

(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
2020 年 10 月 15 日 (15.10.2020)



(10) 国际公布号
WO 2020/207354 A1

(51) 国际专利分类号:
G06F 9/48 (2006.01) **G06F 9/54** (2006.01)

(21) 国际申请号: PCT/CN2020/083330

(22) 国际申请日: 2020 年 4 月 3 日 (03.04.2020)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
201910291330.3 2019 年 4 月 11 日 (11.04.2019) CN

(71) 申请人: 华为技术有限公司 (HUAWEI TECHNOLOGIES CO., LTD.) [CN/CN]; 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。

(72) 发明人: 万承臻 (WAN, Chengzhen); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。 李刚 (LI, Gang); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。 曾广珠 (ZENG, Guangzhu); 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。

(74) 代理人: 北京三高永信知识产权代理有限公司 (BEIJING SAN GAO YONG XIN INTELLECTUAL PROPERTY AGENCY CO., LTD.); 中国北京市海淀区学院路蓟门里和景园 A 座 1 单元 102 室, Beijing 100088 (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,

(54) Title: TASK PROCESSING METHOD, APPARATUS, TERMINAL, AND COMPUTER READABLE STORAGE MEDIUM

(54) 发明名称: 任务处理方法、装置、终端以及计算机可读存储介质

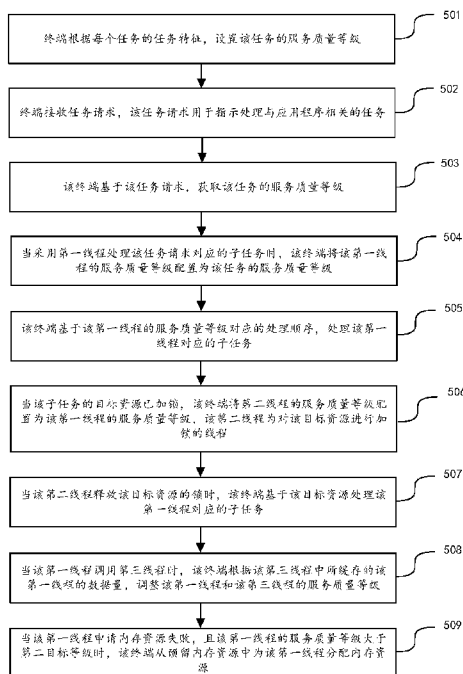


图 5

501 A terminal configures, according to task features of each task, a quality of service level of the task

502 The terminal receives a task request, the task request being used to instruct a task related to an application program to be processed

503 The terminal acquires, on the basis of the task request, the quality of service level of the task

504 When using a first thread to process a sub-task corresponding to the task request, the terminal configures the quality of service level of the first thread as the quality of service level of the task

505 The terminal processes, on the basis of a processing order corresponding to the quality of service level of the first thread, the sub-task corresponding to the first thread

506 When a target resource of the sub-task has been locked, the terminal configures a quality of service level of a second thread as the quality of service level of the first thread, the second thread being a thread locking the target resource

507 When the second thread releases the lock of the target resource, the terminal processes, on the basis of the target resource, the sub-task corresponding to the first thread

508 When the first thread invokes a third thread, the terminal adjusts, according to the amount of data of the first thread cached in the third thread, the quality of service levels of the first thread and the third thread

509 When the first thread fails to apply for a memory resource and the quality of service level of the first thread is greater than a second target level, the terminal allocates, from reserved memory resources, a memory resource to the first thread

(57) Abstract: A task processing method, an apparatus, a terminal, and a computer readable storage medium. Said method comprises: configuring a quality of service level of a task, the quality of service level being related to a user's perception, and configuring the priority of a thread executing the task as the quality of service level of the task, the priority of the thread being used to indicate an order of the thread being executed in a system. Such a method enables the execution order of the thread to conform to the quality of service level of the task. In this way, the higher the quality of service level of a task, i.e. the stronger the user's perception of a task, the faster the execution, thereby avoiding, to a certain extent, lag caused by a long task processing time.

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, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, 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))。

(57) 摘要: 一种任务处理方法、装置、终端以及计算机可读存储介质, 所述方法包括: 设置任务的服务质量等级, 该服务质量等级与用户感知度相关, 并将执行该任务的线程的优先级配置为该任务的服务质量等级, 线程的优先级用于指示线程在系统中被执行的执行顺序。通过这样的方法使得线程的执行顺序符合该任务的服务质量等级。这样服务质量等级越高的任务, 即用户感知度越强的任务, 执行的就越快, 从而可以一定程度上避免任务处理时间过长导致的卡顿现象。

任务处理方法、装置、终端以及计算机可读存储介质

本申请要求于 2019 年 04 月 11 日提交的申请号为 201910291330.3、发明名称为“任务处理方法、装置、终端以及计算机可读存储介质”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，特别涉及一种任务处理方法、装置、终端以及计算机可读存储介质。

背景技术

终端设备被广泛应用到人们的日常生活中，但是，随着使用时间的增长，终端设备在执行任务时会出现卡顿现象，例如，终端设备启动一个应用程序时，启动的时间较长，当 APP 启动完成后，终端设备的显示界面才能显示应用程序上的内容，显示界面在等待应用程序启动的过程中，也就出现了卡顿。应用程序之所以启动缓慢，是因为终端设备在处理应用程序启动任务时，应用程序的进程需要长时间的排队才能被处理，终端设备可以通过为系统内的应用程序分配优先级，并基于优先级的高低，来处理与各个应用相关的任务，以避免终端设备处理任务时的卡顿情况。

目前，任务处理可以是以下过程：终端设备的系统中包括多个应用程序，在终端设备的显示界面上显示的应用程序可以称为前台应用程序，不在显示界面上显示的应用程序可以称为后台应用程序，终端设备根据各个应用程序能否在显示界面上显示，来为各个应用程序分配优先级，其中，前台应用程序的优先级较高，后台应用程序的优先级较低，具体地，当终端设备在处理前台应用程序和后台应用程序的任务时，可以生成与各个应用程序相关的进程，终端设备通过处理进程来完成与应用程序相关的任务，其中，与前台应用程序相关的进程的优先级较高，与后台应用程序相关的进程的优先级较低。例如，终端设备在处理一个任务时，调用多个线程来实现该任务，终端设备的操作系统（例如 Linux）内核在调度层面上使用完全公平调度器（completely fair scheduler, CFS）调度这多个线程，在调度过程中，Linux 内核可以基于线程的优先级来进行调度，具体地，以虚拟时钟所指示的虚拟执行时间为基准，高优先级线程的虚拟时钟走的较慢，低优先级的进程的虚拟时钟走的较快，虽然高低优先级虚拟执行时间相对公平，但是低优先级的进程的执行时间比高优先级的进程的执行时间要少，当该任务所需的线程都处理完成时，该任务处理完成。但是当该多个线程中存在一个或多个线程，该一个或多个线程的优先级与该任务的需求不匹配时，容易导致该任务的执行时间过长。例如该多个线程中存在一个或多个后台应用的线程，而该一个或多个后台应用的线程由于自身的优先级较低而导致执行时间较长，从而导致该任务的总耗时较长，从用户角度则感知到系统卡顿，从而降低了用户体验。

发明内容

本申请实施例提供了一种任务处理方法、装置、终端以及计算机可读存储介质等，能够一定程度上避免卡顿问题，提升终端设备的用户体验。下面通过不同方面介绍本申请，应理解的是，以下各个方面的实现方式和有益效果可互相参考。

第一方面，本申请提供了一种任务处理方法，该方法包括：接收任务请求，所述任务请求用于指示处理与应用程序相关的任务；基于所述任务请求，获取所述任务的服务质量等级；当采用第一线程处理所述任务请求对应的子任务时，将所述第一线程的服务质量等级配置为所述任务的服务质量等级；基于所述第一线程的服务质量等级（或称之为优先级）对应的处理顺序，处理所述第一线程对应的子任务。

前述“线程”也可以为“进程”。任务为该终端通过应用程序为用户提供的服务，任务可以是启动任务，也可以是应用程序运行过程中的任一任务，例如滑动任务，其中，启动任务可以是终端启动应用程序的任务，滑动任务可以是终端在显示界面上进行应用程序切换的任务，当然，任务还可以是其他任务，本申请实施例对任务具体内容不做具体限定。

在上述方式中，任务请求的服务质量等级是新增的，与用户感知度相关。用户感知度强的任务请求的服务质量等级高于用户感知度弱的任务请求的服务质量等级。这个服务质量等级可以根据经验设置，也可以通过机器学习的方式设置。

这里的“子任务”根据任务的不同可以做不同的理解。假如该任务不可拆分或无需拆分执行，那么可以将子任务理解为该任务本身。假如该任务需要多个进程或多个线程来并行或按照某种顺序串行实现，即任务被分解到不同的进程或线程中，那么多个进程或线程分别执行的任务可以理解为该任务的“子任务”。至于任务的具体拆分方式本申请不做限定。

该任务请求可以携带该任务的任务标识以及对象标识，该对象标识用于指示该任务的对象，该对象可以是终端中的任一应用程序，对象标识可以是对象的名称，例如，该任务请求1中携带启动标识 `start` 以及对象标识 `ABC`，则该任务请求用于指示启动名称为 `ABC` 的应用程序。

基于上述可能实现方式，当采用第一线程处理任务请求所对应的子任务时，将该第一线程的服务质量等级配置为该任务的服务质量等级，基于该第一线程的服务质量等级对应的处理顺序，对该第一线程对应的子任务进行处理，可以使得在处理该任务过程中，被采用的第一线程的优先级与任务的服务质量等级相同，而不是一味的使用原始的优先级，这样可以使得第一线程被处理时的处理顺序符合该任务的服务质量等级，对于服务质量等级高的任务（例如用户比较敏感的界面绘制任务等），该线程被执行的也就越快，从而任务实现的也就越快，用户体验也就越好。例如，第一线程可能为某后台应用的线程，该线程若按照后台应用的优先级的执行的话等待时间较长，将导致该任务的执行时间较长，但是由于该任务的服务质量等级较高，所以该第一线程可以根据该任务的服务质量等级提高优先级，从而减少执行等待时间，提高任务的处理效率。

在一种可能实现方式中，所述基于所述第一线程的服务质量等级对应的处理顺序，处理所述第一线程对应的子任务包括：将所述子任务插入到处理队列中服务质量等级低于所述线程的服务质量等级或不具有服务质量等级的子任务之前，按照所述处理队列中的顺序，处理所述子任务。

其中，该处理队列用于存储各个线程的子任务。基于上述可能实现方式，对于服务质量

等级较高的线程的子任务，可以在处理队列中插队，从而可以使得终端优先处理服务等级较高的线程的子任务，进而使得服务等级较高的任务可以优先处理。

在一种可能实现方式中，所述处理所述第一线程对应的子任务包括：当所述子任务的目标资源已加锁时，将第二线程的服务质量等级配置为所述第一线程的服务质量等级，所述第二线程为对所述目标资源进行加锁的线程；当所述第二线程释放所述目标资源的锁时，基于所述目标资源处理所述子任务。

在一种可能实现方式中，所述当所述子任务的目标资源已加锁时，将第二线程的服务质量等级配置为所述第一线程的服务质量等级包括：当执行所述第一线程对应的子任务时，对所述目标资源加锁；当对所述目标资源加锁失败时，确定所述第二线程；当所述第二线程的服务质量等级低于所述第一线程的服务质量等级时，将所述第二线程的服务质量等级更新为所述第一线程的服务质量等级。

该目标资源为系统内可以上锁的任一资源，例如，内存控制块以及 Bind 节点，第一线程对该目标资源加的锁可以包括互斥锁、读写信号量、Java 锁等任一种锁，当然还可以是其他锁，例如，当第一线程需要内存资源时，该终端可以通过读写信号量对内存控制块进行加锁，以便第一线程可以使用与该内存控制块对应的内存资源，再例如，当第一线程通过 Binder 驱动调用其他线程时，终端可以通过互斥锁对 Binder 节点进行加锁，以便第一线程可以调用与对 Binder 节点对应的线程。本申请实施例对目标资源上的锁不做具体限定。

在一种可能实现方式中，所述当所述第二线程释放所述目标资源的锁时，基于所述目标资源处理所述子任务包括：当所述第二线程释放所述目标资源的锁之后，按照所述目标资源的等待队列中顺序，对所述目标资源加锁，恢复所述第二线程的原服务质量等级，所述目标资源的等待队列用于存储待处理所述目标资源的线程；当对所述目标资源加锁完成后，基于所述目标资源处理所述子任务。

基于上述可能实现方式，当第一线程的服务质量较低时，将该第二线程的服务质量等级更新为该第一线程的服务质量等级，从而可以使得该第一线程和第二线程的服务质量等级一致，那么当第二线程释放锁时，基于该目标资源的等待队列中的顺序，该第一线程可以对该目标资源进行加锁并处理该目标资源，从而可以避免出现优先级反转的情况。

在一种可能实现方式中，所述方法还包括：当所述第一线程调用第三线程时，根据所述第三线程中所缓存的所述第一线程的数据量，调整所述第一线程和所述第三线程的服务质量等级。

在一种可能实现方式中，所述当所述第一线程调用第三线程时，根据所述第三线程中所缓存的所述第一线程的数据量，调整所述第一线程和所述第三线程的服务质量等级，包括：

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值，保持所述第一线程和所述第三线程的服务质量等级不变；当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值，降低所述第一线程的服务质量等级。

基于上述可能实现方式，动态调整该终端在处理任务请求时的所执行的线程的 QoS 等级，使得该终端执行第一线程可以保持在做一个最合适的速度上，从而可以避免该第一线程在使用系统中资源时，造成不必要的功耗，或者不够下层线程使用的情况。

在一种可能实现方式中，所述当所述第一线程调用第三线程时，根据所述第三线程中所

缓存的所述第一线程的数据量,调整所述第一线程和所述第三线程的服务质量等级,包括:当所述第一线程调用所述第三线程时,如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值,提高所述第一线程的服务质量等级;当所述第一线程调用所述第三线程时,如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值,提高所述第三线程的服务质量等级。

基于上述可能实现方式,当该第三线程中所缓存的所述线程的数据量小于数据量阈值时,终端提高该第一线程的服务质量等级,从而使得第三线程中该第一线程的数据量达到数据量阈值,当该第三线程突然消耗过快时,可以避免该第三线程内中该第一线程的数据量跟不上消耗进度。当该第三线程中所缓存的该第一线程的数据量等于或大于该数据量阈值时,终端提高该第三线程的服务质量等级,从而使得第三线程中所缓存的该第一线程的数据量消耗的速度加快,从而可以避免该第一线程产生的数据在该第三线程中积累。

在一种可能实现方式中,所述任务的服务质量等级由所述任务的任务特征确定。

基于上述可能实现方式,当根据任务的服务质量等级配置第一线程的服务质量等级时,可以使得第一线程的服务质量等级与任务的服务质量等级相关。

在一种可能实现方式中,所述方法还包括:当处理所述任务时,如果所述应用程序由前台运行切换至后台运行,将与所述任务对应的线程的服务质量等级配置为第一目标等级。

基于上述可能实现方式,通过将后台运行的应用程序与所述任务对应的线程的服务等级配置为第一目标等级可以使得终端不优先处理后台运行的应用程序的任务,从而可以降低后台运行的应用程序的对前台运行的应用程序的干扰。

在一种可能实现方式中,所述将所述第一线程的服务质量等级配置为所述任务的服务质量等级包括:配置所述第一线程的线程控制块结构体内的服务质量等级字段为所述任务的服务质量等级,所述线程控制块结构体用于存储线程的信息。

在一种可能实现方式中,所述方法还包括:当所述第一线程申请内存资源失败,且所述第一线程的服务质量等级大于第二目标等级时,从预留内存资源中为所述第一线程分配内存资源。

基于上述可能实现方式,在内存的紧急的情况,通过在预留内存资源中为服务质量等级较高的线程分配内存,可以保证服务质量等级较高的线程处理进度。

在一种可能实现方式中,不同的任务请求对应于不同的任务质量等级。

基于上述可能实现方式,当应用程序处理不同任务时配置不同的服务质量等级,根据用户的需求,可以使得在应用程序中的线程在处理不同任务时处理优先级不同,从而使得终端的能耗较低。

第二方面,本申请提供了一种任务处理方法,该方法包括:接收任务请求;基于所述任务请求,获取所述任务请求(或所述任务请求对应的任务)对应的服务质量等级;当采用多个第一线程处理所述任务请求对应的子任务时,将所述多个第一线程的服务质量等级配置为所述任务的服务质量等级,所述多个第一线程属于至少两个应用程序;基于所述多个第一线程的服务质量等级对应的处理顺序,处理所述第一线程对应的子任务。

通常应用程序在执行任务时会调用多个线程来实现该任务。当执行该任务的多个线程都采用上述方式配置为与该任务的服务质量等级相同的优先级时,可以一定程度上避免因应用的不同优先级而导致的等待问题,进而避免处理任务时的卡顿现象。

在一些实现方式下,所述多个第一线程包括线程A和线程B,所述线程A调用所述线程B以实现所述任务,所述线程A通过binder机制将线程A的服务质量等级传递给所述线程B。从而使得线程B和线程A的服务指令等级均与所述任务的服务质量等级一致,避免优先级反转。

在一些实现方式下,所述多个第一线程包括线程A和线程B,所述线程A调用所述线程B,根据所述线程B中所缓存的所述线程A的数据量调整所述线程A和所述线程B的服务质量等级。从而实现服务质量等级根据任务执行情况的动态调整,尤其是在线程B中缓存的线程A的数据量较多时下调服务质量等级,避免资源对线程A和线程B的过度供给。

在一些实现方式下,所述多个第一线程中的一个第一线程在访问目标资源时去顶所述目标资源已加锁,则将对所述目标资源加锁的线程的服务质量等级配置为该第一线程的服务质量等级。这样可加快该目标资源的释放,从而加快该第一线程的执行,进而加快该任务的执行。

第三方面,本申请提供了一种任务处理装置,用于执行上述任务处理方法。具体地,该任务处理装置包括用于执行上述任一方面或上述任一方面的任一种可选方式提供的任务处理方法的功能模块。

第四方面,本申请提供一种终端,该终端包括处理器和存储器,该存储器中存储有至少一条指令,该指令由该处理器加载并执行以实现如上述任务处理方法所执行的操作。

第五方面,本申请提供一种计算机可读存储介质,该存储介质中存储有至少一条指令,该指令由处理器加载并执行以实现如上述任务处理方法所执行的操作。

第六方面,本申请提供了一种计算机程序产品(或称计算机程序),该计算机程序产品中存储有至少一条指令,该指令由处理器加载并执行以实现如上述任务处理方法所执行的操作。

附图说明

为了更清楚地说明本申请实施例中的技术方案,下面将对实施例描述中所需要使用的附图作简单地介绍。显而易见地,下面描述中的附图仅仅是本申请的一些实施例,对于本领域普通技术人员来讲,还可以根据这些附图获得其他的附图。

图1是本申请实施例提供的一种终端的结构示意图;

图2是本申请实施例提供的一种终端的软件结构框图;

图3是本申请实施例提供的一种实施环境的示意图;

图4是本申请实施例提供的一种终端的系统示意图;

图5是本申请实施例提供的一种任务处理方法的流程图;

图6是本申请实施例提供的一种热启动任务处理方法的流程图;

图7是本申请实施例提供的一种冷启动任务处理方法的流程图;

图8是本申请实施例提供的一种同步服务质量等级的流程图;

图9是本申请实施例提供的一种同步互斥锁的流程图;

图10是本申请实施例提供的一种同步读写信号量的流程图;

图11是本申请实施例提供的一种基于安卓系统架构任务处理过程的示意图;

图12是本申请实施例提供的一种滑动场景下进行服务质量等级传递的流程图;

图 13 是本申请实施例提供的一种渲染场景下基于流水线的动态服务质量分级模型的示意图；

图 14 是本申请实施例提供的一种任务处理装置的结构装置图。

具体实施方式

为使本申请的目的、技术方案和优点更加清楚，下面将结合附图对本申请实施方式作进一步地详细描述。

参见图 1，图 1 是本申请实施例提供的一种终端的结构示意图，终端 100 可以包括处理器 110，外部存储器接口 120，内部存储器 311，通用串行总线(universal serial bus, USB)接口 130，充电管理模块 140，电源管理模块 141，电池 142，天线 1，天线 2，移动通信模块 150，无线通信模块 160，音频模块 170，扬声器 170A，受话器 170B，麦克风 170C，耳机接口 170D，传感器模块 180，按键 190，马达 191，指示器 192，摄像头 193，显示屏 194，以及用户标识模块(subscriber identification module, SIM)卡接口 195 等。其中传感器模块 180 可以包括压力传感器 180A，陀螺仪传感器 180B，气压传感器 180C，磁传感器 180D，加速度传感器 180E，距离传感器 180F，接近光传感器 180G，指纹传感器 180H，温度传感器 180J，触摸传感器 180K，环境光传感器 180L，骨传导传感器 180M 等。

可以理解的是，本申请实施例示意的结构并不构成对终端 100 的具体限定。在本申请另一些实施例中，终端 100 可以包括比图示更多或更少的部件，或者组合某些部件，或者拆分某些部件，或者不同的部件布置。图示的部件可以以硬件，软件或软件和硬件的组合实现。

处理器 110 可以包括一个或多个处理单元，例如：处理器 110 可以包括应用处理器(application processor, AP)，调制解调处理器，图形处理器(graphics processing unit, GPU)，图像信号处理器(image signal processor, ISP)，控制器，视频编解码器，数字信号处理器(digital signal processor, DSP)，基带处理器，和/或神经网络处理器(neural-network processing unit, NPU)等。其中，不同的处理单元可以是独立的器件，也可以集成在一个或多个处理器中。

控制器可以根据指令操作码和时序信号，产生操作控制信号，完成取指令和执行指令的控制。

处理器 110 中还可以设置存储器，用于存储指令和数据。在一些实施例中，处理器 110 中的存储器为高速缓冲存储器。该存储器可以保存处理器 110 刚用过或循环使用的指令或数据。如果处理器 110 需要再次使用该指令或数据，可从所述存储器中直接调用。避免了重复存取，减少了处理器 110 的等待时间，因而提高了系统的效率。

在一些实施例中，处理器 110 可以包括一个或多个接口。接口可以包括集成电路(inter-integrated circuit, I2C)接口，集成电路内置音频(inter-integrated circuit sound, I2S)接口，脉冲编码调制(pulse code modulation, PCM)接口，通用异步收发传输器(universal asynchronous receiver/transmitter, UART)接口，移动产业处理器接口(mobile industry processor interface, MIPI)，通用输入输出(general-purpose input/output, GPIO)接口，用户标识模块(subscriber identity module, SIM)接口，和/或通用串行总线(universal serial bus, USB)接口等。

I2C 接口是一种双向同步串行总线，包括一根串行数据线(serial data line, SDA)和一

根串行时钟线(derail clock line, SCL)。在一些实施例中,处理器 110 可以包含多组 I2C 总线。处理器 110 可以通过不同的 I2C 总线接口分别耦合触摸传感器 180K,充电器,闪光灯,摄像头 193 等。例如:处理器 110 可以通过 I2C 接口耦合触摸传感器 180K,使处理器 110 与触摸传感器 180K 通过 I2C 总线接口通信,实现终端 100 的触摸功能。

I2S 接口可以用于音频通信。在一些实施例中,处理器 110 可以包含多组 I2S 总线。处理器 110 可以通过 I2S 总线与音频模块 170 耦合,实现处理器 110 与音频模块 170 之间的通信。在一些实施例中,音频模块 170 可以通过 I2S 接口向无线通信模块 160 传递音频信号,实现通过蓝牙耳机接听电话的功能。

PCM 接口也可以用于音频通信,将模拟信号抽样,量化和编码。在一些实施例中,音频模块 170 与无线通信模块 160 可以通过 PCM 总线接口耦合。在一些实施例中,音频模块 170 也可以通过 PCM 接口向无线通信模块 160 传递音频信号,实现通过蓝牙耳机接听电话的功能。所述 I2S 接口和所述 PCM 接口都可以用于音频通信。

UART 接口是一种通用串行数据总线,用于异步通信。该总线可以为双向通信总线。它将要传输的数据在串行通信与并行通信之间转换。在一些实施例中,UART 接口通常被用于连接处理器 110 与无线通信模块 160。例如:处理器 110 通过 UART 接口与无线通信模块 160 中的蓝牙模块通信,实现蓝牙功能。在一些实施例中,音频模块 170 可以通过 UART 接口向无线通信模块 160 传递音频信号,实现通过蓝牙耳机播放音乐的功能。

MIPI 接口可以被用于连接处理器 110 与显示屏 194,摄像头 193 等外围器件。MIPI 接口包括摄像头串行接口(camera serial interface, CSI),显示屏串行接口(display serial interface, DSI)等。在一些实施例中,处理器 110 和摄像头 193 通过 CSI 接口通信,实现终端 100 的拍摄功能。处理器 110 和显示屏 194 通过 DSI 接口通信,实现终端 100 的显示功能。

GPIO 接口可以通过软件配置。GPIO 接口可以被配置为控制信号,也可被配置为数据信号。在一些实施例中,GPIO 接口可以用于连接处理器 110 与摄像头 193,显示屏 194,无线通信模块 160,音频模块 170,传感器模块 180 等。GPIO 接口还可以被配置为 I2C 接口,I2S 接口,UART 接口,MIPI 接口等。

USB 接口 130 是符合 USB 标准规范的接口,具体可以是 Mini USB 接口, Micro USB 接口, USB Type C 接口等。USB 接口 130 可以用于连接充电器为终端 100 充电,也可以用于终端 100 与外围设备之间传输数据。也可以用于连接耳机,通过耳机播放音频。该接口还可以用于连接其他电子设备,例如 AR 设备等。

可以理解的是,本申请实施例示意的各模块间的接口连接关系,只是示意性说明,并不构成对终端 100 的结构限定。在本申请另一些实施例中,终端 100 也可以采用上述实施例中不同的接口连接方式,或多种接口连接方式的组合。

充电管理模块 140 用于从充电器接收充电输入。其中,充电器可以是无线充电器,也可以是有线充电器。在一些有线充电的实施例中,充电管理模块 140 可以通过 USB 接口 130 接收有线充电器的充电输入。在一些无线充电的实施例中,充电管理模块 140 可以通过终端 100 的无线充电线圈接收无线充电输入。充电管理模块 140 为电池 142 充电的同时,还可以通过电源管理模块 141 为电子设备供电。

电源管理模块 141 用于连接电池 142,充电管理模块 140 与处理器 110。电源管理模块 141 接收电池 142 和/或充电管理模块 140 的输入,为处理器 110,内部存储器 311,显示屏

194, 摄像头 193, 和无线通信模块 160 等供电。电源管理模块 141 还可以用于监测电池容量, 电池循环次数, 电池健康状态(漏电, 阻抗)等参数。在其他一些实施例中, 电源管理模块 141 也可以设置于处理器 110 中。在另一些实施例中, 电源管理模块 141 和充电管理模块 140 也可以设置于同一个器件中。

终端 100 的无线通信功能可以通过天线 1, 天线 2, 移动通信模块 150, 无线通信模块 160, 调制解调处理器以及基带处理器等实现。

天线 1 和天线 2 用于发射和接收电磁波信号。终端 100 中的每个天线可用于覆盖单个或多个通信频带。不同的天线还可以复用, 以提高天线的利用率。例如: 可以将天线 1 复用为无线局域网的分集天线。在另外一些实施例中, 天线可以和调谐开关结合使用。

移动通信模块 150 可以提供应用在终端 100 上的包括 2G/3G/4G/5G 等无线通信的解决方案。移动通信模块 150 可以包括至少一个滤波器, 开关, 功率放大器, 低噪声放大器(low noise amplifier, LNA)等。移动通信模块 150 可以由天线 1 接收电磁波, 并对接收的电磁波进行滤波, 放大等处理, 传送至调制解调处理器进行解调。移动通信模块 150 还可以对经调制解调处理器调制后的信号放大, 经天线 1 转为电磁波辐射出去。在一些实施例中, 移动通信模块 150 的至少部分功能模块可以被设置于处理器 110 中。在一些实施例中, 移动通信模块 150 的至少部分功能模块可以与处理器 110 的至少部分模块被设置在同一个器件中。

调制解调处理器可以包括调制器和解调器。其中, 调制器用于将待发送的低频基带信号调制成中高频信号。解调器用于将接收的电磁波信号解调为低频基带信号。随后解调器将解调得到的低频基带信号传送至基带处理器处理。低频基带信号经基带处理器处理后, 被传递给应用处理器。应用处理器通过音频设备(不限于扬声器 170A, 受话器 170B 等)输出声音信号, 或通过显示屏 194 显示图像或视频。在一些实施例中, 调制解调处理器可以是独立的器件。在另一些实施例中, 调制解调处理器可以独立于处理器 110, 与移动通信模块 150 或其他功能模块设置在同一个器件中。

无线通信模块 160 可以提供应用在终端 100 上的包括无线局域网(wireless local area networks, WLAN)(如无线保真(wireless fidelity, Wi-Fi)网络), 蓝牙(bluetooth, BT), 全球导航卫星系统(global navigation satellite system, GNSS), 调频(frequency modulation, FM), 近距离无线通信技术(near field communication, NFC), 红外技术(infrared, IR)等无线通信的解决方案。无线通信模块 160 可以是集成至少一个通信处理模块的一个或多个器件。无线通信模块 160 经由天线 2 接收电磁波, 将电磁波信号调频以及滤波处理, 将处理后的信号发送到处理器 110。无线通信模块 160 还可以从处理器 110 接收待发送的信号, 对其进行调频, 放大, 经天线 2 转为电磁波辐射出去。

在一些实施例中, 终端 100 的天线 1 和移动通信模块 150 耦合, 天线 2 和无线通信模块 160 耦合, 使得终端 100 可以通过无线通信技术与网络以及其他设备通信。所述无线通信技术可以包括全球移动通讯系统(global system for mobile communications, GSM), 通用分组无线服务(general packet radio service, GPRS), 码分多址接入(code division multiple access, CDMA), 宽带码分多址(wideband code division multiple access, WCDMA), 时分码分多址(time-division code division multiple access, TD-SCDMA), 长期演进(long term evolution, LTE), BT, GNSS, WLAN, NFC, FM, 和/或 IR 技术等。所述 GNSS 可以包括全球卫星定位系统(global positioning system, GPS), 全球导航卫星系统(global navigation

satellite system, GLONASS), 北斗卫星导航系统(beidou navigation satellite system, BDS), 准天顶卫星系统(quasi-zenith satellite system, QZSS)和/或星基增强系统(satellite based augmentation systems, SBAS)。

终端 100 通过 GPU, 显示屏 194, 以及应用处理器等实现显示功能。GPU 为图像处理的微处理器, 连接显示屏 194 和应用处理器。GPU 用于执行数学和几何计算, 用于图形渲染。处理器 110 可包括一个或多个 GPU, 其执行程序指令以生成或改变显示信息。

显示屏 194 用于显示图像, 视频等。显示屏 194 包括显示面板。显示面板可以采用液晶显示屏(liquid crystal display, LCD), 有机发光二极管(organic light-emitting diode, OLED), 有源矩阵有机发光二极体或主动矩阵有机发光二极体(active-matrix organic light emitting diode 的, AMOLED), 柔性发光二极管(flex light-emitting diode, FLED), Miniled, MicroLed, Micro-oLed, 量子点发光二极管(quantum dot light emitting diodes, QLED)等。在一些实施例中, 终端 100 可以包括 1 个或 N 个显示屏 194, N 为大于 1 的正整数。

终端 100 可以通过 ISP, 摄像头 193, 视频编解码器, GPU, 显示屏 194 以及应用处理器等实现拍摄功能。

ISP 用于处理摄像头 193 反馈的数据。例如, 拍照时, 打开快门, 光线通过镜头被传递到摄像头感光元件上, 光信号转换为电信号, 摄像头感光元件将所述电信号传递给 ISP 处理, 转化为肉眼可见的图像。ISP 还可以对图像的噪点, 亮度, 肤色进行算法优化。ISP 还可以对拍摄场景的曝光, 色温等参数优化。在一些实施例中, ISP 可以设置在摄像头 193 中。

摄像头 193 用于捕获静态图像或视频。物体通过镜头生成光学图像投射到感光元件。感光元件可以是电荷耦合器件(charge coupled device, CCD)或互补金属氧化物半导体(complementary metal-oxide-semiconductor, CMOS)光电晶体管。感光元件把光信号转换成电信号, 之后将电信号传递给 ISP 转换成数字图像信号。ISP 将数字图像信号输出到 DSP 加工处理。DSP 将数字图像信号转换成标准的 RGB, YUV 等格式的图像信号。在一些实施例中, 终端 100 可以包括 1 个或 N 个摄像头 193, N 为大于 1 的正整数。

数字信号处理器用于处理数字信号, 除了可以处理数字图像信号, 还可以处理其他数字信号。例如, 当终端 100 在频点选择时, 数字信号处理器用于对频点能量进行傅里叶变换等。

视频编解码器用于对数字视频压缩或解压缩。终端 100 可以支持一种或多种视频编解码器。这样, 终端 100 可以播放或录制多种编码格式的视频, 例如: 动态图像专家组(moving picture experts group, MPEG)1, MPEG2, MPEG3, MPEG4 等。

NPU 为神经网络(neural-network, NN)计算处理器, 通过借鉴生物神经网络结构, 例如借鉴人脑神经元之间传递模式, 对输入信息快速处理, 还可以不断的自学习。通过 NPU 可以实现终端 100 的智能认知等应用, 例如: 图像识别, 人脸识别, 语音识别, 文本理解等。

外部存储器接口 120 可以用于连接外部存储卡, 例如微型安全数码(micro secure Digital memory, Micro SD)卡, 实现扩展终端 100 的存储能力。外部存储卡通过外部存储器接口 120 与处理器 110 通信, 实现数据存储功能。例如将音乐, 视频等文件保存在外部存储卡中。

内部存储器 311 可以用于存储计算机可执行程序代码, 所述可执行程序代码包括指令。内部存储器 311 可以包括存储程序区和存储数据区。其中, 存储程序区可存储操作系统, 至少一个功能所需的应用程序(比如声音播放功能, 图像播放功能等)等。存储数据区可存储终

端 100 使用过程中所创建的数据(比如音频数据, 电话本等)等。此外, 内部存储器 311 可以包括高速随机存取存储器, 还可以包括非易失性存储器, 例如至少一个磁盘存储器件, 闪存器件, 通用闪存存储器(universal flash storage, UFS)等。处理器 110 通过运行存储在内部存储器 311 的指令, 和/或存储在设置于处理器中的存储器的指令, 执行终端 100 的各种功能应用以及数据处理。

终端 100 可以通过音频模块 170, 扬声器 170A, 受话器 170B, 麦克风 170C, 耳机接口 170D, 以及应用处理器等实现音频功能。例如音乐播放, 录音等。

音频模块 170 用于将数字音频信息转换成模拟音频信号输出, 也用于将模拟音频输入转换为数字音频信号。音频模块 170 还可以用于对音频信号编码和解码。在一些实施例中, 音频模块 170 可以设置于处理器 110 中, 或将音频模块 170 的部分功能模块设置于处理器 110 中。

扬声器 170A, 也称“喇叭”, 用于将音频电信号转换为声音信号。终端 100 可以通过扬声器 170A 收听音乐, 或收听免提通话。

受话器 170B, 也称“听筒”, 用于将音频电信号转换成声音信号。当终端 100 接听电话或语音信息时, 可以通过将受话器 170B 靠近人耳接听语音。

麦克风 170C, 也称“话筒”, “传声器”, 用于将声音信号转换为电信号。当拨打电话或发送语音信息时, 用户可以通过人嘴靠近麦克风 170C 发声, 将声音信号输入到麦克风 170C。终端 100 可以设置至少一个麦克风 170C。在另一些实施例中, 终端 100 可以设置两个麦克风 170C, 除了采集声音信号, 还可以实现降噪功能。在另一些实施例中, 终端 100 还可以设置三个, 四个或更多麦克风 170C, 实现采集声音信号, 降噪, 还可以识别声音来源, 实现定向录音功能等。

耳机接口 170D 用于连接有线耳机。耳机接口 170D 可以是 USB 接口 130, 也可以是 3.5mm 的开放移动电子设备平台(open mobile terminal platform, OMTp)标准接口, 美国蜂窝电信工业协会(cellular telecommunications industry association of the USA, CTIA)标准接口。

压力传感器 180A 用于感受压力信号, 可以将压力信号转换成电信号。在一些实施例中, 压力传感器 180A 可以设置于显示屏 194。压力传感器 180A 的种类很多, 如电阻式压力传感器, 电感式压力传感器, 电容式压力传感器等。电容式压力传感器可以是包括至少两个具有导电材料的平行板。当有力作用于压力传感器 180A, 电极之间的电容改变。终端 100 根据电容的变化确定压力的强度。当有触摸操作作用于显示屏 194, 终端 100 根据压力传感器 180A 检测所述触摸操作强度。终端 100 也可以根据压力传感器 180A 的检测信号计算触摸的位置。在一些实施例中, 作用于相同触摸位置, 但不同触摸操作强度的触摸操作, 可以对应不同的操作指令。例如: 当有触摸操作强度小于第一压力阈值的触摸操作作用于短消息应用图标时, 执行查看短消息的指令。当有触摸操作强度大于或等于第一压力阈值的触摸操作作用于短消息应用图标时, 执行新建短消息的指令。

陀螺仪传感器 180B 可以用于确定终端 100 的运动姿态。在一些实施例中, 可以通过陀螺仪传感器 180B 确定终端 100 围绕三个轴(即, x, y 和 z 轴)的角速度。陀螺仪传感器 180B 可以用于拍摄防抖。示例性的, 当按下快门, 陀螺仪传感器 180B 检测终端 100 抖动的角度, 根据角度计算出镜头模组需要补偿的距离, 让镜头通过反向运动抵消终端 100 的抖动, 实现防

抖。陀螺仪传感器 180B 还可以用于导航, 体感游戏场景。

气压传感器 180C 用于测量气压。在一些实施例中, 终端 100 通过气压传感器 180C 测得的气压值计算海拔高度, 辅助定位和导航。

磁传感器 180D 包括霍尔传感器。终端 100 可以利用磁传感器 180D 检测翻盖皮套的开合。在一些实施例中, 当终端 100 是翻盖机时, 终端 100 可以根据磁传感器 180D 检测翻盖的开合。进而根据检测到的皮套的开合状态或翻盖的开合状态, 设置翻盖自动解锁等特性。

加速度传感器 180E 可检测终端 100 在各个方向上(一般为三轴)加速度的大小。当终端 100 静止时可检测出重力的大小及方向。还可以用于识别电子设备姿态, 应用于横竖屏切换, 计步器等应用。

距离传感器 180F, 用于测量距离。终端 100 可以通过红外或激光测量距离。在一些实施例中, 拍摄场景, 终端 100 可以利用距离传感器 180F 测距以实现快速对焦。

接近光传感器 180G 可以包括例如发光二极管(LED)和光检测器, 例如光电二极管。发光二极管可以是红外发光二极管。终端 100 通过发光二极管向外发射红外光。终端 100 使用光电二极管检测来自附近物体的红外反射光。当检测到充分的反射光时, 可以确定终端 100 附近有物体。当检测到不充分的反射光时, 终端 100 可以确定终端 100 附近没有物体。终端 100 可以利用接近光传感器 180G 检测用户手持终端 100 贴近耳朵通话, 以便自动熄灭屏幕达到省电的目的。接近光传感器 180G 也可用于皮套模式, 口袋模式自动解锁与锁屏。

环境光传感器 180L 用于感知环境光亮度。终端 100 可以根据感知的环境光亮度自适应调节显示屏 194 亮度。环境光传感器 180L 也可用于拍照时自动调节白平衡。环境光传感器 180L 还可以与接近光传感器 180G 配合, 检测终端 100 是否在口袋里, 以防误触。

指纹传感器 180H 用于采集指纹。终端 100 可以利用采集的指纹特性实现指纹解锁, 访问应用锁, 指纹拍照, 指纹接听来电等。

温度传感器 180J 用于检测温度。在一些实施例中, 终端 100 利用温度传感器 180J 检测的温度, 执行温度处理策略。例如, 当温度传感器 180J 上报的温度超过阈值, 终端 100 执行降低位于温度传感器 180J 附近的处理器的性能, 以便降低功耗实施热保护。在另一些实施例中, 当温度低于另一阈值时, 终端 100 对电池 142 加热, 以避免低温导致终端 100 异常关机。在其他一些实施例中, 当温度低于又一阈值时, 终端 100 对电池 142 的输出电压执行升压, 以避免低温导致的异常关机。

触摸传感器 180K, 也称“触控器件”。触摸传感器 180K 可以设置于显示屏 194, 由触摸传感器 180K 与显示屏 194 组成触摸屏, 也称“触控屏”。触摸传感器 180K 用于检测作用于其上或附近的触摸操作。触摸传感器可以将检测到的触摸操作传递给应用处理器, 以确定触摸事件类型。可以通过显示屏 194 提供与触摸操作相关的视觉输出。在另一些实施例中, 触摸传感器 180K 也可以设置于终端 100 的表面, 与显示屏 194 所处的位置不同。

骨传导传感器 180M 可以获取振动信号。在一些实施例中, 骨传导传感器 180M 可以获取人体声部振动骨块的振动信号。骨传导传感器 180M 也可以接触人体脉搏, 接收血压跳动信号。在一些实施例中, 骨传导传感器 180M 也可以设置于耳机中, 结合成骨传导耳机。音频模块 170 可以基于所述骨传导传感器 180M 获取的声部振动骨块的振动信号, 解析出语音信号, 实现语音功能。应用处理器可以基于所述骨传导传感器 180M 获取的血压跳动信号解析心率信息, 实现心率检测功能。

按键 190 包括开机键, 音量键等。按键 190 可以是机械按键。也可以是触摸式按键。终端 100 可以接收按键输入, 产生与终端 100 的用户设置以及功能控制有关的键信号输入。

马达 191 可以产生振动提示。马达 191 可以用于来电振动提示, 也可以用于触摸振动反馈。例如, 作用于不同应用(例如拍照, 音频播放等)的触摸操作, 可以对应不同的振动反馈效果。作用于显示屏 194 不同区域的触摸操作, 马达 191 也可对应不同的振动反馈效果。不同的应用场景(例如: 时间提醒, 接收信息, 闹钟, 游戏等)也可以对应不同的振动反馈效果。触摸振动反馈效果还可以支持自定义。

指示器 192 可以是指示灯, 可以用于指示充电状态, 电量变化, 也可以用于指示消息, 未接来电, 通知等。

SIM 卡接口 195 用于连接 SIM 卡。SIM 卡可以通过插入 SIM 卡接口 195, 或从 SIM 卡接口 195 拔出, 实现和终端 100 的接触和分离。终端 100 可以支持 1 个或 N 个 SIM 卡接口, N 为大于 1 的正整数。SIM 卡接口 195 可以支持 Nano SIM 卡, Micro SIM 卡, SIM 卡等。同一个 SIM 卡接口 195 可以同时插入多张卡。所述多张卡的类型可以相同, 也可以不同。SIM 卡接口 195 也可以兼容不同类型的 SIM 卡。SIM 卡接口 195 也可以兼容外部存储卡。终端 100 通过 SIM 卡和网络交互, 实现通话以及数据通信等功能。在一些实施例中, 终端 100 采用 eSIM, 即: 嵌入式 SIM 卡。eSIM 卡可以嵌在终端 100 中, 不能和终端 100 分离。

终端 100 的软件系统可以采用分层架构, 事件驱动架构, 微核架构, 微服务架构, 或云架构。本申请实施例以分层架构的 Android 系统为例, 示例性说明终端 100 的软件结构。

图 2 是本申请实施例提供的一种终端的软件结构框图, 参见图 2, 分层架构将软件分成若干个层, 每一层都有清晰的角色和分工。层与层之间通过软件接口通信。在一些实施例中, 将 Android 系统分为四层, 从上至下分别为应用程序层、应用程序框架层、安卓运行时(Android runtime)、本地层以及内核层。

应用程序层可以包括一系列应用程序包。如图 2 所示, 应用程序包可以包括相机, 图库, 日历, 通话, 地图, 导航, WLAN, 蓝牙, 音乐, 视频, 短信息等应用程序。

应用程序框架层为应用程序层的应用程序提供应用编程接口(application programming interface, API)和编程框架。应用程序框架层包括一些预先定义的函数。

如图 2 所示, 应用程序框架层可以包括窗口管理器, 内容提供者, 视图系统, 电话管理器, 资源管理器, 通知管理等。

窗口管理器用于管理窗口程序。窗口管理器可以获取显示屏大小, 判断是否有状态栏, 锁定屏幕, 截取屏幕等。

内容提供者用来存放和获取数据, 并使这些数据可以被应用程序访问。所述数据可以包括视频, 图像, 音频, 拨打和接听的电话, 浏览历史和书签, 电话簿等。

视图系统包括可视控件, 例如显示文字的控件, 显示图片的控件等。视图系统可用于构建应用程序。显示界面可以由一个或多个视图组成的。例如, 包括短信通知图标的显示界面, 可以包括显示文字的视图以及显示图片的视图。

电话管理器用于提供终端 100 的通信功能。例如通话状态的管理(包括接通, 挂断等)。

资源管理器为应用程序提供各种资源, 比如本地化字符串, 图标, 图片, 布局文件, 视频文件等等。

通知管理器使应用程序可以在状态栏中显示通知信息, 可以用于传达告知类型的消息,

可以短暂停留后自动消失，无需用户交互。比如通知管理器被用于告知下载完成，消息提醒等。通知管理器还可以是以图表或者滚动条文本形式出现在系统顶部状态栏的通知，例如后台运行的应用程序的通知，还可以是以对话窗口形式出现在屏幕上的通知。例如在状态栏提示文本信息，发出提示音，电子设备振动，指示灯闪烁等。

Android Runtime 包括核心库和虚拟机。Android runtime 负责安卓系统的调度和管理。

核心库包含两部分：一部分是 java 语言需要调用的功能函数，另一部分是安卓的核心库。

应用程序层和应用程序框架层运行在虚拟机中。虚拟机将应用程序层和应用程序框架层的 java 文件执行为二进制文件。虚拟机用于执行对象生命周期的管理，堆栈管理，线程管理，安全和异常的管理，以及垃圾回收等功能。

本地层可以包括多个功能模块。例如：表面管理器(surface manager)，硬件合成器(hardware composer, HWC)，媒体库，(Media Libraries)，三维图形处理库(例如：OpenGL ES)，2D 图形引擎(例如：SGL)等。

表面管理器用于对显示子系统进行管理，并且为多个应用程序提供了 2D 和 3D 图层的融合。HWC 用于融合 SF 所合成图层。

媒体库支持多种常用的音频，视频格式回放和录制，以及静态图像文件等。媒体库可以支持多种音视频编码格式，例如：MPEG4，H.364，MP3，AAC，AMR，JPG，PNG 等。

三维图形处理库用于实现三维图形绘图，图像渲染，合成，和图层处理等。2D 图形引擎是 2D 绘图的绘图引擎。

内核层是硬件和软件之间的层。内核层至少包含显示驱动，摄像头驱动，音频驱动，传感器驱动。

下面结合前台运行应用程序的场景，示例性说明终端 300 软件以及硬件的工作流程。例如，当任一应用程序在前台显示某一个网页时，该应用程序的主进程采用各个线程，来调用三维图形处理库或者 2D 图形引擎进行网页显示时的各个图像帧的绘制，并通过 HWC 来进行图层的融合，从而该应用程序可以在前台显示这个网页。

图 2 对终端的软件结构进行了描述，为了体现终端中的软件在处理与应用程序相关的任务时所做的工作，参见图 3，图 3 是本申请实施例提供的一种实施环境的示意图，该实施环境中包括终端 300，该终端 300 可以安装有安卓(Android)系统，该终端可以但不限于是膝上型计算机、台式计算机、移动电话、智能手机、平板电脑、多媒体播放器、电子阅读器、智能车载设备、智能家电、人工智能设备、穿戴式设备、物联网设备、或虚拟现实/增强现实/混合现实设备等。

该终端的系统架构可以包括应用层、应用程序架构(Framework)层、本地(Native)层以及内核(Kernel)层。

其中，应用层，用于提供至少一个应用程序，终端中的应用程序可以处于运行状态，也可以处于未运行状态，当任一应用程序处在运行状态时，根据运行的位置不同，应用程序可以分为前台应用程序和后台应用程序，其中，前台应用程序运行在前台，显示在终端的显示界面上，后台应用程序运行在后台，不显示在终端的显示界面上。应用层上可以具有至少一个前台应用程序和至少一个后台应用程序，每个前台应用程序或者后台应用程序对应一个用户界面(user interface, UI)线程以及一个绘制渲染(Render)线程，UI 线程也即是主进程，用于运行该前台应用程序或者后台应用程序，一个主进程中可以包括至少一个线程，每

个线程用于完成主进程中的部分任务，而绘制渲染线程（Render 线程）即是主进程中的一个线程。

Framework 层，用于提供系统服务以及提供构建应用程序时可能用到的各种应用程序编程接口（application programming interface, API），Framework 层上具有活动管理服务（activity manager service, AMS）组件，AMS 组件位于 Framework 层的系统服务（system server）中，具体地，参见图 4，图 4 是本申请实施例提供的一种终端的系统示意图，AMS 组件可以用于创建应用和切换应用，创建应用包括创建前台应用程序和创建后台应用程序，切换应用为将应用程序从后台切换至前台，或者从前台切换至后台。AMS 组件还可以用于设置应用的服务质量（Quality of Service, QoS）等级，其中，QoS 等级的设置与任务的特征有关，前台应用的 QoS 等级高于后台应用的 QoS 等级，例如，前台应用的 UI 线程和 Render 线程为延迟敏感线程，则前台应用的 UI 线程和 Render 线程的 QoS 等级可以设置为 2，而后台 QoS 应用的 UI 线程和 Render 线程不会影响到用户的任务，为延迟不敏感线程，则后台应用的 UI 线程和 Render 线程的 QoS 等级可以设置为 0。从图 4 可知，系统服务中还具有进程孵化器（Zygote），Zygote 用于创建 UI 线程以及配置 UI 线程的 QoS 等级，此时的 UI 线程为应用程序在启动阶段的 UI 线程。Framework 层还包括实现进程间通信的机制，例如 Binder 机制。

Native 层，也即是本地服务层，用于提供本地服务和链接动态库，Native 层上具有安卓运行时间（android run time, ART）虚拟机、Native 服务和 Native 链接动态库。ART 虚拟机可以为 Java 程序提供执行环境，在 ART 虚拟机的执行环境中，当两个线程竞争同一个 Java 对象同步锁时，ART 虚拟机可以在这个两个线程之间传递 QoS 等级。Native 服务和 Native 链接动态库上具有图层合成服务（surface flinger, SF）、媒体服务和 C 函数库（Libc）等，其中，SF 用于将渲染数据合成各个图层，媒体服务可以实现播放终端 300 中的视频、音乐等媒体文件，Libc 用于提供基础的 Native 层库函数，Native 层库函数可以包括用于系统调用封装的函数，终端 300 在 Native 层的 Libc 环境中可以实现在竞争同一个互斥锁（Mutex）的线程之间传递 QoS 等级。

Kernel 层，用于为终端 300 的各种硬件提供底层的驱动，Kernel 层上存在内核态同步机制，基于该内核态同步机制，在 Mutex、用户态快速互斥锁（fast userspace mutex, Futex）、读写信号量（read write semaphore）以及其他锁中，当任一线程获取锁失败时，终端 300 可以将该线程的 QoS 等级传递给持锁者线程，并且让该线程在锁等待队列上按照该线程的 QoS 等级进行排队。Kernel 层上还具有 Binder 驱动、CPU 调度模块、输入/输出（input/output, I/O）管理模块以及双倍数据速率（double data rate, DDR）内存管理模块，其中，Binder 驱动是安卓系统上进程间通信（inter-process communication, IPC）的核心模块，可以实现当具有 QoS 等级的线程请求 Binder 服务时，服务侧的线程继承请求者的 QoS 等级。CPU 调度模块用于实现安卓系统中的线程调度，CPU 调度模块中具有 QoS 调度模块，QoS 调度模块用于在线程控制块结构体中增加 QoS 记录字段，并且在线程创建时初始化 QoS 值，针对 CFS 调度类线程，QoS 调度模块还用于在任务选择时，当就绪队列中具有 QoS 任务就绪时，按照 QoS 等级选择任务执行，否则按照 CFS 调度中的优先级选择任务执行。I/O 调度模块用于实现根据提交 I/O 请求线程的 QoS 等级进行 I/O 入队以及实现限流。DDR 内存管理模块新增内存预留模块，该内存预留模块用于当较高的 QoS 等级的线程申请内存失败时，可以直接从预留的

内存中获取内存，避免进入 Linux 内核原生的慢速路径，较高的 QoS 等级的线程可以是 QoS 等级大于或等于 2 的线程。该 Kernel 层上还具有设备节点，参见图 4，设备节点中配置有 QoS 设备节点，QoS 设备节点用于配置 QoS 等级以及清理 QoS 等级，该终端 300 可以通过设备驱动程序中对设备的 I/O 通道进行管理的函数 (ioctl) 来访问 QoS 设备节点。

以上是对终端的硬件、软件以系统的描述，进一步地，为了体现终端处理任务过程，在此用一个具体的实施例来描述，参见图 5，图 5 是本申请实施例提供的一种任务处理方法的流程图，本申请实施例提供的方法流程包括：

501、终端根据每个任务的任务特征，设置该任务的服务质量等级。

任务的任务特征能够反映用户对某一项任务的服务需求，因此，不同任务的任务特征不同，那么，可以通过任务特征来为任务划分不同的服务质量等级，使得每个服务质量等级可以体现出不同的服务需求，也即是，服务质量等级越高的任务，表示用户对该任务的服务需求越高，服务质量等级越低的任务，表示用户对该任务的服务需求越低。

QoS 等级可以由字符串来表示，不同的字符串表示不同的 QoS 等级，例如，有 4 个 QoS 等级，由低到高可以分别为 1 级、2 级、3 级和 4 级，由低到高还可以分别为 A 级、B 级、C 级和 D 级。根据任务的任务特征所体现的服务需求，技术人员可以预先为不同的任务设置不同的 QoS 等级。在一种可能的实现方式中，每个任务对应一个任务标识，对于任一任务，根据该任务的任务标识所对应的任务特征，技术人员设置该任务的 QoS 等级。其中，该任务标识用于指示该任务，该任务标识可以是启动标识，启动标识用于指示启动任务，该任务标识还可以滑动标识，滑动标识用于指示滑动任务，当然，该任务标识也可以用其他任务的标识来表示，该任务标识可以用字符串来表示，不同的字符串用于指示不同的任务标识，例如，字符串“start”用于表示启动标识，字符串“slip”用于表示滑动标识。本申请实施例对该任务标识不做具体限定。

技术人员可以预先在第一 QoS 等级表中，来设置各个任务的 QoS 等级，其中，第一 QoS 等级表中可以用于记录任务的任务标识、任务特征以及 QoS 等级，参见表 1，由表 1 可知，其中，任务标识 1-4 分别代表任务 1-4，任务 1 为与可靠性体验相关的任务，其 QoS 等级为 4，与可靠性体验相关的任务可以是当终端正在运行大量的应用程序时，用户需要在前台立即运行一个还未运行的应用程序的任务，这种任务主要针对劣化恢复等极端场景的任务，例如，用户需要通过终端打电话时，但此时终端内运行的大量应用程序，会占用终端内大量的资源，但是打电话的任务必须立即执行，此时打电话的任务就是处于极端场景下的任务，用户对任务的可靠性体验要求较高。任务 2 为与交互相关的任务，其 QoS 等级为 3，用户对该任务的服务质量要求为在能效比一般的情况下，延时要求越低越好，主要要求减少反馈和加载的时延。任务 3 为与渲染相关的任务，其 QoS 等级为 2，用户对该任务的服务质量要求为在一定时限内完成，并要求在最佳能效比的情况下，延时和时延抖动越低越好。任务 4 为默认的任务为终端正常执行的任务，其 QoS 等级为 1。任务 5 为后台运行的任务，其 QoS 等级为 0。

表 1

任务标识	任务特征	QoS 等级
任务标识 1	与可靠性体验相关的任务	4
任务标识 2	与交互相关的任务	3
任务标识 3	与渲染相关的任务	2

任务标识 4	默认的任务	1
任务标识 5	后台运行的任务	0

技术人员还可以通过任务类型，来设置各个任务的 QoS 等级，任务类型用于表示任务的重要程度，重要程度越高的任务的 QoS 等级越高，重要程度越低的任务的 QoS 等级越低，技术人员可以预先在第一 QoS 等级表中，来设置各个任务的 QoS 等级，任务类型也可以用字符串表示，不同的字符串用于表示不同的任务类型，例如，任务类型“非常重要”可以用字符串“A”表示。

在一种可能的实现方式中，技术人员可以预先在第二 QoS 等级表中，来设置各个任务的 QoS 等级，其中，在第二 QoS 等级表用于记录任务的任务标识、任务类型、任务特征以及 QoS 等级，参见表 2，其中，QoS 等级为 4 的任务的任务类型可以为非常重要，QoS 等级为 3 的任务的任务类型可以为相当重要，QoS 等级为 2 的任务的任务类型可以为较为重要，QoS 等级为 1 的任务的任务类型可以为重要，QoS 等级为 0 的任务的任务类型可以为非重要。

表 2

任务标识	任务类型	任务特征	QoS 等级
任务标识 1	非常重要(critical)	与可靠性体验相关的任务	4
任务标识 2	相当重要(excellent effort)	与交互相关的任务	3
任务标识 3	较为重要(deadline effort)	与渲染相关的任务	2
任务标识 4	重要(best effort)	默认的任务	1
任务标识 5	非重要(back ground)	后台运行的任务	0

需要说明时，本步骤 501 所示的过程可以是终端在初始化时，获取到技术人员预设的各个任务的 QoS 等级的过程，终端只需获取一次即可，而终端在后续运行过程中无需执行本步骤 501，在后续处理任务过程中，直接根据初始化时获取的 QoS 等级，处理相关任务即可。

502、终端接收任务请求，该任务请求用于指示处理与应用程序相关的任务。

该终端可以基于用户的操作触发执行本步骤 502，本步骤 502 可以通过终端内的 AMS 组件来完成，以启动任务为例，用户点击终端显示界面上名称为 ABC 的应用程序的图标时，该终端可以通过该终端显示界面向 AMS 组件发送该用于启动应用程序 ABC 的任务请求，从而终端的 AMS 组件可以接收到该任务请求。

503、该终端基于该任务请求，获取该任务的服务质量等级。

该终端可以基于该任务请求中的该任务的任务标识，获取该任务的服务质量等级，在一种可能的实现方式中，当该终端在该任务请求中检测到任务标识时，在第一 QoS 等级表或第二 QoS 等级表中，查找与该任务标识对应的 QoS 等级，从而确定该任务的 QoS 等级。

在一些可能的实现方式中，终端在获取该任务的服务质量等级之前，需要先判断与该任务对应的应用程序的状态，当该应用程序在前台运行状态或未运行状态时，终端才可以获取该任务的服务质量等级。其中，应用程序的状态包括运行状态和未运行状态，其中，应用程序的运行状态包括前台运行状态和后台运行状态，前台运行状态用于指示该应用程序正在前台运行，后台运行状态用于指示该应用程序正在后台运行，未运行状态用于指示该应用程序在该终端内未运行。

该终端可以通过 AMS 组件检测该应用程序目前的状态，确定该应用程序的状态，在一种可能的实现方式中，当该终端接收到该任务请求时，该终端通过 AMS 组件检测该任务请求所

指示的应用程序的状态，当该应用程序的状态为前台运行状态或未运行状态时，该终端执行本步骤 503，当该应用程序的运行状态为后台运行状态时，则该终端不执行本步骤 503 以及后续的步骤。

需要说明的是，本申请实施例中仅以当应用程序在前台运行或未运行时，如果获取到关于该应用程序的任务请求，终端才会实时获取该任务的服务质量等级为例进行说明，也即是一旦该应用程序在后台运行时，终端则无需获取该任务的服务质量等级，在一些实施例中，还可以在后台运行时执行特定任务时，获取服务质量等级，本申请实施例对此不做限定。

504、当采用第一线程处理该任务请求对应的子任务时，该终端将该第一线程的服务质量等级配置为该任务的服务质量等级。

第一线程可以属于该应用程序的主进程，也可以是其他线程，本申请实施例对该第一线程不做具体限定。当终端在处理与该应用程序对应的任务时，可以通过该应用程序的多个线程来完成这个任务，这多个线程中的每个线程用于完成该任务中的部分任务，该部分任务也可以视为该任务中的子任务，当这多个线程中的每个线程均完成对应的子任务时，也即是，该终端处理完成该任务。

该终端可以在第一线程的线程控制块结构体内对第一线程的服务质量等级进行配置，线程控制块结构体用于存储线程的信息，例如服务质量等级、调用的线程数目等。在一种可能的实现方式中，该终端配置该应用程序的第一线程的线程控制块结构体内的服务质量等级字段为该任务的服务质量等级。

终端在运行一个应用程序时，可以基于一个主进程来运行该应用程序，且该主进程可以用于处理与该应用程序相关的任务，也即是，一个应用程序对应一个主进程，一个主进程可以处理多个任务。该主进程还可以采用至少一个进程或者至少一个线程来处理一个任务，每个进程可以包括至少一个线程，其中，主进程也可以认为是一个线程。当终端基于主进程来处理该任务时，本步骤 504 可以通过下述步骤 5041-5042 所示的过程来实现。

5041、该终端将该应用程序的主进程的服务质量等级配置为该任务的服务质量等级。

该终端可以在该主进程的进程控制块结构体设置该主进程的服务质量等级，该进程控制块结构体用于存储主进程的信息，例如服务质量等级、线程数目等。在一种可能的实现方式中，在该主进程的进程控制块结构体内的服务质量等级字段中，该终端将该主进程的服务质量等级配置为该任务的服务质量等级。

需要说明的是，该终端在设置该应用程序的主进程的服务质量等级之前，该终端需要基于该任务请求，获取该应用程序的主进程，获取该应用程序的主进程是指确定该应用程序的主进程或创建该应用程序的主进程。在一种可能的实现方式中，该终端基于该任务请求中的该任务的对象标识，获取该任务对应的主进程。该终端可以通过下述方式 1 和 2 中的任一方式，来获取该任务对应的主进程。

方式 1、当该终端检测到与该任务请求对应的主进程时，该终端直接确定该主进程。

该终端可以基于该任务请求中的该任务的对象标识，在该终端的系统中直接获取该任务对应的主进程，一个对象标识对应一个主进程。参见图 6，图 6 是本申请实施例提供的一种热启动任务处理方法的流程图，终端的显示界面向 AMS 组件发送启动任务的任务请求，当终端的 AMS 组件接收到该任务请求时，该 AMS 组件检测该终端的系统内是否存在与该对象标识对应的主进程，当该 AMS 组件在该终端的系统内检测到该主进程时，说明该终端检测到该应

用程序已启动，该终端调用开启流程（start process），通过 start process 向 Zygote 发送主进程获取请求，该主进程获取请求携带该任务的 QoS 等级以及该任务请求中的对象标识，该 Zygote 通过进程获取请求，获取与该任务请求对应的主进程，该终端将与该对象标识对应的应用程序由后台切换到前台，在该终端的显示界面上显示该应用程序的内容。

需要说明的是，上述方式 1 来确定应用程序的主进程是在应用程序已启动的场景下进行的。

方式 2、当该终端检测未到与该任务请求对应的主进程时，该终端创建与该任务请求对应的主进程。

该终端可以基于该任务请求中的该任务的任务标识以及对象标识，创建与该任务请求对应的主进程，参见图 7，图 7 是本申请实施例提供的一种冷启动任务处理方法的流程图，终端的显示界面向 AMS 组件发送启动任务的任务请求，当终端的 AMS 组件接收到该任务请求时，该 AMS 组件检测该终端的系统内是否存在与该对象标识对应的主进程，当该 AMS 组件在该终端的系统内检测不到该主进程时，说明该应用程序还未启动，该终端调用 start process，该终端通过 start process 向 Zygote 发送主进程创建请求，该主进程创建请求携带该任务的 QoS 等级、冷启动标识以及该任务请求中的对象标识，该冷启动标识用于指示待创建的主进程用于冷启动，该冷启动标识可以存储在扩展的 start process 参数字段中，该 Zygote 基于该主进程创建请求，创建与该任务请求对应的主进程。本申请实施例对该冷启动标识的表示形式不做具体限定。

需要说明的是，该终端还可以基于其他方式，获取与该任务请求对应的主进程，本申请实施例对获取该主进程的其他方式不做限定，且通过方式 2 来确定应用程序的主进程是在应用程序未启动的场景下进行的。

该终端可以通过 Zygote 执行本步骤 5041，主进程的 QoS 等级由主进程待处理的任务的 QoS 等级来决定，参见图 7，当该 Zygote 基于主进程创建请求，创建完成该主进程时，该主进程当前用于处理该任务请求，则该 Zygote 将主进程创建请求中的 QoS 等级配置为该主进程的 QoS 等级。

需要说明的是，不同的任务请求对应于不同的任务质量等级，那么，当终端通过应用程序的第一线程处理不同的任务请求时，该第一线程的服务质量等级对应与不同的任务质量等级，该应用程序的前台运行状态不同，该前台运行状态包括启动状态、运行中状态和滑动状态。其中，启动状态用于指示应用程序在启动过程中，运行中状态用于指示应用程序在运行中，滑动状态用于指示应用程序终端的显示界面上滑动。参见图 7，当与主进程获取请求中的对象标识对应的应用程序中的内容在显示界面显示时，说明该终端启动应用程序的任务已经完成，则该应用程序不处于启动状态，当该应用程序处于运行中状态时，该终端可以重新配置该应用程序的主进程的 QoS 等级和线程的 QoS 等级，例如，在渲染场景下，配置某应用程序的渲染任务的 QoS 等级为 2，启动任务的 QoS 等级为 3 以及后台运行任务的 QoS 等级为 0，这样，当终端在启动状态时，将主进程的服务质量等级配置 3（即与启动任务相同），当该终端开始渲染任务时，则该 Zygote 将该主进程以及渲染进程的 QoS 等级配置 2（即与渲染任务相同）。

需要说明的是，当处理该任务时，如果该应用程序由前台运行切换至后台运行，该终端将该应用程序与该任务对应的线程服务质量等级配置为第一目标等级，其中，第一目标等级

可以是服务质量等级中较低的任一等级，例如表 1 中的 0 级，本申请实施例对该第一目标等级不做具体限定。需要说明的是，当终端在处理该任务请求时，将该任务请求对应的应用程序由前台运行切换到后台运行，该终端仅配置与该任务请求对应的线程的服务质量等级，而不配置与其他任务请求对应的线程的服务质量等级。

在一种可能的实现方式中，当处理该任务请求对应的任务时，该终端如果检测与任一应用程序前台运行切换至后台运行时，该终端通过 Zygote 将与该应用程序的主进程或线程的 QoS 等级配置为零，其中，该应用程序的主进程或线程与该任务请求对应。仍以图 6 为例，当应用程序中的内容显示在显示界面显示时，表示该显示界面原来显示的应用程序由前台运行切换为后台运行，那么，该终端通过 Zygote 将与任务请求对应的线程的 QoS 等级配置为零，而将前台运行的与其他任务请求对应的线程配置为 2。

步骤 5042、当该应用程序的主进程采用任一线程执行该任务请求对应的子任务时，该终端将该线程的服务质量等级配置为该任务的服务质量等级。

其中，该主进程采用的任一线程可以是该主进程内的任一线程，也可以是该主进程调用的任一线程，且当该任一线程通过调用其他线程处理该任务请求对应的子任务时，该终端将该其他线程的服务质量等级配置为该任一线程的服务质量等级。

在一种可能的实现方式中，Zygote 基于该主进程创建请求创建的主进程为 UI 线程，UI 线程在启动过程中创建 Render 线程，当 Zygote 检测到 Render 线程创建完成后，该 Zygote 将该 Render 线程的 QoS 等级配置为该 UI 线程的 QoS 等级。

在一种可能的实现方式中，当主进程在执行过程中，可以根据 Binder 同步调用机制，向该终端内的 Binder 驱动发送线程调用请求，当 Binder 驱动确定该主进程所调用的目标线程时，该终端通过该主进程将该目标线程的服务质量等级配置为该主进程的服务质量等级，当该终端执行完该目标线程时，该目标线程通过 Binder 驱动向该主进程发送执行结果，且该目标线程将自己的服务质量等级更新为自己原来的 QoS 等级。

需要说明的是，该主进程采用的任一线程还可以调用其他线程，当该任一线程在调用其他线程时，该终端将其他线程的 QoS 等级配置为该任一线程的服务质量等级。在一种可能的实现方式中，当该线程调用其他线程时，在该其他线程的线程控制块结构体内的服务质量等级字段中，将该其他线程的服务质量等级配置为该任务的服务质量等级。

参见图 8，图 8 是本申请实施例提供的一种同步服务质量等级的流程图，线程 A 向 Binder 驱动发送 Binder 请求，其中，线程 A 的服务质量等级为 3，该 Binder 请求也即是线程调用请求，在 Binder 驱动中的 binder_transaction 函数中，选择 Binder 线程作为被调用的目标线程，Binder 线程为可以被线程 A 调用的任一线程，当 Binder 线程没有调用时，Binder 线程处于休眠中，其服务质量等级为 0，当被 Binder 线程被线程 A 调用时，线程 A 将 Binder 线程的 QoS 等级由 0 配置为 3（即是线程 A 的 QoS 等级），当 Binder 线程运行时，Binder 线程的 QoS 等级为 3，当该终端执行 Binder 线程，该终端暂不执行线程 A，线程 A 处于休眠中，当 Binder 线程执行完成时，Binder 线程通过 Binder 驱动向线程 A 发送执行结果，Binder 线程将自己的 QoS 等级配置为原来的 QoS 等级 0，当线程 A 接收到执行结果后，该终端继续执行线程 A。

505、该终端基于该第一线程的服务质量等级对应的处理顺序，处理该第一线程对应的子任务。

该终端在处理该任务请求时，可以生成该任务的多个子任务，终端可以使用至少一个线程或至少一个进程来完成这多个子任务，每个线程用于完成多个子任务的部分子任务。

处理顺序可以由该第一线程的服务质量等级的高低来确定，在一种可能的实现方式中，将该子任务插入到处理队列中服务质量等级低于该第一线程的服务质量等级或不具有服务质量等级的子任务之前，按照该处理队列中的顺序，处理该第一线程对应的子任务。

当线程的优先级类型有多种时，QoS 类优先级高于其他类型的优先级，QoS 类优先级也即是用 QoS 等级表示的优先级，例如，优先级类型包括先入先出（first in first out, FIFO）类优先级和 QoS 类优先级，其中，FIFO 类优先级遵循先入先出的原则，也即是，任一携带 FIFO 优先级的线程的子任务先进入到处理队列中，相对起携带 FIFO 优先级的线程的子任务，该终端优先处理先进入处理队列的子任务。当处理队列中的子任务所对应的线程具有同一类型的优先级时，该终端优先处理等级最高的线程的子任务。

那么，通过为主进程以及与该任务相关的线程配置相同的 QoS 等级，可以使得该终端在处理该任务请求时采用的所有线程以及进程的 QoS 等级相同，使得被采用的所有线程处理子任务时，都可以基于相同的 QoS 等级处理队列中排队，从而可以避免因为任一被调用的线程的优先级较低，而导致的排队耗时较长的问题，从而避免处理该任务时由于这一问题而造成的卡顿情况。

该终端可以在该处理队列中获取到该第一线程对应的子任务后，对该第一线程对应的子任务进行处理。在一种可能的实现方式中，当该第一线程对应的子任务在该处理队列的队头时，该终端获取该第一线程对应的子任务，对该第一线程对应的子任务进行处理。当终端在处理该子任务时，该子任务从该处理队列中出列，当该终端处理完该子任务时，该终端在该处理队列的队头中获取下一个线程的子任务。

需要说明的是，当该终端执行完与任务请求相关的所有线程时，也即是该终端处理完成该任务请求。

506、当该子任务的目标资源已加锁，该终端将第二线程的服务质量等级配置为该第一线程的服务质量等级，该第二线程为对该目标资源进行加锁的线程。

需要说明的是，当该第二线程的原 QoS 等级低于该第一线程的 QoS 等级时，该终端执行本步骤 506，否则，该终端则不执行本步骤 506。

在一种可能的实现方式中，该终端可以通过下述步骤 506A-506C 所示的过程完成本步骤 507。

步骤 506A、当执行该第一线程对应的子任务时，该终端对该目标资源加锁。

该终端可以通过该第一线程执行本步骤 507A。该终端在对目标资源加锁之前，该终端会对该目标资源进行检测，当该目标资源的状态加锁状态时，该第一线程则不能对该目标资源加锁，当该终端检测到该目标资源的状态为未加锁状态时，该第一线程可以对该目标资源进行加锁。其中，该加锁状态用于指示该目标资源已上锁，未加锁状态用于指示该目标资源未上锁。

步骤 506B、当对该目标资源加锁失败时，该终端确定该第二线程。

该终端可以通过该第一线程执行本步骤 506B。当该第一线程检测到该目标资源的状态为加锁状态时，也即是，该第一线程对该目标资源加锁失败。

该终端可以通过目标函数，来确定该第二线程，在一种可能的实现方式中，终端通过锁

对象, 采用该目标函数来确定该第二线程, 锁对象为目标资源上的锁的内存数据, 该内存数据中存储有可以指示第一线程的信息, 目标函数用于获取锁对象的内存数据中存储的第二线程的信息。例如, 目标资源上的锁为互斥锁, 终端可以通过目标函数 `mutex_lock (A)` 来确定第二线程, 其中, A 为锁对象, A 中存储有指示第二线程的信息。再例如, 目标资源上的锁为读写信号量, 终端可以通过目标函数 `rwsem (B)` 来确定第二线程, 其中, B 为锁对象, B 中存储有指示第二线程的信息。

步骤 506C、当该第二线程的服务质量等级低于该第一线程的服务质量等级时, 该终端将该第二线程的服务质量等级更新为该第一线程的服务质量等级。

该终端可以通过该第一线程执行本步骤 506C。该第一线程通过更新该第二线程的线程控制块结构体内的服务质量等级字段中的内容, 来实现更新第二线程的服务质量等级。在一种可能的实现方式中, 在该第二线程的线程控制块结构体内的服务质量等级字段中, 该第一线程将该第二线程的服务质量等级配置为该第一线程的服务质量等级。

通过将该第二线程的服务质量等级更新成该第一线程的服务质量等级, 可以使得第二线程的服务质量等级与该第一线程的服务质量等级相同, 从而当该第二线程释放目标资源的锁时, 该第一线程可以立即对该目标资源加锁。

507、当该第二线程释放该目标资源的锁时, 该终端基于该目标资源处理该第一线程对应的子任务。

该终端可以通过该第一线程执行本步骤 507, 当第一线程加锁失败时, 该终端基于该第一线程的服务质量等级, 在目标资源的等待队列中排队等待, 当该第二线程释放该目标资源的锁后, 该第一线程基于该目标资源处理该第一线程对应的子任务, 目标资源的等待队列用于存储各用于存储待处理该目标资源的线程。

其中, 该第一线程基于该第一线程的服务质量等级, 在目标资源的等待队列中排队等待的过程与第一线程的子任务在处理队列中的排队等待的过程同理, 在此本申请实施例对该第一线程基于该第一线程的服务质量等级, 在目标资源的等待队列中排队等待的过程不做赘述。

在一种可能的实现方式中, 本步骤 507 可以通过下述步骤 507A-507B 所示的过程来实现。

步骤 507A、当该第二线程释放该目标资源的锁之后, 该终端按照该目标资源的等待队列中顺序, 对该目标资源加锁, 恢复该第二线程的原服务质量等级, 该目标资源的等待队列用于存储待处理该目标资源的线程。

该终端可以通过该第一线程对目标资源加锁, 该终端可以通过第二线程恢复该第二线程的原服务质量等级, 其中, 该第二线程的原服务质量等级为该第一线程配置该第二线程的服务质量等级之前的服务质量等级。

当任一线程释锁资源的锁后, 在该目标资源的等待队列中该任一线程之后, 且与该任一线程相邻的线程对该目标资源加锁。在一种可能的实现方式中, 当该第一线程将该目标资源的状态更新为加锁状态时, 该第一线程对该目标资源加锁成功。

当第二线程的 QoS 等级较低, 或没有 QoS 等级时, 若没有重新配置第二线程的 QoS 等级, 那么, 第二线程释放锁后, 与第二线程同等级的线程就会对目标资源进行上锁, 而该第一线程需要在目标资源的队列中等待低 QoS 等级的线程持有的锁均释放完成后, 该第一线程才能根据在该等待队列中的位置对目标资源进行加锁, 从而导致优先级反转的情况, 而当该终端将该第二线程的 QoS 等级配置为与该第一线程相同的 QoS 等级时, 因为第二线程当前持有锁,

那么，第一线程在等待队列中排队时，与第二线程相邻，也可以认为处于等待队列的队头，当第二线程释放锁之后，该第一线程就可以立即对该目标资源加锁，进而可以避免优先级反转的情况。

步骤 507B、当对该目标资源加锁完成后，该终端基于该目标资源处理该子任务。

该终端可以通过该第一线程执行本步骤 507B，当第一线程在执行该任务请求时，如果需对该目标资源进行目标处理，则在该第一线程对该目标资源加锁后，该第一线程就可以对该目标资源进行目标处理，该目标处理可以是任何处理方式，例如，读数据或者写数据。

为了进一步说明步骤 506-507 所示的过程，参见图 9，图 9 是本申请实施例提供的一种同步互斥锁的流程图。由图 9 可知，线程 A 当前的 QoS 等级为 3，当线程 A 对目标资源加互斥锁失败时，根据目标资源，线程 A 可以在通过 mutex_lock 函数，找到第二线程，该第二线程为线程 B，该线程 A 将线程 B 的线程控制块中 QoS 字段中的内容设置为线程 A 的线程控制块中 QoS 字段中的内容，即将线程 B 的 QoS 等级设置为 3；线程 A 在该目标资源的等待队列上按照自己的 QoS 等级进行排队；等线程 B 线程释放锁时，线程 B 将自己的线程控制块中 QoS 字段中的内容修改为原来的内容，线程 A 对目标资源进行加锁，并基于目标资源处理线程 A 对应的子任务。

图 9 为同步互斥锁的流程，在一些实施例中，终端还可以同步读写信号量，参见图 10，图 10 是本申请实施例提供的一种同步读写信号量的流程图。由图 10 可知，线程 A 当前的 QoS 等级为 3，当线程 A 对目标资源加读写信号量失败时，线程 A 通过 rwsem 函数，线程 A 可以找到第二线程，该第二线程为线程 B，线程 A 将线程 B 的线程控制块中 QoS 字段的内容设置为线程 A 的线程控制块中 QoS 字段中的内容，即将线程 B 的 QoS 等级设置为 3；线程 A 在等待队列上按照自己的 QoS 等级进行排队；等线程 B 线程释放锁时，线程 B 将自己的线程控制块中 QoS 字段中的内容修改为原来的内容。由于对于任一个具有读写信号量的锁资源，可以同时处理多个读请求或者一个写请求，对于处理读请求时可以不执行步骤 506-507。

为了进一步说明步骤 501-507 所示的过程，参见图 11，图 11 是本申请实施例提供的一种基于安卓系统架构任务处理过程的示意图。图 11 用于说明服务质量等级在系统不同软件层之间的传递过程，如图 11 所示，UI 线程通过 IPC 机制将服务质量等级传递至应用程序内其他线程，其他线程对应的执行段标记成与 UI 线程相同，当 IPC 依赖解除时，线程恢复服务质量等级，执行段的标记恢复。本申请实施例通过图 11 中的 2 个过程来说明在执行线程时，如果出现线程调用或目标资源已上锁时，终端对服务质量等级进行传递的过程，过程 1 为服务质量等级为 2 时的服务质量等级的传递过程，过程 2 为服务质量等级为 1 时的服务质量等级的传递过程，过程 1 包括下述步骤 1.1-1.6。

步骤 1.1、前台应用程序的 UI 线程为延迟敏感线程，终端将延迟敏感线程的服务质量等级配置为 QoS=2，也即是该 UI 线程的服务质量等级配置为 2。

步骤 1.2、该 UI 线程通过 Binder 系统中 IPC 的同步请求来调用线程，被调用的线程为 UI 线程执行过程中所采用的线程，例如安卓核心服务中的 Binder 线程，当 Binder 线程确定后，该 UI 线程将 Binder 线程的服务质量等级配置为 2，当该 Binder 线程在被调用过程中需要调用应用管理服务中的线程时，该 Binder 线程将被调用线程的服务质量等级配置为 2，当 Binder 线程调用该线程结束后，该线程恢复到原来的服务质量等级，当 UI 线程调用 Binder 线程结束后，Binder 线程恢复到原来的服务质量等级。

步骤 1.3、当 UI 线程通过 IPC 的同步请求调用应用程序自定义服务 Binder 线程，该 UI 线程将应用程序自定义服务 Binder 线程的服务质量等级配置为 2，当调用结束后，应用程序自定义服务 Binder 线程恢复到原来的服务质量等级。

步骤 1.4、UI 线程还可以调用逻辑线程，该 UI 线程将该逻辑线程的服务质量等级配置为 2，当该逻辑线程在被调用过程中，该逻辑线程需要使用用户态锁所指示的目标资源，但是应用程序自定义服务 Binder 线程持该有用户态锁，则逻辑线程将应用程序自定义服务 Binder 线程的服务质量等级配置为 2，待该应用程序自定义服务 Binder 线程释放锁后，逻辑线程对目标资源进行上锁，该应用程序自定义服务 Binder 线程恢复原先的服务质量等级。

步骤 1.5、当 Binder 线程调用应用程序管理服务中的线程时，如果该线程在被调用的过程中，需要使用 Java 锁所指示的目标资源，但是该应用程序管理服务中的其他线程持有该 Java 锁，被调用的线程将该其他线程的服务质量等级配置为 2，待其他线程释放该 Java 锁后，该线程对该目标资源进行上锁，其他线程恢复到原先的服务质量等级。

步骤 1.6、当应用程序管理服务中的线程通过系统调用来调用驱动时，如果该驱动被上锁，该线程将对该驱动上锁的线程的服务质量等级配置为 2，当对驱动上锁的线程释放锁后，应用程序管理服务中的线程对该驱动进行上锁，持锁线程恢复到原先的服务质量等级。

以上是对过程 1 的描述，而过程 2 包括下述步骤 2.1-2.5。

步骤 2.1、终端将绘制渲染线程的服务质量等级配置为 1。

步骤 2.2、当绘制渲染线程通过 IPC 的同步请求调用安卓核心服务中的 Binder 线程时，绘制渲染线程将该 Binder 线程的服务质量等级配置为 1，当调用 Binder 线程结束后，Binder 线程恢复原先的服务质量等级。

步骤 2.3、当该 Binder 线程在被调用过程中，通过 IPC 的同步请求调用 SF 中的线程时，该 Binder 线程将 SF 中的线程的服务质量等级配置为 1，当调用 SF 中的线程结束后，SF 中的线程恢复原先的服务质量等级。

步骤 2.4、当终端执行绘制渲染线程过程中，需要使用用户态锁所指示的目标资源时，逻辑线程已对该目标资源上锁，绘制渲染线程将逻辑线程的服务质量等级配置为 1，待逻辑线程释放锁后，绘制渲染线程对该目标资源进行上锁，逻辑线程恢复原先的服务质量等级。

步骤 2.5、当终端执行绘制渲染线程过程中，通过系统调用来调用内存资源时，如果内存资源被上锁，该绘制渲染线程将内存资源上锁的线程的服务质量等级配置为 1，当对内存资源上锁的线程释放锁后，该绘制渲染线程对该内存资源进行上锁，持锁线程恢复到原先的服务质量等级。

为了进一步表说明图 8-10 所示的过程，参见本申请实施例提供如图 12 所示的滑动场景下进行服务质量等级传递的流程图，该流程主要包括下述步骤 1-6，

步骤 1、终端接收到滑动操作后，终端就会接收到滑动任务，也即是应用程序由后台切换至前台运行，该终端处理该滑动任务时，将该应用程序的 UI 线程的服务质量等级设置为 3，由于应用程序在后台运行时，服务质量等级较低，以后台运行的应用程序的线程的服务质量等级为 2 为，当线程睡眠时，则睡眠线程的服务质量等级为 2，Binder 线程 A 休眠时服务质量等级为 2，当 UI 线程通过 Binder 进行同步调用来调用系统服务中的 Binder 线程 A 时，UI 线程将 Binder 线程 A 的服务质量等级由 2 更新为 3。由于 UI 线程此时正在调用 Binder 线程 A，则 UI 线程处于睡眠状态。

步骤 2、Binder 线程 A 在执行过程中需要使用用户态锁所指示的锁资源时，但是持锁线程持该有用户态锁，持锁线程的服务质量等级为 2，则 Binder 线程 A 将持锁线程的服务质量等级更新为 3，Binder 线程 A 处于睡眠状态。

步骤 3、当持锁线程释放用户态锁后，持锁线程恢复原先的服务质量等级 2，并唤醒 Binder 线程 A。

对于其他线程也可能同时需要获取该用户态锁，例如图 12 中的锁竞争线程，锁竞争线程的服务质量等级为 2，但是，用户态锁被持锁线程持有该用户态锁，那么，此时，锁竞争线程处于休眠状态，若持锁线程释放锁后，唤醒该锁竞争线程，则锁竞争线程原先的服务质量等级 2。

步骤 4、Binder 线程 A 获取到用户态锁后，按照服务质量等级 3 继续执行。

步骤 5、当 UI 线程执行过程中，需要调用内核里的资源，如果资源已经被内核线程上锁，内核线程的服务质量等级为 2，则 UI 线程将内核线程的服务质量等级更新为 3，UI 线程处于睡眠状态。

步骤 6、当内核线程释放锁后，内核线程唤醒 UI 线程，内核线程恢复到原先的服务质量等级 3。

508、当该第一线程调用第三线程时，该终端根据该第三线程中所缓存的该第一线程的数据量，调整该第一线程和该第三线程的服务质量等级。

该第三线程为被该第一线程调用的任一线程。该终端可以每经过预设时长，对第三线程中所缓存的该第一线程的数据量进行统计，该终端基于统计出的该第一线程的数据量与数据量阈值之间的关系，来调整该第一线程和该第三线程的 QoS 等级，在一种可能的实现方式中，当该第三线程中所缓存的该第一线程的数据量小于数据量阈值时，该终端保持该第一线程和该第三线程的 QoS 等级不变；当该第三线程中所缓存的该第一线程的数据量等于或大于数据量阈值时，该终端降低该第一线程的 QoS 等级。

为了进一步说明本步骤 508，参见图 13，图 13 为本申请实施例提供的一种渲染场景下动态调整线程的 QoS 等级的模型。

从图 13 中可知，该终端系统上前台应用程序的绘制渲染模型为一个流水线（PIPELINE）模型，应用程序的 UI 线程、Render 线程为上游生产者，用于生产渲染数据，Render 线程通过 Binder，将生产的渲染数据发送给 SF，相对于应用程序来说，SF 是下游消费者，其作用是合成各个图层，SF 将合成图层的数据发送给 HWC，对于 HWC 来说 SF 是生产者，HWC 是消费者。

由于终端在执行应用程序的 UI 线程和 Render 线程的 QoS 等级之前，该终端已经配置完成的 UI 线程和 Render 线程的 QoS 等级，假设 UI 线程和 Render 线程的 QoS 等级为 2，那么，当 SF 中缓存的渲染数据的数据量小于数据量阈值时，该终端保持 SF 和 Render 线程的 QoS 等级保持不变也即是 2，当 SF 中缓存的渲染数据的数据量等于或者大于数据量阈值时，说明消费者 SF 消费的太慢或者 Render 线程生产太快，该终端将 Render 线程的 QoS 等级降为 1。

当该终端执行第一线程过快时，导致该终端执行该第一线程时，该第一线程所使用的资源过多，所产生的数据资源也过多，从而造成不必要的功耗，当该终端执行第一线程过慢时，导致该第一线程所产生的数据资源过少，不够下层线程使用。而通过本步骤 508 来动态调整线程的 QoS 等级，使得该终端执行第一线程可以保持在做一个最合适的速度上，从未可以避

免该第一线程在使用系统中资源时，造成不必要的功耗，或者不够下层线程使用的情况。

在一种可能的实现方式中，当该第一线程调用该第三线程时，如果该第三线程中所缓存的该第一线程的数据量小于数据量阈值，该终端提高该第一线程的服务质量等级；当该第一线程调用该第三线程时，如果该第三线程中所缓存的该第一线程的数据量等于或大于该数据量阈值，该终端提高第三线程的服务质量等级。

那么，当该第三线程中所缓存的所述线程的数据量小于数据量阈值时，终端提高该第一线程的服务质量等级，从而使得第三线程中该第一线程的数据量达到数据量阈值，那么，当该第三线程突然消耗过快时，可以避免该第三线程中该第一线程的数据量跟不上消耗进度。

当该第三线程中所缓存的该第一线程的数据量等于或大于该数据量阈值时，终端提高该第三线程的服务质量等级，从而使得第三线程中所缓存的该第一线程的数据量消耗的速度加快，从而可以避免该第一线程产生的数据在该第三线程中积累。

需要说明的是，当终端在处理该任务请求过程中都可以不断执行本步骤 508，以便随时调整处理任务请求时各个线程的 QoS 等级。

509、当该第一线程申请内存资源失败，且该第一线程的服务质量等级大于第二目标等级时，该终端从预留内存资源中为该第一线程分配内存资源。

第二目标等级可以是服务质量较大的等级，例如，第二目标等级为 2 级，本申请实施例对该第二目标等级不做具体限定。

该预留内存资源为终端提前预留出的用于应急的内存资源，在一种可能的实现方式中，当该终端在进行内存资源初始化时，该终端将该终端的内存资源的部分内存资源设置为预留内存资源，以便服务质量等级较高的线程可以直接使用该预留内存资源。

当该终端在执行该第一线程时，需要内存资源存储执行该第一线程过程中需要的数据或者生成的数据时，该第一线程会向该终端申请内存资源，当该终端内为线程提供的内存被占用完，且该第一线程的服务质量等级大于第二目标等级时，该第一线程申请内存资源失败，此时，该终端可以从该预留内存资源中为该第一线程分配内存资源，从而使得该第一线程可以使用该分配的内存资源，从而可以避免该第一线程等待内存资源被释放，终端才能从释放的内存资源中为该第一线程分配内存资源，从而提高了处理该第一线程的效率，因为，该第一线程为服务质量等级较高的线程，从而可以提高处理任务的效率。

本申请实施例提供的方法，通过设置任务的服务质量等级，该服务质量等级与用户感知度相关，并将执行该任务的线程的优先级配置为该任务的服务质量等级，线程的优先级用于指示线程在系统中被执行的执行顺序。通过这样的方法使得线程的执行顺序符合该任务的服务质量等级。这样服务质量等级越高的任务（即用户感知度越强的任务），执行的就越快，从而可以一定程度上避免任务处理时间过长导致的卡顿现象。并且，动态调整该终端在处理任务请求时的所执行的线程的 QoS 等级，使得该终端执行第一线程可以保持在做一个最合适速度上，从未可以避免该第一线程在使用系统的资源时，造成不必要的功耗，或者不够下层线程使用的情况。并且，当该第一线程从预留内存资源中获取内存资源，可以提高处理任务的效率。

图 14 是本申请实施例提供的一种任务处理装置的结构装置图，该装置包括：

接收模块 1401，用于执行上述步骤 502；

获取模块 1402，用于执行上述步骤 503；

第一配置模块 1403, 用于执行上述步骤 504;

第一处理模块 1404, 用于执行上述步骤 505。

可选地, 第一处理模块 1404, 用于将所述子任务插入到处理队列中服务质量等级低于所述线程的服务质量等级或不具有服务质量等级的子任务之前, 按照所述处理队列中的顺序, 处理所述子任务。

可选地, 第一处理模块 1404 还包括:

配置单元, 用于执行上述步骤 506;

处理单元, 用于执行上述步骤 507。

可选地, 所述配置单元用于执行上述步骤 506A-506C。

可选地, 所述处理单元用于上述步骤 507A-507B。

可选地, 所述装置还包括:

调整模块, 用于执行步骤 508。

可选地, 所述调整模块用于:

当所述第一线程调用所述第三线程时, 如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值, 保持所述第一线程和所述第三线程的服务质量等级不变;

当所述第一线程调用所述第三线程时, 如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值, 降低所述第一线程的服务质量等级。

可选地, 所述调整模块用于:

当所述第一线程调用所述第三线程时, 如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值, 提高所述第一线程的服务质量等级;

当所述第一线程调用所述第三线程时, 如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值, 提高所述第三线程的服务质量等级。

可选地, 所述任务的服务质量等级由所述任务的任务特征确定

可选地, 所述装置还包括:

第二配置模块, 用于当处理所述任务时, 如果所述应用程序由前台运行切换至后台运行, 将与所述任务对应的线程的服务质量等级配置为第一目标等级。

可选地, 所述第一配置模块, 用于配置所述第一线程的线程控制块结构体内的服务质量等级字段为所述任务的服务质量等级, 所述线程控制块结构体用于存储线程的信息。

可选地, 所述装置还包括:

分配模块, 用于执行上述步骤 509。

可选地, 所述装置还包括:

对应模块, 用于不同的任务请求对应于不同的任务质量等级。

上述所有可选技术方案, 可以采用任意结合形成本公开的可选实施例, 在此不再一一赘述。

需要说明的是: 上述实施例提供任务处理装置在进行任务处理时, 仅以上述各功能模块的划分进行举例说明, 实际应用中, 可以根据需要而将上述功能分配由不同的功能模块完成, 即将装置的内部结构划分成不同的功能模块, 以完成以上描述的全部或者部分功能。另外, 上述实施例提供的任务处理装置与任务处理方法实施例属于同一构思, 其具体实现过程详见方法实施例, 这里不再赘述。

另外，本申请提供了一种计算机程序产品（或称计算机程序），该计算机程序产品中存储有至少一条指令，该指令由处理器加载并执行以实现如上述任务处理方法所执行的操作。

综上所述，本申请提供的方案根据任务重要性定义和任务相关的 QoS 等级，并将该 QoS 等级设置为实现该任务的线程的 QoS 等级，从而提高重要任务的执行效率。进一步的，根据系统中的同步 IPC 调用（Binder，Socket 等机制）、锁依赖等实现线程 QoS 等级在线程之间的传递，在线程依赖链上的线程能够保证 QoS 一致性，避免反转和丢失。进一步的，基于当前系统绘制的消费者模型反向调整 QoS，避免资源过度供给，带来功耗开销。

本领域普通技术人员可以理解实现上述实施例的全部或部分步骤可以通过硬件来完成，也可以通过程序来指令相关的硬件完成，所述的程序可以存储于一种计算机可读存储介质中，上述提到的存储介质可以是只读存储器，磁盘或光盘等。

以上所述仅为本申请的示例性实施例，并不用以限制本申请，凡在本申请的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本申请的保护范围之内。

权 利 要 求 书

1、一种任务处理方法，其特征在于，所述方法包括：

接收任务请求，所述任务请求用于指示处理与应用程序相关的任务；

基于所述任务请求，获取所述任务对应的服务质量等级；

当采用第一线程处理所述任务请求对应的子任务时，将所述第一线程的服务质量等级配置为所述任务的服务质量等级；

基于所述第一线程的服务质量等级对应的处理顺序，处理所述第一线程对应的子任务。

2、根据权利要求1所述的方法，其特征在于，所述基于所述第一线程的服务质量等级对应的处理顺序，处理所述第一线程对应的子任务包括：

将所述子任务插入到处理队列中服务质量等级低于所述线程的服务质量等级或不具有服务质量等级的子任务之前，按照所述处理队列中的顺序，处理所述子任务。

3、根据权利要求1至2任一项所述的方法，其特征在于，所述处理所述第一线程对应的子任务包括：

当所述子任务的目标资源已加锁时，将第二线程的服务质量等级配置为所述第一线程的服务质量等级，所述第二线程为对所述目标资源进行加锁的线程；

当所述第二线程释放所述目标资源的锁时，基于所述目标资源处理所述子任务。

4、根据权利要求3所述的方法，其特征在于，所述当所述子任务的目标资源已加锁时，将第二线程的服务质量等级配置为所述第一线程的服务质量等级包括：

当执行所述第一线程对应的子任务时，对所述目标资源加锁；

当对所述目标资源加锁失败时，确定所述第二线程；

当所述第二线程的服务质量等级低于所述第一线程的服务质量等级时，将所述第二线程的服务质量等级更新为所述第一线程的服务质量等级。

5、根据权利要求3所述的方法，其特征在于，所述当所述第二线程释放所述目标资源的锁时，基于所述目标资源处理所述子任务包括：

当所述第二线程释放所述目标资源的锁之后，按照所述目标资源的等待队列中顺序，对所述目标资源加锁，恢复所述第二线程的原服务质量等级，所述目标资源的等待队列用于存储待处理所述目标资源的线程；

当对所述目标资源加锁完成后，基于所述目标资源处理所述子任务。

6、根据权利要求1至5任一项所述的方法，其特征在于，所述方法还包括：

当所述第一线程调用第三线程时，根据所述第三线程中所缓存的所述第一线程的数据量，

调整所述第一线程和所述第三线程的服务质量等级。

7、根据权利要求6所述的方法，其特征在于，所述当所述第一线程调用第三线程时，根据所述第三线程中所缓存的所述第一线程的数据量，调整所述第一线程和所述第三线程的服务质量等级，包括：

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值，保持所述第一线程和所述第三线程的服务质量等级不变；

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值，降低所述第一线程的服务质量等级。

8、根据权利要求6所述的方法，其特征在于，所述当所述第一线程调用第三线程时，根据所述第三线程中所缓存的所述第一线程的数据量，调整所述第一线程和所述第三线程的服务质量等级，包括：

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值，提高所述第一线程的服务质量等级；

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值，提高所述第三线程的服务质量等级。

9、根据权利要求1至8任一项所述的方法，其特征在于，所述任务的服务质量等级由所述任务的任务特征确定。

10、根据权利要求1至9任一项所述的方法，其特征在于，所述方法还包括：

当处理所述任务时，如果所述应用程序由前台运行切换至后台运行，将与所述任务对应的线程的服务质量等级配置为第一目标等级。

11、根据权利要求1至10任一项所述的方法，其特征在于，所述将所述第一线程的服务质量等级配置为所述任务的服务质量等级包括：

配置所述第一线程的线程控制块结构体内的服务质量等级字段为所述任务的服务质量等级，所述线程控制块结构体用于存储线程的信息。

12、根据权利要求1至11任一项所述的方法，其特征在于，所述方法还包括：

当所述第一线程申请内存资源失败，且所述第一线程的服务质量等级大于第二目标等级时，从预留内存资源中为所述第一线程分配内存资源。

13、根据权利要求1至12任一项所述的方法，其特征在于，不同的任务请求对应于不同的任务质量等级。

14、一种任务处理装置，其特征在于，所述装置包括：

接收模块，用于接收任务请求，所述任务请求用于指示处理与应用程序相关的任务；

获取模块，用于基于所述任务请求，获取所述任务对应的服务质量等级；

第一配置模块，用于当采用第一线程处理所述任务请求对应的子任务时，将所述第一线程的服务质量等级配置为所述任务的服务质量等级；

第一处理模块，用于基于所述第一线程的服务质量等级对应的处理顺序，处理所述第一线程对应的子任务。

15、根据权利要求 14 所述的装置，其特征在于，所述第一处理模块，用于将所述子任务插入到处理队列中服务质量等级低于所述线程的服务质量等级或不具有服务质量等级的子任务之前，按照所述处理队列中的顺序，处理所述子任务。

16、根据权利要求 14 至 15 任一项所述的装置，其特征在于，所述第一处理模块包括：

配置单元，用于当所述子任务的目标资源已加锁时，将第二线程的服务质量等级配置为所述第一线程的服务质量等级，所述第二线程为对所述目标资源进行加锁的线程；

处理单元，用于当所述第二线程释放所述目标资源的锁时，基于所述目标资源处理所述子任务。

17、根据权利要求 16 所述的装置，其特征在于，所述配置单元用于：

当执行所述第一线程对应的子任务时，对所述目标资源加锁；

当对所述目标资源加锁失败时，确定所述第二线程；

当所述第二线程的服务质量等级低于所述第一线程的服务质量等级时，将所述第二线程的服务质量等级更新为所述第一线程的服务质量等级。

18、根据权利要求 16 所述的装置，其特征在于，所述处理单元用于：

当所述第二线程释放所述目标资源的锁之后，按照所述目标资源的等待队列中顺序，对所述目标资源加锁，恢复所述第二线程的原服务质量等级，所述目标资源的等待队列用于存储待处理所述目标资源的线程；

当对所述目标资源加锁完成后，基于所述目标资源处理所述子任务。

19、根据权利要求 14 至 18 任一项所述的装置，其特征在于，所述装置还包括：

调整模块，用于当所述第一线程调用第三线程时，根据所述第三线程中所缓存的所述第一线程的数据量，调整所述第一线程和所述第三线程的服务质量等级。

20、根据权利要求 19 所述的装置，其特征在于，所述调整模块用于：

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值，保持所述第一线程和所述第三线程的服务质量等级不变；

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值，降低所述第一线程的服务质量等级。

21、根据权利要求 19 所述的装置，其特征在于，所述调整模块用于：

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量小于数据量阈值，提高所述第一线程的服务质量等级；

当所述第一线程调用所述第三线程时，如果所述第三线程中所缓存的所述第一线程的数据量等于或大于所述数据量阈值，提高所述第三线程的服务质量等级。

22、根据权利要求 14 至 21 任一项所述的装置，其特征在于，所述任务的服务质量等级由所述任务的任务特征确定。

23、根据权利要求 14 至 22 任一项所述的装置，其特征在于，所述装置还包括：

第二配置模块，用于当处理所述任务时，如果所述应用程序由前台运行切换至后台运行，将与所述任务对应的线程的服务质量等级配置为第一目标等级。

24、根据权利要求 14-23 任一项所述的装置，其特征在于，所述第一配置模块，用于配置所述第一线程的线程控制块结构体内的服务质量等级字段为所述任务的服务质量等级，所述线程控制块结构体用于存储线程的信息。

25、根据权利要求 14 至 24 任一项所述的装置，其特征在于，所述装置还包括：

分配模块，用于当所述第一线程申请内存资源失败，且所述第一线程的服务质量等级大于第二目标等级时，从预留内存资源中为所述第一线程分配内存资源。

26、根据权利要求 14-25 任一项所述的装置，其特征在于，不同的任务请求对应于不同的任务质量等级。

27、一种终端，其特征在于，所述终端包括一个或多个处理器和一个或多个存储器，所述一个或多个存储器中存储有至少一条指令，所述指令由所述一个或多个处理器执行以实现如权利要求 1 至权利要求 13 任一项所述的任务处理方法。

28、一种计算机可读存储介质，其特征在于，所述存储介质中存储有至少一条指令，所述指令由处理器执行以实现如权利要求 1 至权利要求 13 任一项所述的任务处理方法。

1/12
终端100

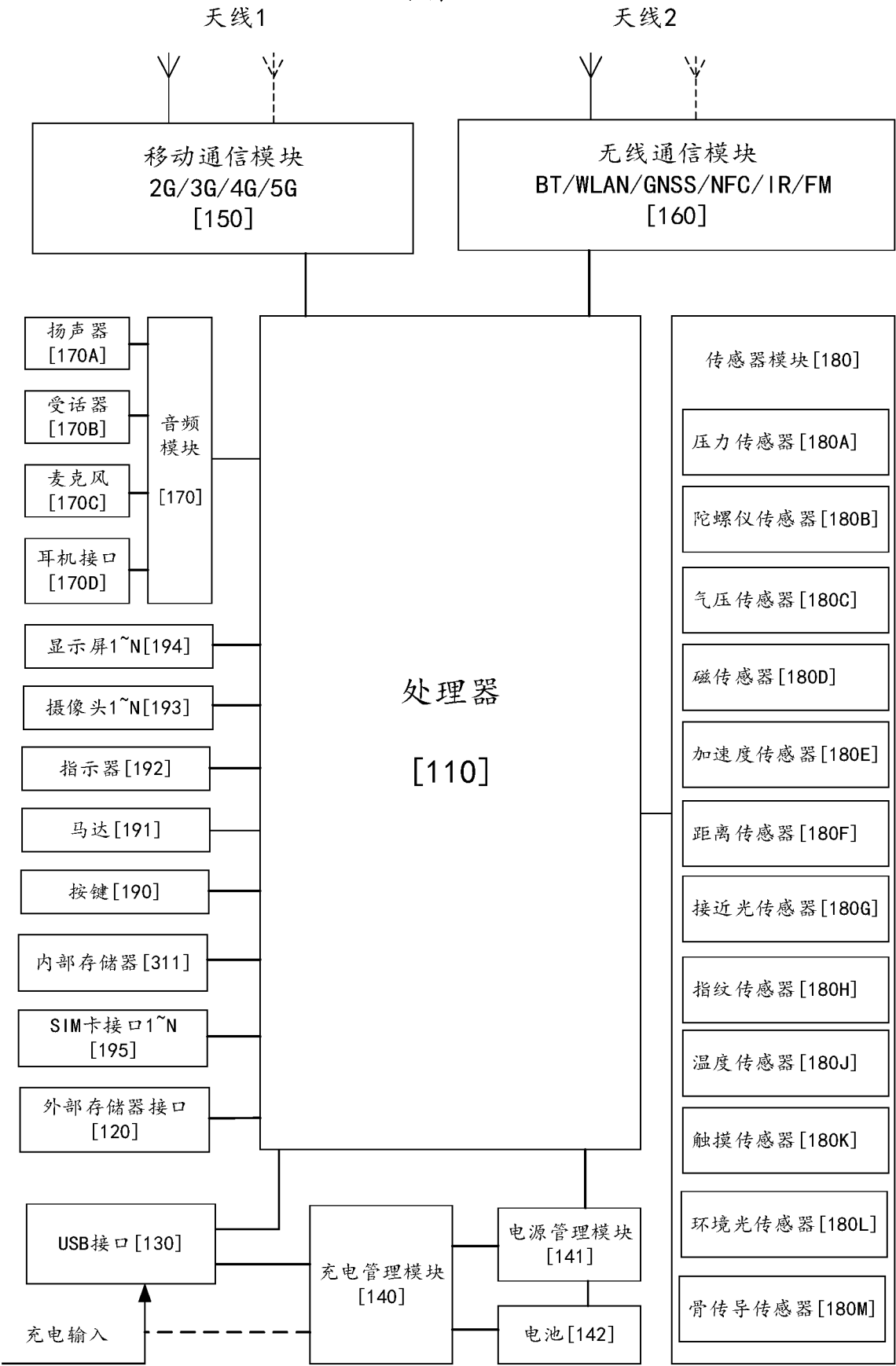


图 1

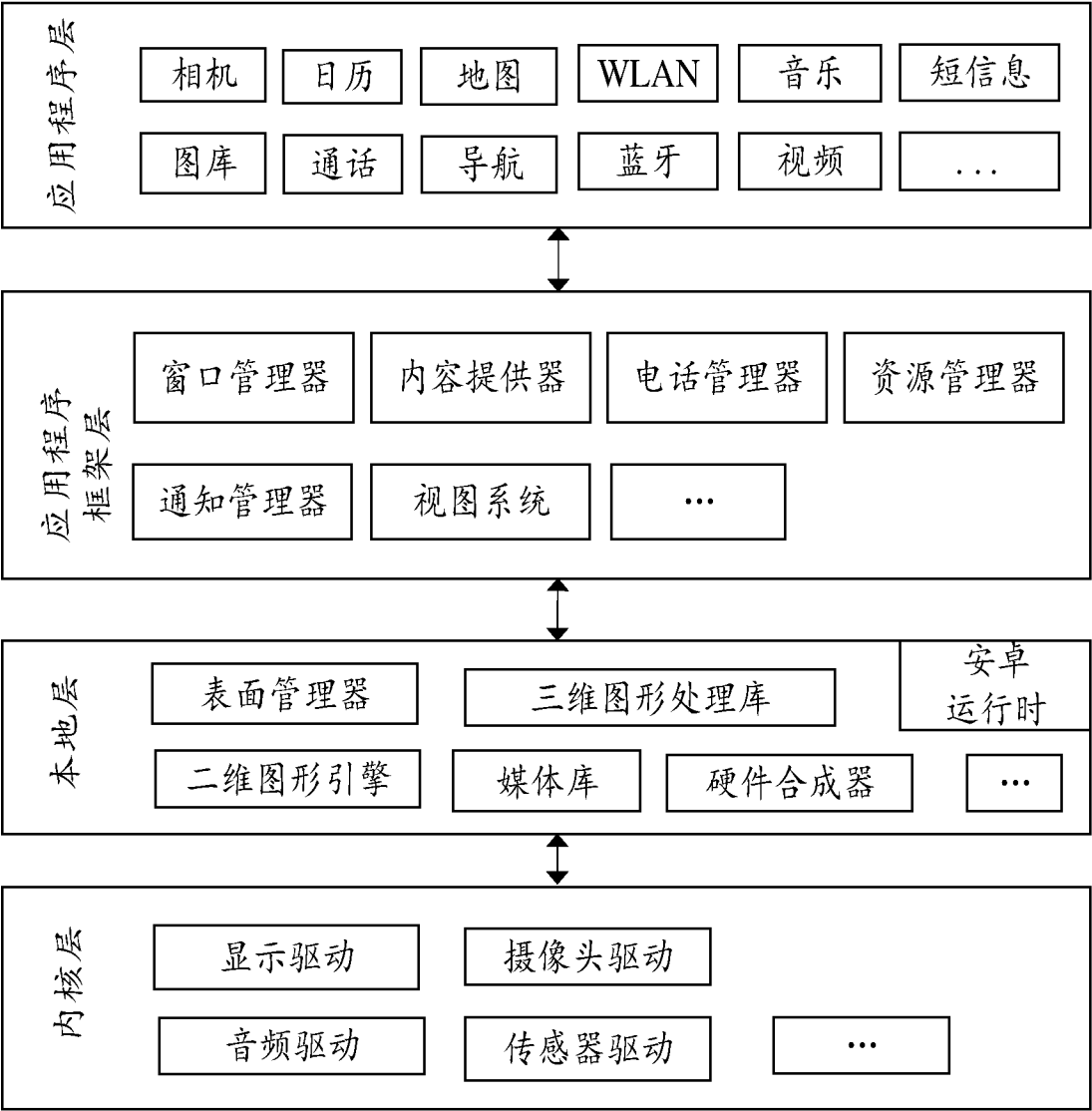
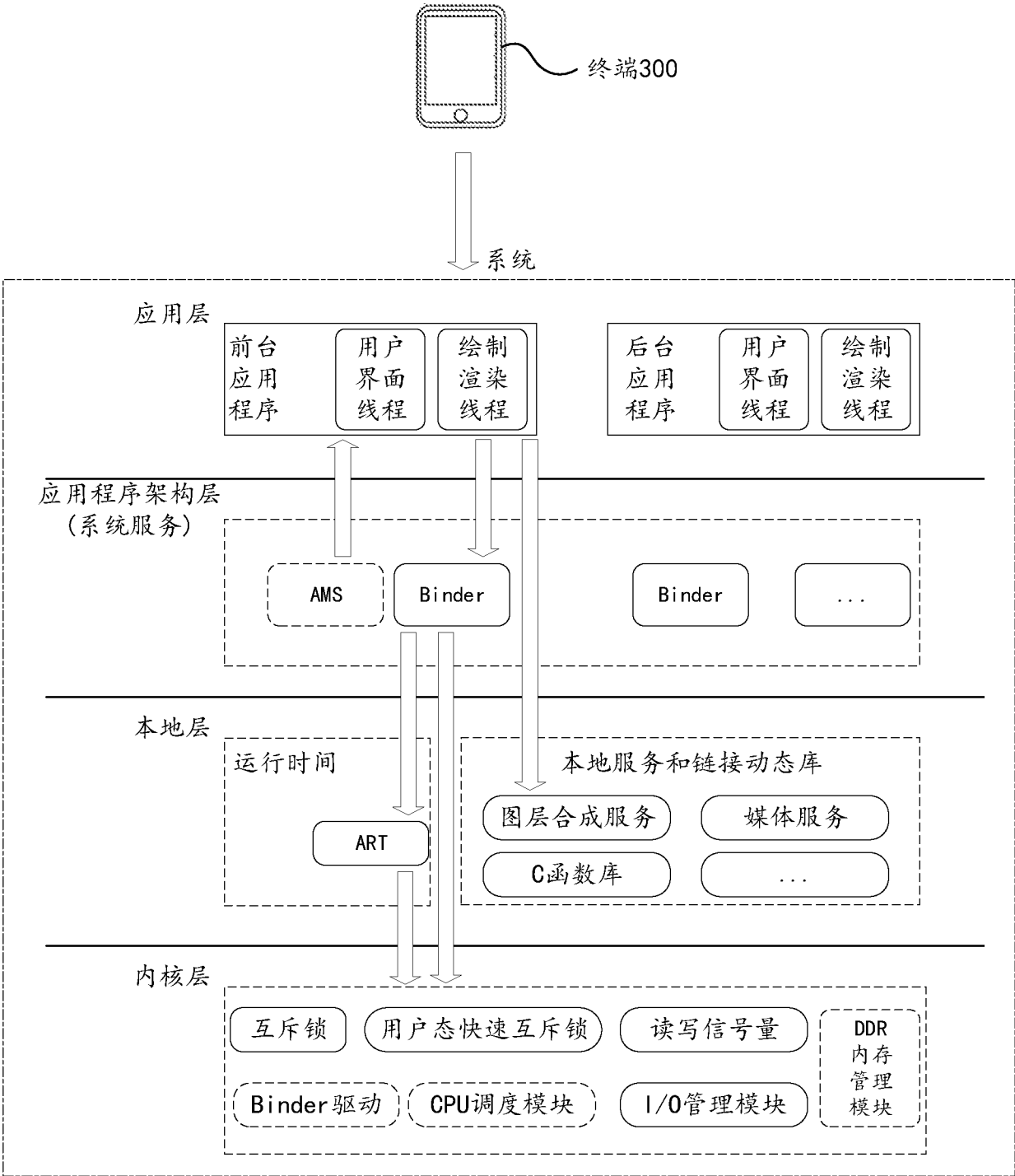
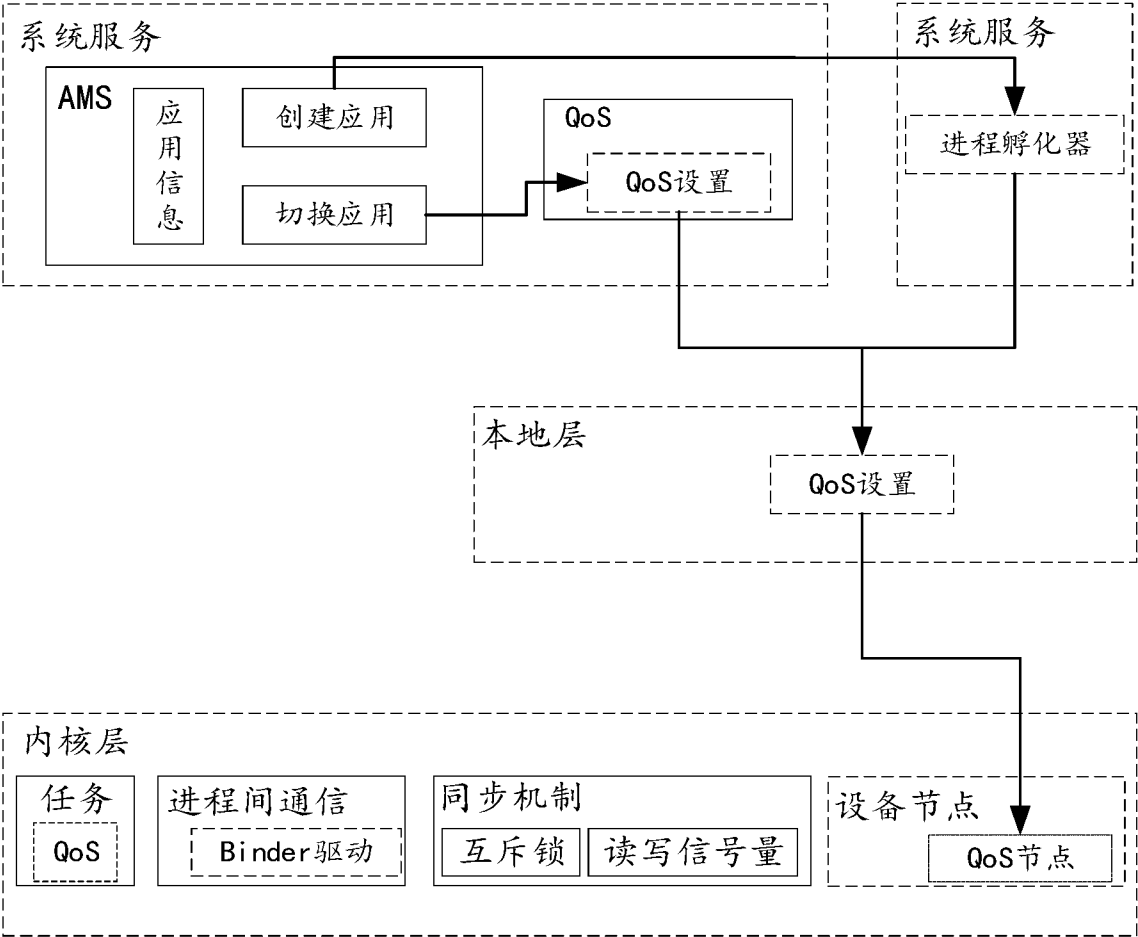


图 2



注： () 为修改后的模块，用于实现新功能

图 3



注：[] 为修改后的模块，用于实现新功能

[] 新增模块

图 4

5/12

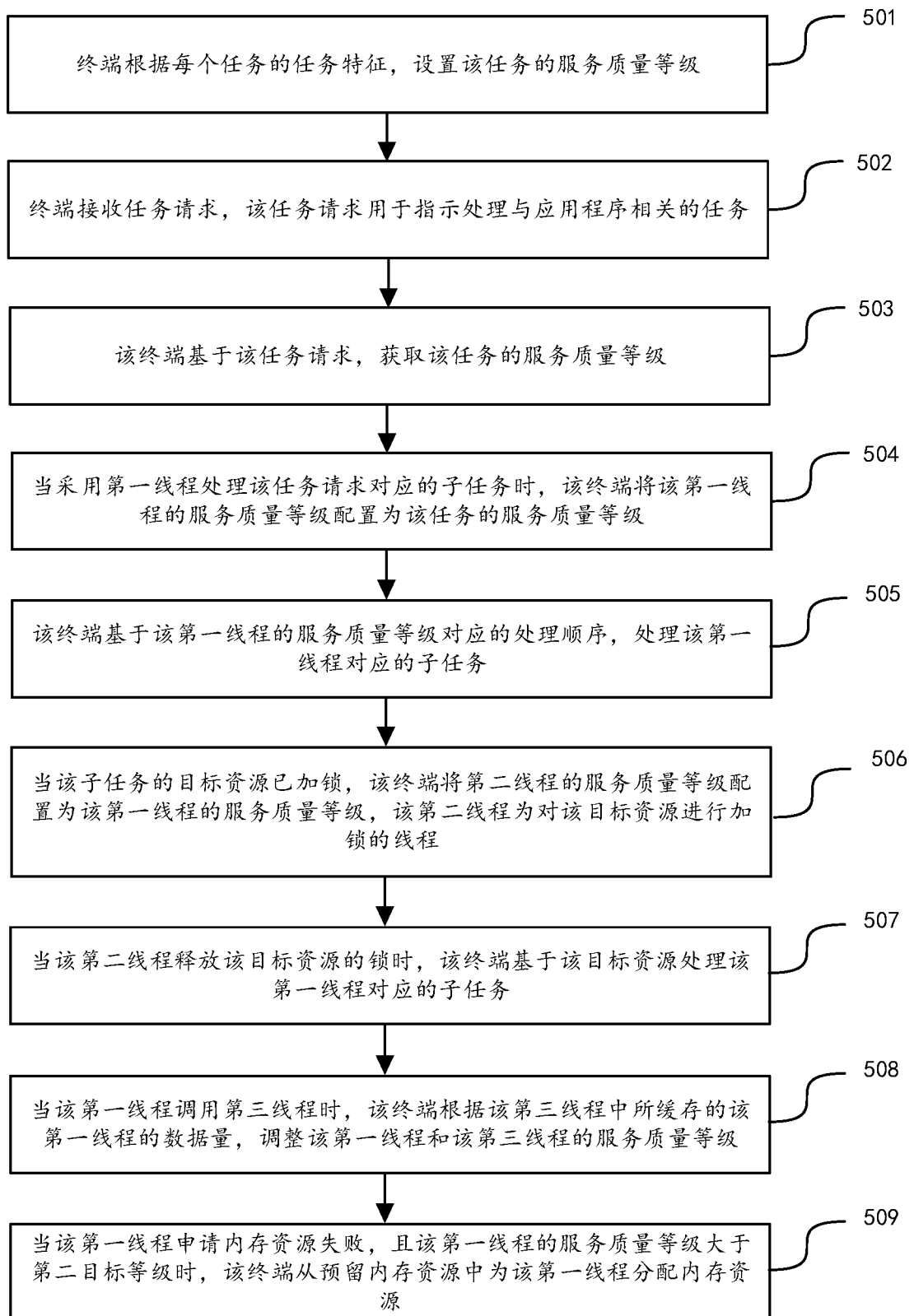


图 5

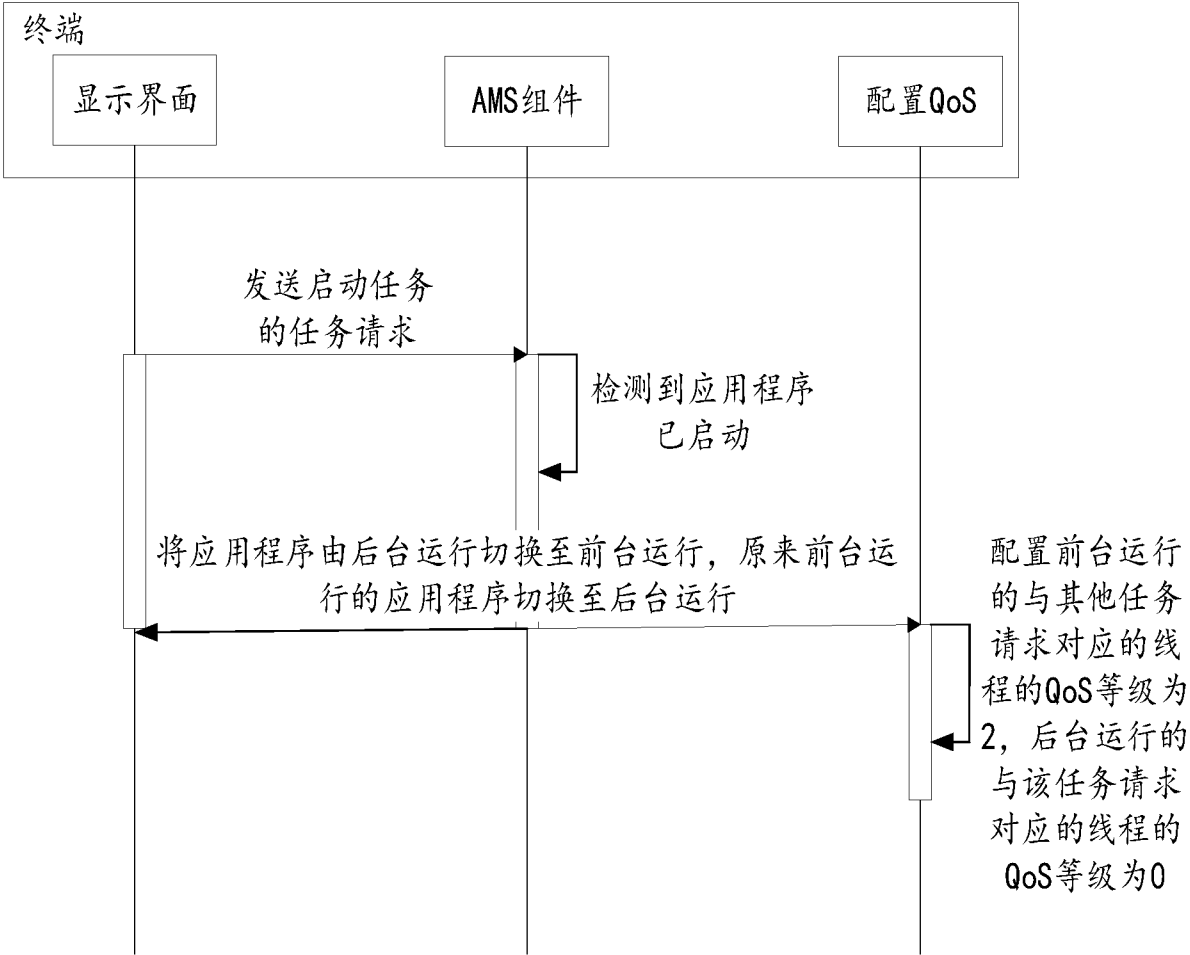


图 6

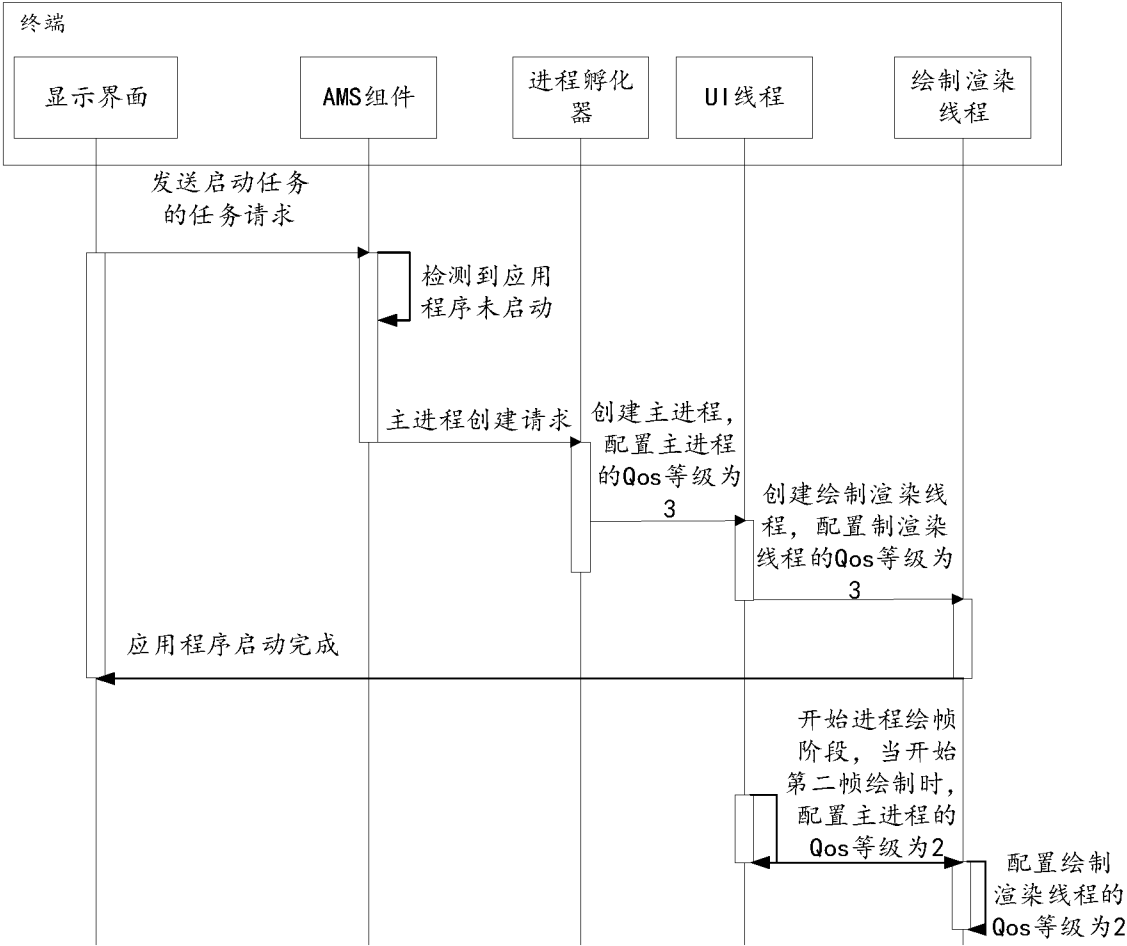


图 7

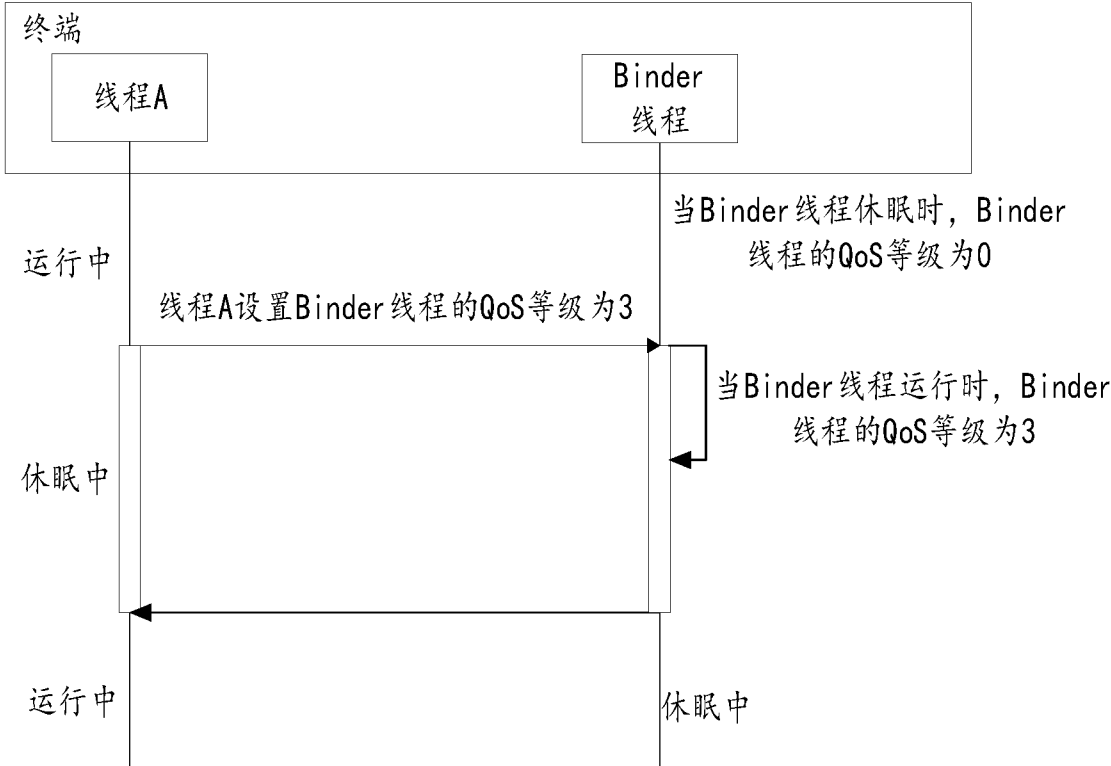


图 8

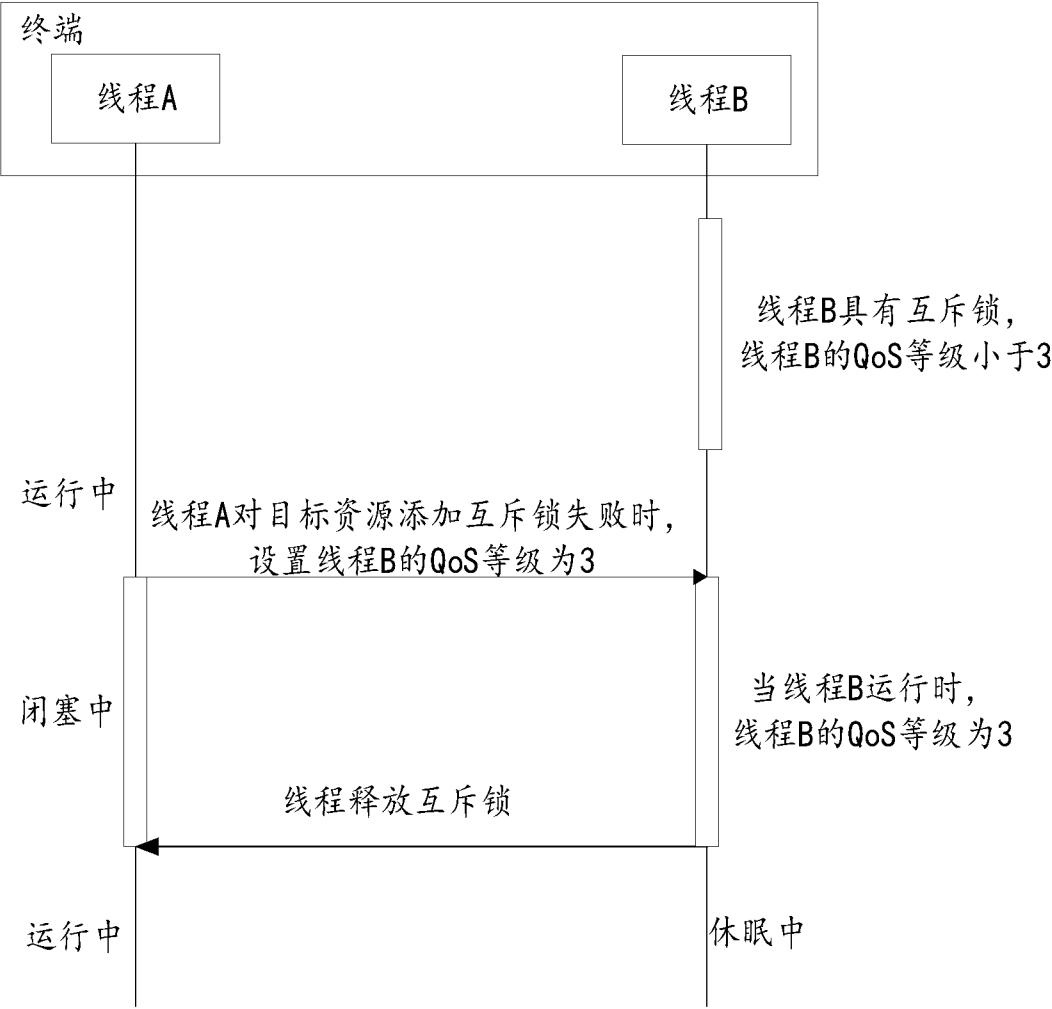


图 9

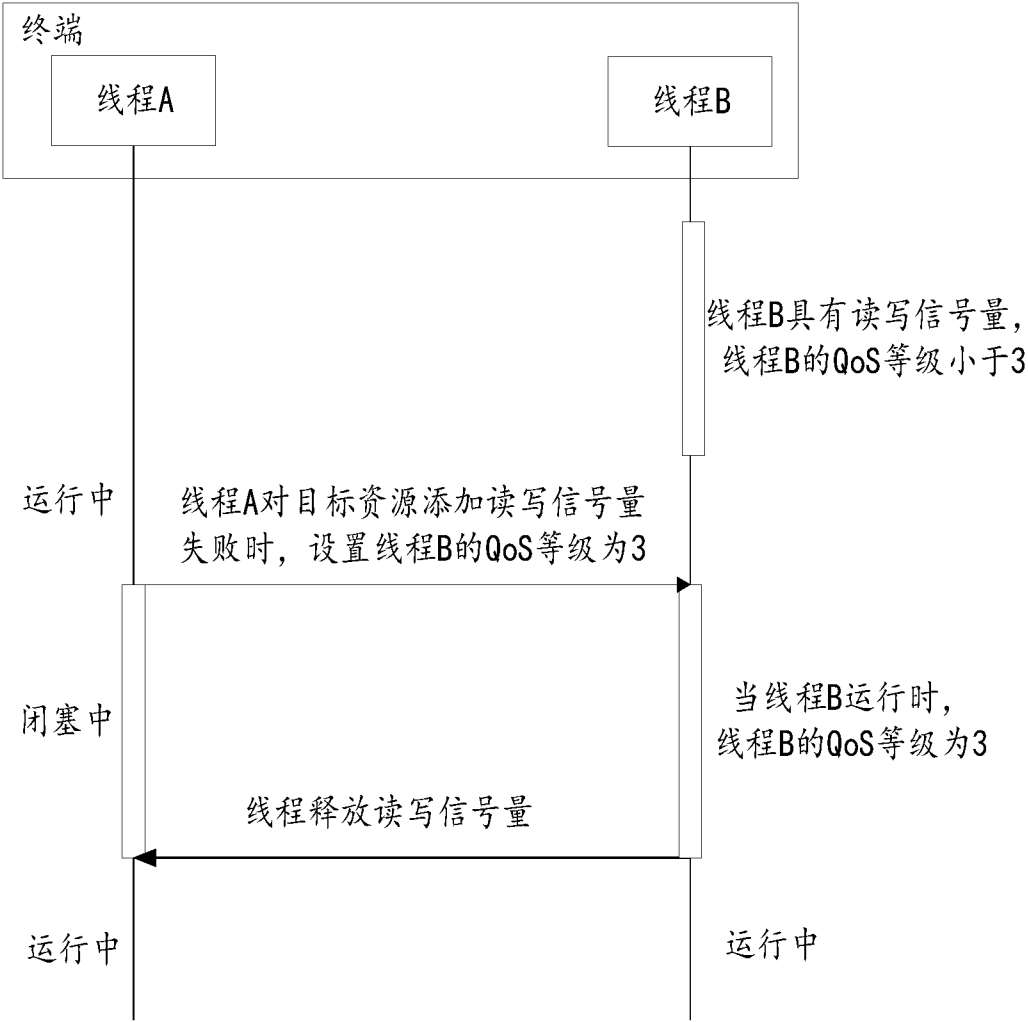
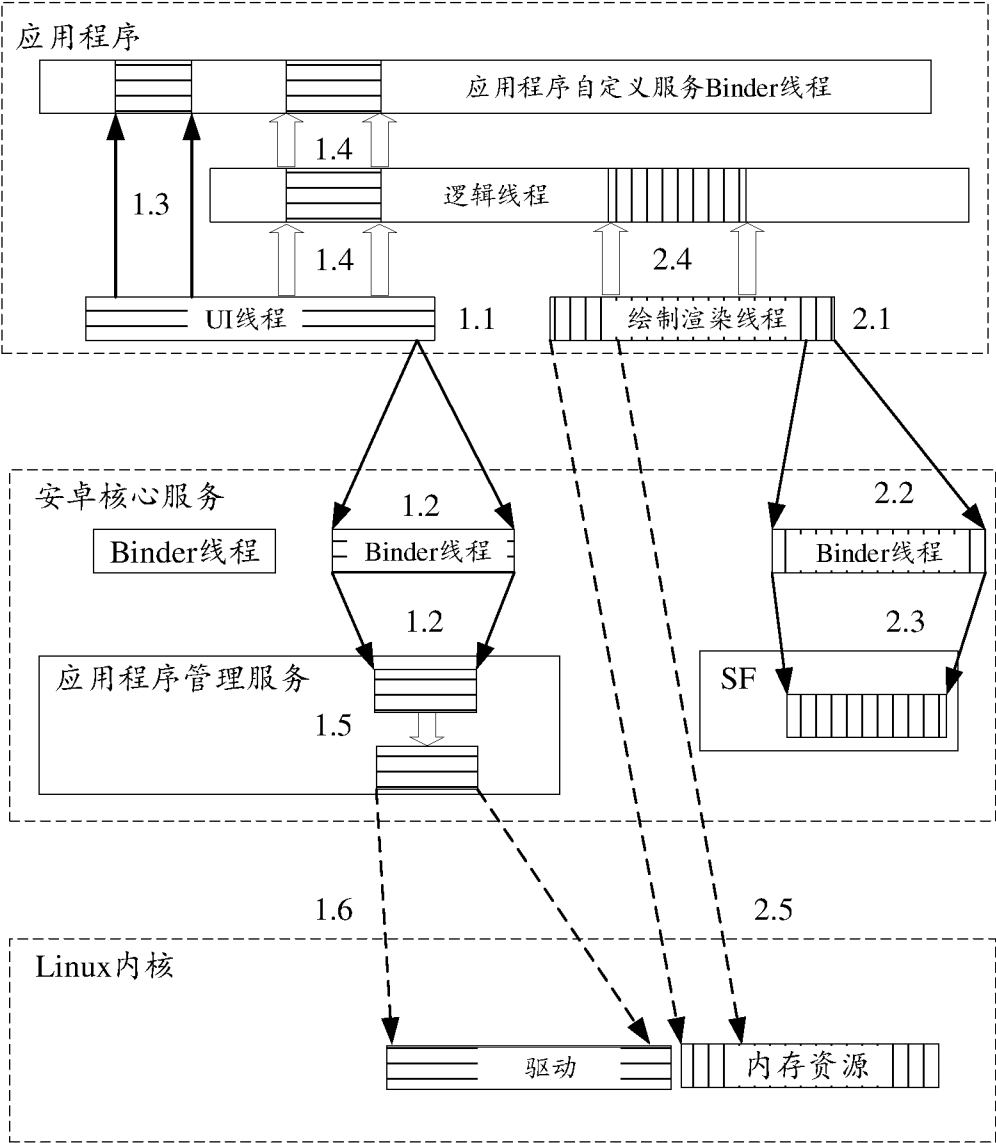
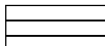


图 10



注：

 原先的QoS等级

 QoS等级为2

 QoS等级为1

 IPC传递的同步调用

 锁传递

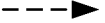
 系统调用

图 11

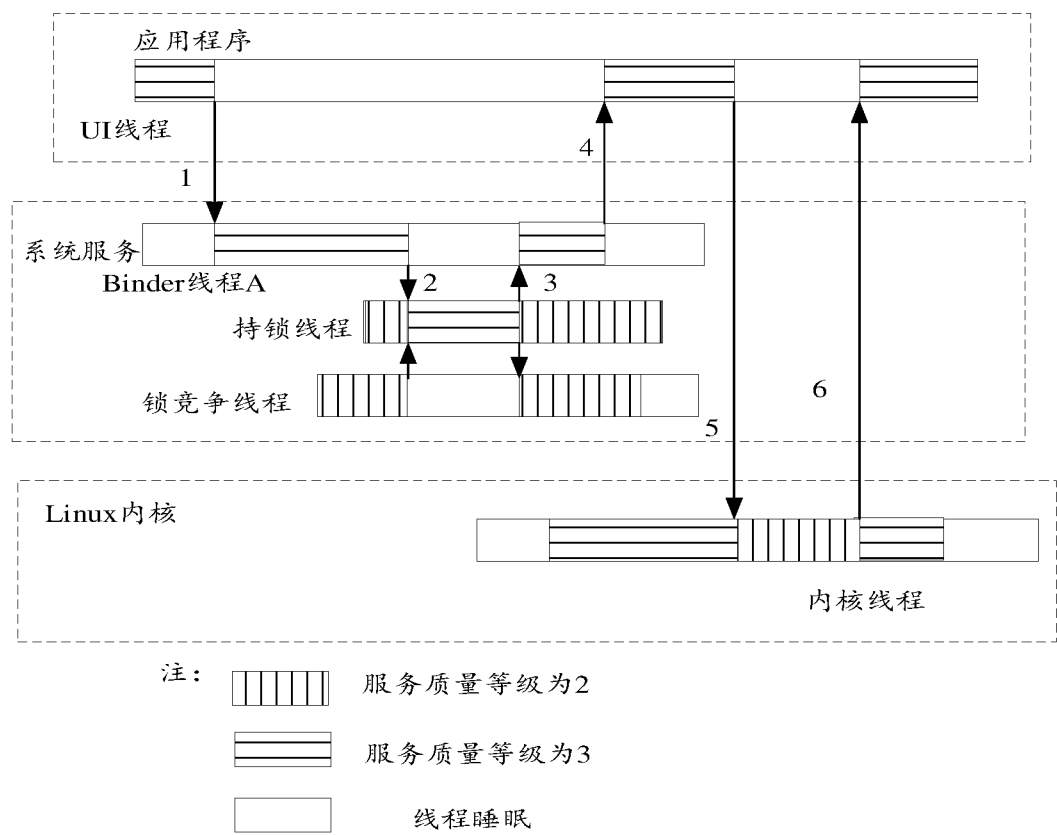


图 12

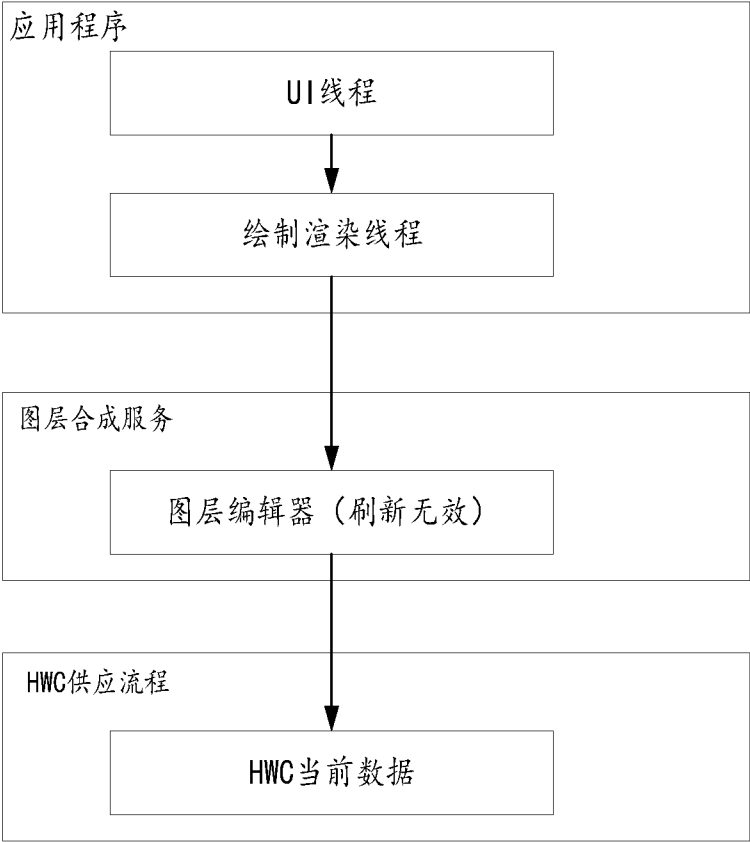


图 13

12/12

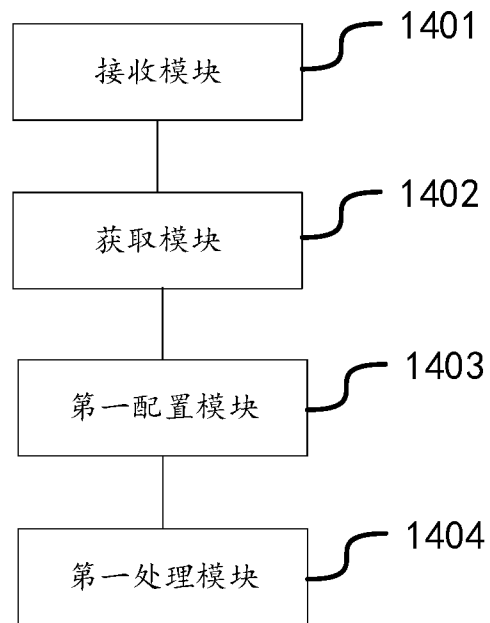


图 14

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2020/083330

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/48(2006.01)i; G06F 9/54(2006.01)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)

G06F9/-

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)

EPTXT; USTXT; VEN; WOTXT; CNABS; CNTXT; CNKI: 任务, 作业, 线程, 进程, 调度, 服务质量, 等级, 级别, 水平, 类别, 优先级, 顺序, task, work, thread, process, schedule, Qos, quality or service, grade, level, class, priority, order

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2015347189 A1 (APPLE INC.) 03 December 2015 (2015-12-03) description paragraphs 7, 13-14, 31-39, figures 1, 4	1-5, 9-18, 22-28
X	CN 109308212 A (SHANGHAI HUAWEI TECHNOLOGIES CO., LTD.) 05 February 2019 (2019-02-05) description paragraphs 79-92, figures 1-2	1, 9-14, 22-28
A	CN 105307277 A (POTEVIO INFORMATION TECHNOLOGY CO., LTD.) 03 February 2016 (2016-02-03) description paragraphs 41-57, figure 1	1-28
A	CN 109144682 A (ALIBABA GROUP HOLDING LIMITED) 04 January 2019 (2019-01-04) entire document	1-28
A	CN 105700939 A (BEIJING JINGDONG SHANGKE INFORMATION TECHNOLOGY CO., LTD.) 22 June 2016 (2016-06-22) entire document	1-28

☐ 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

29 June 2020

Date of mailing of the international search report

08 July 2020

Name and mailing address of the ISA/CN

China National Intellectual Property Administration (ISA/
CN)
No. 6, Xitucheng Road, Jimenqiao Haidian District, Beijing
100088
China

Authorized officer

Facsimile No. (86-10)62019451

Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2020/083330

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
US	2015347189	A1	03 December 2015	US	9582326	B2	28 February 2017
CN	109308212	A	05 February 2019	None			
CN	105307277	A	03 February 2016	CN	105307277	B	18 January 2019
CN	109144682	A	04 January 2019	US	2020133724	A1	30 April 2020
				WO	2019001332	A1	03 January 2019
CN	105700939	A	22 June 2016	CN	105700939	B	02 July 2019

国际检索报告

国际申请号

PCT/CN2020/083330

A. 主题的分类

G06F 9/48(2006.01)i; G06F 9/54(2006.01)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)

G06F9/-

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))

EPTXT;USTXT;VEN;WOTXT;CNABS;CNTXT;CNKI:任务, 作业, 线程, 进程, 调度, 服务质量, 等级, 级别, 水平, 类别, 优先级, 顺序, task, work, thread, process, schedule, Qos, quality or service, grade, level, class, priority, order

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	US 2015347189 A1 (APPLE INC) 2015年 12月 3日 (2015 - 12 - 03) 说明书第7、13-14、31-39段, 图1、4	1-5, 9-18, 22-28
X	CN 109308212 A (上海华为技术有限公司) 2019年 2月 5日 (2019 - 02 - 05) 说明书第79-92段、图1-2	1, 9-14, 22-28
A	CN 105307277 A (普天信息技术有限公司) 2016年 2月 3日 (2016 - 02 - 03) 说明书第41-57段、图1	1-28
A	CN 109144682 A (阿里巴巴集团控股有限公司) 2019年 1月 4日 (2019 - 01 - 04) 全文	1-28
A	CN 105700939 A (北京京东尚科信息技术有限公司) 2016年 6月 22日 (2016 - 06 - 22) 全文	1-28

☐ 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

2020年 6月 29日

国际检索报告邮寄日期

2020年 7月 8日

ISA/CN的名称和邮寄地址

中国国家知识产权局(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

授权官员

梁静静

电话号码 86-(20)-28958273

国际检索报告
关于同族专利的信息

国际申请号

PCT/CN2020/083330

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)
US	2015347189	A1	2015年 12月 3日	US	9582326	B2	2017年 2月 28日
CN	109308212	A	2019年 2月 5日	无			
CN	105307277	A	2016年 2月 3日	CN	105307277	B	2019年 1月 18日
CN	109144682	A	2019年 1月 4日	US	2020133724	A1	2020年 4月 30日
				WO	2019001332	A1	2019年 1月 3日
CN	105700939	A	2016年 6月 22日	CN	105700939	B	2019年 7月 2日