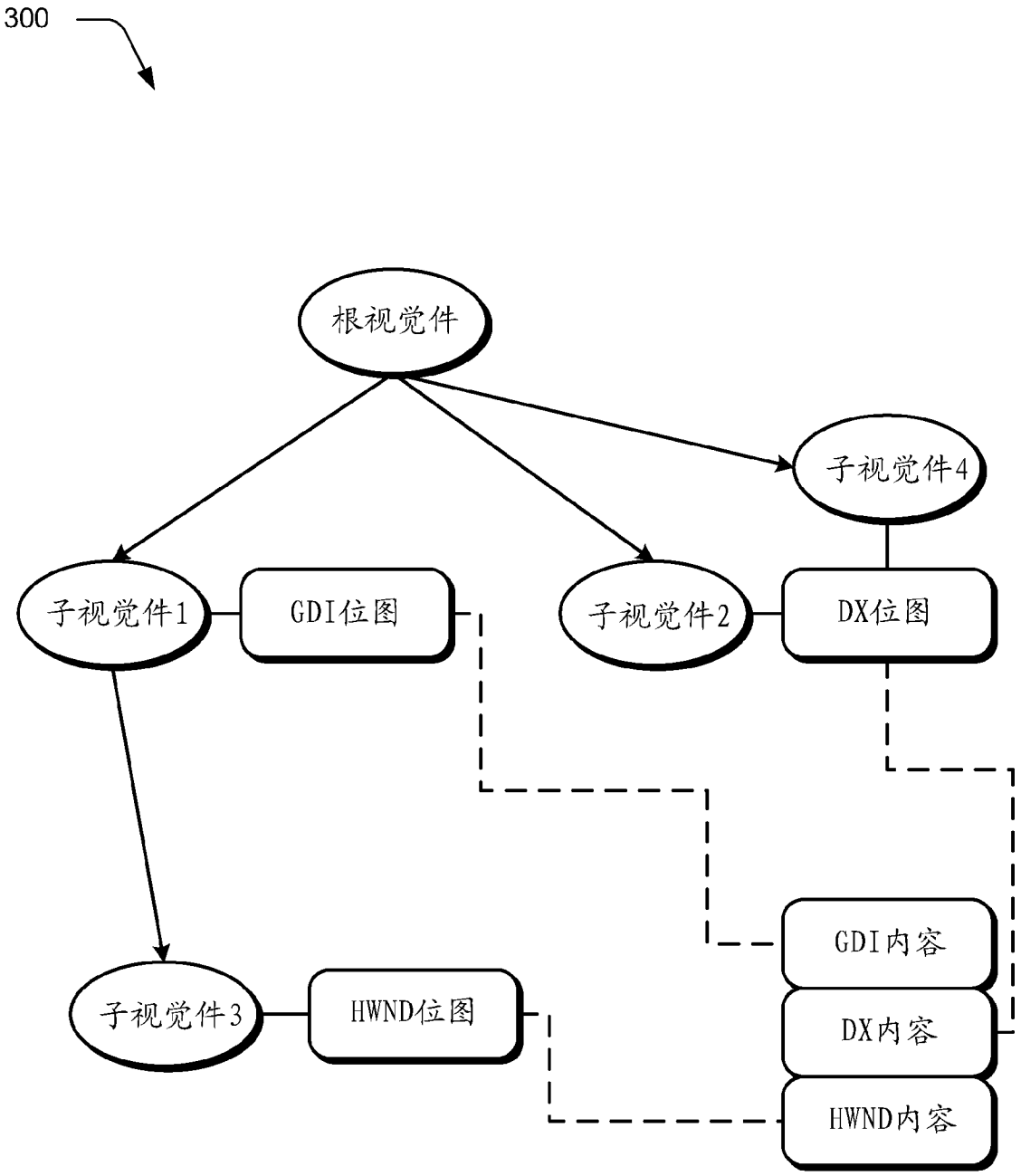

300
根视觉件
子视觉件4
子视觉件1
GDI位图
子视觉件2
DX位图
GDI内容
子视觉件3
HWND位图
DX内容
HWND内容

图 3

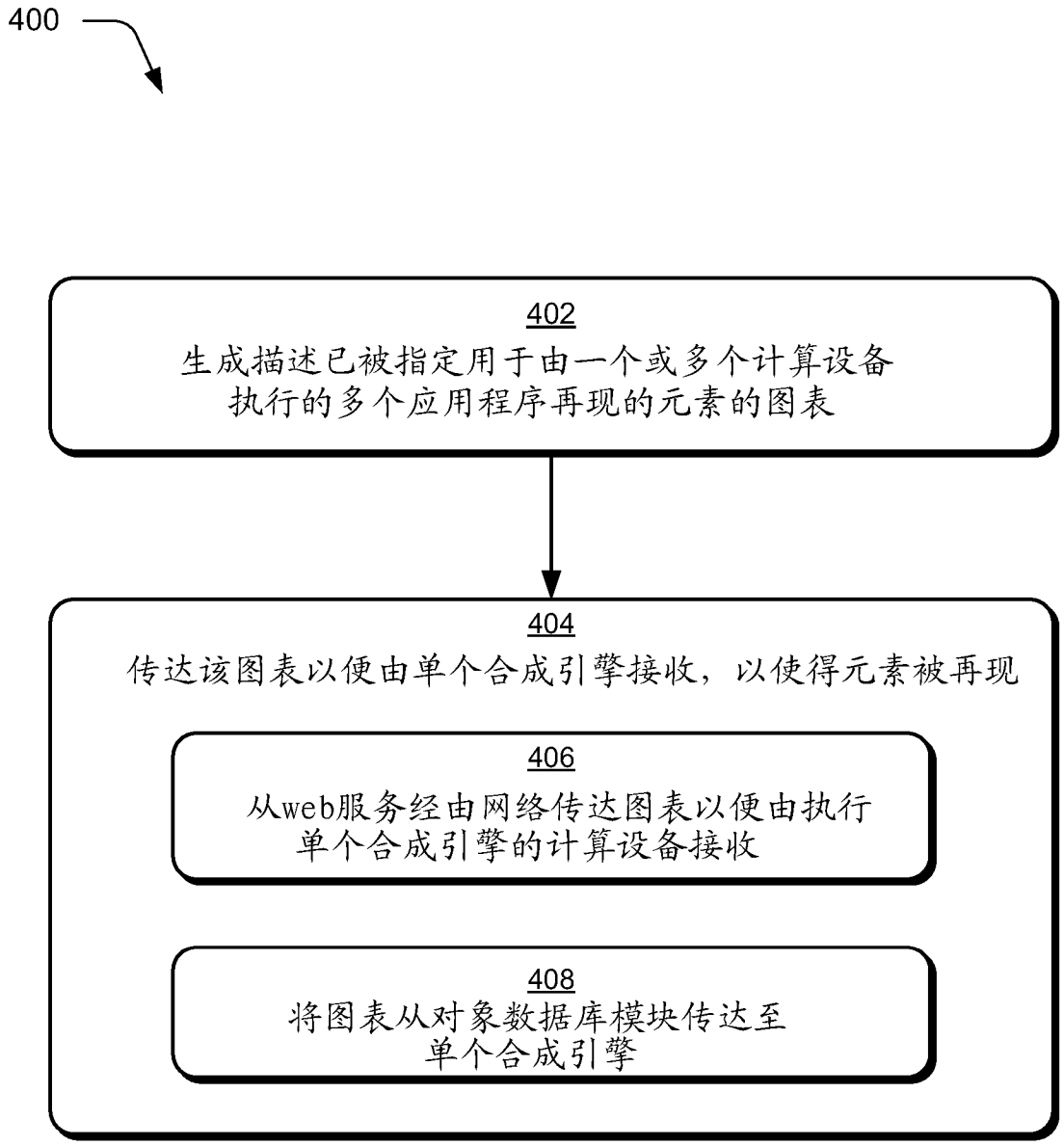

400
402
生成描述已被指定用于由一个或多个计算设备执行的多个应用程序再现的元素的图表
404
传达该图表以便由单个合成引擎接收，以使得元素被再现
406
从web服务经由网络传达图表以便由执行单个合成引擎的计算设备接收
408
将图表从对象数据库模块传达至单个合成引擎

图 4

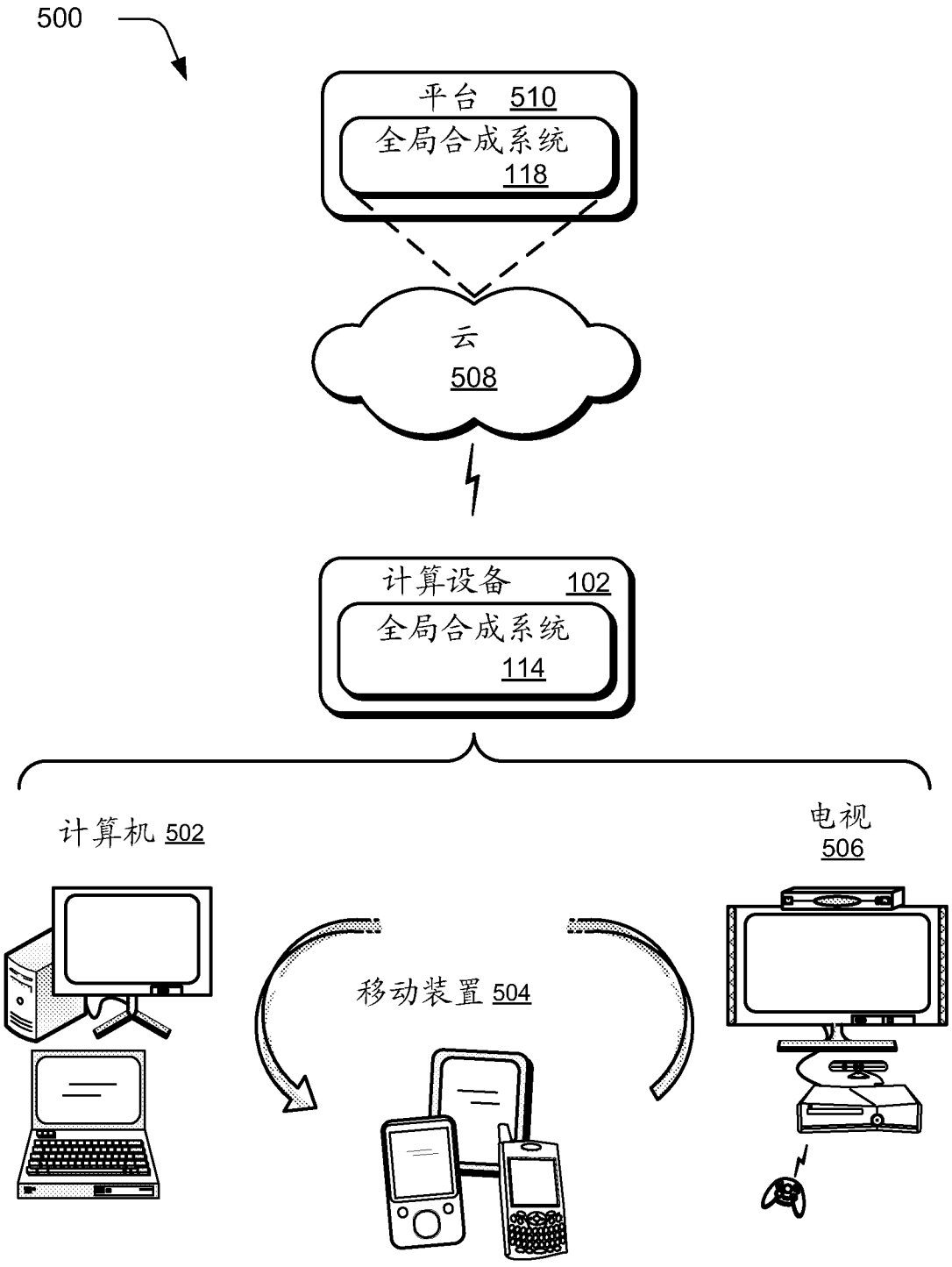

500
平台 510
全局合成系统
118
云
508
计算设备 102
全局合成系统
114
计算机 502
移动装置 504
电视
506

图 5

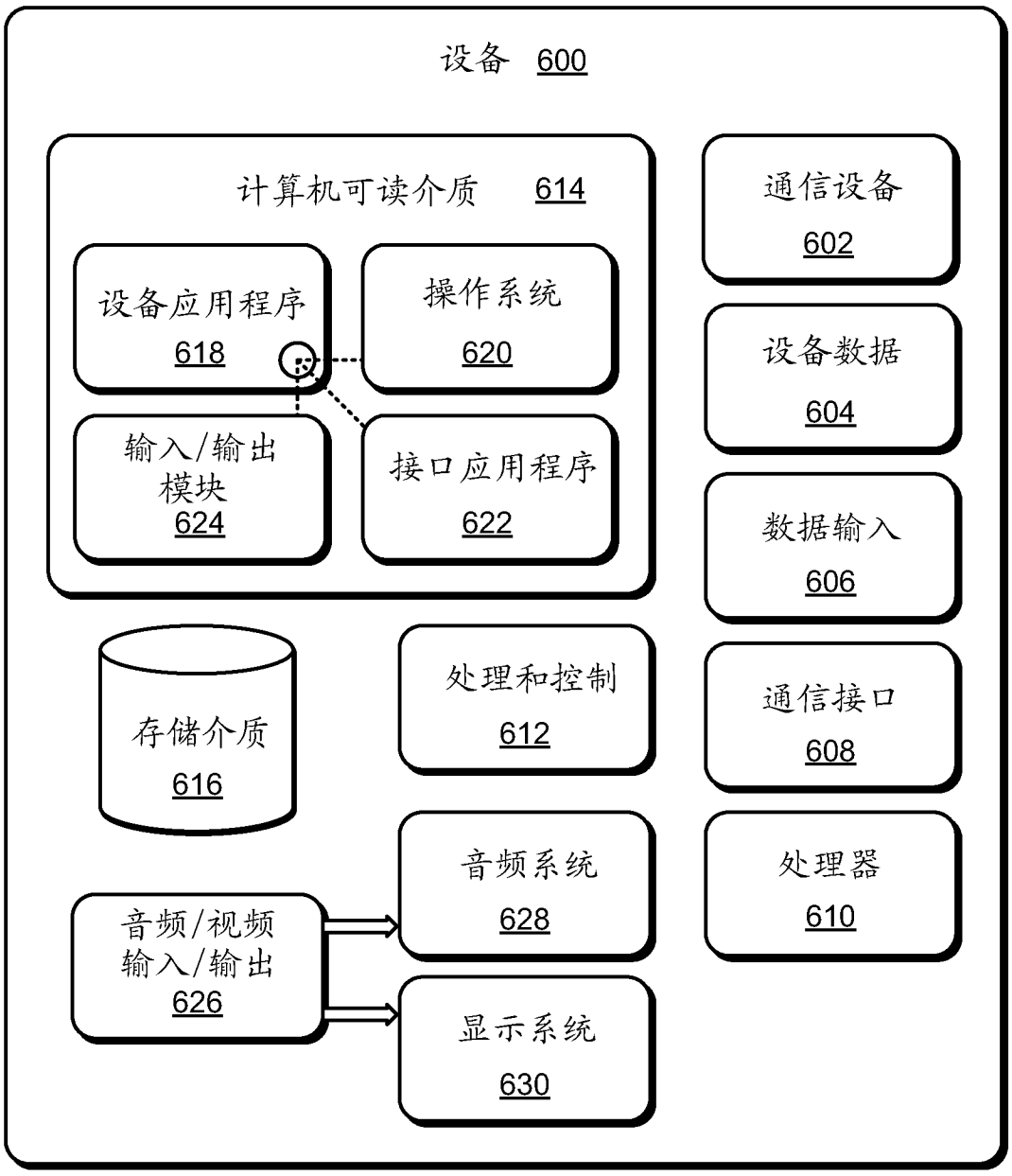

设备 600
计算机可读介质 614
设备应用程序 618
操作系统 620
输入/输出模块 624
接口应用程序 622
通信设备 602
设备数据 604
数据输入 606
通信接口 608
处理器 610
处理和控制 612
存储介质 616
音频系统 628
音频/视频输入/输出 626
显示系统 630

图 6