计算机科学 2000Vol. 27№. 1

操作系统体系结构研究

The Study of Operating System Architecture

潘 清
（总装备部指挥技术学院 北京 101416）

Abstract The development of operating system theory and practice is always behind the development of hardware technology and can't meet the requirement of the application. The microkernel technology, invented in the late of 80, has made a great progress and brought hopeful light on the development of operating system. Now we can see that operating system research has entered into a situation of less excited. In this papaer, through the analysis of two operating system architectures and the implementation of microkernel technology, the author points out the weakness of the microkenel technology. By using virtual computing theory, the author points out the fault areas in the microkernel technology research, describes the reasons of why microkernel technolgoy can not make further progress and the strategy we should take in the research of operating system.

Keywords Operating system, Architecture, Microkernel, Monolithic kernel

1 引言

操作系统作为计算机系统中最重要的一种系统软件，负责管理计算机系统资源，为用户使用计算机提供了方便、快捷的手段。目前还没有有关操作系统的精确的定义。在谈到操作系统时，人们总是用用户环境观点、或虚拟机观点、或资源管理观点，从不同的侧面来说明操作系统的基本特征。操作系统的理论和实践也是围绕着这些观点进行的，

从中断、多道程序设计等概念的提出，到进程概念的建立和实现，建立了现代操作系统的基本框架。操作系统作为计算机系统资源的管理者，负责管理计算机系统内部的资源，如处理器、内存、磁盘设备、其他I/O设备等。虚拟对象概念的提出为操作系统管理这些系统资源提供了有力的工具。通过虚拟处理机用户可以分时共享处理机的资源，通过虚拟地址空间用户可以方便地共享物理存储器空间，并且用户所使用的地址空间可以远远大于物理存储器空间的资源。

应用需求的驱动，硬件水平的提高，促使着操作系统不断向前发展。从早期的单用户操作系统，到批处理、分时、实时操作系统，到目前的网络操作系统、分布式操作系统，操作系统的理论和实践不断丰富和发展，特别是80年代末，提出的微内核技术，使操作系统的理论和实践进入了一个新的发展时期。在微内核技术的研究基础上又提出了所谓的超微内核结构、可扩展内核技术、外核体系结构和垂直型内核体系结构。但是，到了90年代中期以后，这些技术的影响力越来越小了，甚至出现了停滞不前的局面。而且，从总的情况来看，操作系统的理论和实践的发展现状明显滞后于应用需求和硬件的发展，操作系统理论和实践之间也存在着比较严重的脱节现象。因此，操作系统研究向何处去的问题就摆在了我们的面前。

作者在文[6]中通过分析计算机科学中大量出现的与虚拟对象相关的概念，提出了一个新的、用于对计算的本质进行再认识的概念--虚拟计算，得出了虚拟计算的本质就是资源共享的结论，并用虚拟计算的思想分析了当前计算机科学理论和实践发展过程中所遇到的一些有代表性的问题，得出了一些非常有意义的结论。同样，在操作系统的发展过程中也遇到了许多非常复杂的问题，本文就是试图采用虚拟计算的观点来讨论操作系统理论和实践发展过程中出现的这些问题。

2 虚拟计算的基本概念

如果将计算机科学中的所有与虚拟相关的概念的总和定义成虚拟计算，那么，我们就可以得出，抽象只是计算的一种表现形式，资源共享才是计算的本质特征。计算机科学理论和实践的发展就是随着资源共享的数量级的变化而经历了从量变到质变的发展过程。如何提供更高的资源共享可以归结到以下两个方面：(1)进一步增加可共享的资源的数量；(2)通过有效的机制在现有的资源上提供更好的服务的要求。

资源的共享程度可以通过更细粒度的机制来提高，例如，操作系统中的线程机制就是通过提供比进程粒度更细的调度单位，有效地减少系统资源的使用状况，从而达到提高整个系统效率的目的。资源的共享程度也可以通过更加集成化的机制来提高。例如，目前新出现的单片系统(System on a Chip)，将传统的中央处理器、内存、I/O 控制集成在同一个芯片上，从而降低了系统的开销，提高了系统工作效率。

进一步的资源共享需要进一步的数字化理论和技术。数字化具有广泛的定义。可以认为，任何一个软件的实现过程就是一次数字化的过程。例如，从进程概念的提出到进程的实现，这个过程的本身就是一个很好的数字化过程。新的模型的产生过程就是数字化过程。从 MPEG2 到 MPEG3 的发展，从 DVD 到数字化电视的发展过程中，可以清楚地看到数字化技术在资源共享方面所起的重要作用。

进一步的资源共享需要提供新的共享介质，从当前的 ATM 到同步光纤通讯，从半导体芯片到铜介质芯片，都将对资源共享产生革命性的推动作用。新介质的产生过程同样伴随着大量的数字化过程。

如果我们站在更高的层次上看操作系统，那么传统的用户观点、虚拟机观点、作业组织观点和资源管理观点都可以用资源共享的观点来概括。我们不仅可以用资源共享的概念来统一对操作系统的认识，而且还抓住了操作系统的本质特征。通过这个本质特征，我们可以更好地理解在操作系统理论和实践中遇到的各种问题。

3 典型的操作系统体系结构特征

从操作系统的发展来看，最有影响的操作系统体系结构有两种，传统的单一内核体系结构和微内核体系结构。通过分析这两种完全不同的结构的发展，可以更好地理解和解释操作系统理论和实践的发展过程中出现的问题。

3.1 商用操作系统的结构特征

从商用操作系统的发展来看，一个突出特点是，操作系统的功能迅速增加。目前，Microsoft 公司的 Windows NT 操作系统的代码规模已经超过了 5000 万行。不仅是应用程序的规模在急剧增长，操作系统内核的规模也在急剧扩张。这种扩张表现在操作系统内核的每一个方面。

在处理机管理方面，早期的操作系统只负责管理单个处理机资源，目前的操作系统需要管理大量的处理机资源。早期的操作系统中只包含了少数几种调度算法，目前的操作系统中包含了大量的调度策略。进程间的通讯手段也从早期的信号机制发展到消息机制、共享存储器机制、信号灯机制。粗粒度的进程机制发展到了细粒度的线程机制。

在存储管理方面，操作系统从没有虚拟存储管理系统到有虚拟存储管理系统，发展到现在的分布式存储管理系统。

在文件管理方面，操作系统从管理单一的文件系统，发展到采用虚拟文件系统技术来管理多个文件系统。从本地文件系统发展到网络文件系统、远程文件系统以及目前的分布式文件系统。

在设备管理方面，操作系统从管理简单的、数量较少的输入输出设备，到种类繁多的外部设备，从复杂的外部设备配置和安装到即插即用的设备驱动，设备热可插拔技术。

在计算机网络方面，在早期的操作系统中，计算机网络是和操作系统内核分离的，操作系统只提供简单的文件传输功能，之后，不同公司的操作系统实现了各自的网络协议，实现了同种计算机之间的互连。但是，每个操作系统一般只能处理一种网络协议，而且，网络是作为操作系统的附件供用户挑选的，而现在网络功能已经作为操作系统的一个内置功能。自从网络协议栈技术的推出，在同一个操作系统中就可以处理不同的网络协议，为异种机之间的互连打下了坚实的基础。

在用户界面方面，最新推出的基于窗口的图形用户界面为用户的应用软件的开发提供了强大的支持，极大地改善了用户与计算机的交互手段。

操作系统的这种扩张的现象和目前基于复杂指令集的芯片的发展情况非常相似。由于微电子技术的不断发展，在单个芯片中可以容纳的元器件数量目前还在不断增加，这就为复杂指令集芯片提供了更广阔的发展空间。从操作系统的角度来看，由于处理机能力的不断提升，在操作系统内核中不断扩展功能的空间依然很大。因此，采用目前的体系结构，商用型操作系统还有很大的发展空间。

3.2 研究型操作系统的结构特征

微内核技术是操作系统发展的一个里程碑，微内核的核心思想是将传统操作系统内核中的一些组成部分放到内核之外来实现。在微内核中只保留了处理机调度、存储管理、消息通讯等少数几个组成部分。传统操作系统中的文件系统、网络等内核功能都放在内核外作为一个独立的子系统实现。

基于微内核结构的操作系统和传统的操作系统相比具有以下特点：①在内核中第一次全面地引入并完整地实现了内核对象；②面向多处理机，基于微内核的操作系统在内核中引入了多处理机调度和管理机制，并引入了细粒度的并发机制—线程，使得多个处理机可以在同一个任务中并行地执行；③面向分布式系统，在微内核结构的操作系统中，采用基于客户/服务器体系结构，以消息机制为基础。操作系统的各种功能都以服务器方式实现向用户提供服务，用户对服务器的请求，是以消息传递的方式传给服务器的；④内核精巧，

通常内核只由三个部分组成：任务管理、虚存管理和进程间通信。传统操作系统内核中的许多部分都被移出内核，采取服务器方式实现。

微内核技术产生了一种完全不同的操作系统体系结构。微内核操作系统的设计思想和精简指令集芯片的设计思想非常相似。精简指令集芯片采用定长指令格式，根据统计分析结果，将最常用的指令集放在芯片中实现，提高了指令译码和执行的速度，提高了芯片的性能，同时降低了芯片的设计和生产的难度，为芯片功能的进一步扩充打下了良好的基础。

4 操作系统体系结构的发展

4.1 基于微内核结构的操作系统

提出微内核结构以后，一个非常重要的问题是如何在微内核上实现传统的操作系统功能。在传统的系统中，操作系统向用户提供的核心服务是通过系统调用实现的。用户在程序中使用系统调用时，程序通过系统调用接口进入内核，在内核中完成相应的功能，然后退出内核。在微内核体系结构下，微内核只提供了与存储管理，任务线程管理和消息通讯等有限的系统调用，传统的操作系统服务如文件系统服务都必须通过操作系统服务器或文件系统服务器来实现。操作系统服务器与微内核在逻辑上是完全独立的，服务器和微内核的关系是客户和服务器关系，操作系统服务器可以独立地运行在用户地址空间中，便于操作系统功能的扩充。

在研制操作系统服务器时，除了需要考虑二进制兼容方面的问题外，还需要考虑的一个问题是系统性能和代码的重用问题。为了保证二进制兼容，目前主要采用了以下两种体系结构：(1)仿真库结构；(2)连接到中断机制。

在 Mach3.0 系统中，系统调用自陷进入微内核，微内核将系统调用重定向，直接返回到用户进程的一个特定空间(称为仿真库)。在仿真库内，通过向操作系统服务器发送消息，来请求操作系统服务器服务，服务器在完成该系统调用功能时，需要调用微内核服务来完成。当服务完成后，再通过消息将结果传递到用户仿真库。最后，返回用户地址空间。

连接到中断机制采用的是可动态加载的服务器模块结构，服务器地址空间与微内核地址空间相互隔离，服务器模块在运行时动态加载，通过连接到中断机制建立与微内核的联系。用户调用系统调用时，自陷进入微内核，微内核通过连接到中断机制将系统调用参数直接传递到服务器模块(服务器运行在核心方式)，直接在服务器模块中完成系统调用功能，然后直接返回用户进程地址空间。

这两种方法各有其优点。仿真库方案体系结构较为灵活，系统服务器完全运行在用户层，便于调试，但是系统开销较大，并且由于仿真库必须占用一段固定的地址空间，对系统的二进制兼容性有一定的影响。连接到中断机制在系统的二进制兼容性、性能和代码的可重用方面有较大的优势。操作系统服务器也可以在用户层调试，最终放入内核层运行(动态加载)，但是在实现上有一定难度。

4.2 微内核体系结构的固有缺陷

不管采用何种技术在微内核的基础上来实现传统操作系统的全部功能，系统的性能都比采用传统的方法实现要低很多。这是由于面向对象和 client/server 技术都是以消息机制为基础的，它比较适合于松散耦合体系结构的系统。对于属于紧耦合结构、各部分关系密切的操作系统内核来说，如果采用纯对象或 client/server 技术会影响系统性能。因此，目前看到的系统都只采用了一部分面向对象的思想，在实现时并没有完全采用面向对象的技术，对象的管理还基本上是用传统的方法，内核中各个模块也是通过直接调用的方式相互调用。各个系统中采用 client/server 技术的层次也不一样，例如 WindowsNT 的文件系统是放在执行层实现的，文件系统服务器通过内部接口直接和底层驱动器打交道，NT 的用户环境服务器作为用户态的服务器实现，应用程序通过动态库来请求环境服务器的服务。Mach3.0 的文件系统是以用户态服务器的方式实现的，服务器必须通过设备接口以消息的方式来请求设备的服务；应用程序通过仿真库发消息请求操作系统服务。从系统性能来看，前者性能要好一些，也就是说单纯地追求用 client/server 技术，尽可能地将原来的内核中实现的功能移到内核之外，以服务器的方式实现的做法并不十分理想。作者认为产生这种结果的一个原因是松耦合体系结构和功能模块间紧耦合关系的矛盾。采用这些技术需要在系统设计目标和系统性能之间进行权衡。

4.3 用虚拟计算的观点看操作系统体系结构的发展

除了微内核体系结构的固有缺陷以外，在微内核操作系统研究发展中存在着许多误区，导致了微内核技术研究偏离了方向，没有抓住本质的东西。

误区 1：微内核意味着内核越来越小。提出微内核技术的一个出发点是为了解决单一内核操作系统内核日益扩大而引起的诸多矛盾，通过精简内核功能，可以使内核功能更精炼，使系统更可靠。而在微内核技术提出以后，在研究领域形成了一种追求内核极小化的趋势，你提出了微内核，那么我就提出一种比你更小的内核，这种趋势使微内核的研究偏离了方向。

误区 2：微内核使操作系统的实现更简单了，每个人都可以实现操作系统了。微内核技术提出了一种思想，即旨在微内核内部实现操作系统机制，而将操作系统的所有策略放在用户层来实现，用户可以根据自己

的需要来实现自己的操作系统策略，这就使许多人产生了一种幻觉，即有了微内核，操作系统的其他功能就可以放在用户层空间实现，这样，人人都可以根据自己的需要来编写自己的操作系统，或随意增加操作系统的功能了。微内核技术的提出使许多人片面地追求在用户态实现操作系统的功能。

误区 3：多操作系统环境。在微内核的基础上，操作系统的其他功能放在用户态，采用服务器的方式实现。这种思想的进一步扩展就是可以在同一个微内核上实现多个、完全不同类型的操作系统环境。这样，用户想用什么样的操作系统环境就可以用什么样的环境。这种不切合实际的想法也对微内核的发展起到了阻碍作用。

我们从虚拟计算的概念出发，得出了资源共享是计算的本质特征，资源共享是推动计算机科学和实践发展的最重要的一种力量。因此，操作系统研究和实践也必须紧紧地围绕在这个主题上，任何操作系统技术的提出必须有助于资源共享程度的提高和优化。这样，我们就可以得出这样的结论，即微内核技术的发展也必须围绕着资源共享这个中心。微内核技术不是要追求更小的内核，而是要提供更细粒度的运行机制；微内核不是要将不同的操作系统环境叠加在同一个微内核系统上，而是需要提供更好的共享机制，因为，多个操作系统环境在功能上是等价的，并不会为用户提供更好的资源共享，而由此带来的实现上的复杂性并不象有些人想象的那么简单。总之，微内核需要追求的是更高的资源共享程度。

那么，根据虚拟计算的理论，微内核技术对操作系统理论和实践的贡献又是什么呢？首先，微内核技术采用了面向对象的技术实现了内核对象，其他外部对象可以通过内核对象来构造，有利于系统资源的共享的实现；其次，微内核技术提供了细粒度的运行机制，有利于系统性能的提高，有利于系统资源的共享；最后，微内核技术提出并实现了许多性能非常高的资源共享算法，如虚拟存储器写时拷贝算法等等。

目前，商用领域的操作系统仍然采用单一内核体系结构，除了不断向系统内核中增加功能外，系统还从微内核技术的发展中吸取有利于资源共享的内容，例如，采用面向对象的技术来实现内核抽象，为用户提供了更细粒度的运行机制，采用结构化的技术来改善单一内核体系结构。由于微处理器的运算能力的不断增强，使得单一内核结构的操作系统还有很大的发展空间。

相反，微内核操作系统的研究却没有走出当前研究的困境。我们在文章的前面提到过微内核体系结构和精简指令集芯片的设计思想非常相似。现在的问题是，精简指令集芯片中的指令集是经过统计而得出的，而微内核中应包含什么功能却是由设计者根据自己的经验和感觉而提出的。精简指令集芯片采用定长指令格式来提高指令的译码速度，这是复杂指令集芯片中所没有的特性，而微内核系统中却没有一种独特的、单一内核系统中不可能有的快速运行机制。因此，微内核研究的一条出路就是要找到和精简指令集芯片中"定长"机制类似的机制，如果找不到这种机制，就没有找到真正的微内核体系结构。另外，新的操作系统体系结构研究必须跨越传统操作系统中资源共享的概念，提出新的、更有利于资源共享的体系结构。

结束语 操作系统体系结构走过了一个非常曲折的发展道路，用虚拟计算的观点看，微内核技术的最大贡献是采用面向对象技术实现了内核抽象，并提出了细粒度的运行机制。微内核技术的最大失误是没有进一步抓住资源共享这一主要矛盾，不是走向了更细粒度的运行机制，走向更有利于资源共享的道路，而是走向了追求更小的内核规模的道路，从而丧失了良好的发展势头。

在虚拟计算中非常强调在用户层"弱化"计算，在实现层"强化"计算的思想。所谓在实现层"强化"计算的概念就是要在内核中实现功能更为强大的、更有利于资源共享的对象。因此，新一代内核不应该是一个"小"的内核，新一代内核中的内核对象必须超越目前的进程对象、线程对象、处理机等内核对象，才有可能有所突破。

参考文献

1 Black D L, et al. Microkernel Operating System Architecture and Mach. In: Proc. of the USENIX Workshop on Micro-kernels and Other Kernel Architecture. Seattle, Washington, 1992. 11～30

2 Custer H. Inside Windows NT. Microsoft Press

3 Hamilton G, Kougiouris P. The Spring Nucleus: A Microkernel for Objects: [Technical Report]. Sun Microsystems Laboratories, Inc.

4 Rozier M, et al. Overview of the Chorus Distributed Operating system. In: same to[1]. 39～69

5 Cheriton D R, et al. A Caching Model of Operating System Kernel Functionality. In: Proc. of the 1st USENIX Symp. on Operating Systems Design and Implementation, Monterey, California, 1994. 179～193

6 潘清. 虚拟计算. 计算机科学，2000，27(3)

7 潘清，等. 微内核技术研究. 软件学报，1998，9(8)：609～612

8 潘清. 操作系统设计和实现中的面向对象技术的应用问题. 计算机科学，1997，24(3)：73～75