(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 102193810 A
(43)申请公布日 2011.09.21

(21)申请号 201110065928.4

(22)申请日 2011.03.11

(30)优先权数据
12/722,560 2010.03.12 US

(71)申请人 微软公司
地址 美国华盛顿州

(72)发明人 S·比斯沃斯 D·J·希尼克
J·科塔斯 F·V·佩斯彻-盖里

(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 杨洁

(51)Int.Cl.
G06F 9/45(2006.01)

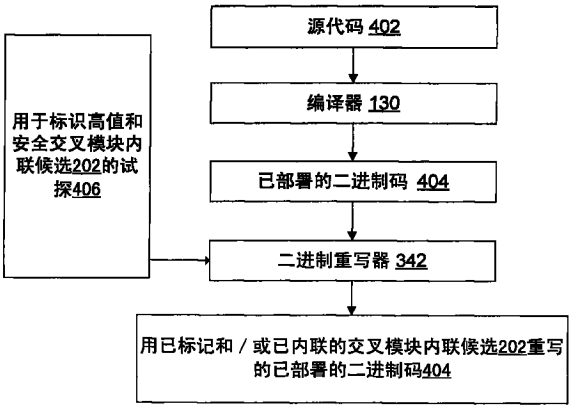
权利要求书 3 页 说明书 11 页 附图 3 页

(54)发明名称

模块间内联候选标识

(57)摘要

根据基于服务历史的准则、编译器内联准则和/或执行性能准则自动标识有可能是模块间内联的好候选的例程。也可以通过满足服务历史、执行性能和/或编译器准则的例程的模式匹配代码自动标识候选。已自动标识的候选例程被呈现在内联咨询工具中，允许开发者批准/禁止已自动标识的候选、添加其他例程以及建议或要求开发工具对特定例程执行模块间内联。对候选例程的改变可以触发本机图像的重新生成，例程已被编译进该本机图像。



CN 102193810 A

1. 一种用于管理模块间内联的方法，所述方法利用具有在操作上与至少一个存储器通信的至少一个逻辑处理器的设备，该方法包括以下步骤：

访问 (302) 服务历史 (126) 以获取例程集合；以及

将根据服务历史已被修改少于已指定阈值 (210) 次数的例程自动标识 (308) 为模块间内联的候选 (202)。

2. 如权利要求 1 所述的方法，其特征在于，还包括引用 (304) 执行性能准则，且其中所述标识步骤包括，将满足所述执行性能准则且根据服务历史也已被修改少于已指定阈值次数的例程自动标识为模块间内联的候选。

3. 如权利要求 1 所述的方法，其特征在于，还包括引用 (306) 编译器内联准则，且其中所述标识步骤包括，将满足所述编译器内联准则且根据服务历史也已被修改少于已指定阈值次数的例程自动标识为模块间内联的候选。

4. 如权利要求 3 所述的方法，其特征在于，还包括引用 (304) 执行性能准则，且其中所述标识步骤包括，将 (a) 满足执行性能准则、(b) 满足编译器内联准则且 (c) 根据服务历史已被修改少于已指定阈值次数的例程 (124) 自动标识为模块间内联的候选。

5. 如权利要求 1 所述的方法，其特征在于，所述标识步骤包括将以下项目的至少一个自动标识 (308) 为模块间内联的候选 (202)：

由字段支持的属性获取例程；

由字段支持的属性设置例程；

由单字段的比特支持的属性；

由恒定值支持的属性；

实现为针对单个比特的校验的布尔属性；

实现为针对空值的校验的布尔属性；

设置构造函数；

实现为对 .Equals() 的调用的相等运算符；

实现为对相等运算符和否定式的调用的不相等运算符；

实现为对单自变量构造函数的调用的显式强制性运算符；

增加恒定值的包装函数。

6. 如权利要求 1 所述的方法，其特征在于，所述方法还包括使用 (310) 内联咨询工具 (216) 来向开发者呈现所述例程以供考虑。

7. 一种用数据和指令配置的计算机可读非瞬态存储介质，所述指令在被至少一个处理器执行时使所述至少一个处理器执行一种用于管理模块间内联的方法，所述方法包括以下步骤：

访问 (302) 服务历史 (126) 以获取被考虑的例程初始集合；

从所述集合中排除 (312) 根据所述服务历史已被修改多于已指定阈值次数的每一个例程；

从所述集合中移除 (314) 不满足所指定的执行性能准则的每一个例程；

对留在所述集合中的例程的相应代码进行模式匹配 (316)，从而在已被修改少于所述已指定阈值次数且满足所述所指定的执行性能准则的例程中定位至少一个相对频繁的例程模式；以及

将具有与至少一个这样的相对频繁的例程模式匹配的代码的例程自动标识 (308) 为模块间内联候选。

8. 如权利要求 7 所述的经配置的介质，其特征在于，所述方法还包括以下步骤中的至少一个：

用指示所述例程是模块间内联的候选的属性来标注 (320) 已自动标识的例程；

使用 (322) 二进制码重写工具来标注已自动标识的例程用于内联；

使用 (310) 内联咨询工具来呈现已标注的已自动标识的例程供开发者考虑；

内联 (324) 已标注的已自动标识的例程。

9. 如权利要求 7 所述的经配置的介质，其特征在于，所述方法还包括引用 (306) 编译器内联准则，且其中所述标识步骤包括，将也满足所述编译器内联准则的例程自动标识为模块间内联的候选。

10. 如权利要求 7 所述的经配置的介质，其特征在于，所述方法将以下项目模式的至少三个定位 (326) 成已被修改少于所述已指定阈值次数且满足所述已指定执行性能准则的例程中的相对频繁例程模式：

由字段支持的属性获取例程；

由字段支持的属性设置例程；

由单字段的比特支持的属性；

由恒定值支持的属性；

实现为针对单个比特的校验的布尔属性；

实现为针对空值的校验的布尔属性；

设置构造函数；

实现为对 .Equals() 的调用的相等运算符；

实现为对相等运算符和否定式的调用的不相等运算符；

实现为对单自变量构造函数的调用的显式强制性运算符；

增加恒定值的包装函数。

11. 如权利要求 7 所述的经配置的介质，其特征在于，所述方法还包括以下步骤中的至少一个：

通过用户界面而不是模式匹配来接受 (328) 例程模式定义以定位例程模式，并且接着将具有与所接受的例程模式定义匹配的代码的例程自动标识为模块间内联的候选；

在将具有与所确认的例程模式匹配的代码的例程自动标识为模块间内联的候选之前，通过用户界面接收 (330) 已定位的相对频繁例程模式的确认。

12. 如权利要求 7 所述的经配置的介质，其特征在于，所述方法在开发环境内执行，且所述方法还包括查明 (332) 已自动标识的例程将能用于至少一个模块，并且接着将企图的或完成了的对例程的修改当做开发环境内的错误来对待，所述至少一个模块的身份在当所述例程被标识为模块间内联的候选时未在所述开发环境中被指定。

13. 一种系统，包括：

处理器 (110)，在操作上与存储器 (112) 通信用于工具执行；

例程集合的服务历史 (126)，所述服务历史驻留在所述存储器中；

也驻留在所述存储器中的执行性能准则 (206)；以及

与所述服务历史和所述执行性能准则在操作上通信的模块间内联候选标识工具(204)。

14. 如权利要求 13 所述的系统，其特征在于，还包括也驻留在所述存储器中且与所述模块间内联候选标识工具在操作上通信的编译器内联准则 (208)。

15. 如权利要求 13 所述的系统，还包括：

根据所述服务历史已被修改少于已指定阈值次数的例程的至少一个相对频繁例程模式 (212)，所述相对频繁例程模式驻留在所述存储器中；以及

与所述相对频繁例程模式和代码库 (120) 在操作上通信的改变监视器工具 (218)；

其中所述系统被配置用于，如果所述改变监视器工具检测到所述代码库的例程的实现中的改变，且所述例程与相对频繁例程模式匹配，则发出警报 (220)。

模块间内联候选标识

背景技术

[0001] 内联扩展，亦称“内联”，是软件代码的手动或自动（例如，通过编译器或链接器）修改，用被调用例程的主体代替例程调用。一些语言，例如 C 和 C++，支持例程定义中的“内联”关键字，允许开发者建议编译器应尝试内联该例程。编译器使用开发者建议和其他准则来决定哪些例程调用应被内联。链接器可以执行内联，例如，与源不可用的例程和与库例程。运行时系统也可以执行内联。

[0002] 内联优化可以改善程序在运行时的时间和空间使用，但是也可能增加程序的二进制文件的大小。内联往往改善代码执行速度，但是内联也可能降低性能。例如，插入例程的多个副本可能增加代码大小足以使得代码不再适合高速缓存，导致更多高速缓存丢失。在嵌入系统中，较小的代码大小可能被较快的代码执行更加重要，使得内联没有吸引力。在一些系统中，从内联增加变量可能增加处理器寄存器使用足以导致额外 RAM 存取，因此减慢了执行速度。一些例程不能被内联，例如一些递归例程。

发明内容

[0003] 代码生成器可以将代码从应用程序的一个模块内联到该应用程序的另一个模块，以节省调用例程的开销。有时候，跨模块边界的内联可以改善应用程序性能。然而，关于模块间内联的决定包含折衷。如果模块可以独立演化，则演化一个模块常常使另一个模块的二进制代码无效。重新生成被无效的代码可能是时间和资源密集的。

[0004] 此处提供的一些实施例帮助标识可能是鉴于所涉及的折衷的模块间内联的好候选的例程。可以根据诸如服务历史准则、编译器内联准则和 / 或执行性能准则这样的准则来自动标识候选。例如，模块间内联的候选可以是这样的例程，其服务历史显示，相比于已指定的阈值它被较不频繁地和 / 或较少扩展地修改。可替代地，或此外，可以通过对满足服务历史、执行性能，和 / 或编译器准则的例程的代码进行模式匹配来自动标识候选。特定类型的例程也可以被标识为模块间内联的候选，例如 ：由字段支持的属性获取或设置例程、由单字段的比特或由恒定值支持的属性、特定布尔属性、设置构造函数、特定相等或不相等的运算符、特定 Cast 运算符，和增加恒定值的包装函数。

[0005] 可以将自动标识的候选例程呈现给开发者，用于内联咨询工具中考虑。内联咨询工具给予开发者批准 / 禁止已自动标识的候选、添加其他例程，以及建议或要求开发工具对特定例程执行模块间内联的权利。在一些情况中，自动标识例程的改变可以触发本地图像的重新生成，在本地图像中例程已被编译。

[0006] 给出的示例仅是说明性的。本发明内容并不旨在标识出所要求保护的主题的关键特征或必要特征，也不旨在用于限定所要求保护的主题的范围。相反，提供本发明内容是为了以简化的形式介绍将在以下具体实施方式中进一步描述的某些概念。本发明用权利要求书来定义，且在本发明内容与权利要求书冲突的情况下，应以权利要求书为准。

附图说明

[0007] 将参考附图来给出更具体的描述。这些附图仅示出了所选方面，且因此未完全确定覆盖或范围。

[0008] 图 1 是示出计算机系统并且还示出经配置的存储介质实施方式的框图，所述计算机系统具有至少一个处理器、至少一个存储器、搜索模块间内联候选的例程集合、以及操作环境中可存在于多个网络节点上的其它项目；

[0009] 图 2 是示出示例体系结构的中模块间内联候选的自动标识的框图。

[0010] 图 3 是示出一些方法的步骤和所配置的存储介质的实施方式的流程图；以及

[0011] 图 4 是进一步示出一些实施方式的数据流程图。

具体实施方式

[0012] 概览

[0013] 许多虚拟执行环境既支持动态代码生成也支持提前 (AOT) 编译。AOT 二进制代码被保持，可以跨进程共享，并且通常导致启动时间和在动态代码生成上应用程序的存储器使用的显著改善。本机执行环境仅支持 AOT 二进制代码。

[0014] 代码生成器可以将代码从一个函数内联到另一个以节省进行函数调用的开销，并启动进一步优化，从而生成运行更快的代码。跨模块边界执行这样的内联（称为模块间内联）可以改善应用程序性能。然而，如果模块可以独立演化，可能希望将高的值与内联决定相关联，因为演化一个模块常常使另一个模块的 AOT 二进制码无效。重新生成被无效的 AOT 二进制代码可能是时间和资源 (CPU、存储器、磁盘）密集的。为本机执行环境重新分发已重新生成的 AOT 二进制代码，例如，对于当应用程序和库厂商是不同实体时许多应用程序所消费的库而言，可能是困难的或不切实际的。

[0015] 跨模块内联的一些熟知方法包括，不允许模块间内联、总是重新生成受影响的 AOT 二进制代码，和 / 或仅依赖于主观的开发者注释来选择跨模块内联候选。

[0016] 相反，此处呈现的方法支持可以被应用到预编译的已管理的（诸如 C# 和 Java) 以及本机（诸如 C 和 C++) 代码以使得性能从模块间内联胜出的简单和自动方案。此处描述的一些实施例使用基于函数的形状的试探来标识这样的函数，所述函数对于跨模块内联为高值并且不大可能演化，并因此不大可能影响其他 AOT 二进制代码。所标识的函数可以通过例如二进制码重写工具来标记，使代码生成器被配置成仅仅跨模块边界内联这种已标记函数。试探也可以被用于创建标志这种方法的内联咨询工具；开发者可以检查已标志的方法并适当地标记它们。

[0017] 一些实施例提供了自动处理二进制代码以允许高值模块间内联而不会影响已经内联进代码的模块的系统。一些提供了标识高值模块间内联候选的试探。一些实施例提供了内联顾问，内联顾问向开发者提供有关特定函数是否是好 / 安全的模块间内联候选的评估。一些提供了允许将自动标识和开发者标识的函数的混合跨模块被内联的方案。在一些实施例中，使用试探来将自动标识对于跨模块内联高值和安全的函数 / 方法的模式

[0018] 在一些实施例中，已标识的函数用二进制码重写工具来标记。在一些中，代码生成器被修改为仅仅跨模块边界内联这些已标记的函数。任何对于这样的已标记函数所作的修改在构建期间被自动标志为违反。被这些试探自动标识的函数可以用开发者标识的函数来补充。该试探也可以由内联咨询工具用来帮助开发者容易且一致地为模块间内联标记函

数。

[0019] 现在将参考诸如附图中示出的示例性实施例，且此处将使用具体语言来描述这些实施例。但是相关领域且拥有本发明的技术人员将想到的、此处所示的特征的更改和进一步修改以及此处所示的原理的其他应用应被认为是在权利要求书的范围内。

[0020] 各术语的意义在本发明中阐明，因此权利要求书应仔细注意这些阐明来阅读。给出了具体示例，但是相关领域的技术人员将理解，其他示例也落入所使用的术语的意义内，且在一个或多个权利要求的范围内。各术语不一定需要具有与它们在一般使用中、在特定行业使用中、或在特定字典或一组字典的使用中所具有的意义相同的意义。与各种措辞一起使用附图标记来帮助显示术语的广度。给定一段文本中省略附图标记并不一定意味着该附图的内容没有被该文本讨论。发明人声称并行使其对于其自己的词典编纂的权利。各术语此处可在详细描述和 / 或申请文件的别处显式或隐式地定义。

[0021] 如本文所使用的，“计算机系统”可包括例如一个或多个服务器、主板、处理节点、个人计算机（便携式或非便携式）、个人数字助理、蜂窝或移动电话、和 / 或提供至少部分地由指令控制的一个或多个处理器的其他设备。指令可以是以存储器和 / 或电路中特定的软件的形式。具体而言，尽管可以想到许多实施例在工作站或膝上型计算机上运行，但其他实施例可以在其他计算设备上运行，且任何一个或多个此类设备可以是给定实施例的一部分。

[0022] “多线程化”计算机系统是支持多个执行线程的计算机系统。术语“线程”应被理解为包括能够或经历同步的任何代码，并且可用另一名称来称呼，如“任务”、“进程”或“协同例程”。线程可以并行地、顺序地、或以并行执行（例如，多处理）和顺序执行（例如，时间分片）的组合来运行。以各种配置设计了多线程化环境。执行线程可以并行地运行，或者线程可以被组织供并行执行但实际依次轮流执行。多线程化可以例如通过在多处理环境中在不同核上运行不同线程、通过对单个处理器核上的不同线程进行时间分片、或通过时间分片和多处理器线程化的某种组合来实现。线程上下文切换可以例如由内核的线程调度器、由用户空间信号、或由用户空间和内核操作的组合来发起。线程可轮流在共享数据上操作，或者每一线程可以例如在其自己的数据上操作。

[0023] “逻辑处理器”或“处理器”是单个独立的硬件线程处理单元。例如，每一个核运行两个线程的超线程化四核芯片具有 8 个逻辑处理器。处理器可以是通用的，或者它们可以针对诸如图形处理、信号处理、浮点算术处理、加密、I/O 处理等特定用途来定制。

[0024] “多处理器”计算机系统是具有多个逻辑处理器的计算机系统。多处理器环境以各种配置出现。在一给定配置中，所有处理器可以在功能上是等价的，而在另一配置中，某些处理器可以借助具有不同硬件能力、不同软件分配或两者而不同于其他处理器。取决于配置，处理器可在单个总线上彼此紧耦合，或者它们可以是松耦合的。在某些配置中，处理器共享中央存储器，在某些配置中它们各自具有其自己的本地存储器，且在某些配置中存在共享和本地存储器两者。

[0025] “内核”包括操作系统、系统管理程序、虚拟机、以及类似的硬件接口软件。

[0026] “代码”表示处理器指令、数据（包括常量、变量和数据结构）或指令和数据两者。

[0027] “例程”意味着函数、方法，或其他代码片段，其非循序地接收控制并接着例如，通过压到调用堆栈上的地址来返回。除了既不接受参数也不返回值的无参数的空例程，例程

也使用堆栈来接收和 / 或返回值。

[0028] “自动”意味着通过使用自动化（例如，由软件针对此处讨论的具体操作配置的通用计算硬件），与不使用自动化相对。具体而言，“自动”执行的步骤并不是在纸张上用手执行或在人的脑海中执行的；它们是用机器来执行的。

[0029] 贯穿本文，对任选的复数的使用意味着存在一个或多个所指示的特征。例如，“（诸）例程”意味着“一个或多个例程”或等效于“至少一个例程”。类似的，“准则”意味着“一个或多个准则”或等效于“至少一个准则”。

[0030] 只要参考了数据或指令，就理解这些项目配置了计算机可读存储器，从而将其变换为特定物品，而非简单地存在于纸张上、人的脑海中、或作为例如线路上的瞬时信号。

[0031] 操作环境

[0032] 参考图 1，用于一个实施例的操作环境 100 可包括计算机系统 102。计算机系统 102 可以是多处理器计算机系统，也可以不是。操作环境可包括给定计算机系统中的一个或多个机器，它们可以群集的、客户机 - 服务器联网的、和 / 或对等联网的。

[0033] 人类用户 104 可以通过使用显示器、键盘和其他外围设备 106 来与计算机系统 102 交互。系统管理员、开发者、工程技术人员、以及最终用户各自都是特定类型的用户 104。代表一个或多个人来行动的自动化代理也可以是用户 104。存储设备和 / 或联网设备在某些实施例中可被认为是外围设备。图 1 中未示出的其他计算机系统可以使用经由例如网络接口设备到网络 108 的一个或多个连接来与计算机系统 102 或与另一系统实施例交互。

[0034] 计算机系统 102 包括至少一个逻辑处理器 110。如其他合适的系统，计算机系统 102 还包括一个或多个计算机可读非瞬态存储介质 112。介质 112 可以是不同的物理类型。介质 112 可以是易失性存储器、非易失性存储器、固定在原处的介质、可移动介质、磁介质、光介质、和 / 或其他类型的非瞬态介质（与诸如仅传播信号的线路等瞬态介质形成对比）。具体地，诸如 CD、DVD、记忆棒或其他可移动非易失性存储介质等已配置介质 114 在被插入或以其他方式安装时可以变为计算机系统的功能部分，从而使得其内容可被存取来供处理器 110 使用。可移动的已配置介质 114 是计算机可读存储介质 112 的一个示例。计算机可读存储介质 112 的某些其他示例包括内置 RAM、ROM、、硬盘、和不可由用户 104 容易地移动的其他存储设备。

[0035] 介质 114 用可由处理器 110 执行的指令 116 来配置；“可执行”在此以宽泛的意义用来包括机器代码、可解释代码、以及在例如虚拟机上运行的代码。介质 114 还用通过指令 116 的执行创建、修改、引用和 / 或以其他方式使用的数据 118 来配置。指令 116 和数据 118 配置它们所在的介质 114；当该存储器是给定计算机系统的功能部分时，指令 116 和数据 118 还配置该计算机系统。在某些实施例中，数据 118 的一部分代表了诸如产品特性、库存、物理测量、设置、图像、读数、目标、量等的真实项目。如本文讨论的，这样的数据也被转换，例如通过内联、绑定、部署、执行、修改、显示、创建、加载、和 / 或其他操作。

[0036] 代码库 120 包含诸模块 122 和诸例程 124、跟踪例程 124 的改变的服务历史 126、诸如（诸）编译器 130、（诸）调试器 132、（诸）链接器 134，和 / 或（诸）分析器 136 等的开发工具 128，以及其他图中所示的项目，可以部分地或全部地驻留在一个或多个介质 112 中，从而配置那些介质。例如，操作环境也可以包括显示器 138，以及其他硬件，例如，总线、电源，和加速器。

[0037] 给定操作环境 100 可包括向开发者提供一组协同的软件开发工具的集成开发环境 (IDE)140。具体而言，对于一些实施方式，合适的操作环境中的一些包括或帮助创建被配置成支持程序开发的Microsoft® Visual Studio®开发环境（微软公司的标记）。一些合适的操作环境包括Java®环境 (Sun Microsystems 公司的标记)，并且一些操作环境包括利用诸如 C++ 或 C#(“C-Sharp”) 等语言的环境，但本文的教导适用于各种各样的程序设计语言、程序设计模型、以及程序，以及本质上适用于软件开发领域之外的使用内联的努力。

[0038] 一些项目在图 1 中以轮廓形式示出以强调它们不是所示操作环境的必需部分，但可以与在此讨论的操作环境中的项目进行互操作。在任何附图或任何实施例中，不能得出不采用轮廓形式的项目就一定是不需要的。

[0039] 系统

[0040] 图 2 示出适用于一些实施方式的体系结构。模块间内联的诸候选 202 由标识工具 204 自动标识。标识工具通过分析例程 124 并具体通过根据准则来测试例程 124 来标识候选，所述准则诸如修改的频次 / 范围（基于服务历史 126）、遵循一个或多个执行性能准则 206 以及一个或多个编译器内联准则 208。

[0041] 对例程修改的频次 / 范围可以用（诸）阈值 210 来指定，例如，“过去六个月中没有修改”、“过去三年中没有修改超过两次”、“没有被修改，除了可能在自动从生产构建中排除的测试代码部分中”等等。

[0042] 执行性能准则 206 也可以用（诸）阈值 210 来指定，例如，“在执行期间至少调用了五十次”或“从代码中的至少十个位置调用”。

[0043] 编译器内联准则 208 也可以用（诸）阈值 210 来指定，例如，“例程主体少于十行源代码”或“例程主体包含少于 64 字节的中间语言代码”。

[0044] 满足所指定的准则的诸例程 124 可以被手动地和 / 或有自动帮助地分类到诸模式 212 中。标识工具可以接着通过用句法性的和 / 或语义的模式匹配器 214 分析诸例程 124 以标识与指定模块间内联的好候选的一个或多个已指定模式 212 相匹配的诸例程 124，从而标识候选。参考Microsoft®的中间语言 (MSIL)，好候选的模式 212 的一个示例是字段支持的属性获取函数和设置函数。在这样的方法中，实现是单个字段提取或存储的已知 MSIL 序列，其已设置了 specialname（特殊名称）标志，且方法名称以“get_”或“set_”开始。模式 212 的另一个示例是设置构造函数，其实现是在对库构造函数的可选调用之后对字段的零或多个参数存储，其设置 specialname（特殊名称）标志，且其方法名称包括“.ctor”。设置构造函数仅仅将它们的参数分配给被构造的对象的字段，而其他构造函数可以简单地分配对象（且不将字段初始化为任何值）或做其他事情。模式 212 的第三个示例是包装函数，其添加恒定值（缺省）；它们的实现是设置零个或多个参数（当前方法的参数或恒定值），其后跟随定标同一类中的方法的单个调用返回指令 (call-ret) 序列。

[0045] 内联咨询工具 216 给予开发者批准 / 禁止已自动标识的候选 202、将其他例程 124 添加为候选，以及建议或要求开发工具 128 对特定例程执行模块间内联的权利。在一个实施例中，内联咨询工具 216 包括图形用户界面 (GUI)，其提供对标识工具 204 的结果的方便访问。内联咨询工具 216、服务器历史 126，和 / 或标识工具 204 可以是集成开发环境 140 的一部分。

[0046] 在一些实施例中，改变监视器工具 218 也访问标识工具 204 的结果，例如，将它们

指定为模块间内联候选的特定例程上的属性或标记。在一些情况中，当尝试改变已内联的例程 124 或内联候选 202 时，发出警报 220，诸如错误消息、警告或构建拒绝。这样的警报可以帮助减少二进制码图像的无效 / 重新生成。

[0047] 参考图 1 和图 2，一些实施例向计算机系统 102 提供了由电路、固件，和 / 或软件配置的逻辑处理器 110 和存储介质 112，以通过用例如标识工具 204、内联咨询工具 216，和 / 或改变监视器工具 218 来扩展功能来方便模块间内联的管理，如此处所描述。

[0048] 一个实施例包括，在操作上与存储器通信用于工具执行的处理器 110，在存储器中用于诸例程 124 的集合的服务历史 126，也驻留在存储器中的执行性能准则 206，以及在操作上与服务历史和执行性能准则通信的模块间内联候选标识工具 204。在一个变化中，一些系统包括也驻留在存储器中在操作上与工具 204 通信的编译器内联准则 208。一些包括存储器中的内联咨询工具 216。一些包括模式匹配器 214，以及例程 124 中根据服务历史 126 相比于已指定的阈值 210 被较少地修改的至少一个相对频繁的例程模式 212。相对频繁例程模式 212 驻留在存储器中，在操作上与候选标识工具 204 通信。

[0049] 一些系统也包括以改变监视器工具 218 为形式的测试自动化，改变监视器工具 218 在操作上与相对频繁例程模式 212 以及与代码库 120 通信。该系统被配置用于，如果改变监视器工具检测到代码库的例程 124 的实现中的改变，且如果那个例程也与相对频繁例程模式 212 匹配，则发出警报 220。在一些实施例中，模式是可调的。如果警报 220 发出得太频繁，开发者可以缩小模式 212 的范围以包括更少的例程。

[0050] 在某些实施例中，诸如人类用户 I/O 设备（屏幕、键盘、鼠标、图形输入板、话筒、扬声器、运动传感器等）等的外围设备 106 将存在于与一个或多个处理器 110 和存储器的可操作通信中。然而，一实施例还可被深嵌入在系统中，使得没有人类用户 104 直接与该实施例交互。软件进程可以是用户 104。

[0051] 在某些实施例中，该系统包括通过网络连接的多个计算机。联网接口设备可使用诸如例如存在于计算机系统中的分组交换网络接口卡、无线收发机、或电话网络接口等组件来提供对网络 108 的接入。然而，一实施例也可通过直接存储器存取、可移动非易失性介质、或其他信息存储 - 检索和 / 或传输方法来通信，或者计算机系统中的一实施例可以在不与其他计算机系统通信的情况下操作。

[0052] 一些实施方式在“云”计算环境中和 / 或“云”存储环境中操作。例如，代码库 120 可以在联网云中的多个设备 / 系统 102 上，服务历史 126 可以存储在云中的还有一些其他设备上，而工具 204、216，和 218 可以在又一云设备 / 系统 102 上配置显示器 138。

[0053] 进程

[0054] 图 3 以流程图 300 示出了某些方法实施例。附图中所示的进程可以在某些实施例中自动地，例如，由在脚本控制下的标识工具 204 来执行，需要极少或不需要用户输入。各进程可以部分地自动执行，且部分地手动执行，除非另外指明。在给定实施例中，可以重复一进程的零个或多个所示步骤，可能对不同参数或数据进行操作。一实施例中的各步骤也可按与图 3 中列出的从上到下次序不同的次序来完成。各步骤可以串行地、以部分重叠的方式、或完全并行地执行。遍历流程图 300 来指示在一进程期间执行的步骤的次序可以在进程的一次执行与进程的另一次执行之间变化。流程图遍历次序也可在一个进程实施例与另一进程实施例之间变化。各步骤可被省略、组合、重命名、重组合、或采用其他方式，而不

脱离所示流程，只要所执行的进程是可操作的且符合至少一个权利要求。

[0055] 此处提供了各示例来帮助说明该技术的各方面，但是本文中给出的示例并未描述所有可能的实施例。各实施例不限于此处提供的具体实现、排列、显示、特征、方法或情形。给定实施例可以例如包括另外的或不同的特征、机制和 / 或数据结构，并且还可另外脱离此处提供的示例。

[0056] 在历史访问步骤 302 中，一实施例访问服务器历史 126 以获取例程集合 124。步骤 302 可以利用版本控制系统、IDE 140，和 / 或维持何时最后一次修改一给定例程和 / 或具有一给定例程的模块的记录的其他工具。

[0057] 在执行性能准则引用步骤 304 中，一实施例引用至少一个执行性能准则 206。引用可以是对准则的显式状态的引用，所述准则诸如由工具 204 用户选择的准则，或者引用可以是隐式的且嵌入在用于实现工具 204 的代码中。

[0058] 在编译器内联准则引用步骤 306 中，一实施例引用至少一个编译器内联准则 208。引用可以是对准则的显式状态的引用，所述准则诸如由工具 204 用户选择的准则，或者引用可以是隐式的且嵌入在用于实现工具 204 的代码中。

[0059] 在标识步骤 308 中，一实施例通过确定例程符合所有已指定的准则来将至少一个例程标识为模块间内联的候选 202。基于准则的步骤 302、304、306 可以被视为标识步骤 308 的一部分，或者是步骤 308 的先驱。然而，当例程没有符合所有已指定的准则时，步骤 302、304、306 也可以发生，且因此不被标识 308 为模块间内联的候选。

[0060] 在咨询工具使用步骤 310 期间，一实施例通过将候选 202 呈现给开发者、通过接收来自开发者的关于特定候选 202 是否应该保留候选的接受 / 拒绝输入、通过接受来自开发者的将未自动标识 308 的例程添加为候选的输入等，来使用内联咨询工具 310。

[0061] 类似其他涉及用户交互的步骤，步骤 310 可以从硬件 / 软件角度或从人类用户角度来看。从人类用户角度，在步骤 310 中，开发者通过观看显示器或候选 202 的其他列表、通过接受 / 拒绝自动提出的特定候选 202、通过将未自动标识 308 的例程输入为候选等，来使用内联咨询工具 310。

[0062] 在排除步骤 312，一实施例从模块间内联的候选（或预期候选）中排除模块间已被过于频繁和 / 或过于广泛修改的例程。也就是，不满足服务历史阈值准则的例程 124 从一组候选 202 中被排除。可以，例如，通过移除先前在不同准则下被标识 308 的例程，或通过阻止例程在第一种情况中被标记为候选，来完成排除。

[0063] 在移除步骤 314，一实施例从模块间内联的候选（或预期候选）中排除模块间未满足执行性能准则的例程。可以，例如，通过移除先前在不同准则下被标识 308 的例程，或通过阻止例程在第一种情况中被标记为候选，来完成步骤 314 下的移除。

[0064] 在模式匹配步骤 316，一实施例将例程 124 与已指定的例程模式 212 对比，并确定该例程是否与该模式匹配。尽管分析在步骤 316 中是针对候选 202 标识或模式 212 生成的模式匹配，而不是直接代码生成的模式匹配，但是模式匹配步骤 316 可以通过例如使用在编译中使用的类型的句法性的分析和 / 或语义的分析来执行。事实上，在一些实施例中，从开发者的角度，模式匹配步骤 316 和候选标识步骤 318 与编译是集成的。

[0065] 在校验步骤，一实施例在已指定阈值 210 的上下文中校验例程的服务历史，以确定例程是否被过于频繁和 / 或过于广泛地修改以有资格成为模块间内联的候选。服务历史

访问步骤 302 可以是校验步骤 318 的一部分，或者服务历史可以是隐含在步骤 318 的上下文中。

[0066] 在标注步骤 320，一实施例标注例程 124 以指示它是候选 202。例如，可以向例程的中间代码表示添加属性以执行标注，或可以更新诸如候选指针或候选列表这样的外部数据以指定例程。

[0067] 在二进制码重写器使用步骤 322，使用二进制码重写工具 342（也称为二进制码重写器）标记例程 124 以指示它应被模块间内联，或它可以被内联。

[0068] 在内联步骤 324，例程 124 被跨模块边界内联。内联可以是响应于在步骤 322 中例程的标记、响应于例程的标注 320，或响应于其他因素。由于编译器和其他工具 128 最终确定哪些例程实际上被内联，在一些情况中，未自动标识 308 的例程可能仍然被内联 324，而在一些情况中，已自动标识 308 的例程可能未被内联 324。

[0069] 在模式定位步骤 326，一实施例定位例程模式 212。例如，模式 212 可以驻留在包含模板的文件中，类似用于定义编程语言句法 / 语义的解析定义文件。模式 212 可以被硬译码为工具 204 中的解析代码，在这种情况中，定位模式涉及将控制传递给这样的解析代码。

[0070] 在模式定义接受步骤 328，一实施例从用户 104 接受例程模式 212 的定义。接受可以包括例如，接收模式文件，或通过 GUI 接收一个或多个预定义的模式的选择。例如，开发者可以使用标识工具 204 中的 GUI 或咨询工具 216 中的 GUI 来开启设置构造函数模式 212 的接受。开发者也可以定义模式 212。例如，开发者可以提供例程模式定义以匹配一函数，该函数的实现直接遵循其函数签名。

[0071] 在模式确认接收步骤 330，一实施例从用户 104 接收例程模式 212 的确认。当开发者通过 GUI 选择已显示的模式或录入指示应该应用所有当前模式定义的一个值时，确认可以是显式的。当给予开发者拒绝模式 212 的机会而开发者没有这样做的时候，确认也可以是隐式的。

[0072] 在可用性查明步骤 332，一实施例查明在诸如系统 102 的特定开发环境内的候选例程是否将可用于至少一个模块 122 中，在当例程被标识为模块间内联的候选的时间点上，所述至少一个候选模块 122 的身份未模块间在该开发环境内被指定。也就是，一些例程将仅对那些身份已知或至少知道可被确定的模块可用；在候选例程被修改后如果需要可定位和重新生成这样的模块。但是其他例程将被公布以用于未被库存的、未严格定位的以及未以其他方式指定的模块。对被如此宽泛地公布的例程的改变可以导致模块二进制代码的无效，对例程的开发者没有可行的方法来标识并因此重新生成那些模块。例如，可以通过从开发者或从关于候选例程的预期的 / 被允许的分发的配置文件接收输入，来执行可用性查明步骤 332。

[0073] 如果一候选例程被查明 332 为可用于未被指定且不能被容易地指定的模块，则在对待步骤 334，一实施例（例如，在 IDE 140 中）将候选例程的一建议的（或在某些情况中实际的）改变当做错误来对待。该实施例可以完全拒绝允许改变（例如，使得例程在编辑器中只读），或该实施例可以拒绝在已改变代码中允许作为构建的一部分。该实施例也可以发出 336 一个警报，例如，通过将消息显示在工具的 GUI 中。

[0074] 然而，在库存保持步骤 338，一些实施例维持列表、表单、目录、注册表，或模块 122 的其他库存及它们对某个（些）例程的特定版本的依存性，所述例程可以是全部例程或仅

是候选例程，这取决于实施例。作为库存步骤 338 的结果，在其中（可以）内联 324 候选例程的全部模块，可以在查明步骤 332 中被指定。接着可以允许对候选例程的改变，然后是包含候选的代码的模块图像的已委托和 / 或已自动化的重新生成 340。

[0075] 上述步骤及其相互关系在下文中结合各种实施例来更详细地讨论。本领域的技术人员将理解，在此实现细节可涉及诸如具体 API 和具体样本程序等具体代码，因此不需要出现在每一实施例中。本领域的技术人员还将理解，讨论细节时所使用的程序标识符和某些其他术语是特定于实现的，且因此不需要属于每一实施例。然而，尽管它们不一定需要存在于此，但仍提供了这些细节，因为它们可通过提供上下文来帮助某些读者，和 / 或可示出此处讨论的技术的许多可能实现中的几个。

[0076] 一些实施例提供了用于管理模块间内联的进程。该进程包括访问 302 服务历史 126 以获取例程集合 124，诸如特定代码库 120 中的例程。该进程也包括，将根据服务历史被修改少于已指定阈值 210 次数的例程 124 自动标识 308 为模块间内联的候选 202。

[0077] 在一个变化中，一些实施例也引用 304 执行性能准则 206。这些实施例将满足执行性能准则且根据服务历史也被修改少于已指定阈值次数的例程 124 自动标识 308 为模块间内联的候选 202。

[0078] 在又一个变化中，一些实施例也引用 306 编译器内联准则 208。这些实施例将满足编译器内联准则且根据服务历史也已被修改少于已指定阈值次数的例程 124 自动标识 308 为模块间内联的候选 202。

[0079] 一些实施例引用 304 执行性能准则 206 并且也引用 306 编译器内联准则 208。这些实施例将 (a) 满足执行性能准则、(b) 满足编译器内联准则且 (c) 根据服务历史被修改少于已指定阈值次数的例程 124 自动标识 308 为模块间内联的候选 202。

[0080] 在一些实施例中，标识步骤 308 将以下项目的至少一个自动标识为模块间内联的候选：由字段支持的属性获取例程、由字段支持的属性设置例程、由单字段的比特支持的属性、由恒定值支持的属性、实现为针对单个比特的校验的布尔属性、实现为针对空 (NULL) 值的校验的布尔属性、设置构造函数、实现为对 .Equals() 的调用的相等运算符、实现为对相等运算符和否定式的调用的不相等运算符、实现为对单自变量构造函数的调用的显式强制性运算符，和增加恒定值的包装函数。例如，这些项目可以作为模式匹配 316 的结果被标识 308，或作为被自动标注 320 的结果被标识 308。在一些实施例中，该进程也使用 310 内联咨询工具 216 来将例程呈现给开发者供考虑。

[0081] 一些实施例提供了用于管理模块间内联的进程。该进程访问 302 服务历史以获取被考虑的例程的初始集合，并接着从该集合中排除 312（根据服务历史）被修改超过已指定阈值次数的每一个例程，而且也从该集合中移除 314 不满足已指定执行性能准则的每一个例程。对留在集合中的例程的相应代码执行模式匹配 316，从而在已被修改少于已指定阈值次数且满足已指定执行性能准则的例程中定位 326 至少一个相对频繁的例程模式。该进程将具有与至少一个这样的相对频繁的例程模式匹配的例程自动标识 308 为模块间内联候选。在一些实施例中，该进程还校验 318 由模式匹配定位的例程是否已被修改超过已指定阈值次数。

[0082] 在一个方法下，一个进程使用例程的第一集合来生成例程模式 212，并接着应用那些例程模式来定位不用于生成模式的其他代码中的匹配。在另一个方法下，同样的例程集

合被用来生成模式 212，并在标识 308 候选 202 时搜索匹配。

[0083] 在一些实施例中，该进程用指示例程是模块间内联的候选的属性来标注 320 已自动标识的例程。在一些实施例中，该进行使用 322 二进制码重写工具来标注已自动标识的例程用于内联。在一些实施例中，该进程使用 310 内联咨询工具来呈现已标注的已自动标识例程供开发者考虑。在一些实施例中，该进程内联 324 已标注已自动标识的例程。这些步骤可以以多种方式混合，一些可以忽略。例如，一种可能性是用属性标注 320 候选。另一种可能性是以某种方式使用二进制码重写器 342 标注（标记）候选。类似地，对已标注例程可以做的一件事是将它示给开发者以在咨询工具 216 中考虑，而另一件事（不需要排除该咨询工具）是继续下去并内联 324 该例程。

[0084] 一些实施例通过用户界面而不是模式匹配来接受 328 例程模式定义以定位例程模式，并且接着将具有与所接受的例程模式 212 定义匹配的代码的例程自动标识 308 为模块间内联的候选。一些实施例在将具有与所确认的例程模式匹配的代码的例程自动标识为模块间内联的候选之前，通过用户界面接收 330 已定位的相对频繁例程模式的确认。也就是，允许开发者在该进程使用模式匹配来标识内联候选之前，对模式匹配结果说“是”（也隐含地，“不”）。

[0085] 在一些实施例中，该进程在开发环境，例如 IDE140 或特定系统 102 中执行。该进程还查明 332 已自动标识的例程将可用于至少一个模块中，当例程被标识为模块间内联的候选时，所述至少一个模块的身份在开发环境中未被指定。接着，该进程将企图的或完成了的对例程的修改当做开发环境内的错误来对待 334。然而，在一些实施例中，该进程将本机图像的库存保持 338 在已自动标识的例程已经被编译进的目标系统上，并在例程被修改后为目标系统重新生成 340 全部的已库存本机图像。

[0086] 如附图 4 所示，在一些实施例中，例程 124 的源代码 402 被提交给编译器 130，该编译器 130 产生接着要被部署的二进制码 404。为了改善已部署代码的性能，由试探 406 引导的二进制码重写器 342 用来生成已部署二进制码的新版本，在其中，模块间内联候选已标记和 / 或已内联。试探 406 使用在此讨论的服务历史 126、阈值 219 和准则 206、208 来实现例如对不频繁修改的例程 124、频繁调用的例程和 / 或短例程的测试。

[0087] 配置的介质

[0088] 某些实施例包括配置的计算机可读存储介质 112。介质 112 可包括盘（磁盘、光盘或其他）、RAM、EEPROM 或其他 ROM、和 / 或其他可配置存储器，具体包括非瞬态计算机可读介质（与电线和其他传播信号介质相对）。配置的存储介质可以特别地是诸如 CD、DVD 或闪存等可移动存储介质 114。可以是可移动的或不可移动的且可以是易失性的或非易失性的通用存储器可被配置到使用诸如候选标识工具 204、内联咨询工具 216 和例程模式 212 的项目的实施例中来形成配置的介质，这些项目采用从可移动介质 114 和 / 或诸如网络连接等另一源读取的数据 118 和指令 116 的形式。经配置的介质 112 能够使计算机系统执行用于通过如此处揭示的管理模块间内联来变换数据的进程步骤。图 1 到 4 因而帮助示出配置的存储介质实施方式和进程实施方式，以及系统和进程实施方式。具体而言，图 3 和 / 或图 4 中示出的各进程步骤中的任一步骤或本文以其他方式教导的任一步骤可被用来帮助配置存储介质来形成配置的介质实施方式。

[0089] 结论

[0090] 尽管此处将具体实施例明确地图示并描述为进程、配置的介质或系统，但可以理解，对一种类型的实施例的讨论一般也可扩展到其他实施例类型。例如，结合图 3 的进程描述还帮助描述经配置的介质，并帮助描述如结合其他附图讨论的那些系统和制品等系统和制品的操作。并不能得出一个实施例中的限制一定要加进另一实施例中。具体而言，各进程不一定要限于在讨论诸如配置的存储器等系统或产品时提出的数据结构和安排。

[0091] 并非附图中所示的每一项目都需要存在于每一实施例中。相反，一实施例可以包含附图中未明确示出的项目。尽管此处在文字和附图中作为具体示例示出了某些可能性，但各实施例可以脱离这些示例。例如，一示例的具体特征可被省略、重命名、不同地分组、重复、以硬件和 / 或软件不同地实例化、或是出现在两个或更多示例中的特征的混合。在某些实施例中，在一个位置处示出的功能也可在不同位置提供。

[0092] 通过附图标记对全部附图作出了引用。在附图或文字中与给定的附图标记相关联的措辞中的任何明显的不一致性应被理解为仅仅是拓宽了该标记所引用的内容的范围。

[0093] 如此处所使用的，诸如“一”和“该”等术语包括了所指示的项目或步骤中的一个或多个。具体而言，在权利要求书中，对一个项目的引用一般意味着存在至少一个这样的项目，且对一个步骤的引用意味着执行该步骤的至少一个实例。

[0094] 标题是仅出于方便起见的；关于给定话题的信息可在其标题指示该话题的节之外找到。

[0095] 所提交的所有权利要求和摘要是该说明书的一部分。

[0096] 尽管在附图中示出并在以上描述了各示例性实施方式，但对本领域技术人员显而易见的是，可以作出不背离权利要求书中阐明的原理和概念的多种修改。尽管用结构特征和 / 或进程动作专用的语言描述了本主题，但可以理解，所附权利要求书中定义的主题不必限于上述权利要求中的具体特征或动作。给定定义或示例中标识的每一装置或方面不一定要存在于每一实施例中，也不一定要在每一实施例中都加以利用。相反，所描述的具体特征和动作是作为供在实现权利要求时考虑的示例而公开的。

[0097] 落入权利要求书的等效方案的含义和范围内的所有改变应被权利要求书的范围所涵盖。

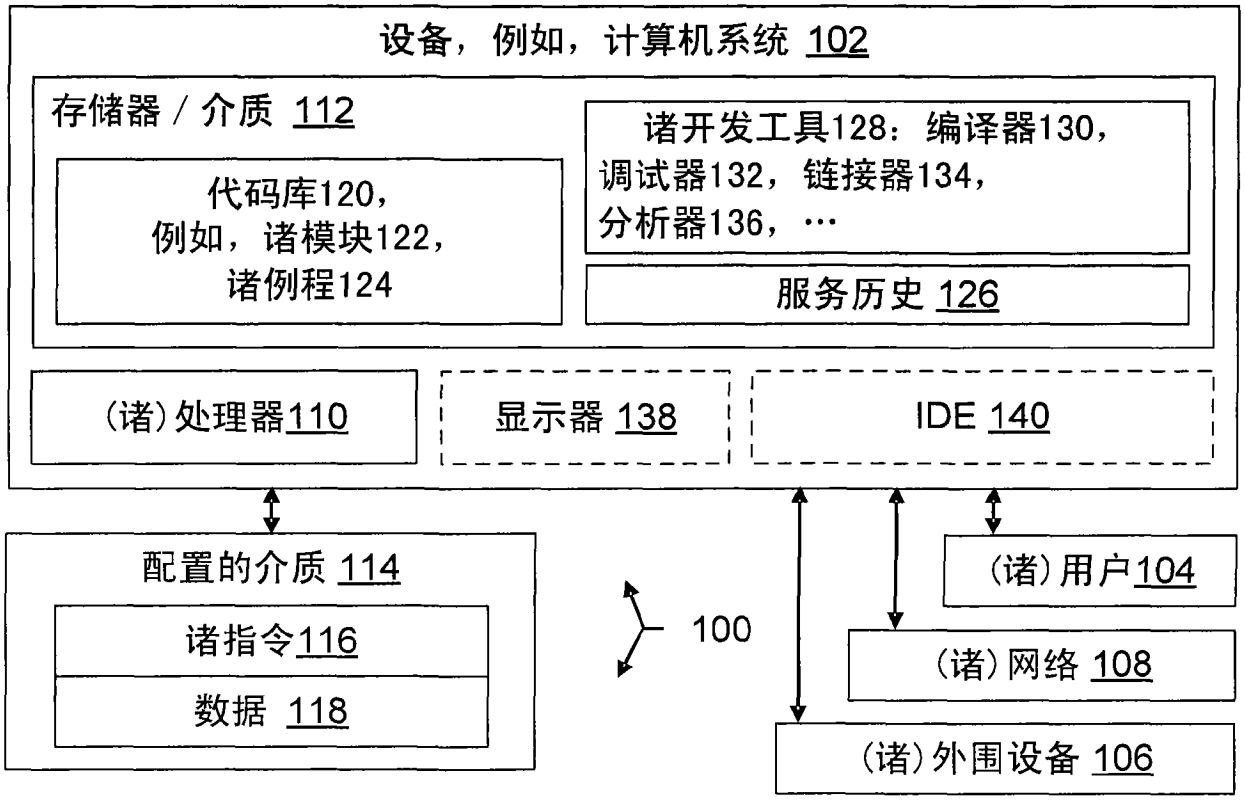

设备，例如，计算机系统 102
存储器 / 介质 112
代码库120，
例如，诸模块122，
诸例程124
诸开发工具128：编译器130，
调试器132，链接器134，
分析器136，…
服务历史 126
(诸)处理器110
显示器 138
IDE 140
配置的介质 114
诸指令116
数据 118
100
(诸)用户104
(诸)网络 108
(诸)外围设备 106

图 1

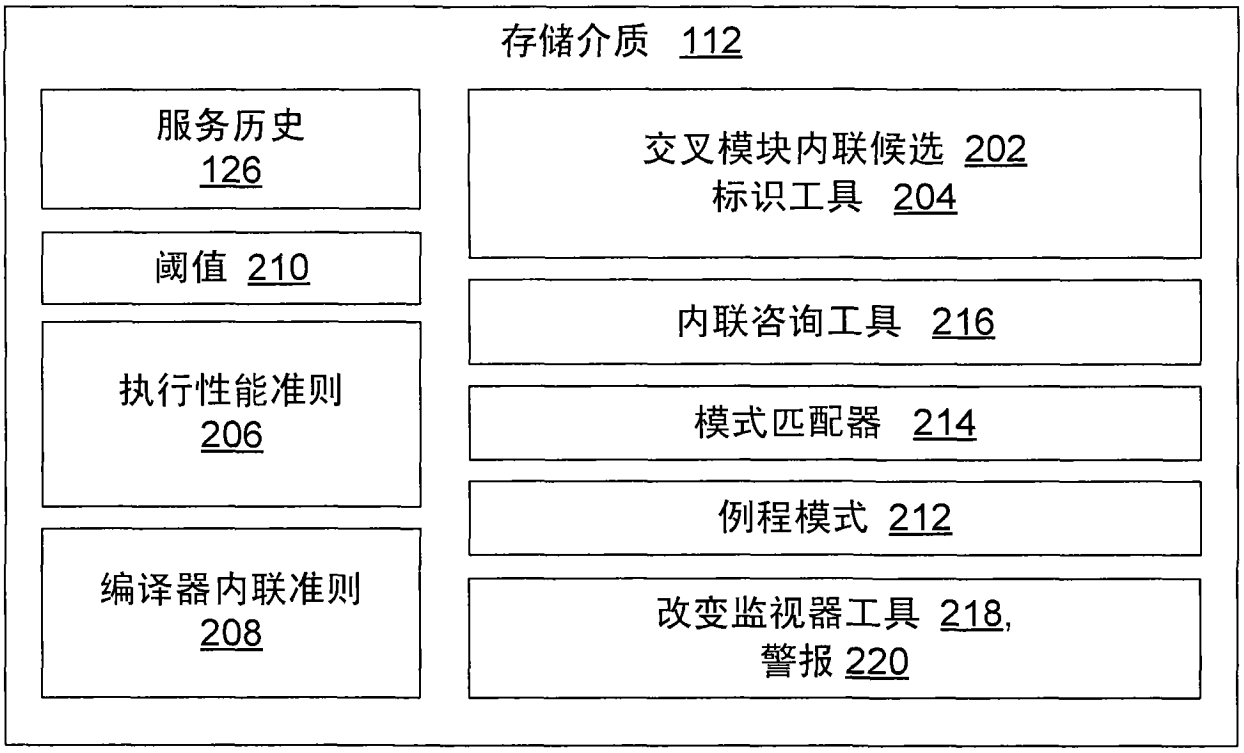

存储介质 112
服务历史
126
阈值 210
执行性能准则
206
编译器内联准则
208
交叉模块内联候选 202
标识工具 204
内联咨询工具 216
模式匹配器 214
例程模式 212
改变监视器工具 218，
警报 220

图 2


300
访问302服务历史
引用304执行性能准则
引用306编译器内联准则
标识308内联的候选
使用310内联咨询工具
排除312过分修改的例程
移除314性能不合格例程
模式匹配316例程代码
校验318是否过分修改
用属性标注320候选
使用322二进制码重写工具342
内联324(已标注)例程
定位326项目模式
接受328例程模式定义
接收330模式确认
查明332例程可用性
将改变当做错误来对待334(发出336警报)
维持338库存，重新生成340

图 3

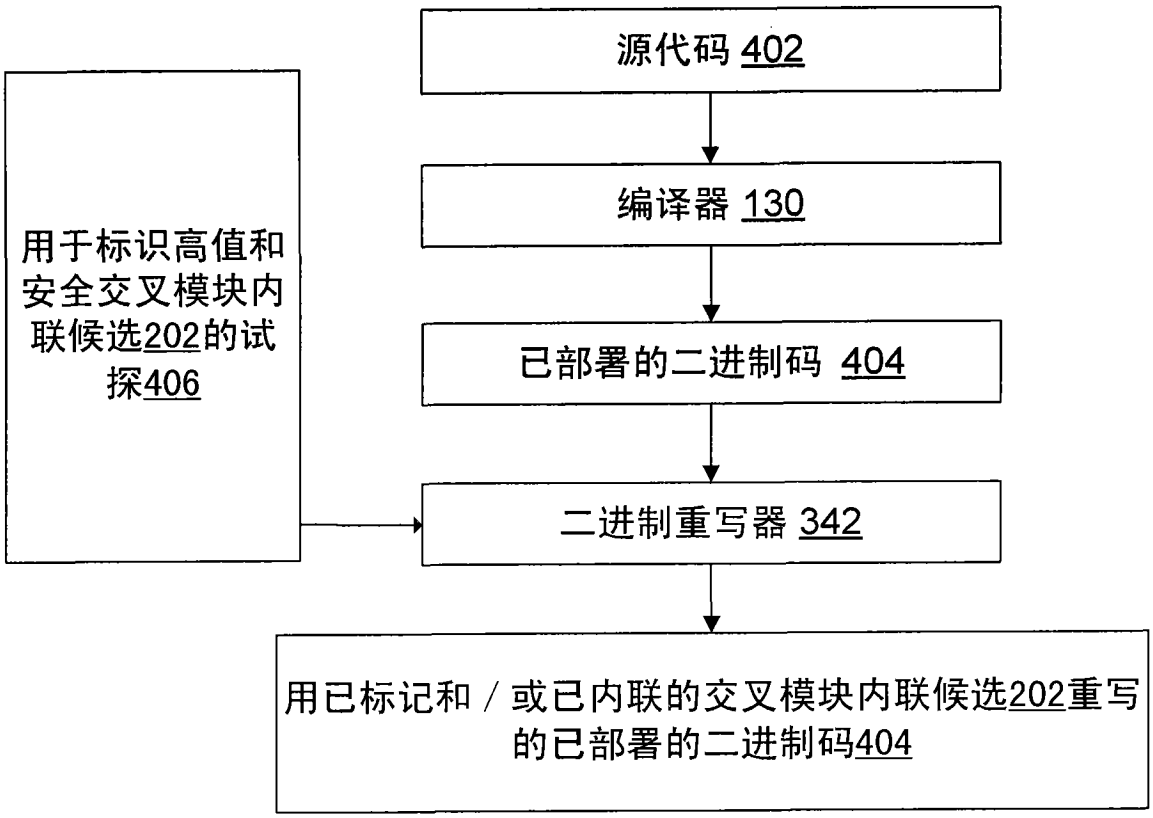

源代码 402
编译器 130
已部署的二进制码 404
二进制重写器 342
用于标识高值和安全交叉模块内联候选202的试探406
用已标记和/或已内联的交叉模块内联候选202重写的已部署的二进制码404

图 4