(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 112334883 B
(45)授权公告日 2024.07.02

(21)申请号 201980034483.8
(22)申请日 2019.03.13
(65)同一申请的已公布的文献号
申请公布号 CN 112334883 A
(43)申请公布日 2021.02.05
(30)优先权数据
15/942,022 2018.03.30 US
(85)PCT国际申请进入国家阶段日
2020.11.23
(86)PCT国际申请的申请数据
PCT/US2019/022066 2019.03.13
(87)PCT国际申请的公布数据
WO2019/190750 EN 2019.10.03

(73)专利权人 起元技术有限责任公司
地址 美国马萨诸塞州
(72)发明人 J·L·威农 M·斯塔克尼斯
李欣
(74)专利代理机构 北京林达刘知识产权代理事务所(普通合伙) 11277
专利代理师 刘新宇

(51)Int.Cl.
G06F 11/36(2006.01)
G06F 11/32(2006.01)
(56)对比文件
US 2002120919 A1,2002.08.29
US 2011047415 A1,2011.02.24
审查员 陈亭玉

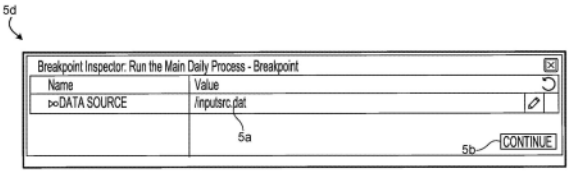
权利要求书4页 说明书25页 附图24页

(54)发明名称

数据处理系统、方法、硬件存储装置和计算机程序产品

(57)摘要

一种用于调试指定多个功能模块之间的控制流的可执行控制流图的计算机实现的方法,该控制流被表示为该多个功能模块之间的转换,该计算机实现的方法包括:指定该可执行控制流图中该可执行控制流图的执行将被中断的位置;其中,所指定的位置表示到给定功能模块的转换、到执行该给定功能模块的内容的状态的转换、或从该给定功能模块的转换;在执行环境中开始该可执行控制流图的执行;以及在表示该指定位置的执行点处,中断该可执行控制流图的执行;以及提供表示正在其中执行该给定功能模块的该执行环境的一个或多个属性的数据。



CN 112334883 B

1.一种由一个或多个计算机系统实现的方法,该方法包括:

识别控制流应用程序的功能模块,所述控制流应用程序指定该控制流应用程序的功能模块之间的控制流,所述控制流指定所述功能模块中的至少一些功能模块要执行的顺序,所识别的功能模块指定一个或多个受控应用程序,并且所识别的功能模块包括方法,该方法在被执行时、启动由所识别的功能模块指定的一个或多个受控应用程序;

将指定如下执行点的断点指示符插入到包括表示一个或多个功能模块的一个或多个图形程序元素的可视化程序中,在该执行点处,通过将该断点指示符与表示所识别的功能模块的图形元素并列地插入到该可视化程序中,来中断该控制流应用程序,

其中,该执行点表示到执行由该图形程序元素表示的所识别的功能模块的内容的状态的转换、从由该图形程序元素表示的所识别的功能模块的转换、或者到由该图形程序元素表示的所识别的功能模块的转换;以及

在执行环境中执行该控制流应用程序,其包括:

在所述断点指示符所指定的执行点处:

中断所述控制流应用程序的执行;以及

提供表示在其中正执行所识别的功能模块的所述执行环境的一个或多个属性的数据。

2.如权利要求1所述的由一个或多个计算机系统实现的方法,其中,该控制流应用程序被表示为多个层次级别,该控制流应用程序本身以这些层次级别中的第一层次级别来表示,并且每个功能模块以这些层次级别中的第二层次级别来表示,该第二层次级别不同于这些层次级别中包括该控制流应用程序的该第一层次级别。

3.如权利要求2所述的由一个或多个计算机系统实现的方法,其中,功能模块被配置用于执行方法,这些方法执行该功能模块的一个或多个动作,其中,这些被配置用于被执行的方法中的每种方法以这些层次级别中的第三层次级别来表示,该第三层次级别不同于这些层次级别中的该第一层次级别和这些层次级别中的该第二层次级别中的每一个。

4.如权利要求3所述的由一个或多个计算机系统实现的方法,其中,该执行点是第一执行点,其中,该控制流应用程序包括这些被配置用于被执行的方法之间的转换,并且其中,该方法进一步包括:

配置该控制流应用程序以在执行该功能模块的给定方法的内容或从该给定方法转换之前在表示到该给定方法的转换的第二执行点处中断该控制流应用程序的执行;

在该第二执行点处,

中断该控制流应用程序的执行并提供表示在其中执行该给定方法的运行时环境的一个或多个属性的数据。

5.如权利要求4所述的由一个或多个计算机系统实现的方法,其中,该给定方法的属性包括执行该给定方法的系统的本地环境。

6.如权利要求1所述的由一个或多个计算机系统实现的方法,其中,第一计算机系统执行该控制流应用程序,其中,第二计算机系统执行给定功能模块的衍生过程,其中,该第一计算机系统不同于该第二计算机系统,并且其中,该方法进一步包括:

响应于该中断,使显示呈现该第二计算机系统的运行时环境的一个或多个属性的一个或多个视觉表示的用户界面。

7.一种数据处理系统,包括:

一个或多个处理装置;以及

一个或多个机器可读的硬件存储装置,其存储指令,这些指令能够由该一个或多个处理装置执行以进行操作,所述操作包括:

识别控制流应用程序的功能模块,所述控制流应用程序指定该控制流应用程序的功能模块之间的控制流,所述控制流指定所述功能模块中的至少一些功能模块要执行的顺序,所识别的功能模块指定一个或多个受控应用程序,并且所识别的功能模块包括方法,该方法在被执行时、启动由所识别的功能模块指定的一个或多个受控应用程序;

将指定如下执行点的断点指示符插入到包括表示一个或多个功能模块的一个或多个图形程序元素的可视化程序中,在该执行点处,通过将该断点指示符与表示所识别的功能模块的图形元素并列地插入到该可视化程序中,来中断该控制流应用程序,

其中,该执行点表示到执行由该图形程序元素表示的所识别的功能模块的内容的状态的转换、从由该图形程序元素表示的所识别的功能模块的转换、或者到由该图形程序元素表示的所识别的功能模块的转换;以及

在执行环境中执行该控制流应用程序,其包括:

在所述断点指示符所指定的执行点处:

中断所述控制流应用程序的执行;以及

提供表示在其中正执行所识别的功能模块的所述执行环境的一个或多个属性的数据。

8.如权利要求7所述的数据处理系统,其中,该控制流应用程序被表示为多个层次级别,该控制流应用程序本身以这些层次级别中的第一层次级别来表示,并且每个功能模块以这些层次级别中的第二层次级别来表示,该第二层次级别不同于这些层次级别中包括该控制流应用程序的该第一层次级别。

9.如权利要求8所述的数据处理系统,其中,功能模块被配置用于执行方法,这些方法执行该功能模块的一个或多个动作,其中,这些被配置用于被执行的方法中的每种方法以这些层次级别中的第三层次级别来表示,该第三层次级别不同于这些层次级别中的该第一层次级别和这些层次级别中的该第二层次级别中的每一个。

10.如权利要求9所述的数据处理系统,其中,该执行点是第一执行点,其中,该控制流应用程序包括这些被配置用于被执行的方法之间的转换,并且其中,一个或多个操作还包括:

配置该控制流应用程序以在执行该功能模块的给定方法的内容或从该给定方法转换之前在表示到该给定方法的转换的第二执行点处中断该控制流应用程序的执行;

在该第二执行点处,

中断该控制流应用程序的执行并提供表示在其中执行该给定方法的运行时环境的一个或多个属性的数据。

11.如权利要求10所述的数据处理系统,其中,该给定方法的属性包括执行该给定方法的系统的本地环境。

12.如权利要求7所述的数据处理系统,其中,第一计算机系统执行该控制流应用程序,其中,第二计算机系统执行给定功能模块的衍生过程,其中,该第一计算机系统不同于该第二计算机系统,并且其中,所述操作还包括:

响应于该中断,使显示呈现该第二计算机系统的运行时环境的一个或多个属性的一个

或多个视觉表示的用户界面。

13.一个或多个机器可读的硬件存储装置,其存储指令,这些指令能够由一个或多个处理装置执行以进行操作,所述操作包括:

识别控制流应用程序的功能模块,所述控制流应用程序指定该控制流应用程序的功能模块之间的控制流,所述控制流指定所述功能模块中的至少一些功能模块要执行的顺序,所识别的功能模块指定一个或多个受控应用程序,并且所识别的功能模块包括方法,该方法在被执行时、启动由所识别的功能模块指定的一个或多个受控应用程序;

将指定如下执行点的断点指示符插入到包括表示一个或多个功能模块的一个或多个图形程序元素的可视化程序中,在该执行点处,通过将该断点指示符与表示所识别的功能模块的图形元素并列地插入到该可视化程序中,来中断该控制流应用程序,

其中,该执行点表示到执行由该图形程序元素表示的所识别的功能模块的内容的状态的转换、从由该图形程序元素表示的所识别的功能模块的转换、或者到由该图形程序元素表示的所识别的功能模块的转换;以及

在执行环境中执行该控制流应用程序,其包括:

在所述断点指示符所指定的执行点处:

中断所述控制流应用程序的执行;以及

提供表示在其中正执行所识别的功能模块的所述执行环境的一个或多个属性的数据。

14.如权利要求13所述的一个或多个机器可读的硬件存储装置,其中,该控制流应用程序被表示为多个层次级别,该控制流应用程序本身以这些层次级别中的第一层次级别来表示,并且每个功能模块以这些层次级别中的第二层次级别来表示,该第二层次级别不同于这些层次级别中包括该控制流应用程序的该第一层次级别。

15.如权利要求14所述的一个或多个机器可读的硬件存储装置,其中,功能模块被配置用于执行方法,这些方法执行该功能模块的一个或多个动作,其中,这些被配置用于被执行的方法中的每种方法以这些层次级别中的第三层次级别来表示,该第三层次级别不同于这些层次级别中的该第一层次级别和这些层次级别中的该第二层次级别中的每一个。

16.如权利要求15所述的一个或多个机器可读的硬件存储装置,其中,该执行点是第一执行点,其中,该控制流应用程序包括这些被配置用于被执行的方法之间的转换,并且其中,一个或多个操作还包括:

配置该控制流应用程序以在执行该功能模块的给定方法的内容或从该给定方法转换之前在表示到该给定方法的转换的第二执行点处中断该控制流应用程序的执行;

在该第二执行点处,

中断该控制流应用程序的执行并提供表示在其中执行该给定方法的运行时环境的一个或多个属性的数据。

17.如权利要求16所述的一个或多个机器可读的硬件存储装置,其中,该给定方法的属性包括执行该给定方法的系统的本地环境。

18.如权利要求17所述的一个或多个机器可读的硬件存储装置,其中,第一计算机系统执行该控制流应用程序,其中,第二计算机系统执行给定功能模块的衍生过程,其中,该第一计算机系统不同于该第二计算机系统,并且其中,所述操作还包括:

响应于该中断,使显示呈现该第二计算机系统的运行时环境的一个或多个属性的一个

或多个视觉表示的用户界面。

19.一种计算机程序产品,其上存储有计算机指令,所述计算机指令在被处理器执行时实现权利要求1至6中任意一项所述的方法。

数据处理系统、方法、硬件存储装置和计算机程序产品

技术领域

[0001] 本描述涉及用于调试指定多个功能模块之间的控制流的可执行控制流图的方法、系统和计算机可读介质。

背景技术

[0002] 调试器或调试工具是用于测试和调试其他计算机程序的计算机程序。

发明内容

[0003] 在总体方面1中,本文描述了一种用于调试指定多个功能模块之间的控制流的可执行控制流图的计算机实现的方法,该控制流被表示为该多个功能模块之间的转换,每个功能模块被配置用于在计算机系统上执行一个或多个动作,该计算机实现的方法包括:指定该可执行控制流图中该可执行控制流图的执行将被中断的位置;其中,所指定的位置表示到给定功能模块的转换、到执行该给定功能模块的内容的状态的转换、或从该给定功能模块的转换;在执行环境中开始该可执行控制流图的执行;以及在表示该指定位置的执行点处,中断该可执行控制流图的执行;以及提供表示正在其中执行该给定功能模块的该执行环境的一个或多个属性的数据。

[0004] 在根据方面1的方面2中,该可执行控制流图被表示为多个层次级别,该可执行控制流图本身以这些层次级别中的第一层次级别来表示,并且每个功能模块以这些层次级别中的第二层次级别来表示,该第二层次级别不同于这些层次级别中包括该可执行控制流图的该第一层次级别。

[0005] 在根据方面1至2中任一项的方面3中,功能模块被配置用于执行方法,这些方法执行该功能模块的该一个或多个动作,其中,这些被配置用于被执行的方法中的每种方法以这些层次级别中的第三层次级别来表示,该第三层次级别不同于这些层次级别中的该第一层次级别和这些层次级别中的该第二层次级别中的每一个。

[0006] 在根据方面1至3中任一项的方面4中,该指定位置是第一指定位置,其中,该可执行控制流图包括这些被配置用于被执行的方法之间的转换,并且该方法进一步包括:配置该可执行控制流图以在执行该功能模块的给定方法的内容或从该给定方法转换之前在表示到该给定方法的转换的第二指定位置处中断该可执行控制流图的执行;在表示该第二指定位置的执行点处,中断该可执行控制流图的执行并提供表示在其中执行该给定方法的运行时环境的一个或多个属性的数据。

[0007] 在根据方面1至4中任一项的方面5中,该给定方法的属性包括执行该给定方法的系统的本地环境。

[0008] 在根据方面1至5中任一项的方面6中,第一计算机系统执行该可执行控制流图,其中,第二计算机系统执行该给定功能模块的衍生过程,其中,该第一计算机系统不同于该第二计算机系统,并且其中,该方法进一步包括:响应于该中断,使显示呈现该第二计算机系统的运行时环境的一个或多个属性的一个或多个视觉表示的用户界面。

[0009] 在根据方面1至6中任一项的方面7中,该方法进一步包括提供图形用户界面,该图形用户界面当在显示设备上被呈现时显示用于配置该可执行控制流图的中断的一个或多个控件。

[0010] 在根据方面1至7中任一项的方面8中,指定该位置包括在该指定位置处将断点插入到该可执行控制流图中。

[0011] 在根据方面1至8中任一项的方面9中,该给定功能模块是第一功能模块,并且该方法进一步包括:当该可执行控制流图被中断时,修改该第一功能模块的一个或多个属性的一个或多个值,其中,这一个或多个属性的该一个或多个值由一个或多个第二功能模块从该第一功能模块访问;恢复该可执行控制流图的执行;以及通过使该一个或多个第二功能模块以该一个或多个属性的该一个或多个修改值来执行从而修改该可执行控制流图的行为。

[0012] 在根据方面1至9中任一项的方面10中,属性是参数。

[0013] 在根据方面1至10中任一项的方面11中,该给定功能模块是被配置用于在满足条件时执行一个或多个第一动作并且在不满足该条件时执行一个或多个第二动作的条件功能模块,该方法进一步包括:通过改变该条件功能模块的成功状态或失败状态来修改该可执行控制流图的控制流。

[0014] 在根据方面1至11中任一项的方面12中,该可执行控制流图被配置用于循环,并且该方法进一步包括:配置该可执行控制流图,使得循环的每次迭代或循环的特定迭代在该可执行控制流图的执行之前、之中或之后被中断。

[0015] 在根据方面1至12中任一项的方面13中,该可执行控制流图包括被配置用于循环的可执行控制流子图,并且其中,该方法进一步包括:配置该可执行控制流子图,使得循环的每次迭代或循环的特定迭代被中断。

[0016] 在根据方面1至13中任一项的方面14中,该方法进一步包括配置该可执行控制流图,使得当满足特定条件时该可执行控制流图的执行在该指定位置处被中断。

[0017] 在根据方面1至14中任一项的方面15中,该方法进一步包括配置该可执行控制流图,以在任一功能模块出现故障时中断该可执行控制流图的执行。

[0018] 在根据方面1至15中任一项的方面16中,该给定功能模块是被配置用于启动第二功能模块的第一功能模块,并且该方法进一步包括:配置该可执行控制流图以在该第二功能模块的执行开始时暂停执行。

[0019] 在根据方面1至16中任一项的方面17中,该可执行控制流图包括图形层和控制处理层,该图形层包括用于图形用户界面的视觉效果和用这些视觉效果编程的可视化编程语言的可视化程序,该可视化编程语言用于指定控制流,其中,该控制处理层包括功能模块、方法、在执行这些方法时使用的基础数据结构、以及使用这些基础数据结构执行这些方法以实现由该可视化编程语言指定的控制流的代码。

[0020] 在根据方面1至17中任一项的方面18中,该方法进一步包括通过用这些视觉效果呈现该图形用户界面来执行该图形层;通过该图形用户界面接收表示用户对在该图形用户界面中显示的这些视觉效果中的一个或多个视觉效果的操纵的数据,这些所接收的数据以该可视化编程语言表示该可视化程序;以及根据在该图形层中创建的该可视化程序来执行该控制处理层,该控制处理层的执行包括执行由该可视化程序指定的一个或多个功能模

块,功能模块的执行使得通过使用该功能模块的方法的一个或多个数据结构执行代码来执行该方法从而实现由该可视化编程语言指定的控制流。

[0021] 在根据方面1至18中任一项的方面19中,特定功能模块表示包括在数据处理层中的受控应用程序,该方法进一步包括:执行该特定功能模块的特定方法;以及响应于执行该特定方法,启动该受控应用程序。

[0022] 在总体方面20中,一种用于调试指定多个功能模块之间的控制流的可执行控制流图的计算机实现的方法,该多个功能模块由图形程序元素表示并且该控制流由这些图形程序元素之间的转换表示,每个功能模块被配置用于在计算机系统上执行一个或多个动作,该计算机实现的方法包括:指定该可执行控制流图中该可执行控制流图的执行将被中断的位置;其中,所指定的位置表示到该多个功能模块中的给定功能模块的转换、到该给定功能模块的该一个或多个动作中的一些动作开始执行的状态的转换、或从该给定功能模块的转换;在执行环境中开始该可执行控制流图的执行,其中,该执行环境托管在一个或多个计算机上并且被配置用于执行该多个功能模块;以及在该执行的表示该指定位置的时间点处,中断该可执行控制流图的执行;以及提供表示在该时间点处盛行的该执行环境的一个或多个属性的数据。

[0023] 在总体方面21中,本文描述了存储用于调试指定多个功能模块之间的控制流的可执行控制流图的一条或多条指令的一个或多个机器可读硬件存储设备,该控制流被表示为该多个功能模块之间的转换,每个功能模块被配置用于在计算机系统上执行一个或多个动作,该一条或多条指令可由一个或多个处理设备执行以执行包括以下各项的操作:指定该可执行控制流图中该可执行控制流图的执行将被中断的位置;其中,所指定的位置表示到给定功能模块的转换、到执行该给定功能模块的内容的状态的转换、或从该给定功能模块的转换;在执行环境中开始该可执行控制流图的执行;以及在表示该指定位置的执行点处,中断该可执行控制流图的执行;以及提供表示正在其中执行该给定功能模块的该执行环境的一个或多个属性的数据。

[0024] 在根据方面21的方面22中,该可执行控制流图被表示为多个层次级别,该可执行控制流图本身以这些层次级别中的第一层次级别来表示,并且每个功能模块以这些层次级别中的第二层次级别来表示,该第二层次级别不同于这些层次级别中包括该可执行控制流图的该第一层次级别。

[0025] 在根据方面21至22中任一项的方面23中,功能模块被配置用于执行方法,这些方法执行该功能模块的该一个或多个动作,其中,这些被配置用于被执行的方法中的每种方法以这些层次级别中的第三层次级别来表示,该第三层次级别不同于这些层次级别中的该第一层次级别和这些层次级别中的该第二层次级别中的每一个。

[0026] 在根据方面21至23中任一项的方面24中,该指定位置是第一指定位置,其中,该可执行控制流图包括这些被配置用于被执行的方法之间的转换,并且该一个或多个操作进一步包括:配置该可执行控制流图以在执行该功能模块的给定方法的内容或从该给定方法转换之前在表示到该给定方法的转换的第二指定位置处中断该可执行控制流图的执行;在表示该第二指定位置的执行点处,中断该可执行控制流图的执行并提供表示在其中执行该给定方法的运行时环境的一个或多个属性的数据。

[0027] 在根据方面21至24中任一项的方面25中,该给定方法的属性包括执行该给定方法

的系统的本地环境。

[0028] 在根据方面21至25中任一项的方面26中,第一计算机系统执行该可执行控制流图,其中,第二计算机系统执行该给定功能模块的衍生过程,其中,该第一计算机系统不同于该第二计算机系统,并且该一个或多个操作进一步包括:响应于该中断,使显示呈现该第二计算机系统的运行时环境的一个或多个属性的一个或多个视觉表示的用户界面。

[0029] 在根据方面21至26中任一项的方面27中,该一个或多个操作进一步包括提供图形用户界面,该图形用户界面当在显示设备上被呈现时显示用于配置该可执行控制流图的中断的一个或多个控件。

[0030] 在根据方面21至27中任一项的方面28中,指定该位置包括在该指定位置处将断点插入到该可执行控制流图中。

[0031] 在根据方面21至28中任一项的方面29中,该给定功能模块是第一功能模块,并且该一个或多个操作进一步包括:当该可执行控制流图被中断时,修改该第一功能模块的一个或多个属性的一个或多个值,其中,这一个或多个属性的该一个或多个值由一个或多个第二功能模块从该第一功能模块访问;恢复该可执行控制流图的执行;以及通过使该一个或多个第二功能模块以该一个或多个属性的该一个或多个修改值来执行从而修改该可执行控制流图的行为。

[0032] 在根据方面21至29中任一项的方面30中,属性是参数。

[0033] 在根据方面21至30中任一项的方面31中,该给定功能模块是被配置用于在满足条件时执行一个或多个第一动作并且在不满足该条件时执行一个或多个第二动作的条件功能模块,该一个或多个操作进一步包括:通过改变该条件功能模块的成功状态或失败状态来修改该可执行控制流图的控制流。

[0034] 在根据方面21至31中任一项的方面32中,该可执行控制流图被配置用于循环,并且该一个或多个操作进一步包括:配置该可执行控制流图,使得循环的每次迭代或循环的特定迭代在该可执行控制流图的执行之前、之中或之后被中断。

[0035] 在根据方面21至32中任一项的方面33中,该可执行控制流图包括被配置用于循环的可执行控制流子图,并且其中,该一个或多个操作进一步包括:配置该可执行控制流子图,使得循环的每次迭代或循环的特定迭代被中断。

[0036] 在根据方面21至33中任一项的方面34中,该一个或多个操作进一步包括配置该可执行控制流图,使得当满足特定条件时该可执行控制流图的执行在该指定位置处被中断。

[0037] 在根据方面21至34中任一项的方面35中,该一个或多个操作进一步包括配置该可执行控制流图,以在任一功能模块出现故障时中断该可执行控制流图的执行。

[0038] 在根据方面21至35中任一项的方面36中,该给定功能模块是被配置用于启动第二功能模块的第一功能模块,并且该一个或多个操作进一步包括:配置该可执行控制流图以在该第二功能模块的执行开始时暂停执行。

[0039] 在根据方面21至36中任一项的方面37中,该可执行控制流图包括图形层和控制处理层,该图形层包括用于图形用户界面的视觉效果和用这些视觉效果编程的可视化编程语言的可视化程序,该可视化编程语言用于指定控制流,其中,该控制处理层包括功能模块、方法、在执行这些方法时使用的基础数据结构、以及使用这些基础数据结构执行这些方法以实现由该可视化编程语言指定的控制流的代码。

[0040] 在根据方面21至37中任一项的方面38中,该一个或多个操作进一步包括通过用这些视觉效果呈现该图形用户界面来执行该图形层;通过该图形用户界面接收表示用户对在该图形用户界面中显示的这些视觉效果中的一个或多个视觉效果的操纵的数据,这些所接收的数据以该可视化编程语言表示该可视化程序;以及根据在该图形层中创建的该可视化程序来执行该控制处理层,该控制处理层的执行包括执行由该可视化程序指定的一个或多个功能模块,功能模块的执行使得通过使用该功能模块的方法的一个或多个数据结构执行代码来执行该方法从而实现由该可视化编程语言指定的控制流。

[0041] 在根据方面21至38中任一项的方面39中,特定功能模块表示包括在数据处理层中的受控应用程序,该一个或多个操作进一步包括:执行该特定功能模块的特定方法;以及响应于执行该特定方法,启动该受控应用程序。

[0042] 在总体方面40中,本文描述了一种存储用于调试指定多个功能模块之间的控制流的可执行控制流图的一条或多条指令的一个或多个机器可读硬件存储设备,该多个功能模块由图形程序元素表示并且控制流由这些图形程序元素之间的转换表示,每个功能模块被配置用于在计算机系统上执行一个或多个动作,该一条或多条指令可由一个或多个处理设备执行以执行包括以下各项的操作:指定该可执行控制流图中该可执行控制流图的执行将被中断的位置;其中,所指定的位置表示到该多个功能模块中的给定功能模块的转换、到该给定功能模块的该一个或多个动作中的一些动作开始执行的状态的转换、或从该给定功能模块的转换;在执行环境中开始该可执行控制流图的执行,其中,该执行环境托管在一个或多个计算机上并且被配置用于执行该多个功能模块;以及在该执行的表示该指定位置的时间点处,中断该可执行控制流图的执行;以及提供表示在该时间点处盛行的该执行环境的一个或多个属性的数据。

[0043] 在总体方面41中,本文描述了一种用于调试指定多个功能模块之间的控制流的可执行控制流图的数据处理系统,该控制流被表示为该多个功能模块之间的转换,每个功能模块被配置用于在计算机系统上执行一个或多个动作,该数据处理系统包括一个或多个处理设备和存储一条或多条指令的一个或多个机器可读硬件存储设备,该一条或多条指令可由该一个或多个处理设备执行以执行包括以下各项的操作:指定该可执行控制流图中该可执行控制流图的执行将被中断的位置;其中,所指定的位置表示到给定功能模块的转换、到执行该给定功能模块的内容的状态的转换、或从该给定功能模块的转换;在执行环境中开始该可执行控制流图的执行;以及在表示该指定位置的执行点处,中断该可执行控制流图的执行;以及提供表示正在其中执行该给定功能模块的该执行环境的一个或多个属性的数据。

[0044] 在根据方面41的方面42中,该可执行控制流图被表示为多个层次级别,该可执行控制流图本身以这些层次级别中的第一层次级别来表示,并且每个功能模块以这些层次级别中的第二层次级别来表示,该第二层次级别不同于这些层次级别中包括该可执行控制流图的该第一层次级别。

[0045] 在根据方面41至42中任一项的方面43中,功能模块被配置用于执行方法,这些方法执行该功能模块的该一个或多个动作,其中,这些被配置用于被执行的方法中的每种方法以这些层次级别中的第三层次级别来表示,该第三层次级别不同于这些层次级别中的该第一层次级别和这些层次级别中的该第二层次级别中的每一个。

[0046] 在根据方面41至43中任一项的方面44中,该指定位置是第一指定位置,其中,该可执行控制流图包括这些被配置用于被执行的方法之间的转换,并且该一个或多个操作进一步包括:配置该可执行控制流图以在执行该功能模块的给定方法的内容或从该给定方法转换之前在表示到该给定方法的转换的第二指定位置处中断该可执行控制流图的执行;在表示该第二指定位置的执行点处,中断该可执行控制流图的执行并提供表示在其中执行该给定方法的运行时环境的一个或多个属性的数据。

[0047] 在根据方面41至44中任一项的方面45中,该给定方法的属性包括执行该给定方法的系统的本地环境。

[0048] 在根据方面41至45中任一项的方面46中,第一计算机系统执行该可执行控制流图,其中,第二计算机系统执行该给定功能模块的衍生过程,其中,该第一计算机系统不同于该第二计算机系统,并且该一个或多个操作进一步包括:响应于该中断,使显示呈现该第二计算机系统的运行时环境的一个或多个属性的一个或多个视觉表示的用户界面。

[0049] 在根据方面41至46中任一项的方面47中,该一个或多个操作进一步包括提供图形用户界面,该图形用户界面当在显示设备上被呈现时显示用于配置该可执行控制流图的中断的一个或多个控件。

[0050] 在根据方面41至47中任一项的方面48中,指定该位置包括在该指定位置处将断点插入到该可执行控制流图中。

[0051] 在根据方面41至48中任一项的方面49中,该给定功能模块是第一功能模块,并且该一个或多个操作进一步包括:当该可执行控制流图被中断时,修改该第一功能模块的一个或多个属性的一个或多个值,其中,这一个或多个属性的该一个或多个值由一个或多个第二功能模块从该第一功能模块访问;恢复该可执行控制流图的执行;以及通过使该一个或多个第二功能模块以该一个或多个属性的该一个或多个修改值来执行从而修改该可执行控制流图的行为。

[0052] 在根据方面41至49中任一项的方面50中,属性是参数。

[0053] 在根据方面41至50中任一项的方面51中,该给定功能模块是被配置用于在满足条件时执行一个或多个第一动作并且在不满足该条件时执行一个或多个第二动作的条件功能模块,该一个或多个操作进一步包括:通过改变该条件功能模块的成功状态或失败状态来修改该可执行控制流图的控制流。

[0054] 在根据方面41至51中任一项的方面52中,该可执行控制流图被配置用于循环,并且该一个或多个操作进一步包括:配置该可执行控制流图,使得循环的每次迭代或循环的特定迭代在该可执行控制流图的执行之前、之中或之后被中断。

[0055] 在根据方面41至52中任一项的方面53中,该可执行控制流图包括被配置用于循环的可执行控制流子图,并且其中,该一个或多个操作进一步包括:配置该可执行控制流子图,使得循环的每次迭代或循环的特定迭代被中断。

[0056] 在根据方面41至53中任一项的方面54中,该一个或多个操作进一步包括配置该可执行控制流图,使得当满足特定条件时该可执行控制流图的执行在该指定位置处被中断。

[0057] 在根据方面41至54中任一项的方面55中,该一个或多个操作进一步包括配置该可执行控制流图,以在任一功能模块出现故障时中断该可执行控制流图的执行。

[0058] 在根据方面41至55中任一项的方面56中,该给定功能模块是被配置用于启动第二

功能模块的第一功能模块,并且该一个或多个操作进一步包括:配置该可执行控制流图以在该第二功能模块的执行开始时暂停执行。

[0059] 在根据方面41至56中任一项的方面57中,该可执行控制流图包括图形层和控制处理层,该图形层包括用于图形用户界面的视觉效果和用这些视觉效果编程的可视化编程语言的可视化程序,该可视化编程语言用于指定控制流,其中,该控制处理层包括功能模块、方法、在执行这些方法时使用的基础数据结构、以及使用这些基础数据结构执行这些方法以实现由该可视化编程语言指定的控制流的代码。

[0060] 在根据方面41至57中任一项的方面58中,该一个或多个操作进一步包括通过用这些视觉效果呈现该图形用户界面来执行该图形层;通过该图形用户界面接收表示用户对在该图形用户界面中显示的这些视觉效果中的一个或多个视觉效果的操纵的数据,这些所接收的数据以该可视化编程语言表示该可视化程序;以及根据在该图形层中创建的该可视化程序来执行该控制处理层,该控制处理层的执行包括执行由该可视化程序指定的一个或多个功能模块,功能模块的执行使得通过使用该功能模块的方法的一个或多个数据结构执行代码来执行该方法从而实现由该可视化编程语言指定的控制流。

[0061] 在根据方面41至58中任一项的方面59中,特定功能模块表示包括在数据处理层中的受控应用程序,该一个或多个操作进一步包括:执行该特定功能模块的特定方法;以及响应于执行该特定方法,启动该受控应用程序。

[0062] 在总体方面60中,本文描述了一种用于调试指定多个功能模块之间的控制流的可执行控制流图的数据处理系统,该多个功能模块由图形程序元素表示并且控制流由这些图形程序元素之间的转换表示,每个功能模块被配置用于在计算机系统上执行一个或多个动作,该数据处理系统包括一个或多个处理设备和存储一条或多条指令的一个或多个机器可读硬件存储设备,该一条或多条指令可由该一个或多个处理设备执行以执行包括以下各项的操作:指定该可执行控制流图中该可执行控制流图的执行将被中断的位置;其中,所指定的位置表示到该多个功能模块中的给定功能模块的转换、到该给定功能模块的该一个或多个动作中的一些动作开始执行的状态的转换、或从该给定功能模块的转换;在执行环境中开始该可执行控制流图的执行,其中,该执行环境托管在一个或多个计算机上并且被配置用于执行该多个功能模块;以及在该执行的表示该指定位置的时间点处,中断该可执行控制流图的执行;以及提供表示在该时间点处盛行的该执行环境的一个或多个属性的数据。

[0063] 通过以下描述、以及通过权利要求书,本发明的其他特征和优点将变得清楚。

附图说明

[0064] 图1A至图1G、图2和图11各自是计划的图示。

[0065] 图3是任务、其方法以及相对于任务及其方法的断点插入的图示。

[0066] 图4、图7A和图7B各自是在图形开发环境中定义的计划的各个方面的用户界面的示例。

[0067] 图5、图6、图9B至图9C各自是计划调试器中的控件的示例。

[0068] 图8和图9A各自是用于查看和编辑计划的元素的属性的断点检查器的图示。

[0069] 图10是计划的执行环境的图示。

[0070] 图12是任务的图示。

[0071] 图13是用于中断图形控制流应用程序的执行的示例过程。

[0072] 图14A是继承的方法的图示。

[0073] 图14B是在继承的方法上设置断点的图示。

[0074] 图15是数据处理系统的框图。

具体实施方式

[0075] 参考图1A,图示2展示了计划的图形部分1。为了方便而非限制,整个计划(包括图形部分1)在下文中被称为计划1。通常,如本文使用的“计划”(本文也称为“可执行控制流图”或“图形控制流应用程序”)包括具有图形程序元素的可视化程序,每个图形程序元素表示功能模块。每个功能模块又表示受控应用程序。

[0076] 计划指定功能模块之间的控制流。通过指定控制流,计划提供对执行计划的各个功能模块的可视化组织。通常,由计划指定的控制流由计划中的图形程序元素之间的转换或连接来表示。计划可以被运行或执行并通过针对要执行的各种受控应用程序的功能模块来封装信息。

[0077] 通常,受控应用程序包括处于计划外部并由计划运行的程序。受控应用程序的示例包括计算机程序、脚本、数据流图、另一个计划等。通常,图形程序元素包括可视化编程语言(VPL)的可视化程序的元素,可以以图形方式而不是以文本方式来操纵该元素。通常,功能模块包括计划的数据结构、计划的方法、以及用于使用这些数据结构实施这些方法的代码。在示例中,一种类型的功能模块是包括要执行的一系列动作(在计划内,而不是受控应用程序内)的任务。为了方便而非限制,功能模块在本文中也可以被称为“任务”。在这个示例中,控制流表示关于计划的功能模块的执行顺序的逻辑。通常,功能模块表示条件逻辑、程序、用户脚本、过程等。功能模块可以使用数据流图来实施或者可以表示另一个可执行控制流图(称为“子计划”或可执行控制流子图)。例如,子计划是嵌入在较大计划中的计划并且为了组织的方便而在计划中使用并且可以包括各种功能,例如,循环功能。子计划可以表示许多不同的任务、以及进一步嵌套的子计划。计划通过任务如何被连接以及如何指定方法和参数来控制任务之间的顺序、关系和通信。在计划中,任务、运行顺序、条件逻辑以及任务之间的关系以图形方式表示。

[0078] 在这个示例中,计划包括图形层(先前被称为图形部分)和控制处理层,图形层和控制处理层中的每一个都不同于数据处理层(例如,受控应用程序)。图形层包括图形用户界面中的视觉效果和VPL的可视化程序。控制处理层包括任务、方法、(在执行方法时使用的)基础数据结构、以及使用这些基础数据结构执行这些方法以实现由可视化编程语言指定的控制流的代码。

[0079] 计划、子计划和图形程序元素包括作为可视化程序的一部分的图形实体,并且计划的任务和其他功能模块由图形实体表示并由被称为“可执行控制流图”的可视化程序控制。也就是说,可执行控制流图不指定图形图标之间的控制流,而是控制实际数据处理的执行顺序。

[0080] 在示例中,可执行控制流图(或计划)包括具有表示功能模块(例如,任务、子计划等)的节点的有向图。这些节点通过表示功能模块之间的控制流的有向链路(本文中也被称为“转换”)连接。具体地,每个有向链路由指定功能模块之间的控制流的数据结构来指定。

在这个示例中,计划本身的执行致使根据由计划指定的控制流来执行由计划中的节点表示的基础控制处理层。计划可以例如以分布式方式在一个或多个计算节点(例如,中央处理单元(CPU))上运行,使得每个图形程序元素在不同的一个计算节点上运行,其中,不同的计算节点通过网络连接通信地互连。

[0081] 在一些情况下,可执行控制流图的构造本质上可以是高度技术性的。可执行控制流图可以在各种子计划和任务之间实施基于图的控制流。可执行控制流图由数据存储装置中的数据结构指定,可执行控制流图具有由数据结构指定的多个节点并表示由一个或多个链路连接的子计划和/或任务,这些链路由数据结构指定并表示节点之间的数据流。可以使用耦接到数据存储装置并托管在一个或多个计算机上的运行时环境(或执行环境),该运行时环境包括预执行模块,该预执行模块被配置用于读取指定可执行控制流图的已存储数据结构并用于分配和配置计算资源,比如用于执行对由该预执行模块分派给可执行控制流图的任务和子计划的计算的过程,该运行时环境包括执行模块,该执行模块用于调度和控制对所分派过程的执行,以使得能够执行本文描述的方法中的任一种。

[0082] 存在不同类型的任务,包括例如程序任务(例如,表示指定程序的任务,该任务是不同于运行计划的过程的过程,和/或使得能够自动将可执行代码并入到计划中)、计划任务(例如,表示子计划的任务,该任务是不同于运行计划的过程的过程(用于执行另一计划))、图任务(例如,表示数据流图的任务,该任务也是不同于运行计划的过程的过程,和/或被配置用于从计划内运行单独的可执行数据流图)、以及在计划中实施条件逻辑的条件任务。在这个示例中,每个任务表示要在计算机系统上执行的一系列有序的动作。

[0083] 任务的主要动作被称为其执行方法。例如,可以实施执行方法来执行任务的主要功能。除了执行方法之外,每个任务还可以可选地在特定时间或响应于特定条件而执行其他动作,如下面进一步详细描述的。执行方法赋予任务其标识。任务的其余部分支持执行方法。各种可选动作为执行方法的执行做准备或设置执行方法的执行,报告其结果,并在其之后进行清理。在一些示例中,计划包括存储表示各种任务和子计划的逻辑以及这些各种任务和子计划之间的关系的基础数据结构。

[0084] 在这个示例中,计划的执行致使执行或运行基础数据结构,例如,通过为计划中的各种任务和方法实例化适当的对象。例如,系统将计划的内容存储在包括计划的层次表示的文件中。层次表示包括节点,每个节点与计划的元素(例如,任务、方法或子计划)相对应。系统通过遍历层次表示中的节点来实例化计划。对于每个节点,系统为由该节点表示的元素实例化对象。例如,当节点表示任务时,系统为该节点实例化任务对象。另外,当计划指定在该任务上运行方法时,系统为该方法实例化方法对象。

[0085] 在图1A的示例中,计划1是用于处理数据记录以例如基于对这些数据记录的处理来确定哪些动作(例如,运行日内过程或计算日终头寸)适合于执行的计划。在这个示例中,系统在金融服务设置中执行计划1作为用于确定日内交易价格或当天的平仓交易的过程的一部分,例如,通过确定是运行特定过程(即,日内过程)还是执行确定日终头寸的特定计算。计划1包括图形程序元素1a、1b、1c、1d、1e、1f、1g、1h和转换1i、1j、1k、1l、1m、1o。图形程序元素1a、1b、1c、1d、1e、1f、1g、1h表示如上所述的相应任务,并且转换1i、1j、1k、1l、1m、1o表示如上所述的由计划1指定的控制流。为了方便读者,每当参考指示图形程序元素或转换的附图标记时,在描述相应的附图时分别直接参考相应的任务和控制流。因此,计划1包括

程序任务1a和1h、图任务1b和1d、条件任务1c、计划任务1e和1f、以及子计划1g。为了方便起见，程序任务1a和1h、图任务1b和1d、条件任务1c、以及计划任务1e和1f中的每一个在本文中都可以被称为“任务”。

[0086] 在这个示例中，任务1b只能在任务1a已经完成之后执行。任务1f只能在任务1e已经完成之后执行。计划1还显示随后评估条件任务1c中指定的条件(“是否应该运行日内过程?”)，并且如果发现为“否”，则系统通过迭代子计划1g来计算每个办公室的日终头寸。如果“是”，则系统执行任务1d以执行日内运行。在这个示例中，各种任务也可以在其他任务失败时被触发——这在计划1中由“错误动作”任务(即任务1h)来说明，如果任务2(即任务1f)由于任何原因失败，则运行该任务。

[0087] 在这个示例中，程序任务1a、1h中的每一个指向(或以其他方式引用)外部程序，或者指向外部脚本或嵌入式脚本。图任务1b和1d中的每一个指向(或以其他方式引用)可执行数据流图(例如，存储在存储装置中的可执行数据流图)。通常，可执行数据流图(也被称为可执行图)包括有向数据流图，图中的顶点表示组件(数据文件或过程)，并且图中的链路或“边”指示组件之间的数据流。在名称为“EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS(执行表示为图的计算)”的在先美国专利号5,966,072中描述了一种用于执行这种基于图的计算的系统，该专利通过引用并入本文。

[0088] 在一些情况下，数据流图的构造本质上可以是高度技术性的。数据流图可以实施基于图的计算，该基于图的计算是对从一个或多个输入数据集流过处理图组件的图到达一个或多个输出数据集的数据执行的，其中，该数据流图由数据存储装置中的数据结构指定，该数据流图具有由这些数据结构指定并表示由一个或多个链路连接的图组件的多个节点，这些链路由这些数据结构指定并表示这些图组件之间的数据流。可以使用耦接到数据存储装置并托管在一个或多个计算机上的运行时环境(或执行环境)，该运行时环境包括预执行模块，该预执行模块被配置用于读取指定数据流图的已存储数据结构并用于分配和配置计算资源，比如用于执行对由该预执行模块分派给数据流图的图组件的计算的过程，该运行时环境包括执行模块，该执行模块用于调度和控制对所分派过程的执行，以使得能够执行本文描述的方法中的任一种。

[0089] 虽然是为了实现特定的业务目的而编写的，但是图的基础结构和构造是基于技术考虑而确定的。例如，可以选择图组件来最大化可重用性，或支持并行处理。另一方面，如何使用图以及在何处使用图可能在很大程度上是业务决策。与参数化的数据流图相关联的参数中的一些参数可以用于使业务用户能够自定义数据流图，而不需要用户理解其实施背后的技术复杂性。参数化的数据流图简化了自定义化并促进重用。

[0090] 计划任务1e、1f中的每一个指向位于外部文件中的计划。条件任务1c基于指定的表达式或程序在计划1中实施决策逻辑。子计划1g将任务(包含在子计划1g中)分组为计划的一部分，并允许迭代地运行这些任务。计划1还包括转换1i至1o，这些转换指示例如任务之间的依赖性并因此控制任务之间的执行顺序。也就是说，转换1i至1o表示这些任务之间的连接，并因此指定了执行依赖性。

[0091] 例如，任务通过指定哪些受控应用程序将被执行并封装关于这些受控应用程序的信息(例如，位置、自变量等)，该任务提供用于执行方法的框架。受控应用程序的编程代码可以存储在存储装置中并由任务引用。这些方法执行任务的动作。也就是说，计划不是基础

代码源的可视化。而是,计划本身是可以被运行或执行并且还封装用于或表示要执行的各种可执行文件的信息的可视化程序。计划还包括用于在任务之间传递值的属性(例如,参数)。在示例中,参数的改变可以影响在参数值被改变的任务的相同(或不同)层次级别的子计划、方法和其他任务。计划还指定了控制任务被运行的顺序的关系。计划根据任务之间的连接以及方法和参数的规范来控制任务之间的顺序、关系和通信。

[0092] 参考图1B,图示1p展示了可以将断点插入计划1中的各种位置。通常,断点包括用于在执行环境中暂停或停止计划的执行的指令(例如,指示、命令或规范)。当执行计划遇到断点时,计划暂停以使得能够评估(例如,自动评估或用户评估)计划的内部属性(例如,比如计划的操作参数和相关联值、计划正在使用的动态参数和相关联值、以及在断点在图或计划被启动之前时将要发送到该图或计划的输入值)和/或评估计划外部的属性(比如正在执行计划的执行环境或系统的属性和/或外部状态和资源(例如,由执行环境在执行计划时使用的数据库和储存库的状态))。这些断点使得能够监控在执行计划遇到断点时盛行的属性,并从而监控基础系统(例如,计划和/或执行环境)的内部状态。例如,使得用户能够查看用于任务的参数值并且还能够例如通过改变参数的值来改变该任务(以及下游的或依赖于该任务的其他任务)的功能。替代性地或另外,使得用户能够查看计划外部的方面,比如指示在计划遇到断点之前计划执行期间执行环境的性能(例如,CPU使用、主存储器的使用、可用或占用的数据存储容量、消耗或可用的网络带宽等)。

[0093] 在这个示例中,可以在位置1q处插入断点,该位置表示到给定功能模块(即,任务1b)的转换。也可以在任务1b中的位置1r处插入断点。在位置1r处设置断点刚好在任务1b准备产生运行程序的所表示过程(或任务)之前或者在其他功能或动作之前中断任务1b的执行。通常,由本文描述的系统对断点进行设置或插入是指由系统对断点进行记录。也就是说,位置1r表示刚好在任务1b或其一些动作的执行开始之前或开始时的时间。也可以在表示从任务1b的转换的位置1s处(例如,位置1s表示任务1b的执行完成之后的时间)插入断点。

[0094] 参考图1C,图示3包括断点指示符2a,该断点指示符展示了为计划1中的任务1b记录断点。断点指示符2a指定一个或多个断点已经被记录在任务1b上(即,这些断点与任务1b相关联),但是没有将断点的位置指定为在任务1b之前、之中或之后。在这个示例中,例如通过选择或悬停在断点指示符2a上,一个或多个断点被设置在何处的细节是“按需”可用的。通常,断点指示符是一个或多个断点的存在的指示或视觉表示。在这个示例中,当系统执行由断点指示符2a表示的断点时,计划1的执行暂停,并且呈现检查器用户界面(例如,被称为“断点检查器”)。每当计划执行被暂停(例如,由于下面描述的故障时中断模式开启并发生故障;或者由于在计划执行过程期间遇到由用户设置的断点)时,就会呈现断点检查器。

[0095] 参考图1D,图示4展示了计划1已经遇到由断点指示符2a表示的断点,如由任务1b周围的虚线指示的。也就是说,任务1b由于到达由断点指示符2a表示的断点而停止,但是计划1的其余部分继续运行。在这个示例中,箭头1i、1j、1l被示出为粗线(例如,相对于图1D所示的其他箭头的粗细程度),以表示计划1执行了任务1a、1e、1f。在这个示例中,任务1a、1e、1f执行(例如,并行执行,或者任务1a与以串行顺序执行的任务1e和1f并行执行),并且计划1尽可能地前进,只要计划执行在任务1b处暂停即可。由于为计划1记录了由断点指示符2a表示的断点,因此为用户显示图形用户界面(例如,断点检查器)以在执行任务1b之前检查

任务1b的属性,如下面参考图1E所述。可以响应于计划1遇到断点指示符2a而触发或发起对该图形用户界面的显示。当计划1的其余部分(例如,任务1e和1f)继续运行时,可以显示该图形用户界面。在这个示例中,任务1c在任务1b完成并成功之前将不会运行,并且当执行在由断点指示符2a表示的断点处停止时,任务1b将不会完成。

[0096] 参考图1E,例如,当到达表示由断点指示符2a(图1D)表示的断点的指定位置的(图1D中的计划1的或表示计划1的基础数据结构的)执行点时,显示图形用户界面5c。在这个示例中,图形用户界面5c在标题中显示执行暂停的原因,即任务名称后的"断点"(即,"运行主日常过程")。图形用户界面5c还包括数据源部分4a,该数据源部分为计划1中的任务1b(图1D)指定数据源。在这个示例中,该数据源未能正确解析,例如,因为其已解析为"/inputsrc#dat"。通过输入部分4a,可以指定正确的数据源路径。数据源在本文中被用作由图形程序元素1b表示的任务在任务执行期间所使用的参数的示例,并且数据源路径是这种参数的值的示例,该值可以由用户通过图形用户界面的输入部分4a来修改或校正。参考图1F,图形用户界面5d是在正确的数据源已经被输入到部分5a(或图1E中的部分4a)中之后的图形用户界面5c的版本。在这个示例中,在将新信息(例如,修改后的参数值)输入到部分5a中时,图形用户界面5d显示继续控件5b,对该控件的选择使任务1b(图1D)继续执行并且可选地使图形用户界面5c、5d中的每一个停止显示。这是在计划继续运行时关于如何对任务进行改变的示例。在这个示例中,选择继续控件5b。在变型中,当任务已失败时,显示图形用户界面以使得能够修改(例如,由任务在其执行期间使用的参数的)各种值和/或由任务使用的输入。在修改后,图形用户界面显示用于重新开始这个失败的任务的控件,例如,同时计划的其余部分仍继续运行。

[0097] 在又一个变型中,在任务已失败之后,用户可以在任务1b开始时设置断点(例如,由断点指示符2a表示的断点)。这样做之后,用户在命令行中执行命令(例如,"计划-管理重新开始<任务>"命令)以重新开始任务1b。然后,当任务1b在开始处暂停时,可以改变DATA SOURCE参数的值(如上所述),使得当任务1b重新开始时,其将使用校正后的值。在这个示例中,新值仅对计划1的这次运行有效。要永久改变参数的值,需要编辑计划本身。

[0098] 参考图1G,图示6示出了任务1b的重新执行(例如,在图1F中选择继续控件5b之后或者在执行命令(例如,上述的"计划-管理重新开始<任务>"命令)之后)。在这个示例中,任务1b使用DATA SOURCE参数的正确值(基于任务1b由于断点指示符2a所表示的断点而在开始处暂停并且DATA SOURCE参数的值被改变为校正后的值)。因此,计划1通过任务1b前进到条件任务1c,如由加粗的箭头1m、1l所示。

[0099] 参考图2,控制流图10(也被称为计划10)包括图形程序元素12、14、16和转换18、20。图形程序元素表示相应的任务12、14、16。转换18、20表示由计划10指定的控制流。为了方便读者,每当参考指示图形程序元素或转换的附图标记时,在描述相应的附图时分别直接参考相应的任务和控制流。也就是说,图形程序元素12、14、16分别被称为任务12、14、16,并且转换18、20分别被称为连接18、20。在这个示例中,计划10包括连接18、20,这些连接指定首先执行任务12,然后执行任务14,并且然后执行任务16。在这个示例中,计划10的行为可以被修改为包括使计划10停止或暂停执行的一个或多个断点,从而允许对计划、任务、子计划和/或参数进行检查和/或修改。

[0100] 该计划调试功能例如通过使得能够在各种层次级别处设置断点来提供关于在何

处设置断点的灵活性。例如,系统可以在整个计划层级、子计划层级、任务层级、或甚至任务的方法层级设置断点。另外,断点可以设置在以下三个位置处:设置在计划、子计划、任务或方法之前;设置在计划、子计划、任务或方法之中;以及设置在计划、子计划、任务或方法之后,下面将进一步详细描述这些位置中的每一个。

[0101] 在这个示例中,可以在任务14之前的位置22处记录断点。位置22处的断点在任务14开始之前停止该任务的执行。对于程序任务、计划任务和图任务,执行的这种停止发生在获取资源(例如,受约束的元素,比如,指定数量的可用令牌)之前。对于图任务,执行的这种停止也发生在图被编译以供执行之前。可以在任务14中的位置24处记录断点。在这个示例中,任务14被配置用于运行计算机程序28。因此,位置24处的断点在任务14执行其执行方法之前中断该任务的执行。位置24处的断点提供对计划10的环境变量的修改以及对将在程序28中使用的输入值(例如,输入参数的表达式)的查看和编辑。可以在任务14之后的位置26处记录断点以在任务14结束之后停止该任务的执行。

[0102] 在这个示例中,断点可能出现的位置是用于检查断点是否已被请求的非常有效的位置。因此,由于对断点的检查,计算资源的消耗(比如CPU消耗)没有(或只有最小的)可测量的增加。这是因为计划中用于检查断点的编程代码是最少的并且因此计算成本不高。在示例中,程序被配置用于每当程序前进到可以插入断点的位置(例如,在表示计划、子计划、任务或方法之前、之中或之后的每个位置处)时,检查哈希表以查找断点的出现。在示例中,第一计划(“计划A”)包括第一任务(“任务A”)之前的断点。在这个示例中,哈希表包括以下条目“/planA/taskA/before”,以指示在任务A之前存在断点,该任务被包括在计划A中。因此,当系统执行与任务A的执行之前的位置相对应的程序部分时,系统执行查找(例如,字符串查找)以标识是否存在表示在任务A的执行之前的该位置的条目。在这个示例中,该条目被填充有字符串“/planA/taskA/before”。因此,系统在任务A之前执行断点。在另一示例中,哈希表为空。在这个示例中,系统不检查指定断点的条目。

[0103] 参考图3,图示40显示了表示任务的图形程序元素42。为了方便读者,每当参考指示图形程序元素的附图标记时,在描述该图时直接参考相应的任务。因此,图形程序元素42也被称为任务42。在这个示例中,如前所述,可以在任务42之前的位置44处记录断点,可以在任务42中的位置46处记录另一个断点,并且可以在任务42之后的位置48处记录又一个断点。在这个示例中,任务42包括或引用脚本,并且进一步包括可以在计算机系统上(例如,在本文描述的系统的操作系统中)执行的若干方法(每种方法包括如前所述的一个或多个动作)。这些方法是由本文描述的系统提供的现有方法。在这个示例中,任务42配置有“触发”方法49、“开始时”方法50、“执行”方法52、“成功时”方法54、“失败时”方法56和“回滚时”方法59,这些方法中的每一种执行任务42的一个或多个动作或者表示任务42的一种或多种状态。在这个示例中,触发方法49可以被实施为表示任务42的起点。该触发方法可以包括开始执行的条件。可能存在各种条件,包括例如特定文件是否存在、标志是否已经被设置为真等。开始时方法50设置执行以执行任务42的动作。例如,开始时方法50可以被实施为使系统为执行方法做好准备的方法,比如将环境变量设置为期望值、或者设置日志文件来记录运行时信息。执行方法52例如通过运行由任务42表示的可执行文件来执行任务的指定动作。例如,执行方法52可以被实施为执行任务42的主要功能。如果执行方法52在其执行期间成功,则执行成功时方法54以退出任务42。如果执行方法52在其执行期间失败,则执行失败时

方法56以退出任务42。在这个示例中,在失败时方法56之后执行回滚时方法59。在这个示例中,执行方法之前和之后的方法被称为“框架方法”。“执行”方法之前的框架方法设置供受控应用程序运行的环境(和/或情况),但不是受控应用程序的一部分。执行方法之后的框架方法(即失败时方法、成功时方法和回滚时方法)也是计划的一部分并基于受控应用程序的运行结果来执行。框架方法不执行受控应用程序的动作。这些框架方法是计划的控制处理层的一部分。

[0104] 在这个示例中,可以在触发方法49之前、之中和之后设置断点,如由断点49a、49b、49c指定的。在这个示例中,可以在开始时方法50之前、之中和之后设置断点,如由位置50a、50b、50c分别指定的。在这个示例中,位置50a处的断点指定在开始时方法50开始之前暂停或中断任务42的执行。位置50b处的断点指定在开始时方法50开始时暂停或中断任务42的执行。位置50c处的断点指定在开始时方法50完成时暂停或中断任务42的执行。可以在执行方法52之前、之中和之后记录断点,如由位置52a、52b、52c指定的。在这个示例中,在任务42中的位置46处记录断点与在执行方法52的执行之前的位置52a中记录断点相同。

[0105] 另外,对于成功时方法54,可以分别在成功时方法54执行其动作之前、在成功时方法54执行其动作时以及在成功时方法54执行其动作之后在位置54a、54b、54c处记录断点以中断成功时方法54。在示例中,可以在位置56a、56b、56c处记录断点——(分别)在失败方法56之前、之中或之后。例如,对于失败时方法56,可以在位置56a处记录断点以在失败时方法56运行之前确定任务42的属性设置。在这个示例中,回滚时方法59例如通过从失败点开始以相反的执行顺序回滚已经进行的工作来回滚失败的任务。例如,本文描述的系统可以存储图或计划的恢复文件。系统使用该恢复文件(连同回滚时方法)以在必要时进行清理。在示例中,可以在位置59a、59b、59c处记录断点,(分别)表示在回滚时方法59之前、之中或之后。位置59a处的断点指定在回滚时方法59开始之前暂停或中断任务42的执行。位置59b处的断点指定在回滚时方法59开始时暂停或中断任务42的执行。位置59c处的断点指定在回滚时方法59完成时暂停或中断任务42的执行。在变型中,成功时方法、失败时方法和回滚时方法是可选的。因此,当这些方法不包含在任务中时,就不能为这些方法设置断点。

[0106] 除了设置断点之外或者代替设置断点,也可以在故障时中断模式下运行计划,这会在计划遇到故障时使计划的执行被中断。例如,如果图任务、计划任务或程序任务的执行方法因其运行的图、子计划或程序失败而失败,则计划的执行将在执行方法结束之后暂停。当计划在故障时中断模式下运行并且遇到故障时,显示断点检查器以使得用户能够查看故障的原因和发生故障的功能模块的属性。

[0107] 在一些示例中,用户可以设置故障时中断、等待任务失败(在执行方法的之后断点处)、诊断问题、修复问题、然后重新运行计划以避免失败。在这个示例中,系统可以执行命令来重新开始任务,例如,任务生命周期的任何点。也就是说,即使任务在其“之前”断点处停止,命令仍然可以成功发出。然后,当任务最终失败并结束时,其将自动重新开始,如下:

[0108] 系统在故障时中断模式下执行计划(或在执行方法之后设置断点)。在已经设置断点的情况下,系统等待在计划执行期间遇到断点。在计划的执行暂停之后,用户或系统检查状态以诊断失败的原因。诊断之后,用户在任务或执行方法上设置(或系统记录)“之前”断点。用户或系统发出重新开始任务的命令并且系统等待遇到该“之前”断点(当自动重新开始任务时)。用户或系统校正失败的原因(例如,通过覆盖参数值、改变文件等)并从该“之

前”断点继续,使得任务将成功。

[0109] 与关于特定计划元素(例如,计划、子计划、任务或方法)指定将在那里暂停执行的特定位置(例如,之前、之中或之后)的断点不同的是,故障时中断模式不指定这种类型的特定计划元素和位置。而是,故障时中断模式是执行计划(并在失败时使执行暂停)的模式并且不指定特定计划元素。替代地,执行将在失败的任务处自动暂停。在一些示例中,故障时中断模式总是使执行在失败任务的执行方法的之后位置处暂停。

[0110] 参考图4,图形用户界面100显示计划的图形层102(为了方便而非限制,在本文中统称为“计划102”),该图形层包括表示相应任务的图形程序元素104、106、108、110并且还包括表示子计划的图形程序元素112。为了方便读者,每当参考指示图形程序元素的附图标记时,在描述下面的图5至图7时直接参考相应的任务和子计划。因此,图形程序元素104、106、108、110在本文中分别被称为任务104、106、108、110,并且图形程序元素112在本文中被称为子计划112。

[0111] 在这个示例中,任务106是移动和重新格式化图任务(在下文中也被称为“移动和重新格式化图任务106”)。移动和重新格式化图任务106是包括或引用可执行数据流图(例如,移动和重新格式化图)的功能模块。执行的数据流图是数据处理层的一部分,而移动和重新格式化图任务106是图形层的一部分并封装关于如何执行数据流图的信息。在这个示例中,任务108是子计划,在下文中也被称为“简单依赖计划108”。通常,子计划是作为新过程(即“子过程”)被执行的计划,该新过程不同于运行执行子计划的计划的过程。数据流图(例如,移动和重新格式化图)在与运行计划的过程不同的过程中被执行。

[0112] 另外,在这个示例中,任务104是条件任务,包括例如被配置用于在满足条件时执行一个或多个第一动作(例如,通过执行一种或多种方法)并且在不满足该条件时执行一个或多个第二动作(例如,通过执行一种或多种其他方法)的任务。在示例中,每个任务可以被配置为条件任务。另外,条件任务可以指定除成功或失败之外的条件,例如特定值。在这个示例中,任务104检查文件是否存在。任务104在下文中也被称为“检查文件是否存在任务104”。检查文件是否存在任务104是条件逻辑的实施,根据特定文件是否存在,将控制引导到移动和重新格式化图106或简单依赖计划108。子计划112本身是控制流图并且在运行计划102的相同过程中执行。

[0113] 参考图5,图形用户界面120包括用于设置断点的控件122、124、126、128、130。在这个示例中,控件122使得用户能够设置断点(通常针对计划),并且控件124使得用户能够在计划开始之前设置断点。控件126使得用户能够在计划中以及计划内容运行之前设置断点。控件128使得用户能够在计划退出之前设置断点。控件130使得用户能够为各种方法和条件设置断点。

[0114] 参考图6,图形用户界面140提供用于在任务上设置断点的控件。在这个示例中,用户选择检查文件是否存在任务104(图4)并从调试菜单144中选择“设置断点”选项142。在该选择之后,图形用户界面140显示控件146、148以使得能够分别在任务104开始之前或任务104结束之后设置断点。在这个示例中,用户选择控件146以在任务104开始之前设置断点。

[0115] 参考图7A,图形用户界面160显示计划102,在该计划中,已经在任务104上记录了断点(如由断点指示符164a指示的)。在这个示例中,断点指示符164a指定已经在任务104上设置了一个或多个断点,但是没有将断点的位置指定为在任务104之前、之中或之后。另外,

已经例如通过选择子计划112并且然后选择“设置断点”选项142(在图6中)在子计划112上记录了断点(如由断点指示符164b指示的),这使得用户能够指定是在子计划112开始之前还是在子计划112结束之后设置断点。在这个示例中,在子计划112开始之前在子计划112上设置断点。可以同时遇到一个以上的断点,例如,从而使计划的执行在计划的不同位置同时暂停。在这个示例中,可以同时遇到由断点指示符164a、164b表示的断点。在示例中,一旦已经设置了断点,调试菜单就可以提供“显示所有断点”控件,对该控件的选择显示断点编辑器对话框,该对话框具有已经在计划中设置的断点的列表。通过这个断点编辑器对话框,用户可以添加、删除、编辑和使用已定义的断点集。一旦到达由断点指示符164a、164b表示的断点中的一个或多个断点,就更新图形用户界面160以呈现覆盖图或界面(被称为“到达断点”窗格),该覆盖图或界面列出计划102由于各个断点的执行而现在被暂停的位置。

[0116] 参考图7B,例如,一旦遇到由断点指示符164a、164b表示的断点中的一个或多个断点,就显示到达断点用户界面170(例如,在图7A中的图形用户界面160附近或与之并列)。在这个示例中,例如同时遇到由断点指示符164a、164b表示的两个断点。因为到达断点用户界面列出了计划当前被暂停的位置,所以到达断点用户界面170包括部分171a,该部分指定计划102当前由于由断点指示符164a指定的(在部分171a的区域172a中表示的)“在任务开始之前”断点而在特定功能模块处(即,在任务104处)被暂停。到达断点用户界面170还包括部分171b,该部分指定计划102当前由于由断点指示符164b指定的(在部分171b的区域172b中表示的)“在子计划开始之前”断点而在另一个功能模块(即,子计划112)处被暂停。在这个示例中,在选择部分171a、171b(或部分171a、171b中分别指定遇到的断点类型的区域172a、172b)之一时,显示断点检查器用户界面,并且该断点检查器用户界面呈现在所选择部分中表示的功能模块的参数、环境变量等。

[0117] 参考图8,一旦到达断点,就显示图形用户界面180,并且该图形用户界面是断点检查器用户界面的示例。在这个示例中,一旦到达由断点指示符164a(图7A)表示的断点并且在选择了图7B中的到达断点用户界面170中的部分171a之后,显示图形用户界面180。在这个示例中,图形用户界面180显示参数选项卡172,用于显示任务(在此情况下是图7A中的任务104)的参数和相关联参数值。在这个示例中,控件182、184提供对各种参数值的编辑。在这个示例中,对控件184的选择提供对指定表达式的值(即,指定该表达式的值是“0”还是“1”)的参数值的编辑。

[0118] 图形用户界面180包括显示任务104的输入值的输入值选项卡174、显示计划102(图4和图7A)的执行环境变量或参数和相关联值的环境选项卡176、以及评估选项卡178,该评估选项卡提供用于撰写表达式的控件和/或输入框,然后根据计划的执行状态和当前暂停位置对这些表达式进行评估。

[0119] 选项卡172、174、176、178中的每一个提供用于编辑(例如,变量、参数、输入值和环境变量或参数的)适当值的控件。一旦用户完成编辑选项卡172、174、176、178中的一个或多个选项卡中的值,用户可以指定(例如,通过选择控件)例如从可能已经到达的所有断点位置继续运行计划。在变型中,用户可以选择特定断点位置并刚好从该断点位置继续。

[0120] 参考图9A,在选择控件184(图8)之后,显示图形用户界面190。图形用户界面190包括用于修改参数或变量的值(例如,用于修改指定表达式的参数的值)的输入部分192。在这个示例中,输入部分192允许录入和修改指定表达式评估为真(值“1”)还是假(值“0”)的参

数值。在这个示例中,由断点指示符164a表示的断点可以用于例如通过强制从由任务104表示的过程(或任务)遍历到由任务106表示的过程(或任务)(例如,当参数表达式被设置为值“1”时)或者通过强制从任务104遍历到任务108(例如,当参数表达式被设置为值“0”时)来修改控制流。

[0121] 参考图9B,例如在图8和图9A中完成编辑之后,显示图形用户界面191。在这个示例中,图形用户界面191显示继续控件191a,对该控件的选择从遇到的断点位置(即,由断点指示符164a、164b表示)继续计划102的执行。

[0122] 参考图9C,图形用户界面192还显示用户可以选择以从特定断点位置继续计划执行的控件(或用户界面的可选择部分)。在这个示例中,图形用户界面192包括具有覆盖图193的到达断点用户界面170(先前在图7B中示出)。对到达断点用户界面170的部分171a中的区域172a的选择使覆盖图193被呈现。覆盖图193包括控件193a,对该控件的选择指示系统从区域172a中表示的断点(即,在图7A中由断点指示符164a表示的断点)继续计划102。

[0123] 参考图10,执行环境200托管在一个或多个计算机上。当参数进入执行范围时,执行环境200内的每个元素(例如,计划202、子计划204和计划202中包括的图形程序元素206(表示任务))对这些参数进行评估。为了方便读者,每当参考指示图形程序元素的附图标记时,在描述下面的图10至图12时直接参考相应的任务。因此,图形程序元素206在下文中被称为任务206。如本文所述,可以设置断点,以便在计划的各个级别查看和修改这些参数值。通常,计划环境包括用于执行和定义计划的计算机化环境。一般来说,参数进入执行范围的顺序如下。执行环境200包括执行环境参数208,包括例如可能通过计划的沙箱或项目在计划的环境中定义的参数。执行环境参数208在计划202开始运行之前可用于该计划。计划202包括或访问计划级参数210,包括例如在计划级声明并在计划202开始运行时评估的参数。计划202还可以从其执行环境(即,执行环境200)引用参数(即,执行环境参数208);当计划202开始运行时,这些值也被解析。

[0124] 在这个示例中,计划202包括具有子计划级参数212的子计划204,当子计划204开始运行时,系统解析这些参数。在这个示例中,子计划204是循环子计划,如由循环符号214指示的。通常,循环子计划包括被配置用于循环指定次数或直到满足条件的子计划。在这个示例中,循环子计划是循环并且可以被重复的功能模块。因此,对于每个循环,可以在该循环之前、之中或之后设置断点。每当子计划204开始新的循环时,系统(或在系统上运行的引擎)也重新评估子计划级参数212。当子计划204循环时,可以使得其参数的值是动态的并将其从一个循环传递到下一个循环。通过对子计划204使用断点,可以查看和修改这些参数的值以便在下一个循环中使用。然而,如果子计划204被配置用于并发循环,则参数值对于每次迭代都保持私有。

[0125] 在这个示例中,子计划204包括任务216(具有任务级资源218)并且计划202包括任务206(具有任务级资源220)。当计划的控制流到达特定任务的输入端口时,该任务进入执行范围。在系统可以运行任务的方法之前,系统为该任务定位并获取资源。在这个示例中,除了系统访问资源所需的隐藏参数之外,系统不在任务级别处解析参数。因此,当任务的执行方法开始时,评估任务的可查看和可编辑的参数(例如,通过图8中的选项卡172)。

[0126] 在这个示例中,任务206包括触发方法222、开始时方法224、执行方法226和成功时方法228。在这个示例中,触发方法222和开始时方法224设置受控应用程序运行的环境(和/

或情况)。执行方法226使受控应用程序运行。当受控应用程序的执行结果指示受控应用程序运行成功时,成功时方法228运行。

[0127] 触发方法222可以被实施为表示任务206的起点。触发方法222可以包括用于开始执行的各种条件。这些各种条件包括例如特定文件是否存在、标志是否已经被设置为真等。在这个示例中,任务206包括方法级参数230、232、234、236。在已经获得资源之后,任务的方法开始进入执行范围。在这个示例中,系统以预定顺序执行方法。因此,本文描述的系统以相同的顺序解析方法级参数。

[0128] 任务进入执行范围的第一种方法是触发方法222。当这种情况发生时,系统解析由触发方法222声明或引用的方法级参数230。在这个示例中,由系统在触发方法222中设置的参数值可以被传递给随后的方法,例如,通过将这些参数声明为动态参数。假设触发方法222成功,进入执行范围的下一种方法是开始时方法224,该方法包括方法级参数232,例如,在触发方法222中设置的动态参数值。

[0129] 在这个示例中,在用户界面中(例如,在任务性质对话框参数选项卡中)设置执行方法226的方法级参数234。当执行方法226分别在触发方法222和开始时方法224已经运行之后进入执行范围时,系统评估这些参数。与前述方法一样,可以在执行方法226之前设置断点,以使得能够修改参数值并将这些修改后的值传递给后续方法,例如,失败时方法(未示出)和成功时方法228。成功时方法228的方法级参数236可以访问通过执行方法226设置的参数值。在这个示例中,当任务的方法结束运行时,其本地参数值不会被保存。但是,下一个任务可以访问计划参数以及导出到执行环境的任何值。任务还会留下其声明为动态的任何参数的更改值。这些更改值可以被后续任务看到。如图11和图12中所述,系统可以被配置用于相对于参数被解析的位置设置断点。

[0130] 参考图11,图示300展示了计划308或子计划310的断点可以被设置(例如,由系统记录,如上文所描述)在各种位置,每个位置提供监控或编辑各种类型的参数值的能力,包括上文描述的那些。在这个示例中,在运行计划308之前设置执行环境参数302。计划308包括或访问计划级参数304。子计划310包括或访问子计划级参数306。在这个示例中,系统可以在位置312处设置断点以在计划308开始之前停止该计划的执行。在位置312处设置断点提供对计划级参数304的修改。系统可以在位置314(计划308内)处设置断点以刚好在计划308开始其内容之前停止该计划的执行。系统还可以在位置316处设置断点以在计划308结束之后并且在该计划退出之前停止该计划的执行。系统可以在位置318处设置断点以在子计划310开始之前停止该子计划的执行。在位置318处设置断点提供对子计划级参数306的修改。系统可以在位置320(子计划310内)处设置断点以刚好在子计划310开始其内容之前停止该子计划的执行。系统(和/或用户)还可以在位置322处设置断点以在子计划310结束之后并且在该子计划退出之前停止该子计划的执行。

[0131] 参考图12,图示340展示了可以在五个不同的位置344、345、346、348、350处设置任务342的断点。在这个示例中,任务342包括方法349(例如,触发方法和开始时方法,每种方法可以是可选的并且是框架方法)、执行方法356和方法352(例如,成功时方法、失败时方法和/或回滚时方法,每种方法可以是可选的并且也是框架方法)。

[0132] 任务342还包括任务级资源358和方法级参数360。在这个示例中,位置344处的断点在任务342开始之前停止该任务的执行。当任务342是程序任务、计划任务或图任务时,该

任务的执行的这种停止在任务级资源358被获取之前。对于图任务,该任务的执行的这种停止也在图被编译之前。在位置344处设置断点使得能够修改方法参数360。

[0133] 在这个示例中,位置345是任务342上的“之中”断点位置(例如,其与执行方法52(图3)上的“之前”位置52a(图3)相同)。也就是说,在位置345处设置断点会在(任务342的)执行方法开始之前(而不是恰好在执行方法运行其脚本或可执行文件之前)停止该执行方法的执行。对于图任务和计划任务,该断点位置345致使启动断点检查器,通过该断点检查器,可编辑(由图任务和计划任务使用或执行的)输入表达式。因此,这些新的表达式将被评估并被发送到启动的图或计划。

[0134] 在执行方法356中,系统还可以在位置346处设置断点。也就是说,位置346是执行方法356的“之中”位置(例如,与图3中的位置52b相同)。当断点设置在位置346处时,系统刚好在其执行方法运行其脚本或可执行文件之前停止程序任务的执行,刚好在其执行方法运行其子计划之前停止计划任务的执行,或者刚好在其执行方法运行其图之前停止图任务的执行。对于计划任务、图任务和程序任务,位置346处的断点使得能够修改环境变量(例如,改变将为计划任务、图任务和程序任务产生的那些过程的环境变量)。对于图任务和计划任务,位置346处的断点还使得能够查看要用于图354的输入值。

[0135] 在这个示例中,执行方法356被配置用于发起或产生图354。(在变型中,执行方法356可以被配置用于发起子计划,而不是图354)。在这个示例中,可以在位置348处设置断点以在图354开始时停止执行,并在新的选项卡(例如,显示图354的内容的窗口370)中打开该图354。窗口370显示图354(在这个示例中为图374)的内容并显示执行被暂停,例如,通过可视化372。通过打开的图或子计划,用户可以检查当前状态、设置额外的断点、或使用“继续控件”来恢复图或子计划的执行。这种类型的断点被称为“在图/子计划内中断”或“内断点”。这就产生了这样一种情况,即计划启动图或子计划,并为该图或子计划自动启动调试器——然后该调试器恰好在该图或子计划的执行开始时停止该图或子计划。例如,当在计划运行时动态地确定要执行的图或子计划时,即当在计划运行之前的编辑时间没有预先确定要执行哪个图或子计划时,这种能力非常有用,因此不能为其显式设置断点。在这个示例中,内断点将致使停止实际运行的图或子计划的执行。

[0136] 对于内断点,用于启动的过程的调试器会自动启动,并且GDE本质上表示两个独立的单个过程之间的关系。这是因为一台机器具有父计划的过程和父计划的调试器的过程。第二台机器具有所启动的图或子计划的过程。该第二台机器也具有所启动的图或子计划的调试器的过程。因此,计划中的一个图可以在本文描述的系统上运行并且另一个图可以在另一台机器上运行,并且用户可以使用图调试器来调查该机器上的本地环境。

[0137] 在另一示例中,图内的中断使GDE切换(调试器)以查看图(例如,图374)的方面和属性,而不是查看父任务(例如,任务342)的方面。另外,图或子计划(即使其作为不同于任务342的过程被执行)可以在与任务342相同的用户界面和相同的图形开发环境(GDE)中被调试。在这个示例中,系统可以在位置350处设置断点以在任务342结束之后停止该任务的执行。

[0138] 在一些示例中,一旦任务342被停止,用户就可以指定任务342应该像其已经成功或者其已经失败一样继续进行。系统记录这些用户改变和规范并且例如通过根据这些用户改变修改任务(或方法)的状态并且通过这样做控制接下来执行哪些任务或方法来作用于

这些用户改变和规范。系统通过将方法的状态从成功变为失败来实施控制流，并且反之亦然。

[0139] 另外，在一些示例中，呈现图形用户界面(在“之后”断点(例如，在任务342上的位置350处的断点)处)，该图形用户界面使得能够修改任务成功还是失败的记录(这随后影响控制的下一步走向)。在一些示例中，用户界面使得能够在执行特定类型的任务或方法(例如，条件任务)之后以及在图任务或计划任务的执行方法之后修改记录。

[0140] 另外，本文描述的系统提供了在专门位置处设置断点。例如，计划或子计划可以是循环计划或循环子计划(例如，被配置用于串行循环或并发循环)。通常，循环计划包括被配置用于循环指定次数或直到满足条件的计划。对于循环计划或子计划，可以将断点设置为在循环的每次迭代上暂停，或者可以设置将仅在指定迭代上暂停的特定于迭代的断点(“迭代断点”)。例如，在记录断点时，系统可以指定执行中的暂停仅发生在计划/子计划/任务/方法的与该循环的特定迭代相对应的实例上(例如，从而指定断点适用于计划/子计划/任务/方法的哪个实例，因为计划/子计划/任务/方法在循环内并且因此将有计划/子计划/任务/方法的许多单独实例被执行)。另外，系统可以被配置用于设置条件断点，包括例如被配置用于仅当满足特定条件时才中断计划、子计划或任务的执行的断点。在这个示例中，断点包括表达式或与表达式相关联，并且系统被配置用于仅当表达式满足指定值(比如“1”或“0”)时才执行断点。系统可以为本文描述的各种类型的断点中的任何一种设置或以其他方式指定条件，以仅当所指定的条件为真时使断点停止，从而使各种类型的断点中的每一种成为条件断点。例如，迭代断点可以具有为其指定的条件。也就是说，例如，在作为第二循环执行的一部分的任务的实例上设置的断点也可以指定只有在满足条件(独立于循环)时执行才应该暂停。

[0141] 除了图形计划调试器之外，系统还提供命令行版本。系统可以将命令行调试器连接到运行计划并提供各种类型的调试，这些调试可以用图形方式执行。

[0142] 参考图13，过程400由系统在调试指定多个功能模块之间的控制流的图形控制流应用程序时实施，控制流被表示为多个功能模块之间的转换并且每个功能模块被配置用于执行一个或多个动作。在操作中，系统指定(402)可执行控制流图中可执行控制流图的执行将被中断的位置。所指定的位置表示到给定功能模块的转换、到执行该给定功能模块的内容的状态的转换、或从该给定功能模块的转换。系统还在执行环境中开始(404)可执行控制流图的执行。通常，执行环境是例如在运行时由操作系统提供给计划或软件的环境。在这个示例中，在表示该指定位置的执行点处，系统中断(406)可执行控制流图的执行并提供(408)表示该给定功能模块正在其中执行的执行环境或运行时环境的一个或多个属性(例如，运行计划的执行环境或运行时环境的内部属性(例如，比如参数和相关联值、计划正在使用的动态参数和相关联值等)和运行计划的执行环境或运行时环境的外部属性(比如，正在执行计划的本地环境或系统以及外部状态和资源(例如，数据库和储存库)))的数据。

[0143] 在一些示例中，可执行控制流图包括图形层和控制处理层，其中，该图形层包括用于图形用户界面的视觉效果和用这些视觉效果编程的可视化编程语言的可视化程序，该可视化编程语言用于指定控制流，其中，该控制处理层包括功能模块、方法、在执行这些方法时使用的基础数据结构、以及使用这些基础数据结构执行这些方法以实现由该可视化编程语言指定的控制流的代码。在这个示例中，该方法还包括通过用这些视觉效果呈现该图形

用户界面来执行该图形层;

[0144] 通过该图形用户界面接收表示用户对在该图形用户界面中显示的这些视觉效果中的一个或多个视觉效果的操纵的数据,这些所接收的数据以该可视化编程语言表示该可视化程序;以及根据在该图形层中创建的该可视化程序来执行该控制处理层,该控制处理层的执行包括执行由该可视化程序指定的一个或多个功能模块,功能模块的执行使得通过使用该功能模块的方法的一个或多个数据结构执行代码来执行该方法从而实现由该可视化编程语言指定的控制流。在这个示例中,特定功能模块表示包括在数据处理层中的受控应用程序,其中,该方法进一步包括:执行该特定功能模块的特定方法;以及响应于执行该特定方法,启动该受控应用程序。

[0145] 在图13的变型中,除了在执行图形控制流应用程序之前配置断点之外或者代替在执行图形控制流应用程序之前配置断点,断点的配置(例如,设置)可以在图形控制流应用程序正在执行时发生。

[0146] 在示例中,本文描述的系统可以被配置用于单步通过断点。单步(single-step)是控件或其他命令,当遇到断点时,该控件或其他命令指示系统转到下一个可以中断计划执行的位置,而不必在此处显式设置断点。在这个示例中,用户指示系统单步通过计划。在这个示例中,系统(或用户)将指定将单步通过的给定级别(例如,任务级或方法级)。在这个示例中,计划可以具有多条控制流路径,从而产生一个以上的“下一个”位置。因此,系统将在所有这些位置处停止执行。

[0147] 参考图14A,图示450示出了计划452——在部分461a中定义和在部分461b中执行两者。在这个示例中,计划452包括表示相应任务的图形程序元素454a……454n。为了方便读者,每当参考指示图形程序元素或转换的附图标记时,在描述相应的图14A至图14B时分别直接参考相应的任务和控制流。因此,图形程序元素454a……454n在下文中被称为任务454a……454n。

[0148] 任务454a……454n中的每一个包括一种或多种方法,每种方法包括要在计算机系统上执行的一个或多个动作。例如,任务454a包括方法456并且任务454n包括方法458。方法456可以是与方法458相同的方法或不同的方法。在这个示例中,任务454a……454n中的每一个从计划452继承所继承的方法460。通常,继承的方法包括在计划级别定义一次并且应用于计划内的任务(由计划内的任务继承)的方法。也就是说,继承的方法是在计划(或子计划)级别定义的、应用于该级别以下的任务的动作。通过使用继承的方法,系统可以将相同的方法应用于计划中的任务。继承的方法是为整个计划定义一次,而不是为每个任务单独定义一次,从而使得能够一次为许多任务分派相同的操作。如图示450的部分461b所示,当系统执行计划452时,继承的方法460在计划452内的每个任务中执行,即在计划452中的每个任务454a……454n中执行。在一些示例中,当任务包括继承的方法和非继承的方法两者时,系统根据这些方法是出现在任务的执行方法之前还是之后以不同的顺序执行这些方法。在任务的执行方法之前,系统首先执行继承的方法,然后执行该特定任务的类似的非继承的方法。在任务的执行方法之后,系统首先执行非继承的方法,然后执行类似的继承的方法。

[0149] 参考图14B,图示470展示了计划调试器使得能够在继承的方法上设置断点。具体地,图示470展示了计划452——在部分471a中定义和在部分471b中执行两者。继承的方法

460在计划级别定义一次(在部分471a中示出)。在执行(在部分471b中示出)期间,计划452内的每个元素(即,每个任务454a……454n)将通过为该特定元素创建继承的方法460的特定实例(即,实例460a……460n)来继承该方法定义。在这个示例中,针对继承的方法460的实例,为每个任务(例如,任务454a……454n中的一个或多个)设置或记录断点。也就是说,可以仅在继承的方法的特定实例上设置断点。在这个示例中,系统通过分别在位置461a、461b、461c之一处记录断点来在任务454a的继承的方法460的实例460a之前、之中或之后记录断点。在这个示例中,系统可以在位置461a、461b、461c中的一个或多个位置处记录断点(尽管出于完整性的目的在图14B中示出了所有位置461a、461b、461c)。

[0150] 参考图15,示出了应用开发和执行系统500的框图。本文描述的调试技术可以在系统500内实施。系统500包括两个图形开发环境,即用于数据流图开发的GDE 512和用于控制流图开发的GDE 514。可替代地,一个图形开发环境可以用于开发数据流图和控制流图两者。或者图形开发环境可以用于开发控制流图并且命令行用户界面可以用于数据流图,或者反之亦然。

[0151] 使用GDE 512,构建包括数据流图580的数据处理应用程序518。使用GDE 514,构建包括控制流图550的任务管理应用程序516。

[0152] 系统500还包括数据库520。数据库520可以是可扩展的面向对象的数据库系统,其为系统500提供各种种类的信息(例如,元数据)的存储。数据库520可以是例如企业元数据数据库,其可以支持基于图的应用程序的开发和执行以及基于图的应用程序与其他系统(例如,操作系统)之间的数据交换。

[0153] 系统500进一步包括操作系统524。操作系统524可以是例如并行操作环境。操作系统524为运行比如GDE 512和GDE 514等应用程序开发环境提供支持,并提供所开发的应用程序的可扩展的并行和分布式执行。

[0154] 在图15中,当数据被数据处理应用程序518处理时,输入数据流530通过数据流图580。数据流图580包括计算组件,即重新格式化组件584。在数据流图580中,数据从输入数据集582流向输出数据集586。

[0155] 在被重新格式化之后,数据流出数据处理应用程序518并进入任务管理应用程序516中,并用于驱动任务管理应用程序516,该任务管理应用程序进而驱动控制流图550。在示例中,如果数据处理应用程序518成功,则任务调度应用程序516将启动控制流图550。控制流图550示出了两个任务,即任务552和任务554,如本文所述,每个任务可以编程有断点(在任务之前、之中或之后)。任务可以是例如通过执行数据流图或脚本(比如Perl脚本)来执行的计算。时间序列556示出了控制流图550中指定的任务的运行顺序。在这种情况下,任务552在任务554之前执行。

[0156] 如图15所示,控制流图550中的任务552是Perl脚本,并且任务554包括可以在本文描述的系统的操作系统524中执行的若干方法。这些方法可以是由本文描述的系统提供的现有方法。用户可以通过针对组成方法的各种参数和脚本指定这些方法自己的内容来自定义这些方法的行为。如本文所述,可以在这些方法中的每种方法之前、之中或之后插入断点。如图15所示,作为示例,任务554包括五种方法:触发方法、开始时方法、执行方法、失败时方法和成功时方法。

[0157] 在一些示例中,以上五种方法可以如下实施。

[0158] 触发方法可以被实施为表示任务554的起点。该触发方法可以包括用于开始执行的各种条件,包括例如特定文件是否存在、标志是否已经被设置为真等。

[0159] 开始时方法可以被实施为使系统为执行方法做好准备的方法,比如将环境变量设置为期望值、或者设置日志文件来记录运行时信息。

[0160] 执行方法可以被实施为执行任务554的主要功能。任务554还可以包含用于处理执行方法之后发生的情况的条件逻辑。如果执行方法在其执行期间成功,则执行成功时方法以退出任务554,其中返回代码为零。如果执行方法在其执行期间失败,则执行失败时方法以退出任务554,其中返回非零代码。可选地,可以为回滚、错误处理和恢复添加额外的方法。例如,可以添加回滚方法,以从失败点开始以相反的执行顺序回滚已经进行的工作。可替代地,可以添加清理方法,以通过重置标志、寄存器等来清理失败的条件。

[0161] 如本文所述,功能模块封装用于或表示受控应用程序的信息,其中,该受控应用程序是可执行计算机程序的实例或计算机程序的可执行部分,并且其中,该功能模块被配置用于在计算机系统上执行一个或多个动作。控制流是各个功能模块被执行或评估的顺序。

[0162] 使用本文描述的技术,系统和方法调试计划和子计划。例如,本文描述了计划调试器以对这些计划进行故障排除——该调试器既有命令行又有图形界面。该计划调试器使得计划的执行能够在这些计划内的许多关键点处暂停(或“中断”)。该计划在异构机器和技术上以分布式方式操作。该调试器提供调试和调试器功能,比如条件断点、或在循环的特定迭代上的中断、或故障时中断。该调试器使得能够在计划的不同级别处中断:计划、子计划、任务或方法。可以在以上任何一项的执行(a)之前、(b)之后、或者(c)之中(对于任务和方法的情况)和在以上任何一项的执行内(对于作为图和子计划的受控应用程序的情况)设置中断,除非在此注明。如本文所述,设置和运行断点不会降低计划的速度。另外,使用该调试器不需要将计划置于“调试模式”。用户可以在开始计划之前或者在计划正在运行时设置断点。

[0163] 本文描述的技术提供了大量的计算和资源分配效率。例如,通过能够在任务之前、之中和之后以及在任务的每种方法之前、之中和之后设置断点,系统能够减少调试任务和/或其方法所需的任务的执行次数。这是因为本文描述的技术不是等待任务或计划运行到完成来识别故障,而是在故障发生时就提供故障识别并提供修改可能导致故障的变量和/或参数值的能力。一旦修改了这些值,计划或任务就可以从先前失败的点恢复执行。也就是说,“下游”任务和/或方法(依赖于另一个任务和/或方法的任务和/或方法)一旦故障已经被识别和解决就被执行,而不是在故障之后被立即执行并且然后在故障被调试时被重复执行。相对于当计划被允许运行到完成并且然后在完成后被调试并且然后再次运行以评估故障是否已经被成功解决时所需的处理能力的量,这使得处理能力的消耗减少(因为调试计划所需的计划执行数更少)。这也使得内存消耗减少,因为相对于多次运行计划以调试计划所需的内存量,计划可能只运行一次并在该单次运行期间调试成功。

[0164] 另外,能够在任务的方法中的每种方法之前、之中和之后设置断点所提供的粒度级别在定义表示任务的方法的数据结构方面提供了更大的灵活性并在调查和控制执行行为方面提供了更大的灵活性。不是必须预先定义这些数据结构并仅在计划已经执行之后改变其值,而是可以在计划正在被执行时“即时(on-the-fly)”修改这些数据结构的参数值和/或值。这些即时修改提供较小的内存需求,因为计划不需要被执行多次来评估这些改变

的影响。

[0165] 上述技术可以使用软件来实施以便在计算机上执行。例如,软件形成一个或多个计算机程序中的在一个或多个编程或可编程计算机系统(可以是比如分布式、客户端/服务器或网格等各种架构)上执行的程序,每个计算机系统包括至少一个处理器、至少一个数据存储系统(包括易失性和非易失性存储器和/或存储元件)、至少一个输入设备或端口以及至少一个输出设备或端口。软件可以形成较大程序的一个或多个模块,例如,提供与图表和流程图的设计和配置相关的其他服务。图表的节点、链路和元素可以被实施为存储在计算机可读介质中的数据结构或者符合存储在数据存储库中的数据模型的其他有组织的数据。

[0166] 本文描述的技术可以在数字电子电路系统中实施,或者在计算机硬件、固件、软件或其组合中实施。装置可以在有形地体现或存储在机器可读存储设备(例如,非暂态机器可读存储设备、机器可读硬件存储设备等)中的计算机程序产品中实施,以供可编程处理器执行;并且方法动作可以由执行指令程序的可编程处理器来执行,以通过对输入数据进行操作并生成输出来执行功能。本文所描述的实施例和权利要求的其他实施例以及本文所描述的技术可以有利地在可编程系统上可执行的一个或多个计算机程序中实施,该可编程系统包括至少一个可编程处理器,该可编程处理器被耦接以从数据存储系统、至少一个输入设备以及至少一个输出设备中接收数据和指令并向数据存储系统、至少一个输入设备以及至少一个输出设备传输数据和指令。每个计算机程序可以用高级程序或面向对象的编程语言来实施,或者如果需要的话用汇编语言或机器语言来实施;并且在任何情况下,语言可以是编译语言或解释语言。

[0167] 适合于执行计算机程序的处理器举例来讲包括通用微处理器和专用微处理器两者、以及任何种类的数字计算机的任何一个或多个处理器。通常,处理器将从只读存储器或随机存取存储器或两者接收指令和数据。计算机必不可少的元件是用于执行指令的处理器和用于存储指令和数据的一个或多个存储器设备。通常,计算机还将包括用于存储数据的一个或多个大容量存储设备(例如,磁盘、磁光盘或者光盘),或者被可操作地耦接以从用于存储数据的一个或多个大容量存储设备接收数据或者向其传送数据,或者既接收又传递数据。用于体现计算机程序指令和数据的计算机可读介质包括所有形式的非易失性存储器,举例来说,包括半导体存储器设备,例如,EPROM、EEPROM和闪存设备;磁盘,例如,内部硬盘或可移动磁盘;磁光盘;以及CD ROM和DVD ROM盘。处理器和存储器可以由专用逻辑电路系统来补充或并入专用逻辑电路系统。上述任何一项都可以由ASIC(专用集成电路)补充或并入ASIC。

[0168] 为了提供与用户的交互,实施例可以在具有用于向用户显示信息的显示设备(例如,LCD(液晶显示器)监视器)和用户可以通过其向计算机提供输入的键盘和指点设备(例如,鼠标或轨迹球)的计算机上实施。其他种类的设备也可以用于提供与用户的交互;举例来说,提供给用户的反馈可以是任何形式的传感反馈,例如,视觉反馈、听觉反馈或触觉反馈;并且可以用任何形式接收来自用户的输入,包括声音输入、语音输入、或触觉输入。

[0169] 实施例可以在计算系统中实施,该计算系统包括后端组件(例如,作为数据服务器),或者包括中间件组件(例如,应用服务器),或者包括前端组件(例如,具有图形用户界面或Web浏览器的客户端计算机,用户可以通过该图形用户界面或Web浏览器与实施例的实施进行交互),或者这些后端、中间件或前端组件的任何组合。系统的组件可以通过数字数

据通信(例如,通信网络)的任何形式或介质来进行互连。通信网络的示例包括局域网(LAN)和广域网(WAN)(例如,互联网)。

[0170] 该系统和方法或其部分可以使用互联网上利用超文本传输协议(HTTP)的服务器集合——“万维网”(Web或WWW)。HTTP是已知的应用协议,其为用户提供对资源的访问权,资源可以是不同格式的信息,比如文本、图形、图像、声音、视频、超文本标记语言(HTML)以及程序。在用户指定链路后,客户端计算机向Web服务器发出TCP/IP请求并接收信息,该信息可以是根据HTML格式化的另一个Web页。用户也可以通过遵循屏幕上的指令、输入特定数据或点击所选图标来访问相同或其他服务器上的其他页。还应当注意,本领域技术人员已知的任何类型的选择设备,比如复选框、下拉框等,可以用于使用web页来允许用户为给定组件选择选项的实施例。服务器在各种平台上运行,包括UNIX机器,但也可以使用其他平台,比如Windows2000/2003、Windows NT、Sun、Linux和Macintosh。计算机用户可以通过使用浏览软件(比如Firefox、Netscape Navigator、Microsoft Internet Explorer或Mosaic浏览器)在Web上查看服务器或网络上可用的信息。计算系统可以包括客户端和服务器。客户端和服务器通常远离彼此并且通常通过通信网络交互。客户端与服务器的关系借助于在相应的计算机上运行并且彼此具有客户端-服务器关系的计算机程序产生。

[0171] 其他实施例在说明书和权利要求的范围和精神内。例如,由于软件的性质,上文描述的功能可以使用软件、硬件、固件、硬连线或任何这些的组合来实施。实现功能的特征也可以物理地位于各种位置,包括被分布使得功能的各部分在不同的物理位置处实施。术语“a(一个)”在本文和整个申请中的使用不是以限制的方式使用的,并且因此不意味着排除术语“a(一个)”的多个含义或“一个或多个”含义。另外,就要求对临时专利申请的优先权而言,应当理解,临时专利申请不是限制性的,而是包括可以如何实施本文描述的技术的示例。

[0172] 已经描述了本发明的多个实施例。然而,本领域普通技术人员将会理解,在不脱离权利要求和本文描述的技术的精神和范围的情况下,可以进行各种修改。

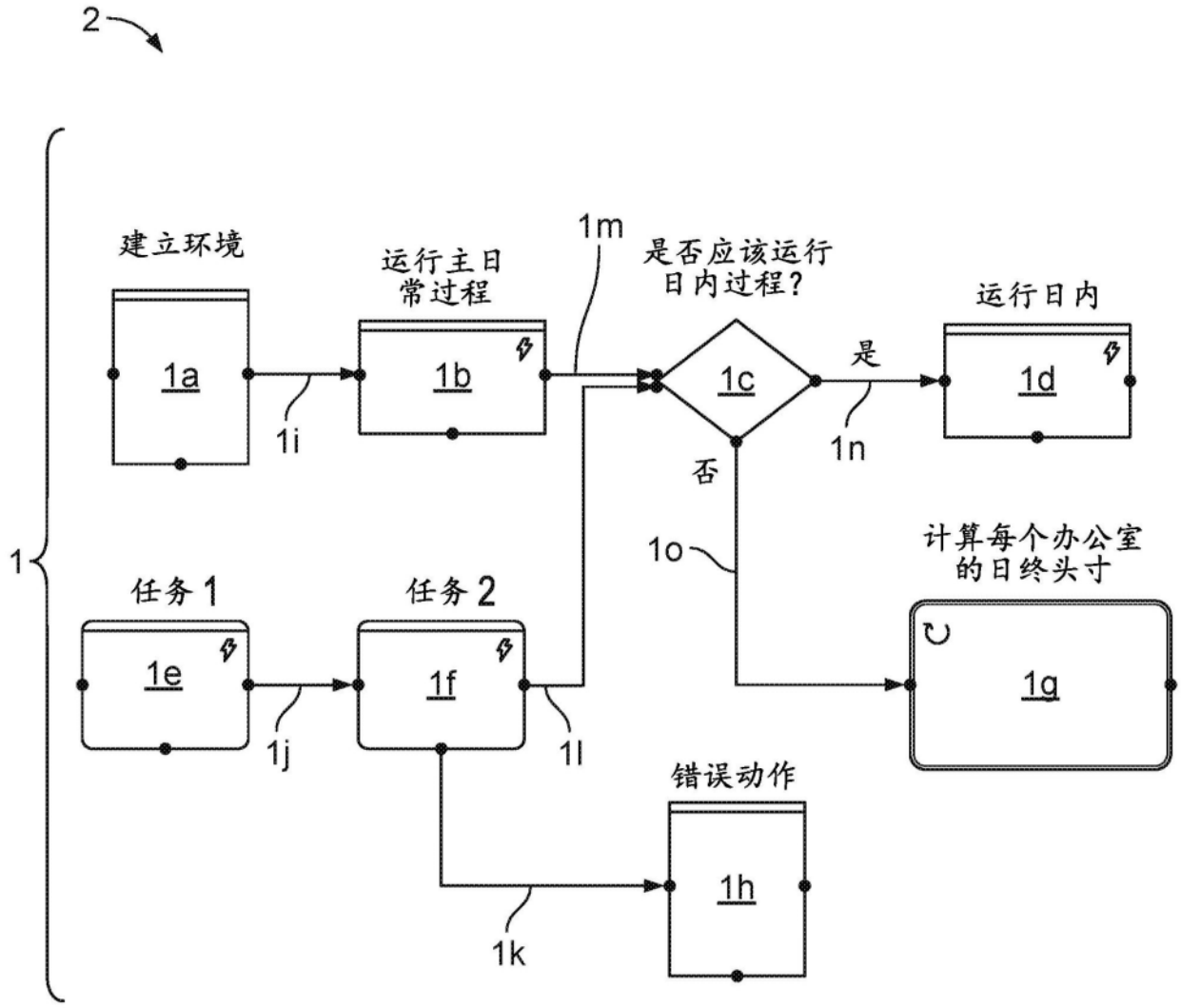

2
1
建立环境
1a
1i
运行主日常过程
1b
1m
是否应该运行日内过程?
1c
是
1n
运行日内
1d
否
1o
计算每个办公室的日终头寸
1g
任务 1
1e
1j
任务 2
1f
1l
错误动作
1h
1k

图1A

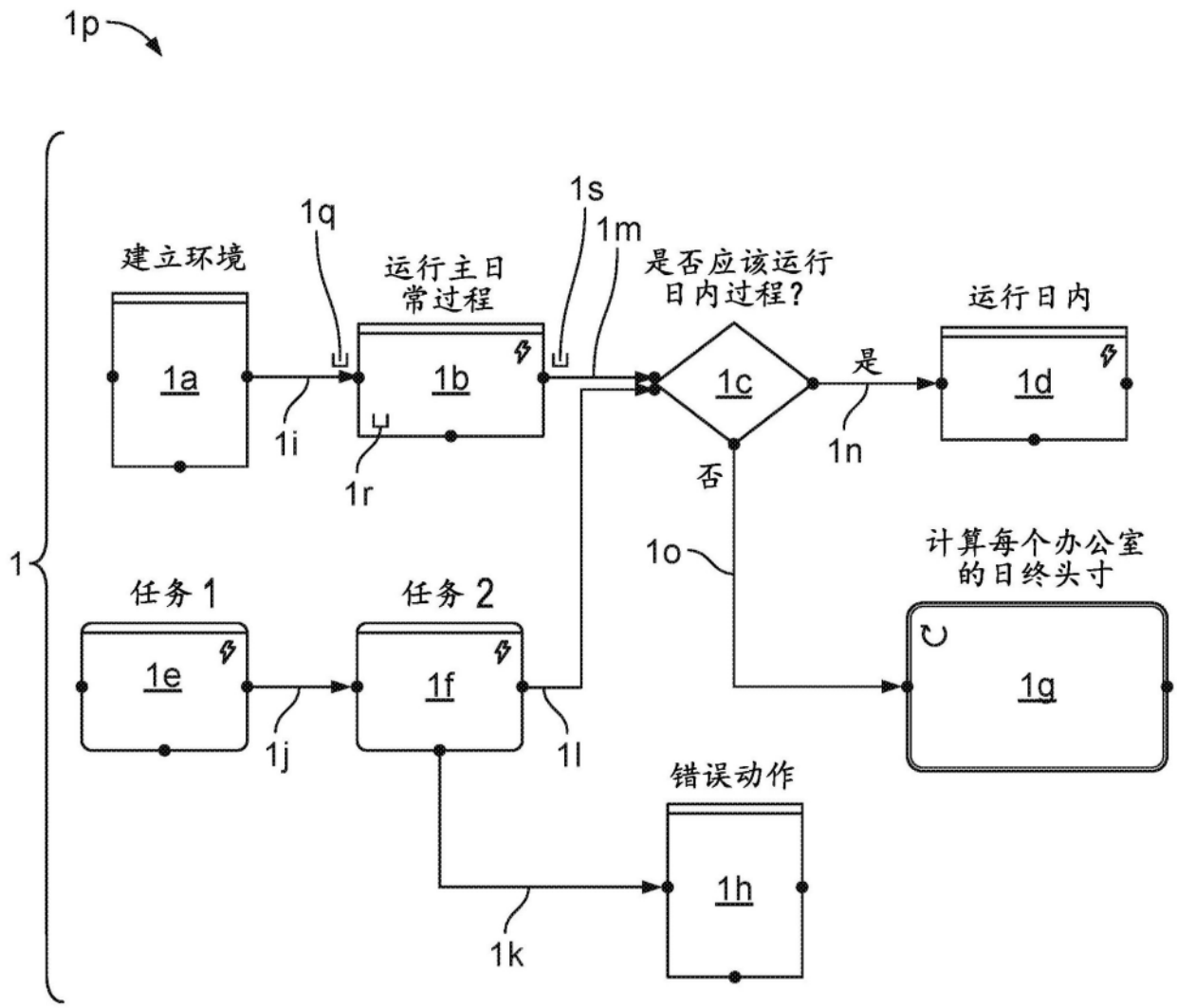

1p
1
建立环境
1a
1i
1q
运行主日常过程
1b
1r
1s
1m
是否应该运行日内过程?
1c
是
1n
运行日内
1d
否
1o
计算每个办公室的日终头寸
1g
任务 1
1e
1j
任务 2
1f
1l
错误动作
1h
1k

图1B

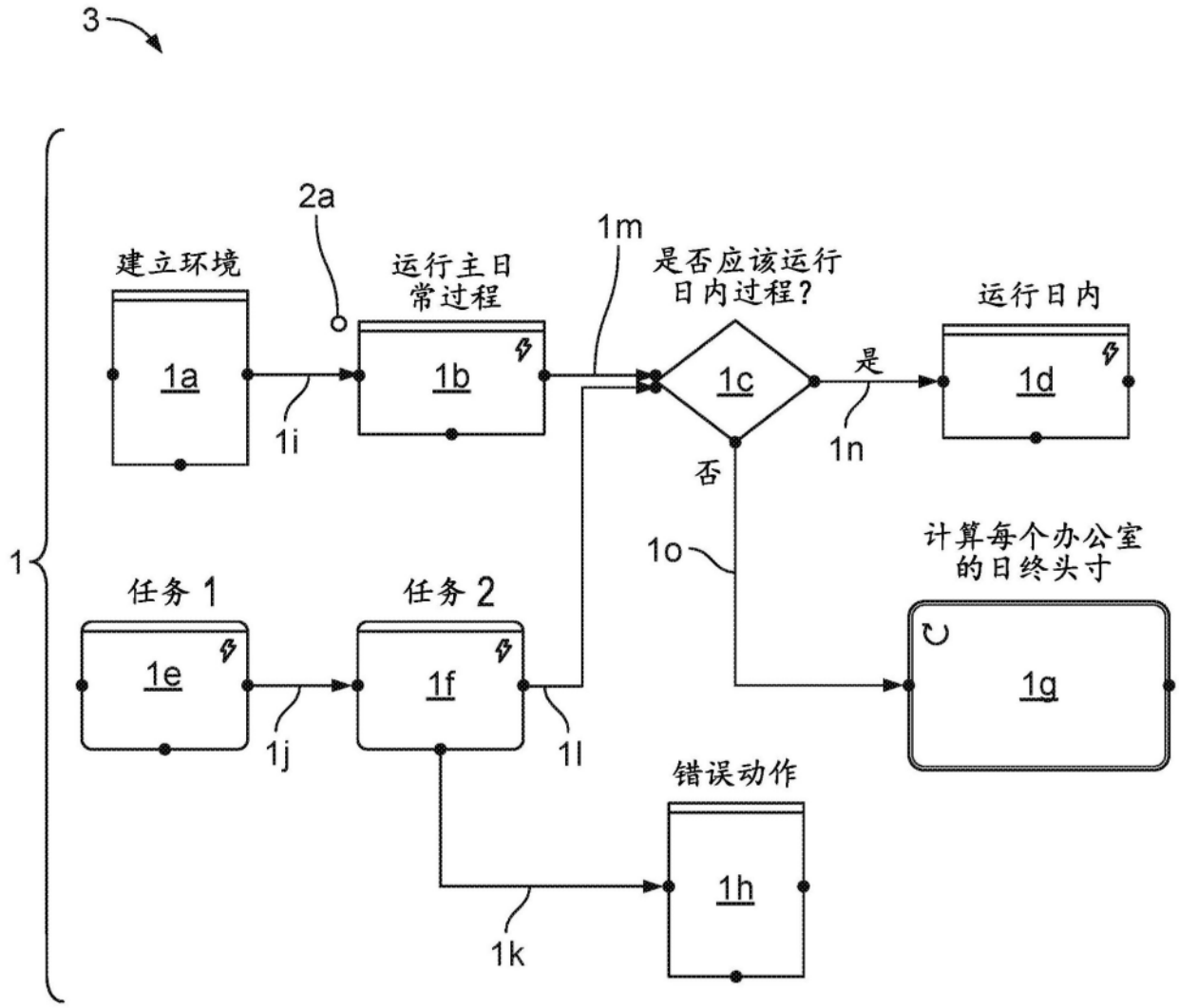

3
1
2a
建立环境
1a
1i
运行主日
常过程
1b
1m
是否应该运行
日内过程?
1c
是
1n
运行日内
1d
否
1o
计算每个办公室
的日终头寸
1g
任务 1
1e
1j
任务 2
1f
1l
错误动作
1h
1k

图1C

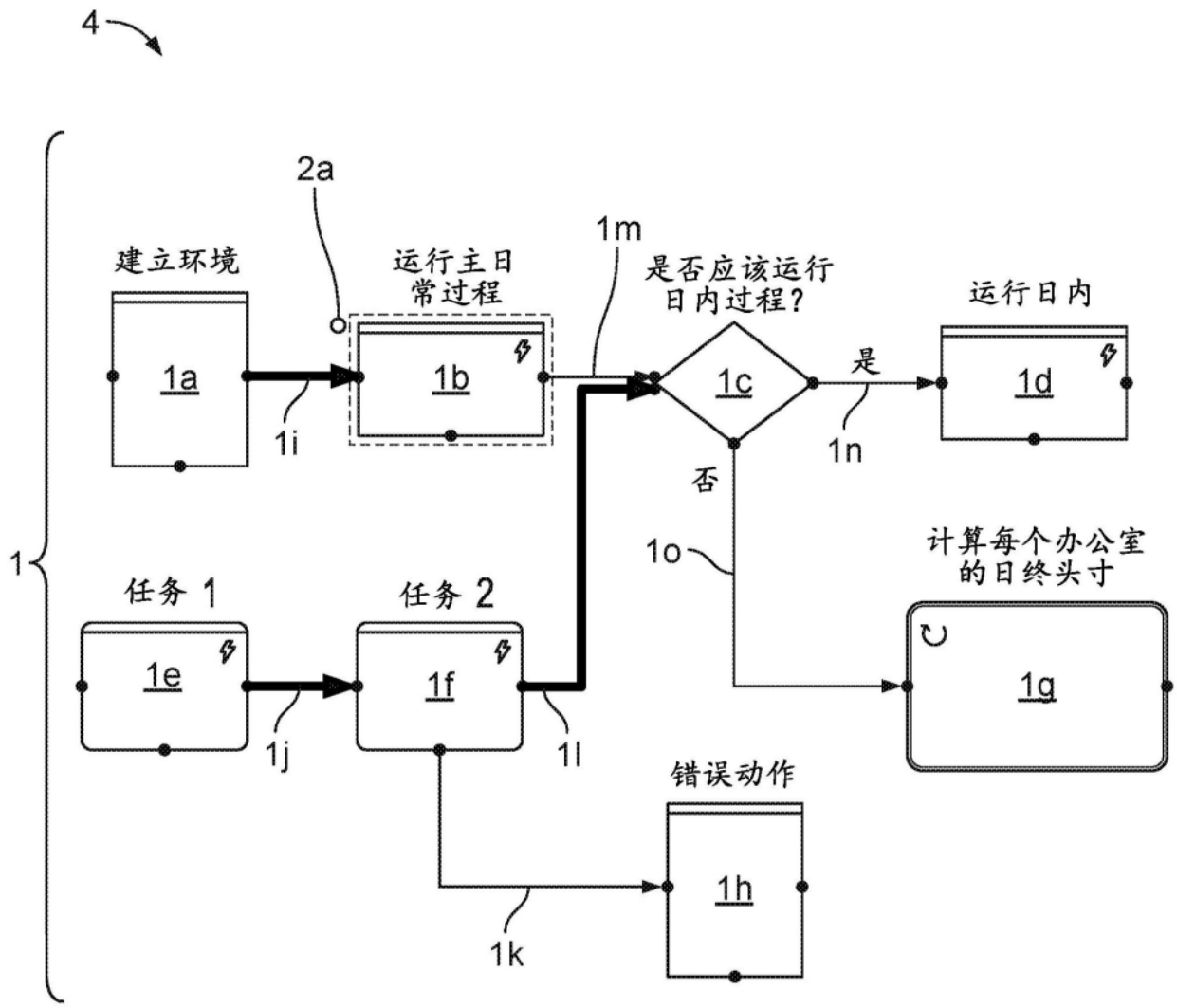

4
1
2a
建立环境
1a
1i
运行主日
常过程
1b
1m
是否应该运行
日内过程?
1c
是
1n
运行日内
1d
否
1o
计算每个办公室
的日终头寸
1g
任务 1
1e
1j
任务 2
1f
1l
错误动作
1h
1k

图1D

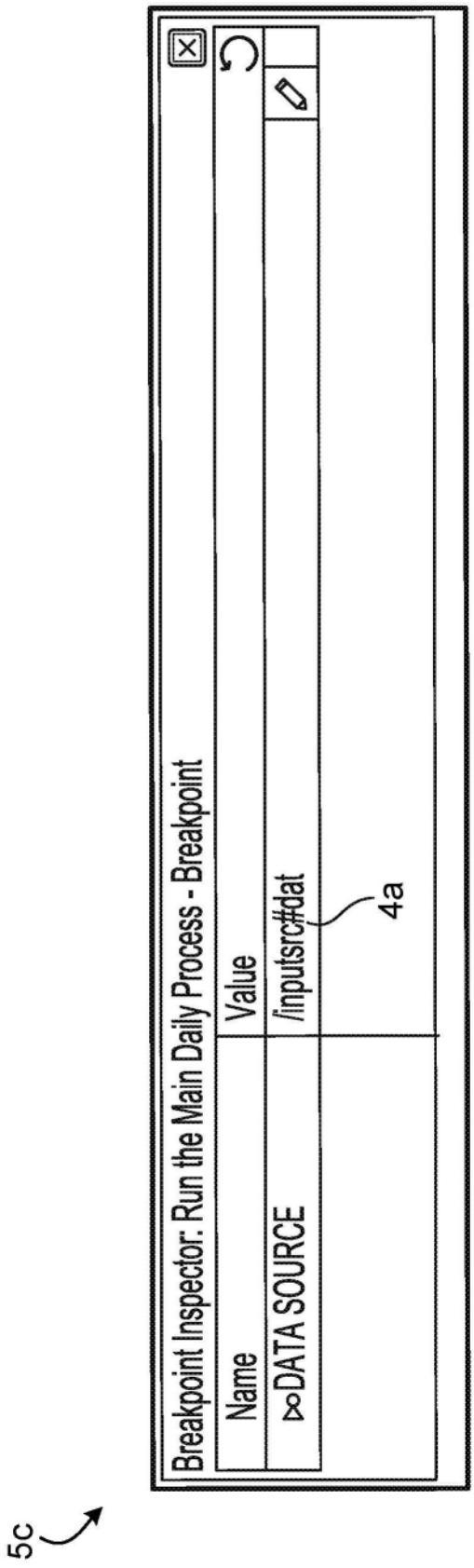

Breakpoint Inspector: Run the Main Daily Process - Breakpoint
Name
Value
⋈DATA SOURCE
/inputsrc#dat
4a
5c

图1E

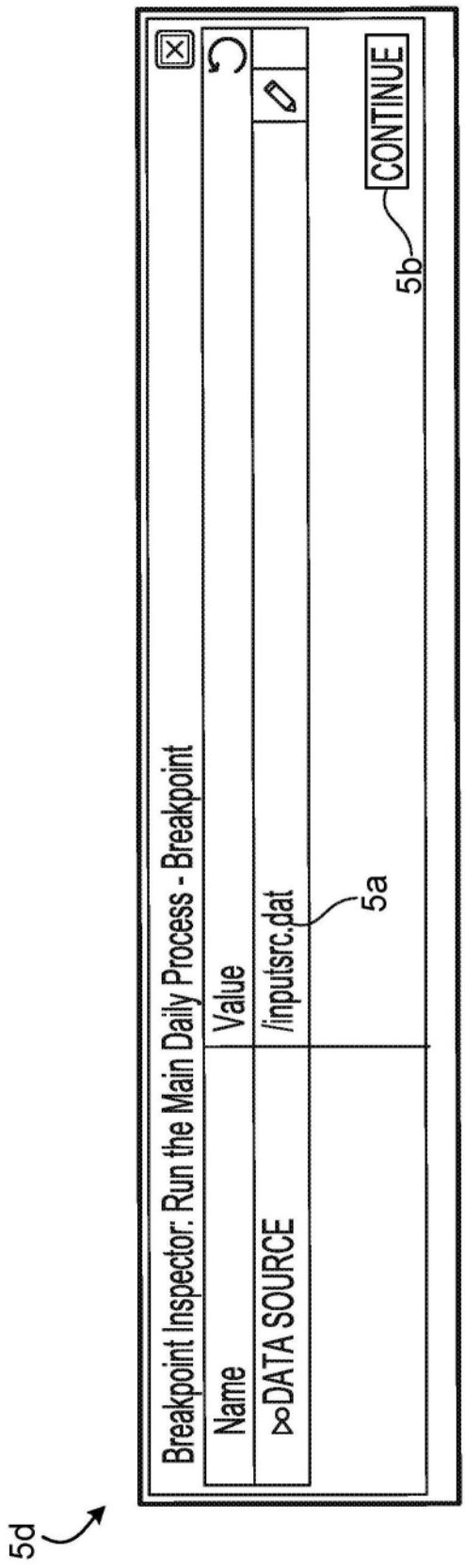

Breakpoint Inspector: Run the Main Daily Process - Breakpoint
Name
Value
DATA SOURCE
/inputsrc.dat
CONTINUE
5a
5b
5d

图1F

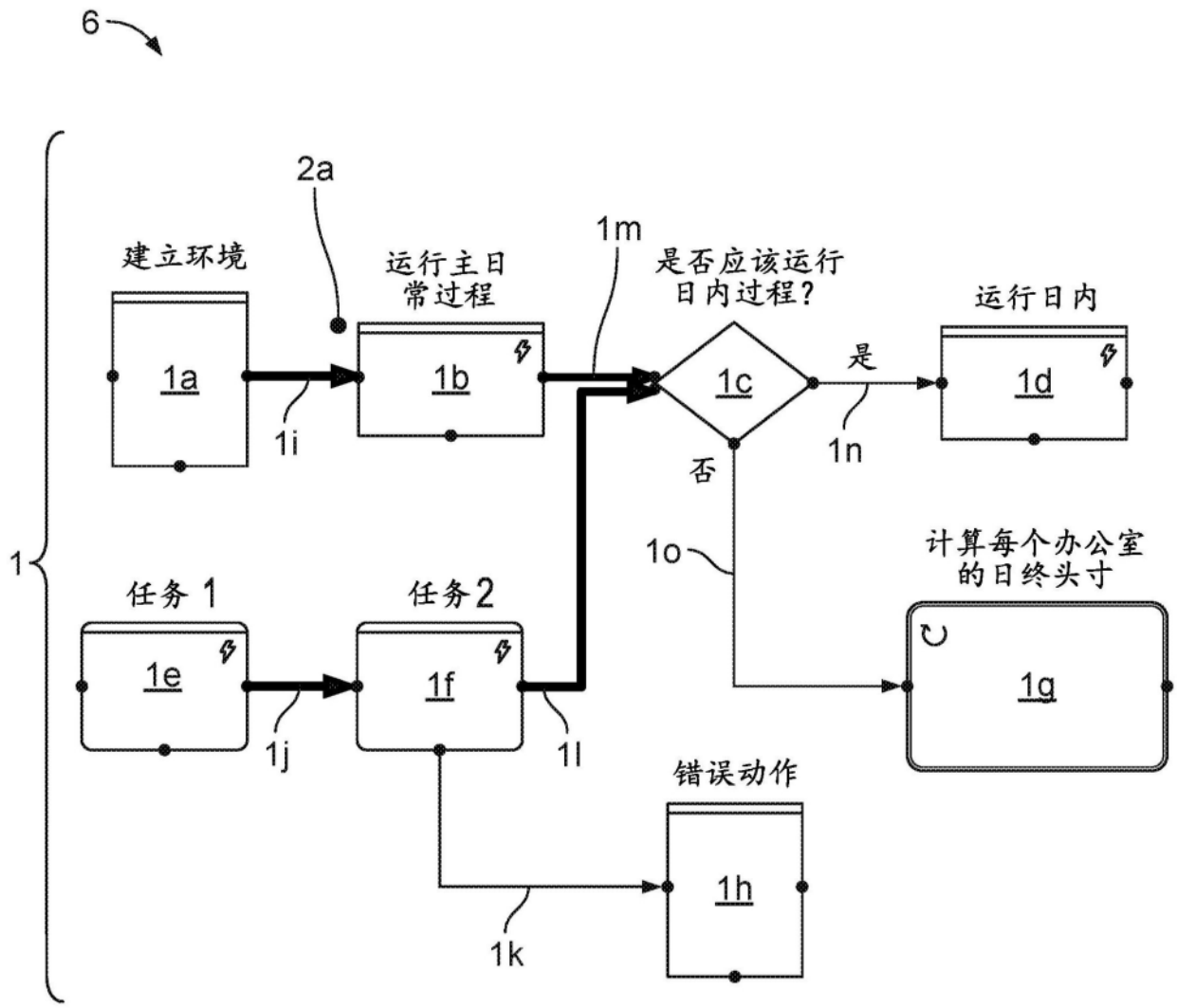

6
1
2a
建立环境
1a
1i
运行主日常过程
1b
1m
是否应该运行日内过程?
1c
是
1n
运行日内
1d
否
1o
计算每个办公室的日终头寸
1g
任务 1
1e
1j
任务2
1f
1l
错误动作
1h
1k

图1G

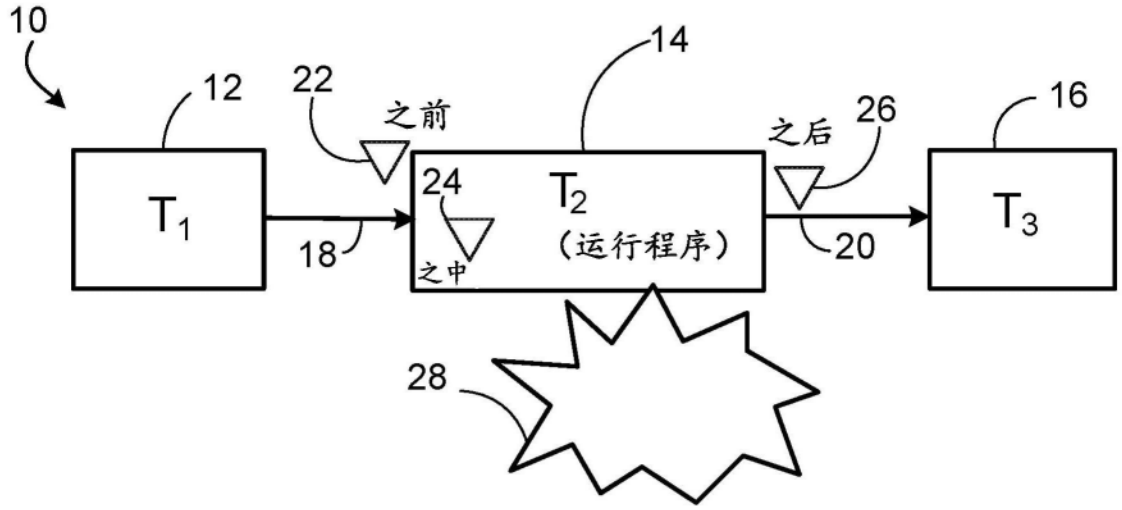

10
12
T1
22
之前
18
14
24
之中
T2
(运行程序)
之后
26
20
16
T3
28

图2

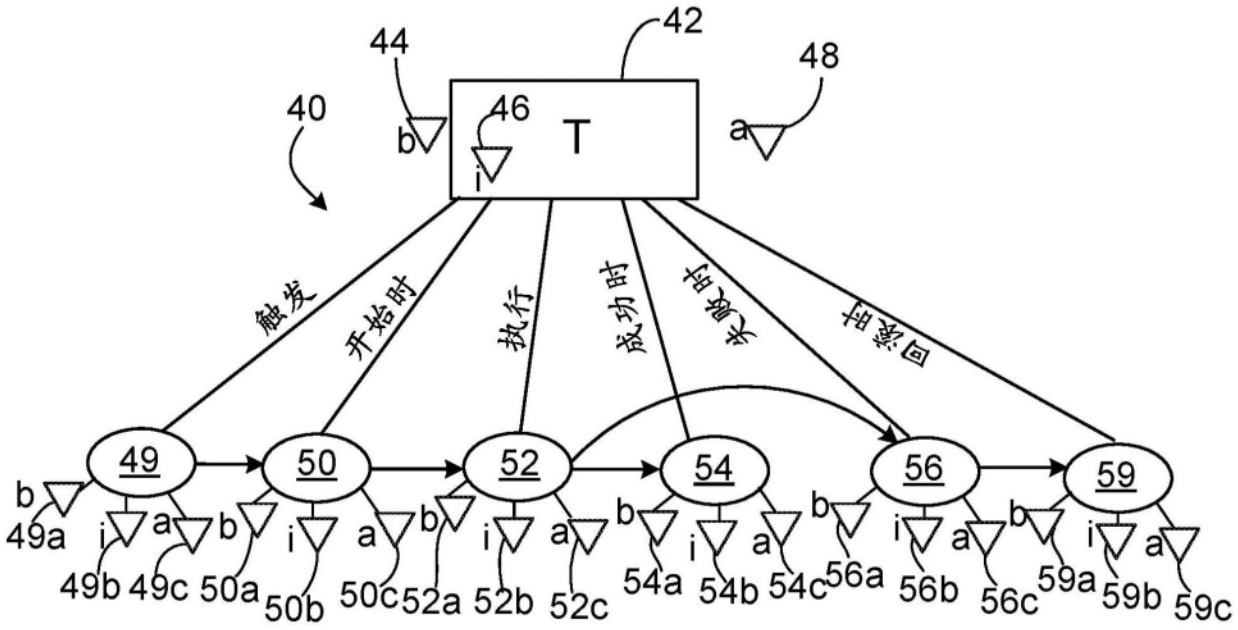

42
44
48
40
46
T
b
i
a
触发
开始时
执行
成功时
失败时
回滚
49
50
52
54
56
59
b
49a
i
a
49b
49c
b
50a
i
50b
a
50c
b
52a
i
52b
a
52c
b
54a
i
54b
a
54c
b
56a
i
56b
a
56c
b
59a
i
59b
a
59c

图3

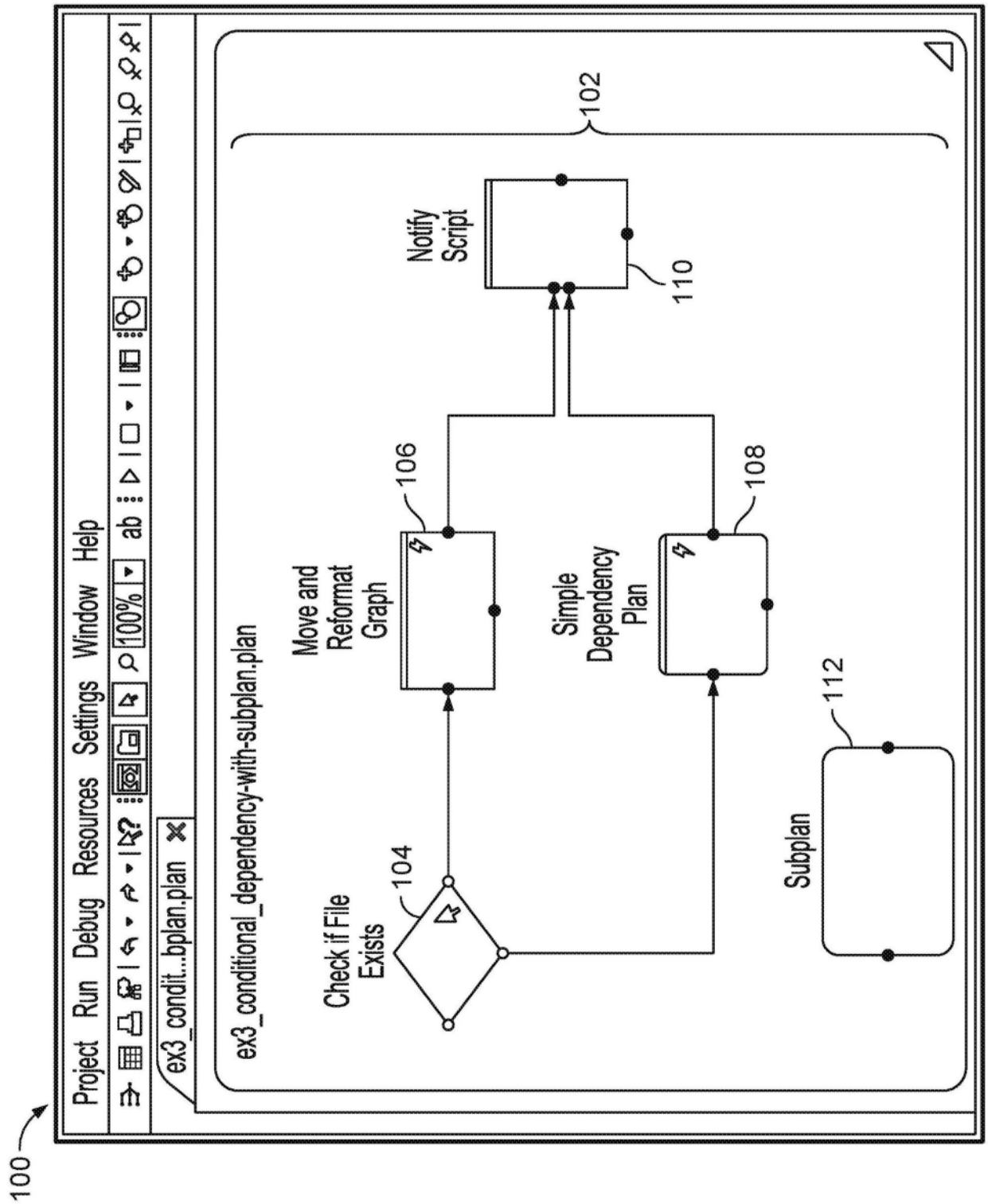

100
Project Run Debug Resources Settings Window Help
ex3_condit...bplan.plan
ex3_conditional_dependency-with-subplan.plan
102
Notify Script
110
Move and Reformat Graph
106
Simple Dependency Plan
108
Check if File Exists
104
Subplan
112

图4

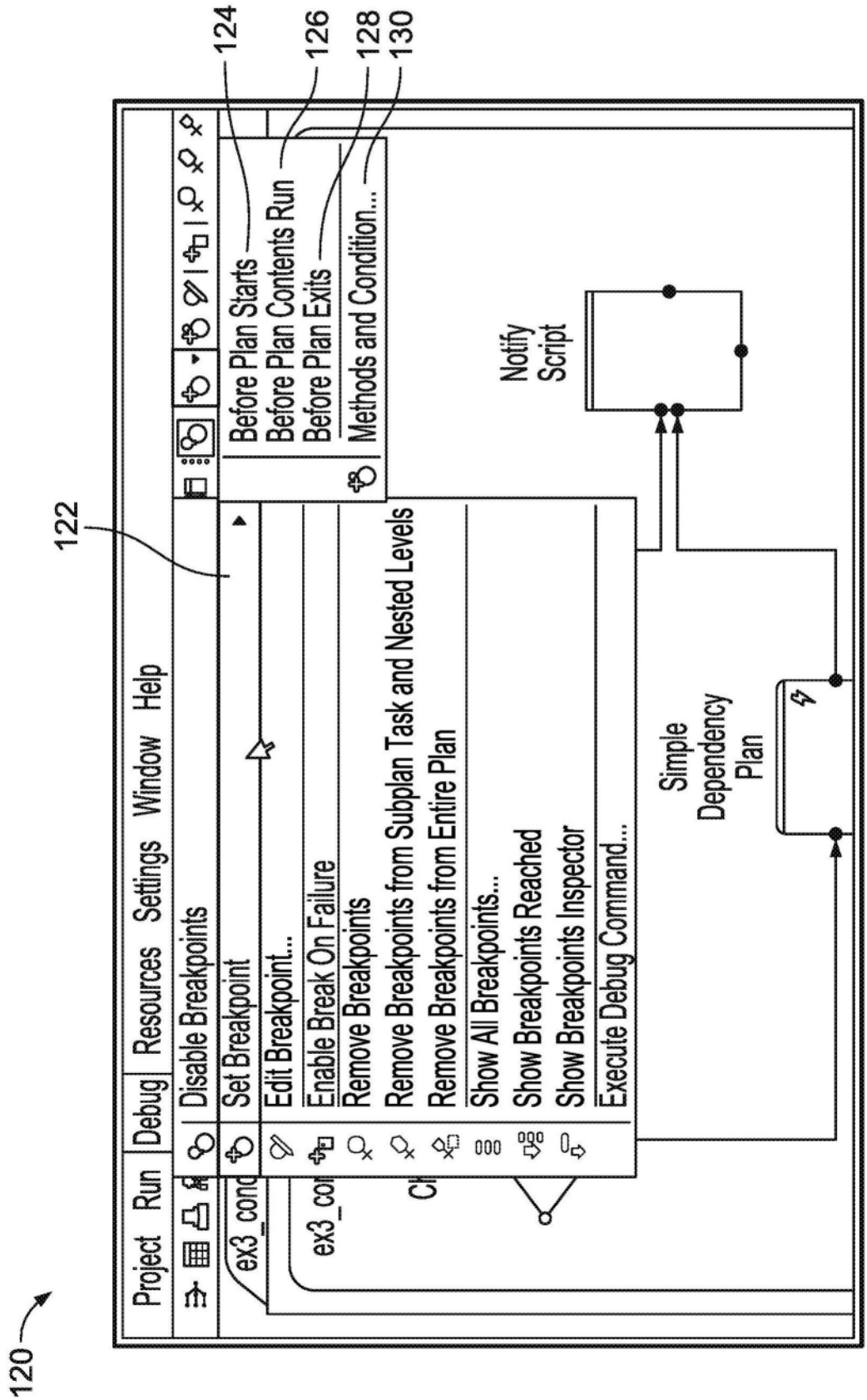

120
122
124
126
128
130
Project Run Debug Resources Settings Window Help
Disable Breakpoints
Set Breakpoint
Edit Breakpoint...
Enable Break On Failure
Remove Breakpoints
Remove Breakpoints from Subplan Task and Nested Levels
Remove Breakpoints from Entire Plan
Show All Breakpoints...
Show Breakpoints Reached
Show Breakpoints Inspector
Execute Debug Command...
Before Plan Starts
Before Plan Contents Run
Before Plan Exits
Methods and Condition...
Notify Script
Simple Dependency Plan

图5

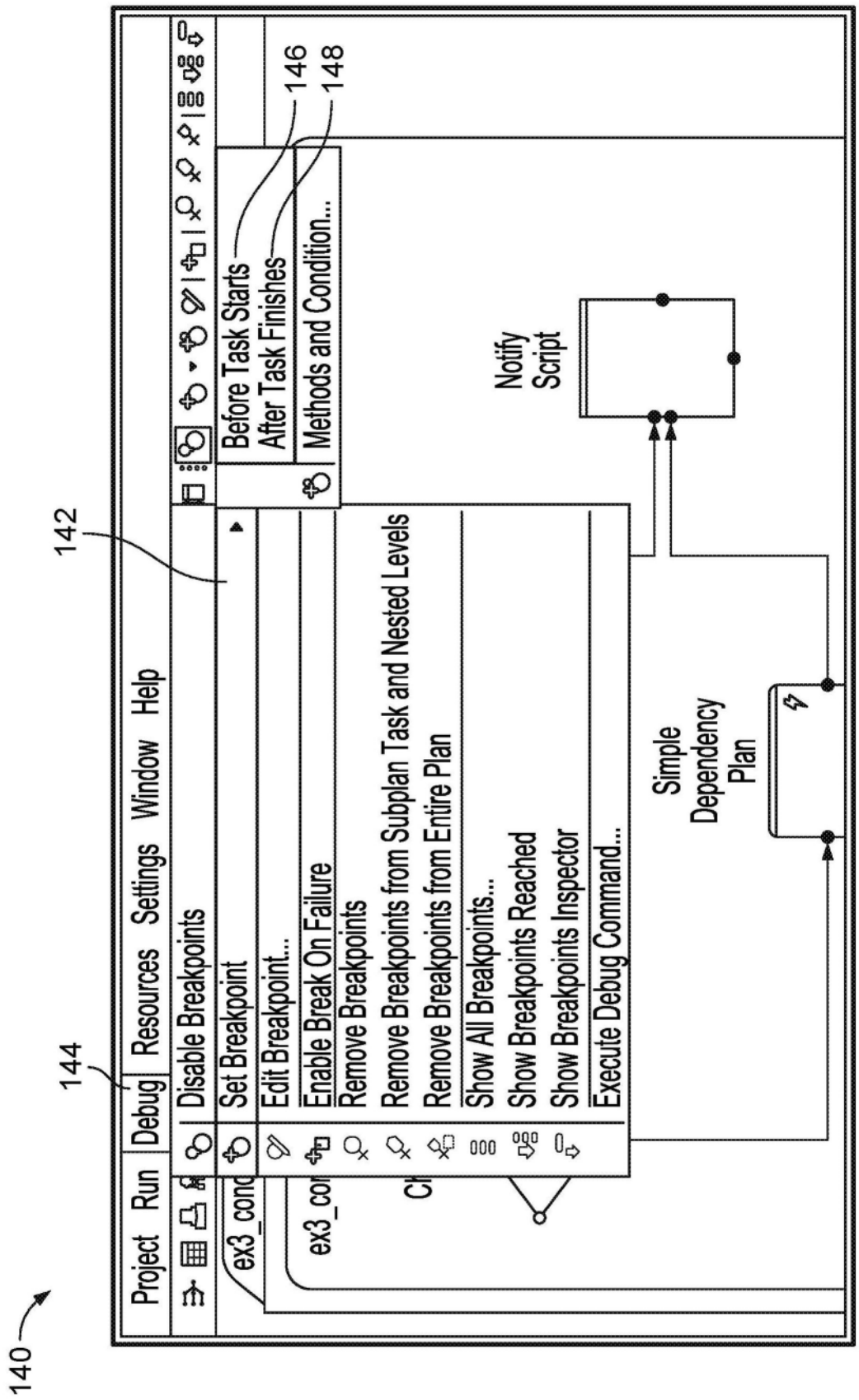

140
142
144
146
148
Project
Run
Debug
Resources
Settings
Window
Help
Disable Breakpoints
Set Breakpoint
Edit Breakpoint...
Enable Break On Failure
Remove Breakpoints
Remove Breakpoints from Subplan Task and Nested Levels
Remove Breakpoints from Entire Plan
Show All Breakpoints...
Show Breakpoints Reached
Show Breakpoints Inspector
Execute Debug Command...
Before Task Starts
After Task Finishes
Methods and Condition...
Notify Script
Simple Dependency Plan

图6

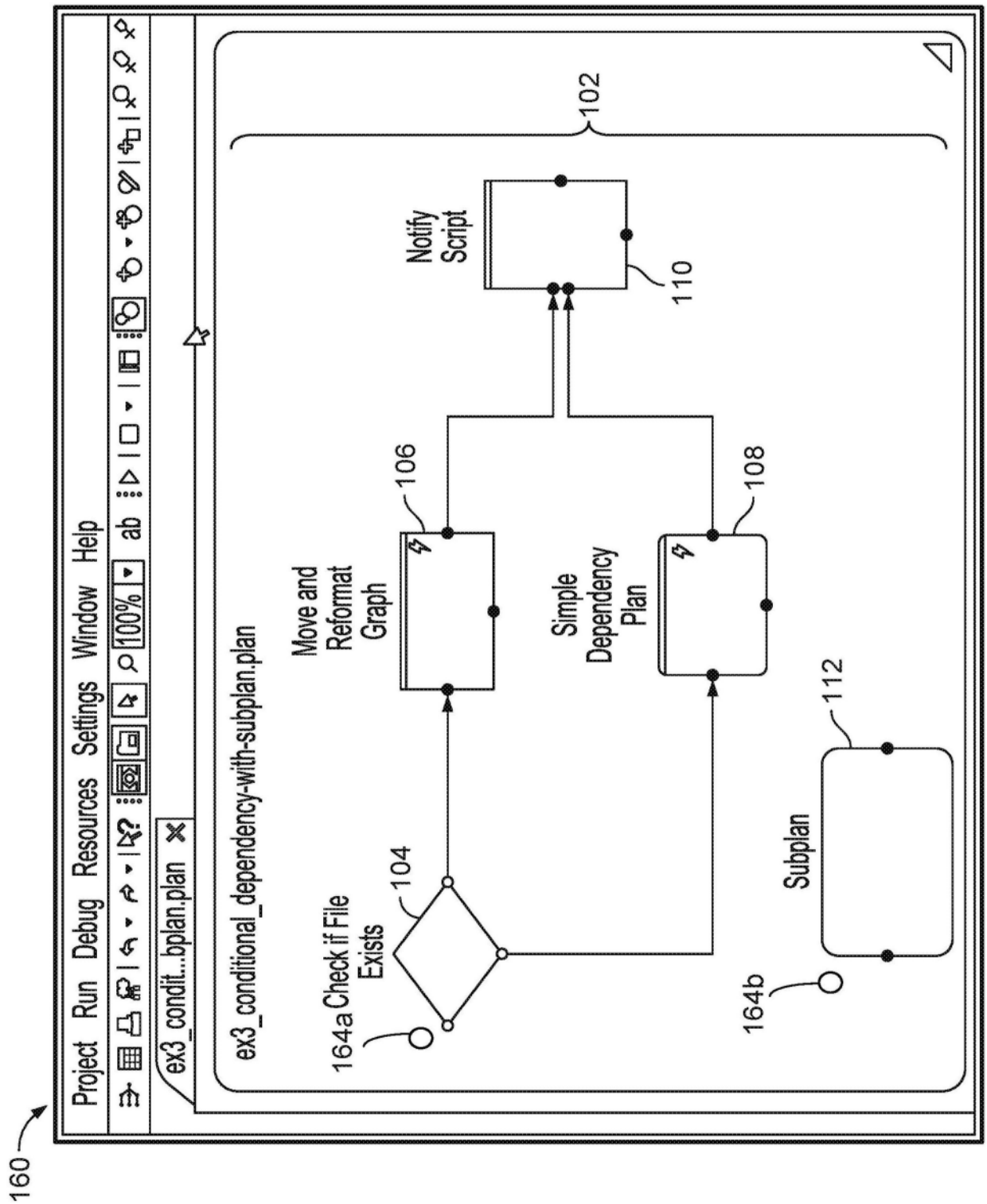

160
Project Run Debug Resources Settings Window Help
ex3_condit...bplan.plan
ex3_conditional_dependency-with-subplan.plan
102
Notify Script
110
Move and Reformat Graph
106
Simple Dependency Plan
108
164a Check if File Exists
104
Subplan
112
164b

图7A

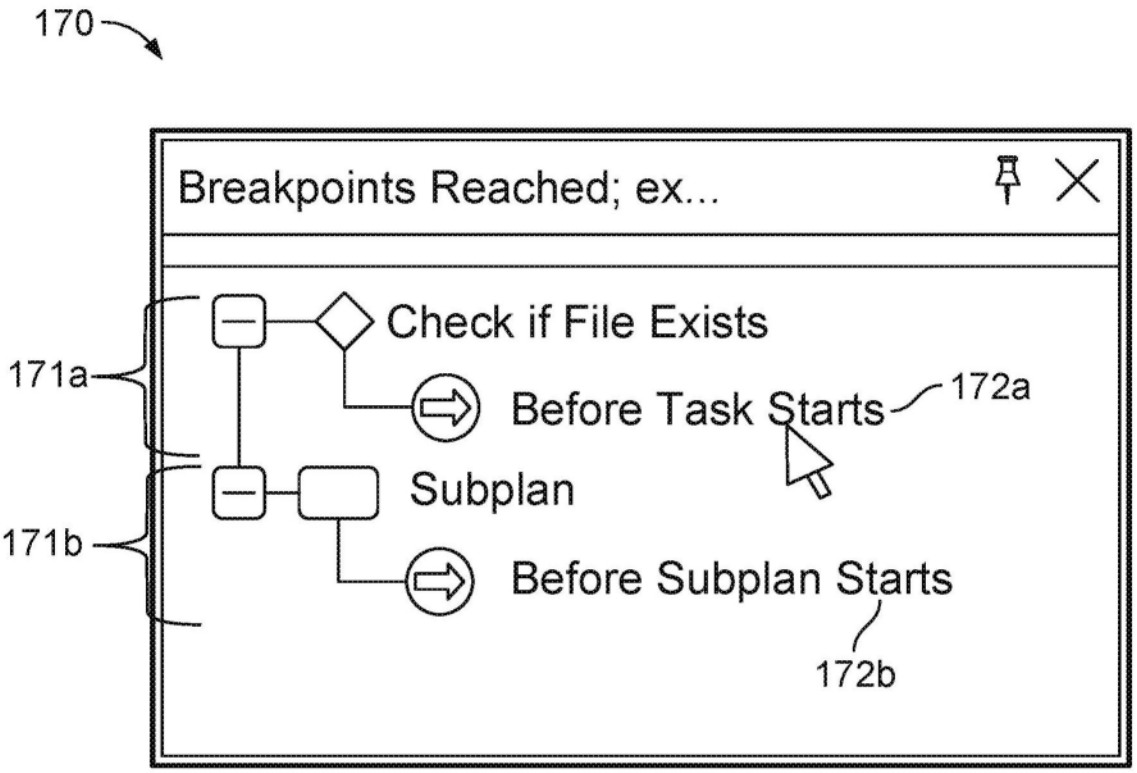

170
Breakpoints Reached; ex...
Check if File Exists
Before Task Starts
172a
Subplan
Before Subplan Starts
172b
171a
171b

图7B

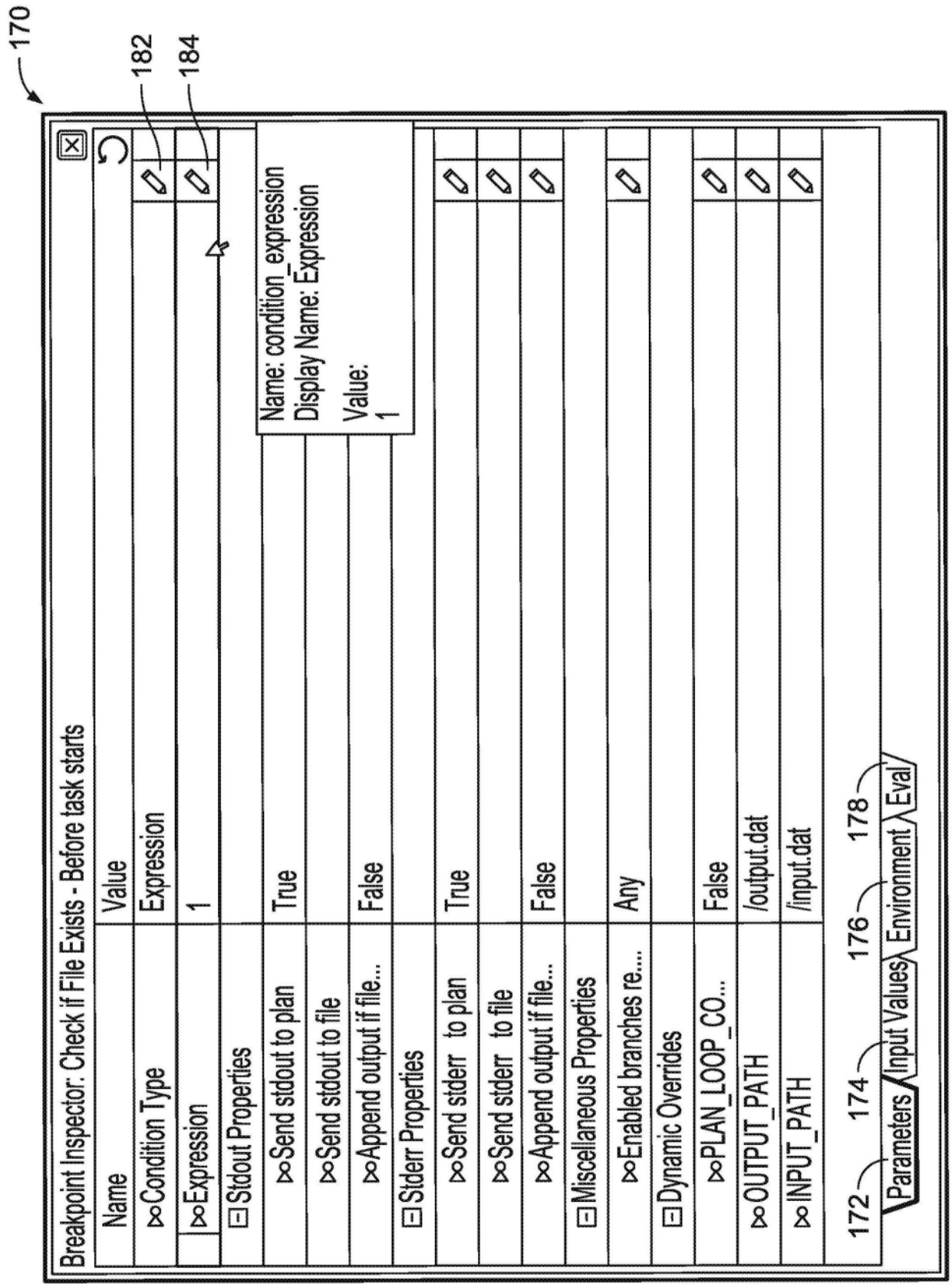

170
182
184
Breakpoint Inspector: Check if File Exists - Before task starts
Name
Value
Condition Type
Expression
Expression
1
Stdout Properties
Send stdout to plan
True
Send stdout to file
Append output if file...
False
Stderr Properties
Send stderr to plan
True
Send stderr to file
Append output if file...
False
Miscellaneous Properties
Enabled branches re....
Any
Dynamic Overrides
PLAN_LOOP_CO...
False
OUTPUT_PATH
/output.dat
INPUT_PATH
/input.dat
Name: condition_expression
Display Name: Expression
Value:
1
172
174
176
178
Parameters
Input Values
Environment
Eval

图8

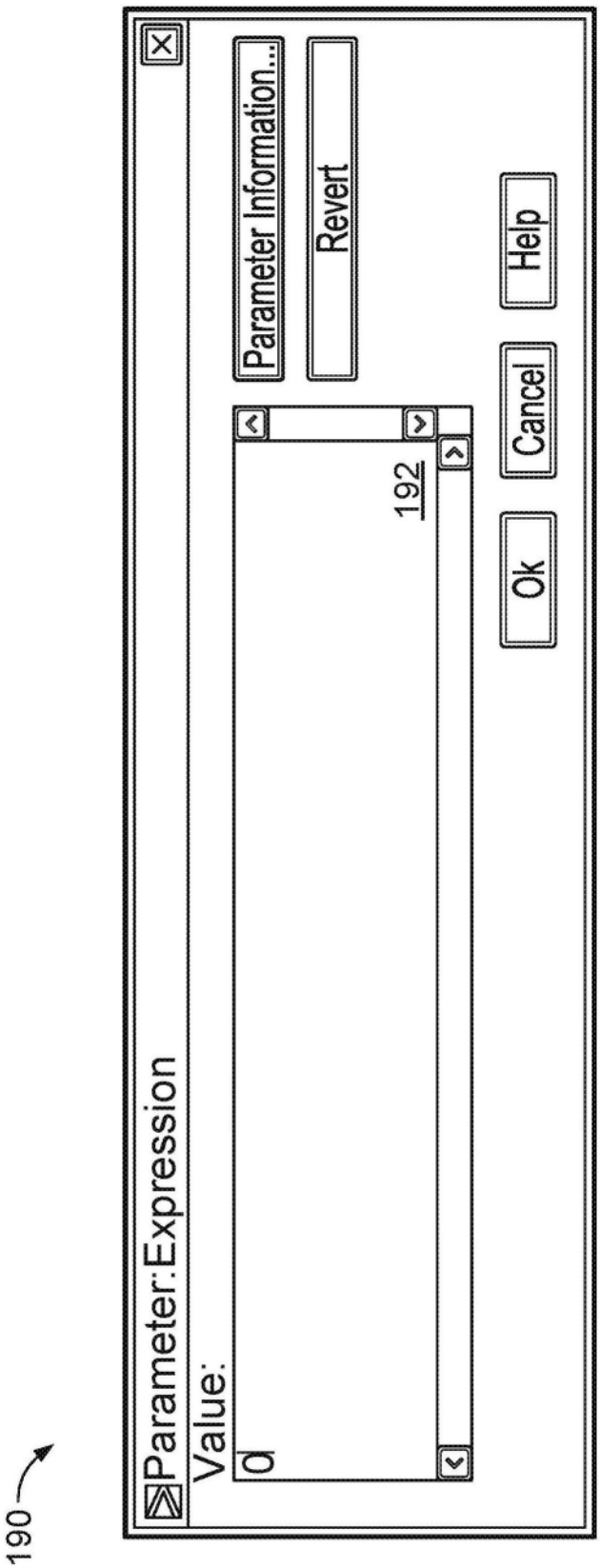

190
Parameter:Expression
Value:
0
192
Parameter Information...
Revert
Ok
Cancel
Help

图9A

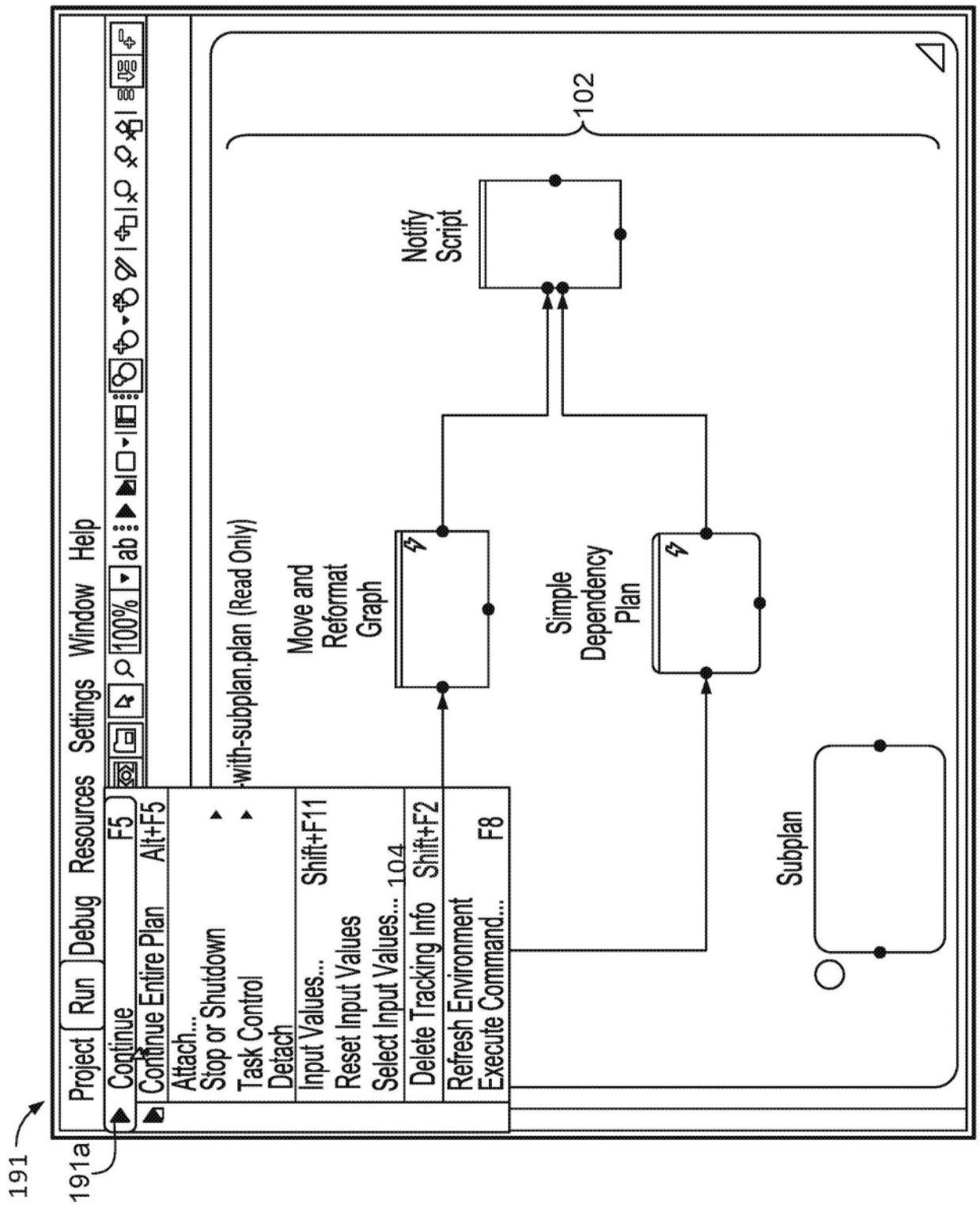

191
191a
Project
Run
Debug
Resources
Settings
Window
Help
Continue
F5
Continue Entire Plan
Alt+F5
Attach...
Stop or Shutdown
Task Control
Detach
Input Values...
Shift+F11
Reset Input Values
Select Input Values...
104
Delete Tracking Info
Shift+F2
Refresh Environment
Execute Command...
F8
with-subplan.plan (Read Only)
Move and Reformat Graph
Simple Dependency Plan
Notify Script
Subplan
102

图9B

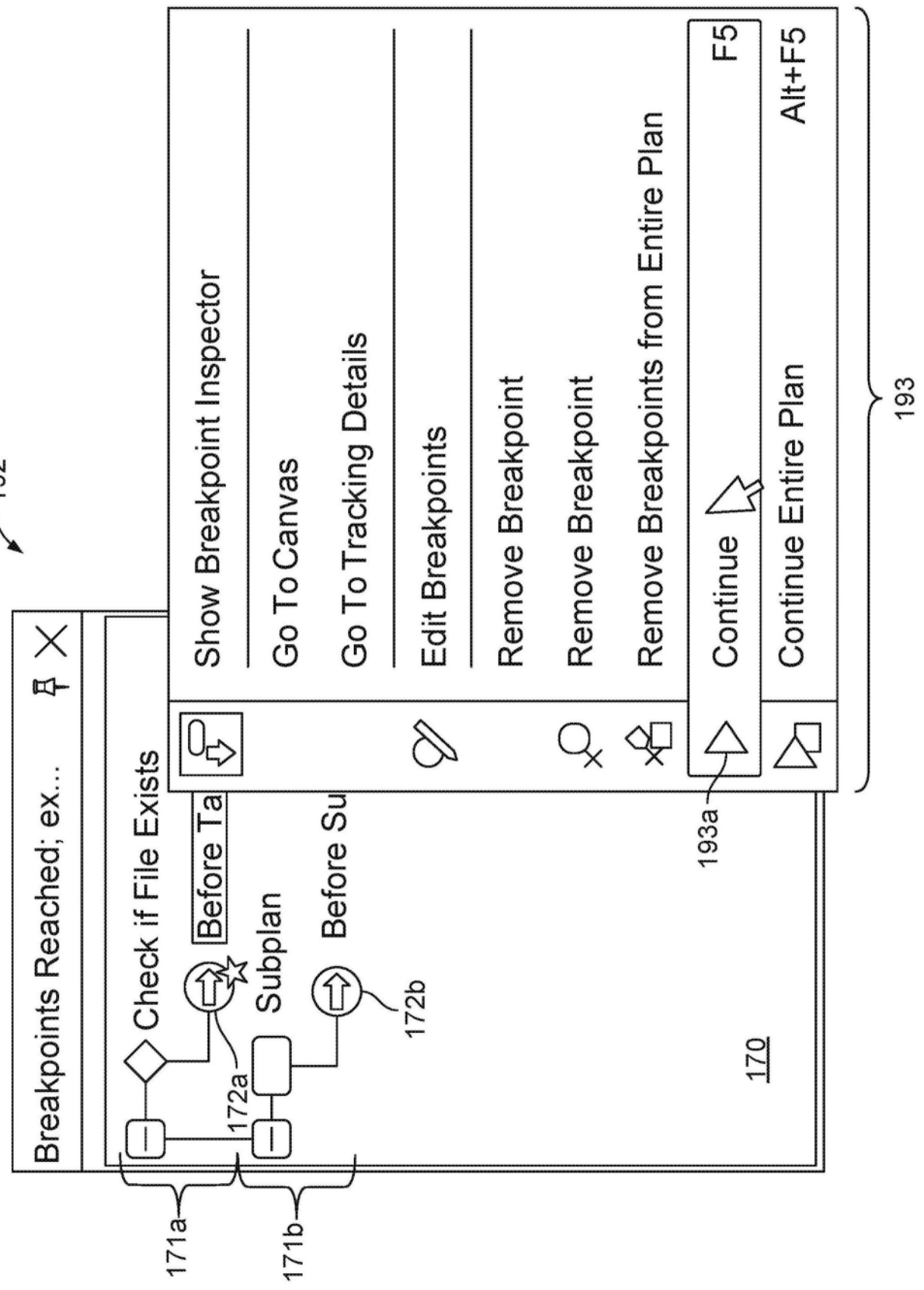

192
Breakpoints Reached; ex...
Check if File Exists
Before Ta
Subplan
Before Su
172a
172b
171a
171b
170
Show Breakpoint Inspector
Go To Canvas
Go To Tracking Details
Edit Breakpoints
Remove Breakpoint
Remove Breakpoint
Remove Breakpoints from Entire Plan
Continue
F5
Continue Entire Plan
Alt+F5
193
193a

图9C

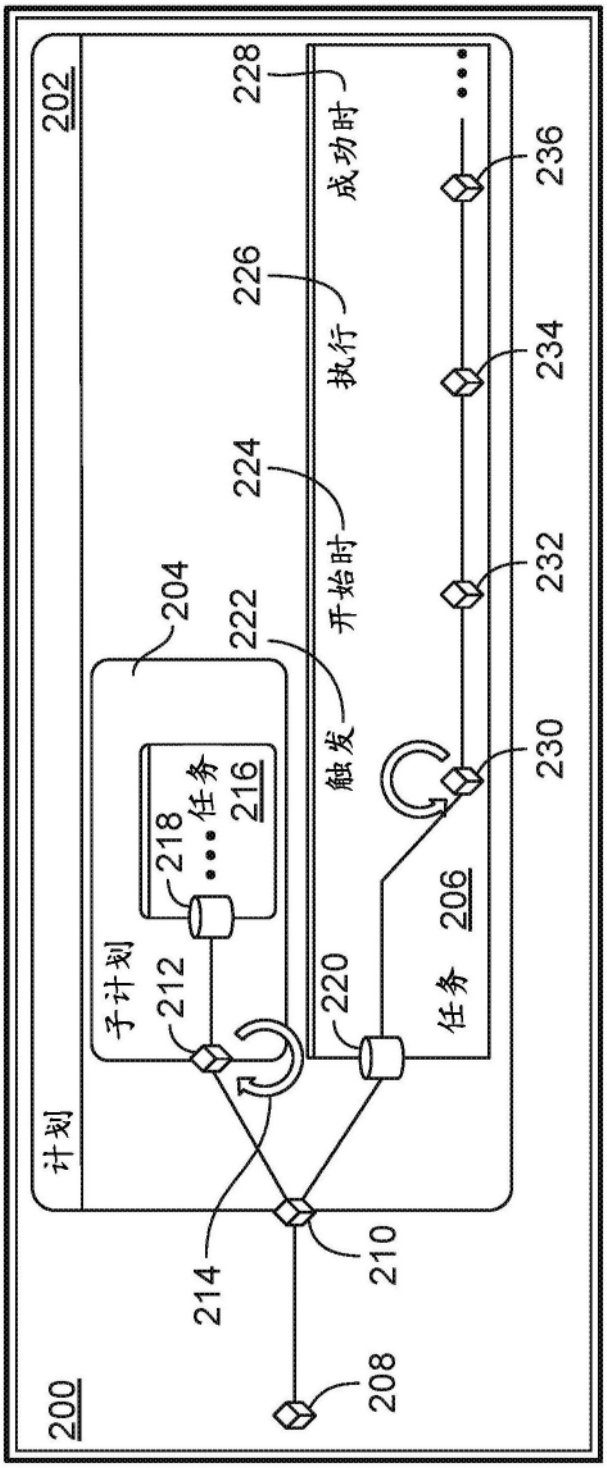

200
202
计划
204
子计划
212
218
...任务
216
222
触发
224
开始时
226
执行
228
成功时
...
236
234
232
230
任务 206
220
214
210
208

图10

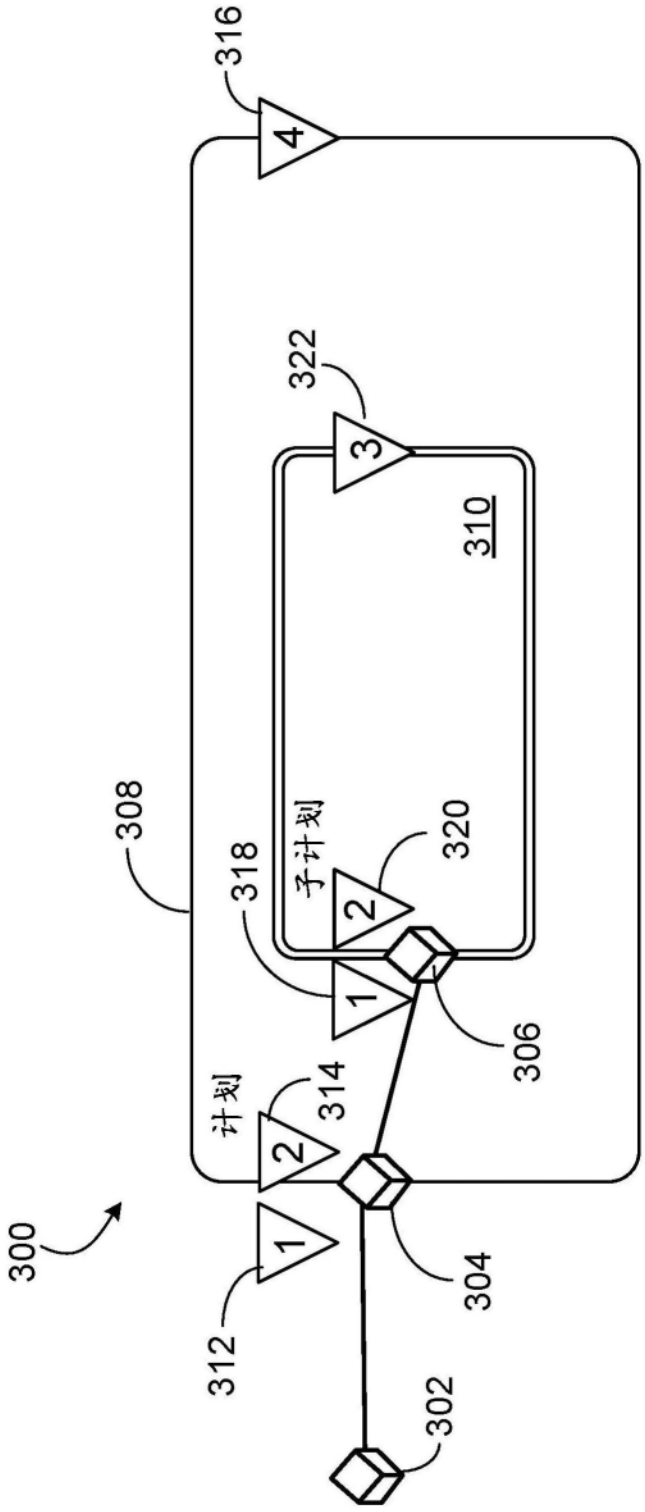


图11

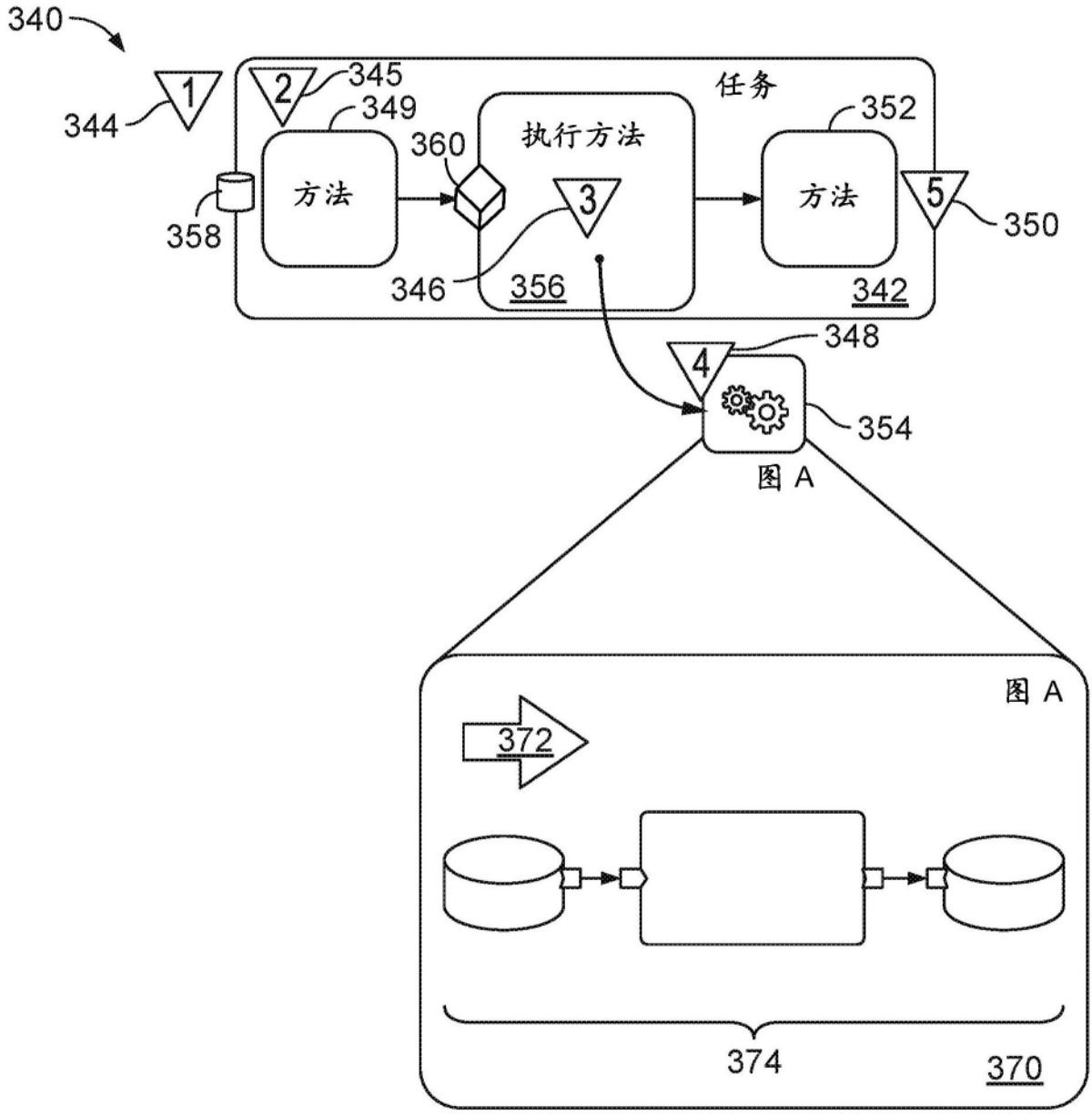

340
344
1
2
345
349
任务
352
执行方法
360
方法
3
方法
5
350
358
346
356
342
348
4
354
图 A
图 A
372
374
370

图12

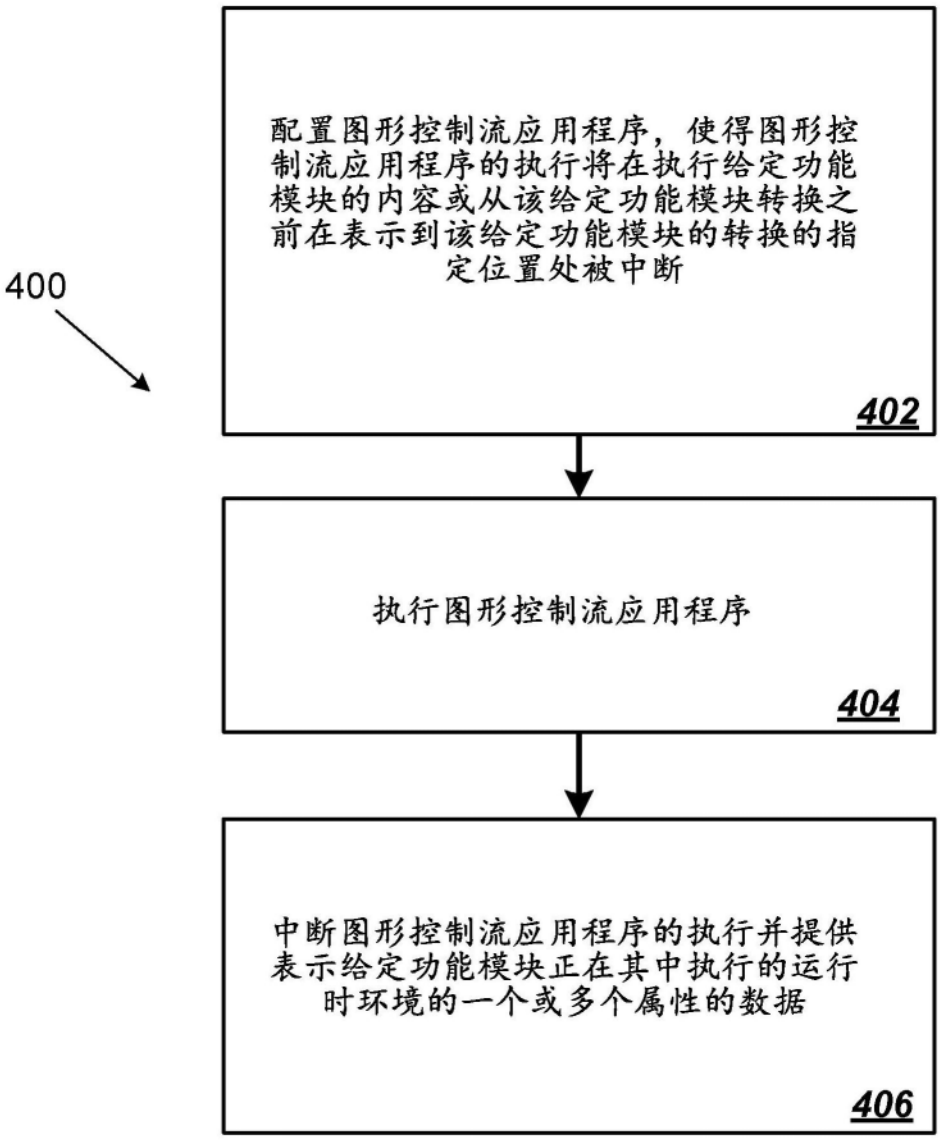

400
配置图形控制流应用程序，使得图形控制流应用程序的执行将在执行给定功能模块的内容或从该给定功能模块转换之前在表示到该给定功能模块的转换的指定位置处被中断
402
执行图形控制流应用程序
404
中断图形控制流应用程序的执行并提供表示给定功能模块正在其中执行的运行时环境的一个或多个属性的数据
406

图13

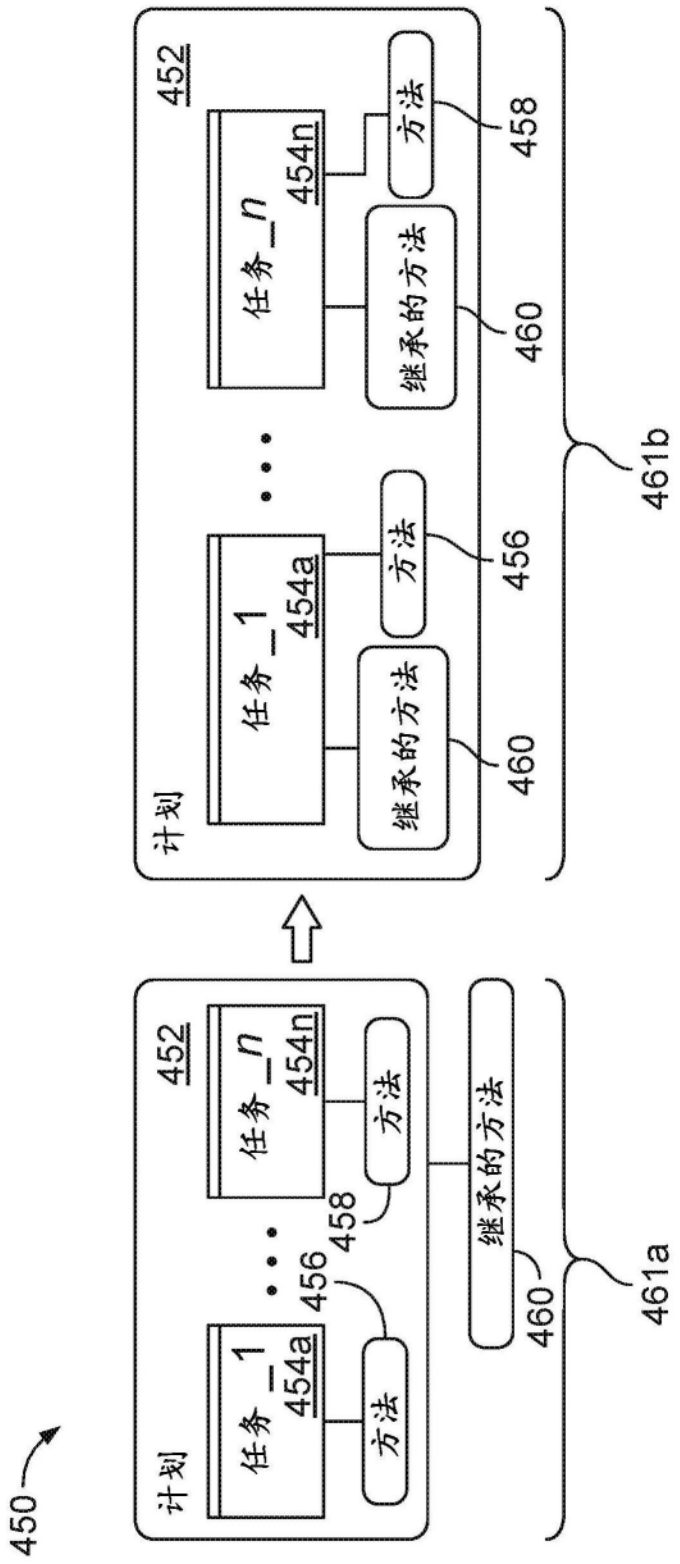


图14A

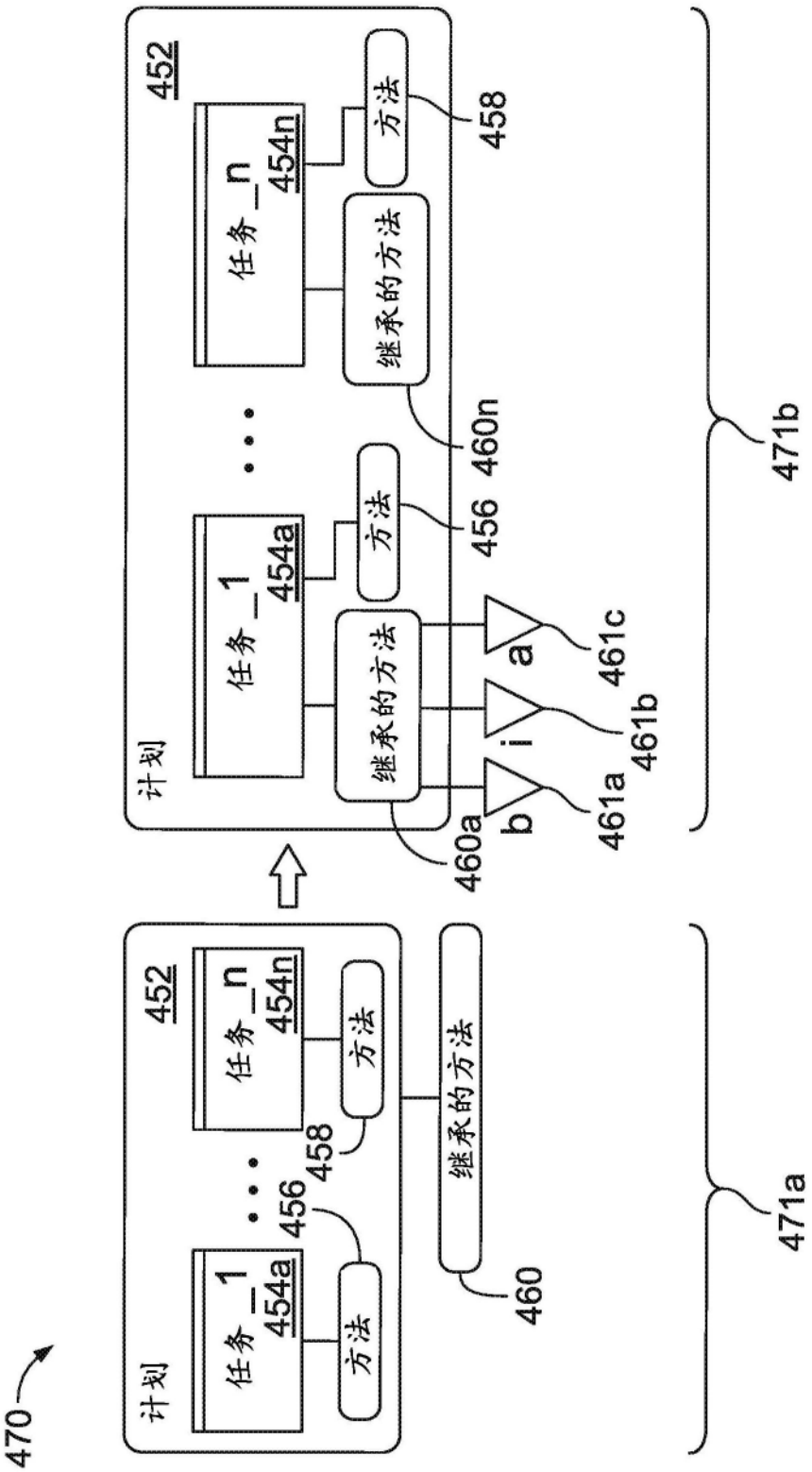

470
计划
452
任务_1
454a
任务_n
454n
方法
456
458
继承的方法
460
471a
460a
460n
461a
461b
461c
a
i
b
471b

图14B

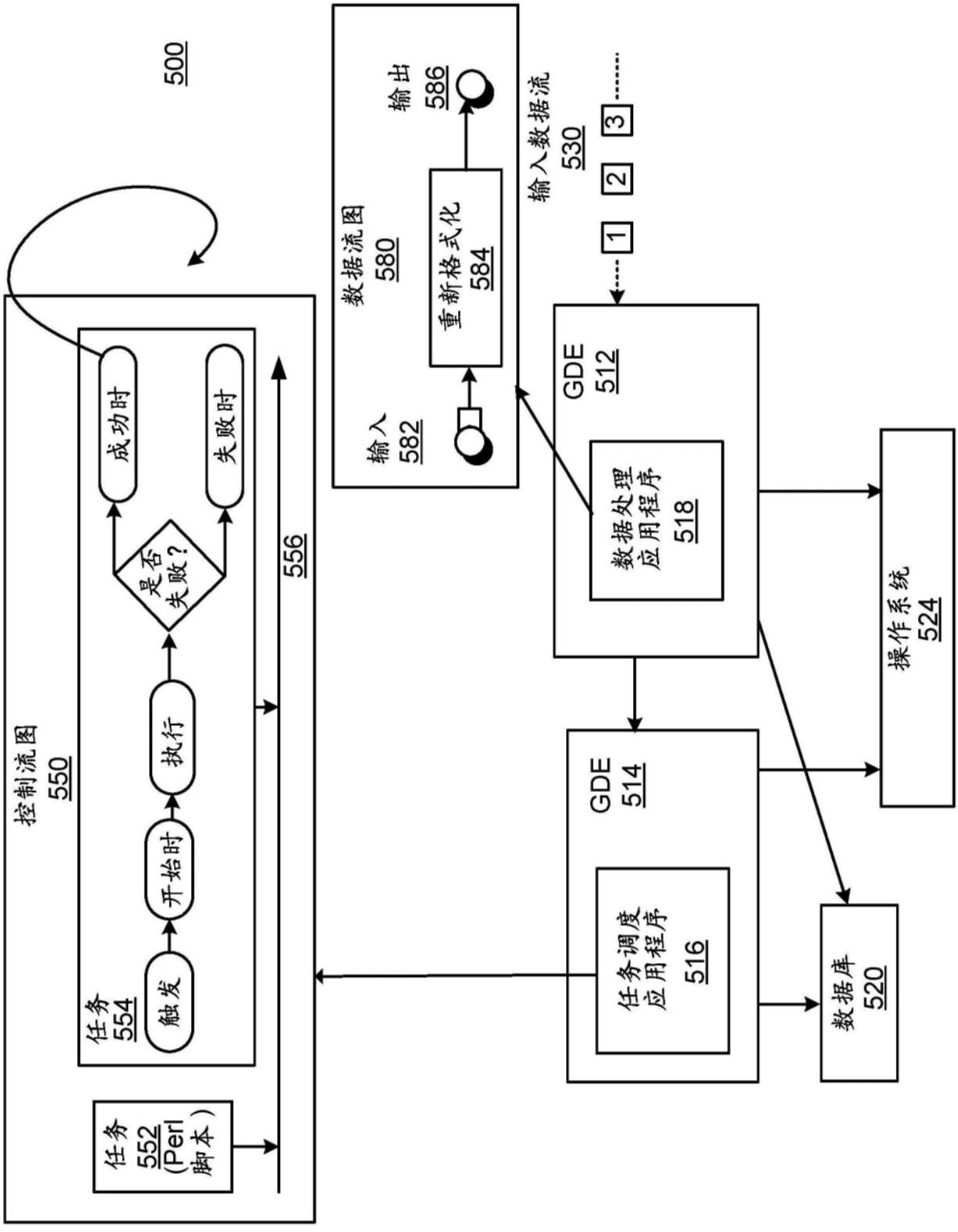

500
控制流图
550
任务
554
触发
开始时
执行
是否
失败?
成功时
失败时
556
任务
552
(Perl
脚本)
数据流图
580
输入
582
重新格式化
584
输出
586
输入数据流
530
1
2
3
GDE
512
数据处理
应用程序
518
GDE
514
任务调度
应用程序
516
操作系统
524
数据库
520

图15