

[19] 中华人民共和国国家知识产权局



## [12] 发明专利申请公布说明书

[21] 申请号 200710003753.8

[43] 公开日 2007 年 10 月 17 日

[51] Int. Cl.  
G06F 12/08 (2006.01)  
G06F 9/46 (2006.01)

[11] 公开号 CN 101055543A

[22] 申请日 2007.1.24

[21] 申请号 200710003753.8

[30] 优先权

[32] 2006.2.23 [33] US [31] 11/360,351

[71] 申请人 国际商业机器公司

地址 美国纽约阿芒克

[72] 发明人 理查德·K·柯克曼  
保罗·L·戈特兰德  
迈克尔·J·科里根  
小乔治·D·蒂姆斯  
韦德·B·奥伦

[74] 专利代理机构 北京市柳沈律师事务所  
代理人 邸万奎 黄小临

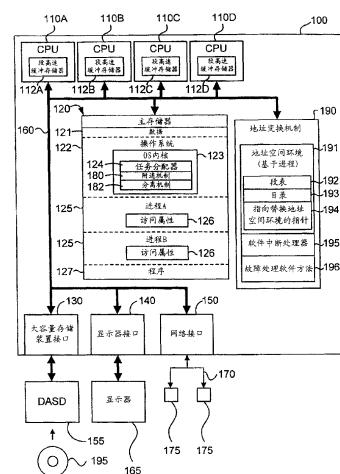
权利要求书 5 页 说明书 19 页 附图 7 页

### [54] 发明名称

用于访问另一个进程的进程本地存储装置的方法和设备

### [57] 摘要

操作系统内核包括附连机制和分离机制。另外，通过将进程标识为客户机进程或服务器进程的访问属性来标记进程。基于访问属性，操作系统内核取决于进程是客户机进程还是服务器进程而不同地布置进程本地地址空间。服务器进程可“附连”到客户机进程，并且引用客户机进程的本地存储装置的全部，如同客户机进程的本地存储装置是其自己的那样。服务器进程继续引用其自己的进程本地存储装置，但是另外，其可使用客户机进程的本地地址引用其它存储装置。当不再需要对其它存储装置的访问时，服务器进程可与客户机进程“分离”。一旦分离，便不再能够引用该其它存储装置。



1、一种设备，包括：

至少一个处理器；

耦接到所述至少一个处理器的存储器；

驻留在存储器中的客户机进程，该客户机进程仅可引用其自己的进程本地存储装置；

驻留在存储器中的服务器进程，该服务器进程可引用其自己的进程本地存储装置，并要求访问客户机进程的进程本地存储装置的能力；

驻留在存储器中、并由所述至少一个处理器执行的操作系统内核，该操作系统内核具有附连机制，在附连机制被调用时，允许服务器进程附连到客户机进程，使得服务器进程可使用客户机进程的进程本地地址来引用客户机进程的进程本地存储装置，并保留对其自己的进程本地存储装置的访问权。

2、如权利要求 1 所述的设备，其中，客户机进程具有与其关联的、具有指示将客户机进程约束为引用其自己的进程本地存储装置的属性值的访问属性，并且，服务器进程具有与其关联的、具有指示服务器进程需要引用客户机进程的进程本地存储装置的能力的属性值的访问属性。

3、如权利要求 2 所述的设备，其中，基于访问属性的属性值，操作系统内核为客户机进程布置与服务器进程的进程本地地址空间不同的进程本地地址空间。

4、如权利要求 3 所述的设备，其中，通过具有至少第一部分和第二部分的第一进程本地地址范围，定义客户机进程的进程本地存储装置，其中，客户机进程拥有第一进程本地地址范围的第一部分，且客户机进程不拥有第一进程本地地址范围的第二部分，并且其中，通过具有至少第一部分和第二部分的第二进程本地地址范围，定义服务器进程的进程本地存储装置，其中，该至少第一部分和第二部分分别对应于第一进程本地地址范围的第一部分和第二部分，并且其中服务器进程不拥有第二进程本地地址范围的第一部分，且服务器进程拥有第二进程本地地址范围的第二部分。

5、如权利要求 1 所述的设备，还包括驻留在存储器中、并由处理器执行的中断处理器和故障处理方法中的至少一个，其中，如果服务器进程正在引用其自己的进程本地存储装置，则中断处理器/故障处理方法尝试使用地址

空间环境的段表和目录中的至少一个来解决段故障，并且，如果服务器进程正在引用客户机进程的进程本地存储装置，则中断处理器/故障处理方法尝试使用替换地址空间环境的段表和目录中的至少一个来解决段故障。

6、如权利要求 5 所述的设备，其中，地址空间环境包括指向替换地址空间环境的指针，并且其中，一旦调用了附连机制，则操作系统内核便设置该指针。

7、如权利要求 6 所述的设备，其中，操作系统内核还包括分离机制，在分离机制被调用时，允许服务器进程与客户机进程分离，使得服务器进程不再能够引用客户机进程的进程本地存储装置，并且其中，一旦调用了分离机制，则操作系统便将该指针设置为空。

8、如权利要求 4 所述的设备，其中，操作系统内核还包括分离机制，在分离机制被调用时，允许服务器进程与客户机进程分离，使得服务器进程不再能够引用客户机进程的进程本地存储装置。

9、如权利要求 1 所述的设备，其中，操作系统内核还包括分离机制，在分离机制被调用时，允许服务器进程与客户机进程分离，使得服务器进程不再能够引用客户机进程的进程本地存储装置。

10、一种设备，包括：

至少一个处理器；

耦接到所述至少一个处理器的存储器；

驻留在存储器中的第一进程；

驻留在存储器中的第二进程；

可通过由至少一个处理器执行的至少一个指令而寻址的地址空间，该地址空间包括：

通过具有至少第一部分和第二部分的第一进程本地地址范围定义的第一进程的第一进程本地存储装置，其中，第一进程拥有第一进程本地地址范围的第一部分，且第一进程不拥有第一进程本地地址范围的第二部分；

通过具有至少第一部分和第二部分的第二进程本地地址范围定义的第二进程的第二进程本地存储装置，其中，该至少第一部分和第二部分分别对应于第一进程本地地址范围的第一部分和第二部分，并且第二进程不拥有第二进程本地地址范围的第一部分，且第二进程拥有第二进

程本地地址范围的第二部分。

11、如权利要求 10 所述的设备，其中，第一进程具有带有属性值“客户机”的访问属性，其中属性值“客户机”指示第一进程是被约束为引用第一进程本地存储装置的第一部分的客户机进程，并且其中，第二进程具有带有属性值“服务器”的访问属性，其中属性值“服务器”指示第二进程是需要引用第一进程本地存储装置的第一部分的能力、并且保留对其进程本地存储装置的拥有部分的访问权的服务器进程。

12、如权利要求 10 所述的设备，其中，第一进程本地地址范围的第一部分以及第二进程本地地址范围的第一部分各自定义比第一进程本地地址范围的第二部分以及第二进程本地存储装置地址范围的第二部分大的地址空间。

13、如权利要求 10 所述的设备，其中，第一进程本地地址范围以及第二进程本地地址范围各自具有与第一部分和第二部分互斥的第三部分，并且其中，第一进程和第二进程分别不拥有或不可使用第一进程本地地址范围的第三部分和第二进程本地地址范围的第三部分。

14、如权利要求 10 所述的设备，还包括驻留在存储器中、并由所述至少一个处理器执行的操作系统内核，该操作系统内核具有附连机制，在附连机制被调用时，允许第二进程附连到第一进程，使得第二进程可使用第一进程的进程本地地址来引用第一进程的进程本地存储装置的第一部分，并保留对其进程本地存储装置的拥有部分的访问权。

15、一种用于基于访问属性的属性值而布置进程本地地址空间的计算机实现的方法，该布置有助于一个进程访问另一个进程的进程本地存储装置，该方法包括以下步骤：

(A) 基于访问属性的属性值，确定进程是服务器进程还是客户机进程，其中，客户机进程仅可引用其自己的进程本地存储装置，服务器进程可引用其自己的进程本地存储装置，并需要访问客户机进程的进程本地存储装置的能力；

(B) 如果进程的属性值指示该进程是客户机进程，则布置进程本地地址空间，使得通过具有至少第一部分和第二部分的进程本地地址范围来定义客户机进程，其中，客户机进程仅拥有其进程本地地址范围的第一部分；

(C) 如果进程的属性值指示该进程是服务器进程，则布置进程本地地

址空间，使得通过具有至少第一部分和第二部分的进程本地地址范围来定义服务器进程，其中，所述第一部分和第二部分分别对应于客户机进程的进程本地地址范围的第一部分和第二部分，并且其中，服务器进程仅拥有其进程本地地址范围的进程本地地址范围的第二部分。

16、如权利要求 15 所述的计算机实现的方法，还包括以下步骤：

(D) 在执行进程中的任何程序之前，对每个进程重复步骤 (A) - (C)。

17、一种用于允许一个进程访问另一个进程的进程本地存储装置的计算机实现的方法，包括以下步骤：

(A) 创建客户机进程，其中客户机进程仅可引用其自己的进程本地存储装置；

(B) 创建服务器进程，其中服务器进程可引用其自己的进程本地存储装置，并需要访问客户机进程的进程本地存储装置的能力；

(C) 创建基于进程的地址空间环境，其包括段表、目录和指向替换地址空间环境的指针；

(D) 将服务器进程的地址空间环境中的指针设置为指向客户机进程的地址空间环境；

(E) 如果出现段故障，则确定服务器进程正在引用的故障地址是否落入客户机进程的地址空间中的客户机拥有的部分中；

(F) 如果在步骤 (E) 中确定故障地址未落入客户机进程的地址空间中的客户机拥有的部分中，则尝试使用服务器进程的地址空间环境的段表和目录中的至少一个来解决段故障；

(G) 如果在步骤 (E) 中确定故障地址落入客户机进程的地址空间中的客户机拥有的部分中，则尝试基于在步骤 (D) 中设置的指针，使用客户机进程的地址空间环境的段表和目录中的至少一个来解决段故障。

18、如权利要求 17 所述的计算机实现的方法，其中，步骤 (F) 包括以下步骤：

(F1) 尝试使用服务器进程的地址空间环境的段表来解决段故障；

(F2) 如果步骤 (F1) 失败，则尝试使用服务器进程的地址空间环境的目录来解决段故障。

19、如权利要求 18 所述的计算机实现的方法，其中，步骤 (G) 包括以下步骤：

- (G1) 尝试使用客户机进程的地址空间环境的段表来解决段故障；
- (G2) 如果步骤 (G1) 失败，则尝试使用客户机进程的地址空间环境的目录来解决段故障。

20、如权利要求 17 所述的计算机实现的方法，其中，步骤 (G) 包括以下步骤：

- (G1) 尝试使用客户机进程的地址空间环境的段表来解决段故障；
- (G2) 如果步骤 (G1) 失败，则尝试使用客户机进程的地址空间环境的目录来解决段故障。

21、一种计算机程序产品，用于允许一个进程访问另一个进程的进程本地存储装置，该计算机程序产品包括在计算机可读信号承载介质上提供的多个计算机可执行的指令，所述程序包括权利要求 1 至 20 中的任一方法的步骤。

## 用于访问另一个进程的进程本地存储装置的方法和设备

### 技术领域

本发明一般涉及数据处理领域。更具体地，本发明涉及计算机系统中的寻址方案的领域。

### 背景技术

自从计算机时代的来临，计算机系统已演变为可在很多不同设置中找到的极为复杂的装置。典型地，计算机系统包括硬件（例如，半导体、电路板等）和软件（例如，计算机程序）的组合。随着半导体工艺和计算机架构的进步使计算机硬件的性能推向更高，已为了利用硬件的更高的性能而发展出更复杂的计算机软件，从而导致今天的计算机系统比几年前强大得多。

计算机系统具有由计算机硬件定义的寻址能力。计算机系统的地址空间是可用来引用数据、指令等的地址的范围，并通过地址的大小（以位为单位）而被确定。地址大小是计算机系统的基础架构特征中的一个。早先，计算机系统是单用户计算机，其仅可同时处理单个任务，每当需要执行新任务时，便将所有数据映射到单个地址空间、并将数据调入和调出地址空间。之后，开发出支持多用户和进程的计算机。支持多进程的计算机系统必须管理进程之间的地址空间的分配。通常，每个进程具有对于该进程来说唯一的、其自己的工作数据。对于单处理器和多处理器计算机系统来说均是这样。

通常，计算机架构支持“进程本地寻址”，其中，仅在执行进程的环境（context）中解析存储装置访问。处理器使用进程唯一的段表（segment table）（其在本领域中还被称为“变换表”），以将进程本地地址变换为虚拟地址。两个不同的进程可使用相同的进程本地地址而引用存储装置，但是，因为操作系统内核利用用于每个进程的独立虚拟地址范围来填充（populate）段表，所以，引用了不同的虚拟地址，其随后导致对不同存储装置的访问。一般在行业中使用进程本地寻址。使用进程本地寻址机制来防止一个进程偶然或有意地访问另一个进程的存储装置。由此，期望操作系统内核将每个进程的工作数据布置为驻留在被进程本地地址引用的进程本地存储装置中。

然而，存在协作进程需要共享它们的工作存储装置的至少一部分的情形。由此，典型地，操作系统内核提供一个或多个共享机制，协作进程借此共享它们的工作存储装置的至少一部分。两个这样的共享机制分别被称为“共享存储器”和“存储器映射文件”。第一个共享机制，即共享存储器，是这样的技术：操作系统内核将每个协作进程的段表的一部分填充相同的虚拟地址范围，并提供后备存储装置，以支持那些虚拟地址。由此，协作进程可引用同一存储装置。此外，如果操作系统内核将相同的进程本地地址与每个进程的段表中的相同的虚拟地址相关联，那么，协作进程可使用相同的进程本地地址而引用相同的存储装置。

第二个共享机制，即存储器映射文件，是类似的技术，但是，在此情况下，通过存在于文件系统中某处的文件而提供后备存储装置。协作进程引用相同的存储装置，其是该文件的一部分。

这些共享机制是有用的，但需要进程之间的某种协作。在这一点上，应注意，进程的工作数据不位于共享区域中的一个中。进程必须显式地创建共享存储器对象、或映射该文件，并且随后，必须基于所得到的共享区域的地址而定义其工作存储装置中其想要共享的部分。并且，进程必须在对共享的存储装置进行寻址时进行协作。如果共享的存储装置中的任一个包含共享的存储装置的其它部分的地址，那么，所有进程必须能够使用相同的进程本地地址来引用共享的存储装置，或者，必须能够在引用共享的存储装置时向该地址施加偏移。另外，当将地址存储在共享的存储装置中时，必须考虑相同问题。

在某些情形中，难以或者不可能实现这些共享机制。进程经常必须根据共享机制中的一个的需要，而共享不能被组织的数据。例如，考虑执行用于客户机进程的给定服务器功能的服务器进程。该服务器功能可涉及从客户机进程的存储装置提取大量数据、或将大量数据存储到客户机进程的存储装置中。客户机进程的存储装置可在任何位置的事实引入了复杂性。例如，它可在其自己的工作存储装置中、或在调用它的过程的工作存储装置中；它可在与共享存储器对象或存储器映射文件相关联的存储装置中；它可在堆栈或静态存储装置中，或者它可在这些位置的某个组合中。客户机进程的存储装置可能需要遍历复杂的结构，这复杂的结构包括若干级别的间接寻址(indirection)和指针废弃，以定位要访问的所有存储装置。

可能存在用于处理这些复杂性的各种技术，但这些技术中的每个不期望地影响性能。在一个这样的技术中，使用共享存储器作为中间缓冲器。服务器进程从此中间存储器提取数据，或将数据置入此中间存储器，并且，客户机进程在调用服务器功能之前，将数据从其自己的工作存储装置复制到共享的区域中，或者，在调用服务器功能之后，将数据从共享的区域复制到其自己的工作存储装置中。与设置中间存储装置、以及副本复制（尤其是在涉及大量数据的情况下）有关的、与中间复制相关联的开销是不理想的。

在用于处理这些复杂性的另一种技术中，服务器进程使用操作系统内核功能，以从客户机进程的本地存储装置“获得”、或“置入”客户机进程的本地存储装置。取决于如何实现它，这可能还涉及数据的中间复制。无论如何，此类功能在本质上是非常通用的，例如，获得/放置在指定地址上开始的特定数目( $n$ )的字节，并且，此类功能不拥有用来遍历客户机数据结构的实际知识。结果，服务器进程需要对遍历的每一个分支调用此功能，并且，对性能的影响同样是不期望的。

存在对于以受控方式访问另一个进程的进程本地存储装置的增强机制的需要。

## 发明内容

根据本发明的优选实施例，操作系统内核包括附连机制和分离机制。另外，通过将进程标识为客户机进程或服务器进程的访问属性来标记进程。基于访问属性，操作系统内核取决于进程是客户机进程还是服务器进程而不同地布置进程本地地址空间。服务器进程可“附连”到客户机进程，并且，如同客户机进程的本地存储装置是其自己的那样，引用客户机进程的本地存储装置的全部。服务器进程继续引用其自己的进程本地存储装置，但是另外，其可使用客户机进程的本地地址而引用客户机进程的本地存储装置。通过带着附连请求进入到操作系统内核，服务器进程调用内核的附连机制来执行“附连”。一旦附连，便可以内核或用户模式来引用其它存储装置。当完成了访问客户机进程的本地存储装置的服务器进程时，通过带着分离请求进入到操作系统内核，服务器进程调用内核的分离机制来执行“分离”。一旦分离，便不再能够以内核或用户模式来引用其它存储装置。

从下面对附图中图解的本发明的优选实施例的更具体的描述中，本发明

的前述和其它特征、以及优点将变得清楚。

### 附图说明

下文中，将通过结合附图而描述本发明的优选示例实施例，其中，相同的附图标记表示相同的元素。

图 1 是根据本发明的优选实施例的计算机设备的框图。

图 2 是图解根据本发明的优选实施例的与几个示例并发进程中的每个相关联的各种机制的框图。

图 3 是图解根据本发明的优选实施例的几个示例并发进程中的每个的地 址空间内的分配的框图。

图 4 是图解根据本发明的优选实施例、用于基于访问属性而布置进程的 地址空间的方法的流程图。

图 5 是图解根据本发明的优选实施例、用于使服务器进程能够附连 (attach) 到客户机进程的方法的流程图。

图 6 是图解根据本发明的优选实施例用于处理段故障的方法的流程图。

图 7 是图解根据本发明的优选实施例、用于使服务器进程能够与客户机 进程分离 (detach) 的方法的流程图。

### 具体实施方式

#### 1.0 概述

本发明使服务器进程能够直接访问所有客户机进程的本地存储装置，如同客户机进程的本地存储装置是它自己的那样。服务器进程继续引用其自己的进程本地存储装置，但是另外，其可使用客户机进程的本地地址而引用客户机进程的本地存储装置。根据本发明的优选实施例，使用包括附连机制和分离机制的操作系统内核来提供此能力。另外，用将进程标识为客户机进程或服务器进程的访问属性来标记进程。基于访问属性，操作系统内核取决于该进程是客户机进程还是服务器进程而区别地布置进程本地地址空间。对于不熟悉本地寻址的一般概念的人员，下面连同在这里使用的定义而更详细地讨论此寻址方案。

在讨论寻址模式时，已在该领域中使用相同的术语来表示对于不同人来说不同的事物。例如，在本领域中最公知的术语“虚拟地址”表示这样的地

址：可被访问、或可通过执行地址计算而导出，并且，对于执行程序来说可见、并可被执行程序操作。然而，在PowerPC术语中，此类虚拟地址被称为“有效地址”。PowerPC术语中的“虚拟地址”是根据有效地址而生成的、并在PowerPC地址变换方案中被内部使用的地址。这通常就是如何在这里的“背景技术”部分中使用术语“虚拟地址”。一般说来，此类地址在后面被称为“内部地址”，以指示此地址对程序员来说不可见，而是由地址变换机制生成并使用的地址。PowerPC术语中的“真实地址”是用来访问真实（或物理）存储器的地址，并在这里被称为“真实地址”。

在现今的行业中通常使用的“进程本地寻址”中，仅在执行进程的环境中解析存储装置访问。硬件使用进程唯一的段表来将“进程本地地址”变换为内部地址。下面更详细地讨论的操作系统内核将分配进程本地地址空间、以及用于程序工作数据的存储装置。两个不同的进程可使用相同的进程本地地址来引用存储装置，但是，因为操作系统内核可对每个进程用不同的内部地址范围来填充段表，所以进程访问不同的存储装置。与防止一个进程偶然或有意地访问另一个进程的存储装置的传统的进程本地寻址机制不同，本发明使服务器进程能够以受控方式直接访问所有客户机进程的本地存储装置，如同客户机进程的本地存储装置是其自己的那样。根据本发明的优选实施例，操作系统内核在请求时允许此功能，并且，如果准许访问，则提供对另一个进程的存储装置的访问。

一般说来，对于这里的讨论以及现今的行业中所共识的，进程本地地址空间包括一组连续的段。硬件/软件使用段表逐段地将进程本地地址变换为内部地址。自此，硬件/软件继续使用传统的方式（其对于本发明的目的来说并不重要），而将内部地址解析为真实地址。

## 2.0 详细的描述

本发明的优选实施例提供了包括附连机制和分离机制的操作系统内核。另外，用将进程标识为客户机进程或服务器进程的访问属性来标记进程。基于访问属性，操作系统内核取决于该进程是客户机进程还是服务器进程而区别地布置进程本地地址空间。服务器进程可“附连”到客户机进程，并且引用所有客户机进程的本地存储装置，如同客户机进程的本地存储装置是其自己的那样。服务器进程继续引用其自己的进程本地存储装置，但是另外，它可使用客户机进程的本地地址来引用客户机进程的本地存储装置。通过带着

附连请求进入到操作系统内核，服务器进程调用内核的附连机制来执行“附连”。一旦附连，便可以内核或用户模式来引用其它存储装置。当完成了访问客户机进程的本地存储装置的服务器进程时，通过带着分离请求进入到操作系统内核，服务器进程调用内核的分离机制来执行“分离”。一旦分离，便不再能够以内核或用户模式来引用其它存储装置。

现在参照图 1，计算机系统 100 是根据本发明的优选实施例的设备的一个适用的实现。计算机系统 100 是增强型 IBM eServer iSeries 计算机系统。然而，本领域的技术人员将理解，还可将本发明的机制和设备应用于任意计算机系统，而无论该计算机系统是复杂的多用户计算设备、单用户工作站、还是嵌入式控制系统。如图 1 所示，计算机系统 100 包括多个处理器 110A、110B、110C 和 110D、主存储器 120、大容量存储装置接口 130、显示器接口 140、以及网络接口 150。通过使用系统总线 160 而互连这些系统组件。大容量存储装置接口 130 用来将大容量存储装置（如直接存取存储装置 155）连接到计算机系统 100。直接存取存储装置 155 的一个具体类型是可读且可写 CD-RW 驱动器，其可将数据存储到 CD-RW 195、并从 CD-RW 195 读取数据。

图 1 意图绘出计算机系统 100 在高层上的代表性的主要组件，应理解，各个组件可能具有比图 1 中表示的更大的复杂性，并且，这样的组件的数目、类型、以及配置可能变化。具体地，计算机系统 100 可能包含与所示出的数目不同的处理器。本领域的技术人员将理解，可使用具有一个或多个处理器的计算机系统来实践本发明。

根据优选实施例的主存储器 120 包含数据 121、操作系统 122、一个或多个进程 125、以及一个或多个程序 127。数据 121 表示用作到计算机系统 100 中的任意程序的输入、或来自计算机系统 100 中的任意程序的输出的任意数据，或者用于计算机系统 100 中的任意程序的工作存储装置。操作系统 122 是多任务操作系统，其在行业中被称为 OS/400 或 IBM i5/OS；然而，本领域的技术人员将理解，本发明的精神和范围不限于任一个操作系统。作为一般情况，操作系统 122 包括操作系统内核 123（在此也被称为“内核”）。

内核 123 被集成到操作系统 122 中，并向操作系统组件和其它程序（如程序 127）提供内核级服务。例如，内核 123 包括系统任务分配器 124，其被操作系统 122 调用，以使用本领域中公知的各种机制中的任一个将进程 125 分配到一个或多个处理器 110A、110B、110C 和 110D。内核 123 是安全、审

核、和控制的判优器。根据本发明的优选实施例，内核 123 提供两个新的机制：附连机制 180、以及分离机制 182。附连机制 180 允许服务器进程附连到客户机进程。分离机制 182 允许服务器进程与客户机进程分离。

内核 123 还创建并维护与每个进程 125 相关联的一组“属性”。从用户应用程序 127 导出所述属性中的一些，并从程序自身导出其它属性。传统属性的一些例子为用户 ID、组 ID、以及特权 (privilege)。另外，根据本发明的优选实施例，内核 123 还提供与每个进程 125 相关联的、在这里被称为“访问属性” 126 的新属性。如早先所提到的，访问属性 126 将进程 125 标记为客户机进程或服务器进程。

并且，根据本发明的优选实施例，内核 123 在执行程序 127 之前布置一个或多个进程 125。例如，在执行诸如“网页浏览器”或字处理程序的应用程序之前，内核 123 布置一个或多个进程 125。首先布置进程 125 (即，在程序开始执行之前)，这是因为，如下面更详细地讨论的那样，每个进程 125 必须通过访问属性 126 而被标记为客户机进程或服务器进程，于是，内核 123 获知在何处安排程序的进程本地存储空间。

每个进程 125 由一个或多个处理器 110A、110B、110C 和 110D 执行的程序指令组成。如前所述，由操作系统 122 创建一个或多个进程 125。这些进程可包括客户机进程和服务器进程。典型地，进程包含有关程序资源和程序执行状态的信息。

计算机系统 100 利用公知的虚拟寻址机制，其允许计算机系统 100 的程序进行操作，如同它们具有对大容量单个存储实体的访问权、而不是对多个较小的存储实体（如主存储器 120 和 DASD 装置 155）的访问权。因此，尽管将数据 121、操作系统 122、操作系统内核 123、系统任务分配器 124、附连机制 180、分离机制 182、一个或多个进程 125、访问属性 126、以及一个或多个程序 127 示出为驻留在主存储器 120 中，但本领域的技术人员将理解，这些项目不必同时完全包含在主存储器 120 中。

处理器 110A、110B、110C 和 110D 各自可由一个或多个微处理器和/或集成电路构成。优选地，处理器 110A、110B、110C 和 110D 是在行业中被共称为 PowerPC 架构的处理器架构的部件；然而，本领域的技术人员将理解，本发明的精神和范围不限于任一种处理器架构。处理器 110A、110B、110C 和 110D 执行存储在主存储器 120 中的程序指令。主存储器 120 存储处理器

110A、110B、110C 和 110D 可访问的程序和数据。当计算机系统 100 启动时，处理器 110A、110B、110C 和 110D 首先执行构成操作系统 122 的程序指令。操作系统 122 是管理计算机系统 100 的资源的复杂的程序。这些资源中的一些是处理器 110A、110B、110C 和 110D、主存储器 120、大容量存储装置接口 130、显示器接口 140、网络接口 150、以及系统总线 160。

优选地，每个处理器具有用于将进程本地地址变换为内部地址的变换信息的关联段高速缓冲存储器 112A、112B、112C 和 112D，如在下面更详细地讨论的那样。为了简化起见，一般在这里分别由附图标号 110 和 112 来指定处理器和段高速缓冲存储器。尽管对于每个处理器 110 示出了单个段高速缓冲存储器 112，但可能存在多级段高速缓冲存储器，或者可能根本不存在段高速缓冲存储器。处理器 110 还可具有用于其它目的的其它高速缓冲存储器，如存储器的高速缓冲存储器，其中的一些可专用于指令，而其它用于不可执行的数据。

地址变换机制 190 是提供使用地址空间来间接地访问真实存储器、或通过地址空间的间接引用而访问真实存储器的能力的机制。地址变换机制 190 是逻辑类型的实体，如在下面更详细地讨论的那样，其通过嵌入在处理器 110 和段高速缓冲存储器 112 内的硬件附件而扩充。根据本发明的优选实施例，地址变换机制 190 包括基于进程的地址空间环境 (context) 191，其将进程本地地址变换为内部地址。将地址空间环境 191 定义为数据结构和关联代码，该关联代码使用户所感觉的作为地址空间。如在下面详细地讨论的那样，地址空间环境 191 包括基于进程的段表 192、目录 193、以及指向替换 (alternate) 地址空间环境的指针 194。根据本发明的优选实施例，在将进程本地地址变换为内部地址时使用多级地址变换信息。以大小增大的次序列出的此地址变换信息包括段高速缓冲存储器 112、段表 192、以及目录 193。通常，地址空间环境 191 全部或至少部分地以软件实现。如前所述，进程本地地址空间包括一组连续段。如在本领域中惯用且公知的那样，段高速缓冲存储器 112 中的条目用来将进程本地地址变换为内部地址。段高速缓冲存储器 112 是在尝试将进程本地地址变换为内部地址时使用的第一级变换信息。通常，由硬件并行 (即，并发) 地搜索段高速缓冲存储器 112 中的条目，以便快速确定是否存在匹配。当在段高速缓冲存储器 112 中的条目中未发现进程本地地址时，出现段故障。如果出现段故障，则根据本发明的优选实施例、并且如下面更

详细地描述的地址变换机制 190 使用段表 192、目录 193、以及指向替换地址空间环境的指针 194，逐段地将进程本地地址变换为内部地址。自此，地址变换机制 190 继续使用传统的方式将内部地址解析为真实的地址，该传统的方式对于本发明的目的来说不重要。

如在现有技术中公知的那样，由硬件执行在段高速缓冲存储器 112 中的条目中的进程本地地址的搜索。如果该硬件在段高速缓冲存储器 112 中的条目中发现进程本地地址，则该硬件将进程本地地址变换为内部地址，随后将内部地址解析为真实的地址。典型地，如果该硬件在段高速缓冲存储器 112 中的条目中发现进程本地地址，则不通知软件。另一方面，如果该硬件未在段高速缓冲存储器 112 中的条目中发现进程本地地址，则出现段故障。在此异常情况下，取决于所使用的具体实现，如在下面更详细地讨论的那样，可能、或者可能不向软件通知出现段故障。

段高速缓冲存储器 112 未大到足以包含用于所有进程本地寻址段的变换信息。由此，如现今在行业中通常的那样，基于进程而维护具有较大的变换信息集的段表 192。如果在变换期间出现段故障，则生成中断，并且，软件中断处理器 195 快速地尝试根据段表 192 而定位变换信息，并将变换信息作为条目置入段高速缓冲存储器 112，并且，可重新执行引起段故障的指令。软件中断处理器 195 在本领域中是传统且公知的，并且由此，不在这里详细讨论。可替换地、且在本领域中也是公知的，可以硬件方式实现段表的搜索，一般不通知软件。例如，处理器 110 可获知段表位于何处。在该情况下，如果在变换期间出现段故障，则处理器将尝试在段表中找到变换信息，并将变换信息作为条目置入段高速缓冲存储器 112，并且，可重新执行引起段故障的指令。

目录 193 比段表 192 大，这是因为，其包含基于进程的、用于进程 125 的所有进程本地地址到内部地址变换信息。如果软件中断处理器 195 不能在段表中找到必要的变换信息，那么，调用长得的故障处理软件方法 196，以在目录 193 中查找变换信息，并且，在重新执行故障指令之前使用它来填充段高速缓冲存储器 112 和/或段表 192。故障处理软件方法 196 在本领域中是惯用且公知的，并且由此，不在这里详细讨论它。可替换地，在以硬件方式实现段表的搜索、且该硬件不能在段表中发现必要的变换信息的情况下，生成中断，并调用故障处理软件 196。

由此，根据本发明的优选实施例，当出现段故障时，软件（例如，软件中断处理器 195 和故障处理软件方法 196）用来自在段表 192 或目录 193 中找到的变换信息的条目来加载段高速缓冲存储器 112。并且，根据本发明的优选实施例，当条目不再被需要、或无效时，该软件从段高速缓冲存储器 112 中清除（flush）所述条目，如在下面更详细地讨论的那样。并且，根据本发明的优选实施例，处理器 110 不断引用并使用段高速缓冲存储器 112 中的条目，并且，只要变换信息的搜索成功，即在段高速缓冲存储器 112 中的条目中找到变换信息，就不用打搅来通知软件。然而，当搜索不成功时，硬件通知软件已出现了段故障。

如上所述，地址空间环境 191 包括基于进程的、到用附图标号 194 表示的替换地址空间环境的指针（也被称为“指针”和“替换地址空间环境指针”）。如在下面更详细地讨论的那样，在附连进程期间，将与服务器进程相关联的地址空间环境的指针 194 设置为指向与客户机进程相关联的地址空间环境。

尽管为了说明的目的而将地址空间环境 191、段表 192、目录 193、指针 194、软件中断处理器 195、以及故障处理软件方法 196 示出为驻留在地址变换机制 190 中，但本领域的技术人员将认识到，这些组件不需要一起驻留，并且，这些组件中的一个或多个可驻留在其它位置。例如，地址空间环境 191 可驻留在内核 123 中，而软件中断处理器 195、以及故障处理软件方法 196 可驻留在主存储器 120 中的其它位置。另外，但本领域的技术人员还将认识到，可组合这些组件，或者，可省略一个或多个组件。

尽管将计算机系统 100 示出为仅包含单个系统总线，但本领域的技术人员将理解，可使用具有多个总线的计算机系统来实践本发明。另外，在优选实施例中使用的接口各自包括分离的、全编程的微处理器，其用来从处理器 110A、110B、110C 和 110D 中卸载计算密集的处理。然而，本领域的技术人员将理解，可将本发明同等地应用于简单地使用 I/O 适配器来执行类似功能的计算机系统。

显示器接口 140 用来将一个或多个显示器 165 直接连接到计算机系统 100。可以是非智能（即，哑（dumb））终端或可全编程的工作站的这些显示器用来允许系统管理员和用户与计算机系统 100 通信。然而，注意，尽管提供显示器接口 140 来支持与一个或多个显示器 165 的通信，但计算机系统 100 不必需要显示器 165，这是因为，可经由网络接口 150 而出现与用户和其它

进程的所有所需交互。

网络接口 150 用来跨越网络 170 而将其它计算机系统和/或工作站(例如, 图 1 中的 175)连接到计算机系统 100。无论计算机系统 100 可如何连接到其它计算机系统和/或工作站, 无论使用现今的模拟和/或数字技术还是通过将来的某种联网机制而产生网络连接 170, 本发明均同等地适用。另外, 可使用很多不同的网络协议来实现网络。这些协议是允许计算机跨越网络 170 通信的专用计算机程序。TCP/IP (传输控制协议/因特网协议) 是适用的网络协议的例子。

在这一点上, 重要的是要注意, 尽管已经并将继续在全功能计算机系统的上下文中描述本发明, 但本领域的技术人员将理解, 本发明能够以各种形式、作为程序产品而分发, 并且, 无论用来实际实现该分发的信号承载介质的特定类型如何, 本发明均同等地适用。适用的信号承载介质的例子包括: 可记录类型的介质, 如软盘和 CD-RW (例如, 图 1 中的 195); 以及传输型介质, 如数字和模拟通信链路。

图 2 示出了图解与几个并发用户进程 214、216 和 218 中的每个相关联的各种机制的功能框图。包括访问属性 126、地址空间环境 191、段表 192、目录 193、以及指针 194 的这些机制存在于内核 123 和例如处理器 110 的硬件中, 并在内核 123 和例如处理器 110 的硬件上运行。每个进程具有与其相关联的指向“替换 (alternate)” 地址空间环境 191 (以及替换地址空间环境的段表 192 和目录 193, 其分别被称为“进程的替换段表”以及“进程的替换目录”) 的指针 194。例如, 与服务器进程相关联的指针 194 可指向与客户机进程 216 或客户机进程 218 相关联的地址空间环境 191。为了下面讨论的原因, 不允许与客户机进程 (例如, 客户机进程 216) 相关联的指针 194 指向与其它客户机进程 (例如, 客户机进程 218) 相关联的地址空间环境 191。同样地, 不允许与服务器进程 (例如, 服务器进程 214) 相关联的指针 194 指向与其它服务器进程相关联的地址空间环境 191。首先, 指针 194 为空, 以指示该进程目前不具有替换地址空间环境。在优选实施例中, 与客户机进程 216 (或客户机进程 218) 相关联的指针 194 总是保持为空, 这是由于, 客户机进程从不具有替换地址空间环境。

可通过“用户模式”程序以及具有全系统特权的“内核模式”程序 - 包括(图 1 中示出的)内核 123 和所关联的硬件 (其在图 2 中一起被表示为内核/

硬件 250) - 来访问进程 214、216 和 218。内核 123 还可访问与图 2 中示出的各种机制相关联的元描述对象和代码，其中图 2 中示出的各种机制与每个进程相关联。

如前所述，内核 123 提供与每个进程相关联的访问属性 126。根据本发明的优选实施例，进程 214、216 和 218 中的每个具有呈现两个值中的一个的访问属性 126。“客户机 (*Client*)”是由多数进程（示例为客户机进程 216 和 218）使用的缺省属性值。具有此属性的进程仅可引用它们自己的进程本地存储装置。“服务器 (*Server*)”是专用于需要引用客户机进程的进程本地存储装置的能力的进程的属性值。服务器进程 214 例示了这样的进程。

图 3 是图解进程 214、216 和 218 中的每个的地址空间内的分配的框图。与每个进程相关联的地址空间示例为用于服务器进程 214 的地址空间 342、用于客户机进程 216 的地址空间 344、以及用于客户机进程 218 的地址空间 346。在这里使用术语“地址空间”来表示进程本地地址范围在存储器上的映射。尽管在图 3 中示出，但内核/硬件 250 实际上不是地址空间的一部分，而是被示出用来图解内核/硬件 250 对进程 214、216 和 218 提供的基础 (underpinning) 和支持。例如，在请求时、并且一旦验证该请求通过了各种安全、审核和控制规则，内核 123 便设置影响内核或用户随后进行的与该地址空间相关的任何事物的数据结构。如图 3 所示，每个进程 214、216 和 218 被约束在其所拥有的其进程本地地址范围的量中。此约束在具有 32 位寻址的系统上可能是不可行的，这是因为，合理地，该进程可能已经需要使用其全部地址空间。然而，在例如具有 64 位寻址的系统的较大的系统上，典型地，将进程约束为使用比其全部进程本地寻址范围小的范围不是问题。

为了简单起见，并且非限制性地，假定仅允许进程 214、216 和 218 拥有大约它们各自的地址范围的一半。例如，这可通过允许目录 193 仅包含用于允许每个相应的进程 214、216 和 218 拥有的地址范围的变换信息，而相对简单地完成。此外，基于访问属性 126 的值，假定诸如客户机进程 216 和 218 的客户机进程仅分别拥有进程本地地址范围的“前一半”部分 302 和 304，并且，例如服务器进程 214 的服务器进程仅拥有进程本地地址范围的“后一半”部分 306。由此，诸如客户机进程 216 和 218 的客户机进程分别不拥有进程本地地址范围的“后一半”部分 312 和 314；并且，例如服务器进程 214 的服务器进程不拥有进程本地地址范围的“前一半”部分 316。另外，如图 3

所示，有可能包括每个相应的进程本地地址范围的可选的保留部分 326。如果被包括，则保留部分 326 未被使用，并为了可能的将来使用而被保留。

在以上描述中的进程本地地址空间被分割为前一和后一半的事实仅为了说明的方便。在优选实施例中，根本未使用地址空间中的一些（例如，保留部分 326），并且，不在客户机和服务器进程之间均匀地分割可使用的部分（例如，拥有的部分 302、304 和 306、以及未拥有的部分 312、314 和 316）。典型地，服务器进程是专用的，并且，通常不需要拥有对于其它进程必须可用的进程本地存储装置的全部。因此，典型地，服务器进程所拥有的地址空间的共享（例如，拥有的部分 306）将比客户机进程所拥有的地址空间的共享（例如，拥有的部分 302 和 304）小得多。

### 基于访问属性的进程创建

如前所述，内核在执行进程中的任何程序之前，开始该进程。必须通过访问属性而将每个进程标记为客户机进程或服务器进程，于是，内核获知在何处安排程序的进程本地存储装置。

现在，参照图 4，当内核基于期望的属性值确定进程是客户机进程还是服务器进程（步骤 410）时，根据本发明的优选实施例的用于布置进程的地址空间的方法 400 开始。在运行程序的时刻或之前指定访问属性。在进程创建时刻指定访问属性是优选的。创建进程机制在本领域中是公知的。可替换地，可在布置地址空间的任意时刻指定访问属性，例如，在 UNIX 系统的 exec 函数上。

如果属性值是客户机，则内核布置进程本地地址空间，于是，客户机进程仅拥有客户机进程的进程本地地址范围的第一部分（例如，图 3 中的“前一半”部分 302）（步骤 412）。在这一点上，内核布置进程本地地址空间，于是，客户机进程的进程本地地址范围的其余部分是未被拥有、或未被使用的。例如，在图 3 中示出的客户机进程 216 的情况下，内核布置进程本地地址空间 344，使得“后一半”部分 312 是未被拥有的，并且，如果存在，保留部分 326 是未被使用的。如果属性值是服务器，则内核布置进程本地地址空间，于是，服务器进程仅拥有服务器进程的进程本地地址范围的第二部分（例如，图 3 中的“后一半”部分 306）（步骤 414）。在这一点上，内核布置进程本地地址空间，于是，服务器进程的进程本地地址范围的其余部分是未被拥有、

或未被使用的。例如，在图 3 中示出的服务器进程 214 的情况下，内核布置进程本地地址空间 342，使得“前一半”部分 316 是未被拥有的，并且，如果存在，保留部分 326 是未被使用的。对每个进程重复步骤 410、412 和 414。一旦内核布置进程、并获知将程序的进程本地存储装置安排在何处，便可在该进程内开始程序执行（步骤 416）。

### 附连机制

如前所述，内核 123 提供允许服务器进程以受控方式附连到客户机进程的新的功能。例如，暂时返回到图 2，服务器进程 214 可通过调用内核的附连机制而附连到客户机进程 216。

现在参照图 5，当服务器进程调用附连机制（步骤 510）时，根据本发明的优选实施例的允许服务器进程以受控方式附连到客户机进程的方法 500 开始。更具体地，服务器进程调用附连机制，作为参数而将引用传递到客户机进程。通过带着附连请求进入到操作系统内核，服务器进程调用内核的附连机制来执行“附连”。内核在请求时允许此功能，并且，如果批准了访问（例如，内核可执行安全检查来确保服务器进程具有附连到客户机进程的权力），便经由附连机制而提供对客户机进程的存储装置的访问。在执行附连机制时，执行下面讨论的步骤（步骤 512-524）。这些步骤以其优选次序而被阐述。然而，必须理解，各种步骤可在与所示出不同的、相对于彼此的不同时刻上出现，或可同时出现。此外，本领域的技术人员将理解，可省略一个或多个步骤。在任意情形中，一旦附连，便可以内核或用户模式引用其它存储装置。注意，服务器进程也能够使用其自己的进程本地地址引用其自己的进程本地存储装置。

内核验证执行进程具有服务器属性、且引用进程具有客户机属性（步骤 512）。此步骤的要点是在于为了避免两个进程必须拥有进程本地地址范围的不同部分的冲突。内核还验证当前未将执行进程附连到其它进程（步骤 514）。

另外，内核执行确保服务器进程具有附连到客户机进程的权力所期望或必要的任何安全性检查（步骤 516）。期望以受控方式提供访问，以防止对客户机进程的存储装置的未授权访问。如果服务器进程具有附连到客户机进程的权力，则批准访问。另一方面，如果服务器进程缺少这样的权力，则拒绝访问。

如可能期望或需要的那样，内核创建审核服务器进程已获取了对客户机进程的本地存储装置的访问权的事实所需的信息（步骤 518）。例如，一些操作系统需要审核。

内核使客户机进程的地址空间环境中的引用计数递增（步骤 520）。这确保了：如果要终止客户机进程，则其段表和目录会继续存在，于是，仍可从服务器进程引用它们。

另外，优选地，内核创建与客户机进程的本地地址范围的所拥有部分相对应的、服务器进程的目录中的条目（步骤 522）。不是将这些进程本地地址与内部地址相关，此目录条目将客户机本地地址范围的全部拥有部分（否则其不被服务器进程拥有）与客户机地址空间环境相关。此条目对于故障处理来说不需要，但对于记帐（book-keeping）目的来说是期望的。例如，此条目会有助于添加使目录具体化（materialize）以查看其内容的功能。例如，使目录具体化使得有可能确定服务器进程是否具有对其它进程的存储装置的寻址能力。

内核还将服务器进程的替换地址空间环境指针指向客户机进程的地址空间环境（步骤 524）。

### 用于处理段故障的机制

图 6 是图解根据本发明的优选实施例、用于处理段故障的方法 600 的流程图。在执行用于处理段故障的机制时，执行下面讨论的步骤（步骤 602-650）。这些步骤以它们的优选次序被阐述。然而，必须理解，各种步骤可在与所示不同的、相对于彼此的不同时刻上出现，或可同时出现。此外，本领域的技术人员将理解，可省略一个或多个步骤。当出现段故障（步骤 602）时，方法 600 开始。更具体地，硬件执行了段高速缓冲存储器中的不成功的查找，并且，该硬件随后呈现出段故障。只要出现段故障，便确定该进程是否具有替换地址空间环境的确定（步骤 604）。更具体地，检查进程的地址空间环境指针，以确定其是否已被设置为非空值。如果确定进程不具有替换地址空间环境，中断处理器便尝试通过在该进程的段表中查找故障地址（步骤 608）来解决该故障。当然，本领域的技术人员将理解，可替换地，如上所述，可以硬件方式实现段表的搜索。随后，确定中断处理器（或硬件）尝试解析故障指令的成功（步骤 610）。如果中断处理器（或硬件）成功，便将来自进程

的段表的变换信息作为条目而提供给段高速缓冲存储器，将该条目加载到处理器中，并且随后，重新执行故障指令（步骤 612）。

另一方面，如果中断处理器（或硬件）不能使用段表解决该故障，则采用较长的路径来从该进程的目录解决故障（步骤 614）。在这一点上，调用传统的故障处理软件方法，以查找进程的目录中的变换信息。可替换地，在以硬件方式实现段表的搜索、且硬件不能在段表中找到必要的变换信息的情况下，生成中断，并调用故障处理软件。在任意情形中，随后确定故障处理软件方法尝试解析故障指令的成功（步骤 616）。如果故障处理软件方法不能够使用进程的目录来解决该故障，则返回错误（步骤 618）。另一方面，如果故障处理软件方法成功，则使用在目录中找到的信息来更新进程的段表，并且随后，重新执行故障指令（步骤 620）。本领域的技术人员将理解，步骤 620 中的故障指令的重新执行将再次导致段故障。然而，段故障的这个第二次出现将导致采用到步骤 612 的更短路径，其中，将变换信息作为条目下推（push down）到段高速缓冲存储器中，并且随后，成功地重新执行故障指令。可替换地，在步骤 620 中，可与更新进程的段表同时地、作为条目而将变换信息加载到段高速缓冲存储器中。

返回到步骤 604，如果确定进程具有替换地址空间环境，则内核确定故障地址是否落入地址空间的客户机拥有的部分中（步骤 630）。如果故障地址未落入地址空间的客户机拥有的部分中，则该方法返回到步骤 608。另一方面，如果故障地址落入地址空间的客户机拥有的部分中，则执行步骤 638-650（其一般分别对应于步骤 608-620 的替换地址空间环境版本）。

如果该地址在地址空间的客户机拥有的部分中，则中断处理器尝试通过在进程的替换段表中查找故障地址（步骤 638）来解决该故障。注意，在第一次服务器进程尝试使用客户机地址来引用存储装置时，将出现段故障，并且，中断处理器将尝试使用进程的替换段表来解决它。在引用客户机地址的新的段时，将出现后续故障。随后，确定中断处理器尝试解析故障指令的成功（步骤 640）。如果中断处理器成功，则将来自进程的替换段表的变换信息作为条目而提供给段高速缓冲存储器，将该条目加载到处理器中，并且随后，重新执行故障指令（步骤 642）。在以硬件方式实现进程的段表的搜索的替换情况下，必须在步骤 642 中将进程的替换段表中的条目复制到进程的段表中，使得该硬件可找到该副本。

另一方面，如果中断处理器不能使用进程的替换段表来解决该故障，则采用较长的路径来从该进程的替换目录解决该故障（步骤 644）。在这一点上，调用传统的故障处理软件方法，以查找进程的替换目录中的变换信息。可替换地，在以硬件方式实现段表的搜索、且硬件不能在进程的段表中找到必要的变换信息的副本的情况下，生成中断，并调用故障处理软件。在任意情形中，随后确定故障处理软件方法尝试解析故障指令的成功（步骤 646）。如果故障处理软件方法不能够使用进程的替换目录来解决该故障，则返回错误（步骤 648）。另一方面，如果故障处理软件方法成功，则使用在进程的替换目录中找到的信息来更新进程的替换段表，并且随后，重新执行故障指令（步骤 650）。并且，本领域的技术人员将理解，步骤 650 中的故障指令的重新执行将再次导致段故障。然而，段故障的这个第二次出现将导致采用到步骤 642 的更短路径，其中，将变换信息作为条目下推到段高速缓冲存储器中，并且随后，成功地重新执行故障指令。可替换地，在步骤 650 中，可与更新进程的替换段表同时地、作为条目而将变换信息加载到段高速缓冲存储器中。

### 分离机制

如上所述，内核 123 还提供允许服务器进程与客户机进程分离的新功能。例如，暂时返回到图 2，服务器进程 214 可通过调用内核的分离机制而与客户机进程 216 分离。

现在参照图 7，当服务器进程调用分离机制（步骤 710）时，根据本发明的优选实施例的用于允许服务器进程与客户机进程分离的方法 700 开始。当不再需要对客户机进程的本地存储装置的访问时，服务器进程调用此功能。当完成访问客户机进程的本地存储装置的服务器进程时，通过带着分离请求进入到操作系统内核，服务器进程调用内核的分离机制来执行“分离”。在执行分离机制时，执行下面讨论的步骤（步骤 712-722）。这些步骤以其优选次序被阐述。然而，必须理解，各种步骤可在与所示出不同的、相对于彼此的不同时刻上出现，或可同时出现。此外，本领域的技术人员将理解，可省略一个或多个步骤。在任意情形中，一旦分离，便不再能够以内核或用户模式引用该其它存储装置。在这一点上，服务器进程仅可访问其自己的本地存储装置。

内核验证服务器进程附连到客户机进程（步骤 712）。内核还将服务器进

程的替换地址空间环境指针设置为空（步骤 714）。另外，内核从服务器进程的目录中移除在步骤 522（如图 5 所示）中创建的、将客户机进程本地地址与客户机进程的地址空间环境相关的条目（步骤 716）。

内核从所有处理器的段高速缓冲存储器中清除当前运行该进程的线程的条目（步骤 718）。并且，在以硬件方式实现服务器进程的段表的搜索的替换情况下，内核还使服务器进程的段表中的相关条目无效。这是因为，在该替换情况下，如上所述，客户机进程的段表中的条目可能已被复制到服务器进程的段表中。

对于通常的清除，本领域的技术人员将理解，当一个进程正在观察（即，附连）另一个进程时，如果被观察的进程对被观察的进程的地址空间做出改变，则对于观察该进程的所有进程，内核有责任进行特定级别的通知、段高速缓冲存储器的清除等。另外，正在进行观察的进程将使用被观察的进程已被移除或被替换为不同进程的陈旧（stale）地址变换信息。换句话说，正确的数据将不会在正在进行观察的进程的视图中被查看到，即，正在进行观察的进程将查看到与被观察的进程不同的数据。通常，当进程从地址空间移除段或存储器时，在本领域中公知的是，向可能正在运行该进程的其它线程的处理器广播。根据本发明的优选实施例，不仅向可能正在运行该进程的其它线程的处理器、还向可能正在运行观察进程的处理器作出清空变换信息的广播。

现在，回来参照图 7，内核还将客户机进程的地址空间环境中的引用计数递减（步骤 720）。如果客户机进程已经被终止，则现在这将允许消灭其进程本地寻址结构的全部。

如可能期望或需要的那样，内核创建审核服务器进程不再具有对客户机进程的本地存储装置的访问权的事实所需的信息（步骤 722）。

本领域的技术人员将理解，在本发明的范围内，很多变化是有可能的。例如，在优选实施例中，仅存在两类进程、以及它们附连的一种方式。然而，本领域的技术人员将理解，存在此机制的其它扩展。这些扩展包括：客户机进程也向回附连到相同的服务器进程、或附连到不同的服务器进程的能力。另一个扩展涉及多种类型的进程，其中，每个种类拥有进程本地地址范围的唯一部分，并且，任意种类的进程将能够附连到任意其它种类的进程。在另一个扩展中，如果每个进程具有到多个替换地址空间环境的多个指针，则可

将一个进程同时附连到多个其它种类的进程。并且，为避免冲突，每个种类的进程拥有进程本地地址范围的唯一部分。由此，尽管已通过参照本发明的优选实施例而具体地示出并描述了本发明，但本领域的技术人员将理解，可在其中做出这些和其它形式和细节上的改变，而不会背离本发明的精神和范围。

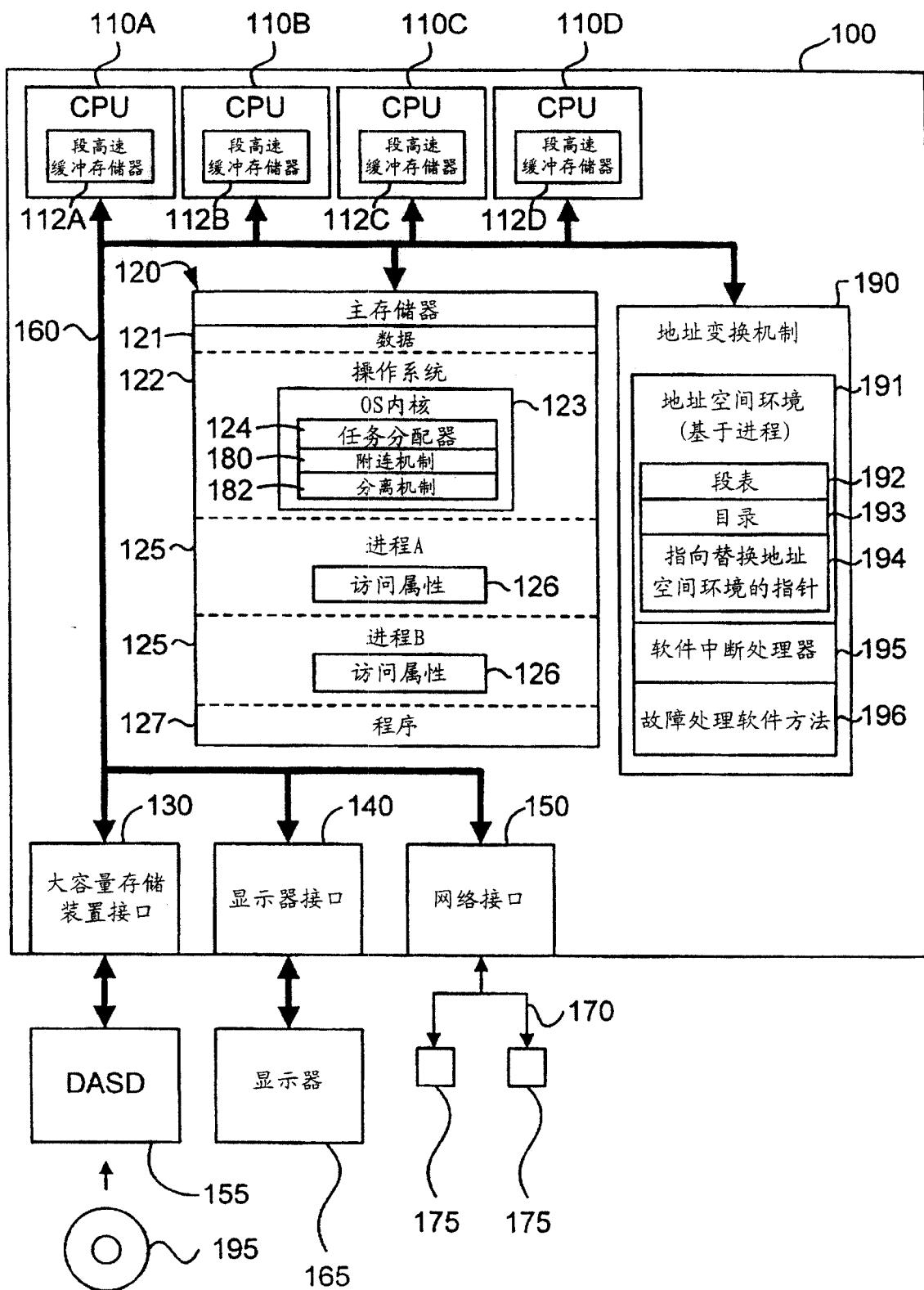


图 1

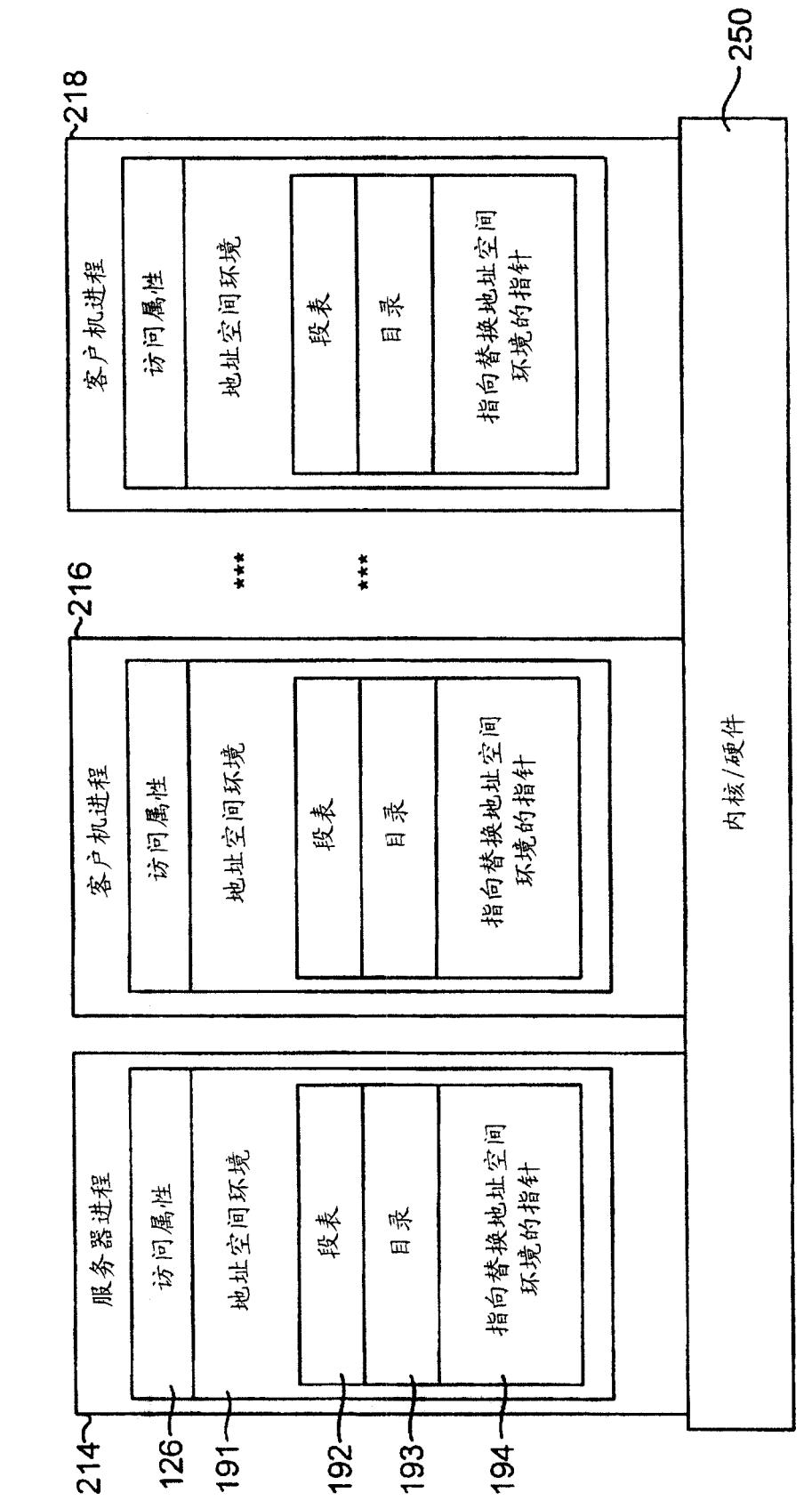


图 2

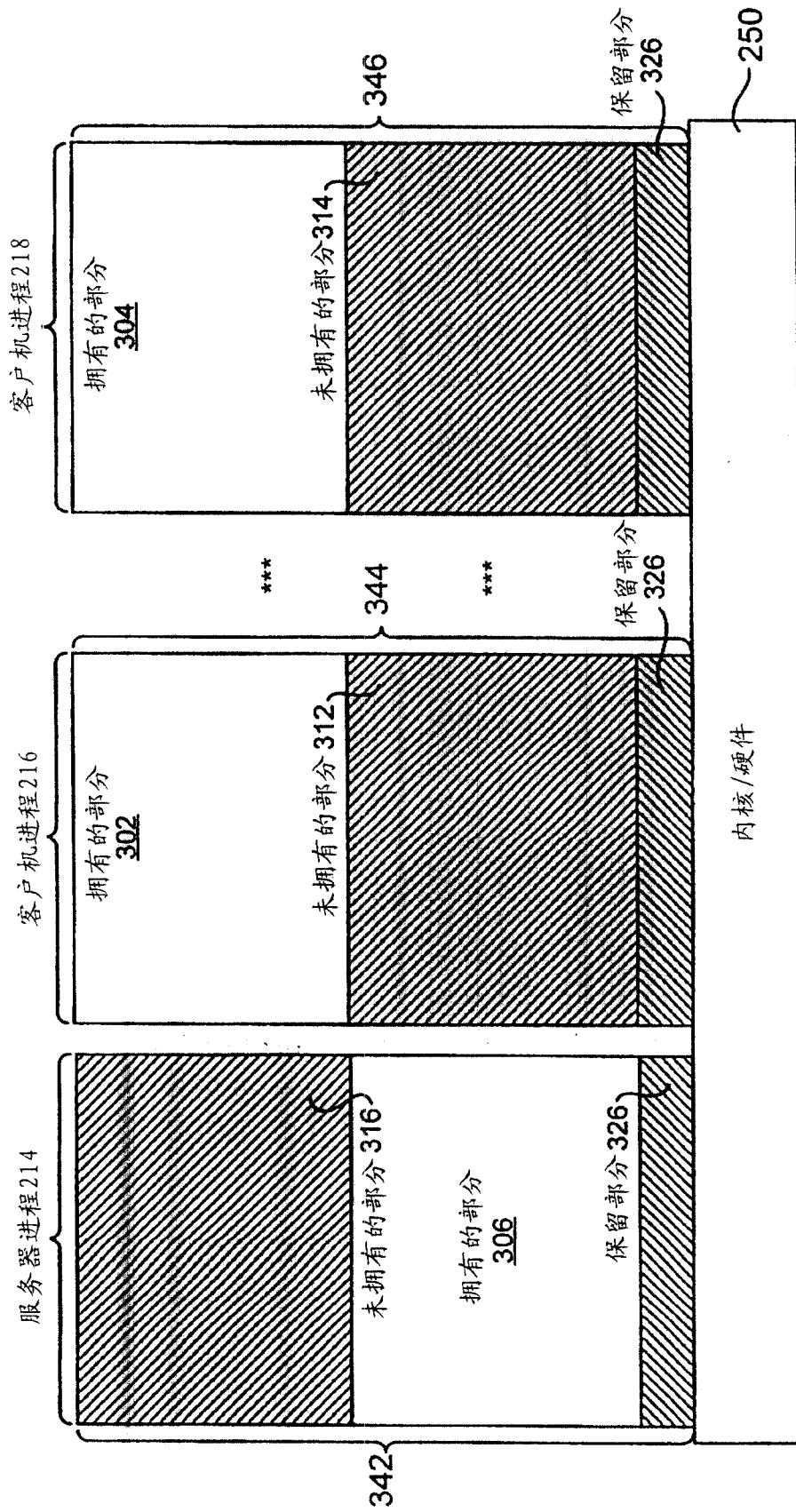


图 3

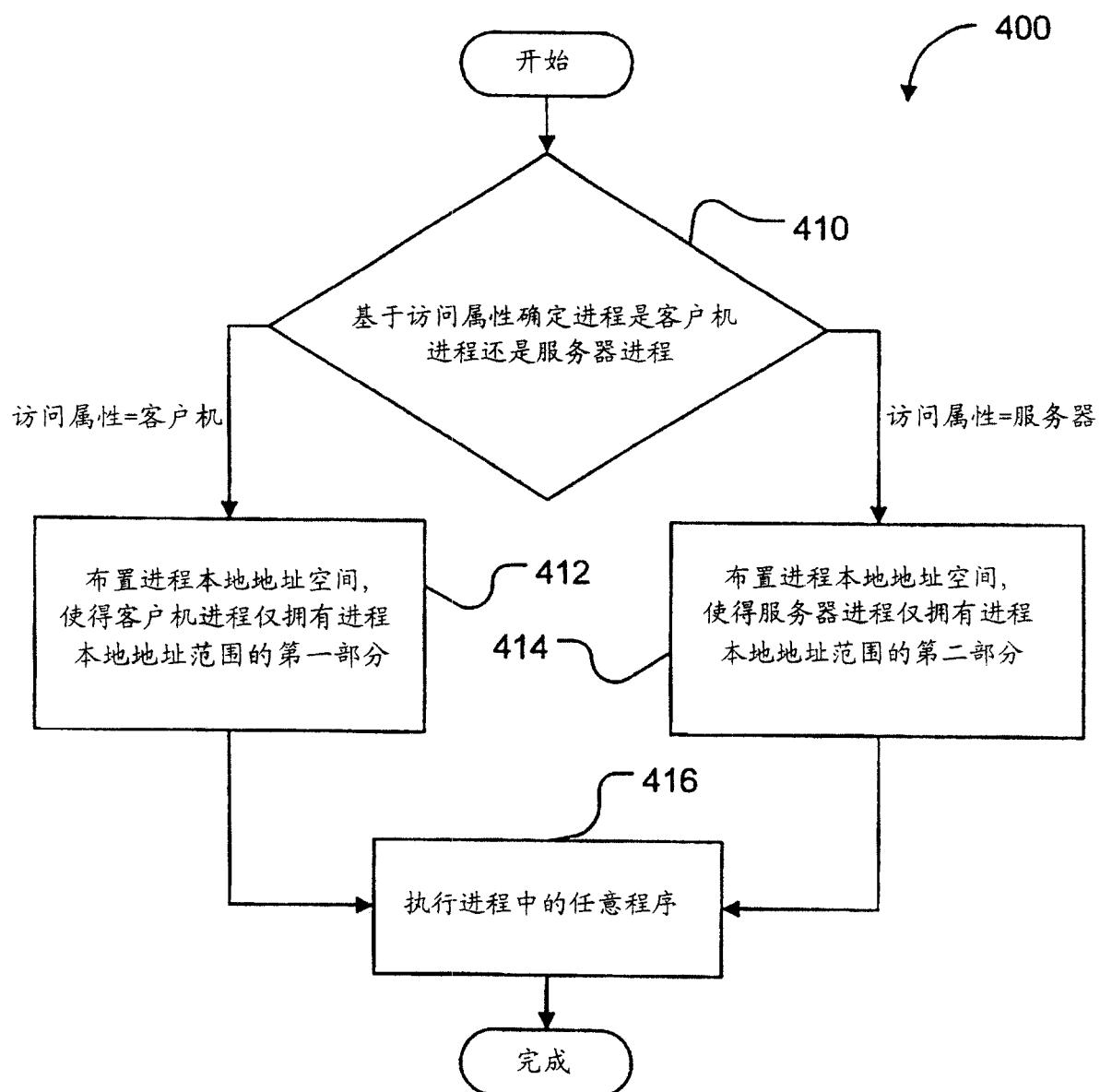


图 4

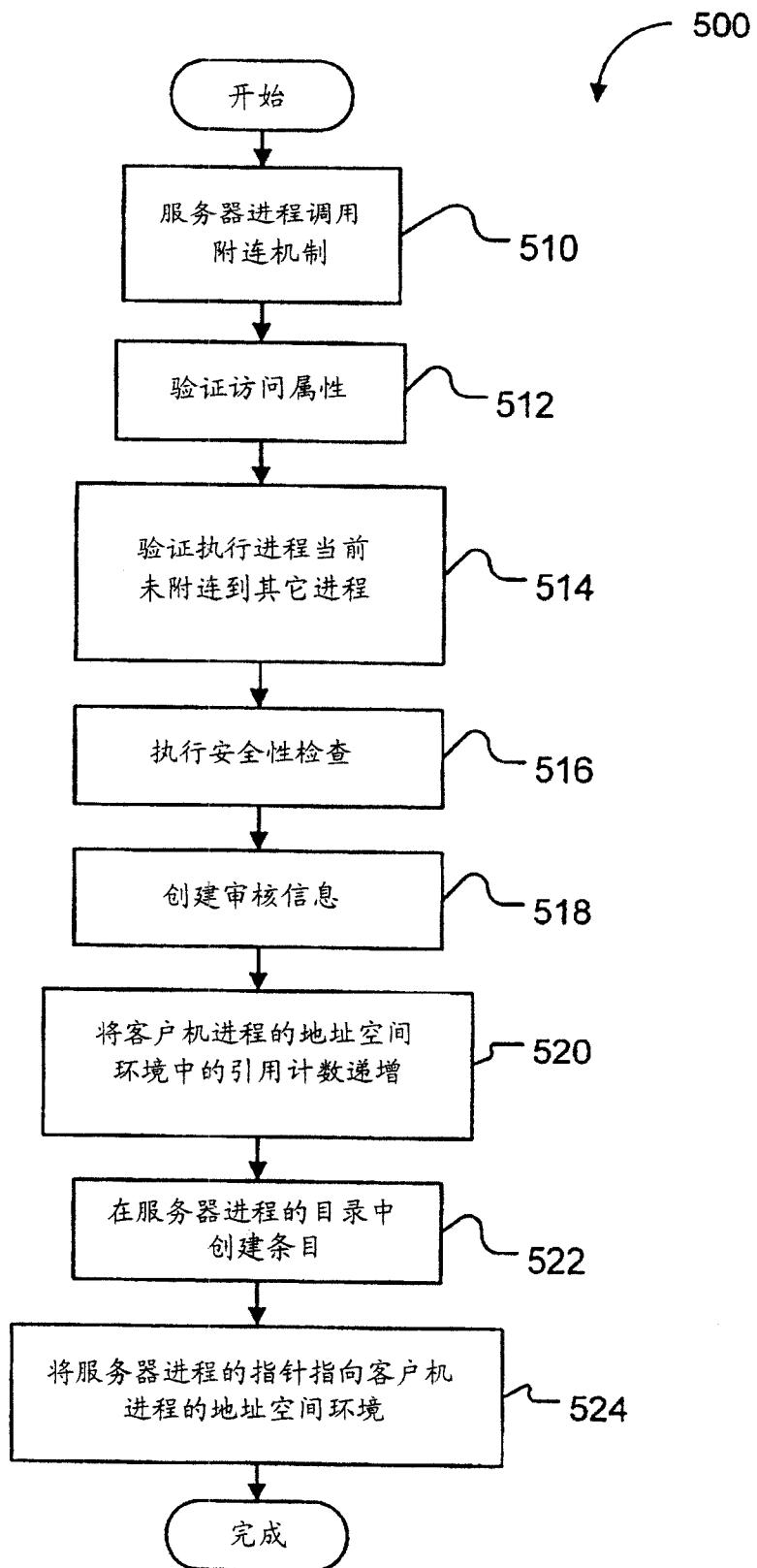


图 5

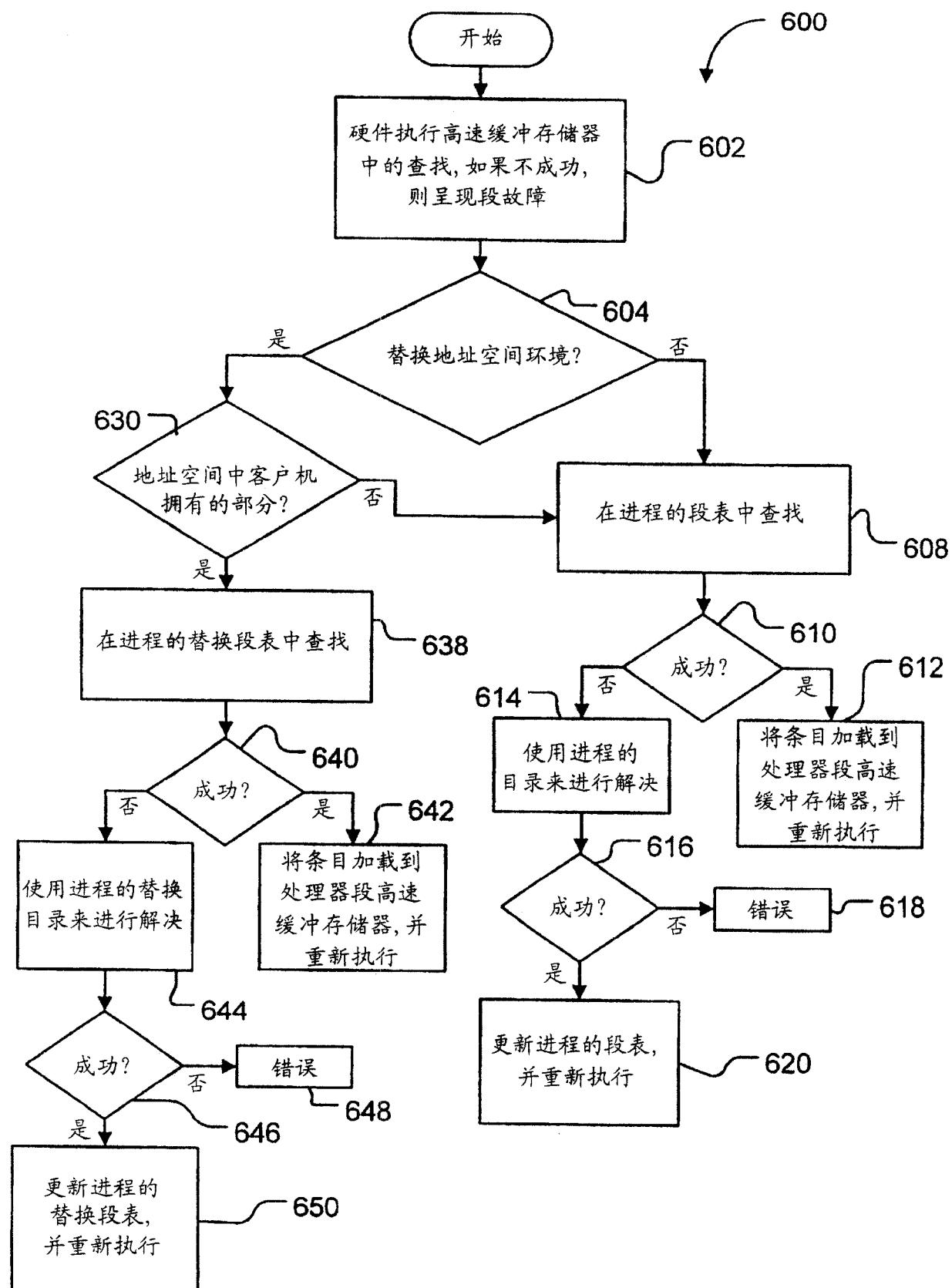


图 6

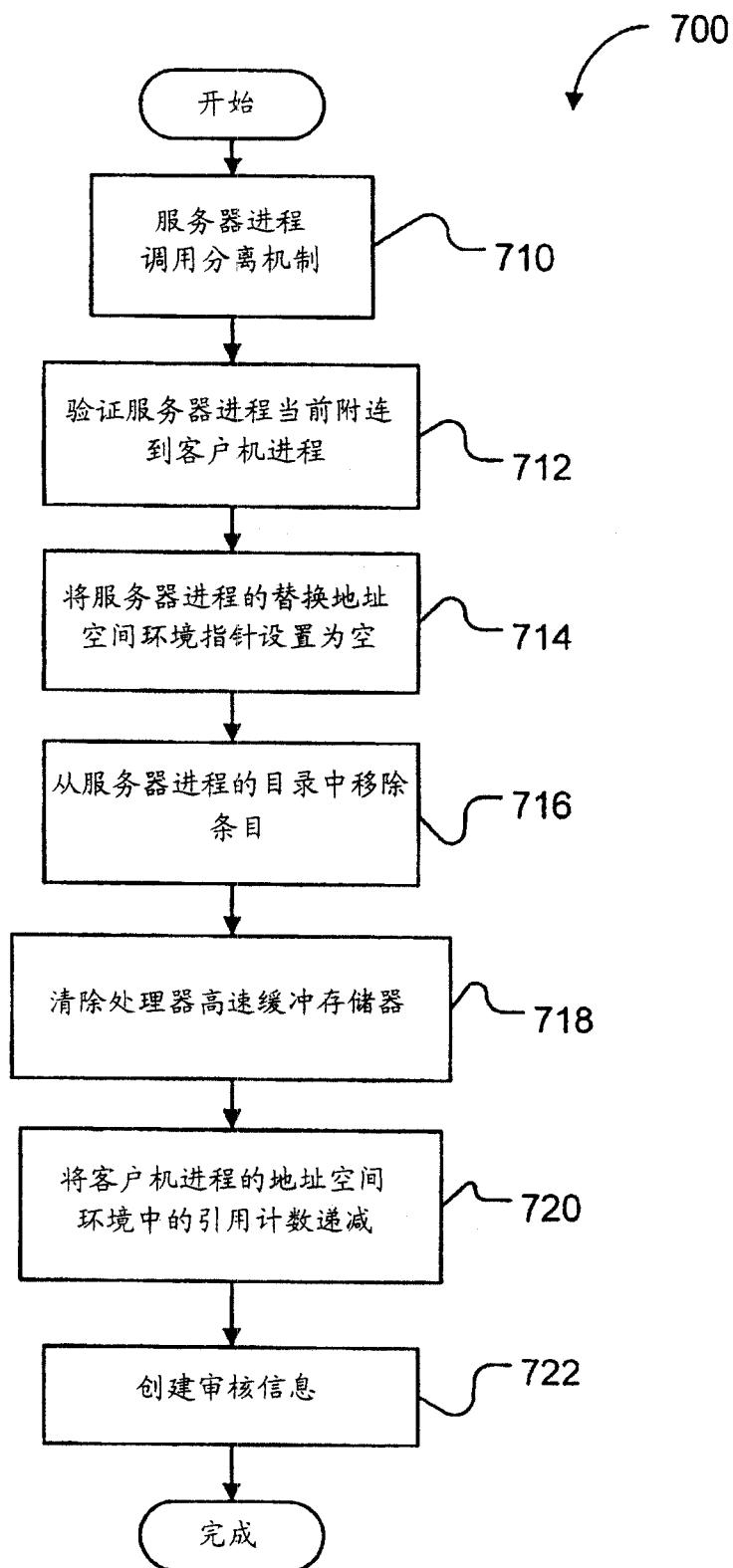


图 7