(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 113656189 A
(43)申请公布日 2021.11.16

(21)申请号 202110882353.9

(22)申请日 2018.05.15

(62)分案原申请数据
201810463425.4 2018.05.15

(71)申请人 西安万像电子科技有限公司
地址 710075 陕西省西安市高新区唐延南路8号3G智能终端产业园4号厂房第3层302室

(72)发明人 杨星亮 王官军 苏睿

(74)专利代理机构 北京康信知识产权代理有限责任公司 11240
代理人 李静茹

(51)Int.Cl.
G06F 9/52(2006.01)
G06F 9/54(2006.01)

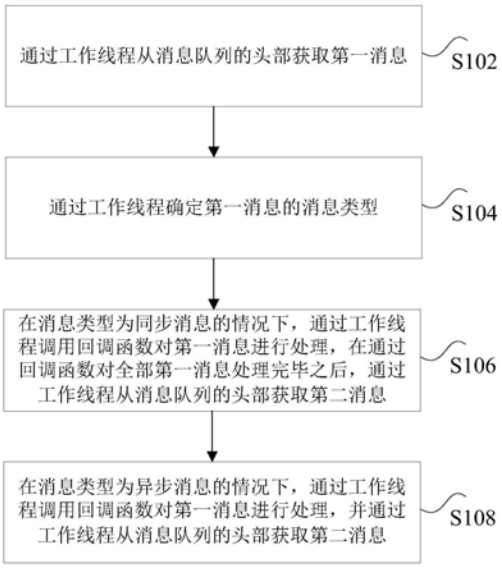
权利要求书2页 说明书11页 附图6页

(54)发明名称

消息处理方法和装置

(57)摘要

本发明公开了一种消息处理方法和装置。其中,该方法包括:通过工作线程从消息队列的头部获取第一消息,消息队列中存储至少一条消息,消息队列由与用户业务对应的回调函数构建;通过工作线程确定第一消息的消息类型,消息类型包括:同步消息或异步消息;在消息类型为同步消息的情况下,通过工作线程调用回调函数对第一消息进行处理,在通过回调函数对全部第一消息处理完毕之后,通过工作线程从消息队列的头部获取第二消息;在消息类型为异步消息的情况下,通过工作线程调用回调函数对第一消息进行处理,并通过工作线程从消息队列的头部获取第二消息。本发明解决了现有技术中基于多线程的消息处理方法存在线程间同步复杂,资源竞争、死锁的技术问题。



CN 113656189 A

1.一种消息处理方法,其特征在于,包括:

通过工作线程从消息队列的头部获取第一消息,其中,所述消息队列中存储至少一条消息,所述消息队列由与用户业务对应的回调函数构建;

通过所述工作线程确定所述第一消息的消息类型,其中,所述消息类型包括:同步消息或异步消息;

在所述消息类型为所述同步消息的情况下,通过所述工作线程调用回调函数对所述第一消息进行处理,在通过所述回调函数对全部所述第一消息处理完毕之后,通过所述工作线程从所述消息队列的头部获取第二消息;

在所述消息类型为所述异步消息的情况下,通过所述工作线程调用所述回调函数对所述第一消息进行处理,并通过所述工作线程从所述消息队列的头部获取所述第二消息。

2.根据权利要求1所述的方法,其特征在于,通过所述工作线程调用回调函数对所述第一消息进行处理,包括:

通过工作线程和所述第一消息的消息数据,调用所述回调函数对所述第一消息进行处理。

3.根据权利要求2所述的方法,其特征在于,在所述消息类型为所述同步消息的情况下,通过所述工作线程调用回调函数对所述第一消息进行处理,包括:

通过所述工作线程唤醒同步调用线程,并为所述同步调用线程分配第一存储空间;

通过所述同步调用线程调用所述回调函数对所述第一消息进行处理。

4.根据权利要求3所述的方法,其特征在于,在通过所述工作线程唤醒同步调用线程,并为所述同步调用线程分配第一存储空间之后,所述方法还包括:

通过所述同步调用线程将所述第一消息的消息数据存储在所述第一存储空间内;

通过所述同步调用线程在所述消息队列的尾部增加一条新消息。

5.根据权利要求3所述的方法,其特征在于,通过所述工作线程从所述消息队列的头部获取第二消息包括:

通过所述同步调用线程唤醒所述工作线程;

通过所述工作线程从所述消息队列的头部获取所述第二消息。

6.根据权利要求2所述的方法,其特征在于,在所述消息类型为所述异步消息的情况下,通过所述工作线程调用回调函数对所述第一消息进行处理,包括:

通过所述工作线程唤醒异步调用线程,并为所述异步调用线程分配第二存储空间;

通过所述异步调用线程调用所述回调函数对所述第一消息进行处理。

7.根据权利要求6所述的方法,其特征在于,在通过所述工作线程唤醒异步调用线程,并为所述异步调用线程分配第二存储空间之后,所述方法还包括:

通过所述异步调用线程将所述第一消息的消息数据存储在所述第二存储空间内;

通过所述异步调用线程在所述消息队列的尾部增加一条新消息。

8.根据权利要求6所述的方法,其特征在于,通过所述工作线程从所述消息队列的头部获取所述第二消息包括:

通过所述异步调用线程唤醒所述工作线程;

通过所述工作线程从所述消息队列的头部获取所述第二消息。

9.根据权利要求8所述的方法,其特征在于,在通过所述异步调用线程唤醒所述工作线

程之后，释放所述第二存储空间。

10.根据权利要求1所述的方法，其特征在于，在通过工作线程从消息队列的头部获取第一消息之前，构造所述消息队列，其中，在通过所述工作线程从所述消息队列的头部获取所述第二消息之前，所述方法还包括：

判断是否检测到预设标志信息，其中，所述预设标志信息用于表征所述工作线程停止从所述消息队列的头部获取所述第二消息；

如果检测到所述预设标志信息，则析构所述消息队列；

如果未检测到所述预设标志信息，则通过所述工作线程从所述消息队列的头部获取所述第二消息。

11.根据权利要求1所述的方法，其特征在于，在通过工作线程从消息队列的头部获取第一消息之前，所述方法还包括：

基于所述回调函数创建所述工作线程，并为所述工作线程分配第三存储空间。

12.一种消息处理装置，其特征在于，包括：

获取模块，用于通过工作线程从消息队列的头部获取第一消息，其中，所述消息队列中存储至少一条消息，所述消息队列由与用户业务对应的回调函数构建；

确定模块，用于通过所述工作线程确定所述第一消息的消息类型，其中，所述消息类型包括：同步消息或异步消息；

第一调用模块，用于在所述消息类型为所述同步消息的情况下，通过所述工作线程调用回调函数对所述第一消息进行处理，在通过所述回调函数对全部所述第一消息处理完毕之后，通过所述工作线程从所述消息队列的头部获取第二消息；

第二调用模块，用于在所述消息类型为所述异步消息的情况下，通过所述工作线程调用所述回调函数对所述第一消息进行处理，并通过所述工作线程从所述消息队列的头部获取第二消息。

13.一种消息处理方法，其特征在于，包括：

通过工作线程从消息队列的头部获取第一消息，其中，所述消息队列中存储至少一条消息，所述消息队列由与用户业务对应的回调函数构建；

通过所述工作线程调用回调函数对所述第一消息进行处理；

通过所述工作线程从所述消息队列的头部获取第二消息。

消息处理方法和装置

技术领域

[0001] 本发明涉及线程编程领域，具体而言，涉及一种消息处理方法和装置。

背景技术

[0002] 在很多软件解决方案中，不可避免的要涉及到多线程编程，多线程一方面为编程提供了便利，但也同时带来了很多问题，例如，windows api并不完全兼容posix标准，这使得线程及相关的api上有一些差异，需要用户自己进行平台相容的封装，目前主要有三种解决方案，第一种是windows上使用pthead实现库；第二种是利用最新的c11标准；第三种是用户自行编译条件区分，导致多线程跨平台性差。又例如，线程间同步问题，尤其当线程模块较多时，资源竞争，死锁等问题往往使程序问题很难维护和跟踪。

[0003] 针对现有技术中基于多线程的消息处理方法存在线程间同步复杂，资源竞争、死锁的问题，目前尚未提出有效的解决方案。

发明内容

[0004] 本发明实施例提供了一种消息处理方法和装置，以至少解决现有技术中基于多线程的消息处理方法存在线程间同步复杂，资源竞争、死锁的技术问题。

[0005] 根据本发明实施例的一个方面，提供了一种消息处理方法，包括：通过工作线程从消息队列的头部获取第一消息，其中，消息队列中存储至少一条消息，消息队列由与用户业务对应的回调函数构建；通过工作线程确定第一消息的消息类型，其中，消息类型包括：同步消息或异步消息；在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息；在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消息。

[0006] 进一步地，通过工作线程调用回调函数对第一消息进行处理，包括：通过工作线程和第一消息的消息数据，调用回调函数对第一消息进行处理。

[0007] 进一步地，在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，包括：通过工作线程唤醒同步调用线程，并为同步调用线程分配第一存储空间；通过同步调用线程调用回调函数对第一消息进行处理。

[0008] 进一步地，在通过工作线程唤醒同步调用线程，并为同步调用线程分配第一存储空间之后，上述方法还包括：通过同步调用线程将第一消息的消息数据存储在第一存储空间内；通过同步调用线程在消息队列的尾部增加一条新消息。

[0009] 进一步地，通过工作线程从消息队列的头部获取第二消息包括：通过同步调用线程唤醒工作线程；通过工作线程从消息队列的头部获取第二消息。

[0010] 进一步地，在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，包括：通过工作线程唤醒异步调用线程，并为异步调用线程分配第二存储空间；通过异步调用线程调用回调函数对第一消息进行处理。

[0011] 进一步地，在通过工作线程唤醒异步调用线程，并为异步调用线程分配第二存储空间之后，上述方法还包括：通过异步调用线程将第一消息的消息数据存储在第二存储空间内；通过异步调用线程在消息队列的尾部增加一条新消息。

[0012] 进一步地，通过工作线程从消息队列的头部获取第二消息包括：通过异步调用线程唤醒工作线程；通过工作线程从消息队列的头部获取第二消息。

[0013] 进一步地，在通过异步调用线程唤醒工作线程之后，释放第二存储空间。

[0014] 进一步地，在通过工作线程从消息队列的头部获取第一消息之前，构造消息队列，其中，在通过工作线程从消息队列的头部获取第二消息之前，上述方法还包括：判断是否检测到预设标志信息，其中，预设标志信息用于表征工作线程停止从消息队列的头部获取第二消息；如果检测到预设标志信息，则析构消息队列；如果未检测到预设标志信息，则通过工作线程从消息队列的头部获取第二消息。

[0015] 进一步地，在通过工作线程从消息队列的头部获取第一消息之前，上述方法还包括：基于回调函数创建工作线程，并为工作线程分配第三存储空间。

[0016] 进一步地，在通过工作线程从消息队列的头部获取第一消息之后，通过工作线程为第一消息分配第四存储空间，并将第一消息的消息数据存储在第四存储空间内。

[0017] 进一步地，通过工作线程从消息队列的头部获取第一消息或第二消息包括：判断消息队列中是否存在消息；在确定消息队列中存在消息的情况下，通过工作线程从消息队列的头部获取第一消息或第二消息。

[0018] 进一步地，通过工作线程从消息队列的头部获取第一消息或第二消息包括：利用互斥机制，通过工作线程从消息队列的头部获取第一消息或第二消息；通过同步调用线程或异步调用线程在消息队列的尾部增加一条新消息包括：利用互斥机制，通过同步调用线程或异步调用线程在消息队列的尾部增加一条新消息。

[0019] 根据本发明实施例的另一方面，还提供了一种消息处理装置，包括：获取模块，用于通过工作线程从消息队列的头部获取第一消息，其中，消息队列中存储至少一条消息；确定模块，用于通过工作线程确定第一消息的消息类型，其中，消息类型包括：同步消息或异步消息；第一调用模块，用于在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息；第二调用模块，用于在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消息。

[0020] 根据本发明实施例的另一方面，还提供了一种存储介质，存储介质包括存储的程序，其中，在程序运行时控制存储介质所在设备执行上述的消息处理方法。

[0021] 根据本发明实施例的另一方面，还提供了一种处理器，处理器用于运行程序，其中，程序运行时执行上述的消息处理方法。

[0022] 根据本发明实施例的另一方面，还提供了一种消息处理方法，包括：通过工作线程从消息队列的头部获取第一消息，其中，消息队列中存储至少一条消息；通过工作线程调用回调函数对第一消息进行处理；通过工作线程从消息队列的头部获取第二消息。

[0023] 在本发明实施例中，在通过工作线程从消息队列的头部获取到第一消息之后，可以首先判断第一消息的消息类型，在消息类型为同步消息的情况下，通过工作线程调用回

调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息，在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消，从而实现多线程消息处理。与现有技术相比，仅通过工作线程调用回调函数对消息队列中存储的第一消息进行处理，而且可以对不同消息类型的消息执行不同的操作，从而达到了对用户隐藏多线程及其同步的细节，提高消息处理的便捷性，提高消息处理的兼容性，提升用户体验感和好感度的而技术效果，进而解决了现有技术中基于多线程的消息处理方法存在线程间同步复杂，资源竞争、死锁的技术问题。

附图说明

[0024] 此处所说明的附图用来提供对本发明的进一步理解，构成本申请的一部分，本发明的示意性实施例及其说明用于解释本发明，并不构成对本发明的不当限定。在附图中：

[0025] 图1是根据本发明实施例的一种消息处理方法的流程图；

[0026] 图2是根据本发明实施例的一种可选的工作线程的构造流程图；

[0027] 图3是根据本发明实施例的一种可选的消息队列的实现逻辑图；

[0028] 图4是根据本发明实施例的一种可选的工作线程运行流程图；

[0029] 图5是根据本发明实施例的一种可选的消息调用流程图；

[0030] 图6是根据本发明实施例的一种消息处理装置的示意图；以及

[0031] 图7是根据本发明实施例的另一种消息处理方法的流程图。

具体实施方式

[0032] 为了使本技术领域的人员更好地理解本发明方案，下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅仅是本发明一部分的实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都应当属于本发明保护的范围。

[0033] 需要说明的是，本发明的说明书和权利要求书及上述附图中的术语“第一”、“第二”等是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。应该理解这样使用的数据在适当情况下可以互换，以便这里描述的本发明的实施例能够以除了在这里图示或描述的那些以外的顺序实施。此外，术语“包括”和“具有”以及他们的任何变形，意图在于覆盖不排他的包含，例如，包含了一系列步骤或单元的过程、方法、系统、产品或设备不必限于清楚地列出的那些步骤或单元，而是可包括没有清楚地列出的或对于这些过程、方法、产品或设备固有的其它步骤或单元。

[0034] 实施例1

[0035] 根据本发明实施例，提供了一种消息处理方法的实施例，需要说明的是，在附图的流程图示出的步骤可以在诸如一组计算机可执行指令的计算机系统中执行，并且，虽然在流程图中示出了逻辑顺序，但是在某些情况下，可以以不同于此处的顺序执行所示出或描述的步骤。

[0036] 图1是根据本发明实施例的一种消息处理方法的流程图，如图1所示，该方法包括

如下步骤：

[0037] 步骤S102，通过工作线程从消息队列的头部获取第一消息，其中，消息队列中存储至少一条消息，消息队列由与用户业务对应的回调函数构建。

[0038] 具体地，上述的工作线程可以是预先建立的用于对至少一条消息进行处理的线程，通过工作线程对至少一条消息进行处理，对于用户来说，可以实现多线程同步处理的效果，并且隐藏了多线程及其同步的细节，使得使用极为便利；上述的消息队列可以是根据需要处理需要预先建立的队列，至少一个待处理处理的消息存储在消息队列中，工作线程通过按顺序对消息队列中存储的消息进行处理，实现用户级的多线程同步处理。

[0039] 步骤S104，通过工作线程确定第一消息的消息类型，其中，消息类型包括：同步消息或异步消息。

[0040] 具体地，消息队列中存储的消息可以分为同步消息和异步消息，对于同步消息，工作线程需要在同步消息执行完毕，得到执行结果之后才能执行其他消息，而对于异步消息，工作线程无需等待执行结果即可执行其他消息。

[0041] 步骤S106，在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息。

[0042] 具体地，上述的回调函数可以是用户业务相关的消息处理函数，通过工作线程进行调用，可以处理消息队列中存储的消息。

[0043] 步骤S108，在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消息。

[0044] 需要说明的是，上述的工作线程、消息队列和回调函数可以是利用成熟的开源项目libuv实现跨平台的底层支持，将业务进行合理包装，并通过纯C语言编写的多线程消息通信框架，具体地，工作线程定义如下；typedef voID*Handler;Handler newHandler(voID*obj,HandlerCallBack cb)；回调函数定义如下：typedef int64_t(*HandlerCallBack)(voID*obj,int msg,int64_t wparam,int64_t lparam)。

[0045] 在一种可选的方案中，用户可以利用自己业务相关的回调函数，构建消息队列，消息队列中存储多个待处理的消息，由于通过工作线程同时只能对一个消息进行处理，可以通过工作线程获取消息队列头部的一个消息，并判断该消息时同步消息还是异步消息，如果确定该消息是同步消息，则可以调用回调函数对该消息进行处理，在确定回调函数对该消息处理完毕之后，可以继续对其他消息进行处理；如果确定该消息是异步消息，则可以调用回调函数对该消息进行处理，无需等待该消息的执行结果，直接对其他消息进行处理。对其他消息进行处理可以通过工作线程再次从消息队列的头部获取一个消息实现。

[0046] 在本发明上述实施例中，在通过工作线程从消息队列的头部获取到第一消息之后，可以首先判断第一消息的消息类型，在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息，在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消，从而实现多线程消息处理。与现有技术相比，仅通过工作线程调用回调函数对消息队列中存储的第一消息进行处理，而且可以对不同消息类型的消息执行不同的操作，从而达到了对用

户隐藏多线程及其同步的细节,提高消息处理的便捷性,提高消息处理的兼容性,提升用户体验感和好感度的而技术效果,进而解决了现有技术中基于多线程的消息处理方法存在线程间同步复杂,资源竞争、死锁的技术问题。

[0047] 可选地,通过工作线程调用回调函数对第一消息进行处理,包括:通过工作线程和第一消息的消息数据,调用回调函数对第一消息进行处理。

[0048] 具体地,上述的消息数据可以是消息标识(例如,消息ID)和参数(例如,抽象参数)。

[0049] 在一种可选的方案中,用户提供消息标识、参数,通过工作线程可以调用回调函数对消息标识、参数对应的第一消息进行处理,得到处理结果。

[0050] 可选地,在消息类型为同步消息的情况下,通过工作线程调用回调函数对第一消息进行处理,包括:通过工作线程唤醒同步调用线程,并为同步调用线程分配第一存储空间;通过同步调用线程调用回调函数对第一消息进行处理。

[0051] 具体地,上述的第一存储空间可以是栈内存,可以用于保存同步消息的消息数据。

[0052] 在一种可选的方案中,对于同步消息,可以通过同步消息api,也即同步调用线程传递队列对象,具体地,使用栈内存,并创建同步对象,唤醒同步调用线程,并通过该同步调用线程对第一消息进行处理。

[0053] 需要说明的是,同步调用线程定义如下:int64_t sendSyncMsg(Handler h,int ID,int64_t wparam,int64_t lparam)。

[0054] 可选地,在通过工作线程唤醒同步调用线程,并为同步调用线程分配第一存储空间之后,上述方法还包括:通过同步调用线程将第一消息的消息数据存储在第一存储空间内;通过同步调用线程在消息队列的尾部增加一条新消息。

[0055] 在一种可选的方案中,对于同步消息,可以将消息数据存储在栈内存中,并在消息队列末尾添加一条消息。

[0056] 可选地,通过工作线程从消息队列的头部获取第二消息包括:通过同步调用线程唤醒工作线程;通过工作线程从消息队列的头部获取第二消息。

[0057] 在一种可选的方案中,在同步消息处理完毕之后,可以激活条件变量,并唤醒工作线程,通过工作线程继续从消息队列的头部获取消息,得到第二消息,继续对第二消息进行处理。

[0058] 可选地,在消息类型为异步消息的情况下,通过工作线程调用回调函数对第一消息进行处理,包括:通过工作线程唤醒异步调用线程,并为异步调用线程分配第二存储空间;通过异步调用线程调用回调函数对第一消息进行处理。

[0059] 具体地,上述的第二存储空间可以是消息体内存,用于存储异步消息的消息数据。

[0060] 在一种可选的方案中,对于异步消息,可以通过异步消息api,也即通过异步调用函数传递队列对象,具体地,使用消息体内存,唤醒异步调用线程,并通过该异步调用线程对第一消息进行处理。

[0061] 需要说明的是,异步调用线程定义如下:voID sendAsyncMsg(Handler h,int ID,int64_t wparam,int64_t lparam)。

[0062] 可选地,在通过工作线程唤醒异步调用线程,并为异步调用线程分配第二存储空间之后,上述方法还包括:通过异步调用线程将第一消息的消息数据存储在第二存储空间

内;通过异步调用线程在消息队列的尾部增加一条新消息。

[0063] 在一种可选的方案中,对于异步消息,可以将消息数据存储在消息体内存中,并在消息队列末尾添加一条消息。

[0064] 可选地,通过工作线程从消息队列的头部获取第二消息包括:通过异步调用线程唤醒工作线程;通过工作线程从消息队列的头部获取第二消息。

[0065] 在一种可选的方案中,在通过异步调用线程调用回调函数的同时,无需等待异步消息的处理结果,可以直接继续处理其他消息,也即可以激活条件变量,并唤醒工作线程,通过工作线程继续从消息队列的头部获取消息,得到第二消息,继续对第二消息进行处理。

[0066] 可选地,在通过异步调用线程唤醒工作线程之后,释放第二存储空间。

[0067] 在一种可选的方案中,为了避免存储空间浪费,在回调函数处理异步消息的同时,可以将存储异步消息的消息数据的消息体内存进行存储

[0068] 可选地,在通过工作线程从消息队列的头部获取第一消息之前,构造消息队列,其中,在通过工作线程从消息队列的头部获取第二消息之前,上述方法还包括:判断是否检测到预设标志信息,其中,预设标志信息用于表征工作线程停止从消息队列的头部获取第二消息;如果检测到预设标志信息,则析构消息队列;如果未检测到预设标志信息,则通过工作线程从消息队列的头部获取第二消息。

[0069] 具体地,上述的预设标志信息可以是退出标志,用于表征消息队列使用完毕。

[0070] 在一种可选的方案中,在从消息队列中获取消息并进行处理之前,首先判断消息对了是否使用完毕,如果未使用完毕,则可以从消息队列中获取消息;如果已经使用完毕,则可以直接析构消息队列,结束消息处理过程。

[0071] 需要说明的是,析构消息队列定义如下:voID freeHandler(Handler handler)。

[0072] 可选地,在通过工作线程从消息队列的头部获取第一消息之前,上述方法还包括:基于回调函数创建工作线程,并为工作线程分配第三存储空间。

[0073] 具体地,上述的第三存储空间可以是工作线程的内存,用于存储工作线程相关的消息对象和回调函数。

[0074] 在一种可选的方案中,如图2所示,工作线程的创建流程为:输入消息对象和回调函数等参数,申请对象内存,保存消息对象及回调函数,初始化互斥及条件变量对象,并创建工作线程。

[0075] 可选地,在通过工作线程从消息队列的头部获取第一消息之后,通过工作线程为第一消息分配第四存储空间,并将第一消息的消息数据存储在第四存储空间内。

[0076] 具体地,上述的第四存储空间可以是消息内存,用于存储消息数据。

[0077] 在一种可选的方案中,可以根据消息ID和参数,申请消息内存并保存消息数据。

[0078] 可选地,通过工作线程从消息队列的头部获取第一消息或第二消息包括:判断消息队列中是否存在消息;在确定消息队列中存在消息的情况下,通过工作线程从消息队列的头部获取第一消息或第二消息。

[0079] 在一种可选的方案中,为了避免消息队列中未存储消息,工作线程正常工作,可以首先判断消息队列中是否存在消息,也即判断消息队列是否为空,如果消息队列不为空,可以确定消息队列中存在消息,从消息队列头部获取消息得到第一消息或第二消息;如果消息队列为空,则可以等待消息队列中存入消息,然后在获取消息。

[0080] 可选地，通过工作线程从消息队列的头部获取第一消息或第二消息包括：利用互斥机制，通过工作线程从消息队列的头部获取第一消息或第二消息；通过同步调用线程或异步调用线程在消息队列的尾部增加一条新消息包括：利用互斥机制，通过同步调用线程或异步调用线程在消息队列的尾部增加一条新消息。

[0081] 在一种可选的方案中，当需要从消息队列中获取消息时，可以利用互斥机制，进入互斥区，并从消息队列的头部读取一条消息，然后离开互斥区，完成从消息队列中获取消息的过程；当需要在消息队列的尾部增加一条消息时，可以进入互斥区，在消息队列末尾添加一条消息，然后离开互斥区，完成将消息增加至消息队列中的过程。

[0082] 图3是根据本发明实施例的一种可选的消息队列的实现逻辑图，图4是根据本发明实施例的一种可选的工作线程运行流程图，图5是根据本发明实施例的一种可选的消息调用流程图。下面结合图3至图5对本发明一种优选的实施例进行详细说明。

[0083] 如图3所示，用户可以提供消息处理回调函数，并利用回调函数构造消息队列，以及同步或异步消息调用消息队列，当消息队列使用完毕之后，析构消息队列。

[0084] 如图4所示，工作线程运行之后，可以首先检测退出标志，如果检测到退出标志，则运行结束；如果未检测到退出标志，则可以进入互斥区，并判断消息队列是否为空，如果确定消息队列为空，则直接离开互斥区；如果消息队列不为空，则从消息队列头取一条有效消息，并离开互斥区。进一步判断是否成功取到消息，如果未成功取到，也即消息队列为空，则开始执行下一次循环；如果成功取到，也即消息队列不为空，则利用消息对象和消息数据调用回调函数，并判断取到的消息是否为同步消息，如果是同步消息，则激活同步对象，唤醒同步调用线程，并开始执行下一次循环；如果是异步消息，则释放消息体内存，并开始执行下一次循环。下一次循环执行过程与上述流程相同，在此不作赘述。

[0085] 如图5所示，工作线程运行过程中的消息调用流程具体如下：工作流程在成功获取到消息并调用回调函数之后，可以输入消息id和参数，申请消息体内存并保存消息数据，判断取到的消息是否为同步消息，如果是同步消息，则使用栈内存，并创建同步对象，进一步保存消息数据；如果是异步消息，则申请消息体内存，并保存消息数据。在消息数据保存完毕之后，进入互斥区，在消息队列末尾添加一条消息，离开互斥区，并激活条件变量，唤醒工作线程。进一步判断是否为同步消息，如果是同步消息，则无需进行处理，如果不是同步消息则离开互斥区，并激活条件变量，唤醒工作线程。

[0086] 通过上述方案，利用成熟的开源项目libuv实现跨平台的底层支持，将业务进行了合理包装，提供给使用一个极为便利的纯c的多线程消息通信框架，而其api也极为简单，创建、析构，同步消息、异步消息共四个api，对用户隐藏了多线程及其同步的细节，使得使用极为便利。

[0087] 实施例2

[0088] 根据本发明实施例，提供了一种消息处理装置的实施例。

[0089] 图6是根据本发明实施例的一种消息处理装置的示意图，如图6所示，该装置包括：

[0090] 获取模块62，用于通过工作线程从消息队列的头部获取第一消息，其中，消息队列中存储至少一条消息，消息队列由与用户业务对应的回调函数构建。

[0091] 具体地，上述的工作线程可以是预先建立的用于对至少一条消息进行处理的线程，通过工作线程对至少一条消息进行处理，对于用户来说，可以实现多线程同步处理的效

果，并且隐藏了多线程及其同步的细节，使得使用极为便利；上述的消息队列可以是根据需要处理需要预先建立的队列，至少一个待处理处理的消息存储在消息队列中，工作线程通过按顺序对消息队列中存储的消息进行处理，实现用户级的多线程同步处理。

[0092] 确定模块64，用于通过工作线程确定第一消息的消息类型，其中，消息类型包括：同步消息或异步消息。

[0093] 具体地，消息队列中存储的消息可以分为同步消息和异步消息，对于同步消息，工作线程需要在同步消息执行完毕，得到执行结果之后才能执行其他消息，而对于异步消息，工作线程无需等待执行结果即可执行其他消息。

[0094] 第一调用模块66，用于在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息。

[0095] 具体地，上述的回调函数可以是用户业务相关的消息处理函数，通过工作线程进行调用，可以处理消息队列中存储的消息。

[0096] 第二调用模块68，用于在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消息。

[0097] 需要说明的是，上述的工作线程、消息队列和回调函数可以是利用成熟的开源项目libuv实现跨平台的底层支持，将业务进行合理包装，并通过纯C语言编写的多线程消息通信框架，具体地，工作线程定义如下；typedef voID*Handler;Handler newHandler(voID*obj,HandlerCallBack cb)；回调函数定义如下：typedef int64_t(*HandlerCallBack)(voID*obj,int msg,int64_t wparam,int64_t lparam)。

[0098] 在一种可选的方案中，用户可以利用自己业务相关的回调函数，构建消息队列，消息队列中存储多个待处理的消息，由于通过工作线程同时只能对一个消息进行处理，可以通过工作线程获取消息队列头部的一个消息，并判断该消息时同步消息还是异步消息，如果确定该消息是同步消息，则可以调用回调函数对该消息进行处理，在确定回调函数对该消息处理完毕之后，可以继续对其他消息进行处理；如果确定该消息是异步消息，则可以调用回调函数对该消息进行处理，无需等待该消息的执行结果，直接对其他消息进行处理。对其他消息进行处理可以通过工作线程再次从消息队列的头部获取一个消息实现。

[0099] 在本发明上述实施例中，在通过工作线程从消息队列的头部获取到第一消息之后，可以首先判断第一消息的消息类型，在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息，在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消，从而实现多线程消息处理。与现有技术相比，仅通过工作线程调用回调函数对消息队列中存储的第一消息进行处理，而且可以对不同消息类型的消息执行不同的操作，从而达到了对用户隐藏多线程及其同步的细节，提高消息处理的便捷性，提高消息处理的兼容性，提升用户体验感和好感度的而技术效果，进而解决了现有技术中基于多线程的消息处理方法存在线程间同步复杂，资源竞争、死锁的技术问题。

[0100] 可选地，第一调用模块或第二调用模块还用于通过工作线程和第一消息的消息数据，调用回调函数对第一消息进行处理。

[0101] 可选地，第一调用模块还包括：第一唤醒单元，用于在消息类型为同步消息的情况下，通过工作线程唤醒同步调用线程，并为同步调用线程分配第一存储空间；第一调用单元，用于通过同步调用线程调用回调函数对第一消息进行处理。

[0102] 可选地，第一调用模块还包括：第一存储单元，用于通过同步调用线程将第一消息的消息数据存储在第一存储空间内；第一增加单元，用于通过同步调用线程在消息队列的尾部增加一条新消息。

[0103] 可选地，第一调用模块还包括：第二唤醒单元，用于通过同步调用线程唤醒工作线程；第一获取单元，用于通过工作线程从消息队列的头部获取第二消息。

[0104] 可选地，第二调用模块还包括：第三唤醒单元，用于在消息类型为异步消息的情况下，通过工作线程唤醒异步调用线程，并为异步调用线程分配第二存储空间；第二调用单元，用于通过异步调用线程调用回调函数对第一消息进行处理。

[0105] 可选地，第二调用模块还包括：第二存储单元，用于通过异步调用线程将第一消息的消息数据存储在第二存储空间内；第二增加单元，用于通过异步调用线程在消息队列的尾部增加一条新消息。

[0106] 可选地，第二调用模块还包括：第四唤醒单元，用于通过异步调用线程唤醒工作线程；第二获取单元，用于通过工作线程从消息队列的头部获取第二消息。

[0107] 可选地，第二调用模块还包括：释放单元，用于在通过异步调用线程唤醒工作线程之后，释放第二存储空间。

[0108] 可选地，该装置还包括：构造模块，用于在通过工作线程从消息队列的头部获取第一消息之前，构造消息队列，其中，该装置还包括：判断模块，用于在通过工作线程从消息队列的头部获取第二消息之前，判断是否检测到预设标志信息，其中，预设标志信息用于表征工作线程停止从消息队列的头部获取第二消息；析构模块，用于如果检测到预设标志信息，则析构消息队列；获取模块还用于如果未检测到预设标志信息，则通过工作线程从消息队列的头部获取第二消息。

[0109] 可选地，该装置还包括：创建模块，用于在通过工作线程从消息队列的头部获取第一消息之前，基于回调函数创建工作线程，并为工作线程分配第三存储空间。

[0110] 可选地，该装置还包括：分配模块，用于在通过工作线程从消息队列的头部获取第一消息之后，通过工作线程为第一消息分配第四存储空间，并将第一消息的消息数据存储在第四存储空间内。

[0111] 可选地，获取模块、第一调用模块或第二调用模块还包括：判断单元，用于判断消息队列中是否存在消息；第一获取单元、第二获取单元或第三获取单元还用于在确定消息队列中存在消息的情况下，通过工作线程从消息队列的头部获取第一消息或第二消息。

[0112] 可选地，第一获取单元、第二获取单元或第三获取单元还用于利用互斥机制，通过工作线程从消息队列的头部获取第一消息或第二消息；第一增加单元或第二增加单元还用于利用互斥机制，通过同步调用线程或异步调用线程在消息队列的尾部增加一条新消息。

[0113] 实施例3

[0114] 根据本发明实施例，还提供了一种消息处理方法的实施例，需要说明的是，在附图的流程图示出的步骤可以在诸如一组计算机可执行指令的计算机系统中执行，并且，虽然在流程图中示出了逻辑顺序，但是在某些情况下，可以以不同于此处的顺序执行所示出或

描述的步骤。

[0115] 图7是根据本发明实施例的另一种消息处理方法的流程图，如图7所示，该方法包括如下步骤：

[0116] 步骤S702，通过工作线程从消息队列的头部获取第一消息，其中，消息队列中存储至少一条消息。

[0117] 具体地，上述的工作线程可以是预先建立的用于对至少一条消息进行处理的线程，通过工作线程对至少一条消息进行处理，对于用户来说，可以实现多线程同步处理的效果，并且隐藏了多线程及其同步的细节，使得使用极为便利；上述的消息队列可以是根据需要处理需要预先建立的队列，至少一个待处理处理的消息存储在消息队列中，工作线程通过按顺序对消息队列中存储的消息进行处理，实现用户级的多线程同步处理。

[0118] 步骤S704，通过工作线程调用回调函数对第一消息进行处理。

[0119] 具体地，上述的回调函数可以是用户业务相关的消息处理函数，通过工作线程进行调用，可以处理消息队列中存储的消息。

[0120] 步骤S706，通过工作线程从消息队列的头部获取第二消息。

[0121] 具体地，消息队列中存储的消息可以分为同步消息和异步消息，对于同步消息，工作线程需要在同步消息执行完毕，得到执行结果之后才能执行其他消息，而对于异步消息，工作线程无需等待执行结果即可执行其他消息。

[0122] 需要说明的是，上述的工作线程、消息队列和回调函数可以是利用成熟的开源项目libuv实现跨平台的底层支持，将业务进行合理包装，并通过纯C语言编写的多线程消息通信框架，具体地，工作线程定义如下；typedef voID*Handler;Handler newHandler(voID*obj,HandlerCallBack cb)；回调函数定义如下：typedef int64_t(*HandlerCallBack)(voID*obj,int msg,int64_t wparam,int64_t lparam)。

[0123] 在一种可选的方案中，用户可以利用自己业务相关的回调函数，构建消息队列，消息队列中存储多个待处理的消息，由于通过工作线程同时只能对一个消息进行处理，可以通过工作线程获取消息队列头部的一个消息，并判断该消息时同步消息还是异步消息，如果确定该消息是同步消息，则可以调用回调函数对该消息进行处理，在确定回调函数对该消息处理完毕之后，可以继续对其他消息进行处理；如果确定该消息是异步消息，则可以调用回调函数对该消息进行处理，无需等待该消息的执行结果，直接对其他消息进行处理。对其他消息进行处理可以通过工作线程再次从消息队列的头部获取一个消息实现。

[0124] 在本发明上述实施例中，在通过工作线程从消息队列的头部获取到第一消息之后，可以通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消息。与现有技术相比，仅通过工作线程调用回调函数对消息队列中存储的第一消息进行处理，而且可以对不同消息类型的消息执行不同的操作，从而达到了对用户隐藏多线程及其同步的细节，提高消息处理的便捷性，提高消息处理的兼容性，提升用户体验感和好感度的而技术效果，进而解决了现有技术中基于多线程的消息处理方法存在线程间同步复杂，资源竞争、死锁的技术问题。

[0125] 实施例4

[0126] 根据本发明实施例，还提供了一种存储介质的实施例，存储介质包括存储的程序，其中，在程序运行时控制存储介质所在设备执行上述实施例1和3的消息处理方法。

[0127] 实施例5

[0128] 根据本发明实施例，还提供了一种处理器的实施例，处理器用于运行程序，其中，程序运行时执行上述实施例1和3的消息处理方法。

[0129] 上述本发明实施例序号仅仅为了描述，不代表实施例的优劣。

[0130] 在本发明的上述实施例中，对各个实施例的描述都各有侧重，某个实施例中没有详述的部分，可以参见其他实施例的相关描述。

[0131] 在本申请所提供的几个实施例中，应该理解到，所揭露的技术内容，可通过其它的方式实现。其中，以上所描述的装置实施例仅仅是示意性的，例如所述单元的划分，可以为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，单元或模块的间接耦合或通信连接，可以是电性或其它的形式。

[0132] 所述作为分离部件说明的单元可以是或者也可以不是物理上分开的，作为单元显示的部件可以是或者也可以不是物理单元，即可以位于一个地方，或者也可以分布到多个单元上。可以根据实际的需要选择其中的部分或者全部单元来实现本实施例方案的目的。

[0133] 另外，在本发明各个实施例中的各功能单元可以集成在一个处理单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现，也可以采用软件功能单元的形式实现。

[0134] 所述集成的单元如果以软件功能单元的形式实现并作为独立的产品销售或使用时，可以存储在一个计算机可读取存储介质中。基于这样的理解，本发明的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台计算机设备(可为个人计算机、服务器或者网络设备等)执行本发明各个实施例所述方法的全部或部分步骤。而前述的存储介质包括:U盘、只读存储器(ROM,Read-Only Memory)、随机存取存储器(RAM,Random Access Memory)、移动硬盘、磁碟或者光盘等各种可以存储程序代码的介质。

[0135] 以上所述仅是本发明的优选实施方式，应当指出，对于本技术领域的普通技术人员来说，在不脱离本发明原理的前提下，还可以做出若干改进和润饰，这些改进和润饰也应视为本发明的保护范围。

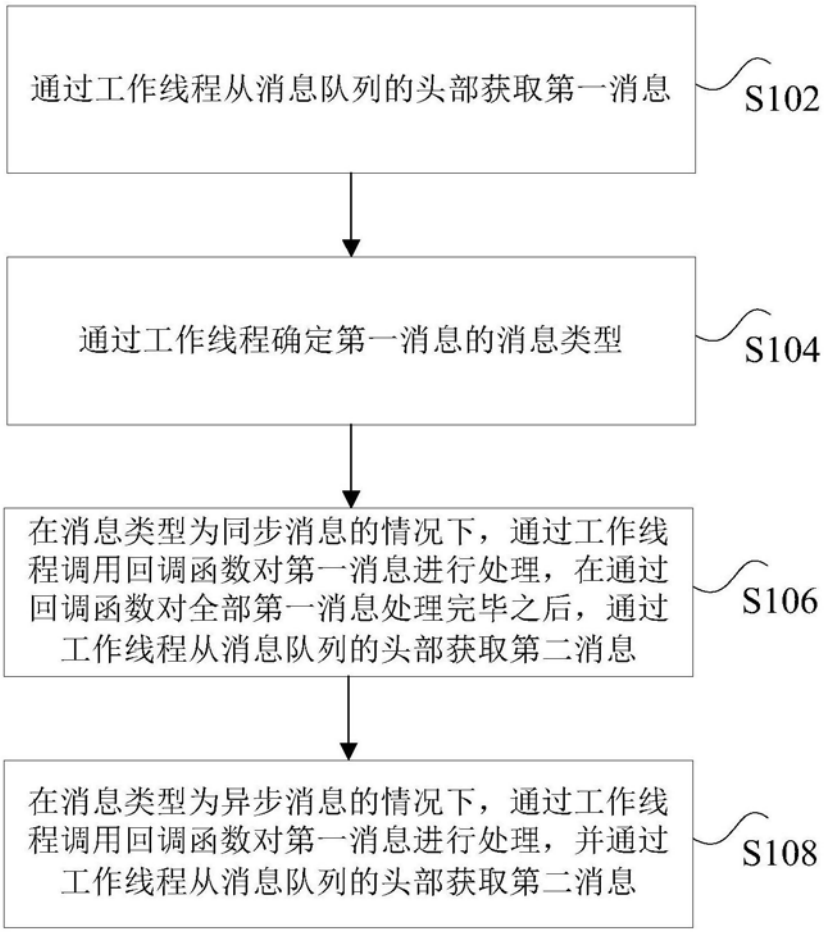

通过工作线程从消息队列的头部获取第一消息
S102
通过工作线程确定第一消息的消息类型
S104
在消息类型为同步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，在通过回调函数对全部第一消息处理完毕之后，通过工作线程从消息队列的头部获取第二消息
S106
在消息类型为异步消息的情况下，通过工作线程调用回调函数对第一消息进行处理，并通过工作线程从消息队列的头部获取第二消息
S108

图1

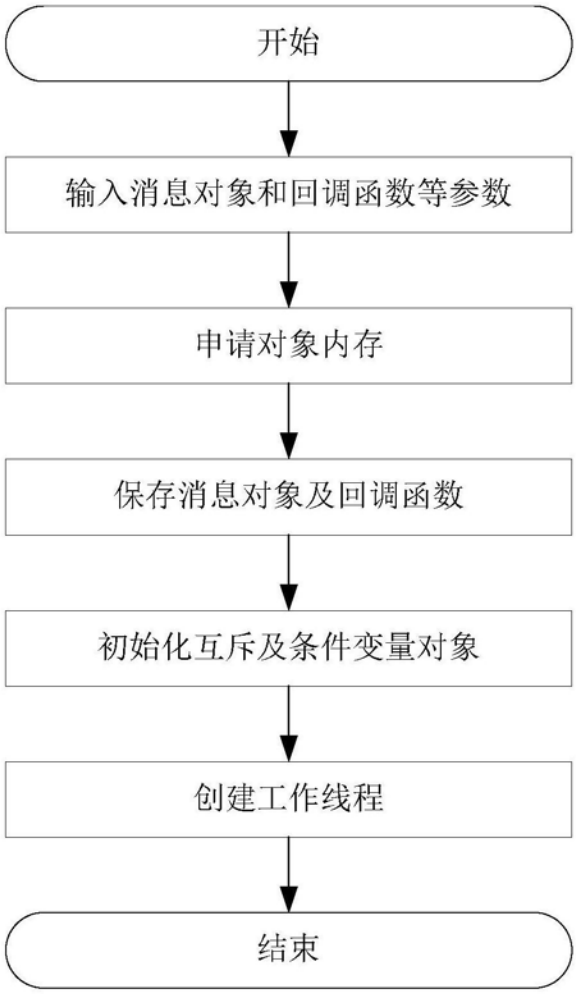

开始
输入消息对象和回调函数等参数
申请对象内存
保存消息对象及回调函数
初始化互斥及条件变量对象
创建工作线程
结束

图2

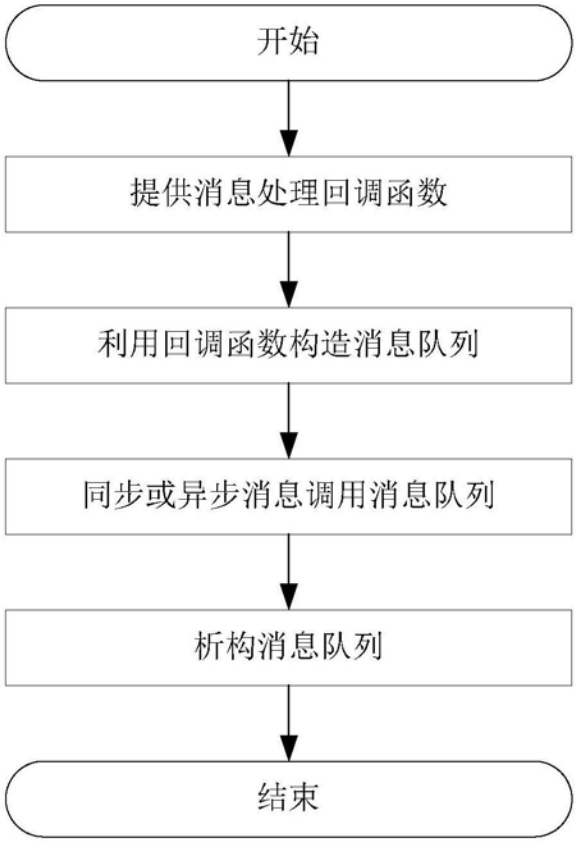


图3

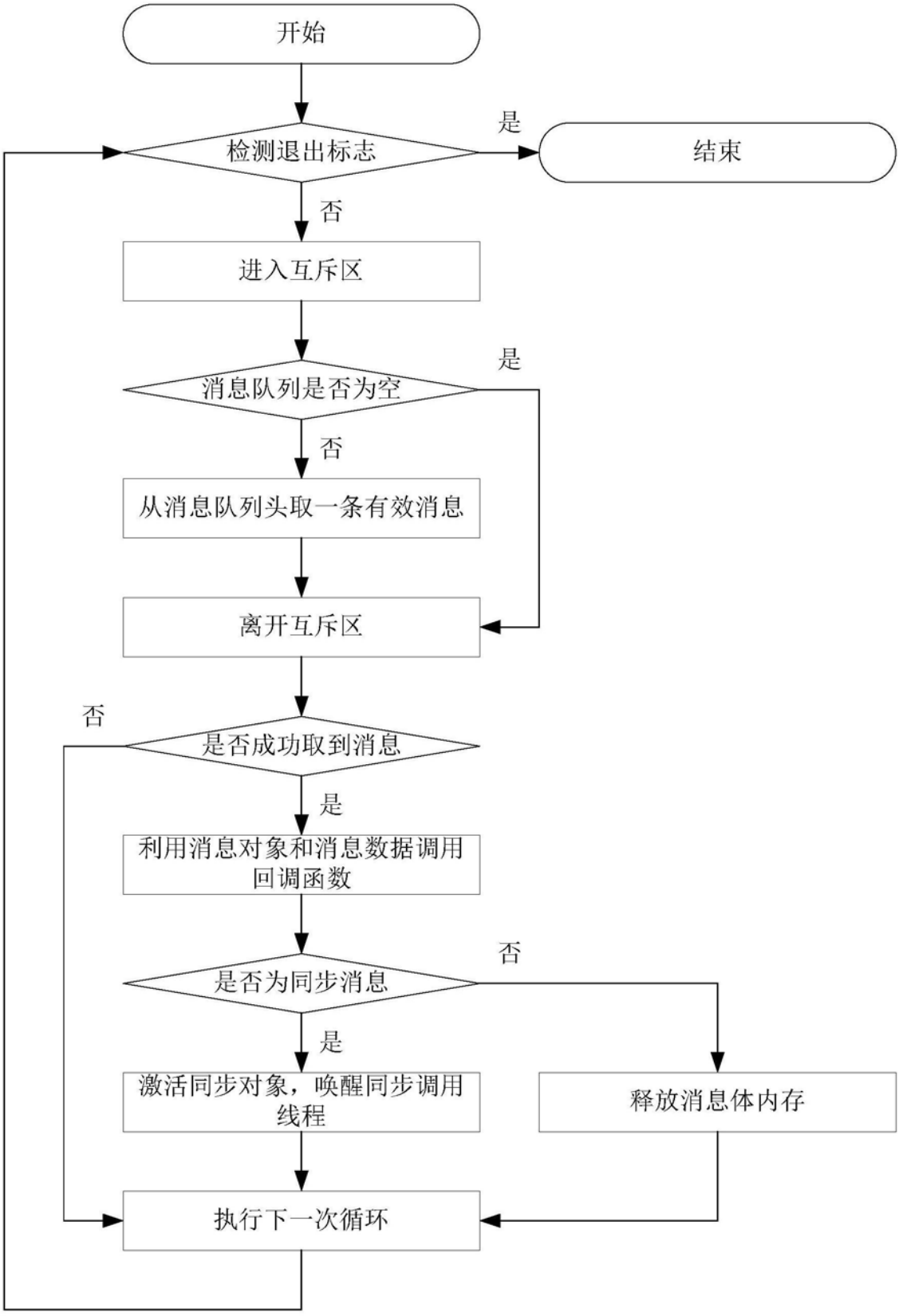

开始
检测退出标志
是
结束
否
进入互斥区
消息队列是否为空
是
否
从消息队列头取一条有效消息
离开互斥区
否
是否成功取到消息
是
利用消息对象和消息数据调用回调函数
否
是否为同步消息
是
激活同步对象，唤醒同步调用线程
释放消息体内存
执行下一次循环

图4

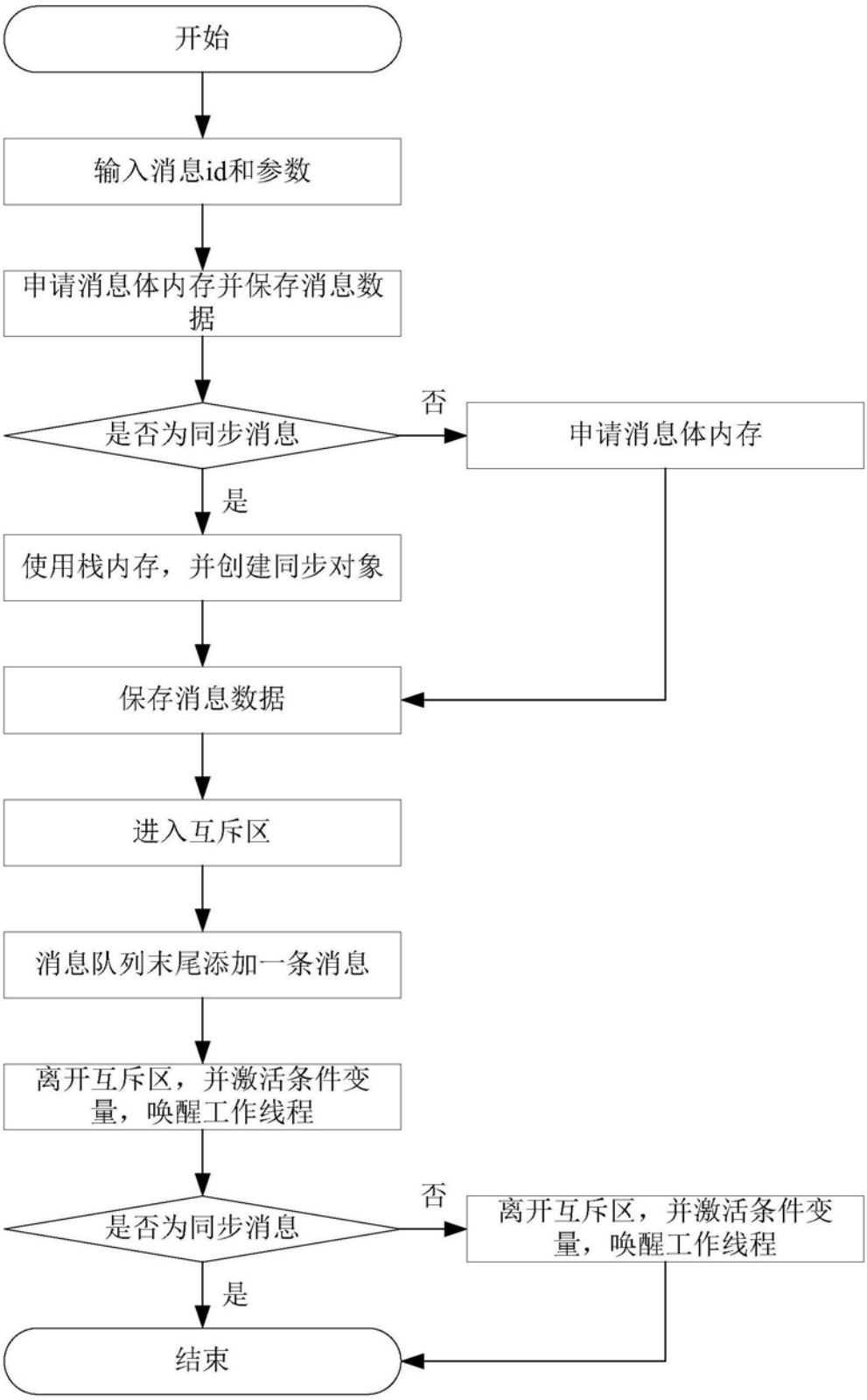

开始
输入消息id和参数
申请消息体内存并保存消息数据
是否为同步消息
否
申请消息体内存
是
使用栈内存，并创建同步对象
保存消息数据
进入互斥区
消息队列末尾添加一条消息
离开互斥区，并激活条件变量，唤醒工作线程
是否为同步消息
否
离开互斥区，并激活条件变量，唤醒工作线程
是
结束

图5

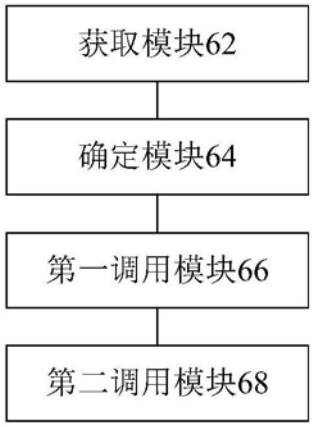

获取模块62
确定模块64
第一调用模块66
第二调用模块68

图6

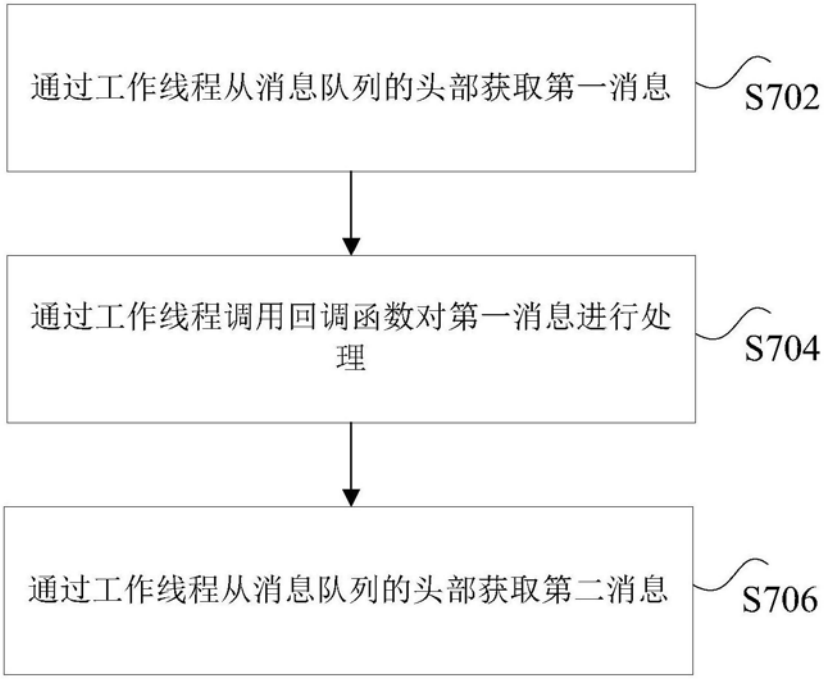

通过工作线程从消息队列的头部获取第一消息
S702
通过工作线程调用回调函数对第一消息进行处理
S704
通过工作线程从消息队列的头部获取第二消息
S706

图7