(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国 际 局

(43) 国际公布日
2018 年 7 月 26 日 (26.07.2018)

WIPO | PCT

(10) 国际公布号
WO 2018/133713 A1

(51) 国际专利分类号:
G06F 9/44 (2018.01)

(21) 国际申请号: PCT/CN2018/072039

(22) 国际申请日: 2018 年 1 月 10 日 (10.01.2018)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
201710062788.2 2017年1月23日 (23.01.2017) CN

(71) 申请人: 阿里巴巴集团控股有限公司 (ALIBABA GROUP HOLDING LIMITED) [—/CN]; 开曼群岛大开曼资本大厦一座四层 847 号邮箱, Grand Cayman (KY)。

(72) 发明人: 徐伟刚 (XU, Weigang); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 吴哲锋 (WU, Zhefeng); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 孙哲 (SUN, Zhe); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 龚凯 (GONG, Kai); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 李庆岩 (LI, Qingyan); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 李勇彪 (LI, Yongbiao); 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 廖彬 (LIAO, Bin); 中国浙江省杭州市余杭区文一西路

(54) **Title:** THREAD MANAGEMENT METHOD AND APPARATUS

(54) 发明名称: 一种线程管理方法及装置

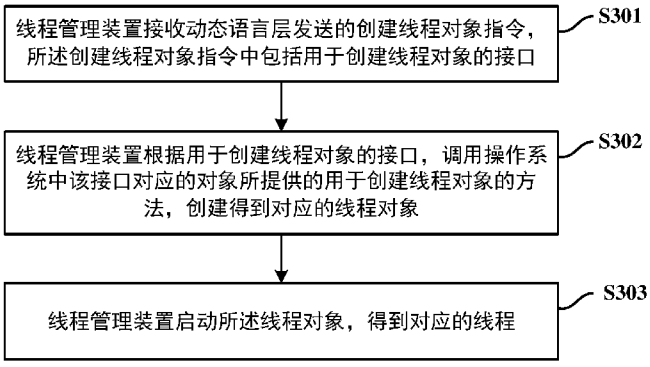

图 3

S301 A THREAD MANAGEMENT APPARATUS RECEIVES A THREAD OBJECT CREATION INSTRUCTION SENT BY A DYNAMIC LANGUAGE LAYER, THE THREAD OBJECT CREATION INSTRUCTION COMPRISING AN INTERFACE FOR CREATING A THREAD OBJECT

S302 THE THREAD MANAGEMENT APPARATUS INVOKES, ACCORDING TO THE INTERFACE USED FOR CREATING A THREAD OBJECT, A THREAD OBJECT CREATION METHOD PROVIDED BY AN OBJECT CORRESPONDING TO THE INTERFACE IN AN OPERATING SYSTEM, AND CREATES A CORRESPONDING THREAD OBJECT

S303 THE THREAD MANAGEMENT APPARATUS STARTS THE THREAD OBJECT TO OBTAIN A CORRESPONDING THREAD

(57) **Abstract:** A thread management method and apparatus. The method comprises: a thread management apparatus receives a thread object creation instruction sent by a dynamic language layer, the thread object creation instruction comprising an interface for creating a thread object (S301); the thread management apparatus invokes, according to the interface used for creating a thread object, a thread object creation method provided by an object corresponding to the interface in an operating system, and creates a corresponding thread object (S302); and the thread management apparatus starts the thread object to obtain a corresponding thread (S303). In the method,

WO 2018/133713 A1

[见续页]

969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 **余超君(YU, Chaojun)**; 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 **王帆(WANG, Fan)**; 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。 **金跃(JIN, Yue)**; 中国浙江省杭州市余杭区文一西路969号3号楼5楼阿里巴巴集团法务部, Zhejiang 311121 (CN)。

(74) 代理人: **北京三友知识产权代理有限公司(BEIJING SANYOU INTELLECTUAL PROPERTY AGENCY LTD.)**; 中国北京市金融街35号国际企业大厦A座16层, Beijing 100033 (CN)。

(81) **指定国**(除另有指明，要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW。

(84) **指定国**(除另有指明，要求每一种可提供的地区保护): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布:

— 包括国际检索报告(条约第21条(3))。

a common thread creation process is provided based on a multilayer architecture, thereby providing a possibility for creating and managing multiple threads.

(57) 摘要: 一种线程管理方法及装置。所述方法包括，线程管理装置接收动态语言层发送的创建线程对象指令，所述创建线程对象指令中包括用于创建线程对象的接口（S301）；线程管理装置根据所述用于创建线程对象的接口，调用操作系统中所述接口对应的对象所提供的用于创建线程对象的方法，创建得到对应的线程对象（S302）；线程管理装置启动所述线程对象，得到对应的线程（S303）。上述方法基于多层架构提供了通用的线程创建过程，为多线程的创建与管理提供了可能性。

一种线程管理方法及装置

本申请要求2017年01月23日递交的申请号为201710062788.2、发明名称为“一种线程管理方法及装置”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

本申请涉及计算机技术领域，尤其涉及一种线程管理方法及装置。

背景技术

JavaScript 是一种动态类型、面向对象的高级编程语言，由欧洲电脑制造商协会（ECMA）制定的标准，被大多数网站所使用，也被目前的主流浏览器支持。JavaScript 语言程序的执行依赖于 JavaScript 虚拟机，也就是说，JavaScript 语言程序的执行并不是直接在操作系统上运行，而是需要通过 JavaScript 虚拟机来执行。

JavaScript 虚拟机负责执行 JavaScript 程序以及为 JavaScript 程序调度资源。

发明内容

本申请实施例公开了一种线程管理方法及装置。

第一方面，本申请实施例提供了一种线程管理方法，包括：

接收动态语言层发送的创建线程对象指令，所述创建线程对象指令中包括用于创建线程对象的接口，所述线程对象对应指定的应用程序；

根据所述用于创建线程对象的接口，调用操作系统中所述接口对应的对象所提供的用于创建线程对象的方法，创建得到对应的线程对象；

启动所述线程对象，得到对应的线程。

第二方面，本申请实施例提供了一种线程管理装置，包括：

接收模块，用于接收动态语言层发送的创建线程对象指令，所述创建线程对象指令中包括用于创建线程对象的接口，所述线程对象属于指定的应用程序；

管理模块，用于根据所述用于创建线程对象的接口，调用操作系统中所述接口对应的对象所提供的用于创建线程对象的方法，创建得到对应的线程对象；并启动所述线程对象，得到对应的线程。

第三方面，本申请实施例提供了一个或多个计算机可读介质，所述可读介质上存储有指令，所述指令被一个或多个处理器执行时，使得通信设备执行本申请实施例提供的

方法。

第四方面，本申请实施例提供的一种装置，包括：一个或多个处理器；以及，一个或多个计算机可读介质，所述可读介质上存储有指令，所述指令被所述一个或多个处理器执行时，使得所述装置执行本申请实施例提供的方法。

本申请的上述实施例中，在接收到动态语言层发送的创建线程对象指令后，由于所述创建线程对象指令中包括用于创建线程对象的接口，因此可以根据所述用于创建线程对象的接口，调用操作系统中所述接口对应的对象所提供的创建线程对象的方法，创建得到对应的线程对象，并进而启动所述线程对象，得到对应的线程。从而基于多层架构提供了通用的线程创建过程，为多线程的创建与管理提供了可能性。

附图说明

图 1 为本申请实施例中动态语言系统架构示意图；

图 2 为本申请实施例中多线程架构的示意图；

图 3 为本申请实施例提供的线程对象的创建流程示意图；

图 4 为本申请实施例中线程对象管理流程示意图；

图 5 为本申请实施例中采用同步锁进行线程同步互斥的流程示意图；

图 6 为本申请实施例中采用同步声明实现对方法进行同步互斥的流程示意图；

图 7 为本申请实施例中采用同步声明实现对语句进行同步互斥的流程示意图；

图 8a 和图 8b 为本申请实施例中的 JavaScript 对象的布局示意图；

图 9 为本申请实施例提供的针对对象进行同步互斥的流程示意图；

图 10 为本申请实施例提供的线程管理装置的结构示意图；

图 11 为本申请实施例提供的装置的结构示意图。

具体实施方式

虽然本申请的概念易于进行各种修改和替代形式，但是其具体实施例已经通过附图中的示例示出并且将在本文中详细描述。然而，应当理解，没有意图将本申请的概念限制为所公开的特定形式，而是相反，意图是覆盖与本申请以及所附权利要求一致的所有修改、等同物和替代物。

说明书中对“一个实施例”、“实施例”、“说明性实施例”等的引用，指示所描述的实施例可包括特定特征、结构或特性，但是每个实施例可以或可以不必包括特定特

征、结构或特性。此外，这样的短语不一定指的是相同的实施例。进一步地，认为在本领域技术人员的知识范围内，当结合实施例描述特定特征、结构或特性时，结合无论是否明确描述的其它实施例影响这样的特征，结构或特性。另外，应当理解，以“A，B和 C 中的至少一个”的形式包括在列表中的项目可以表示（A）；（B）；（C）；（A和 B）；（A 和 C）；（B 和 C）；或（A，B 和 C）。类似地，以“A，B 或 C 中的至少一个”的形式列出的项目可以表示（A）；（B）；（C）；（A 和 B）；（A 和 C）；（B 和 C）或（A，B 和 C）。

在一些情况下，所公开的实施例可以在硬件、固件、软件或其任何组合中实现。所公开的实施例还可以被实现为由一个或多个暂时性或非暂时性机器可读（例如，计算机可读）存储介质携带或存储的指令，其可以由一个或多个处理器读取和执行。机器可读存储介质可以体现为用于以机器可读形式（例如，易失性或非易失性存储器、介质盘或其他介质）存储或传输信息的任何存储设备，机制或其他物理结构的设备）。

在附图中，一些结构或方法特征可以以特定布置和/或顺序示出。然而，应当理解，可能不需要这样的具体布置和/或排序。相反，在一些实施例中，这些特征可以以与说明性附图中所示不同的方式和/或顺序来布置。另外，在特定图中包括结构或方法特征并不意味着暗示这种特征在所有实施例中都是需要的，并且在一些实施例中可以不包括或可以与其他特征组合。

本申请实施例提供了语言层面的动态语言多线程编程能力，使开发者能够使用多线程编程的能力，增加程序的性能以及应用范围。

动态语言是计算机编程语言中的一个语言类别，是一类在运行时可以动态地改变类型、结构的语言，在运行时函数和属性可以被增加、修改和删除。例如 JavaScript、Python、Ruby 等都属于动态语言。动态语言不需要编译即可运行，在运行时需要运行环境的支撑，这个环境叫做运行时环境，它包含动态语言运行所需要的所有要素，例如 Java 虚拟机、JavaScript 虚拟机等。

其中，JavaScrip 虚拟机也可称为 JavaScript 引擎，可以被解释为 JavaScript 托管运行时环境（managed runtime environment for JavaScript），简称为运行时环境，或者虚拟可执行环境（Virtual Execution Environment）。它可以通过强制的自动内存管理，配合强制的类型系统安全性保证，以及数组越界检测等功能，保证 JavaScript 程序在内存分配、访问、释放上的安全性；还可以对代码执行进行权限管理、可见性限制等，保证代码只在其应用的权限内执行。

为了更清楚地理解本申请实施例，首先对与本申请实施例相关的面向对象编程技术所涉及的一些技术术语进行简要说明。

对象：对象中可定义有属性（property）和方法（method），可以将属性和方法（函数）封装在对象中。在内存中，这些对象其实是一些内存块，里面保存了数据和可运行的方法（函数）。

线程对象：是线程类的实例对象。线程对象封装了线程的一些信息（比如包括线程执行的方法（函数）），一个线程对象中的方法（函数）可以被其他线程运行。以 JavaScript 为例，线程对象可通过继承 thread 类或者通过实现 runnable 接口得到，比如从 thread 派生出一个新类，在其中加入属性和方法（函数），并可覆盖 run()方法，即可创建完成一个派生类的新线程对象。其中，run()方法中包含了线程所要执行的代码。

线程：线程是对象中定义的方法的执行路径，也就是说，线程是代码的一次执行过程，在这个执行过程中，线程可以执行代表它的线程对象所定义的方法（函数），也可以执行其它的对象中的方法（函数）。以 JavaScript 为例，当通过继承 thread 类创建一个线程对象后执行 start()方法（函数），从而启动该线程对象对应的线程。

本申请实施例能够使动态语言虚拟机支持多线程，并在动态语言层提供相应配套的多线程开发语言接口。基于本申请实施例提供给开发者语言级别的多线程开发能力，开发者可以使用本申请实施例提供的接口进行多线程程序的开发，如进行线程创建、运行以及通信。本申请实施例通过提供多线程的能力，拓宽了动态语言的应用范围，比如后台事件处理和前台页面渲染分线程处理可以更高效地开发交互式程序；同时由于多线程能更好地利用多核处理器的计算资源，在并发数据处理等大量计算的应用场景，本申请实施例可提高程序的运行性能。

本申请实施例提供了多线程管理方案以实现多线程开发能力，多线程管理具体可包括：线程的创建、线程的同步、线程的销毁等。

本申请实施例可适用于基于单线程机制的动态语言，使基于单线程机制的动态语言实现多线程能力。其中，用于实现线程管理方法的装置可称为线程管理装置。以动态语言为 JavaScript 语言为例，该装置可以是为 JavaScript 虚拟机。

为方便理解，下面首先对本申请实施例的整体框架进行描述。如图 1 所示，线程管理装置（比如 JavaScript 虚拟机）位于动态语言层（比如 JavaScript 语言层）和操作系统之间。动态语言程序由动态语言代码实现，对应于动态语言层，动态语言程序的多线程操作运行于线程管理装置之上，通过线程管理装置调用操作系统中的底层多线程库进行

相应的多线程操作。需要指出的是，线程管理装置运行的操作系统拥有一套底层的线程库。

线程管理装置可对动态语言线程进行管理，图 2 示出了本申请实施例提供的由线程管理装置管理的多线程架构。如图 2 所示，每个线程各自拥有不同的栈（Stack），多个线程可共享一个堆（Heap）。

栈是为执行线程留出的内存空间，每个线程有自己独立的栈和私有数据区，与其他线程隔离。当函数被调用的时候，栈顶为局部变量等数据预留块，当函数执行完毕，该块被释放，在下次的函数调用时再被使用。堆是为动态分配预留的内存空间，所存放的变量可以是对象，该变量为引用类型，即，该变量中实际保存的是一个指针，这个指针指向另一个位置。每个线程都有一个栈，但是每一个应用程序通常只有一个堆（尽管为不同类型分配内存使用多个堆的情况也是有的）。

本申请实施例基于上述架构，提供了多线程管理方案，其中可包括：线程对象的创建、线程对象的释放、线程对象的同步以及除线程对象之外的其它对象的同步等。下面分别对这些过程进行详细描述。

（一）线程的创建

本申请实施例中，在操作系统层中定义有用于创建线程对象的方法（函数），线程管理装置将该方法的调用接口提供给动态语言层，以便开发者调用该接口来创建线程对象。线程管理装置提供给动态语言层的接口具体可以是应用程序编程接口（Application Programming Interface，简称 API），更具体地，该 API 可以包括函数名和参数等信息。线程管理装置可向动态语言层提供多个 API，每个 API 对应相应的方法（函数）。

图 3 示出了本申请实施例中的线程对象的创建过程，如图所示，该流程可包括：

在图 3 的 S301 中：线程管理装置接收动态语言层发送的创建线程对象指令，该创建线程对象指令中包括用于创建线程对象的接口，该线程对象属于指定的应用程序。在一个实施例中，开发者可根据提供给动态语言层的线程对象创建接口编写动态语言应用程序代码，当该应用程序代码被执行时，线程管理装置解析该应用程序代码，得到其中用于创建线程对象的接口，从而通过后续步骤执行线程对象创建方法（函数），创建得到线程对象。

在图 3 的 S302 中：线程管理装置根据该创建线程对象的接口，调用操作系统中该接口对应的对象所提供的用于创建线程对象方法（函数），创建得到对应的线程对象。在一个实施例中，以线程管理装置为 JavaScript 虚拟机为例，JavaScript 虚拟机可根据操作

系统提供的用于创建线程对象的对象（以下为描述方便将该对象称为 obj 对象），通过创建该对象的一个实例，创建得到线程对象。该 obj 对象是操作系统提供的对象，该 obj 对象中封装有属性和方法，比如其中可包含 new 方法。通过针对该 obj 对象执行 new 方法可创建得到新的线程对象，obj 对象中的属性和方法可被新的线程对象所继承，还可向新的线程对象中添加属性和方法。创建得到的线程对象中包括 run 方法（函数），该方法（函数）为线程启动时所要执行的方法（函数），该 run 方法（函数）可由开发者根据所要实现的功能或所要执行的操作进行自定义。

在创建线程对象过程中，线程管理装置创建并注册一个内部的线程对象管理结构来管理对应线程所拥有的资源。该线程对象管理结构可如图 2 所示。其中，新创建得到的线程对象的指针可存储在堆（Heap）中，该线程的变量等数据可存储于该线程的栈（Stack）中。

在图 3 的 S303 中：线程管理装置启动所创建的线程对象，得到对应的线程。

进一步地，线程管理装置在创建线程对象后，可将该线程对象所对应的线程句柄提供给应用程序，供应用程序根据该调用操作对应的线程。在一个实施例中，以线程管理装置为 JavaScript 虚拟机为例，JavaScript 虚拟机可将线程句柄提供给 JavaScript 语言层，以便开发者通过该线程句柄操作对应的线程。线程是在线程对象启动后创建的，一个线程可以有一个或多个句柄，不同的句柄可以具有不同的操作权限。当然，JavaScript 虚拟机也可以将其他能够指示线程的信息提供给 JavaScript 语言层，用于在开发者操作线程时使用。

创建线程对象后，可以通过线程句柄操作对应的线程，如执行线程、等待线程结束、回收线程等。

比如，启动线程的过程可包括：线程管理装置解析动态语言应用程序代码，得到基于线程句柄的启动线程对象的指令，线程管理装置根据该指令，通过调用相应线程对象提供的启动方法（如 start 方法）来启动该线程对象。当调用线程对象的 start 方法时，线程管理装置通过操作系统启动该线程。线程管理装置根据该线程对象提供的运行属性（如 run 属性）的属性值，调用该属性值所指示的方法（该属性的属性值为该方法的名称）。

其中，当线程管理装置根据 run 属性的属性值得到基于动态语言程序代码封装的方法（函数）后，由于操作系统并不直接对接动态语言层，无法解析该方法，因此线程管理装置负责解释该方法，得到操作系统层提供的接口，并调用该接口以执行操作系统提供的相应方法。

再比如，等待线程结束的过程可包括：线程管理装置解析动态语言应用程序代码，得到基于线程句柄的等待线程结束的指令，线程管理装置根据该指令，通过执行线程对象提供的用于等待线程结束的方法（如 join 方法）来等待线程结束。其中，由于操作系统并不直接对接动态语言层，无法解析该 join 方法，因此线程管理装置解释该方法，得到操作系统层提供的接口，并通过调用该接口以执行相应的 join 方法。

再比如，停止线程的过程可包括：线程管理装置解析动态语言应用程序代码，得到基于线程句柄的停止线程的指令，线程管理装置根据该指令，通过执行线程对象中的用于停止线程的方法（如 sleep 方法）来等停止线程。其中，由于操作系统并不直接对接动态语言层，无法解析该 sleep 方法，因此线程管理装置解释并执行该方法。

为了更清楚地说明线程对象的管理过程，下面以 JavaScript 虚拟机为例，结合图 4 进行举例说明。

参见图 4，在 S401 至 S402 中，JavaScript 虚拟机对基于 JavaScript 语言层的 JavaScript 应用程序代码进行解析，得到用于表示调用 obj 对象中的新建实例方法（new 方法）的程序语句，JavaScript 虚拟机根据该程序语句中的 API（如 new 方法的函数名）调用操作系统提供的线程库接口，基于操作系统提供的 obj 对象创建该对象的实例。此过程中，JavaScript 虚拟机将该 obj 对象的属性、方法进行封装，创建得到线程对象，并将线程句柄返回给 JavaScript 语言层。

在 S403 中，JavaScript 虚拟机对基于 JavaScript 语言层的 JavaScript 应用程序代码进行解析，得到用于表示调用对应的线程对象（通过线程句柄来标识对应的线程对象）的启动方法（start 方法）的程序语句时，JavaScript 虚拟机根据该程序语句中的 API（如 start 方法的函数名）调用操作系统提供的线程库接口，通过操作系统创建对应的线程，然后启动该线程。该过程中，JavaScript 虚拟机根据该线程对象从 obj 对象继承的 run 属性的属性值，执行该属性值所指示的方法（该属性的属性值为该方法的名称）。

在 S404 中，JavaScript 虚拟机对基于 JavaScript 语言层的 JavaScript 应用程序代码进行解析，得到用于表示调用对应的线程对象（通过线程句柄来标识对应的线程对象）的等待线程结束方法（join 方法）时，JavaScript 虚拟机根据该程序语句中的 API（如 join 方法的函数名）调用操作系统提供的线程库接口，通过操作系统针对线程句柄所对应的线程执行相应的方法。

在 S405 中，JavaScript 虚拟机对基于 JavaScript 语言层的 JavaScript 应用程序代码进行解析，得到用于表示调用对应的线程对象（通过线程句柄来标识对应的线程对象）的

线程结束方法（sleep 方法）时，JavaScript 虚拟机根据该程序语句中的 API（如 sleep 方法的函数名）调用操作系统提供的线程库接口，通过操作系统针对该线程句柄所对应的线程执行相应的方法。

采用上述实施例提供的方法，可创建多个线程，并可对每个线程进行管理，从而实现多线程管理。

通过以上流程可以看出，线程管理装置从操作系统中封装好线程的能力（包括属性和方法），并将这些能力封装到线程对象中，并将该线程对象的句柄提供给动态语言层。当创建线程时，线程管理装置会通过操作系统创建一个线程对象，然后启动该线程对象得到相应的线程。线程管理装置通过封装线程对象并暴露到动态语言层，从而为开发者提供操作线程的能力。

（二）线程同步

（1）使用同步锁实现线程同步

在多线程系统中，多个线程同时运行时可能调用函数或访问相同的数据，在多个线程同时对同一个内存地址进行写入的情况下，由于 CPU 时间调度上的问题，写入数据会被多次的覆盖。对于一些敏感数据不允许被多个线程同时读写，此时需要使用同步互斥技术，保证数据在任何时刻，最多有一个线程读写，以保证数据的完整性。

线程同步，是指当有一个线程在对内存进行读写操作时，其他线程都不可以对这个内存地址进行读写操作，直到该线程完成读写操作，其他线程才能对该内存地址进行读写操作，而其他线程又处于等待状态。

在本申请的一些实施例中，采用互斥对象机制实现线程同步，只有获得同步锁的线程才有访问公共资源的权限。因为同步锁只有一个，所以能保证公共资源不会同时被多个线程访问。同步锁不仅能实现同一应用程序的公共资源安全共享，还能实现不同应用程序的公共资源安全共享。

上述同步锁是由线程管理装置对动态语言层提供的全局对象，用来进行某一段代码的线程同步。开发者通过构建同步锁，能够实现线程之间执行互斥操作。同步锁的创建过程与线程对象的创建过程类似，可通过调用动态语言层提供的用于创建同步锁的接口，使线程管理装置通过操作系统提供的接口，执行操作系统中的创建同步锁的方法，创建得到同步锁。所创建得到的同步锁为全局对象。线程可以使用此同步锁进行同步，比如线程 1 在对一组全局数据进行读写操作时，若不希望其他线程进行读写，则可以使用此同步锁。

同步锁中可包括如下属性和方法：

锁标识属性：该属性值用于唯一标识同步锁，一个同步锁与一个内存区域对应，不同的内存区域对应不同的同步锁；

加锁方法：获得该线程对象的线程可执行该方法，执行该方法可对锁标识属性的属性值所指示的内存区域进行加锁，被加锁的内存区域只能被获得该线程对象的线程进行读写操作，其他线程不允许对该内存区域进行读写操作；

解锁方法：获得该线程对象的线程可执行该方法，执行该方法可对锁标识属性的属性值所指示的内存区域进行解锁。

图 5 示出了线程 1 和线程 2 使用同步锁实现互斥操作的示意图，如图所示，线程 1 和线程 2 同时需要计算全局数据，开发者希望两个线程不要同时修改该全局数据。开发者就可以使用同步锁进行线程同步。首先，针对请求读写的全局数据所在的内存区域新建一个全局的同步锁。当线程 1 用该同步锁对需要保护的内存区域进行保护，对该内存区域进行读写前先对该内存区域加锁，对该内存区域读写完成后，对该内存区域进行解锁。同理，当线程 2 用该同步锁对需要保护的内存区域进行保护，对该内存区域进行读写前先对该内存区域加锁，对该内存区域读写完成后，对该内存区域进行解锁。

如图 5 所示，在线程 1 需要对该全局数据进行读写时，线程 1 向线程管理装置（如 JavaScript 虚拟机）请求获取该全局数据所在的内存区域的锁对象，线程管理装置判断该内存区域的锁状态，若为未被加锁状态，则线程管理装置将该同步锁给线程 1，线程 1 获得该同步锁后，对该同步锁所对应的内存区域执行加锁方法（lock 方法），该内存区域的状态变为加锁状态，线程 1 对被该线程加锁的该内存区域进行读写操作，读写操作完成后，对该内存区域执行解锁（unlock）方法，该内存区域的状态变为未加锁状态。若线程 1 向线程管理装置请求获取同步锁后，线程管理装置判断该同步锁对应的内存区域当前为加锁状态，说明该内存区域当前正在被其他线程读写，因此不糊将该内存区域对应的同步锁给线程 1，线程 1 处于等待状态。当该内存区域的状态变为未被加锁状态时，线程管理装置将该内存区域对应的同步锁给线程 1。同理，线程 2 基于该同步锁进行数据读写的过程与此类似。

需要说明的是，线程对内存区域进行加锁或解锁的过程中，由于操作系统并不直接对接动态语言层，无法解析加锁方法或解锁方法，因此线程管理装置负责在操作系统创建的线程内解释并执行该方法。另外，线程对内存区域进行读写的过程中，由于操作系统并不直接对接动态语言层，无法解析内存区域读写方法，因此线程管理装置负责在操

作系统创建的线程内解释并执行该方法。

（2）采用同步声明实现线程同步

在本申请的另外一些实施例中，可对方法或动态语言程序语句进行声明，以表明被声明的方法或动态语言程序语句需要同步互斥执行，即，一个线程执行该方法或动态语言程序语句时，其他线程不能执行相同的方法或动态语言程序语句。

以 JavaScript 虚拟机举例来说，如果线程 1 和其他线程（如线程 2）均需要执行方法 a，并且在线程 1 和线程 2 调用方法 a 的语句中，该方法 a 均被声明为进行同步互斥，这样，线程 1 和线程 2 不能同时执行方法 a，只能在一个线程执行完成方法 a 后，另一个线程才能执行方法 a。具体实现过程可如图 6 所示，可包括：

在图 6 的 S601 中：JavaScript 虚拟机接收到线程 1 调用方法 a 的指令；

在图 6 的 S602 中：JavaScript 虚拟机判断当前是否有其他线程正在执行该方法，若有，则转入图 6 中的 S603，否则转入图 6 中的 S604；

在图 6 的 S603 中：JavaScript 虚拟机暂不执行线程 1 中的方法 a，当其他线程（比如线程 2）执行完成方法 a 后，转入图 6 中的 S604：

在图 6 的 S604 中：JavaScript 虚拟机执行线程 1 中的方法 a。

再举例来说，如果线程 1 和其他线程（如线程 2）均需要执行某语句（如对某全局变量进行赋值的操作），并且该语句被声明为进行同步互斥，这样，线程 1 和线程 2 不能同时执行该语句（比如不能同时对该全局变量进行赋值），只能在一个线程执行完成该语句后，另一个线程才能执行该语句。具体实现过程可如图 7 所示，可包括：

在图 7 的 S701 中：JavaScript 虚拟机接收到线程 1 执行该语句的指令；

在图 7 的 S702 中：JavaScript 虚拟机判断当前是否有其他线程正在执行该语句，若有，则转入图 7 中的 703，否则转入图 7 的 S704；

在图 7 的 S703 中：JavaScript 虚拟机暂不执行线程 1 中的该语句，当其他线程（比如线程 2）执行完成该语句后，转入图 7 的 S704：

在图 7 的 S704 中：JavaScript 虚拟机执行线程 1 中的该语句。

声明的方式可以是显式声明也可以是隐式声明。其中，在一些例子中，如果在调用该方法的语句之前和之后增加同步关键字进行声明，则称为显式声明，同样，如果在该动态语言程序语句之前和之后增加同步关键字进行声明，也称为显式声明。在另一些例子中，如果在定义或描述方法的程序代码中，将该方法放入同步关键字的范围之内，这种声明方式称为隐式声明，线程管理装置在通过调用操作系统提供的接口执行该方法时，

操作系统可根据该方法的程序代码中的同步关键字，对该方法的执行过程采用同步互斥操作；同样，如果将该动态语言程序语句放入同步关键字的范围之内，则也称为隐式声明。

（三）对象同步

在多线程系统中，多个线程同时运行时可能需要对同一对象进行读写操作，在多个线程同时对同一个对象进行写入的情况下，由于 CPU 时间调度上的问题，写入数据会被多次的覆盖，导致错误发生。此时需要使用同步互斥技术，保证数据在任何时刻，最多有一个线程对特定的 JavaScript 对象进行读写。

本申请实施例中，线程管理装置对语言层提供了对象同步的能力。可预先指定对某个或某些对象的读写进行同步互斥。采用互斥对象机制实现对象同步，只有获得同步锁的线程才有读写指定的对象的权限。

该同步锁的创建过程与前述实施例类似，可通过调用动态语言层提供的用于创建同步锁的接口，使线程管理装置通过操作系统提供的接口，执行操作系统中的创建同步锁的方法，创建得到同步锁。

同步锁中可包括如下属性和方法：

锁标识属性：该属性值用于唯一标识同步锁，一个同步锁与一个内存区域（这里是一个对象对应的内存区域）对应，不同的对象对应不同的同步锁；

加锁方法：获得该线程对象的线程可执行该方法，执行该方法可对锁标识属性的属性值所指示的对象（或该对象所在的内存区域）进行加锁，被加锁的对象只能被获得该线程对象的线程进行读写操作，其他线程不允许对该对象进行读写操作；

解锁方法：获得该线程对象的线程可执行该方法，执行该方法可对锁标识属性的属性值所指示的对象进行解锁。

本申请实施例中，可以采用以下方式指定某个或某些对象采用同步互斥操作：

方法 1：对象可以被划分为不同类型，本申请实施例中，提供了 ConcurrentArray、ConcurrentSet 和 ConcurrentMap 三种常用对象。该三种对象在内部虚拟机实现上已经保证线程间安全，该三种对象的操作不需要开发者在 JavaScript 层使用同步关键字同步；

方法 2：可使用同步关键字对需要采用同步互斥操作的对象进行声明。

对象的布局由线程管理装置规定。对象的布局可以理解为对象在内存中的存储结构。比如，以对象为 JavaScript 对象为例，如图 8a 所示，JavaScript 对象的布局中可包括对象头部分和对象实例部分，对象头部分中可存储有对象名称等信息，对象实例部分中可存

储对象中的方法使用的变量等。

本申请实施例中，JavaScript 对象的布局头中设置有用于标识同步锁状态的信息域，该信息域可以是一个或多个比特，如图 8b 所示。该信息域的不同取值可标识该 JavaScript 对象的锁状态。例如，如果该信息域的取值为 0 时，表示该 JavaScript 对象未加锁，该信息域的取值为 1 时，表示该 JavaScript 对象已经加锁。

参见图 9 所示，当线程管理装置在接收到线程 1 对某对象进行读写的指令时（图 9 中的 S901），可判断该对象是否是被指定为进行同步互斥的对象（图 9 中的 S902），若是，则请求获取同步锁（图 9 中的 S903），若获取到同步锁，则将该同步锁给线程 1，线程 1 收到同步锁后，将对该对象加锁，比如设置该对象的布局中的锁状态标志位取值为 1，并对该对象进行读写操作，并在读写完成后，对该对象进行解锁，比如设置该对象的布局中的锁状态标志位取值为 0（图 9 中的 S904），若未获取到同步锁，则说明当前该对象正在被读写，此种情况下，线程管理装置等待该对象读写完成后，将同步锁给线程 1，线程 1 获取到同步锁之后的操作同前所述（图 9 中的 S905）；若该对象未被指定为进行同步互斥，则对该对象进行读写操作（图 9 中的 S906）。

需要说明的是，线程对对象进行加锁或解锁的过程中，由于操作系统并不直接对接动态语言层，无法解析加锁方法或解锁方法，因此线程管理装置负责在操作系统创建的线程内解释并执行该方法。另外，线程对对象进行读写的过程中，由于操作系统并不直接对接动态语言层，无法解析内存区域读写方法，因此线程管理装置负责在操作系统创建的线程内解释并执行该方法。

基于相同的技术构思，本申请实施例还提供了一种线程管理装置，该线程管理装置具体可以是前述实施例中的动态语言虚拟机，更具体地，可以是 JavaScript 虚拟机。

参见图 10，为本申请实施例提供的线程管理装置的结构示意图，如图所示，该装置可包括：接收模块 1001、管理模块 1002，其中：

接收模块 1001，用于接收动态语言层发送的创建线程对象指令，所述创建线程对象指令中包括用于创建线程对象的接口，所述线程对象属于指定的应用程序；

管理模块 1102，用于根据所述用于创建线程对象的接口，调用操作系统中所述接口对应的对象所提供的用于创建线程对象的方法，创建得到对应的线程对象；并启动所述线程对象，得到对应的线程。

可选地，所述应用程序中对应多个线程，所述多个线程中的每个线程拥有各自独立的栈以及私有数据区，所述多个线程共享一个堆；其中，一个线程的栈是该线程所占用

的内存空间，所述堆是所述应用程序的所有线程共享的内存空间。

可选地，所述接收模块还用于：接收动态语言层发送的对全局数据进行读写的指令；管理模块 1002 可还用于：请求获得用于对所述全局变量进行读写的第一同步锁，所述第一同步锁为全局对象；调用获得到的第一同步锁的加锁方法对所述全局变量进行加锁，对所述全局变量进行读写，并在读写完成后调用所述第一同步锁的解锁方法对所述全局变量进行解锁。

可选地，接收模块 1001 还用于：接收动态语言层发送的执行指定对象提供的指定方法的指令。管理模块 1002 还用于：确定所述指定方法是否被声明为进行同步互斥；若是，则请求获得所述指定方法对应的第二同步锁，所述第二同步锁为全局对象；调用获得到的所述第二同步锁的加锁方法对所述指定方法进行加锁，执行所述指定方法，并在执行完成后调用所述第二同步锁的解锁方法对所述指定方法进行解锁。

可选地，接收模块 1001 还用于：接收用于对目标对象进行读写的指令.管理模块 1002 还用于：若所述目标对象所属的类型为指定类型或被声明为进行同步互斥，则请求获得对所述目标对象进行读写的第三同步锁，所述第三同步锁为全局对象；调用获得到的所述第三同步锁的加锁方法对所述目标对象进行加锁，对所述目标对象进行读写，并在读写完成后调用所述第三同步锁的解锁方法对所述目标对象进行解锁。

可选地，管理模块 1002 具体用于：获取所述目标对象所在的内存区域中用于存储对象头信息的内存区域中的对象状态标志位，若所述对象状态标志位的取值表明所述目标对象未被加锁，则获得用于对所述目标对象进行读写的第三同步锁。可选地，管理模块 1002 具体用于：对所述目标对象进行加锁时，设置所述状态标志位的取值以表明所述目标对象已被加锁.。可选地，管理模块 1002 具体用于：对所述目标对象进行解锁时，设置所述状态标志位的取值以表明所述目标对象未被加锁。

可选地，所述管理模块还用于：将创建得到的线程对象所对应的句柄提供给所述动态语言层。

可选地，管理模块 1002 具体用于：接收动态语言层发送的启动所述线程对象的指令，所述启动线程对象的指令中包括所述线程对象对应的句柄；根据所述启动线程对象的指令，通过操作系统调用所述句柄对应的线程对象提供的启动方法，所述启动方法用于根据所述线程对象的运行属性的属性值，执行该属性值所指示的方法。

可选地，接收模块 1001 还用于：接收动态语言层发送的等待线程结束的指令，所述等待线程结束的指令中包括所述线程对象所对应的句柄。管理模块 1002 还用于：根据接

收到的等待线程结束的指令，通过操作系统调用所述句柄对应的线程对象提供的用于等待线程结束的方法来等待线程运行结束。

可选地，接收模块 1001 还用于：接收动态语言层发送的停止线程的指令，所述停止线程的指令中包括所述线程对象所对应的句柄。管理模块 1002 还用于：根据接收到的停止线程的指令，通过操作系统调用所述句柄对应的线程对象提供的用于停止线程的方法来停止线程运行。

基于相同的技术构思，本申请实施例还提供了一个或多个计算机可读介质，所述可读介质上存储有指令，所述指令被一个或多个处理器执行时，使得通信设备执行前述实施例描述的线程管理方法。

基于相同的技术构思，本申请实施例还提供了一种装置 1100，该装置 1100 可实现前述实施例描述的流程。

图 11 示例性地示出了根据各种实施例的示例装置 1100，装置 1100 可包括一个或多个处理器 1102，系统控制逻辑 1101 耦合于至少一个处理器 1102，非易失性存储器（non-volatile memory，NMV）/存储器 1104 耦合于系统控制逻辑 1101，网络接口 1106 耦合于系统控制逻辑 1101。

处理器 1102 可包括一个或多个单核处理器或多核处理器。处理器 1102 可包括任何一般用途处理器或专用处理器（如图像处理器、应用处理器基带处理器等）的组合。

一个实施例中的系统控制逻辑 1101，可包括任何适当的接口控制器，以提供到处理器 1102 中的至少一个的任何合适的接口，和/或提供到与系统控制逻辑 1101 通信的任何合适的设备或组件的任何合适的接口。

一个实施例中的系统控制逻辑 1101，可包括一个或多个内存控制器，以提供到系统内存 1103 的接口。系统内存 1103 用来加载以及存储数据和/或指令。例如，对应装置 1100，在一个实施例中，系统内存 1103 可包括任何合适的易失性存储器。

NVM/存储器 1104 可包括一个或多个有形的非暂时的计算机可读介质，用于存储数据和/或指令。例如，NVM/存储器 1104 可包括任何合适的非易失性存储装置，如一个或多个硬盘（hard disk device，HDD），一个或多个光盘（compact disk，CD），和/或一个或多个数字通用盘（digital versatile disk，DVD）。

NVM/存储器 1104 可包括存储资源，该存储资源物理上是该系统所安装的或者可以被访问的设备的一部分，但不一定是设备的一部分。例如，NVM/存储器 1104 可经由网络接口 1106 被网络访问。

系统内存1103以及NVM/存储器1104可分别包括临时的或持久的指令1110的副本。指令1110可包括当由处理器1102中的至少一个执行时导致装置1100实现图3至图7、图9描述的方法之一或组合的指令。各实施例中，指令1110或硬件、固件，和/或软件组件可另外地/可替换地被置于系统控制逻辑1101，网络接口1106和/或处理器1102。

网络接口1106可包括一个接收器来为装置1100提供无线接口来与一个或多个网络和/或任何合适的设备进行通信。网络接口1106可包括任何合适的硬件和/或固件。网络接口1106可包括多个天线来提供多输入多输出无线接口。在一个实施例中，网络接口1106可包括一个网络适配器、一个无线网络适配器、一个电话调制解调器，和/或无线调制解调器。

在一个实施例中，处理器1102中的至少一个可以与用于系统控制逻辑的一个或多个控制器的逻辑一起封装。在一个实施例中，处理器中的至少一个可以与用于系统控制逻辑的一个或多个控制器的逻辑一起封装以形成系统级封装。在一个实施例中，处理器中的至少一个可以与用于系统控制逻辑的一个或多个控制器的逻辑集成在相同的管芯上。在一个实施例中，处理器中的至少一个可以与用于系统控制逻辑的一个或多个控制器的逻辑集成在相同的管芯上以形成系统芯片。

装置1100可进一步包括输入/输出装置1105。输入/输出装置1105可包括用户接口旨在使用户与装置1100进行交互，可包括外围组件接口，其被设计为使得外围组件能够与系统交互，和/或，可包括传感器，旨在确定环境条件和/或有关装置1100的位置信息。

权 利 要 求 书

1.一种线程管理方法，其特征在于，包括：

接收动态语言层发送的创建线程对象指令，所述创建线程对象指令中包括用于创建线程对象的接口，所述线程对象对应指定的应用程序；

根据所述用于创建线程对象的接口，调用操作系统中所述接口对应的对象所提供的用于创建线程对象的方法，创建得到对应的线程对象；

启动所述线程对象，得到对应的线程。

2.如权利要求 1 所述的方法，其特征在于，所述应用程序中对应多个线程，所述多个线程中的每个线程拥有各自独立的栈以及私有数据区，所述多个线程共享一个堆；其中，一个线程的栈为该线程所占用的内存空间，所述堆为所述应用程序的所有线程共享的内存空间。

3.如权利要求 1 所述的方法，其特征在于，还包括：

接收动态语言层发送的对全局数据进行读写的指令；

请求获得用于对所述全局变量进行读写的第一同步锁，所述第一同步锁为全局对象；

调用获得到的第一同步锁的加锁方法对所述全局变量进行加锁，对所述全局变量进行读写，并在读写完成后调用所述第一同步锁的解锁方法对所述全局变量进行解锁。

4.如权利要求 1 所述的方法，其特征在于，还包括：

接收动态语言层发送的执行指定对象提供的指定方法的指令；

确定所述指定方法是否被声明为进行同步互斥；

若是，则请求获得所述指定方法对应的第二同步锁，所述第二同步锁为全局对象；调用获得到的所述第二同步锁的加锁方法对所述指定方法进行加锁，执行所述指定方法，并在执行完成后调用所述第二同步锁的解锁方法对所述指定方法进行解锁。

5.如权利要求 1 所述的方法，其特征在于，还包括：

接收用于对目标对象进行读写的指令；

若所述目标对象所属的类型为指定类型或被声明为进行同步互斥，则请求获得对所述目标对象进行读写的第三同步锁，所述第三同步锁为全局对象；

调用获得到的所述第三同步锁的加锁方法对所述目标对象进行加锁，对所述目标对象进行读写，并在读写完成后调用所述第三同步锁的解锁方法对所述目标对象进行解锁。

6.如权利要求 5 所述的方法，其特征在于，请求获得对所述目标对象进行读写的第三同步锁，包括：

获取所述目标对象所在的内存区域中用于存储对象头信息的内存区域中的对象状态标志位，若所述对象状态标志位的取值表明所述目标对象未被加锁，则获得用于对所述目标对象进行读写的第三同步锁；

对所述目标对象进行加锁，包括：

设置所述状态标志位的取值以表明所述目标对象已被加锁；

对所述目标对象进行解锁，包括：

设置所述状态标志位的取值以表明所述目标对象未被加锁。

7.如权利要求 1 至 6 中任一项所述的方法，其特征在于，还包括：

将创建得到的线程对象所对应的句柄提供给所述动态语言层。

8.如权利要求 1 至 6 中任一项所述的方法，其特征在于，启动所述线程对象，包括：

接收动态语言层发送的启动所述线程对象的指令，所述启动线程对象的指令中包括所述线程对象对应的句柄；

根据所述启动线程对象的指令，通过操作系统调用所述句柄对应的线程对象提供的启动方法，所述启动方法用于根据所述线程对象的运行属性的属性值，执行该属性值所指示的方法。

9.如权利要求 1 所述的方法，其特征在于，还包括：

接收动态语言层发送的等待线程结束的指令，所述等待线程结束的指令中包括所述线程对象所对应的句柄；

根据接收到的等待线程结束的指令，通过操作系统调用所述句柄对应的线程对象提供的用于等待线程结束的方法来等待线程运行结束。

10.如权利要求 1 所述的方法，其特征在于，还包括：

接收动态语言层发送的停止线程的指令，所述停止线程的指令中包括所述线程对象所对应的句柄；

根据接收到的停止线程的指令，通过操作系统调用所述句柄对应的线程对象提供的用于停止线程的方法来停止线程运行。

11.如权利要求 1 至 6 中任一项所述的方法，其特征在于，所述接收、所述调用和所述启动的操作的执行主体为 JavaScript 虚拟机，所述动态语言层为 JavaScript 语言层。

12.一种线程管理装置，其特征在于，包括：

接收模块，用于接收动态语言层发送的创建线程对象指令，所述创建线程对象指令中包括用于创建线程对象的接口，所述线程对象属于指定的应用程序；

管理模块，用于根据所述用于创建线程对象的接口，调用操作系统中所述接口对应的对象所提供的用于创建线程对象的方法，创建得到对应的线程对象；并启动所述线程对象，得到对应的线程。

13.如权利要求 12 所述的装置，其特征在于，所述应用程序中对应多个线程，所述多个线程中的每个线程拥有各自独立的栈以及私有数据区，所述多个线程共享一个堆；其中，一个线程的栈为该线程所占用的内存空间，所述堆为所述应用程序的所有线程共享的内存空间。

14.如权利要求 12 所述的装置，其特征在于，所述接收模块还用于：接收动态语言层发送的对全局数据进行读写的指令；

所述管理模块还用于：请求获得用于对所述全局变量进行读写的第一同步锁，所述第一同步锁为全局对象；调用获得到的第一同步锁的加锁方法对所述全局变量进行加锁，对所述全局变量进行读写，并在读写完成后调用所述第一同步锁的解锁方法对所述全局变量进行解锁。

15.如权利要求 12 所述的装置，其特征在于，所述接收模块还用于：接收动态语言层发送的执行指定对象提供的指定方法的指令；

所述管理模块还用于：确定所述指定方法是否被声明为进行同步互斥；若是，则请求获得所述指定方法对应的第二同步锁，所述第二同步锁为全局对象；调用获得到的所述第二同步锁的加锁方法对所述指定方法进行加锁，执行所述指定方法，并在执行完成后调用所述第二同步锁的解锁方法对所述指定方法进行解锁。

16.如权利要求 12 所述的装置，其特征在于，所述接收模块还用于：接收用于对目标对象进行读写的指令；

所述管理模块还用于：若所述目标对象所属的类型为指定类型或被声明为进行同步互斥，则请求获得对所述目标对象进行读写的第三同步锁，所述第三同步锁为全局对象；调用获得到的所述第三同步锁的加锁方法对所述目标对象进行加锁，对所述目标对象进行读写，并在读写完成后调用所述第三同步锁的解锁方法对所述目标对象进行解锁。

17.如权利要求 16 所述的装置，其特征在于，所述管理模块具体用于：获取所述目标对象所在的内存区域中用于存储对象头信息的内存区域中的对象状态标志位，若所述对象状态标志位的取值表明所述目标对象未被加锁，则获得用于对所述目标对象进行读

写的第三同步锁；

所述管理模块具体用于：对所述目标对象进行加锁时，设置所述状态标志位的取值以表明所述目标对象已被加锁；

所述管理模块具体用于：对所述目标对象进行解锁时，设置所述状态标志位的取值以表明所述目标对象未被加锁。

18.如权利要求 12 至 17 中任一项所述的装置，其特征在于，所述管理模块还用于：将创建得到的线程对象所对应的句柄提供给所述动态语言层。

19.如权利要求 12 至 17 中任一项所述的装置，其特征在于，所述管理模块具体用于：接收动态语言层发送的启动所述线程对象的指令，所述启动线程对象的指令中包括所述线程对象对应的句柄；根据所述启动线程对象的指令，通过操作系统调用所述句柄对应的线程对象提供的启动方法，所述启动方法用于根据所述线程对象的运行属性的属性值，执行该属性值所指示的方法。

20.如权利要求 12 所述的装置，其特征在于，所述接收模块还用于：接收动态语言层发送的等待线程结束的指令，所述等待线程结束的指令中包括所述线程对象所对应的句柄；

所述管理模块还用于：根据接收到的等待线程结束的指令，通过操作系统调用所述句柄对应的线程对象提供的用于等待线程结束的方法来等待线程运行结束。

21.如权利要求 12 所述的装置，其特征在于，所述接收模块还用于：接收动态语言层发送的停止线程的指令，所述停止线程的指令中包括所述线程对象所对应的句柄；

所述管理模块还用于：根据接收到的停止线程的指令，通过操作系统调用所述句柄对应的线程对象提供的用于停止线程的方法来停止线程运行。

22.如权利要求 12 至 17 中任一项所述的装置，其特征在于，所述装置为 JavaScript 虚拟机，所述动态语言层为 JavaScript 语言层。

23.一个或多个计算机可读介质，所述可读介质上存储有指令，所述指令被一个或多个处理器执行时，使得通信设备执行如权利要求 1-11 中任一项所述的方法。

24.一种线程管理装置，包括：

一个或多个处理器；以及

一个或多个计算机可读介质，所述可读介质上存储有指令，所述指令被所述一个或多个处理器执行时，使得所述装置执行如权利要求 1-11 中任一项所述的方法。

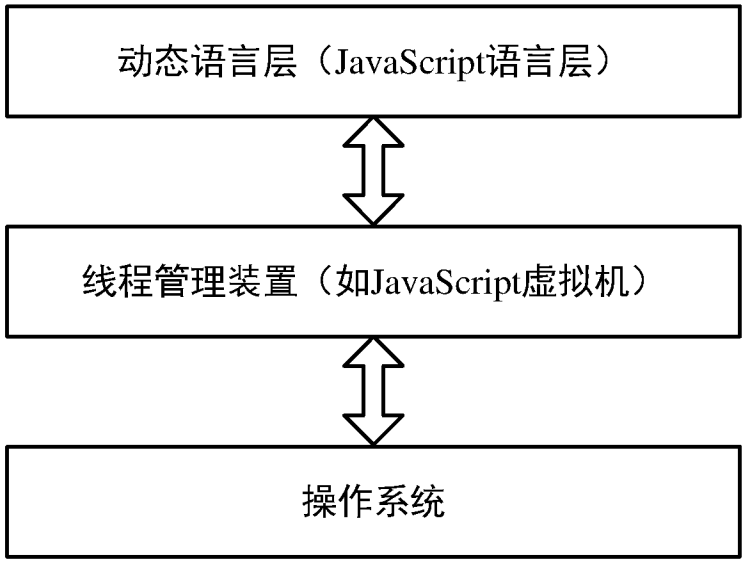
动态语言层（JavaScript语言层）
线程管理装置（如JavaScript虚拟机）
操作系统

图 1

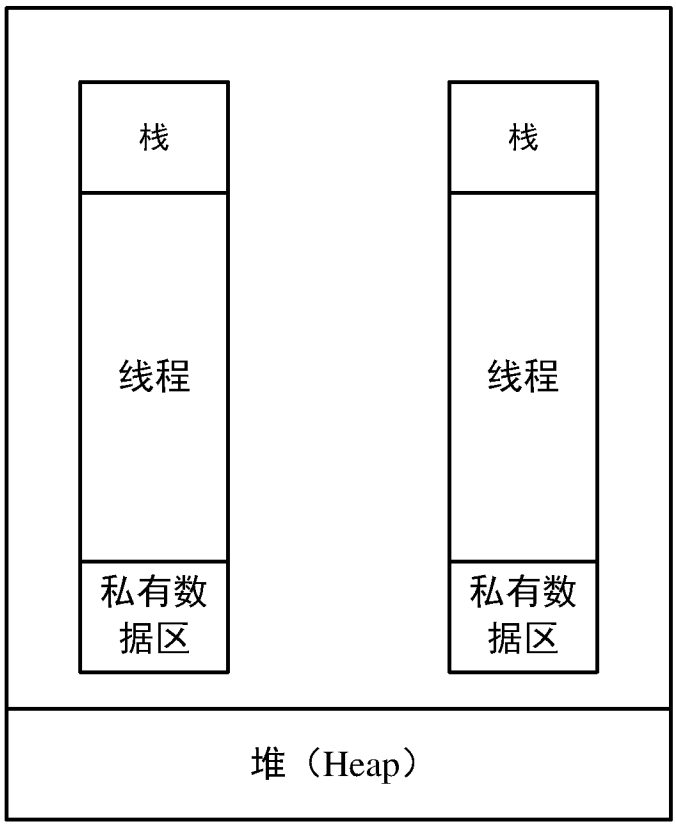
栈
线程
私有数据区
栈
线程
私有数据区
堆（Heap）

图 2

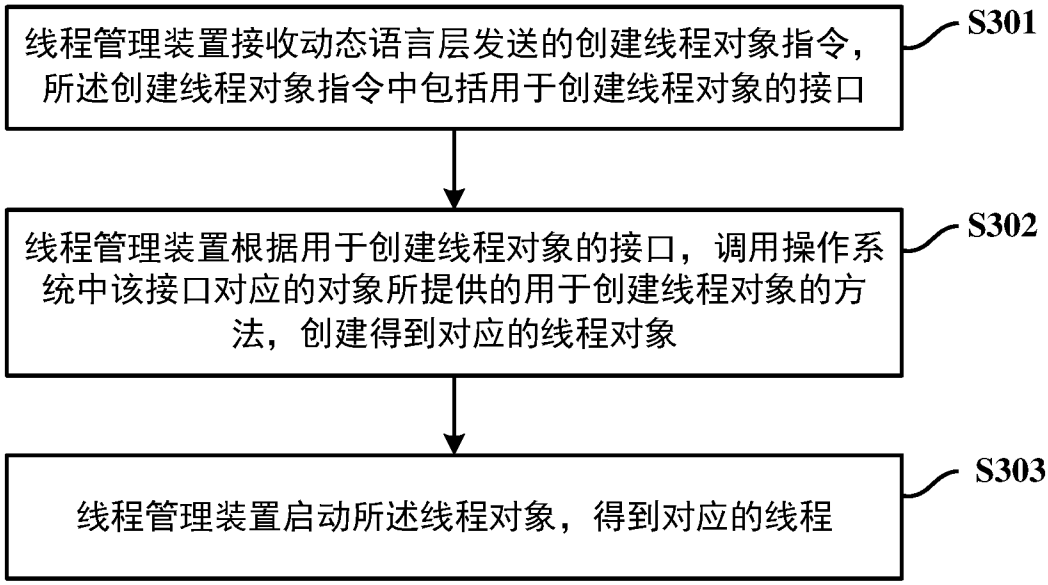
线程管理装置接收动态语言层发送的创建线程对象指令，所述创建线程对象指令中包括用于创建线程对象的接口
S301
线程管理装置根据用于创建线程对象的接口，调用操作系统中该接口对应的对象所提供的用于创建线程对象的方法，创建得到对应的线程对象
S302
线程管理装置启动所述线程对象，得到对应的线程
S303

图 3

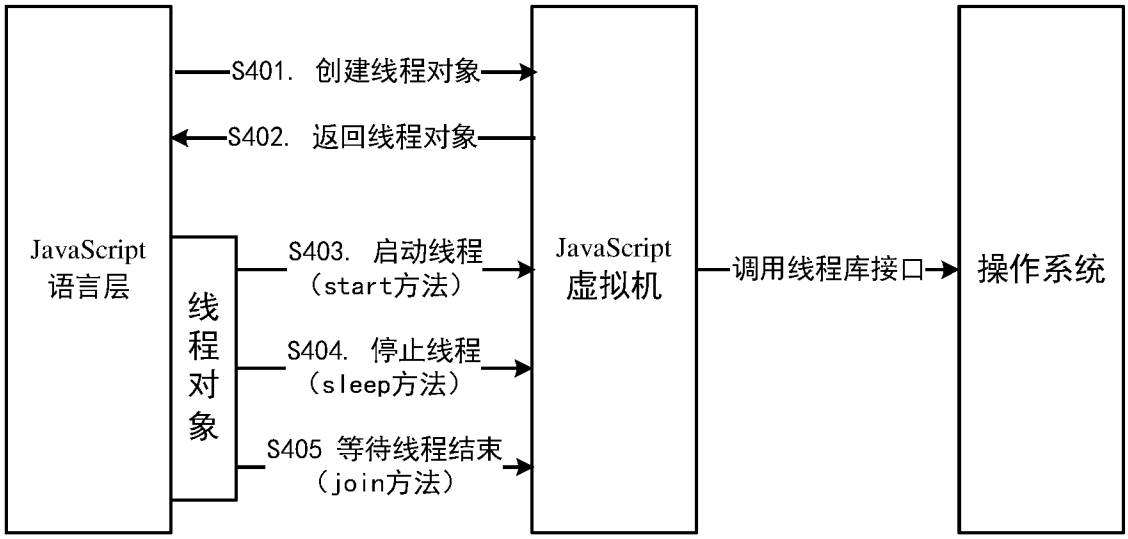
JavaScript 语言层
线程对象
S401. 创建线程对象
S402. 返回线程对象
S403. 启动线程（start方法）
S404. 停止线程（sleep方法）
S405 等待线程结束（join方法）
JavaScript 虚拟机
调用线程库接口
操作系统

图 4

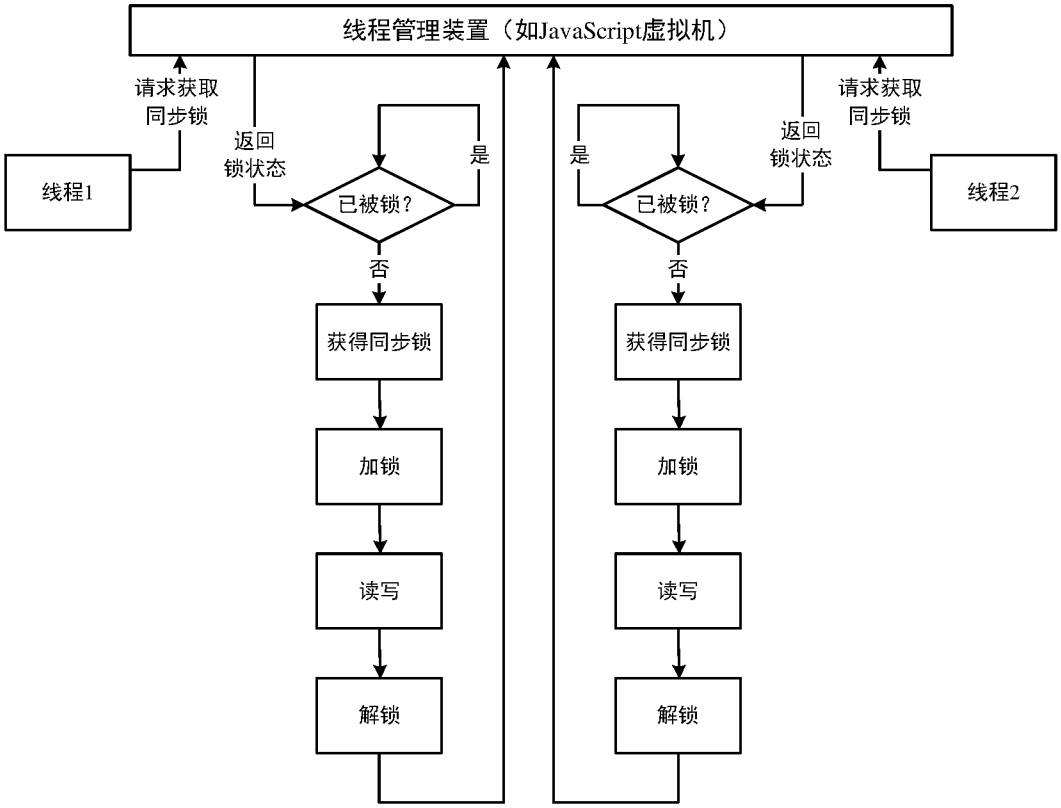
线程管理装置（如JavaScript虚拟机）
请求获取同步锁
返回锁状态
线程1
已被锁？
是
否
获得同步锁
加锁
读写
解锁
请求获取同步锁
返回锁状态
线程2
已被锁？
是
否
获得同步锁
加锁
读写
解锁

图 5

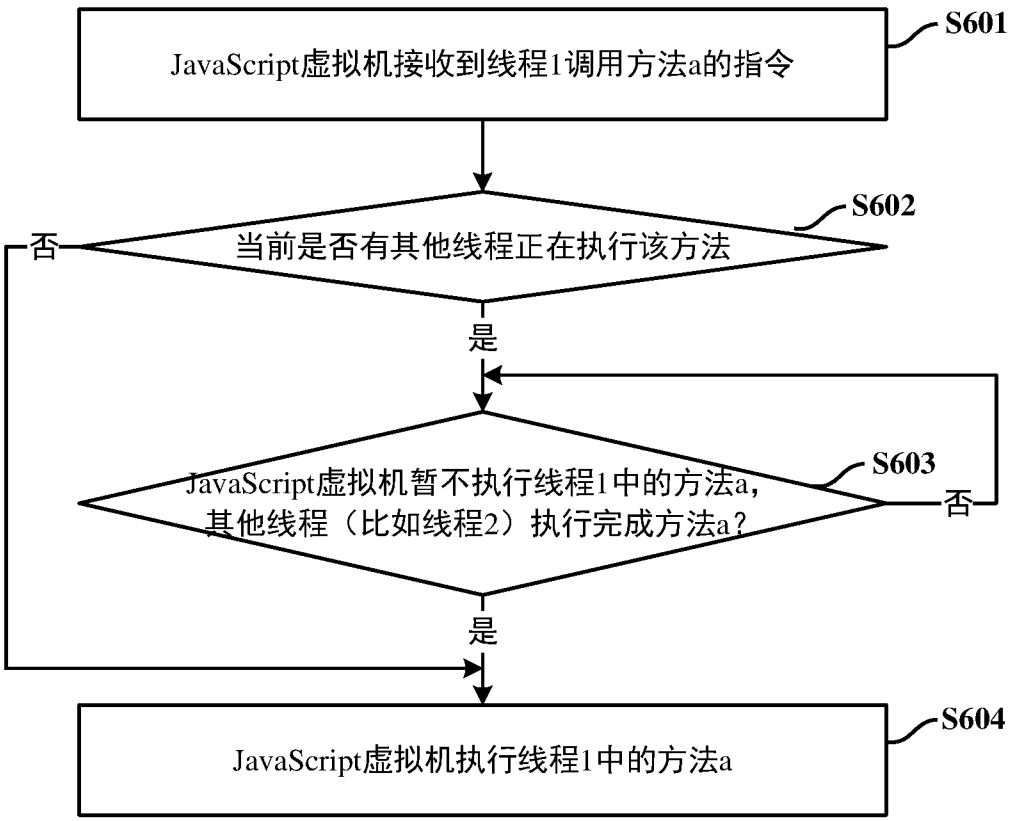
JavaScript虚拟机接收到线程1调用方法a的指令
S601
当前是否有其他线程正在执行该方法
S602
否
是
JavaScript虚拟机暂不执行线程1中的方法a，
其他线程（比如线程2）执行完成方法a？
S603
否
是
JavaScript虚拟机执行线程1中的方法a
S604

图 6

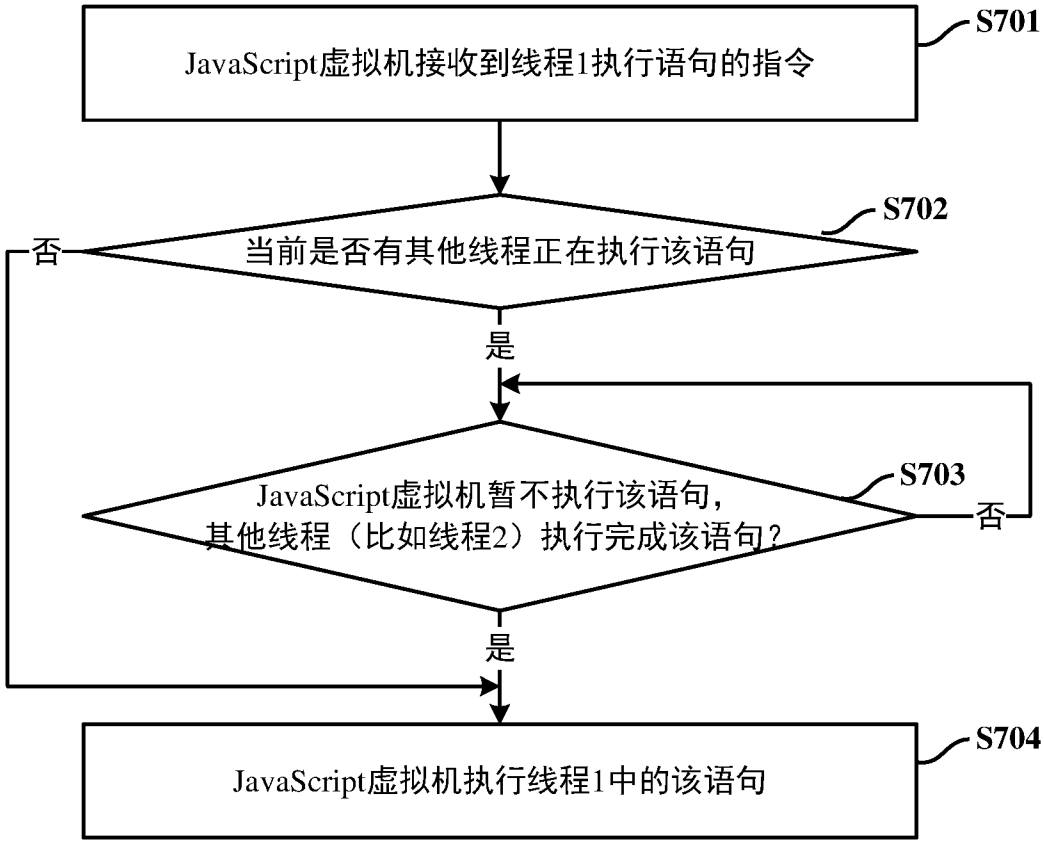
JavaScript虚拟机接收到线程1执行语句的指令
S701
当前是否有其他线程正在执行该语句
S702
否
是
JavaScript虚拟机暂不执行该语句，
其他线程（比如线程2）执行完成该语句？
S703
否
是
JavaScript虚拟机执行线程1中的该语句
S704

图 7


JavaScript对象头
JavaScript对象实例

图 8a

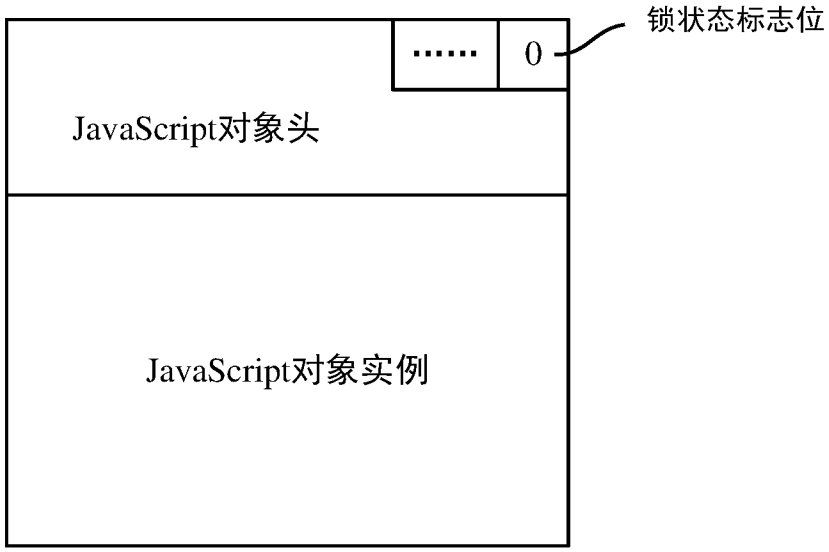
锁状态标志位
.....
0
JavaScript对象头
JavaScript对象实例

图 8b

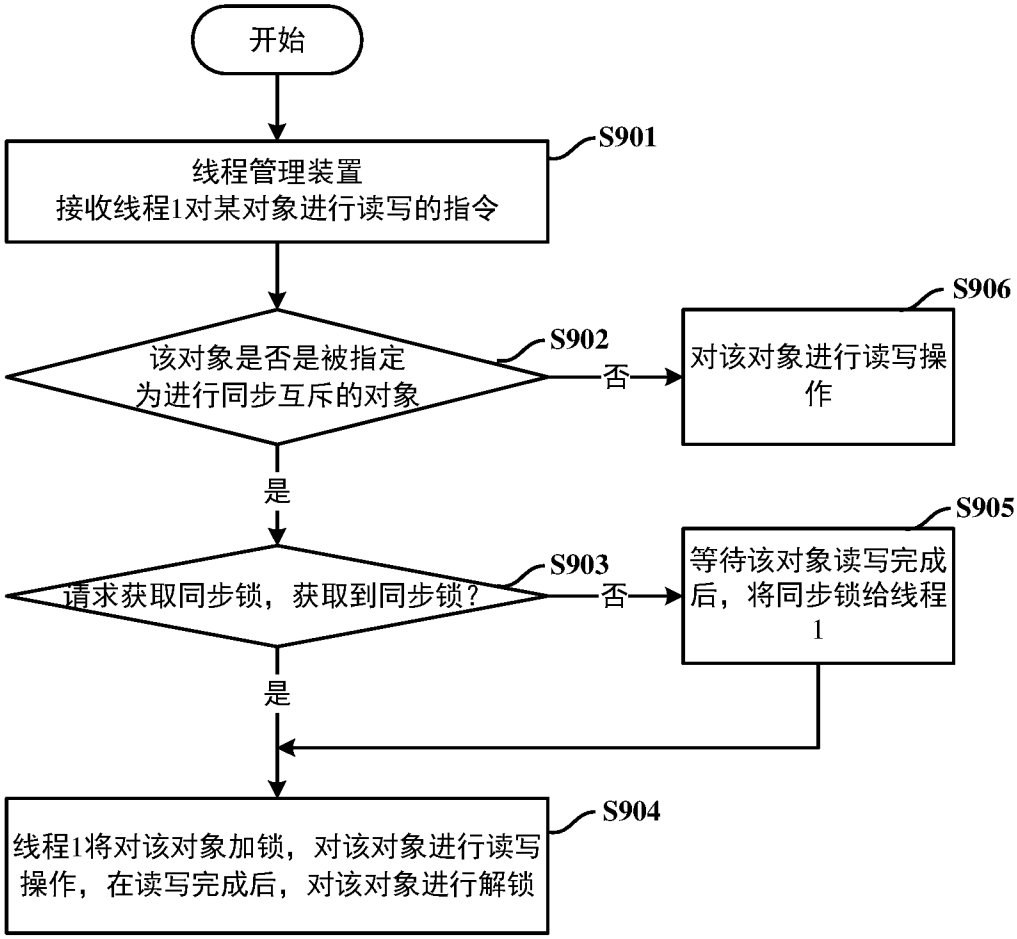
开始
线程管理装置
接收线程1对某对象进行读写的指令
S901
该对象是否是被指定
为进行同步互斥的对象
S902
否
对该对象进行读写操作
S906
是
请求获取同步锁，获取到同步锁？
S903
否
等待该对象读写完成后，将同步锁给线程1
S905
是
线程1将对该对象加锁，对该对象进行读写操作，在读写完成后，对该对象进行解锁
S904

图 9

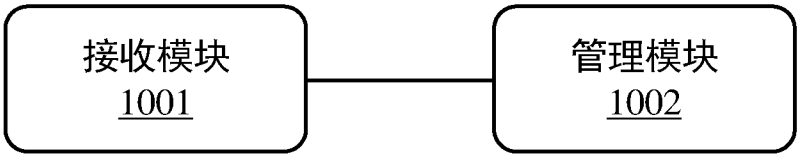
接收模块
1001
管理模块
1002

图 10

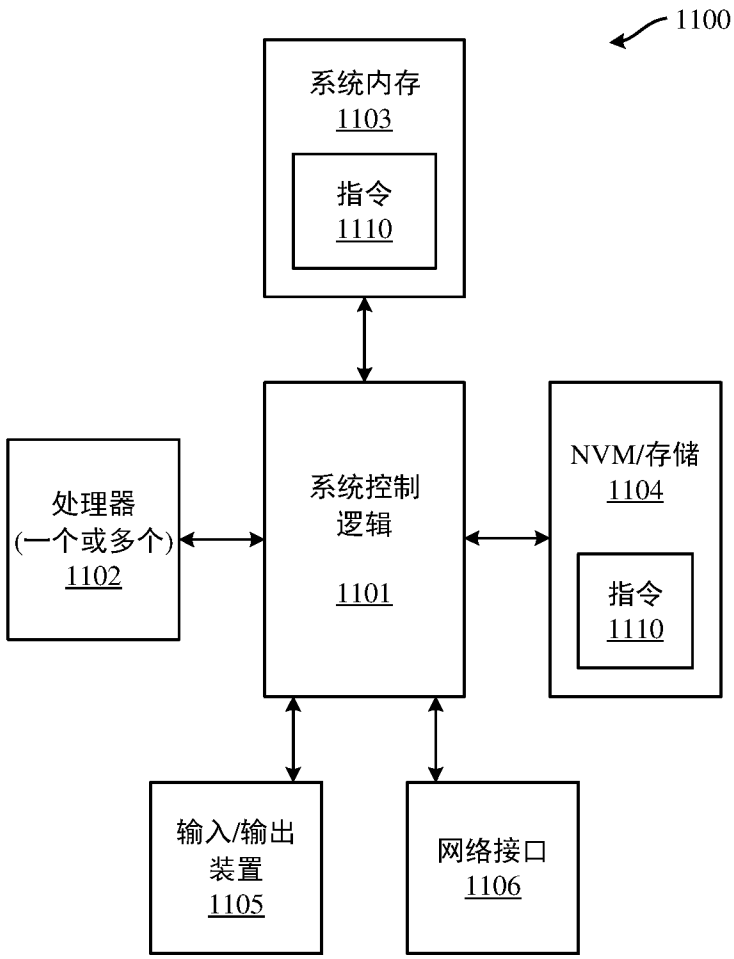
1100
系统内存
1103
指令
1110
处理器
(一个或多个)
1102
系统控制
逻辑
1101
NVM/存储
1104
指令
1110
输入/输出
装置
1105
网络接口
1106

图 11

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2018/072039

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/44 (2018.01) i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNKI, CNPAT, WPI, EPODOC: 线程, 动态语言, 创建, 启动, 调用, 接口, 对象, 应用程序, 栈, 堆, 锁, 句柄, Java, API, thread, creat, start, call, interface, object, stack, heap, application

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	CN 105373414 A (LOONGSON TECHNOLOGY CO., LTD.) 02 March 2016 (02.03.2016), description, paragraphs [0047], [0055]-[0071] and [0093], and figure 1	1-24
A	CN 103744723 A (SHENZHEN LAN-YOU TECHNOLOGY CO., LTD.) 23 April 2014 (23.04.2014), entire document	1-24
A	CN 104850460 A (SHANGHAI FEIXUN COMMUNICATION CO., LTD.) 19 August 2015 (19.08.2015), entire document	1-24
A	CN 101421711 A (MICROSOFT CORPORATION) 29 April 2009 (29.04.2009), entire document	1-24

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 March 2018	09 April 2018
Name and mailing address of the ISA State Intellectual Property Office of the P. R. China No. 6, Xitucheng Road, Jimenqiao Haidian District, Beijing 100088, China Facsimile No. (86-10) 62019451	Authorized officer LIU, Yuru Telephone No. (86-10) 53961294

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2018/072039

Patent Documents referred in the Report	Publication Date	Patent Family	Publication Date
CN 105373414 A	02 March 2016	None	
CN 103744723 A	23 April 2014	None	
CN 104850460 A	19 August 2015	None	
CN 101421711 A	29 April 2009	US 2009307308 A1	10 December 2009
		KR 20080112269 A	24 December 2008
		DE 602006014360 D1	01 July 2010
		EP 1845444 A1	17 October 2007
		WO 2007120436 A1	25 October 2007
		AT 468556 T	15 June 2010

国际检索报告

国际申请号
PCT/CN2018/072039

A. 主题的分类

G06F 9/44(2018.01)i

按照国际专利分类(IPC)或者同时按照国家分类和IPC两种分类

B. 检索领域

检索的最低限度文献(标明分类系统和分类号)

G06F

包含在检索领域中的除最低限度文献以外的检索文献

在国际检索时查阅的电子数据库(数据库的名称，和使用的检索词（如使用））

CNKI,CNPAT,WPI,EPODOC：线程，动态语言，创建，启动，调用，接口，对象，应用程序，栈，堆，锁，句柄，Java, API, thread, creat, start, call, interface, object, stack, heap, application

C. 相关文件

类 型*	引用文件，必要时，指明相关段落	相关的权利要求
X	CN 105373414 A (龙芯中科技术有限公司) 2016年 3月 2日 (2016-03-02) 说明书第[0047]、[0055]-[0071]、[0093]段，图1	1-24
A	CN 103744723 A (深圳联友科技有限公司) 2014年 4月 23日 (2014-04-23) 全文	1-24
A	CN 104850460 A (上海斐讯数据通信技术有限公司) 2015年 8月 19日 (2015-08-19) 全文	1-24
A	CN 101421711 A (微软公司) 2009年 4月 29日 (2009-04-29) 全文	1-24

☐ 其余文件在C栏的续页中列出。

☑ 见同族专利附件。

* 引用文件的具体类型：

“A” 认为不特别相关的表示了现有技术一般状态的文件

“E” 在国际申请日的当天或之后公布的在先申请或专利

“L” 可能对优先权要求构成怀疑的文件，或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件（如具体说明的）

“O” 涉及口头公开、使用、展览或其他方式公开的文件

“P” 公布日先于国际申请日但迟于所要求的优先权日的文件

“T” 在申请日或优先权日之后公布，与申请不相抵触，但为了理解发明之理论或原理的在后文件

“X” 特别相关的文件，单独考虑该文件，认定要求保护的发明不是新颖的或不具有创造性

“Y” 特别相关的文件，当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时，要求保护的发明不具有创造性

“&” 同族专利的文件

国际检索实际完成的日期	国际检索报告邮寄日期
2018年 3月 15日	2018年 4月 9日
ISA/CN的名称和邮寄地址 中华人民共和国国家知识产权局(ISA/CN) 中国北京市海淀区蓟门桥西土城路6号 100088 传真号 (86-10)62019451	受权官员 刘宇儒 电话号码 (86-10)53961294

国际检索报告
关于同族专利的信息

国际申请号
PCT/CN2018/072039

检索报告引用的专利文件			公布日（年/月/日）	同族专利			公布日（年/月/日）
CN	105373414	A	2016年 3月 2日	无			
CN	103744723	A	2014年 4月 23日	无			
CN	104850460	A	2015年 8月 19日	无			
CN	101421711	A	2009年 4月 29日	US	2009307308	A1	2009年 12月 10日
				KR	20080112269	A	2008年 12月 24日
				DE	602006014360	D1	2010年 7月 1日
				EP	1845444	A1	2007年 10月 17日
				WO	2007120436	A1	2007年 10月 25日
				AT	468556	T	2010年 6月 15日