(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 103003815 B
(45)授权公告日 2014.04.16

(21)申请号 201180030891.X
(22)申请日 2011.06.13
(30)优先权数据
12/821,971 2010.06.23 US
(85)PCT国际申请进入国家阶段日
2012.12.21
(86)PCT国际申请的申请数据
PCT/US2011/040235 2011.06.13
(87)PCT国际申请的公布数据
WO2011/163001 EN 2011.12.29
(73)专利权人 微软公司
地址 美国华盛顿州
(72)发明人 M·S·奥古斯汀 J·博克哈德特
B·M·兰伯特 R·E·奥齐
J·E·施莱费尔 R·Z·斯派尔
P·S·苏塞
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 陈斌
(51)Int.Cl.
G06F 17/40(2006.01)
G06F 9/44(2006.01)
(56)对比文件
US 7120687 B1,2006.10.10,
CN 101615203 A,2009.12.30,
CN 101495976 A,2009.07.29,
CN 102893272 A,2013.01.23,
审查员 姜晓庆

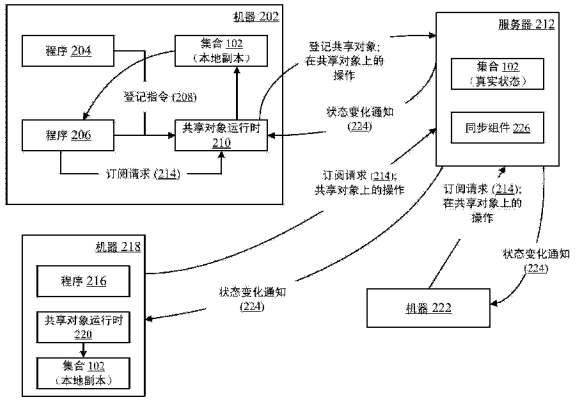
权利要求书2页 说明书9页 附图6页

(54)发明名称

共享的数据集合

(57)摘要

数据共享机制可允许程序共享对于数据集合的访问。实现共享的机制可允许以任何语言编写的程序读取和写入共享集合。通过允许程序或多或少以数据是纯粹本地数据的方式在数据上操作，该机制可使得集合的共享特性相对于程序和编程者是透明的。集合的共享可由在其上使用集合的每一个机器上的共享对象运行时、和由共享对象服务器来管理。共享对象服务器维持集合的真实状态，且当程序在不了解彼此操作的情况下操作同一个集合时，确定性地解决集合。通过其来共享集合的机制可被实现为对于集合中数据种类不可知。



CN 103003815 B

1. 一种帮助数据集合共享的方法，所述方法包括：

接收所述数据集合的登记，其中所述数据集合由第一程序创建，其中所述登记从在其上执行所述第一程序的机器上的对象运行时处接收；

从与所述第一程序不同的第二程序处接收对于所述数据集合的订阅，其中存在对于所述数据集合的一组订阅者，且其中所述一组订阅者包括所述第一程序和所述第二程序；

接收第一通知，所述订阅者中的第一个已经在所述数据集合上执行了第一操作，对于所述数据集合的状态做出改变；和

传播所述状态的所述改变至所述一组订阅者；

其中，所述方法还包括：

接收第二通知，所述订阅者中的第二个已经在所述数据集合上执行了第二操作，所述第二操作不同于所述第一操作；和

通过将所述第二操作相对于所述第一操作转换，来解决所述第一操作和所述第二操作之间的冲突。

2. 如权利要求 1 所述的方法，其特征在于，所述第二通知使用所述数据集合的项的顺序位置描述所述第二操作，其中取决于是在所述第二操作前执行还是在所述第二操作后执行所述第一操作，所述顺序位置是指不同的项。

3. 如权利要求 1 所述的方法，其特征在于，还包括：

维持所述数据集合的副本在所述数据集合的真实状态中；且其中，所述传播包括：

将所述真实状态通信至所述订阅者。

4. 如权利要求 1 所述的方法，其特征在于，所述传播包括：

将相对于所述数据集合的状态的所述订阅者版本的改变描述通信至所述订阅者，所述改变源自执行所述第一操作。

5. 如权利要求 1 所述的方法，其特征在于，执行所述方法，而不考虑在所述数据集合中的项的内容的结构如何。

6. 如权利要求 1 所述的方法，其特征在于，所述第一通知包括在所述数据集合上执行的操作的类型、正被操作的项在所述数据集合中的位置、和所述数据集合的状态。

7. 一种帮助数据集合共享的系统，所述系统包括：

用于接收所述数据集合的登记的装置，其中所述数据集合由第一程序创建，其中所述登记从在其上执行所述第一程序的机器上的对象运行时处接收；

用于从与所述第一程序不同的第二程序处接收对于所述数据集合的订阅的装置，其中存在对于所述数据集合的一组订阅者，且其中所述一组订阅者包括所述第一程序和所述第二程序；

用于接收第一通知的装置，所述订阅者中的第一个已经在所述数据集合上执行了第一操作，对于所述数据集合的状态做出改变；和

用于传播所述状态的所述改变至所述一组订阅者的装置；

其中，所述系统还包括：

用于接收第二通知的装置，所述订阅者中的第二个已经在所述数据集合上执行了第二操作，所述第二操作不同于所述第一操作；和

用于通过将所述第二操作相对于所述第一操作转换，来解决所述第一操作和所述第二

操作之间的冲突的装置。

8. 如权利要求 7 所述的系统，其特征在于，所述第二通知使用所述数据集合的项的顺序位置描述所述第二操作，其中取决于是在所述第二操作前执行还是在所述第二操作后执行所述第一操作，所述顺序位置是指不同项。

9. 如权利要求 7 所述的系统，其特征在于，还包括用于维持所述数据集合的副本在所述数据集合的真实状态中的装置，且其中所述用于传播的装置包括用于将所述真实状态通信至所述订阅者的装置。

10. 如权利要求 7 所述的系统，其特征在于，所述用于传播的装置包括用于将相对于所述数据集合的状态的所述订阅者版本的改变的描述通信至所述订阅者的装置，所述改变源自执行所述第一操作。

11. 如权利要求 7 所述的系统，其特征在于，执行所述系统而不考虑所述数据集合中项的内容结构如何。

12. 如权利要求 7 所述的系统，其特征在于，所述第一通知包括在所述数据集合上执行的操作的类型、正被操作的项在所述数据集合中的位置、和所述数据集合的状态。

共享的数据集合

背景技术

[0001] 集合是将被一起操作的一组数据项。集合的一些示例包括列表、阵列、集、包、和各种其他数据组。在编程的早期，程序一般是不彼此互操作的单片实体。因此，程序能采用编程者选择的任何方式的集合内在地管理集合。在现代编程中，然而，通过操纵共享的数据组，使得同一程序的不同实例、或不同程序彼此互操作，已成为较为流行的实践。

[0002] 尽管可能使得程序实现它们用于共享数据的自身机制，这样做一般对于编程者而言是麻烦的。编程者可能必须将共享机制实现为程序的紧密结合的一部分。即使编程者可获得已知共享机制的代码，该机制一般专用于程序的特性、且专用于被共享的数据类型。且，当其他程序想要与已有程序共享数据时，这些程序必须以使用相同共享机制的方式被实现。

[0003] 尽管允许程序共享数据出现了各种问题，集合的共享提出了附加问题。对于很多类型的集合，集合的当前状态不仅由集合的内容定义、还由其中这些内容所出现的顺序所定义。例如，阵列 {1,2,3,4,5} 不同于阵列 {2,3,1,5,4}。即使这两个阵列包含了相同的底层元素(从一到五的数字)，顺序是不同的，且因此这两个阵列具有不同的状态。当集合的状态可由数个程序改变时，维持集合的状态、特别是顺序，提出了特定挑战。

[0004] 概述

[0005] 共享集合的概念可被以使得共享机制对于编程者而言是透明的方式被实现。另外，当集合由数个不同实体操作时，实现共享的集合的机制可，对于集合提供数据收敛。

[0006] 在一个示例中，共享的对象服务器，通过在任何给定时间维持集合的真实状态，来管理集合的共享。获得对于集合的访问的程序可连接至机器(程序在该机器上运行)上的共享对象运行时。访问共享集合的程序将该集合登记为共享集合，且这个登记由运行时通信至共享的对象服务器。想要访问该集合的其他程序也将该集合登记为共享集合。类似地，这个登记由运行时通信至共享的对象服务器。此外，程序可订阅在该集合上的改变提醒，其中订阅请求由运行时处理。每一个程序保持集合的本地副本。当集合的状态以任何形式被改变时，在其上发生改变的机器上的运行时将这个改变通信至共享的对象服务器。共享的对象服务器更新集合的真实状态，并将这个改变通信至通过共享对象运行时登记过这个集合的所有程序，该共享对象运行时位于在其上运行订阅程序的机器上。然后这些程序的每一个根据服务器所通信的状态改变，更新其本地集合副本。在该集合上已发生并发的改变的情况下，共享的对象服务器解决这些改变中的任何冲突，并确定性地到达集合的真实状态。

[0007] 在另一个示例中，以对等 (peer-to-peer) 方式、而不是客户机 / 服务器方式，来管理共享的对象。在对等实现中，对于共享的集合的每一个订阅者可将改变通信至其他订阅者。然后，订阅者在其上运行的机器上的运行时可改变它们的本地副本，同时确定性地解决任何冲突。

[0008] 几乎可以任何编程语言(如，C, C++, Java, Visual Basic 等)实现集合。由于共享集合的概念可以多种编程模型实现，以不同语言编写的程序可在同一个共享集合上操作。另外，编程者可编写程序，该程序将与在纯粹本地的集合上操作一样的方式在共享集合上操

作。因此，程序可能够操作共享的集合，而不必采用除了发出命令以登记新的集合或订阅已有集合的改变之外的任何方式，来识别集合的共享本质。以此方式，有效地使得集合的共享对于程序以及对于编程者而言是透明的。此外，共享机制对于集合中的数据的类型可以是不可知的，藉此允许分享任何类型数据的集合。

[0009] 提供本发明内容以便以简化形式介绍将在以下具体实施方式中进一步描述的一些概念。本发明内容并不旨在标识所要求保护主题的关键特征或必要特征，也不旨在用于限制所要求保护主题的范围。

[0010] 附图简述

[0011] 图 1 是共享集合的示例的框图。

[0012] 图 2 是示例性场景的框图，其中程序可共享对于共享集合的访问。

[0013] 图 3 是集合共享的对等实现的示例的框图。

[0014] 图 4 和 5 是共享的集合可被如何使用的一些示例的框图。

[0015] 图 6 是其中可共享集合的示例性过程的流程图。

[0016] 图 7 是可以结合本文描述的主题的实现来使用的示例组件的框图。

具体实施方式

[0017] 在现代计算中，通过共享对于同一底层数据的访问，程序经常彼此互操作。当程序共享数据时，经常以认知数据被共享的事实的方式实现程序。即，程序一般必须包含与其他程序、或与同一程序的其他实例共享数据的机制的实现。用于共享的机制经常专用于被共享的数据的本质、以及编程者的设计选择。因此，当两个程序想要共享数据时，它们各自的编程者一般必须协定、并实现特定的共享机制。存在特定的允许有限形式的数据共享的一般化机制，但是不适用于特定场合。

[0018] 尽管各种数据的共享提出了各种实现问题，当要共享的数据是集合时，提出了附加问题。集合是可被一起操作的一组数据项。集合的一些示例包括列表、阵列、集、包、和各种其他数据组。一些类型的集合被排序。因此，顺序集合的当前状态不是仅仅由集合的内容定义、还由其中这些内容所出现的顺序所定义。如，{1,2,3,4,5} 是包含从一到五的整数的数据集合。如果该集合被排序，则 {1,2,3,4,5} 是与 {2,3,1,5,4} 不同的状态，因为这两个状态具有不同的顺序，即使它们都包含相同的项。

[0019] 当在同时运行的程序中共享任何类型的数据时，数据的两个修改可能冲突 - 如，一个程序可想要修改数据，同时，另一个程序正想删除该数据。然而，当这样的冲突发生且正被共享的数据是集合的一部分时，发生附加难题。一个程序可请求在集合中的第二项后插入项。另一个程序可请求删除第二项，藉此使得被插入的项成为第二项。为了处理这些改变并达到集合的真实状态，必须决定原始集合中的哪一项现在是第二个。由于每一个程序可能不了解另一个程序正在对同一个集合操作，两个程序可能涉及集合中的“第二”项，同时对于集合中的哪个项真正是顺序的第二项具有不同的理解。

[0020] 运算转换字段涉及允许对于计算中的数据进行决定性排序的机制。可使用一些这样的机制来解决同时被操作的文件中的冲突—如，如何确定在正同时被两个用户编辑的文本文件中已经发生了什么编辑。然而，这些类型的技术一般没有被应用于此处描述类型的集合的一般化共享。

[0021] 此处描述的主题提供了共享数据集合的机制。此处描述的技术可被用于提供一般化的集合共享机制，其允许数个程序和 / 或同一程序的数个实例来共享对于集合的访问。这些技术可允许采用不同的编程语言实现程序。这些技术可允许被排序的数据集合以这样的方式同时被多个程序和 / 或程序实例操作：该集合决定性地达到特定状态，该状态可被传播至使用该集合的所有程序。另外，此处描述的技术可允许编程者编写程序，该程序以或多或少与程序将在纯粹本地集合上操作的方式一样的方式在共享集合上操作。此外，可使得共享机制对于集合中的底层数据类型是不可知的，藉此使得对于任何合适的数据种类使用相同的一般化的共享机制。换言之，可在无关于集合中的数据项内容的情况下、以及无关于这些项的内容的结构的情况下，执行此处描述的各种动作。

[0022] 共享数据的程序可在单个机器上运行、或可在通过网络连接的数个机器上运行。为了帮助数据的共享，在其上运行程序的每一个机器可具有共享的对象的运行时。共享的对象的运行时可有助于管理共享的数据对象（包括共享的集合）的共享和同步。在一个示例中，共享的对象的服务器维持共享的集合的真实状态。当对于机器上的共享的集合做出改变时，机器上的运行时提醒共享的对象服务器有关于这个改变。响应于该本地改变，服务器可改变集合的真实状态。如果两个程序对于集合做出并发的改变，共享的对象服务器可解决这些改变从而达到真实状态，此举在可能的程度上可反映两个程序意在实现的集合的状态。因此，如果两个不同程序各自在不了解另一个的改变的情况下改变同一个集合，服务器可执行这些改变之一来创建集合的新状态，且然后可转换另一个改变（在可能的程度上）以反映：如果另一个程序在其做出改变时已经知道了集合的新状态，该另一个程序应会作出的改变。服务器通知各机器上的运行时关于该改变（或经转换的改变，如果适用的话），且已经订阅该集合的程序根据服务器所传播的改变来更新它们的集合的本地副本。在另一个示例中，以对等体系结构实现集合共享，在对等体系结构中，订阅共享集合的每一个程序通知其他订阅者有关于该程序已经做出的改变，且其他订阅者（或者在正运行这些订阅者的机器上的运行时组件）在不需要服务器帮助的情况下将它们的本地副本同步至这些改变。

[0023] 当程序创建集合时，程序可发出指令将该集合登记为共享集合。这个指令可由在其上运行该程序的机器上的运行时接收，且该运行时可将该登记报告给共享对象服务器。然后，共享对象服务器接收集合的副本，其代表了集合的初始状态。从这个角度而言，创建该集合的程序可以与集合是纯粹地本地时采用的非常相似的方式在该集合上操作，因为运行时管理将改变报告至共享对象服务器（或在对等实现中，报告至其他订阅者）、以及从共享的对象服务器接收状态改变的同步。

[0024] 如果程序想要使用已经由另一个程序创建的共享的集合，该程序发出指令来登记该集合。这个指令可由运行时接收并被传送至共享的对象服务器。然后，共享的对象服务器知道来通知在订阅程序在其上运行的机器上的运行时有关该集合状态的任何改变。当程序订阅集合时，其可接受集合的当前真实状态的副本，这副本然后存储为本地集合。然后订阅程序可用与集合是本地的非常类似方式在共享集合上操作。以此方式，使得集合的有效共享相对于创建和登记集合的程序、以及对于订阅该集合的程序而言都是透明的。如上所述，系统可对于集合中的数据的特性是不可知的，从而共享机制可作用于包含任何类型底层数据的集合。

[0025] 图 1 示出共享集合 102 的示例。共享集合 102 可包括一组数据项，其以某种方

式彼此相关，且其可由多个程序（诸如程序 104、106、和 108）访问（如，读取和写入）。程序 104-108 可以是相同程序（如，在数个不同过程上运行的相同应用）的不同运行实例，或可以是不同程序。另外，程序 104-108 可全部在同一个机器上或在不同机器上运行。在两种情况下，程序 104-108 可以是不同程序、或同一程序的分开的运行实例，在此意义上，它们可被描述为“不同的”。集合共享机制 110 可帮助共享集合 102 在程序 104-108 之间的共享。下文结合图 2-7 描述集合共享机制 110 的示例性实现。

[0026] 共享集合 102 可具有名称 112，允许共享集合 102 由共享相同命名空间的程序所标识。作为名称 112 的替代物（或除了名称 112 外），共享集合 102 还可具有标识符 114，将共享集合 102 与其他共享集合区分开来。例如，标识符 114 可被应用于还没有被指派名称的集合，从而在“友好”的名称已经被指派给集合前，可标识这些集合。

[0027] 共享集合 102 可具有多个数据项。在图 1 的示例中，示出了四个数据项 116、118、120、和 122，但是集合可具有任意数量的数据项。每一个数据项可包含任意类型的数据。下文，结合图 4 和 5 将更为具体地描述可被存储在共享集合中的数据的一些示例。然而，作为一般示例，可被存储于集合中的一些数据类型是诸如数字或单词之类的元素数据类型、或更为复杂的由程序员定义的数据类型，诸如代表待做列表上的任务的结构、或用户界面的元素。此处描述的机制可对于数据类型是不可知的，从而集合中的独立的数据项可以是（详细示意为）从元素数据类型到复杂的用户定义的数据类型的任何项。

[0028] 在图 1 的示例中，共享集合 102 是排序的集合。如上所述，排序的集合是一种集合，其中集合的状态不仅由集合中的数据项所定义、还由这些数据项上的顺序所定义。数据项的插入或移除改变了排序的集合的状态，不过也改变了两个现有数据项之间位置。因此，排序的集合 102 中的每一个数据项具有顺序位置。顺序位置 124、126、128、和 130 各自对应于数据项 116、118、120、和 122。因此，数据项 116 处在第一位置、数据项 118 处在第二位置，以此类推。如果要改变任一项的顺序号，即使对于项本身没有改变，这个改变将导致共享集合 102 的状态的改变。

[0029] 可在共享集合 102 上执行的示例性操作包括转换操作 132 和删除操作 134。转换操作 132 交换集合中两个（或更多个）项的顺序位置，且删除操作 134 移除集合中的其中一项。还有，可通过执行插入操作 136 来向集合增加新的数据项 138，以修改集合。下文的讨论中，我们稍后将涉及这些操作。

[0030] 图 2 示出其中程序可共享对于共享集合 102 的访问的示例性场景。机器 202 是在其上执行程序的机器。机器 202 可以是具有某种计算能力的任何类型的机器 - 如，诸如台式计算机或膝上型计算机之类的个人计算机、服务器计算机、手持式计算机、智能电话、机顶盒等。在图示示例中，两个程序 204 和 206 在机器 202 上执行，尽管任何数量的程序可在机器上运行。

[0031] 机器 202 可存储数据，且被存储在机器 202 上的数据的一个类型是共享集合 102 的本地副本。在图 2 的示例中，共享集合 102 由程序 204 创建。在集合被创建时，集合可以是本地的且专用于程序 204。然而，程序 204 可能想要将该集合登记为用于共享的集合，以允许其他程序（可能包括，在其他机器上运行的程序）共享对于集合 102 的访问。因此，程序 204 发出登记指令 208。登记指令 208 可通过编写程序 204 的任何编程语言发出，且可由共享对象运行时 210 所接收。例如，共享对象运行时 210 可提供应用程序编程接口（API），允

许程序 204 向共享对象运行时 210 发出登记指令(和订阅请求、和其他类型的指令)。

[0032] 一旦接收登记指令 208,共享对象运行时 210 可向共享对象服务器 212 登记共享集合 102。共享对象服务器 212 是维持共享数据对象(诸如共享集合 102)的真实状态的服务器,且还用作共享对象的改变的交换场所。即,当共享数据在机器上改变时,共享对象服务器 212 接收通知,并将对共享数据的状态的改变传播至其他机器。当共享集合 102 在共享对象服务器 212 上登记时,共享对象运行时 210 可提供共享集合 102 的副本。一旦集合 102 已经被登记为共享,共享的对象服务器 212 维持集合 102 的真实状态,如在图 2 的代表共享对象服务器 212 的框中的标记为“(真实状态)”的框中所示那样。(要注意的是,贯穿此处描述,我们将程序或订阅者发送动作称为向服务器或其他订阅者发送登记或订阅请求。程序发送这些类型的请求的动作包括其中程序将请求通信至运行时、且运行时将该请求通信至服务器、或其他订阅者或运行时的情况。类似地,如下文所述地,订阅者可将状态改变通信至服务器或其他订阅者 ;订阅者通信状态改变的动作包括其中在订阅者机器上的运行时执行通信的情况。)

[0033] 当共享集合 102 被登记为共享时,除了程序 204 外的其他程序可通过订阅该集合来访问该集合。为了订阅集合,该程序可提交订阅请求 214。一些集合可经受它们的创建者所指定的访问控制限制。这样的访问控制限制可限制可访问共享集合的一组程序和 / 或用户和 / 或机器。然而,假设在共享集合 102 上没有访问限制(或存在访问限制,但是寻求对于共享集合 102 的访问的那些限制具有访问该集合的允许),然后订阅共享集合 102 允许订阅者读取和 / 或修改该集合。

[0034] 订阅请求可来自任何机器上的任何程序。例如,通过经共享对象运行时 210 提交订阅请求,程序 206 可订阅共享集合 102。值得注意的是,程序 206 与共享集合 102 的创建者(即,程序 204)在同一机器 202 上。然而,订阅程序可在不同机器上。例如,机器 218 上的程序 216 也通过提交订阅请求 214 来订阅共享集合 102。来自程序 216 的订阅请求通过共享对象运行时 220 (机器 218 的运行时)提交。(每一个机器可具有其自己的共享对象运行时的实例。)另外,共享集合 102 的订阅请求可来自在机器 222、或在任何其他机器上执行的程序。

[0035] 当程序已经订阅共享集合 102 时,该程序可接收共享集合 102 的当前真实状态的副本。例如,机器 218 上的程序 216 接收共享集合 102 的副本,其可将副本存储在机器 218 上(如在代表机器 218 的图 1 的框中标记“(本地副本)”的共享集合 102 的实例)。然后订阅程序可如本地副本是纯粹本地集合一样读取和写入本地副本。在其上运行有程序的机器上的共享对象运行时的实例(如,在程序 216 的情况下,是共享对象运行时 220)可监测共享集合 102 的本地副本来确定程序是否已经对于共享集合做出将被报告至共享对象服务器 212 的改变。

[0036] 当对于共享集合 102 的任何订阅者执行对共享集合 102 的本地副本做出改变的操作,订阅者机器上的共享对象运行时将该操作报告至共享对象服务器 212。然后,共享对象服务器 212 更新共享集合的当前状态,并提醒对象订阅者关于(包括对象的创建者)状态改变。

[0037] 共享对象服务器 212 可包含同步组件 226,其帮助在共享集合 102 的订阅者之间的状态信息的收集和传播。同步组件 226 的操作方式可以是依赖于实现的。在一个示例中,同

步组件维持处于其真实状态中的共享集合 102 的主副本，且无论何时在这个集合的状态变化时，仅传播共享集合 102 的新副本。在另一个示例中，同步组件 226 维持相对于当前版本的共享集合 102 的改变列表，并将这些改变传播至各客户机。然后，每一个客户机上的运行时能基于共享集合 102 的本地状态和从共享对象服务器 212 接收到的改变来计算共享集合 102 的真实状态。即使共享对象服务器 212 没有响应于每一次状态改变传播共享集合 102 的完整副本，共享对象服务器 212 可确定如何解决对于共享集合 102 的同时改变，从而－当客户机侧的运行时接收改变的通知时－其可从旧的状态计算共享集合 102 的新的状态。如下文所述，需要时，同步组件 226 可转换变化，从而在不了解第二订阅者的同步改变的情况下，第一订阅者做出的改变，可被发送至第一订阅者，且可被以有意义的，且导致所有订阅者的结果收敛在共享集合 102 的相同状态上的方式来应用。

[0038] 同步组件 226 所执行的一个功能可以是确定如何由两个客户机将所做出的改变应用于集合，其中没有一个客户机了解另一个的改变。如果两个客户机在大约同时做出涉及集合中同一项的改变，从而每一个客户机在接收另一个该改变的通知之前做出改变，这个情况可能发生。例如，假设集合中的最先两项是阿尔法(alpha)和贝塔(beta)。两个客户机—称它们为 A 和 B—在大约同时对于集合做出改变。A 想要在阿尔法后的位置添加新项，伽马(gamma)，且 B 想要删除贝塔。所以 A 执行在顺序位置二处增加新项的操作，且 B 执行在顺序位置二删除项的操作。如果首先执行 A 的操作，则两个操作后的最终状态是阿尔法和贝塔分别占据位置一和二。如果首先执行 B 的操作，则两个操作后的最终状态是阿尔法和伽马分别占据位置一和二。同步组件 226 可以如下方式解决这个明显的冲突。

[0039] 为了这个示例的目的，调用服务器 S 和客户机 A 和 B。S、A、和 B 全部在状态 0 开始 A 执行操作 X 且 B 执行操作 Y。由于 A 或 B 均不了解另一个操作已经被执行，A 和 B 各自以状态 0 向 S 报告－即，A 和 B 各自报告，当集合在状态 0 中存在时，它们已经在集合上执行了操作。已经执行了一个操作，现在 A 和 B 各自将它们的状态增至 1。这些操作的一个，然而，将首先到达 S。为这个示例，假设，X 首先到达。由于操作 X 在状态 0 (这与 S 的当前状态匹配)操作，其在没有被转换的情况下在 S 上执行并发送至 B。此外，确认被传送至 A。S 现在处于状态 1。此时，B 从 S 接收操作 X。由于操作 Y 还没有被 S 所确认，操作 X 被相对于 Y 进行转换，然后应用至集合。B 现在处于状态 2。当操作 Y 到达 S 时，该操作表示它在状态 0 被执行。由于 0 不再是 S 的当前状态，S 确定在 B 执行 Y 时不了解操作 X。因此，操作 Y 被相对操作 X 而转换然后传送至 A。确认被传送至 B。S 现在处于状态 2。此时，A 接收操作 Y 的经转换的版本。由于 A 不具有代办操作，在不将经转换的 Y 进一步转换的情况下应用经转换的 Y。(即，经转换的 Y 表示其相对于状态 1 而被执行。由于 A 已经在状态 1，没有进一步转换 Y 的基础。)A 现在处于状态 2。此外，在 A 和 B 的集合的副本现在以相同顺序显示相同数据。

[0040] 图 2 示出示例性客户机／服务器－类型实现，其中服务器管理集合的共享。然而，还可使用对等实现。图 3 示出集合共享的对等实现的示例。

[0041] 在图 3 的示例中，机器 202、218、和 222 参与集合 102 的共享(就像它们在图 2 中所做的那样)。例如，每一个机器可具有作为共享集合 102 的订阅者的一个或多个程序(其中共享集合的创建者被包括在“订阅者”的概念中)。每一个机器具有共享对象运行时的实例—即，共享对象运行时 210、220、和 302 分别在机器 202、218、和 222 上运行。在对等实现

中，共享对象运行时可直接彼此通信，而不是与服务器通信。另外，共享的对象运行时可负责转换传入改变，而不是由服务器执行那些改变。因此，只要不同运行时实现用于解决对于集合的冲突改变的相同算法，集合的当前状态可被确定性地了解。

[0042] 如果共享集合 102 例如在机器 202 上被改变，则机器 202 上的共享对象运行时 202 将状态改变 304 的通知传送至机器 218 和 222 上的共享对象运行时 220 和 302。每一个共享对象运行时将状态改变 304 应用于共享集合 102 的当前状态，从而到达集合 102 的新的状态。在两个不同状态改变同时来自两个不同机器的情况下，共享对象运行时可应用冲突解决算法(如，上述结合同步组件 226 描述的算法)来确定要应用什么状态变化。如上所述，可使得该算法为确定性的，从而接收同一组冲突改变的任何两个运行时可以同样的方式解决冲突，藉此使得集合在各不同的机器上被改变至相同状态。

[0043] 图 4 和 5 示出共享集合可如何被使用的一些示例。图 4 和 5 中的示例是示意性的，且并不限制此处的主题。其被提供仅用于给出共享集合可被如何使用的一些特定示例，但是可理解的是存在其中可使用共享集合的众多其他上下文。

[0044] 图 4 示出示例性任务列表，其可被实现为共享集合。任务列表中的每一个任务可以是将被执行的任务。所示出的任务列表包括四个示例性项，402、404、406、和 408，且可以是将由为公司开发新产品的工作组所执行的任务。任务列表可被实现为集合，其中将被执行的每一个任务是集合中的项，且其中这些项所定义的顺序代表了其中这些项将被执行的顺序。工作组的成员可全部对于任务列表具有控制。例如，可通过 e-mail 和日历程序访问该任务列表，其中每一个组员正在运行实例。因此，不同的组员可在列表上添加或删除项，或可重新排列现有项。因此，程序的每一个实例可以是代表任务列表的集合的订阅者。当任何组员使用他或她的程序实例来对任务列表做出改变时，该改变可被以上述方式传播至程序的其他实例。所以，工作人员可使用程序的第一实例来转换第一和第二任务的顺序(如数字 410 所指示的)、且另一个工作人员可使用程序的第二实例来删除第三任务(如数字 412 所指示的)。这些改变可被处理，且可到达集合的新的状态。

[0045] 图 5 示出示例性用户界面，其可被实现为集合。用户界面可例如，是桌面工具条，其中图 5 中所示的各数据项(即，数据项 502、504、506、508、和 510) 是以对应于项显示在集合中的顺序以某种垂直顺序图是在工具条中的窗口小部件。窗口小部件的列表可被实现为集合，且不同程序可共享对于集合的访问。例如，用户可确定他想要使得时钟位于新的反馈的顶部，且因此可转换集合中最先两个元素的顺序(如数字 512 所示)。然后，用户可将工具条中的错误报告至信息技术管理器。然后，管理器可使用管理程序来连接至工具条。管理器可识别，比如，照片查看程序具有错误，且因此可从工具条中删除该窗口小部件(如数字 514 所示)。以此方式，两个人使用两个不同应用来修改被实现为集合的用户界面。

[0046] 要注意的是，在图 4 和 5 的示例中，图示的集合可涉及复杂的数据类型。例如，图 4 中的数据项是任务(可包含用于概述、详细描述、截止日期、提醒日期等的字段)。且图 5 中的数据项可以是应用处理(其可包含字段，诸如应用的名称、在磁盘上位置、运行时参数，等)。换言之，这些示例示出可被用在共享集合中的任何类型的数据，且集合共享机制可以相同方式处理各种类型数据，而不管数据的特性或结构。

[0047] 图 6 示出其中可共享集合的示例性处理。在转向图 6 的描述之前，注意，参考图 1 － 5 中所示的组件作为示例地描述了包含在图 6 中的流程图，尽管图 6 的过程可以在任何

系统中实现，并且不限于图 1-5 中示出的场景。另外地，图 6 中的流程图示出了其中按特定顺序来实现过程的各阶段的一示例，如连接各框的线所示，但这个示图中示出的各种阶段可以按任何顺序、或以任何组合或子组合来执行。

[0048] 在602，创建集合。例如，程序可创建集合，可通过使用通常将被用于创建本地集合的机制来这样做。该集合可以是任意类型的集合—如，列表、阵列等。在604，集合可被登记为用于共享。例如，创建集合的程序可发出指令，这可由在其上运行程序的机器上的共享对象运行时所接收。在一个示例中，运行时展示了API，其允许程序发出登记指令，尽管可以其他方式发出登记指令。

[0049] 在集合被登记为共享后，可接收到对于集合的订阅请求(在 606)。例如，与创建集合的程序共享同一命名空间的其他程序可对于该集合发出订阅请求。这些订阅请求可被向在其上运行有订阅程序的机器上的共享对象运行时发出。在一个示例中，可通过运行时所展示的 API 发出订阅请求。为了此处的主题的目的，创建者被认为是订阅者之一。(即使创建者发出登记请求而不是订阅请求，从订阅者是可访问该集合的那些程序的意义上而言，共享对象的创建者是订阅者。)订阅者可接收集合的当前状态(在 608)，且它们可存储集合的本地副本。

[0050] 在某一点，订阅者对于集合做出改变(在 610)。示例性改变包括插入 652、删除 654、和移动 656。插入 652 在集合顺序中特定位置处向集合添加新的项。删除 654 从集合中移除现有的项。移动 656 重新排序集合中两个或更多个项的顺序。

[0051] 当订阅者机器上的共享对象运行时检测到改变时，共享对象运行时通知可适用的实体(多个)有关该改变(在 612)，且这些变化由这些实体(多个)所接收(在 614)。哪些实体是“可适用的实体”可取决于实现。如上所述，对于此处描述的主题的实现，存在例如，客户机 / 服务器和对等实现。在客户机 / 服务器实现中，检测改变的运行时可通知管理集合的真实状态的共享对象服务器。在对等实现中，每一个运行时直接通知其他运行时。

[0052] 当接收到对于集合的改变时，以确定性方式解决(如，表示两个客户机已经在集合的同一状态执行操作的改变)任何冲突改变(在 616)。例如，如果两个订阅者已经做出了影响集合中“第二”项的改变(如，一个实体转换第一和第二实体的位置，而另一个实体，同时地，删除了第二实体)，这些冲突的改变可被解决来达到集合的真实状态 - 如，使用上述结合同步组件 226 (图示于图 2 中)描述的算法。解决冲突的方式可能会也可能不会实现做出改变的订阅者的意图，但是可明白地确定所得状态。

[0053] 在客户机 / 服务器实现中，所解决的变化可被传送至订阅者(在 618)。在对等实现中，订阅者和 / 或在订阅者机器上的运行时从其他订阅者和 / 或运行时处接收改变信息，并在没有服务器帮助的情况下，自己解决冲突。确定性冲突解决处理允许每一个客户机以同样方式解决改变，从而各订阅者可收敛于集合的真实状态的相同结果上。(同样地，可明白地解决所得状态，但是所得状态可能会也可能不会实现做出改变的订阅者的意图。)

[0054] 当任何所存在的冲突已经被解决时(且在客户机 / 服务器实现的情况下，是当冲突解决的结果已经从服务器被通信至订阅者时)，可作出对于集合的本地副本的改变(在 620)来使得这些本地副本与真实状态一致。然后订阅者继续读取和写入集合。

[0055] 图 7 示出其中可部署此处所述的本发明的各方面的示例环境。

[0056] 计算机 700 包括一个或多个处理器 702 和一个或多个数据记忆组件 704。(诸)处

理器 702 通常是微处理器，如在个人台式计算机或膝上型计算机、服务器、手持式计算机或另一类型的计算设备中含有的那些微处理器。(诸)数据记忆组件 704 是能够短期或长期存储数据的组件。(诸)数据记忆组件 704 的示例包括硬盘、可移动盘(包括光盘和磁盘)、易失性和非易失性随机存取存储器(RAM)、只读存储器(ROM)、闪存、磁带等等。(诸)数据记忆组件是计算机可读存储介质的示例。计算机 700 可以包括显示器 712 或与其关联，显示器 912 可以是阴极射线管(CRT)监视器、液晶显示(LCD)监视器或任何其他类型的监视器。

[0057] 软件可以被存储在(诸)数据记忆组件 704 中，而且可以在一个或多个处理器(多个)702 上运行。此类软件的一个示例是可以实现以上结合图 1-6 描述的一些或所有功能的集合共享软件 706，但可以使用任何类型的软件。可以例如通过一个或多个组件来实现软件 706，这些组件可以是分布式系统中的组件、分开的文件、分开的函数、分开的对象、分开的代码行等等。其中程序被存储在硬盘、被加载到 RAM 中并被(诸)计算机处理器上执行的计算机(例如，个人计算机、服务器计算机、手持式计算机等等)代表图 7 中所描绘的场景，但在此所描述的主题不限于这一示例。

[0058] 在此所描述的主题可以被实现为被存储在数据记忆组件 704 中的一个或多个内并在一个或多个处理器 702 上执行的软件。作为另一示例，本主题可以被实现为存储在一个或多个计算机可读存储介质上的指令。诸如光盘或磁盘等有形介质是存储介质的示例。指令可以存在于非暂态介质上。此类指令在由计算机或其他机器执行时可以使得计算机或其他机器执行一种方法的一个或多个动作。执行动作的指令可以被存储在一个介质上，或可以开跨多个介质来分布，以使得这些指令共同出现在一个或多个计算机可读存储介质上而无论所有指令是否恰好是在同一介质上。

[0059] 此外，作为方法的一部分，在此所描述的任何动作(无论是否在图中示出)可以由处理器(例如，一个或多个处理器 702)执行。因此，如果在此描述动作 A、B 和 C，则可以执行一种包括动作 A、B 和 C 的方法。此外，如果在此描述动作 A、B 和 C，则，可以执行一种包括使用处理器执行动作 A、B 和 C 的方法。

[0060] 在一个示例环境中，计算机 700 可以通过网络 708 在通信上连接到一个或多个其他设备。结构上可以类似于计算机 700 的计算机 710 是可以被连接到计算机 700 的设备示例，但其他类型的设备也可以这样连接。

[0061] 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具体特征或动作。更确切而言，上述具体特征和动作是作为实现权利要求的示例形式公开的。

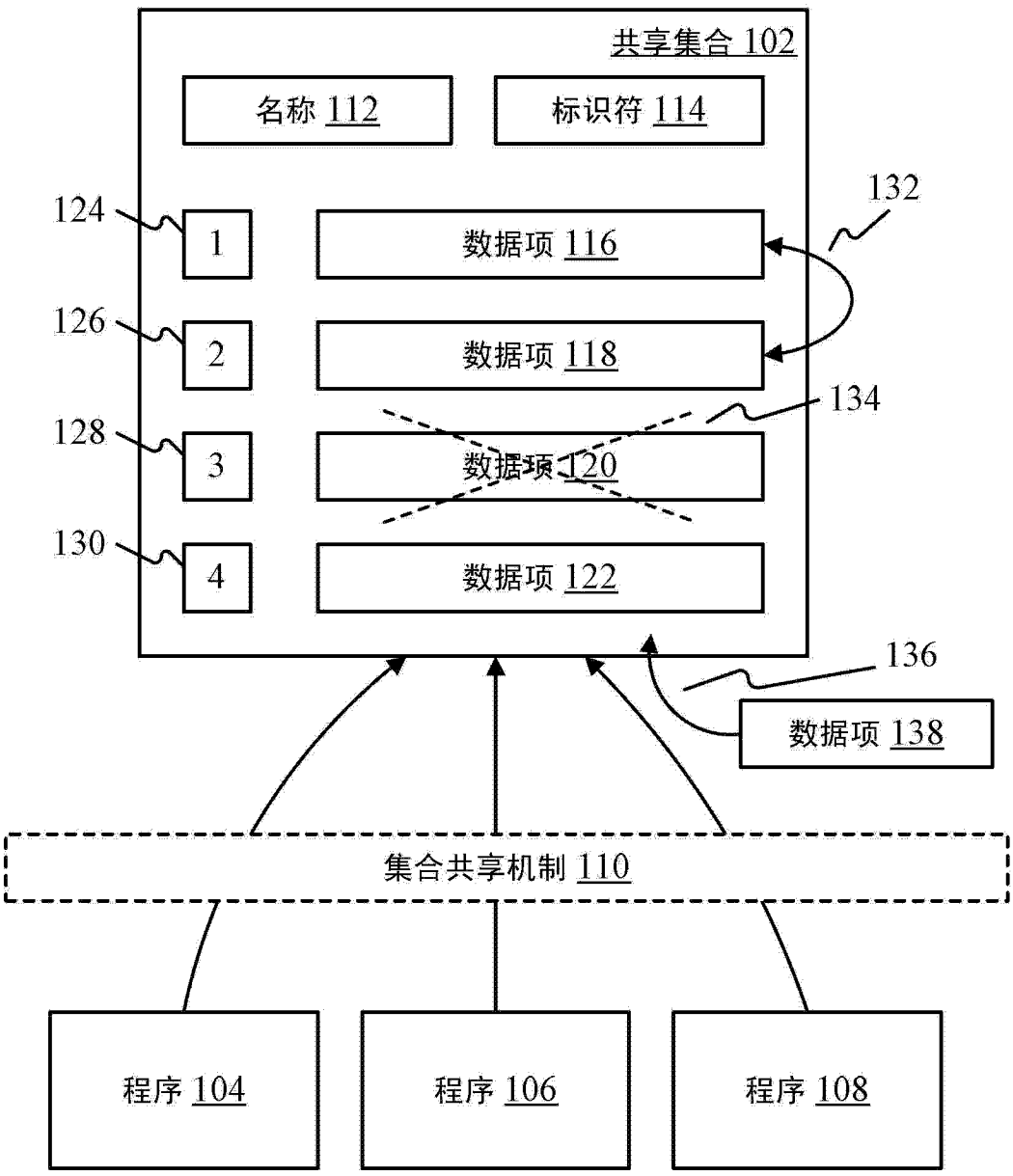

共享集合 102
名称 112
标识符 114
124
1
数据项 116
132
126
2
数据项 118
134
128
3
数据项 120
130
4
数据项 122
136
数据项 138
集合共享机制 110
程序 104
程序 106
程序 108

图 1

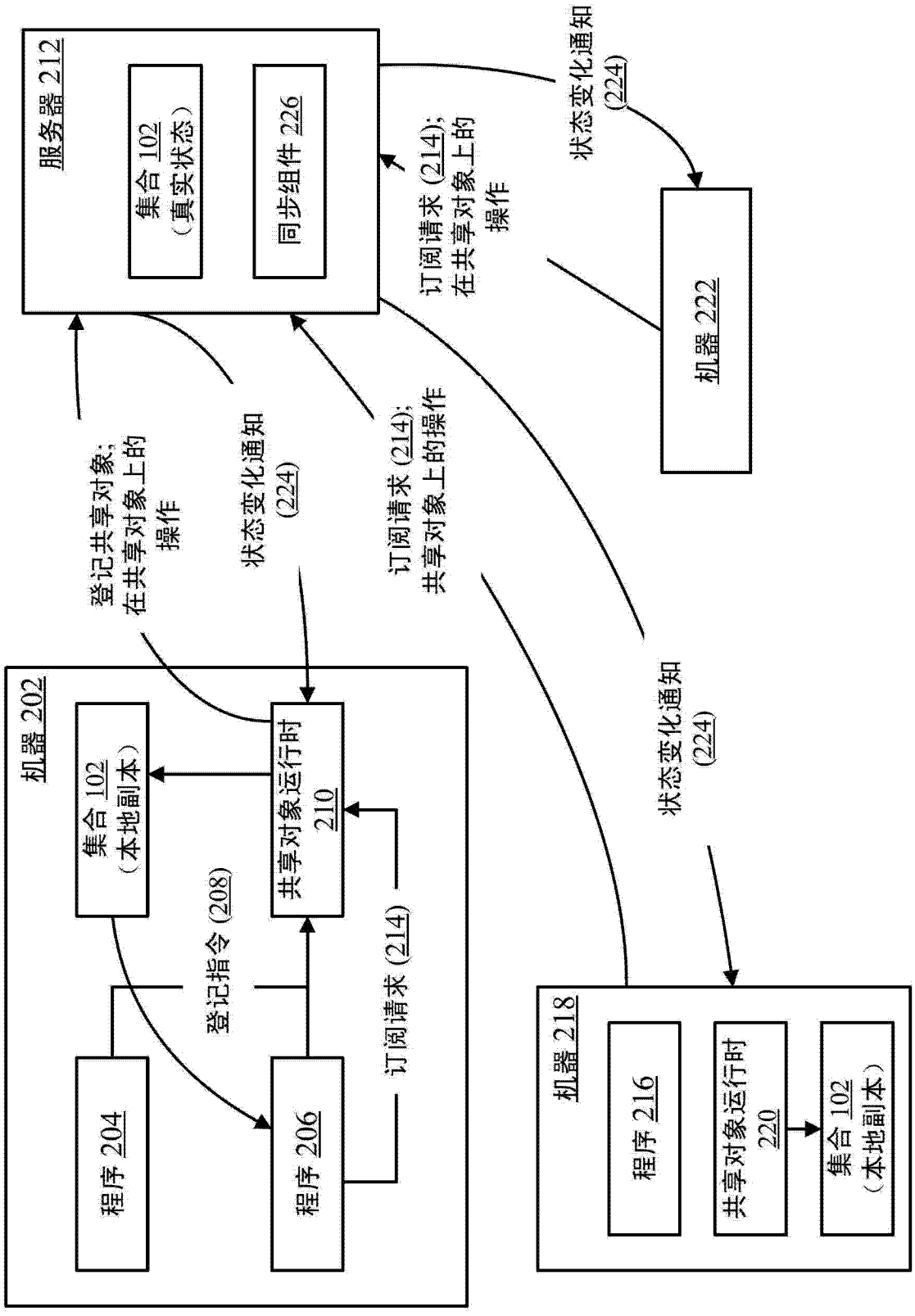

服务器 212
集合 102
（真实状态）
同步组件 226
状态变化通知
(224)
订阅请求 (214);
在共享对象上的
操作
机器 222
登记共享对象;
在共享对象上的
操作
状态变化通知
(224)
订阅请求 (214);
共享对象上的操作
状态变化通知
(224)
机器 202
集合 102
（本地副本）
共享对象运行时
210
登记指令 (208)
订阅请求 (214)
程序 204
程序 206
机器 218
程序 216
共享对象运行时
220
集合 102
（本地副本）

图 2

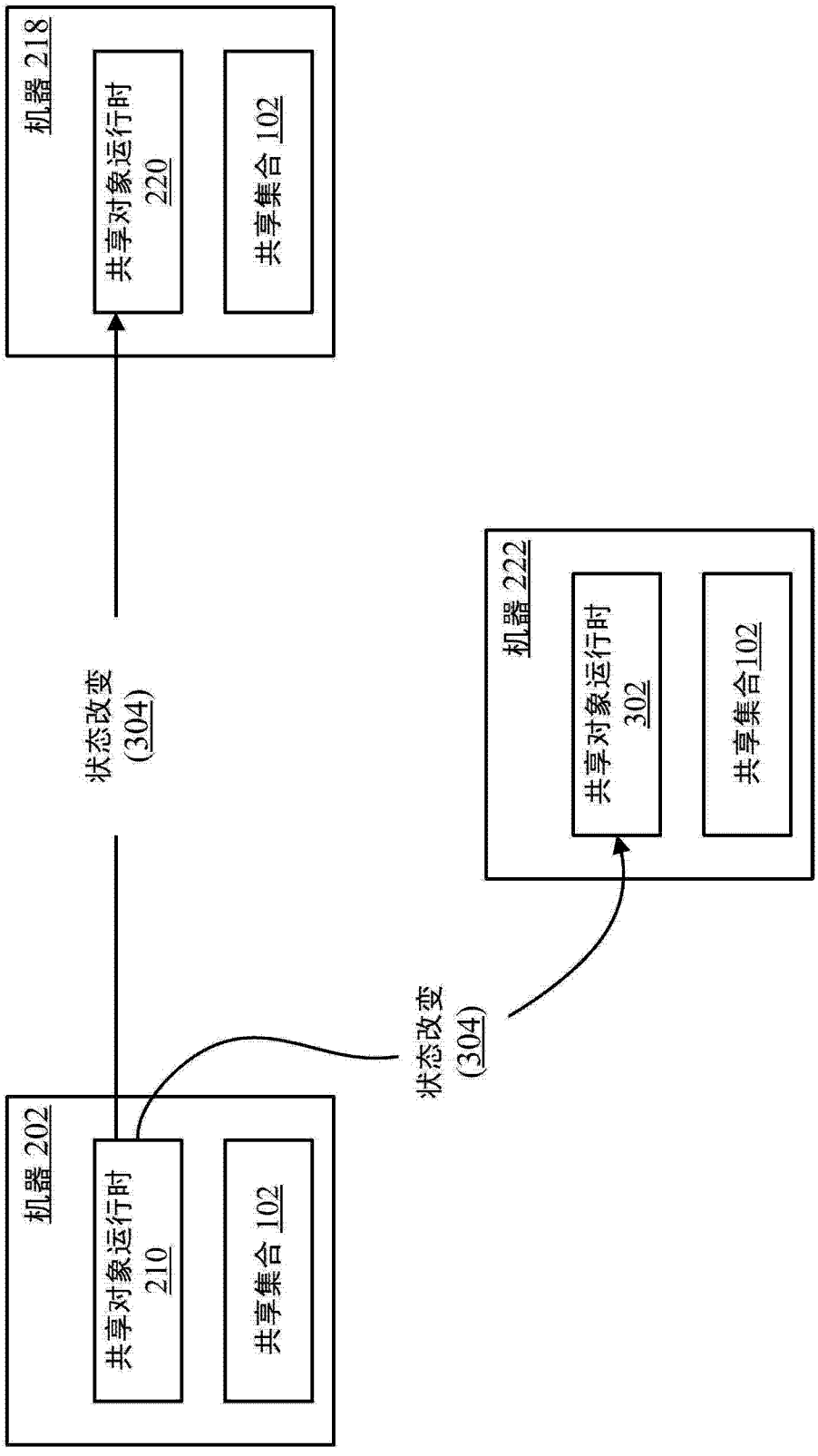

机器 218
共享对象运行时 220
共享集合 102
状态改变 (304)
机器 222
共享对象运行时 302
共享集合102
状态改变 (304)
机器 202
共享对象运行时 210
共享集合 102

图 3

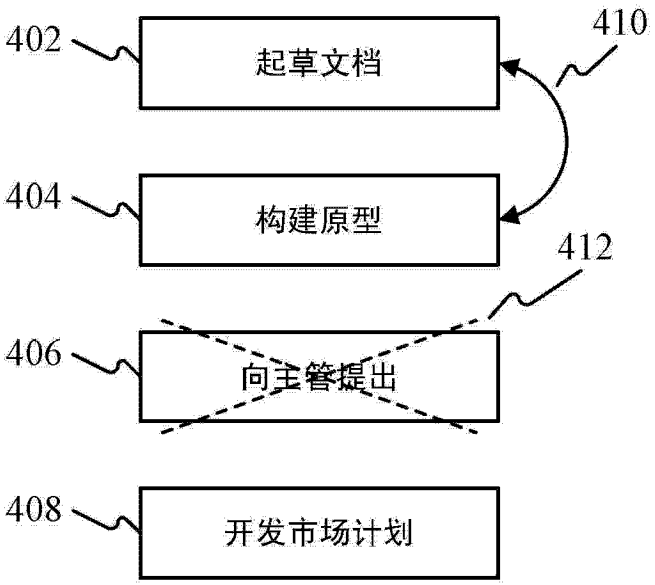

402
起草文档
410
404
构建原型
412
406
向主管提出
408
开发市场计划

图 4

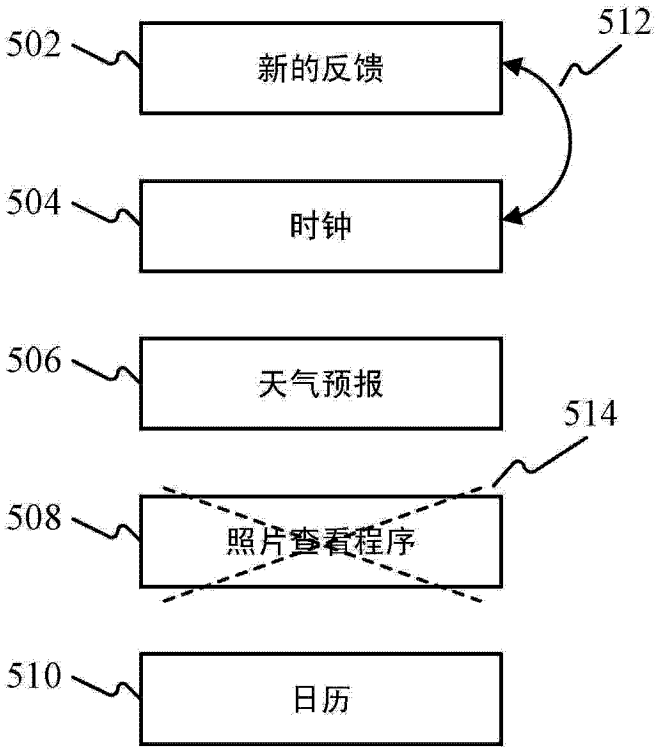

502
新的反馈
512
504
时钟
506
天气预报
514
508
照片查看程序
510
日历

图 5

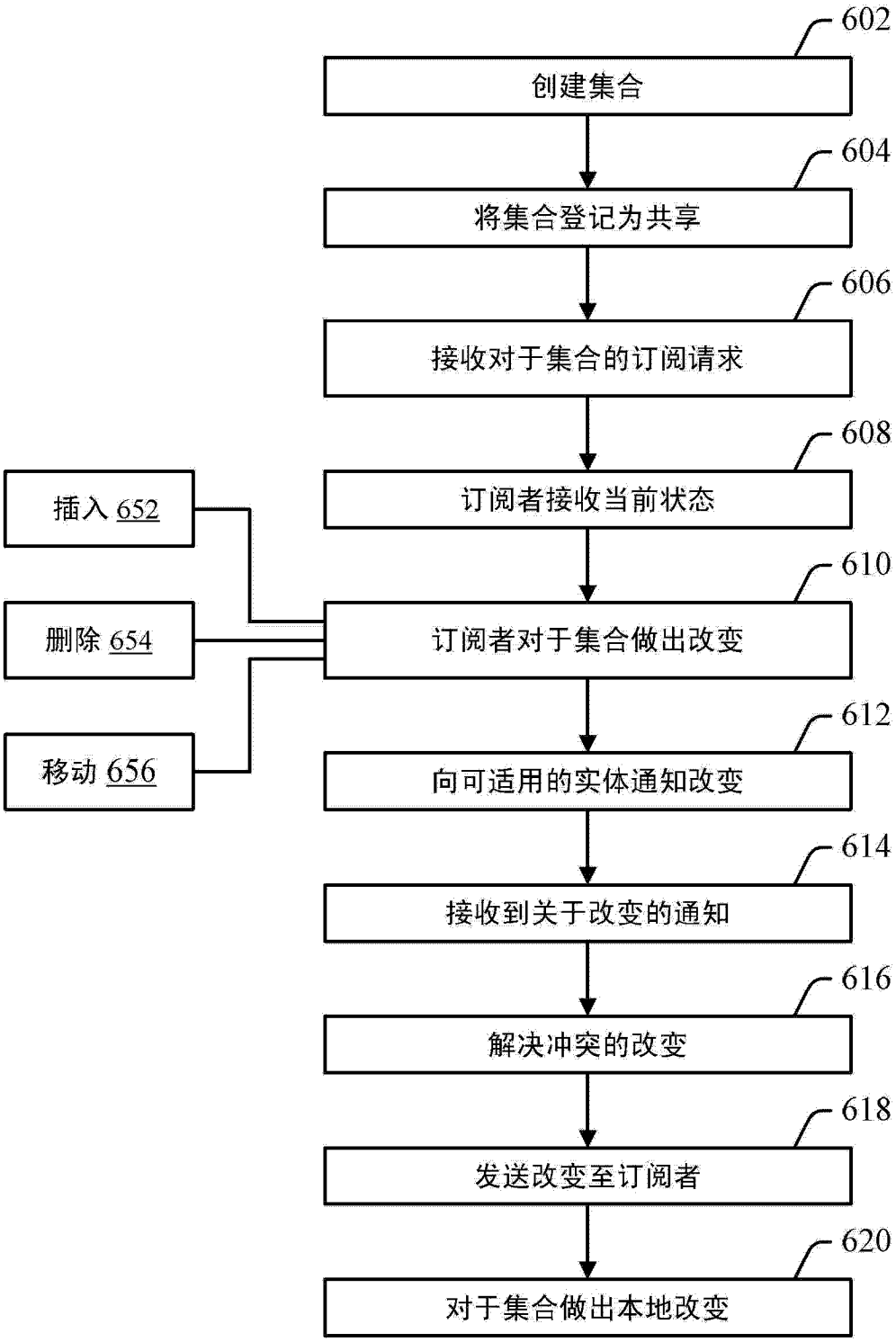

602
创建集合
604
将集合登记为共享
606
接收对于集合的订阅请求
608
订阅者接收当前状态
610
订阅者对于集合做出改变
612
向可适用的实体通知改变
614
接收到关于改变的通知
616
解决冲突的改变
618
发送改变至订阅者
620
对于集合做出本地改变
插入 652
删除 654
移动 656

图 6

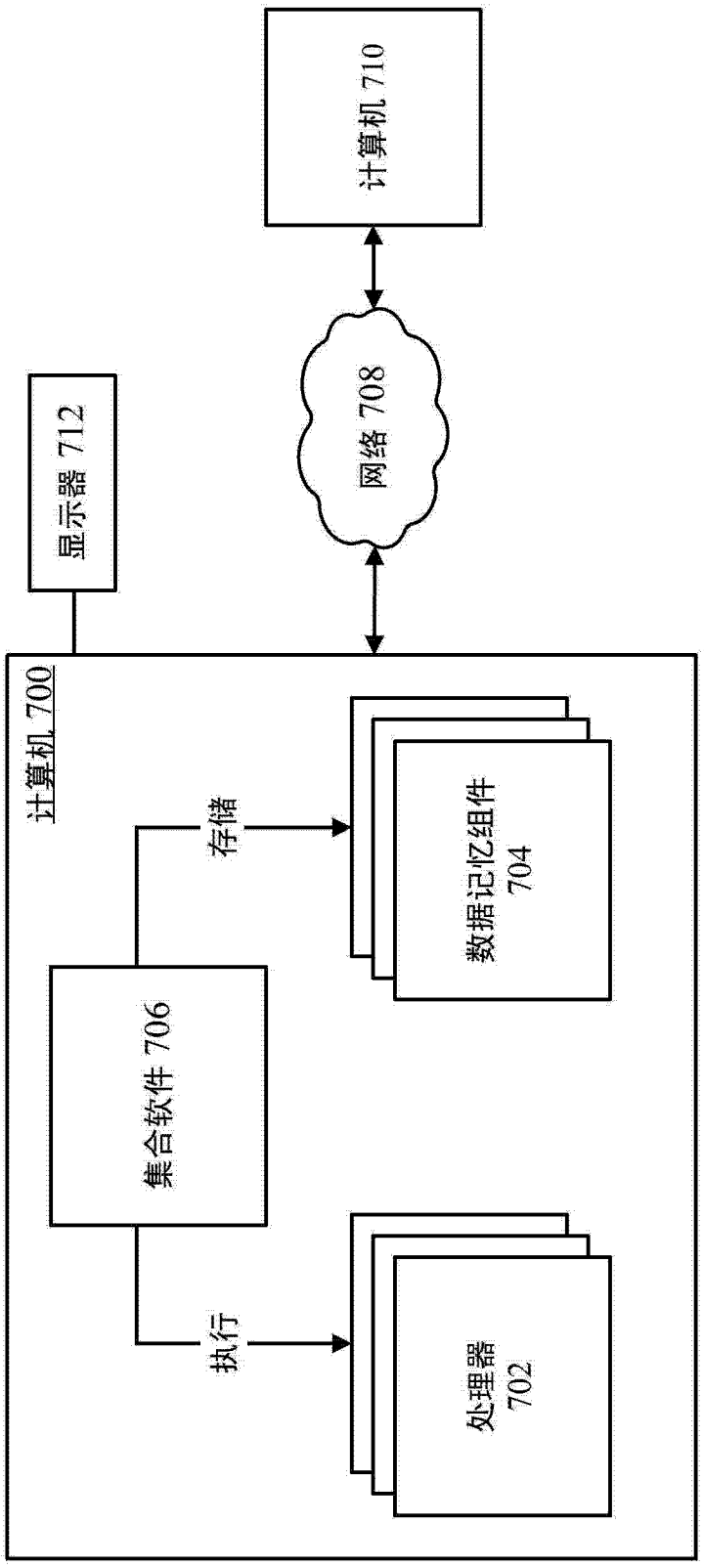


图 7