



(12) 发明专利

(10) 授权公告号 CN 112650613 B

(45) 授权公告日 2023. 09. 26

(21) 申请号 202011596034.3

(22) 申请日 2020.12.29

(65) 同一申请的已公布的文献号

申请公布号 CN 112650613 A

(43) 申请公布日 2021.04.13

(73) 专利权人 北京天融信网络安全技术有限公司

地址 100000 北京市海淀区上地东路1号院  
3号楼四层

专利权人 北京天融信科技有限公司  
北京天融信软件有限公司

(72) 发明人 邢国峰

(74) 专利代理机构 北京超凡宏宇知识产权代理有限公司 11463

专利代理师 余菲

(51) Int.Cl.

G06F 11/07 (2006.01)

G06F 16/174 (2019.01)

(56) 对比文件

CN 112035311 A, 2020.12.04

CN 109460343 A, 2019.03.12

CN 108256816 A, 2018.07.06

CN 104850487 A, 2015.08.19

CN 101916302 A, 2010.12.15

US 9201804 B1, 2015.12.01

审查员 林佳丽

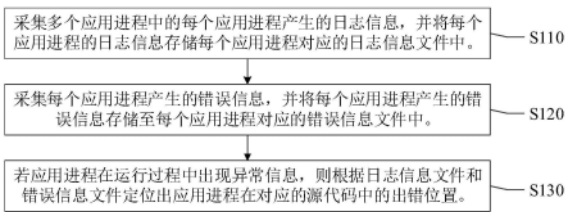
权利要求书2页 说明书10页 附图2页

(54) 发明名称

一种错误信息处理方法、装置、电子设备及存储介质

(57) 摘要

本申请提供一种错误信息处理方法、装置、电子设备及存储介质,用于改善对应用进程的出错位置进行定位的效率较低的问题。该方法包括:采集多个应用进程中的每个应用进程产生的日志信息,并将每个应用进程产生的日志信息存储至每个应用进程对应的日志信息文件中;采集每个应用进程产生的错误信息,并将每个应用进程产生的错误信息存储至每个应用进程对应的错误信息文件中;若应用进程在运行过程中出现异常信息,则根据日志信息文件和错误信息文件定位出应用进程在对应的源代码中的出错位置,在该方法通过使用自适应存储机制对错误信息进行处理,以使该方法适应于不同的硬件环境的存储要求。



1. 一种错误信息处理方法,其特征在于,包括:

采集多个应用进程中的每个应用进程产生的日志信息,并将所述每个应用进程产生的日志信息存储至所述每个应用进程对应的日志信息文件中;

采集所述每个应用进程产生的错误信息,并将所述每个应用进程产生的错误信息存储至所述每个应用进程对应的错误信息文件中;

若所述应用进程在运行过程中出现异常信息,则根据所述日志信息文件和所述错误信息文件定位出所述应用进程在对应的源代码中的出错位置;

其中,在所述采集多个应用进程中的每个应用进程产生的日志信息之前,还包括:从配置文件中获取总存储空间和单个文件阈值,所述总存储空间是分配给所述日志信息文件和所述错误信息文件所占用的总存储空间,所述单个文件阈值是所述错误信息文件所占存储空间的总阈值;将所述总存储空间除以所述单个文件阈值,获得文件个数阈值,所述文件个数阈值是存储所述错误信息文件的最小阈值;

若所述文件个数阈值小于预设个数阈值,且所述总存储空间小于物理存储空间,则将所述总存储空间设置为所述预设个数阈值与所述单个文件阈值的乘积。

2. 根据权利要求1所述的方法,其特征在于,所述将所述总存储空间除以所述单个文件阈值,获得文件个数阈值,包括:

判断从所述配置文件中是否查找到压缩开关配置项;

若是,则将所述单个文件阈值乘以预设压缩比,获得压缩文件阈值,并将所述总存储空间除以所述压缩文件阈值,获得文件个数阈值。

3. 根据权利要求1所述的方法,其特征在于,还包括:

若所述日志信息文件和所述错误信息文件的占用存储空间超过所述总存储空间的第一预设比例,则停止采集所述每个应用进程产生的日志信息。

4. 根据权利要求3所述的方法,其特征在于,在所述停止采集所述每个应用进程产生的日志信息之后,还包括:

若所述日志信息文件和所述错误信息文件的占用存储空间超过所述总存储空间的第二预设比例,则删除所述每个应用进程产生的日志信息,所述第二预设比例大于所述第一预设比例。

5. 根据权利要求4所述的方法,其特征在于,在所述删除所述每个应用进程产生的日志信息之后,还包括:

若所述错误信息文件的占用存储空间超过所述总存储空间的第三预设比例,则对所述错误信息文件进行压缩,获得压缩后的错误信息文件,所述第三预设比例大于所述第二预设比例。

6. 根据权利要求1-4任一所述的方法,其特征在于,所述根据所述日志信息文件和所述错误信息文件定位出所述应用进程在对应的源代码中的出错位置,包括:

获取所述异常信息在所述应用进程对应日志信息文件中的异常位置,以及所述异常信息在所述应用进程对应错误信息文件中的堆栈信息;

根据所述异常位置和所述堆栈信息对所述应用进程对应的源代码进行错误定位,获得所述应用进程在所述源代码中的错误位置。

7. 一种错误信息处理装置,其特征在于,包括:

日志信息采集模块,用于采集多个应用进程中的每个应用进程产生的日志信息,并将所述每个应用进程产生的日志信息存储至所述每个应用进程对应的日志信息文件中;

错误信息采集模块,用于采集所述每个应用进程产生的错误信息,并将所述每个应用进程产生的错误信息存储至所述每个应用进程对应的错误信息文件中;

出错位置定位模块,用于若所述应用进程在运行过程中出现异常信息,则根据所述日志信息文件和所述错误信息文件定位出所述应用进程在对应的源代码中的出错位置;

其中,在所述采集多个应用进程中的每个应用进程产生的日志信息之前,还包括:从配置文件中获取总存储空间和单个文件阈值,所述总存储空间是分配给所述日志信息文件和所述错误信息文件所占用的总存储空间,所述单个文件阈值是所述错误信息文件所占存储空间的阈值;将所述总存储空间除以所述单个文件阈值,获得文件个数阈值,所述文件个数阈值是存储所述错误信息文件的最小阈值;若所述文件个数阈值小于预设个数阈值,且所述总存储空间小于物理存储空间,则将所述总存储空间设置为所述预设个数阈值与所述单个文件阈值的乘积。

8.一种电子设备,其特征在于,包括:处理器和存储器,所述存储器存储有所述处理器可执行的机器可读指令,所述机器可读指令被所述处理器执行时执行如权利要求1至6任一所述的方法。

9.一种存储介质,其特征在于,该存储介质上存储有计算机程序,该计算机程序被处理器运行时执行如权利要求1至6任一所述的方法。

## 一种错误信息处理方法、装置、电子设备及存储介质

### 技术领域

[0001] 本申请涉及计算机信息处理和网络安全的技术领域,具体而言,涉及一种错误信息处理方法、装置、电子设备及存储介质。

### 背景技术

[0002] 内存转储是用于系统崩溃时,将内存中的数据转储保存在转储文件中,供给有关人员进行排错分析用途;而它所保存生成的文件就叫做内存转储文件;此处的内存转储文件又被称为核心文件、虚拟内存文件或核心内存转储等。

[0003] 为了定位出应用进程出现问题的原因,需要事先收集该应用进程相应的程序运行信息,然后对程序运行信息进行分析,找到问题原因。目前针对程序运行信息的收集和保存大都是依赖于操作系统的内存转储文件,具体例如:运行于Linux操作系统下的应用进程发生崩溃时,由于操作系统会在运行目录下产生一个名字为“core.进程号”的内存转储文件,可以周期性地检测内存转储文件是否存在,若存在,则将内存转储文件从运行目录移动至临时目录,再使用脚本程序和调试程序对临时目录下的内存转储文件进行分析定位,从而定位出该应用进程在源代码的出错位置。在具体的实践过程中发现,由于内存转储文件依赖于操作系统,所以在获取内存存储文件时存在很多缺陷,这些缺陷例如:需要打开生成开关、内存转储文件的尺寸较大、占用空间大、内容繁多以及信息不全面,这些缺陷导致使用内存转储文件对应用进程的出错位置进行定位的效率较低。

### 发明内容

[0004] 本申请实施例的目的在于提供一种错误信息处理方法、装置、电子设备及存储介质,用于改善对应用进程的出错位置进行定位的效率较低的问题,与此同时,也能够摆脱对操作系统中内存转储文件的依赖,并减少了对存储空间占用,从而提高了对应用进程的出错位置进行定位的效率。

[0005] 本申请实施例提供了一种错误信息处理方法,包括:采集多个应用进程中的每个应用进程产生的日志信息,并将每个应用进程产生的日志信息存储至每个应用进程对应的日志信息文件中;采集每个应用进程产生的错误信息,并将每个应用进程产生的错误信息存储至每个应用进程对应的错误信息文件中;若应用进程在运行过程中出现异常信息,则根据日志信息文件和错误信息文件定位出应用进程在对应的源代码中的出错位置。在上述的实现过程中,通过将每个应用进程产生的日志信息和错误信息分开存放,分别存储至日志信息文件和错误信息文件中,并根据日志信息文件和错误信息文件来定位出应用进程在对应的源代码中的出错位置,具体地,可以优先使用错误信息文件进行错误定位,在需要辅助信息时,可以结合使用错误信息文件和错误信息文件对应的日志信息文件来定位出应用进程在对应的源代码中的出错位置;避免了依赖于操作系统的内存转储文件对应用进程的出错位置进行定位的情况,有效地利用日志信息文件和错误信息文件中的信息,从而提高了对应用进程的出错位置进行定位的效率。

[0006] 可选地,在本申请实施例中,在采集多个应用进程中的每个应用进程产生的日志信息之前,还包括:从配置文件中获取总存储空间和单个文件阈值,总存储空间是分配给日志信息文件和错误信息文件所占用的总存储空间,单个文件阈值是错误信息文件所占用存储空间的最大阈值;将总存储空间除以单个文件阈值,获得文件个数阈值,文件个数阈值是存储错误信息文件的最小阈值;若文件个数阈值小于预设个数阈值,且总存储空间小于物理存储空间,则将总存储空间设置为预设个数阈值与单个文件阈值的乘积。在上述的实现过程中,通过预先评估总存储空间不够的情况下,将总存储空间设置为预设个数阈值与单个文件阈值的乘积,从而避免在不同的硬件设备允许使用存储空间不一致导致可能存储失败的问题,屏蔽了各种设备对存储空间的限制,有效地增加了对各种设备的存储空间的适应性。

[0007] 可选地,在本申请实施例中,将总存储空间除以单个文件阈值,获得文件个数阈值,包括:判断从配置文件中是否查找到压缩开关配置项;若是,则将单个文件阈值乘以预设压缩比,获得压缩文件阈值,并将总存储空间除以压缩文件阈值,获得文件个数阈值。

[0008] 可选地,在本申请实施例中,还包括:若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第一预设比例,则停止采集每个应用进程产生的日志信息。在上述的实现过程中,通过在日志信息文件和错误信息文件的占用存储空间超过总存储空间的第一预设比例的情况下,停止采集每个应用进程产生的日志信息,从而避免了日志信息采集过多导致错误信息无法存储的问题,有效地优先保证了错误信息能够得以存储,从而保证应用进程在发生异常时,能够有效地进行错误定位。

[0009] 可选地,在本申请实施例中,在停止采集每个应用进程产生的日志信息之后,还包括:若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第二预设比例,则删除每个应用进程产生的日志信息,第二预设比例大于第一预设比例。在上述的实现过程中,通过在日志信息文件和错误信息文件的占用存储空间超过总存储空间的第二预设比例的情况下,删除每个应用进程产生的日志信息,从而避免了日志信息采集过多导致错误信息无法存储的问题,有效地优先保证了错误信息能够得以存储,从而保证应用进程在发生异常时,能够有效地进行错误定位。

[0010] 可选地,在本申请实施例中,在删除每个应用进程产生的日志信息之后,还包括:若错误信息文件的占用存储空间超过总存储空间的第三预设比例,则对错误信息文件进行压缩,获得压缩后的错误信息文件,第三预设比例大于第二预设比例。在上述的实现过程中,通过在错误信息文件的占用存储空间超过总存储空间的第三预设比例的情况下,对错误信息文件进行压缩,获得压缩后的错误信息文件;避免了错误信息文件过大而导致无法将错误信息存储的问题,从而保证应用进程在发生异常时,能够有效地进行错误定位。

[0011] 可选地,在本申请实施例中,根据日志信息文件和错误信息文件定位出应用进程在对应的源代码中的出错位置,包括:获取异常信息在应用进程对应日志信息文件中的异常位置,以及异常信息在应用进程对应错误信息文件中的堆栈信息;根据异常位置和堆栈信息对应用进程对应的源代码进行错误定位,获得应用进程在源代码中的错误位置。在上述的实现过程中,通过获取异常信息在应用进程对应日志信息文件中的异常位置,以及异常信息在应用进程对应错误信息文件中的堆栈信息;然后根据异常位置和堆栈信息对应用进程对应的源代码进行错误定位,从而获得应用进程在源代码中的错误位置,有效地提高

了错误定位的速度和效率。

[0012] 本申请实施例还提供了一种错误信息处理装置,包括:日志信息采集模块,用于采集多个应用进程中的每个应用进程产生的日志信息,并将每个应用进程产生的日志信息存储至每个应用进程对应的日志信息文件中;错误信息采集模块,用于采集每个应用进程产生的错误信息,并将每个应用进程产生的错误信息存储至每个应用进程对应的错误信息文件中;出错位置定位模块,用于若应用进程在运行过程中出现异常信息,则根据日志信息文件和错误信息文件定位出应用进程在对应的源代码中的出错位置。

[0013] 可选地,在本申请实施例中,错误信息处理装置,还包括:配置参数获取模块,用于从配置文件中获取总存储空间和单个文件阈值,总存储空间是分配给日志信息文件和错误信息文件所占用的总存储空间,单个文件阈值是错误信息文件所占用存储空间的最大阈值;个数阈值获得模块,用于将总存储空间除以单个文件阈值,获得文件个数阈值,文件个数阈值是存储错误信息文件的最小阈值;存储空间设置模块,用于若文件个数阈值小于预设个数阈值,且总存储空间小于物理存储空间,则将总存储空间设置为预设个数阈值与单个文件阈值的乘积。

[0014] 可选地,在本申请实施例中,个数阈值获得模块,包括:压缩开关判断模块,用于判断从配置文件中是否查找到压缩开关配置项;个数阈值计算模块,用于若从配置文件中查找到压缩开关配置项,则将单个文件阈值乘以预设压缩比,获得压缩文件阈值,并将总存储空间除以压缩文件阈值,获得文件个数阈值。

[0015] 可选地,在本申请实施例中,错误信息处理装置,还包括:若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第一预设比例,则停止采集每个应用进程产生的日志信息。

[0016] 可选地,在本申请实施例中,错误信息处理装置,还包括:若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第二预设比例,则删除每个应用进程产生的日志信息,第二预设比例大于第一预设比例。

[0017] 可选地,在本申请实施例中,错误信息处理装置,还包括:若错误信息文件的占用存储空间超过总存储空间的第三预设比例,则对错误信息文件进行压缩,获得压缩后的错误信息文件,第三预设比例大于第二预设比例。

[0018] 可选地,在本申请实施例中,出错位置定位模块,包括:堆栈信息获取模块,用于获取异常信息在应用进程对应日志信息文件中的异常位置,以及异常信息在应用进程对应错误信息文件中的堆栈信息;代码错误定位模块,用于根据异常位置和堆栈信息对应用进程对应的源代码进行错误定位,获得应用进程在源代码中的错误位置。

[0019] 本申请实施例还提供了一种电子设备,包括:处理器和存储器,存储器存储有处理器可执行的机器可读指令,机器可读指令被处理器执行时执行如上面描述的方法。

[0020] 本申请实施例还提供了一种存储介质,该存储介质上存储有计算机程序,该计算机程序被处理器运行时执行如上面描述的方法。

## 附图说明

[0021] 为了更清楚地说明本申请实施例的技术方案,下面将对本申请实施例中所需要使用的附图作简单地介绍,应当理解,以下附图仅示出了本申请的某些实施例,因此不应被看

作是对范围的限定,对于本领域普通技术人员来讲,在不付出创造性劳动的前提下,还可以根据这些附图获得其他相关的附图。

[0022] 图1示出的本申请实施例提供的错误信息处理方法的流程示意图;

[0023] 图2示出的本申请实施例提供的对日志信息文件和错误信息文件的存储空间进行管理的流程示意图;

[0024] 图3示出的本申请实施例提供的错误信息处理装置的结构示意图;

[0025] 图4示出的本申请实施例提供的电子设备的结构示意图。

## 具体实施方式

[0026] 下面将结合本申请实施例中的附图,对本申请实施例中的技术方案进行清楚、完整的描述。

[0027] 在介绍本申请实施例提供的错误信息处理方法之前,先介绍本申请实施例中所涉及的一些概念:

[0028] 日志信息(Log Information),是指记录系统中硬件、软件和系统运行时的必要信息,同时还可以软件进程中发生的事件信息;可以通过它来辅助检查错误发生的原因;这里的日志信息包括应用进程在运行过程中产生的日志信息。

[0029] 错误信息(Error Information),是指当应用程序进程出现异常情况时,需要收集的异常点信息,此处的错误信息可以包括:出现异常的进程号、异常发生的时间、异常进程的程序路径、异常产生的原因、进程栈信息、所有线程栈信息、当前运行的进程记录、错误描述以及错误点的堆栈信息等等信息。

[0030] 面向切面编程(Aспект-Оriented Programming,AOP),又被称为面向切面程序设计或者面向剖面导向程序设计,是指计算机科学中的一种程序设计思想,旨在将横切关注点与业务主体进行进一步分离,以提高程序代码的模块化程度;通过在现有代码基础上增加额外的通知(Advice)机制,能够对被声明为“切点(Pointcut)”的代码块进行统一管理与装饰;该思想使得开发人员能够将与代码核心业务逻辑关系不那么密切的功能(如日志功能)添加至程序中,同时又不降低业务代码的可读性。

[0031] 服务器是指通过网络提供计算服务的设备,服务器例如:x86服务器以及非x86服务器,非x86服务器包括:大型机、小型机和UNIX服务器。

[0032] 需要说明的是,本申请实施例提供的错误信息处理方法可以被电子设备执行,这里的电子设备是指具有执行计算机程序功能的设备终端或者上述的服务器,设备终端例如:智能手机、个人电脑(personal computer,PC)、平板电脑、个人数字助理(personal digital assistant,PDA)、移动上网设备(mobile Internet device,MID)、网络交换机或网络路由器等。

[0033] 在介绍本申请实施例提供的错误信息处理方法之前,先介绍该错误信息处理方法适用的应用场景,这里的应用场景包括但不限于:使用该错误信息处理方法对应用进程在运行过程中出现的异常情况进行错误定位,或者,使用该错误信息处理方法增强监控系统或者监控程序的功能,或者,使用该错误信息处理方法收集操作系统上的应用进程的日志信息或者错误信息等等。

[0034] 请参见图1示出的本申请实施例提供的错误信息处理方法的流程示意图;该错误

信息处理方法的主要思路是,通过将每个应用进程产生的日志信息和错误信息分开存放,分别存储至日志信息文件和错误信息文件中,并根据日志信息文件和错误信息文件来定位出应用进程在对应的源代码中的出错位置,避免了依赖于操作系统的内存转储文件对应用进程的出错位置进行定位的情况,有效地利用日志信息文件和错误信息文件中的信息,从而提高了对应用进程的出错位置进行定位的效率;上述的错误信息处理方法可以包括:

[0035] 步骤S110:采集多个应用进程中的每个应用进程产生的日志信息,并将每个应用进程产生的日志信息存储至每个应用进程对应的日志信息文件中。

[0036] 日志信息文件,是指存储多个应用进程中的每个应用进程产生的日志信息的文件,该日志信息文件可以存储需要关注的重要信息,该文件中的日志信息的格式例如:进程号(又称为进程标识或者进程ID)、信息关键字和信息内容等等。

[0037] 上述步骤S110的实施方式例如:使用面向切面编程(AOP)的方式在代码中设置常规日志信息采集标记点,使用常规日志收集程序模块在采集标记点处来收集常规的日志信息;可以使用常规日志收集程序模块采集多个应用进程中的每个应用进程在运行过程中产生的日志信息,并将每个应用进程产生的日志信息存储至每个应用进程对应的日志信息文件中;其中,此处的日志信息是常规的日志信息,即在程序正常运行过程中,需要关注的重要信息,包括:过程信息、函数调用信息、运行状态信息要参数的值等信息。在具体的实施过程中,使用面向切面编程(AOP)的方式可以采用AOP框架来采集日志信息,可以使用的AOP框架包括但不限于:AspectWerkz、JBoss AOP、DynamicAspects和SpringAOP等等框架。

[0038] 可选地,由于应用进程所在的硬件环境千差万别,不同环境的存储空间大小不同;尤其是网络安全设备,存储空间十分宝贵,所以需要错误信息文件所占用的空间进行限制,防止出现耗尽设备存储空间的问题。如果针对每种硬件环境单独限制错误信息的控制文件,操作管理会比较复杂。因此,在采集日志信息以及错误信息之前,或者,在采集日志信息以及错误信息的过程中,还可以动态地对错误信息文件和压缩错误信息文件所需的存储空间进行管理,设计优先保存错误信息文件和压缩错误信息文件的机制,这种机制又被称为自适应机制,该实施方式可以包括:

[0039] 步骤S111:从配置文件中获取总存储空间和单个文件阈值。

[0040] 上述步骤S111的实施方式例如:从INI文件(Initialization File)格式和可扩展标记语言(eXtensible Markup Language,XML)格式的配置文件中获取总存储空间和单个文件阈值;其中,总存储空间是分配给日志信息文件和错误信息文件所占用的总存储空间,单个文件阈值是错误信息文件所占用存储空间的最大阈值。

[0041] 步骤S112:将总存储空间除以单个文件阈值,获得文件个数阈值。

[0042] 上述步骤S112的实施方式包括:在采集信息之前,先从配置文件中查找压缩开关;然后,判断从配置文件中是否查找到压缩开关配置项;若从配置文件中查找到压缩开关配置项,则将单个文件阈值乘以预设压缩比,获得压缩文件阈值,最后,将总存储空间除以压缩文件阈值,获得文件个数阈值;其中,文件个数阈值是存储错误信息文件的最小阈值,上述的预设压缩比可以根据具体情况进行设置,例如将预设压缩比设置为10:1或者20:1等等。

[0043] 步骤S113:若文件个数阈值小于预设个数阈值,且总存储空间小于物理存储空间,则将总存储空间设置为预设个数阈值与单个文件阈值的乘积。

[0044] 步骤S114:若文件个数阈值大于预设个数阈值,且总存储空间小于物理存储空间,则将总存储空间设置为文件个数阈值与单个文件阈值的乘积。

[0045] 步骤S115:若总存储空间大于或等于物理存储空间,则将总存储空间设置为物理存储空间,并生成和输出预警信息。

[0046] 上述步骤S113至步骤S115的实施方式包括:从INI格式或者XML格式的配置文件获取预设个数阈值,并通过操作系统的应用程序接口(Application Programming Interface,API)获取物理存储空间,此处的物理存储空间是指操作系统实际访问的物理存储空间,然后根据文件个数阈值、预设个数阈值、总存储空间和物理存储空间之间的关系来设置总存储空间,具体例如:若文件个数阈值小于预设个数阈值,且总存储空间小于物理存储空间,则将总存储空间设置为预设个数阈值与单个文件阈值的乘积;若文件个数阈值大于预设个数阈值,且总存储空间小于物理存储空间,则将总存储空间设置为文件个数阈值与单个文件阈值的乘积;若总存储空间大于或等于物理存储空间,则将总存储空间设置为物理存储空间,并生成和输出预警信息。

[0047] 在上述实现过程中,通过使用自适应存储机制对错误信息进行处理,以使上述的错误信息处理方法可以适应于不同的硬件环境的存储要求。

[0048] 步骤S120:采集每个应用进程产生的错误信息,并将每个应用进程产生的错误信息存储至每个应用进程对应的错误信息文件中。

[0049] 错误信息文件,是指存储多个应用进程中的每个应用进程产生的错误信息的文件,该错误信息文件可以采集的信息包括:出错点的进程栈信息、所有线程栈信息、崩溃现场的函数栈信息和当前运行的进程记录等。其中,当监测到程序进程中有异常发生时,可以调用错误信息程序收集模块来采集当前异常点错误信息,并将当前异常点错误信息存储至错误信息文件;错误信息文件中的崩溃现场函数栈信息非常重要,在实际实施过程中可以借助适当的堆栈回溯方式,然后使用适当的堆栈回溯方式,获取程序出错点的堆栈信息。

[0050] 上述步骤S120的实施方式例如:通过事先在应用进程对应的应用程序源代码中添加满足错误出现的预设条件,若应用进程在运行到满足该预设条件时就抛出异常(ThrowException)或者抛出错误(Throw Error)的方式来产生错误信息;然后对源代码进行编译、打包和运行,获得应用进程。在应用进程的运行过程中,若监测到应用进程异常发生(即应用进程抛出了异常)或者应用进程发生崩溃(即应用进程抛出了错误),则使用错误信息程序收集模块来采集每个应用进程产生的错误信息,并将每个应用进程产生的错误信息存储至每个应用进程对应的错误信息文件中。其中,上述使用错误信息程序收集模块来采集每个应用进程产生的错误信息的具体过程例如:在应用经常进入系统默认崩溃处理流程前,调用错误收集程序模块采集当前异常点的错误信息,并将收集到的错误信息追加(Append)至对应的错误信息文件中。

[0051] 可选地,当上述多个应用进程中的每个应用进程对应的日志信息文件和错误信息文件记录到一段时间后,这段时间可以根据具体情况进行设置,例如:在一个小时内采集完每个应用进程对应的日志信息文件和错误信息文件之后,可以将该日志信息文件和错误信息文件命名为“进程号-日志类型-当前时间戳”的文件,日志类型包括日志信息和错误信息,这个过程可以理解为日志回滚,日志回滚过程可以由错误信息处理程序启动日志回滚程序进程来完成。在日志回滚完成后,就获得多个进程在多个时间段内产生的日志信息文

件和错误信息文件,然后操作系统还可以单独启动日志合并程序进程对上述多个进程在多个时间段内产生的日志信息文件或错误信息文件进行合并,获得总日志信息文件或总错误信息文件;其中,合并的方式有很多种,包括但不限于:第一种方式,根据进程号合并当天的或者当月的日志信息文件或错误信息文件;第二种方式,根据日志类型合并当天的或者当月的所有进程号的日志信息文件或错误信息文件。

[0052] 需要说明的是,上述的步骤S110和步骤S120的执行顺序可以不作限制,可以步骤S110先执行且步骤S120后执行,也可以步骤S120先执行且步骤S110后执行,当然也可以步骤S110和步骤S120并行执行,即步骤S110和步骤S120一起执行。

[0053] 在步骤S110和步骤S120之后,执行步骤S130:若应用进程在运行过程中出现异常信息,则根据日志信息文件和错误信息文件定位出应用进程在对应的源代码中的出错位置。

[0054] 上述步骤S130的实施方式例如:若应用进程在运行过程中出现异常信息,则根据日志信息文件和错误信息文件定位出应用进程在对应的源代码中的出错位置,包括:获取异常信息在应用进程对应日志信息文件中的异常位置,以及异常信息在应用进程对应错误信息文件中的堆栈信息。根据异常位置和堆栈信息对应用进程对应的源代码进行错误定位,获得应用进程在源代码中的错误位置;错误定位的具体过程例如:首先读取到错误信息文件的内容,并对错误信息文件进行解析,在解析完成后,获取最后一行错误信息记录,该行的错误信息记录就是出错点时刻的信息状态;然后,从该行的错误信息记录中获取发生错误的进程号(进程号又被称为进程标识或者进程ID)、异常发生的时间、异常进程的程序路径、异常产生的原因、进程栈信息、所有线程栈信息、错误描述以及错误点的堆栈信息等等信息;最后,可以结合日志信息文件中的相关信息和错误点的堆栈信息分析原因,如果需要其它信息,则通过进程号(ID)找到该应用进程对应的日志信息文件,然后根据日志信息文件中的关键信息标识(ID)获取到对应问题点的日志记录,根据该日志记录和获取到的堆栈信息确定出应用进程在对应的源代码中的出错位置;也可以根据错误信息文件确定应用进程在对应的源代码中的出错位置,即单独解析错误点的堆栈信息,再结合错误描述以及错误堆栈信息可以确定应用进程出现问题的原因,以及应用进程在对应的源代码中的出错位置。

[0055] 在上述的获取堆栈信息的过程中,可以采用堆栈回溯的方式来获取异常信息在应用进程对应错误信息文件中的堆栈信息,具体例如:获得当前函数的帧基指针(Base Pointer, BP);然后通过帧基指针偏移获得主调函数的返回地址,并读取当前帧基指针指向的内存内容,获得主调函数的帧基指针地址;最后,循环执行以上步骤直至到达堆栈底部,即可完成获取堆栈信息。当然,在具体的实施过程中,可以直接使辅助工具来完成获取堆栈信息,这些辅助工具包括但不限于:backtrace函数、dump\_stack函数或者UNIX及UNIX-like操作系统下的gdb调试工具等等工具来获取函数调用的堆栈信息。

[0056] 在上述的实现过程中,通过将每个应用进程产生的日志信息和错误信息分开存放,分别存储至日志信息文件和错误信息文件中,并根据日志信息文件和错误信息文件来定位出应用进程在对应的源代码中的出错位置,具体地,可以优先使用错误信息文件进行错误定位,在需要辅助信息时,可以结合使用错误信息文件和错误信息文件对应的日志信息文件来定位出应用进程在对应的源代码中的出错位置;避免了依赖于操作系统的内存转

储文件对应用进程的出错位置进行定位的情况,有效地利用日志信息文件和错误信息文件中的信息,从而提高了对应用进程的出错位置进行定位的效率,与此同时,也能够摆脱对操作系统中内存转储文件的依赖,并减少了对存储空间占用,从而提高了对应用进程的出错位置进行定位的效率。

[0057] 请参见图2示出的本申请实施例提供的对日志信息文件和错误信息文件的存储空间进行管理的流程示意图;可选地,在获得日志信息文件和错误信息文件之后,随着日志信息文件和错误信息文件不断存储信息,日志信息文件和错误信息文件所占的存储空间也不断增大,此时可以根据下面的步骤对日志信息文件和错误信息文件的存储空间进行管理,上述的错误信息处理方法还可以包括:

[0058] 步骤S210:若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第一预设比例,则停止采集每个应用进程产生的日志信息。

[0059] 预设比例,是指所需要的存储空间占总存储空间的比例,此处的预设比例可以根据具体情况进行设置;具体例如:将第一预设比例设置为70%,第二预设比例设置80%,并将第三预设比例设置为90%等等。

[0060] 上述步骤S210的实施方式例如:获取操作系统中配置的允许该日志信息文件和错误信息文件所占用的总存储空间,以及从文件系统中获取日志信息文件和错误信息文件的实际占用存储空间;若日志信息文件和错误信息文件的实际占用存储空间超过总存储空间的70%,则停止采集每个应用进程产生的日志信息,即仅采集每个应用进程产生的错误信息,并将采集的错误信息保存至错误信息文件中。

[0061] 可选地,在步骤S210之后,执行步骤S220:若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第二预设比例,则删除每个应用进程产生的日志信息,第二预设比例大于第一预设比例。

[0062] 上述步骤S220的实施方式例如:获取操作系统中配置的允许该日志信息文件和错误信息文件所占用的总存储空间,以及从文件系统中获取日志信息文件和错误信息文件的实际占用存储空间;若日志信息文件和错误信息文件的实际占用存储空间超过总存储空间的80%,则删除每个应用进程产生的日志信息,从而为错误信息的采集和存储提供存储空间,以保证错误信息能够存储到错误信息文件中。

[0063] 可选地,在步骤S220之后,执行步骤S230:若错误信息文件的占用存储空间超过总存储空间的第三预设比例,则对错误信息文件进行压缩,获得压缩后的错误信息文件,第三预设比例大于第二预设比例。

[0064] 上述步骤S230的实施方式例如:获取操作系统中配置的允许该日志信息文件和错误信息文件所占用的总存储空间,以及从文件系统中获取错误信息文件的实际占用存储空间;若错误信息文件的实际占用存储空间超过总存储空间的90%,则按照每日或者每月的时间段对错误信息文件进行压缩,获得压缩后的错误信息文件。在具体的实施过程中,若多个压缩后的错误信息文件的实际占用存储空间仍然超过总存储空间的90%,则可以生成并输出提醒信息,以提醒操作系统管理员压缩后的错误信息文件的实际占用存储空间即将超过总存储空间;或者,可以根据配置文件中的配置项提醒管理员是否将最早压缩的日志文件删除。

[0065] 在上述的实现过程中,通过停止采集日志信息,或者,删除日志信息文件,或者,对

错误信息文件进行压缩,获得压缩后的错误信息文件;避免了无法将错误信息存储的问题,从而保证应用进程在发生异常时,能够有效地进行错误定位。

[0066] 请参见图3示出的本申请实施例提供的错误信息处理装置的结构示意图;本申请实施例提供了一种错误信息处理装置300,包括:

[0067] 日志信息采集模块310,用于采集多个应用进程中的每个应用进程产生的日志信息,并将每个应用进程产生的日志信息存储至每个应用进程对应的日志信息文件中。

[0068] 错误信息采集模块320,用于采集每个应用进程产生的错误信息,并将每个应用进程产生的错误信息存储至每个应用进程对应的错误信息文件中。

[0069] 出错位置定位模块330,用于若应用进程在运行过程中出现异常信息,则根据日志信息文件和错误信息文件定位出应用进程在对应的源代码中的出错位置。

[0070] 可选地,在本申请实施例中,错误信息处理装置,还包括:

[0071] 配置参数获取模块,用于从配置文件中获取总存储空间和单个文件阈值,总存储空间是分配给日志信息文件和错误信息文件所占用的总存储空间,单个文件阈值是错误信息文件所占用存储空间的最大阈值。

[0072] 个数阈值获得模块,用于将总存储空间除以单个文件阈值,获得文件个数阈值,文件个数阈值是存储错误信息文件的最小阈值。

[0073] 存储空间设置模块,用于若文件个数阈值小于预设个数阈值,且总存储空间小于物理存储空间,则将总存储空间设置为预设个数阈值与单个文件阈值的乘积。

[0074] 可选地,在本申请实施例中,个数阈值获得模块,包括:

[0075] 压缩开关判断模块,用于判断从配置文件中是否查找到压缩开关配置项。

[0076] 个数阈值计算模块,用于若从配置文件中查找到压缩开关配置项,则将单个文件阈值乘以预设压缩比,获得压缩文件阈值,并将总存储空间除以压缩文件阈值,获得文件个数阈值。

[0077] 可选地,在本申请实施例中,错误信息处理装置,还可以包括:

[0078] 若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第一预设比例,则停止采集每个应用进程产生的日志信息。

[0079] 可选地,在本申请实施例中,该错误信息处理装置,还包括:

[0080] 若日志信息文件和错误信息文件的占用存储空间超过总存储空间的第二预设比例,则删除每个应用进程产生的日志信息,第二预设比例大于第一预设比例。

[0081] 可选地,在本申请实施例中,该错误信息处理装置,还可以包括:

[0082] 若错误信息文件的占用存储空间超过总存储空间的第三预设比例,则对错误信息文件进行压缩,获得压缩后的错误信息文件,第三预设比例大于第二预设比例。

[0083] 可选地,在本申请实施例中,出错位置定位模块,包括:

[0084] 堆栈信息获取模块,用于获取异常信息在应用进程对应日志信息文件中的异常位置,以及异常信息在应用进程对应错误信息文件中的堆栈信息。

[0085] 代码错误定位模块,用于根据异常位置和堆栈信息对应用进程对应的源代码进行错误定位,获得应用进程在源代码中的错误位置。

[0086] 应理解的是,该装置与上述的错误信息处理方法实施例对应,能够执行上述方法实施例涉及的各个步骤,该装置具体的功能可以参见上文中的描述,为避免重复,此处适当

省略详细描述。该装置包括至少一个能以软件或固件(firmware)的形式存储于存储器中或固化在装置的操作系统(operating system, OS)中的软件功能模块。

[0087] 请参见图4示出的本申请实施例提供的电子设备的结构示意图。本申请实施例提供的一种电子设备400, 包括: 处理器410和存储器420, 存储器420存储有处理器410可执行的机器可读指令, 机器可读指令被处理器410执行时执行如上的方法。

[0088] 本申请实施例还提供了一种存储介质430, 该存储介质430上存储有计算机程序, 该计算机程序被处理器410运行时执行如上的方法。

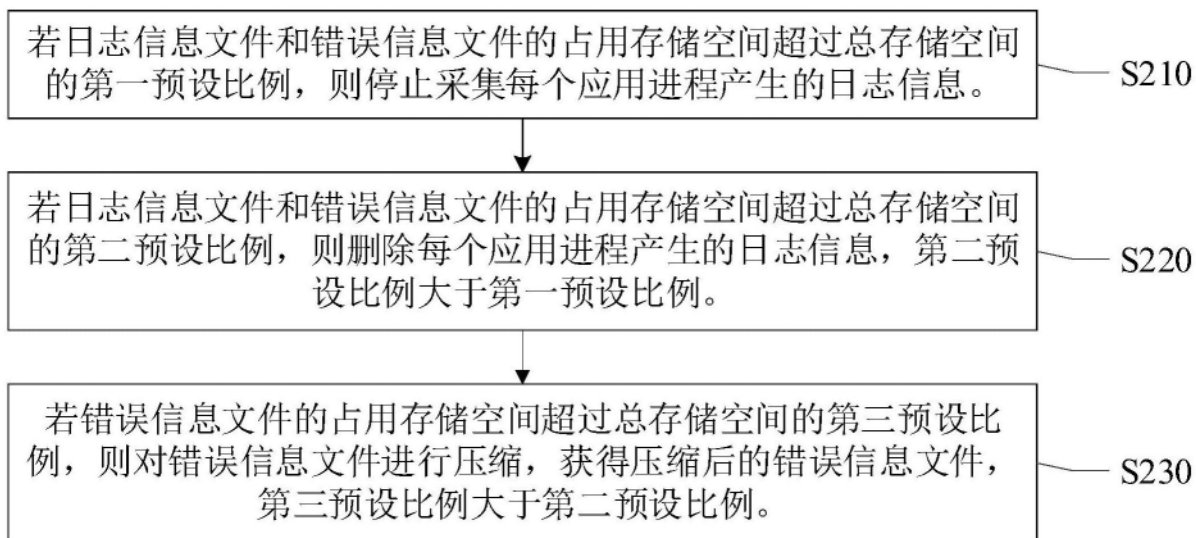
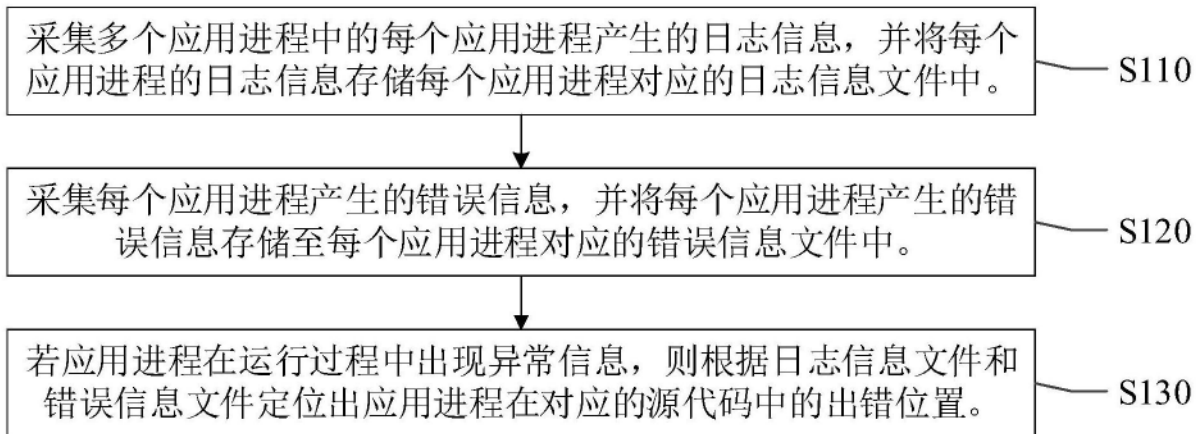
[0089] 其中, 存储介质430可以由任何类型的易失性或非易失性存储设备或者它们的组合实现, 如静态随机存取存储器(Static Random Access Memory, 简称SRAM), 电可擦除可编程只读存储器(Electrically Erasable Programmable Read-Only Memory, 简称EEPROM), 可擦除可编程只读存储器(Erasable Programmable Read Only Memory, 简称EPROM), 可编程只读存储器(Programmable Red-Only Memory, 简称PROM), 只读存储器(Read-Only Memory, 简称ROM), 磁存储器, 快闪存储器, 磁盘或光盘。

[0090] 本申请实施例提供的几个实施例中, 应该理解到, 所揭露的装置和方法, 也可以通过其他的方式实现。以上所描述的装置实施例仅是示意性的, 例如, 附图中的流程图和框图显示了根据本申请实施例的多个实施例的装置、方法和计算机程序产品的可能实现的体系架构、功能和操作。在这点上, 流程图或框图中的每个方框可以代表一个模块、程序段或代码的一部分, 模块、程序段或代码的一部分包含一个或多个用于实现规定的逻辑功能的可执行指令。也应当注意, 在有些作为替换的实现方式中, 方框中所标注的功能也可以和附图中所标注的发生顺序不同。例如, 两个连续的方框实际上可以基本并行地执行, 它们有时也可以按相反的顺序执行, 这主要根据所涉及的功能而定。也要注意的, 框图和/或流程图中的每个方框、以及框图和/或流程图中的方框的组合, 可以使用执行规定的功能或动作的专用的基于硬件的系统来实现, 或者可以用专用硬件与计算机指令的组合来实现。

[0091] 另外, 在本申请实施例中的各个实施例的各功能模块可以集成在一起形成一个独立的部分, 也可以是各个模块单独存在, 也可以两个或两个以上模块集成形成一个独立的部分。

[0092] 在本文中, 诸如第一和第二等之类的关系术语仅仅用来将一个实体或者操作与另一个实体或操作区分开来, 而不一定要求或者暗示这些实体或操作之间存在任何这种实际的关系或者顺序。

[0093] 以上的描述, 仅为本申请实施例的可选实施方式, 但本申请实施例的保护范围并不局限于此, 任何熟悉本技术领域的技术人员在本申请实施例揭露的技术范围内, 可轻易想到变化或替换, 都应涵盖在本申请实施例的保护范围之内。



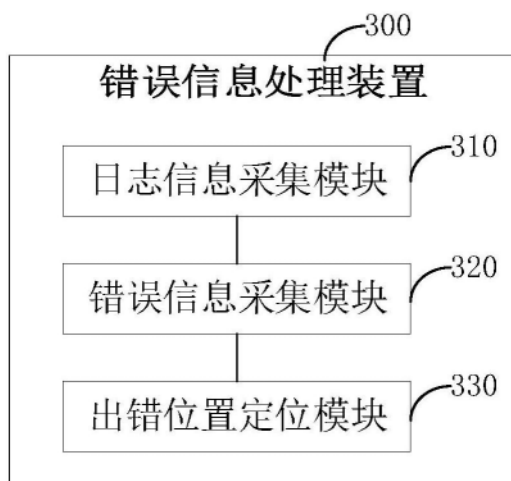


图3

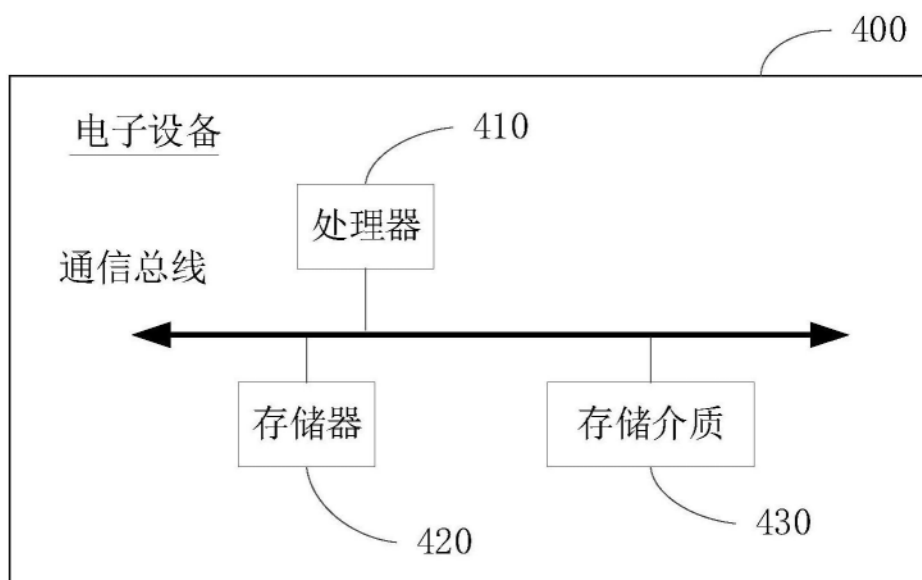


图4