[19]中华人民共和国国家知识产权局

[51]Int. Cl[7]
G06F 15/00

[12]发明专利说明书

[21] ZL 专利号 96121333.7

[45]授权公告日 2002 年 12 月 18 日

[11]授权公告号 CN 1096642C

[22]申请日 1996.12.8 [21]申请号 96121333.7

[30]优先权

[32]1995.12.8 [33]US [31]569753

[32]1996.4.30 [33]US [31]640244

[73]专利权人 太阳微系统有限公司

地址 美国加利福尼亚州

[72]发明人 威廉·N·乔伊 阿瑟·A·冯霍夫

[56]参考文献

US4941083 1990. 7.10 G06F13/42

US5175837 1992.12.29 G06F13/18

US5202990 1993. 4.13 G06F13/00

审查员 刘春霞

[74]专利代理机构 北京市柳沈律师事务所

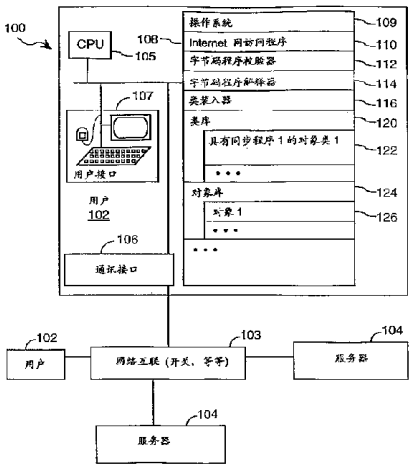
代理人 马 莹

权利要求书 4 页 说明书 14 页 附图 6 页

[54]发明名称 高效利用空间的对象加锁的系统和方法

[57]摘要

一种具有多个对象和过程存储器的多线程计算机系统，其中各对象具有锁定或未锁定的锁状态并含有对数据结构的数据指针。该系统分别采用全局对象和局部特定对象加锁过程来服务从未锁定与最近未锁定或锁定与最近锁定的对象的锁请求，全局过程的指令将未锁定对象的锁状态改变为锁定并为它调用局部过程，后者的指令将存储并更新它的锁数据。若它最近未被锁定，则系统的无用数据收集过程启动锁数据清除过程以释放此局部过程。



ISSN1008-4274

1、一种计算机系统，包括：

存储多个对象的存储器，每一存储的对象具有从锁定和非锁定组成的集合中选出的锁状态，每一个存储的对象包括指向数据结构的数据链路；

全局对象加锁装置，用于改变特定未锁定对象的锁状态为锁定，并为所述特定未锁定的对象产生局部对象加锁装置，每一个局部对象加锁装置包括用于存储锁数据的锁数据子阵列和用于更新特定未锁定对象的存储的锁数据的装置；和

锁数据清除装置，用于当最近还没有锁定所述特定对象时释放特定对象的局部对象加锁装置；

其中所述系统使用所述全局对象加锁装置来服务于不具有锁数据子阵列的对象的锁请求，使用所述局部对象加锁装置来服务于具有锁数据子阵列的对象的锁请求，和使用所述锁数据清除装置去释放特定对象的局部对象加锁装置。

2、如权利要求1的计算机系统，其中，

最近没有锁定的锁状态的每一存储的对象包括一至所述全局对象加锁装置的方法链路；

最近具有锁定的锁状态的每一存储的对象包括一至所述局部对象加锁装置的方法链路；和

所述全局对象加锁装置用于更新所述特定对象的方法链路以链接到所述局部对象加锁装置。

3、如权利要求1的计算机系统，其中，

从未具有锁定的锁状态的每一个存储的对象包括一至所述全局对象加锁装置的方法链路；

具有锁定的锁状态的每一个存储的对象包括一至所述局部对象加锁装置的方法链路；

所述全局对象加锁装置用于更新所述特定对象的方法链路以链接到所述局部对象加锁装置；和

所述锁数据清除装置用于改变所述特定对象的方法链路以链接到所述

全局对象加锁装置。

4、如权利要求 1 的计算机系统，其中

所述存储器还存储一组对象类，每一个对象类包括初始虚拟功能表(VFT)，该表包括至与所述对象类相关的一组方法的诸链路和至所述全局对象加锁装置的一链路；

最近没有锁定的锁状态的每一个所述存储的对象包括一至对应所述对象类中一个的所述初始 VFT 的方法链路；

用于具有锁定的锁状态的每一个所述存储的对象的局部虚拟功能表(VFT)包括至所述局部对象加锁装置的链路；

所述全局对象加锁装置用于更新所述特定对象的方法链路以链接到所述特定对象的所述局部 VFT。

5、如权利要求 1 的计算机系统，其中，

所述存储器还存储一组对象类，每一个对象类包括初始虚拟功能表(VFT)，该表包括至与所述对象类相关的一组方法的诸链路和至所述全局对象加锁装置的一链路；

从来没有锁定的锁状态的每一个所述存储的对象包括一至对应所述对象类中一个的所述初始 VFT 的方法链路；

用于具有锁定的锁状态的每一个所述存储的对象的局部虚拟功能表(VFT)包括至所述局部对象加锁装置的链路；

所述全局对象加锁装置用于更新所述特定对象的方法链路以链接到所述特定对象的所述局部 VFT；和

所述锁数据清除装置用于改变所述特定对象的方法链路以链接到对应于所述特定对象的对象类的所述初始 VFT。

6、一种操作计算机系统的方法，包括以下步骤：

在计算机存储器中存储多个对象，每一个存储的对象具有从锁定和未锁定组成的集合中选出的锁状态，每一个存储的对象包括至数据结构的数据链路；

当服务从未被锁定的特定对象的锁请求时，执行全局对象加锁过程，该全局对象加锁过程用于改变特定对象的锁状态为锁定及为特定对象产生局部对象加锁过程，每一个局部对象加锁过程包括用于存储锁数据的锁数据子阵列和用于更新所述特定对象的存储的锁数据的过程；和

当最近还没有锁定所述特定对象时，执行锁数据清除过程以释放特定对象的局部对象加锁过程。

7、如权利要求6的方法，其中，

与最近没有锁定的锁状态的每一个存储的对象相关的方法链路至所述全局对象加锁过程；

与最近具有锁定的锁状态的每一个存储的对象相关的方法链路至所述局部对象加锁过程；和

当执行全局对象加锁过程以服务于未锁定对象的锁请求时，更新所述特定对象的方法链路以链接到所述局部对象加锁过程。

8、如权利要求6的方法，其中，

与从未有锁定的锁状态的每一个存储的对象相关的方法链路至所述全局对象加锁过程；

与具有锁定的锁状态的每一个存储的对象相关的方法链路至包括所述局部对象加锁过程的所述过程子集；

当执行所述全局对象加锁过程以服务未锁定对象的锁请求时，更新所述特定对象的方法链路以链接到所述局部对象加锁过程；和

当执行所述锁数据清除过程时，改变所述特定对象的方法链路以链接到所述全局对象加锁过程。

9、如权利要求6的方法，其中

在所述计算机存储器内存储一组对象类，每一个对象类包括初始虚拟功能表(VFT)，该表包括至与所述对象类相关的一组方法的诸链路和至所述全局对象加锁过程的一链路；

与最近没有锁定的锁状态的每一个所述存储的对象相关的方法链路至对应一个所述对象类的所述初始VFT；

在所述计算机存储器内存储用于具有锁定的锁状态的每一个所述存储对象的局部虚拟功能表(VFT)，该表包括至所述局部对象加锁过程的链路；和

当执行所述全局对象加锁过程时，更新所述特定对象的方法链路以链接到特定对象的所述局部VFT。

10、如权利要求6的方法，其中，

在所述计算机存储器中存储一组对象类，每一个对象类包括初始虚拟

功能表(VFT)，该表包括至与所述对象类相关的一组方法的诸链路和至所述全局对象加锁过程的一链路；

与从未具有锁定的锁状态的每一个所述存储对象相关的方法链路至对应一个所述对象类的所述初始 VFT；

在所述计算机存储器中存储用于具有锁定的锁状态的每一个所述存储的对象的局部虚拟功能表(VFT)，该表包括至所述局部对象加锁过程的链路；

当执行所述全局对象加锁过程时，更新所述特定对象的方法链路以链接到所述特定对象的所述局部 VFT；和

当执行所述锁数据清除过程时，改变所述特定对象的方法链路以链接到对应所述特定对象的对象类的所述初始 VFT。

高效利用空间的对象加锁的系统和方法

本申请为在 1995 年 12 月 8 日递交的美国申请 08/569753 的后继申请。

技术领域

本发明一般是关于面向对象的计算机系统，其中两个或多个执行线程能相对于对象同步，更具体是关于在一个系统中有效分配锁数据结构的系统和方法，在该系统中，尽管大多数或所有的对象是能够锁定的，但事实上只有少数对象曾被锁定。

背景技术

在多处理器计算机系统中，软件程序可以通过在构成多处理器计算机系统的处理器上并行运行的线程来执行，故而，由于同时运行的线程能同时地执行程序，一个程序可以在不同的处理器上并行地运行。进而，如果程序可以分为分过程，这种计算机系统就可以非常快地运行该程序，原因是同时运行的线程可以并行地执行分过程。单个处理器、多任务计算机系统通过使用各种资源调度机构也可以虚拟同时执行多线程，这对于多任务操作系统设计领域的技术人员是熟知的。

在这样计算机系统上运行的程序经常是面向对象的，换言之，执行程序的线程可以调用执行特定功能的对象方法。然而，由于计算机系统硬件或软件的制约，一些对象的一些方法在某一时间上仅能执行一个。例如，一个对象可以要求访问共享的计算机资源，例如 I/D 设备，该资源在某一时间上仅能处理一个线程的一次访问。这样，由于同时运行多个线程会同时地寻求调用这样一个对象，该对象在某一时间上必须仅和一个线程同步，以使得只有那个线程唯一地使用该对象(即，在某一时间上只有一个线程能够拥有该对象的锁)。

在过去，已经有各种方法被用来将一对象和一线程同步。这些方法包括使用象互斥(mutexe)、条件变量和监视器等同步结构。当使用监视器时，每一个监视器鉴别当前拥有该对象的线程和等待拥有该对象的那些线程。然而，在使用这些监视器的计算机系统中经常存在用于每一个可同步对象

的监视器，所以，这种途径具有需要多个存储器的明显缺点。

减少这些监视器所需存储器的简单途径是给这些监视器分配高速缓冲存储器并对每一个监视器操作在该高速缓冲存储器中查阅一散列表。该途径其实增加了监视器操作的总开销，并且此种解决方案速度低，所以提出了本发明。

发明内容

本发明的目的是提供一对象加锁系统，其中存储空间按需要分配给锁数据以避免将存储器资源分配给虽可锁定但在事实上从未被锁定的对象的锁数据结构。

本发明的另一个目的是提供锁数据分配系统和方法，该系统和方法计算效率高并且基本上没有增加用于常用对象的计算总开销，该系统和方法使用的存储器资源量和锁定对象的数目成正比。

总之，本发明是具有存储多个对象和多个过程的存储器的多线程计算机系统，每一个对象具有锁定或非锁定的锁状态。这里有两种表示对象的方式，一种是由数据结构的数据指针和方法阵列的方法指针组成的句柄，另一种作为指向对象数据结构的直接指针，该指针的第一个元素是方法阵列的方法指针。对于本发明而言，这两种表示没有显著差别。

在任一情况下，方法阵列包括用于该对象的加锁和非加锁过程，每一个对象还可以是一些类的实例，并且具有与该类相关的类数据结构的数据基准，该基准和该对象的方法阵列一起存储。作为同类实例的两个对象则需共享此类数据结构，并且具有该类数据结构的相同基准。该类数据结构包括永久分配的与该类相关的类锁数据结构，该类锁数据结构能用于对不然就没有相关锁数据结构的诸对象的同步修改。

该系统使用单一全局对象加锁过程作为加锁过程以服务于还没有分配锁数据子阵列的诸对象(从未被锁住的对象和近来没被锁住的对象)的锁请求，并且使用局部特定对象加锁过程作为加锁过程以服务于已经分配锁数据子阵列的诸对象(即被锁定的对象和近来已经被锁定的对象)的锁请求。

全局对象加锁过程具有产生特别用于待加锁对象的局部对象加锁过程的指令。局部特定对象加锁过程包括作为私用数据的用于存储锁数据的锁数据子阵列。局部特定对象加锁过程具有更新该对象的存储的锁数据的诸指令。如果该对象近来没有被锁定的话，当系统的无用数据收集过程执行

时执行的锁数据清除过程释放局部特定对象加锁过程使用的存储器。

在优选实施例中，没有分配锁数据子阵列的每一个对象具有一方法指针，该指针指定包括全局对象加锁过程的一组过程，这些对象不需在锁定的状态。已经分配局部特定对象加锁过程的每一个对象具有一方法指针，该方法指针指定包括该对象的局部特定对象加锁过程的一组过程。另外，全局对象加锁过程包括用于更新特定对象的方法指针以指向包括局部特定对象加锁过程的一组过程的诸指令。

锁数据清除过程包括一些指令，当特定对象的更新锁数据指出近来没有被锁定的那个特定对象时，这些指令被激活以改变此特定对象的方法指针以指向包括全局对象加锁过程的一组过程。

更具体而言，优选实施例中的计算机系统包括一组对象类，每一个对象类中包括一个虚拟功能表(VFT)，该表包括指定与对象类相关的一组方法的诸指针及指定全局对象加锁过程的一指针。近来没有被锁定的每一个对象具有一方法指针，该方法指针指定用于对应的对象类的 VFT。

对于近来被锁定的每一个对象而言，该系统存储一特定对象虚拟功能表(VFT)，该表包括指定一组和它的对象类相关的方法的诸指针和指定对象的局部对象加锁过程的一指针。全局对象加锁过程包括产生用于此特定对象的 VFT 和更新特定对象的方法指针以指定它的特定对象 VFT 的诸指令。

附图说明

通过参照附图详细地说明和所附权利要求，将清楚地理解本发明的其它目的和特点。附图中：

图 1 是本发明优选实施例的计算机系统的方框图。

图 2 是本发明优选实施例的还没有分配锁数据子阵列的对象的数据结构的框图。

图 3 是本发明优选实施例的已经分配了锁数据子阵列的对象的数据结构的框图。

图 4 和 5 是本发明优选实施例中锁定一对象过程的流程图。

图 6 是锁数据清除方法的优选实施例的流程图。

图 7 是锁数据清除方法的另一实施例的流程图。

具体实施方式

参看图 1，这里示出了具有多用户计算机 102 和多服务器计算机 104

的分布式计算机系统100。在优选实施例中，每一个用户计算机102通过互连网络(Internet)103连接到服务器104，当然也可使用其它类型的通讯连接。尽管大多数用户计算机是台式计算机，例如Sun工作站，IBM兼容计算机和苹果(Macintosh)计算机，实际上任何类型计算机都可成为用户计算机。在优选实施例中，每一个用户计算机包括CPU105、通讯接口106、用户接口107、和存储器108。存储器108存储：

· 操作系统109；

· 网络通讯管理器程序110；

· 校验是否特定程序满足特定预先规定完整性判据的字节码程序校验器112；

· 执行应用程序的字节码程序解释器114；

· 类装入器116，它把对象类装入用户地址空间和使用字节码程序校验器去校验与每一装入对象类相关的方法的完整性；

· 至少一个类库120，用于局部存储用户计算机102使用和/或可加以使用的对象类122；

· 用于存储对象126的至少一个对象库124，该存储的对象是存储在对象库120内诸对象类的对象的实例。

在优选实施例中的操作系统109是面向对象的多任务操作系统，它支持在每一个定义的地址空间中执行多线程。该操作系统还使用无用数据收集过程去恢复同被释放的数据结构相关的存储单元。无用数据收集过程周期性地自动地执行，并且当可被分配的存储单元量低于一阈值水平时，自动地附加调用次数。本文中可以假定在系统109中所有对象是可锁定的对象，然而在实际中仅有相对少数对象实际上曾经被锁定。

当用户最初初始化一执行过程时，一般要调用类装入器116，这就要求产生适当的对象类的对象。类装入器116装入适当的对象类，并调用字节码程序校验器112去校验在装入的对象类的所有字节码程序的完整性，如果所有方法被成功地校验就产生该对象类的对象，并且字节码解释器114被调用以执行常被称为方法的用户要求的过程。如果用户要求的过程不是字节码程序且允许执行非字节码程序(这超出了本文的范围)，则该程序是由被编辑的程序执行器(未示出)执行。

无论何时执行的字节码程序遇到调用还没有被装入用户的地址空间的

对象类的对象方法，就调用类装入器，类装入器 116 再次装入适当的对象类和调用字节码程序校验器 112 去校验装入对象类的所有字节码程序的完整性。在许多情况下，对象类从远地计算机例如图 1 所示服务器 104 中的一个被装入。如果成功地校验了装入对象类的所有方法，就产生该对象类的对象实例，而且字节码解释器 114 被调用以执行被调用的对象方法。

本文中定义的同步方法为包括使用加锁方案以限制能同时使用系统资源的执行线程(以后称为线程)的数目的方法。最常用的同步工具是互斥，互斥能使唯一的线程在任何一时间上使用特定的系统资源，互斥包括用于跟踪等待使用资源的执行线程的机构。虽然本文中描述的同步机构是互斥型加锁机构，但本发明的方案同样适用于具有其它同步机构的计算机系统，这些同步机构包括但并不限于标志信号、时间锁限期等等。

在优选实施例中，被同步的方法总是对特定对象同步。例如，多线程可以执行要求唯一使用特定对象代表的系统资源的被同步的方法。当线程中的任何一个占有了一对象的锁时，所有其它要求占有该对象的锁的线程被迫等待，直到所有较早的线程得到并且然后释放该对象的锁为止。

非锁定和锁定对象的数据结构

图 2 示出了本发明优选实施例的近来还没有被锁定的对象的数据结构 200。如随后所描述的，所有这样的对象必需是非锁定的而且没有被分配锁数据子阵列。在一个优选的实施例中，短语“对象 X 最近没有被锁定”所定义的意思是，从由操作系统的最后一个无用数据收集周期以来对象 X 没有被锁住。在另一个优选实施例中，术语“最近”可以定义为预定的时间量，例如特定的秒数或从计算机系统的可靠周期性事件算起的时间周期而不是从执行无用数据收集过程时算起。

对象类 A 的对象具有对象句柄 202，该对象句柄包括指向对象方法的指针 204 和指向对象数据阵列 208 的指针 206。

对象方法指针 204 实际上是指向相关对象类的方法的间接指针。更具体而言，方法指针 204 指向对象的对象类的虚拟功能表(VFT)210。每一个对象类具有 VFT 210，它包括：(A)指向该对象类的每一个方法 214 的指针 212；(B)指向使对象与线程同步的全局锁方法(全局锁 1)216 的指针 215；(C)特定类对象 218 的指针 217；和(D)指向释放锁监视器的开锁方法 221 的指针 220。对每一个定义的对象类，存在着一个类对象 218，该类对象包括永

久分配给该类的锁数据子阵列(有时称锁监视器)219。在优选实施例中类对象 218 被用于同步作为相应对象类实例的所有对象的对锁数据子阵列的访问。

如图 2 所示，不管可以存在多少个对象类 A 的对象，仅有整个对象类 A 的 VFT 210 和对象方法 214 的一个拷贝。另外，全局锁方法(全局锁 1)216 和开锁方法 221 是“对象”对象类的方法，这一对象类处在优选实施例中对象类层级的顶部。

全局锁 1 方法 216 被用来处理线程请求去同步还没有分配锁数据子阵列的对象。开锁方法 221 被用来处理线程请求使对象去同步。如果不存在等待特定对象监视器的线程，开锁方法 221 按通常方法操作，移开特定对象的监视器的当前拥有者，设置锁状态为开锁状态，否则使最上边等待线程为特定对象监视器的拥有者，而将锁状态变为锁定状态。

图 2 也示出了“对象”对象类也包括第二锁方法 222，称为锁数据清除方法，用于从一对象回收锁数据子阵列。应当指出的是，三个锁方法 216、221 和 222 可作为公知的采用本发明方案的所有系统中可获得的任何对象类的方法来实现，而并不必是“对象”对象类的一部分。

图 3 示出了本发明优选实施例锁定对象的数据结构 240。这也是最近已经被锁住和已经被分配了锁数据子阵列的任何对象的数据结构。对象类 A 的锁定的对象具有包括指向对象方法的指针 244 和指向对象数据阵列 208 的指针 246 的对象句柄 242。

近来已经被锁定的对象的方法指针 244 指向特定对象的虚拟功能表 250(VFT, A-01)。每一个已经分配了锁数据子阵列的对象具有特定的 VFT(VFT, A-01)250，它包括：(A)指向该对象类的每一方法 214 的指针 212；(B)指向用于使对象与线程同步的局部特定对象加锁过程(局部锁 2)260 的指针 256；(C)指向特定类对象 218 的指针 217；和(D)指向用于释放锁监视器的开锁方法的指针 220。对于每一个定义的对象类，存在着一类对象 218，该类对象包括被永久分配的锁数据子阵列(或指锁监视器)219。

局部特定对象加锁过程 260 用于服务于被锁定的对象和近来已经被锁定对象的锁请求。局部特定对象的加锁过程 260 包括作为私用数据的用于存储锁数据的锁数据子阵列。局部特定对象加锁过程具有用于更新该对象的存储的锁数据的诸指令，更具体而言，局部特定对象加锁过程 260 具有

使相应对象与线程同步和存储相应锁信息到对象的锁数据子阵列 249 的特定对象锁方法(局部锁 2)。

锁数据子阵列249 包括对象锁状态指示器、锁拥有者的值和等待者(即等待与对象同步的诸线程)表的表头和表尾，在优选实施例中，锁状态指示器包括指示是否对象被锁定或未锁定的第一标志值(锁标志)，和指示是否对象最近已经被锁定的第二标志值(非最近锁定标志)。如果对象最近没有被锁定，非最近锁定标志被设置(置为“真”)。在优选实施例中，非最近锁定标志被锁数据清除方法置位，由全局锁 1 和局部锁 2 方法清除。

对象加锁办法

每一个计算机系统，例如用户计算机 102，具有多个对象，而每一个对象具有相关的对象类，每一个对象被认为是它相关对象类的一实例，每一对象类继承了它的最高级类的性质，每一对象类是被称为“对象”对象类的顶级对象类的子类。

对于存于特定地址空间中的每一个对象类，存在着虚拟功能表(VFT)，该表包括与该对象类相关的拥有方法(即可执行过程)的列表和指向这些方法每一个的指针。如图 2 所示，每一对象类的 VFT 也包括对全局锁 1 方法的指定，在优选实施例中该方法是与“对象”对象类相关的方法。无论何时对象还没有分配锁数据子阵列，它的方法指针指向对象的对象类的缺省 VFT。

依照本发明的第一个优选实施例，每一个对象类具有上述的相关虚拟功能表作为第一 VFT，有时指“初始 VFT”。另外，对于每一个最近已经被锁定的对象，系统产生特定对象虚拟功能表(VFT)，特定对象 VFT 指定局部锁方法，局部锁 2 260，该局部锁方法不同于初始 VFT 指定的全局锁方法，全局锁 1 216。

表 1, 2, 3 和 4 包括与本发明相关的全局锁 1，局部锁 2，和锁数据清除软件过程的伪代码表示，在这些附录中使用的伪代码使用通用的计算机语言的惯例。这里采用的伪代码仅仅用于描述本发明，以便使本技术领域的计算机程序员容易理解。

参看图 4 和表 1 示出的全局锁 1 方法 216 的伪代码，当还没有分配锁数据子阵列的对象是同步的方法调用的主题时，与该对象相关的全局锁方法(全局锁 1)被调用。通过请求和等待与要被锁定的对象相关的类对象的锁，

全局锁 1 方法开始(步骤 270)。全局锁 1 的其余部分并不执行，直到使全局锁 1 方法调用的线程获得了类对象的锁。

全局锁 1 和锁数据清除方法需要被同步，通过获取类对象的锁来防止由于并发引起的错误。例如，在多处理机系统中，全局锁 1 过程可同时被同一对象的两个处理器调用。除非采取防护措施，不然这可能导致引发多个特定对象 VFT 和两个特定对象局部锁 2 的过程。为了解决这个问题，在重新安排特定对象的内部数据结构时，需要在全局锁 1 和锁数据清除过程中对特定对象类型的类对象加锁。

在获得类对象的锁之后，检查是否对象被锁定(或最近被锁定)或未被锁定(或最近未被锁定)。是否对象被锁定可以通过校验特定对象加锁过程和特定对象 VFT 的存在加以确定(步骤 271)。在特定对象加锁过程和特定对象 VFT 存在的情况下，方法前进到步骤 276。否则的话，全局锁 1 方法产生包括作为私用数据的局部锁数据子阵列的局部特定对象锁过程(步骤 272)。通过存储在锁数据子阵列中表示局部拥有者线程标识的数据和通过清除非最近锁定标志，全局锁 1 也就初始化了局部锁数据子阵列。进而，全局锁 1 方法为这个对象产生了特定对象 VFT，以指定特定对象加锁过程(步骤 272)。接着，对象的方法指针被改为指向特定对象 VFT。然后，类对象的锁被释放(步骤 276)。最后，局部加锁方法，局部锁 2 被调用(步骤 278)。由于对象方法指针的更换和特定对象的 VFT 的产生，局部锁 2 方法将自动地被调用来处理该对象的后续同步请求。

参看图 5 和在表 2 示出的局部锁 2 方法 260 的伪代码，局部锁 2 方法简单地执行常规的接收同步请求的服务，这包括锁数据的正常更新(步骤 290)。另外，任何对局部锁 2 方法的调用都使特定对象的非最近锁定标志被清除(步骤 300)。在另一实施例中，在同步请求被处理之后，仅当对象具有锁定的锁状态时，特定对象的非最近锁定标志才被清除。局部锁 2 方法是如此简单，这是因为众所共知，无论何时局部锁 2 方法被调用，特定的对象(即锁处理的主题)已经具有了锁数据子阵列。

对于正常的互斥操作，如果线程的锁处理请求(即由局部锁 2 方法处理的请求)要与相关的对象同步，如果对象已经被锁定，那么该线程被加入到对象的等待线程表。否则，如果对象没有被锁住，请求的线程成为锁的拥有者和锁状态被变为锁定。

开锁方法 221 对要求释放线程保持的锁的请求进行处理。等待的线程，如果有一些的话，在等待线程表中的最高的线程被作为锁的拥有者并且允许恢复执行。如果不存在等待的线程，那么锁状态被更新为“未锁定”，该“未锁定”在某些应用情况下可以通过锁的拥有者数据被改为“空”值和锁状态标志被重置为“假”来简单地表示。

局部锁 2 处理对对象的所有同步请求，甚至对象变成了未锁定之后，直至对象的锁数据子阵列被锁清除方法解除。

图 6 描述了锁数据清除方法的第一个优选的实施例，该清除方法释放了存储的特定对象 VFT 和特定对象加锁过程。该实施例是在锁数据清除方法不要求与另一个同时发生的线程活动同步的环境下被使用。参看图 6 和在表 3 示出的锁数据清除方法 222 的伪代码，在优选的实施例中，每次无用数据收集过程的执行被初始化时，锁数据清除方法被无用数据收集过程过程调用以校验分配了锁数据子阵列的所有对象。或者，每次无用数据收集过程被初始化时，锁数据清除方法被无用数据收集过程所调用以检验所有对象。

在本发明优选实施例中，如果任何对象的锁数据子阵列在两个无用数据收集周期之间的时间间隔内没被使用，那么该对象被认为最近没有被锁定。更具体而言，在一个无用数据收集周期中，如果对象的非最近锁定标志被锁数据清除过程置为“真”，并且在下一个无用数据收集周期仍维持为“真”，那么那个对象的锁数据子阵列(实际上是特定对象 VFT 和特定对象局部锁过程)被释放。

通过确定由调用过程(例如，无用数据收集过程)指定的对象是否具有局部的特定对象加锁过程，锁数据清除过程开始(步骤 302)，局部的特定对象加锁过程的存在表示存在要回收的存储单元，否则过程返回。其次，该过程确定是否特定的对象被锁定或者是否该对象具有至少一个等待的线程(步骤 304)。这些条件表明分配给特定对象加锁过程和 VFT 的存储单元在此时并不需要被回收，这是因为仍存在着供它们使用的需要。在这种情况下(步骤 304-Y)，该方法返回。

否则，做进一步校验以确定是否该对象最近被锁定(步骤 306)。在优选的实施例中，如果在它的锁数据子阵列中非最近锁定标志被设置为“真”，该对象被认为最近没有被锁定。如果它具有锁数据子阵列和非最近锁定标

志被置定为“假”(即标志没有被设置)，那对象被认为最近已经被锁定。

如果对象最近已经被锁定，该对象的非最近锁定标志被设置为“真”(步骤308)和过程返回。如果锁数据子阵列在那个时间内未被使用，在步骤308准备下一个无用数据收集周期内释放的对象的锁数据子阵列。

如果对象最近没有被锁定，为局部的特定对象加锁过程和为特定对象VFT分配的存储单元被释放(步骤310)。最后，该对象的方法指针被设置到对象的类对象的初始VFT(步骤312)。

图7描述了锁数据清除过程的另一实施例。在这一实施例中，锁数据清除过程与其它线程的加锁活动同时执行，从而要求锁数据清除过程与这些同时发生的过程同步。参看图7和表4示出的锁数据清除方法222的伪代码，每当执行无用数据收集过程的初始化时，该锁数据清除方法被无用数据收集过程调用以校验分配了锁数据子阵列的所有对象。或者，每当执行无用数据收集过程的初始化时，锁数据清除方法被无用数据收集过程调用以校验所有的对象。

通过确定是否由调用过程(例如，无用数据收集过程)指定的对象具有局部的特定对象加锁过程，锁数据清除过程开始(步骤402)。局部的特定对象加锁过程的存在表明事先分配了可能要回收的存储单元。如果不存在(步骤402-N)，过程返回。接着，过程确定是否特定的对象被锁定或者是否对象具有至少一个等待的线程(步骤404)。这些条件表明，为特定对象加锁过程和VFT分配的存储单元在此时并不需被回收，原因是仍存在着供它们使用的需要。在这种情况下(步骤404-Y)，过程返回。执行下一步的校验以确定是否对象最近已经被锁定(步骤406)。在优选实施例中，如果在它的锁数据子阵列中的非最近锁定标志被设置为“真”，那么对象被认为最近没有被锁住。如果它有锁数据子阵列和非最近锁定标志被设置为“假”(即标志没有被设置)，对象被认为最近被锁住。

如果对象最近被锁住，对象的非最近锁定标志被设置为“真”(步骤408)且程序返回。如果在那个时间内锁数据子阵列未被使用，在步骤408准备下一个无用数据收集周期内释放的对象的锁数据子阵列。

接下来，锁数据清除方法请求与执行加锁活动的其它同时运行的线程同步。该同步通过请求和等待与要被锁定对象相关的类对象的锁的方法来完成(步骤412)。锁数据清除方法的其余部分并不执行，直到进行锁数据清

除方法调用的线程获得了该类对象的锁为止。

一旦已经获得该类对象的锁，通过检查是否非最近锁定标志被设置，锁数据清除方法再一次校验来观察是否该锁最近被锁定(步骤 414)。注意，有可能当调用的线程正在等待类对象的锁时，另一个线程锁定该特定对象。在那种情况，解除该特定对象的锁数据子阵列是不合适的，这就是为什么必须第二次校验该特定对象的非最近锁定标志(414)。如果非最近锁定标志没有被设置(414-N)，锁数据清除方法释放该类对象的锁(步骤 420)并返回。如果非最近锁定标志仍被设置 (414-Y)，锁数据清除方法释放为特定对象 VFT 和特定对象加锁过程预先分配的存储单元(416)。接着，对象的对象方法指针被设置到对象的类对象的初始 VFT(步骤 418)。最后，对象的类对象的锁被释放(步骤 420)。

使用上述的办法，没有被锁定的对象不增加支持对象加锁的存储总开销。仅仅被锁定的对象占有被分配以存储锁数据的存储空间。

虽然参考几个特定的实施例描述了本发明，但该说明是示范性的且并不能被认为是对本发明的限制。本领域技术人员对此进行的各种修改并未脱离由所附权利要求定义的本发明的精神和范围。

例如，不同于非最近锁定标志的其它机构能被用于确定是否对象最近被锁定。例如，在它们未被锁定时，时间标志可以存储在未锁定对象的锁数据子阵列中，该时间标志可以由锁数据清除过程校验。如果时间标志表示的一段过去的时间大于一时间阈值量，则能够确定该对象最近没有被锁定。

虽然以上描述的锁数据子阵列适用于实现互斥，但相同的锁数据子阵列分配和释放办法和机构能够用于分配和释放更复杂的锁数据结构，例如标志信号的数据结构和监视器的数据结构，来处理诸通知事件的等待。

另外，以上描述的方法和系统还适用于除了例如随机存取存储器的存储器件之外的任何类型的可行介质。可以使用其它类型的可行介质，例如存储器器件、光盘或软盘但不局限于计算机可读存储介质。

表 1

全局锁 1 方法的伪代码表示

```
过程：全局_锁 1(对象，指令)
/*对象自变量是要被锁定的对象*/
{
/*要求锁定以确保多线程不试图处理同一个对象*/
请求和等待类(对象)的类对象的锁。
/*处理方法的步骤*/
如果对象没有特定对象的虚拟功能表和特定对象加锁过程
  {产生局部的、特定对象的、对象加锁过程，它包括作为私用数据的
   该对象的锁数据子阵列。
   产生该对象的特定对象虚拟功能表，它指定该对象的局部的、特定
   对象的、对象加锁过程。
   改变对象的方法指针以指向该对象的特定对象 VFT。
  }
释放类对象的锁。
调用局部的、特定对象的、由对象的 VFT 指定的对象加锁过程。
返回
}
```

表 2

局部锁 2 方法的伪代码表示

```
过程：局部_锁 2(对象，锁命令)
{
/*对象已经具有锁数据子阵列*/
依照接收到的锁命令执行正常的锁更新处理。
清除对象的非最近锁定标志。
返回
}
```

表 3

锁数据清除方法的伪代码表示

```
过程：锁_数据_清除(对象)
{
如果对象没有局部的特定对象的对象加锁过程。
  {返回}
如果对象被锁定或对象具有至少一等待线程
  {返回}
/*如果对象未被锁定但它的非最近锁定标志被设置为“假”，建立在下一个无用数据收集周期释放的对象的局部对象加锁过程（和锁数据子阵列）*/
如果对象的非最近锁定标志被设置为“假”(即，从前一个无用数据收集周期以来，它已经被一个局部_锁方法清除)
  {
   设置非最近锁定标志为“真”
   返回
  }
/*对象最近没有被锁定*/
   释放局部的、特定对象的、对象加锁过程使用的存储单元(包括锁数 据使用的数据阵列)
   释放特定对象 VFT 使用的存储单元
   改变对象的方法指针以指向对象类的标准 VFT
  }
返回
}
```

表4

另一锁数据清除方法的伪代码表示

```
过程：锁数据_清除(对象)
{
如果对象没有局部的特定对象的对象加锁过程
    {返回}
如果对象被锁定或对象具有至少一等待线程
    {返回}
如果对象最近被锁定
/*建立在下一个无用数据收集周期释放的对象的局部对象加锁过程(和锁数据子阵列)*/
    {
    非最近锁定标志＝“真”
    返回
    }
/*获得锁以确保多线程不试图处理同一个对象*/
要求和等待类(对象)的类对象的锁
/*再次校验非最近锁定标志*/
如果对象最近没有被锁定(例如，自前一个无用数据收集周期以来还没有)
    {
    释放局部的、特定对象的、对象加锁过程使用的存储单元(包括锁数据子阵列使用的数据阵列)
    释放特定对象 VFT 使用的存储单元
    改变对象的方法指针以指向对象类的标准 VFT
    }
释放类对象的锁
返回
}
```

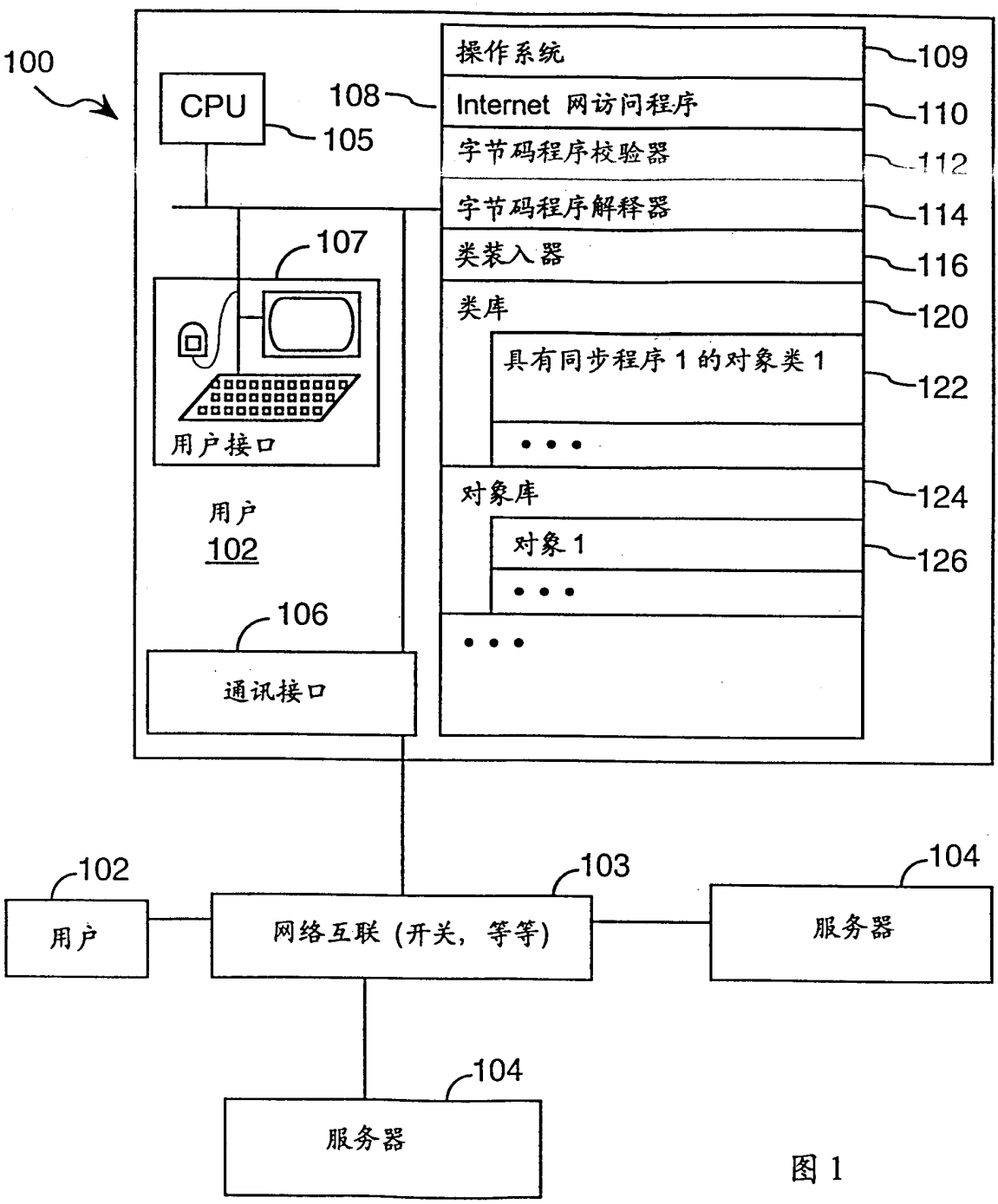

100
CPU
105
108
操作系统
109
Internet 网访问程序
110
字节码程序校验器
112
字节码程序解释器
114
类装入器
116
类库
120
具有同步程序 1 的对象类 1
122
• • •
对象库
124
对象 1
126
• • •
• • •
107
用户接口
用户
102
106
通讯接口
102
用户
103
网络互联 (开关，等等)
104
服务器
104
服务器

图 1

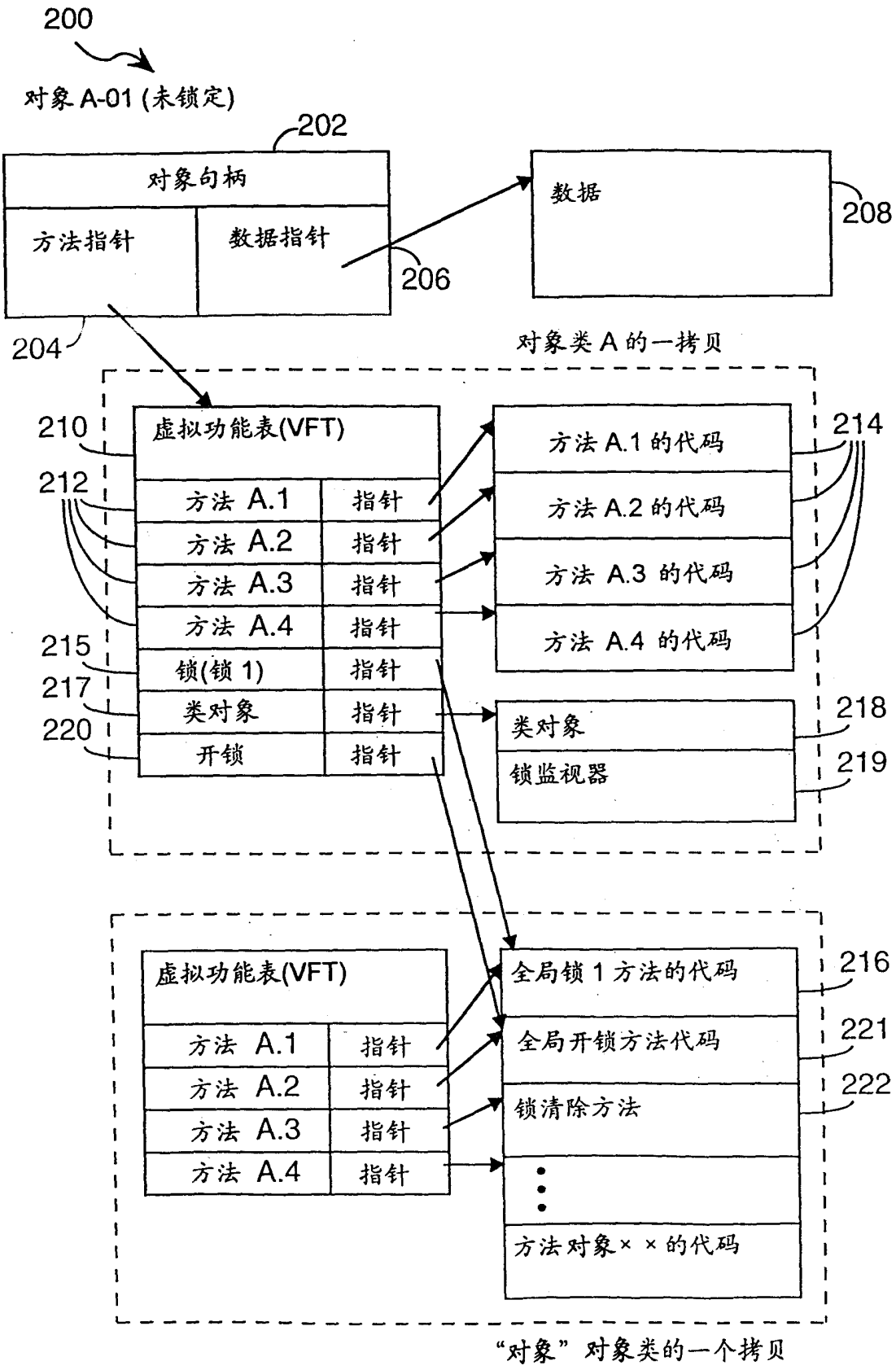

200
对象 A-01 (未锁定)
202
对象句柄
方法指针
数据指针
206
数据
208
204
对象类 A 的一拷贝
210
虚拟功能表(VFT)
212
方法 A.1
指针
方法 A.2
指针
方法 A.3
指针
方法 A.4
指针
215
锁(锁 1)
指针
217
类对象
指针
220
开锁
指针
方法 A.1 的代码
方法 A.2 的代码
方法 A.3 的代码
方法 A.4 的代码
214
类对象
锁监视器
218
219
虚拟功能表(VFT)
方法 A.1
指针
方法 A.2
指针
方法 A.3
指针
方法 A.4
指针
全局锁 1 方法的代码
全局开锁方法代码
锁清除方法
方法对象× ×的代码
216
221
222
“对象”对象类的一个拷贝

图 2

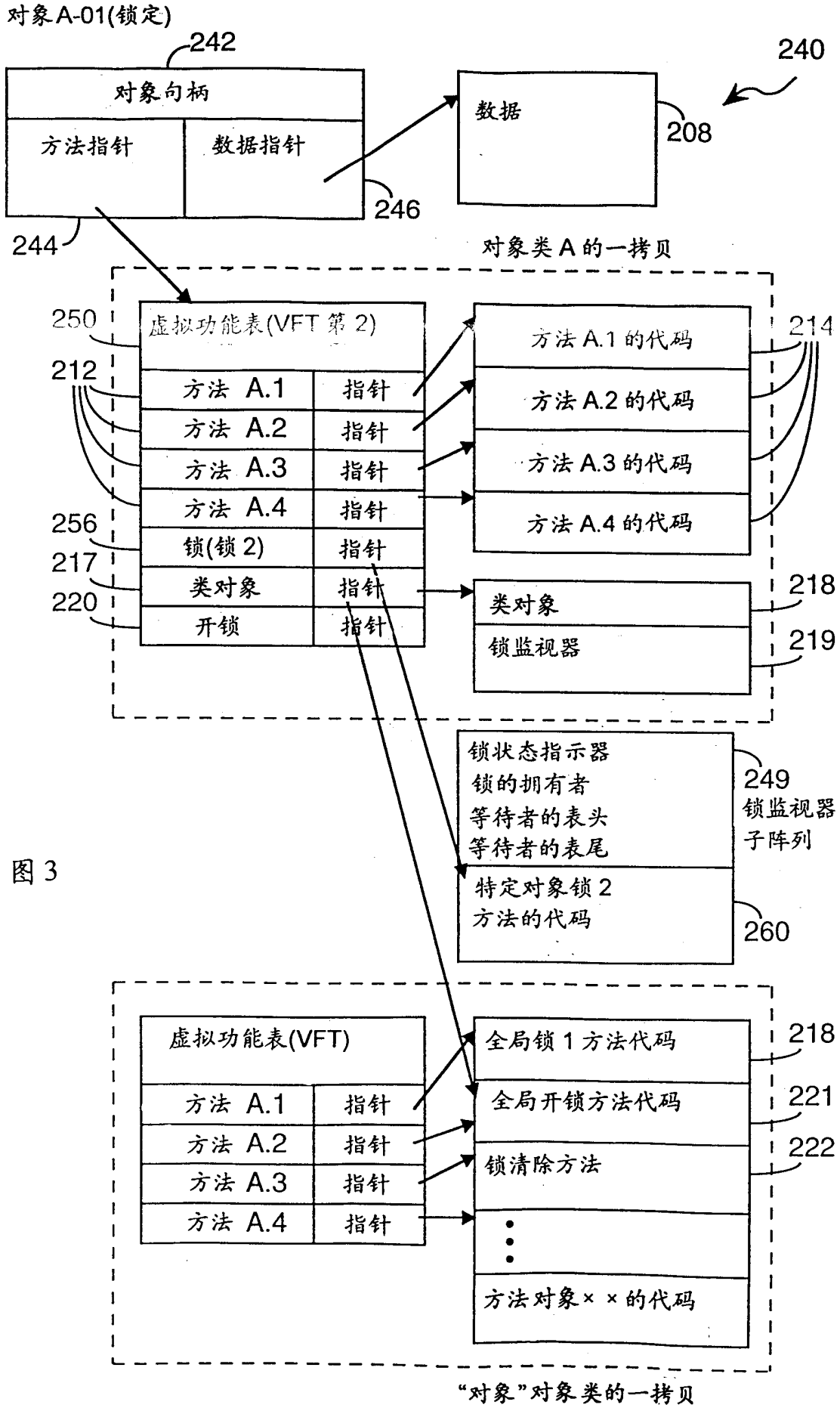

对象A-01(锁定)
242
对象句柄
方法指针
数据指针
246
244
240
数据
208
对象类A的一拷贝
250
虚拟功能表(VFT 第2)
212
方法 A.1
方法 A.2
方法 A.3
方法 A.4
256
锁(锁2)
217
类对象
220
开锁
指针
方法A.1的代码
方法A.2的代码
方法A.3的代码
方法A.4的代码
214
类对象
锁监视器
218
219
锁状态指示器
锁的拥有者
等待者的表头
等待者的表尾
249
锁监视器
子阵列
特定对象锁2
方法的代码
260
虚拟功能表(VFT)
全局锁1方法代码
全局开锁方法代码
锁清除方法
方法对象××的代码
221
222
"对象"对象类的一拷贝

图 3

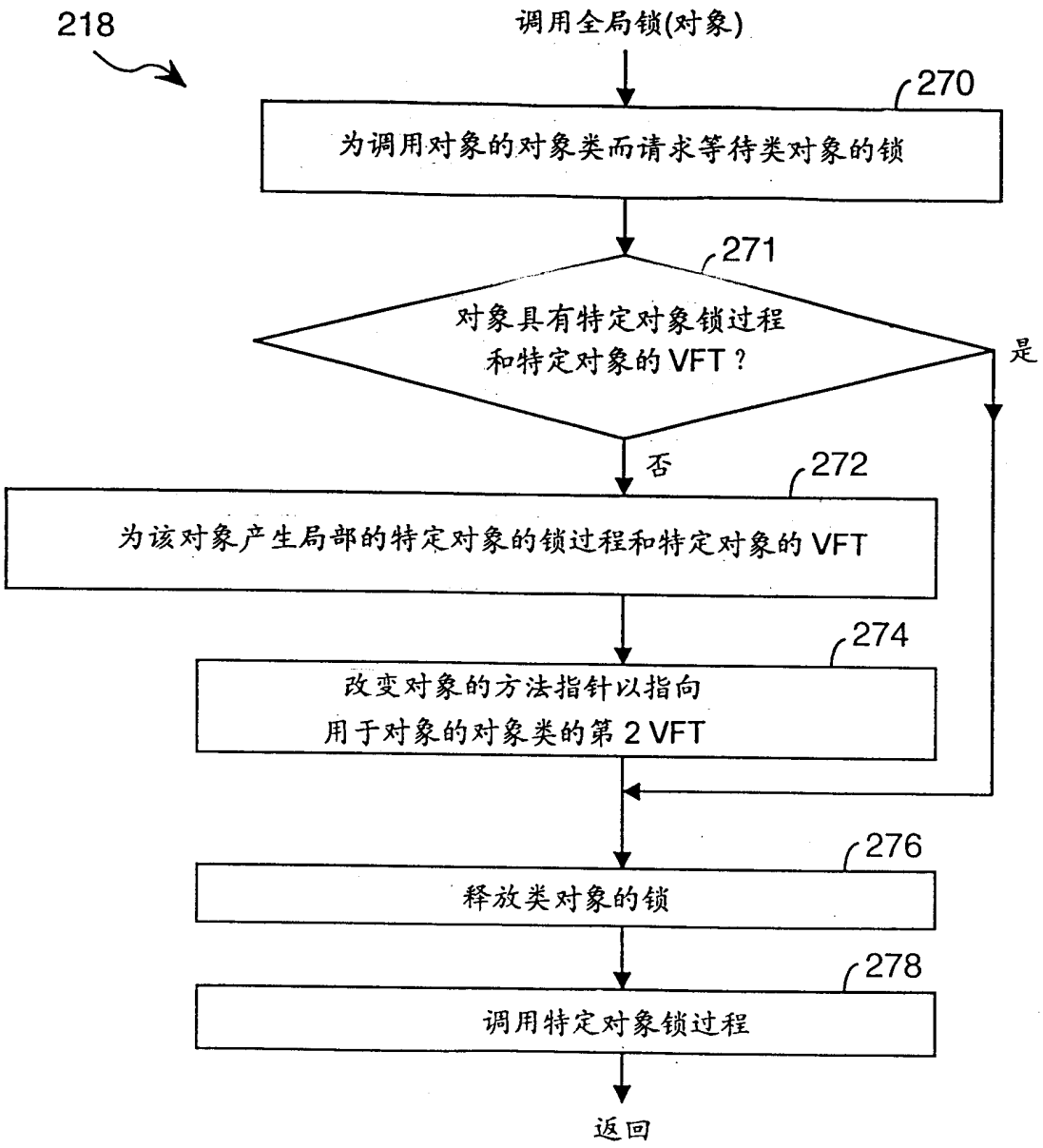

218
调用全局锁(对象)
270
为调用对象的对象类而请求等待类对象的锁
271
对象具有特定对象锁过程
和特定对象的 VFT ?
是
否
272
为该对象产生局部的特定对象的锁过程和特定对象的 VFT
274
改变对象的方法指针以指向
用于对象的对象类的第 2 VFT
276
释放类对象的锁
278
调用特定对象锁过程
返回

图 4

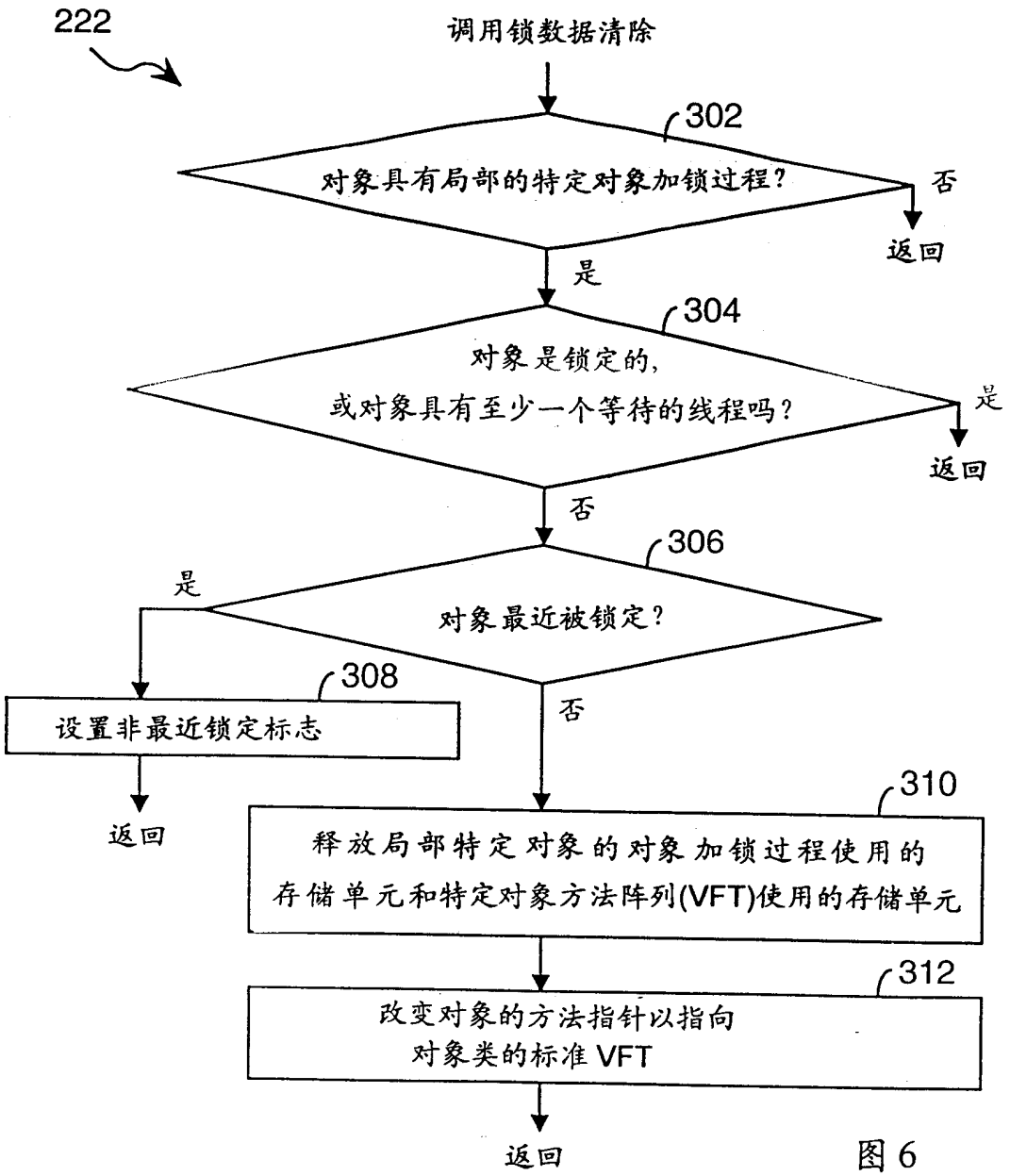

222
调用锁数据清除
302
对象具有局部的特定对象加锁过程?
否
返回
是
304
对象是锁定的,
或对象具有至少一个等待的线程吗?
是
返回
否
306
是
对象最近被锁定?
308
设置非最近锁定标志
返回
否
310
释放局部特定对象的对象加锁过程使用的
存储单元和特定对象方法阵列(VFT)使用的存储单元
312
改变对象的方法指针以指向
对象类的标准 VFT
返回

图 6

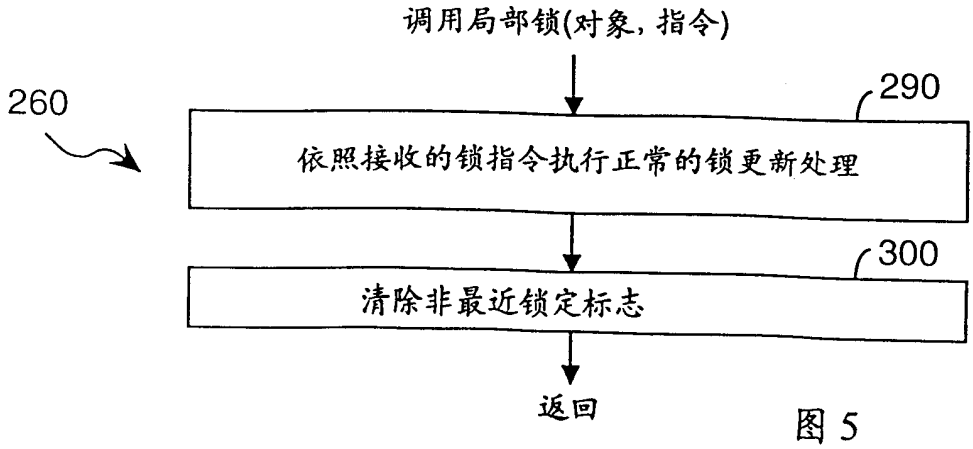

调用局部锁(对象, 指令)
260
290
依照接收的锁指令执行正常的锁更新处理
300
清除非最近锁定标志
返回

图 5

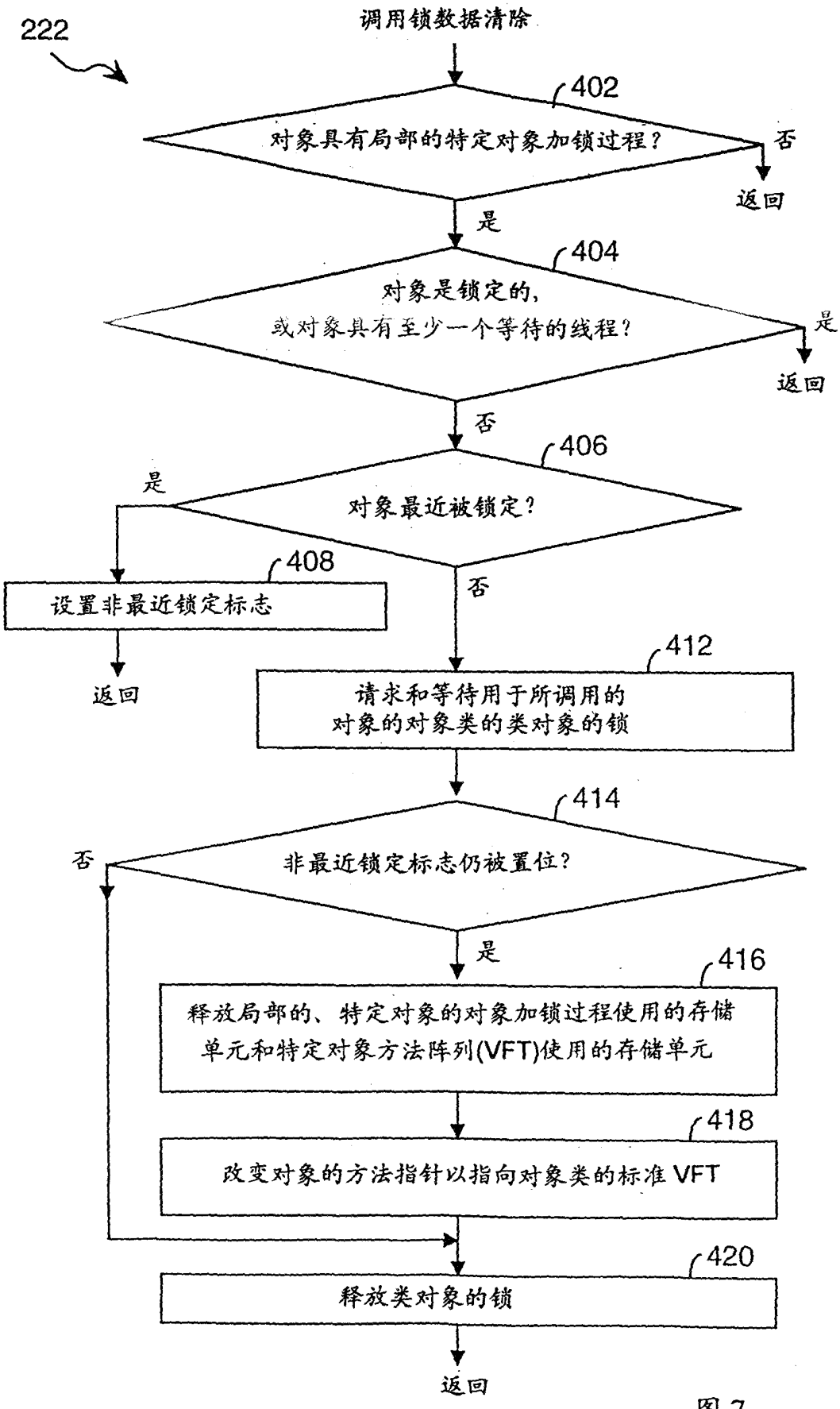

222
调用锁数据清除
402
对象具有局部的特定对象加锁过程?
否
返回
是
404
对象是锁定的,
或对象具有至少一个等待的线程?
是
返回
否
406
对象最近被锁定?
是
否
408
设置非最近锁定标志
返回
412
请求和等待用于所调用的
对象的对象类的类对象的锁
414
非最近锁定标志仍被置位?
否
是
416
释放局部的、特定对象的对象加锁过程使用的存储
单元和特定对象方法阵列(VFT)使用的存储单元
418
改变对象的方法指针以指向对象类的标准 VFT
420
释放类对象的锁
返回

图 7