中国科学 : 信息科学　2013 年　第 43 卷　第 3 期 : 303–321
www.scichina.com　info.scichina.com
《中国科学》杂志社
SCIENCE CHINA PRESS

评　述

面向网络的操作系统 —— 现状和挑战

梅宏*, 郭耀*

高可信软件技术教育部重点实验室, 北京大学信息科学技术学院软件研究所, 北京 100871
* 通信作者. E-mail: meih@pku.edu.cn, yaoguo@pku.edu.cn

收稿日期: 2012–06–13; 接受日期: 2012–11–08
国家重点基础研究发展计划 (批准号: 2009CB320703)、国家高技术研究发展计划 (批准号: 2011AA01A202) 和国家自然科学基金 (批准号: 60821003, 61103026) 资助项目

摘要　操作系统是计算机系统中最为关键的一层系统软件. 长期以来, 操作系统发展的主线是面向单机, 追求更好地发挥计算机硬件的计算能力, 同时为上层应用和用户提供更友好易用的接口. 随着网络技术的发展, 如何更好地支持网络构成了操作系统发展的一个重要辅线. 近年来, 由于互联网的迅速普及, 面向网络的操作系统得到了广泛的关注, 并逐渐成为操作系统发展的新主线. 为了更好地管理互联网上的分布海量资源, 同时为互联网时代的新型应用和服务提供支持, 操作系统技术正在产生许多重要的变革. 本文简要回顾了操作系统的发展历史, 分析了在互联网时代操作系统面临的主要挑战. 在总结现有面向网络的操作系统的研究进展的基础上, 讨论了其主要特点和未来发展趋势. 最后, 也介绍了我们在此领域针对网构软件的研发尝试.

关键词　操作系统　互联网　中间件　网络操作系统　网构软件

1　引言

操作系统是计算机系统中最为关键的一层系统软件. 按照中国计算机科学技术百科全书 (第二版) 的定义, “操作系统是管理硬件资源、控制程序运行、改善人机界面和为应用软件提供支持的一种系统软件”[1]. 根据 Wikipedia 提供的定义, “操作系统是包含程序和数据的软件, 它运行在计算机上, 管理计算机硬件, 并为各种应用程序的有效执行提供公共服务”[2]. 简言之, 操作系统的主要功能是: 向下管理资源 (包括存储、外设和计算等资源), 向上为用户和应用程序提供公共服务.

结构上, 操作系统大致可划分为如图 1 所示的 3 个层次, 分别是人机接口、系统调用和资源管理. 人机接口负责提供操作系统对外服务、与人进行交互的功能, 从最简单的命令行操作, 到早期 Unix 系统上采用的传统 shell, 进而到 Windows 等现代操作系统中采用的 GUI (图形用户界面) 窗口系统, 人机接口不断向易用性和用户友好发展. 资源管理指的是对各种底层资源进行管理, 存储、外设和计算单元等都是操作系统管理的对象, 随着计算机系统的发展, 新的软硬件资源不断出现, 操作系统的资源管理功能也越来越庞大和复杂. 系统调用是位于人机接口和资源管理之间的一个层次, 提供从人机接口到资源管理功能的系统调用功能.

引用格式: 梅宏, 郭耀. 面向网络的操作系统 —— 现状和挑战. 中国科学: 信息科学, 2013, 43: 303–321, doi: 10.1360/112012-413

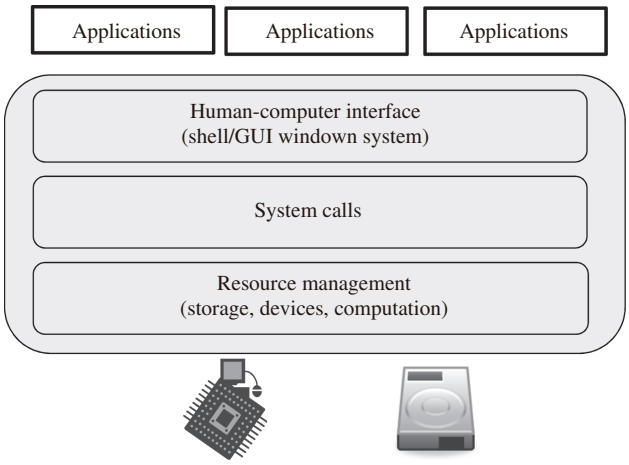

图 1　操作系统结构的 3 个层次

Figure 1　Three layers of the operating system architecture

从不同的视角, 操作系统呈现不同的功用:

• 从计算机系统的视角来看, 操作系统是一个资源管理器. 通过管理和协调对各种底层软硬件资源的使用, 发挥底层软硬件资源所提供的计算能力. 同时通过硬件驱动程序来桥接异构硬件资源, 提高系统的互操作性.

• 从系统使用者的视角来看, 操作系统是一台虚拟机. 一方面提供对底层资源细节的抽象, 另一方面为使用者提供更方便易用的用户界面. 对于软件开发人员来说, 操作系统虚拟机还决定了其面对的编程模型.

• 从应用软件的视角来看, 操作系统是软件的开发和运行平台. 操作系统为应用软件的开发和运行提供各种必要的支撑, 包括: 应用软件的运行环境及其框架设施, 应用软件运行所需资源及其调度和管理, 以及应用软件开发和维护的若干工具.

回顾操作系统的发展历史, 其主线是面向单机的操作系统的发展, 主要目的是为了更好地管理计算机硬件, 同时为上层应用和用户提供更友好易用的服务. 最早的计算机上没有操作系统, 只是一台硬件裸机, 用户可以在某个指定的时间段单独占用计算机上的所有资源, 通过打孔带 (卡) 或者磁带来手动输入程序和数据. 随着处理器速度越来越快, 手动的任务切换方式会浪费大量的处理器时间, 从而出现了批处理功能来对任务进行自动切换, 以提高处理器的利用率. 计算机的操作也逐步从最早的程序员自己动手, 变成了专业的操作人员负责, 再进而出现了自动的监控程序 (monitor, 也译作管程). 自动监控的内容不仅包括 CPU 使用时间的统计, 还包括对打印页数、打孔卡片、读入卡片、磁盘使用的统计, 以及在任务切换时提醒操作员需要完成的动作等等. 这些包含了基本硬件管理、简单任务调度和资源监控功能的管理程序就形成了操作系统的雏形. 随着新的应用需求的不断出现, 越来越多的管理功能逐渐被添加到了操作系统中, 并逐步沉淀为操作系统的标准功能. 在个人计算机出现之后, 为了适应非专业用户的易用性需求, 图形用户界面 (GUI) 也逐渐成为了操作系统中必需的功能. 进入 21 世纪, 随着新的移动智能终端 (例如智能手机和平板电脑等) 的出现, 面向这一类设备的操作系统也在向轻量化、易用性等方向发展. 总体而言, 单机操作系统的发展主要是面向计算机硬件的发展提供更好的资源管理功能, 同时面向新的用户需求提供更好的易用性和交互方式.

从 20 世纪 80 年代开始, 随着网络技术的发展, 计算机不再是孤立的计算单元, 而是要经常通过网络同其他计算机进行交互和协作. 因此, 在上述单机操作系统的发展主线之外, 对网络提供更好的支持成为操作系统发展的一个重要辅线. 在操作系统中逐渐集成了专门提供网络功能的模块, 并出现了最早的网络操作系统 (networking operating system) 的概念 [1]. 另外, 为了更好地提供对网络的支持, 还进一步在操作系统之上凝练出了新的一层系统软件 —— 网络中间件, 作为对操作系统的补充, 专门向上提供屏蔽下层异构性和操作细节的网络相关的共性功能.

进入 21 世纪以来, 随着互联网的快速发展和普及, 几乎所有的计算机系统和其上的操作系统都提供了方便的网络接入和访问能力. 然而, 虽然传统操作系统提供了基本的网络访问功能, 但是主要的管理目标依然是单台计算机上的资源. 如果把互联网当作一台巨大的计算机 (Internet as a computer), 那么如何能够管理好互联网平台上的海量资源, 为用户提供更好的服务, 已经成为互联网时代操作系统亟需解决的问题. 在传统操作系统的核心功能基本定型之后, 面向互联网就成为操作系统发展的新主线. 本文关注的网络化操作系统, 主要指的也就是面向互联网的操作系统 (Internet operating system). 为简便起见, 下面本文均用 “网络化操作系统” 指代面向网络 (特别是面向互联网) 的操作系统.

本文通过对操作系统发展历史的简要回顾, 分析了在互联网时代操作系统面临的主要挑战. 在总结现有面向网络的操作系统的研究进展的基础上, 讨论了其主要特点和未来发展趋势.

2 操作系统历史的简要回顾

历史上第一个实际可用的操作系统是 1956 年出现的 GM-NAA I/O [2], 其主要功能是在一个程序执行完成之后, 自动开始下一个程序的执行 (即批处理). 在具体实现上提供了在程序间共享的一些例程, 从而使它们都可以访问相应的输入/输出设备. 随着操作系统的发展, 逐步添加了越来越多的公共功能, 不仅在功能和结构上越来越接近现代操作系统, 而且还逐渐添加了与网络相关的功能.

下面分别从两条线索来总结操作系统的发展历程: 单机操作系统的发展和操作系统网络支持能力的发展.

2.1 单机操作系统的发展

如前所述, 操作系统发展的主线是面向单机的操作系统. 最早的操作系统为了弥补处理器速度和 I/O 之间的差异, 提供了批处理的功能来提高系统效率. 随着计算机系统的能力越来越强, 又出现了分时系统和虚拟机的概念, 从而可以把一台大型计算机共享给多个用户同时使用. 最早的计算机只用来满足科学与工程计算等专用功能, 因此, 其上的操作系统并没有考虑通用性. 但是随着新的应用需求的不断出现, 特别是个人计算机的普及, 最早软硬件捆绑的系统无法满足灵活多变的应用需求, 因此, 提供通用和易用的用户接口就逐渐成为操作系统发展的必然选择.

在 20 世纪 60 年代 IBM 推出 System/360 系列计算机之前, 几乎每台计算机都是为特定的用户和目的而设计和制造的. IBM S/360 系列中拥有不同规模和能力的计算机都采用了相同的指令集, 并且为这些计算机设计和开发了统一的 OS/360 系列操作系统. OS/360 不仅支持批处理, 而且可以通过

1) 当时的网络操作系统 (networking operating system) 指的是以 Novell Netware 为代表的提供了局域网联网和数据共享等功能的操作系统扩展, 并不是现代意义上的网络化操作系统

2) GM-NAA I/O 输入输出管理系统是通用汽车公司 (General Motors) 和北美航空 (North American Aviation) 联合研制的在 IBM 704 计算机上运行的管理程序, 采用该系统的 IBM 704 计算机总共有大约 40 台

对内存进行划分支持多个程序的同时运行 (即: 多道程序, multi-programming). 另外, IBM 也为 360 系列计算机提供了支持多用户分时使用的功能, 包括 TSS/360 (分时共享系统)、TSO/360 (分时共享选项)、VM/370 (虚拟机) 等机制. 在采用 VM/370 之后, 可以把每台虚拟机都当作是一台真正的 360 机器, 在其上运行不同版本的 OS/360 操作系统, 通过分时使用来进一步提高计算机的利用率. 根据 OS/360 项目负责人 Fred Brooks 的回忆, 虽然该操作系统在开发过程中遇到了许多困难, 例如系统过于庞大, 采用的实现语言可扩展性不强等, 但是 OS/360 依然是历史上最为成功的操作系统之一 [3]. 在首个版本开发完成 40 多年之后, 在 IBM 的许多主机系统 (例如 Z/90) 上, 还能看到 OS/360 的影子 [4].

公认的第一个现代操作系统是从 20 世纪 70 年代开始得到广泛应用的 Unix 系统. 与 OS/360 采用汇编语言不同, Unix 是第一个采用与机器无关的语言 (C 语言) 来编写的操作系统, 从而可以支持更好的可移植性. 在发展过程中, C 语言和 Unix 彼此促进, 都在各自领域成为了最为成功的典范. 采用高级语言来编写操作系统具有革命性意义, 不仅极大地提高了操作系统的可移植性, 还促进了 Unix 和类 Unix 系统的广泛使用. 同时, Unix 还提供了标准化的编程接口 (API) 以及相应的函数库 (lib), 并且集成了 C 语言的开发环境, 从而为用户提供了更易用的编程平台.

从 20 世纪 80 年代开始, 以 IBM PC 为代表的个人计算机 (PC) 开始流行, 开启了个人计算时代. PC 上的典型操作系统包括 Apple 公司的 Mac OS 系列, Microsoft 公司的 DOS/Windows 系列, 以及从 Unix 系统中衍生出来的 Linux 操作系统. PC 时代的操作系统主要面向个人用户的易用性和通用性需求, 一方面提供现代的图形用户界面 (GUI), 可以很好地支持鼠标等新的人机交互设备, 另一方面提供丰富的硬件驱动程序, 从而使用户可以在不同计算机上都使用相同的操作系统. 目前, 90%以上的个人计算机采用的都是 Microsoft 公司的 Windows 系列操作系统 (包括 Windows XP/Vista/7), 由于篇幅有限, 本文不讨论 Windows 操作系统的各个版本之间的发展及其技术进展.

进入 21 世纪之后, 在个人计算机普及的同时, 出现了以智能手机为代表的新一代移动计算设备, 从黑莓 (BlackBerry) 到 iPhone, 再到 Google Android 手机的广泛流行, 智能手机已经成为了新一代的小型计算设备. 在智能手机上运行的这一类操作系统从核心技术上并无实质性变化, 主要是着眼于易用性和低功耗等移动设备的特点, 对传统操作系统 (例如 Linux) 进行了相应的裁剪, 并开发了新的人机交互方式与图形用户界面. 伴随着这些智能手机操作系统, 也出现了以 Apple AppStore 为代表的新型应用软件发布模式, 基于客户端操作系统和后端应用软件商店提供的服务, 用户可以在线查找所需的应用, 并按照需要进行安装和使用, 这在很大程度上改变了传统操作系统上的应用开发和部署模式.

近年来, 绝大多数计算机采用的处理器已经从单核处理器发展为双核、四核甚至更多核的处理器, 然而, 目前的多核处理器上采用的操作系统依然是基于多线程的传统操作系统架构, 很难充分利用多核处理器的处理能力. 为了更好地提高多核处理器的执行效率, 研究人员已经在尝试专门针对多核处理器开发多核操作系统的原型 [5], 但是, 目前均还未得到广泛的推广和应用.

表 1 给出了单机操作系统和相关技术的主要发展过程, 表中最后一列的主要特点是指该操作系统引入的主要新特性, 一般来讲, 新一代的操作系统中会包容前一代操作系统中的主要技术. 总的来看, 单机操作系统发展的主要目的是为了更好地发挥计算机硬件的效率, 以及满足不同的应用环境与用户需求. 在 UNIX 系统出现之后, 单机操作系统的结构和核心功能都基本上定型, 在此之后的发展主要是为了适应不同的应用环境与用户需求而推出的新型用户界面与应用模式, 以及在不同应用领域的操作系统功能的裁剪. 在这条发展主线中, 我们略去了与网络相关的技术. 下面会专门介绍与网络相关的操作系统发展过程.

表 1 单机操作系统和相关技术的主要发展历程

Table 1 The development history and main technologies of single-machine operating systems

Time	Representative OS	Computer systems	Main characteristics of OS
1956	GM-NAA I/O	IBM 704	– The first practical OS – Simple batch processing – I/O management
1960s	IBM OS/360 Series	IBM 360 series mainframe	– Time-sharing – Multi-batch processing – Memory management – Virtual machines (VM/370)
1970s	Unix	Minicomputers / Workstations	– The first modern OS – Developed with machine-independent languages (C) – Provides standard interfaces – Integrated development environment
1980s∼1990s	Mac OS, Windows, Linux	Personal computers	– Provides modern graphical user interfaces (GUI) – Improves usability for personal users
2000s	Apple iOS, Google Android, Windows Phone	Smart phones	– Customization of traditional OSes – Improves usability for mobile devices – New app delivery model (App Store, Android Market)

2.2 操作系统网络支持能力的发展

除了单机操作系统的发展主线之外, 在操作系统的发展历史上还存在另外一条重要辅线, 那就是扩展操作系统的能力为网络提供支持. 操作系统上的网络支持能力大致可以分为两个层次: 一个层次是随着局域网、广域网以及 Internet 的逐步普及, 通过扩展操作系统的功能来支持网络化的环境, 主要提供网络访问和网络化资源管理的能力; 另一个层次是在操作系统和应用程序之间出现了新的一层系统软件 —— 中间件 (middleware), 用以提供通用的网络相关功能, 支撑以网络为平台的网络应用软件的运行和开发. 下面分别按照这两个层次介绍操作系统网络支持能力的发展.

1) 扩展单机操作系统以支持网络化环境.

由于最早的计算机系统都是孤立运行的, 因此, 在最早出现的操作系统中并没有考虑网络支持的功能, 例如 Windows 系列操作系统直到 Windows 95 才开始把对网络的支持 (例如网卡驱动和 TCP/IP 协议) 内置到系统中.

在 20 世纪 70 年代, 首先在局域网中出现了以太网的概念 [6], 可以实现分布式的数据包交换. 随着广域网的发展, 在 1983 年的 ARPANET 中提出了 TCP/IP 协议, 然后到 1985 年, Unix 系统中出现了网络化的文件管理功能, 也就是 NFS (network file system). 有了这些基本的网络支持功能, 随后在

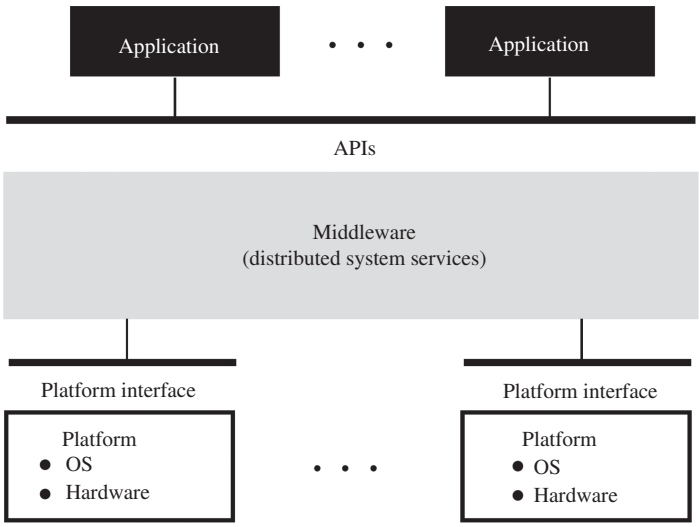

图 2　中间件的概念模型 [10]

Figure 2　The concept model of middleware

20 世纪 90 年代出现了 “网络操作系统 (networking operating system)” 的概念, 例如 Novell Netware, Artisoft LANtastic 等系统. 严格来讲, 这一类网络操作系统只是在原来单机操作系统之上添加了对网络协议的支持, 从而使得原本独立的计算机可以通过网络协议来访问局域网 (或者广域网) 之上的资源, 这样的操作系统并不是现代意义上的网络化操作系统.

在网络操作系统之外, 还出现过分布式操作系统 (distributed operating system)[7] 的概念. 分布式操作系统对用户来讲看起来像是一个普通的集中式操作系统, 但是却运行在多个独立的中央处理单元 (CPU) 之上. 它的关键概念是透明, 也就是说, 多个处理器单元的使用对于用户来讲应当是不可见 (透明) 的. 换句话说, 用户可以把分布式系统看作是一台 “虚拟的单机”, 而不会看到分布式系统中的独立计算机. 分布式操作系统在提供网络支持的前提下, 面向多处理器系统提供完全透明的任务分配、并行执行的功能, 并进一步可以扩展到对网络计算资源的分布式管理. 由于分布式操作系统的概念过于理想化, 虽然关于分布式操作系统的研究非常多, 但是事实上并没有真正商用的分布式操作系统的实现.

2) 基于单机操作系统的系统软件层 —— 中间件.

中间件的概念最早出现在 1968 年 NATO 软件工程会议的报告中 [8]. 在 1972 年出版的英国杂志《Accountant》中给出了对中间件的完整描述: “由于有些系统过于复杂, 需要对标准操作系统进行增强或修改, 由此得到的程序被称作是 ‘中间件’, 因为它们位于操作系统和应用程序之间”[9] . Bernstein 在 1996 年的 Communications of the ACM 上发表的综述文章 [10] 中给出了中间件的分类, 解释了中间件的特性和演进过程, 并提出了一个广泛使用的中间件概念模型, 如图 2 所示.

作为对操作系统的补充, 中间件为应用程序提供了一系列的应用编程接口 (API), 以辅助应用程序以更加透明的方式访问网络资源. 网络中间件的主要特点是凝练网络应用程序共性, 简化其开发和运行, 并且对系统底层分布环境 (网络、主机、操作系统、编程语言) 的复杂性和异构性进行抽象, 使得网络化应用程序只需关注应用逻辑自身. 网络中间件的功能主要包括远程过程调用、负载均衡、事

务处理、容错、安全保障等, 典型例子包括: DCE/RPC [3], CORBA, .Net, J2EE 等.

2.3 操作系统发展的主要驱动力

分析操作系统发展的历史, 可以总结出在操作系统发展过程中的几个主要驱动力:

1) 追求更高效地发挥硬件资源所提供的计算能力. 随着计算机硬件的发展, 操作系统必须能够提供更加高效的利用新硬件的能力, 利用软件技术统筹管理好硬件以形成灵活、高效、可信、统一的虚拟资源. 例如: 在 Unix/Linux 中, 可以把许多外设都当作文件来进行管理, 从而可以提供统一的管理和操作方式, 提高应用程序的开发效率; 为了解决处理器、内存和 I/O 设备之间的速度差异, OS/360 操作系统通过对内存的划分支持多个程序同时驻留在内存中, 从而可以实现多道程序执行的功能; 在现代操作系统中, 还可以通过虚拟内存的概念扩展计算机的物理内存, 使应用程序在运行时几乎可以拥有无穷大的内存空间; 在局域网上, 大多数操作系统都支持通过网络文件系统 (例如 NFS) 扩展计算机的本地磁盘, 使应用程序可以以透明的方式访问其他计算机上的磁盘空间.

2) 通过凝练应用软件中的共性并进行复用, 尽可能提高软件开发和运行的效率. 从前面对操作系统发展的历史分析可以看到, 操作系统的很多功能是通过凝练应用软件中的共性而得到的, 并将应用程序的共性沉淀为操作系统 (中间件) 的一部分, 通过复用这些共性功能, 可以为软件的开发和运行提供更为丰富的应用程序接口和运行时函数库, 提高软件开发和运行的效率. 例如, 在 Unix 操作系统中, 集成了完整的开发环境, 包括 cc (C 语言编译器)、make (build 管理工具)、头文件、系统库 (主要是 libc) 等开发工具; 在图形用户界面的发展过程中, 从 1991 年开始发布的 Linux 操作系统中并没有包括图形化桌面, KDE 和 Gnome 图形桌面环境是分别在 1996 年和 1997 才发布的, 而直到更晚的时候, 它们才逐渐成为 GNU/Linux 操作系统标准发布版中的一部分.

3) 更好地满足用户对易用性的需求. 随着新的应用模式和人机交互方式的出现, 操作系统本身还引领了新型人机交互界面及相关技术的发展. 早期的 Unix shell 就是一种非常经典的用户界面, 直到现在还是许多系统程序员的首选. 随着个人计算机的流行, 基于图形用户界面 (GUI) 的窗口系统成为了使用最为广泛的交互方式. 随着互联网的迅速发展和应用的服务化, 在浏览器中可以执行原来本地应用程序的大部分功能, 因此 Web 浏览器已经不再是提供简单的网络浏览功能, 而正在成为一种新的用户界面模式.

2.4 操作系统现行发展模式存在的问题

回顾操作系统的发展, 可以看到其发展模式还存在一些问题:

1) 从单机操作系统的角度来看, 传统操作系统日趋庞大和复杂, 但内核和核心功能变化并不大, 所增加的主要是 “应用共性” 和易用性的能力. 目前主流的桌面操作系统, 例如 Windows 7 的典型安装需要占用 10 GB 左右的系统空间, 而其中超过一半 (约 5.6 GB) 是动态链接库, 其中包括为支持相同功能的动态链接库的各种版本, 以支持各种不同的遗产应用. 另外, 一般的 Linux 系统 (例如 RedHat 9) 的最大安装和最小安装之间的差别有 10 倍之多, 而在操作系统提供的这些额外功能中包括各种应用程序 (包括全套办公软件、网络访问软件等)、多套图形化桌面系统 (例如 KDE 和 Gnome)、以及庞大的开发环境 (gcc 编译器以及系统库等).

3) DCE/RPC 的全名是 distributed computing environment/remote procedure calls (分布式计算环境/远程过程调用), 它是开放软件基金会 (Open Software Fundation, OSF) 于 1984 年提出的用于分布式计算环境的 RPC 协议规范, 被认为是最早的网络中间件之一

现代操作系统日益庞大, 结构也越来越复杂, 但是对于不同用户而言, 可能只会用到操作系统中提供的全部功能中的一小部分. 这不仅在某种程度上造成了资源浪费, 而且因此给系统带来了不必要的复杂性. 因此, 如果能够对传统操作系统进行瘦身, 使之通过个性化定制和裁剪的方式来支撑不同的应用和用户, 将会使操作系统更加符合未来的发展趋势.

2) 另一方面, 虽然目前的操作系统可以依赖其上的网络中间件来解决与网络相关的部分挑战, 但是由于中间件基于复杂的操作系统层, 不直接操控底层硬件资源, 因此会使计算系统性能受到影响. 虽然采用层次结构有利于控制系统复杂性, 但是通常要付出一定的性能上的代价. 另外, 由于传统中间件系统是在封闭静态环境中发展出来的, 对开放性和动态性的支持能力不足, 因此互联网软件的开放性和动态性对传统中间件系统的可扩展性和自适应性提出了巨大的挑战.

3) 此外, 互联网还为操作系统带来了新的可信性挑战. 由于传统操作系统发展模式的局限性, 新的功能大多是在以前的功能之上进行叠加, 从而形成难以维护的庞大源代码, 使得目前的各种商用操作系统中都或多或少存在一些漏洞. 在互联网时代, 由于可以通过网络攻击远程计算机上的操作系统, 使得操作系统上的漏洞被放大化; 例如, 2010 年出现的 Stuxnet 病毒 [4] 感染了全球超过 500 万台的计算机, 其中伊朗全国受到感染的计算机达到将近 60%. 显然, 在现有操作系统之上通过安全性补丁的方式很难真正实现安全可信的操作系统. 因此, 如何从根源上解决操作系统的可信性问题, 研究和开发基于新一代安全可信体系结构的新型操作系统将是一个非常重要的挑战.

3 网络化操作系统的研究现状

随着近 20 年来互联网的快速发展, 操作系统面对的计算平台正在从单机平台和局域网平台向互联网平台转移. 操作系统不仅需要提供网络支持能力, 更重要的是, 需要解决如何管理互联网平台上的庞大的计算资源和数据资源, 如何更好地利用分布式的计算能力等诸多问题. 在互联网时代, 随着单机操作系统的核心功能基本定型, 网络化逐渐成为主流, 因此, 面向互联网就成为了操作系统发展的新主线.

在互联网流行之后, 就出现了互联网操作系统的提法, 各种组织和个人都曾经提出或者尝试开发过被称作是 Internet OS 的软件和系统, 例如著名操作系统专家, 曾在 Amiga 个人计算机上首次引入多任务概念的 Carl Sassenrath 就曾推出过基于他所发明的 REBOL 语言的 REBOL Internet Operating System: IOS [5]. IOS 主要面向企业级用户, 提供比 Email, Web 和即时通信 (IM) 更为先进的群组交流功能, 包括实时的交互、协作和共享机制. IOS 中还提供了大量的常用应用, 所有应用都可以动态更新, 并且开发周期非常短, 可以在很短时间内部署到一个新的平台.

与 Internet OS 比较接近的还有一个概念是 WebOS [6], 这是一个从 1997 年就开始出现的术语 [11], 主要思想是在浏览器中实现类似操作系统的功能, 从而用户可以随时随地通过任意计算机上的浏览器访问自己的操作系统桌面. 在 2000 年前 UC Berkeley 就实现了 WebOS 的原型系统 [12,13], 后来又出现了 VirtualOS, YouOS, G.ho.st 等流行的 WebOS. 其中, EyeOS [7] 是 WebOS 的一个典型代表, 目前由开源社区维护, 并曾被评为 2009 年 SourceForge 开源社区最有潜力的项目. EyeOS 在浏览器中

4) http://en.wikipedia.org/wiki/Stuxnet

5) http://www.rebol.com/index-ios.html

6) 这里的 WebOS 指的是一类 Web 操作系统, 不要与 Palm 公司 (后被 HP 收购) 推出的智能手机操作系统 webOS 混淆

7) http://eyeos.org

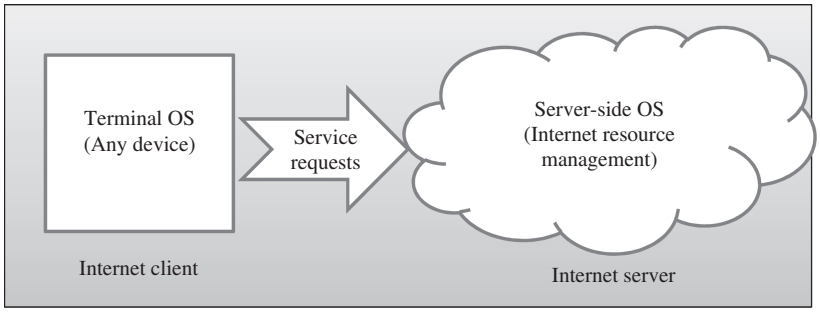

图 3 互联网时代的网络化操作系统

Figure 3 Networking operating systems in the Internet era

提供了一种应用程序的开发环境, 并且通过其开源社区提供包括在线 Office 在内的上千个网页应用, 同时基于开源版本为公司和组织提供定制的办公系统.

严格来讲, WebOS 并不能被称作是操作系统, 它们只是在浏览器中采用 Web 页面展示技术 (例如 JavaScript) 来实现类似于操作系统界面的功能, 例如在网页中实现 Office 文本处理应用, 并且可以在多个应用之间进行切换 (类似于操作系统中的进程切换), 其主要计算和存储功能在服务器端实现, 而应用程序则都在浏览器中执行. 这一类 WebOS 更为确切的名字应该是 WebTop (即 Web 桌面), 也就是说把传统的桌面 (desktop) 转移到 Web 上.

对于 Internet OS 到底应该是什么样子, 以及它所涉及的范围到底有多大, 一直都没有一个公认的定义. Tim O'Reilly [8)]在 2010 年发表了他关于 Internet OS 现状的看法 ("The State of Internet Operating Systems"), 提出, "包括 Amazon Web Services, Google App Engine 和 Microsoft Azure 在内为开发者提供存储和计算访问的云计算平台是正在涌现的 Internet 操作系统的核心" . O'Reilly 认为, 现代的 Internet OS 应当包括如下的功能: 搜索、多媒体访问、通信机制、身份识别和社交关系图、支付机制、广告、位置、时间、图形和语音识别、浏览器. 这从一个侧面表明, 新兴互联网应用拥有的更多共性功能正在逐渐凝练为新的共性基础设施, 这些共性在未来会逐步沉淀为网络化操作系统中的一部分.

图 3 给出了一个互联网时代的网络化操作系统的示意图, 在总体上体现为 "云 + 端" (即互联网服务器 + 客户端) 结构. 网络化操作系统建立在传统终端和服务器操作系统之上, 这些传统操作系统依然拥有如前所示的 3 层结构. 从总体上来看, 网络化操作系统的基本结构如下: 服务器端提供资源管理功能, 通过服务器端操作系统实现海量资源的管理, 并通过虚拟机和服务化技术提供服务; 互联网客户端作为 "人机交互界面" , 可以是包括 PC、上网本 (平板电脑)、智能手机等在内的各种计算设备; 作为二者之间的桥梁, 客户端通过服务请求来访问服务器端提供的计算和存储服务. 在这种模式下, 客户端已经不需要把所有的应用程序 (甚至系统程序) 都安装在本地, 而是可以通过服务请求的方式从互联网 "云" 上按需获取相应的服务.

面向互联网的网络化操作系统面临一系列挑战: 1) 操作系统管理的资源从单机变成了整个互联网上的复杂和海量资源; 2) 互联网时代的新型应用需要新的共性凝练的沉淀, 以及基于庞大共性基础设施的个性化定制; 3) 操作系统需要支持服务化 (SaaS) 及服务的按需使用 (包括即用即丢); 4) 网络化操作系统还要支持不同计算终端 (包括 PC、智能手机、平板电脑等) 的无缝透明访问, 以及云一端之间的协同.

8) 著名 IT 出版商 O'Reilly 出版公司的创办人, Web 2.0 的倡导者之一

近年来, 国内外学术界和工业界都在网络化操作系统方向进行了大量的尝试, 产生了许多有代表性的工作. 下面我们将分别介绍国际上和国内相关的研究进展.

3.1 国际研究进展

互联网时代网络化操作系统发展的一个重要特点是应用驱动, 这是因为新型应用模式的涌现推动了网络化操作系统的技术进步, 而大量的新型应用则主要来源于用户需求和企业的内部需求. 从国际上看, 互联网企业已经成为网络化操作系统发展的主要推动力量之一. 下面我们先介绍工业界的进展情况, 然后再介绍相关的学术界研究进展.

3.1.1 工业界进展

网络化操作系统的需求在很大程度上来自于互联网企业自身的需求. 例如 Google 公司在运营包括搜索引擎在内的大量互联网应用的时候, 就逐渐凝练出了新的文件系统、数据管理功能等操作系统共性, 推动了网络化操作系统的发展. 而 Amazon 公司也是在运营自身支持电子商务的大型数据中心的同时, 提炼出了支持云计算的系统软件, 成为了最成功的云计算基础设施提供商之一. 下面以几个相关的软件公司为例, 介绍工业界对网络化操作系统的发展所做出的主要贡献.

作为全球领先的搜索引擎提供商, 在某种意义上, Google 已经成为了互联网时代的代名词之一. 为了支撑包括搜索引擎在内的大量业务, Google 公司运营了多个大规模、低成本的数据中心. 为了降低成本、提高效率, Google 公司研制了多种新技术来支撑这些数据中心, 其中最著名的包括 Google File System[14], BigTable[15], Map/Reduce[16], Spanner[17] 等, 这些技术均发表在操作系统领域的重要会议之上, 分别从大型数据中心的文件系统、数据管理和分布式计算方面展现了 Google 在网络化操作系统方面的核心技术. 虽然这些技术本身来自于 Google 业务的需求, 但是却在较大程度上引领了研究的潮流, 成为近年来令人瞩目的成果之一.

另一方面, Google 也通过对传统操作系统的剪裁和定制, 迎合互联网和移动互联网时代的新型应用模式, 推出了自己的新型智能终端操作系统 Android 和 Chrome OS. 在智能手机操作方面, 采用开源的 Google 手机操作系统 Android 的智能手机已经超过 Apple 的 iPhone, 占据了智能手机超过一半的市场份额 [9]. Google 还以其 Chrome 浏览器为基础, 研制了 Chrome OS 操作系统, 该操作系统主要面向上网本, 以 Google 提供的云服务为基础, 搭建以浏览器为核心的功能简单、启动速度快 (30 秒以内)、安全的终端操作系统. 2012 年初, Google 对 Chrome OS 的界面进行了改造, 除了类似浏览器的全屏界面之外, 也提供了类似 Windows 和 Mac OS 系统的窗口功能.

作为传统的操作系统和桌面办公软件的主要领导者, Microsoft 的操作系统和办公软件等产品对互联网的支持虽然稍有滞后, 但是在 Google 推出了各种在线服务之后, Microsoft 也在试图把包括 Office 系列办公软件在内的软件转移到互联网上, 以服务的方式提供给用户. 在新推出的 Windows 8 系列操作系统的不同发布版中都添加了对 Microsoft 软件应用商店 MarketPlace 的支持, 同时 Microsoft 也在尝试把其智能手机操作系统 Windows Phone OS 和下一代 Windows 桌面操作系统 Windows 8 通过 Marketplace 紧密结合起来, 以便更好地为互联网和移动互联网提供支持. 另外, Microsoft 研究院作为全球最大的企业研究团队, 在网络化文件管理、浏览器安全、智能终端操作系统的安全和能耗优化等方面均完成了大量的研究工作, 是新型网络化操作系统的重要推动力量.

9) 根据美国尼尔森公司的调查数据, 2012 年第一季度, Android 在美国的市场份额已经达到了 61%

其他新型网络公司, 例如 Amazon 的主业虽然是电子商务, 但是借助其弹性云计算 (EC2) 和云存储 (S3) 系统, 在 IaaS (基础设施即服务) 领域占据领先地位, 掌握了网络化操作系统的大量先进技术. 另外, 借鉴 Google 的成功发展模式, 全球领先的社交网络网站 Facebook 也有计划借助其海量的用户和数据资源, 把社交网络的支撑技术凝练为操作系统的共性功能, 从而推出基于社交网络的 Facebook 操作系统. Firefox 网络浏览器提供商 Mozilla 公司也在 2012 年提出, 基于之前的 Boot to Gecko 项目, 构建面向智能手机和平板电脑的 Firefox 操作系统 [10], 在设备硬件中实现对 HTML5 的支持, 目前 Firefox OS 已经在支持 Android 的移动设备上进行了测试.

除了面向传统计算设备的网络化操作系统之外, 随着物联网等新型应用模式的普及, 还出现了各种面向具体应用领域的新型 "操作系统". 例如, 英国公司 Living Plan IT 正计划在伦敦推出一种 "智慧城市" 操作系统 [11] 该操作系统提供把包括水、电、交通等服务连接在一起的统一城市平台. 与传统操作系统相比, 这种城市操作系统更注重系统的鲁棒性, 因为在该系统上会运行许多生死攸关的服务. 该公司同时计划通过嵌入成千上万的传感器作为监控设备, 为一个新建设的办公街区建立智能照明和供热系统. 美国 Microsoft 公司在 2012 年推出了家庭操作系统 HomeOS[18], 为家庭智能应用的用户和开发者提供了类似于 PC 的技术抽象, 包括网络设备抽象接口、跨设备的任务接口, 以及专为家庭环境设计的管理界面. HomeOS 还提供了面向家庭的应用程序商店, 拥有数十个家庭管理应用, 并在多个真实家庭环境中进行了长时间试用. 从这些例子可以看到, 互联网时代操作系统的概念正在呈现扩展和泛化的态势, 面向各个不同领域的信息化需求, 所构建的向下管理资源、向上支撑各类应用开发、部署和运行的软件平台均被视为新型操作系统, 而网络化正是它们的共性特征之一.

3.1.2 学术界进展

考察近几年 SOSP (international symposium on operating systems principles) 和 OSDI (international symposium on operating systems design and implementation) 等操作系统 (和中间件) 领域的重要学术会议发表的论文, 可以看到, 除了对传统操作系统内核相关技术的继续优化之外, 有几个热点研究方向: 多核操作系统、操作系统的安全机制、以及大量与网络化操作系统相关的研究. 在网络化操作系统方面的研究工作主要集中在如下几个方面:

1) 网络化操作系统资源管理技术: 作为网络化操作系统的服务器端, 数据中心承担了网络化操作系统中资源管理的主要功能, 是网络化操作系统中最为关键的组成部分之一. 近年来和数据中心相关的主要研究热点包括:

- 新型网络化数据的存储机制, 包括新型文件系统与分布式数据存储技术 [14,15,19,20];
- 基于 "键 – 值" (key-value) 的存储方法、性能优化、一致性与故障恢复等 [21~25];
- 提高广域分布式数据存储的性能和可用性 [17,26~28];
- 云存储中的数据安全与隐私保护技术 [29~31];
- 新型分布式计算技术、Map-Reduce 及其优化算法 [16,32~34]
- 数据中心的资源分配与节能技术 [35~37], 等.

2) 网络化操作系统的计算虚拟化技术: 虚拟化技术为网络化操作系统提供了易伸缩的能力, 是网络化操作系统的关键技术, 同时也是云计算的使能技术之一. 对虚拟化技术的研究主要关注虚拟机的效率、安全和节能等方面, 包括:

10) http://www.mozilla.org/firefoxos/
11) http://living-planit.com/

■ 支持多租户的嵌套虚拟化技术 (nested virtualization)[38,39];

■ 虚拟机监控器中的 IO 资源分配技术、优化算法 [40,41];

■ 虚拟机的安全监控技术 [42,43];

■ 智能手机设备上的轻量级虚拟化技术 [44];

■ 通过虚拟机技术降低联网个人计算的能耗 [45], 等.

3) 网络化操作系统客户端技术: 新型网络化操作系统需要支持多种客户端平台, 需要支持新型网络化应用模式、Web 浏览器以及新型移动智能客户端 (包括手机、上网本和平板电脑等). 在客户端技术方面的相关研究主要包括:

■ 基于服务的新型网络化应用模式及其执行环境 [46~49];

■ Web浏览器的功能扩展、浏览器操作系统及其安全保护机制 [50~53];

■ 移动智能手机操作系统的新型应用模式、安全和节能技术 [54~57], 等.

这些学术会议发表的论文不仅来自美国和欧洲的顶尖大学和研究所, 还包括来自 Google, Microsoft, IBM, Facebook 等公司和研究院的大量论文. 从近几年操作系统重要会议的论文发表可以看出, 由于计算机系统和操作系统 (和中间件) 领域的研究越来越趋于应用模式导向, 因此, 目前操作系统领域的研究呈现出了一个非常鲜明的特点: 以新型应用模式作为主要驱动力, 拥有大量数据和基础设施的大型互联网公司在很大程度上引领了研究方向和技术潮流, 而学术界和科研机构则关注于追踪解决具体技术问题和进行优化, 通过与公司合作, 充分利用其积累的实际经验和用户数据, 也在网络化操作系统的研究方向上取得有价值的突破.

3.2 国内研究进展

虽然国内在操作系统等基础软件技术方面的研究还落后于以美国为代表的发达国家, 但是, 长期以来, 在国家重点基础研究发展 (973) 计划、国家高技术研究发展 (863) 计划、“核心电子器件、高端通用芯片及基础软件产品” 国家科技重大专项（简称 “核高基重大专项”）等科研计划的支持下，我国在操作系统和中间件领域已取得了一些可喜的进展, 其中包括以网构软件 [58] 运行支撑环境、虚拟计算环境 iVCE[59]、透明操作系统 TransOS[60], 网络化中间件系统为代表的一系列网络化操作系统和中间件方面的研究成果.

网构软件 (Internetware)[58,61,62] 是我国学者提出的一种新型软件范型. 在对互联网计算环境下软件形态的分析和抽象的基础上, 用来描述具有自主性、协同性、演化性、情境性、涌现性和可信性等性质的互联网应用软件. 针对网络环境下的复杂网构软件, 北京大学等单位从可信的角度研究了网构软件的概念和技术体系. 研究满足质量需求并保障可信度和服务质量的软件构造方法, 以及凝练共性管理功能并保证软件可信、高服务质量运行的软件运行支撑技术. 作为一种新型软件范型, 网构软件及其技术体系为研究面向互联网计算环境的操作系统提供了方法与技术基础.

互联网资源的 “成长性”、“自治性” 和 “多样性” 等自然特性给资源的有效共享和综合利用带来了巨大的挑战. 国防科技大学等单位提出以网络资源的按需聚合和自主协同为核心, 建立虚拟计算环境 (iVCE)[59] 的思路. 以 “聚合与协同” 为构建虚拟计算环境的新途径, 针对互联网资源的自然特性, 研究了开放环境下的按需聚合问题、分布自治资源的自主协同问题以及聚合与协同的计算性质. iVCE 建立在开放的网络基础设施之上, 为终端用户或应用系统提供和谐、可信、透明的一体化服务.

在透明计算模型的基础之上, 清华大学从用户控制的云计算 (customer controlled cloud computing) 角度出发, 以用户端提供用户服务、网络服务器提供程序和数据的存储、对网络软硬件资源进行管理

提出了基于透明计算的云计算操作系统 (TransOS) 的概念 [60]. 把传统操作系统, 例如 Linux、Windows 等也定义为云操作系统中的资源. 把云计算操作系统定义为运行在传统操作系统与计算机主板 BIOS 之间, 对包含各种传统操作系统在内的网络资源进行管理的元级操作系统 (Meta OS) .

在 “十一五” 期间启动的国家 “核高基重大专项” , 不仅重视传统单机操作系统的研发, 同时也部署了面向网络的网络化操作系统、网络中间件和网络化应用支撑工具的研发, 其中, 集成化中间件套件 “四方国件” [12] 就是一项典型的代表性成果, 汇聚了国内主要中间件研究机构和中间件厂商的研发成果, 形成了良好的中间件产学研用生态链.

4 网络化操作系统的发展趋势

虽然许多公司和研究机构都提出了各具特色的网络化操作系统的概念和系统, 包括前面提到的各种 Internet OS 和 WebOS, 以及面向大型数据中心的云计算操作系统 (cloud operating system), 但是, 已有的各种网络化操作系统关注的重点或者是服务器端的运行和管理 (面向数据中心的操作系统), 或者是新型人机交互模式和终端操作系统的裁剪 (移动智能手机操作系统), 或者是以浏览器为代表的云客户端 (浏览器操作系统).

在互联网时代, 网络化操作系统的管理对象变成了互联网平台上的海量存储和计算资源, 而用户可以通过 PC、智能手机、平板电脑等多种互联网终端访问互联网平台上的资源和服务. 为了适应这样的变化, 互联网时代的网络化操作系统需要为海量数据资源和大规模计算提供高效的虚拟化支撑, 使应用和服务可以通过透明的方式访问互联网上的资源; 同时以服务化作为主要技术把云和端进行统一管理, 为服务器端 (云) 和客户端 (浏览器或智能手机) 提供 “云—端” 融合的解决方案.

为了更好地支持新型网络应用模式, 管理海量的计算和数据资源, 并为用户提供充分个性化的服务, 网络化操作系统的发展呈现如下趋势:

• **结构化**: 庞大的单体化的操作系统 (或中间件) 难以满足个性化定制需求, 操作系统和中间件的结构会呈现构件化的形态. 以构件化技术为基础, 能够更好地支持以构件为单元的 “按需使用” 的总体目标, 不仅可以支持应用软件的按需加载, 而且对操作系统等基础软件也可以做到以构件为单元的按需定制、按需加载和使用.

• **服务化**: 互联网时代的应用模式的一个重要特点是服务化. 服务与本地应用的最大区别在于 “只求使用、不求拥有” , 由于服务不需要在本地执行, 因此可以达到 “按需使用, 即用即丢” 的目标. 随着互联网操作系统的进一步发展, 互联网上将会提供无处不在的软件服务, 用户可以随时随地通过任何设备访问这些服务. 另外, 由于网络化操作系统要支持包括桌面计算机、智能手机、上网本 (平板电脑) 在内的多种终端, 因此通过应用的服务化 (和标准化), 可以比较容易地解决应用的多终端适配和 “云端” 协同的问题.

• **易伸缩**: 互联网正在逐步发展成为一个大型虚拟计算平台, 为其上应用或终端用户提供分布式的数据存储和计算能力. 网络化操作系统以统一的方式管理分布式的软硬件资源, 根据应用或用户需求提供易伸缩的资源访问和计算能力.

• **可定制**: 集中计算模式和大量共性沉淀使得互联网操作系统可能成为一个包含巨大资源和应用的管理系统, 但是, 对于面向用户的操作系统来说, 由于每个用户的需求不同, 因此都不可能用到所有

12) http://www.orientware.org/

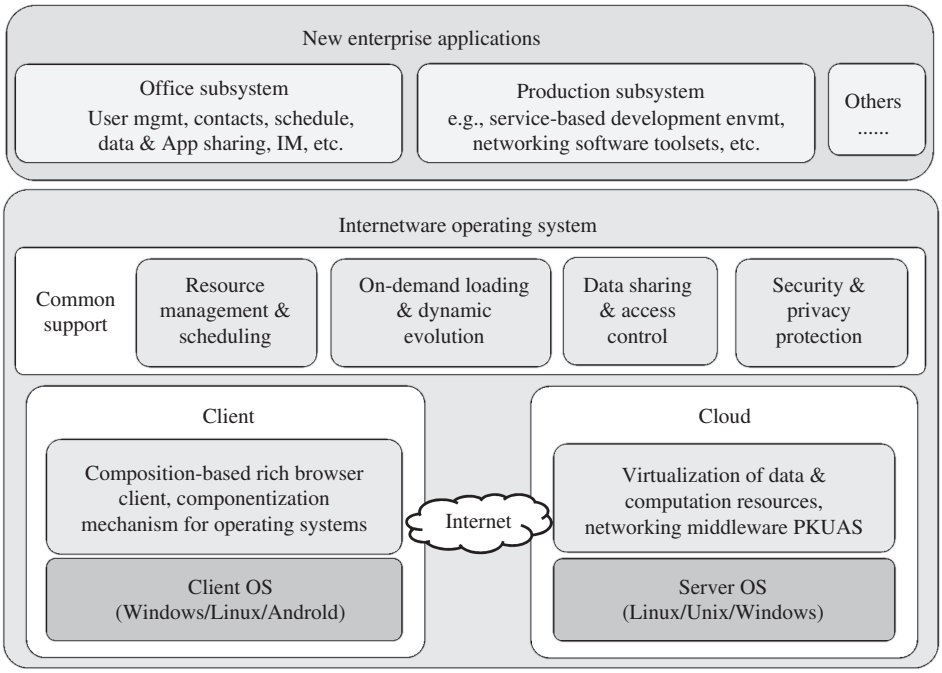

图 4　网构操作系统的概念结构

Figure 4　The architecture of the Internetware operating system

的资源. 另外, 互联网上的资源太多, 也使得用户可能难以找到自己所需的特定资源. 因此, 网络化操作系统需要支持基于强大共性资源的个性化定制功能, 使之不仅可以支持公共的软件和服务, 还可以通过个性化定制方便地构建面向公众、部门、行业、家庭或个人的网络化操作系统.

5　网构操作系统 —— 新型网络化软件运行支撑环境

在前期操作系统和中间件方面多年的研究工作基础上, 北京大学正以网构软件这一新型软件形态作为切入点, 针对新型企业计算 (enterprise computing) 的需求, 构造面向网构软件的网络化操作系统, 简称**网构操作系统** (internetware operating system) , 其概念结构如图 4 所示.

北京大学在操作系统和中间件方面的主要前期研究工作包括: 1) 操作系统的构件化机制, 包括嵌入式和智能手机操作系统 (例如 Android) 的构件化机制 [63]; 2) 新型网络化中间件技术, 包括自主知识产权的应用服务器 PKUAS[64]; 3) 客户端的服务化机制, 包括新型互联网应用的 mash-up 构造技术 [65]、基于组装的浏览器富客户端技术 [66] 等; 4) 在 “云－端” 融合与协同技术方面的初步研究工作等.

以这些前期研究工作为基础, 网构操作系统的研发将主要包括以下几个方面的工作:

• **网构操作系统体系结构**: 针对网络环境下软件的新型性质、特征和应用模式, 研究网构软件操作系统主要特性并构造其体系结构, 包括客户端－服务器深度融合的体系结构风格、服务器端单机操作系统构件化和服务化、客户端应用和服务的按需加载与在线访问等.

• **网构操作系统资源管理技术**: 在网构操作系统服务器端, 主要关注资源管理的相关技术, 包括计算和存储资源虚拟化、集中式管理和可伸缩调度与分配技术、安全隐私保护技术、数据中心能耗模型

与节能技术等;

• **网构操作系统应用支撑技术**: 为支持网构应用软件的开发与运行, 将研究面向网构特征的应用支撑技术, 包括网构软件动态组装和在线演化支持、规模化计算任务的分布并发支持、自治管理等;

• **网构操作系统人机界面技术**: 网构操作系统将支持多种网络终端的访问, 研究内容包括异构客户端上的界面展现模型、用户个性定制策略配置、应用界面的终端屏幕尺寸自动适应、终端电池管理和节能、面向多终端的虚拟工作台等;

• **网构操作系统典型应用示范**: 网构操作系统主要面向新型企业计算的需求, 一般可分为办公系统和生产系统两大部分. 对于不同机构或组织来说, 办公系统拥有较大共性, 但是不同组织的生产系统则会因行业不同而不同, 甚至即使相同行业也可能会有较大差别. 根据我们自身的需求 —— 软件技术研究与开发, 计划在网构操作系统上开发包括办公系统 (用户为中心的数据与应用共享、日程管理、即时通讯等) 和生产系统 (如服务化的编译和开发环境、网络化应用工具集等) 在内的示范应用集.

6 结束语

随着计算向互联网平台变迁, 层出不穷的新型应用模式需要新的运行平台, 因此, 面向互联网的新一代网络化操作系统成为工业界和学术界关心的热点. 本文在总结操作系统发展趋势的基础上, 总结了当前网络化操作系统的主要研究方向, 并对未来可能的发展趋势做了简要分析, 也介绍了北京大学在此领域的相关尝试. 由于互联网计算平台的复杂性, 网络化操作系统的功能和概念都还停留在众说纷纭的阶段, 并未达成共识, 因此, 本文观点只是提供一种视角, 难免以偏概全. 谨供同行参考、批评.

参考文献

1 Zhang X X. Encyclopedia of Computer Science and Technology. 2nd ed. Beijing: Tsinghua University Press. 2005 [张效祥. 计算机科学技术百科全书. 第 2 版. 北京: 清华大学出版社, 2005]

2 Wikipedia. Operating System. Retrieved: November 2011. http://en.wikipedia.org/wiki/Operating_system

3 Brooks F P. The Mythical Man-Month: Essays in Software Engineering. 20th Anniversary ed. Reading: Addison-Wesley Professional, 1995

4 Brooks F P. The Design of Design: Essays from a Computer Scientist. Reading: Addison-Wesley Professional, 2010

5 Boyd-Wickizer S, Chen H, Chen R, et al. Corey: an operating system for many cores. In: Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI'08), Berkeley, 2008. 43–57

6 Metcalfe R M, Boggs D R. Ethernet: distributed packet switching for local computer networks. Commun ACM, 1976, 19: 395–404

7 Tanenbaum A S, van Renesse R. Distributed operating systems. ACM Comput Surv, 1985, 17: 419–470

8 Emmerich W, Aoyama M, Sventek J. The impact of research on the development of middleware technology. ACM Trans Softw Eng Methodol, 2008, 17: 11–19

9 Jenkins B. Developments in computer auditing. Accountant, 1972

10 Bernstein P A. Middleware: a model for distributed system services. Commun ACM, 1996, 39: 86–98

11 Wright A. Ready for a Web OS? Commun ACM, 2009, 52: 16–17

12 Vahdat A, Eastham P, Yoshokawa C, et al. WebOS: Operating System Services for Wide Area Applications. Technique Report CSD-97-938, 1997

13 Vahdat A, Anderson T, Dahlin M, et al. WebOS: Operating system services for wide area applications. In: Proceedings of the 7th IEEE International Symposium on High Performance Distributed Computing (HPDC '98), Washington, 1998. 52–63

14 Ghemawat S, Gobioff H, Leung S. The Google file system. In: Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP'03), Bolton Landing, 2003. 29–43
15 Chang F, Dean J, Ghemawat S, et al. Bigtable: a distributed storage system for structured data. In: Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI'06), Seattle, 2006. 205–218
16 Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. Commun ACM, 2008, 51: 107–113
17 Corbett J C, Dean J, Epstein M, et al. Spanner: Google's globally-distributed database. In: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI'12), Hollywood, 2012. 251–264
18 Dixon C, Mahajan R, Agarwal S, et al. An Operating System for the Home. In: Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI'12), San Jose, 2012
19 Beaver D, Kumar S, Li H C, et al. Finding a needle in Haystack: facebook's photo storage. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 47–60
20 Calder B, Wang J, Ogus A, et al. Windows Azure Storage: a highly available cloud storage service with strong consistency. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 143–157
21 Geambasu R, Levy A A, Kohno T, et al. Comet: an active distributed key-value store. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Berkeley, 2010
22 Lim H, Fan B, Andersen D G, et al. SILT: a memory-efficient, high-performance key-value store. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 1–13
23 Glendenning L, Beschastnikh I, Krishnamurthy A, et al. Scalable consistency in Scatter. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 15–28
24 Ongaro D, Rumble S M, Stutsman R, et al. Fast crash recovery in RAMCloud. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 29–41
25 Shue D, Freedman M J, Shaikh A. Performance Isolation and fairness for multi-tenant cloud storage. In: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI'12), Hollywood, 2012. 349–362
26 Ford D, Labelle F, Popovici F I, et al. Availability in globally distributed storage systems. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 61–74
27 Sovran Y, Power R, Aguilera M K, et al. Transactional storage for geo-replicated systems. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 385–400
28 Lloyd W, Freedman M J, Kaminsky M, et al. Don't settle for eventual: scalable causal consistency for wide-area storage with COPS. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 401–416
29 Mahajan P, Setty S, Lee S, et al. Depot: Cloud Storage with Minimal Trust. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 307–322
30 Feldman A J, Zeller W P, Freedman M J, et al. SPORC: group collaboration using untrusted cloud resources. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 337–350
31 Liu J, George M D, Vikram K, et al. Fabric: a platform for secure distributed computation and storage. In: Proceedings of the 22nd ACM Symposium on Operating Systems Principles (SOSP 2009), Big Sky, 2009. 321–334
32 Yu Y, Isard M, Fetterly D, et al. DryadLINQ: A system for general-purpose distributed data-parallel computing using a high-level language. In: Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI'08), San Diego, 2008. 1–14
33 Zaharia M, Konwinski A, Joseph A D, et al. Improving MapReduce performance in heterogeneous environments. In: Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI'08), San Diego, 2008. 29–42
34 Gonzalez J E, Low Y, Gu H, et al. PowerGraph: Distributed graph-parallel computation on natural graphs. In: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI'12), Hollywood, 2012. 17–30
35 Zhang Y, Wang Y, Wang X. GreenWare: Greening Cloud-Scale Data Centers to Maximize the Use of Renewable Energy. In: Proceedings of the 12th ACM/IFIP/USENIX international conference on Middleware (Middleware'11). 2011. 143–164

36 Verma A, Cherkasova L, Campbell R H. Resource Provisioning Framework for MapReduce jobs with performance goals. In: Middleware, In: Proceedings of the 12th ACM/IFIP/USENIX international conference on Middleware (Middleware'11), Lisboa, 2011. 165–186
37 Verma A, De P, Mann V, et al. BrownMap: Enforcing power budget in shared data centers. In: Proceedings of the 11th ACM/IFIP/USENIX international conference on Middleware (Middleware'10), Bangalore, 2010. 42–63
38 Ben-Yehuda M, Day M D, Dubitzky Z, et al. The turtles project: design and implementation of nested virtualization. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 423–436
39 Zhang F, Chen J, Chen H, et al. CloudVisor: retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 203–216
40 Gulati A, Merchant A, Varman P J. mClock: handling throughput variability for hypervisor IO scheduling. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 437–450
41 Broomhead T, Cremean L, Ridoux J, et al. Virtualize everything but time. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 451–464
42 Colp P, Nanavati M, Zhu J, et al. Breaking up is hard to do: security and functionality in a commodity hypervisor. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 189–202
43 Haeberlen A, Aditya P, Rodrigues R, et al. Accountable virtual machines. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 119–134
44 Andrus J, Dall C, Hof A V, et al. Cells: a virtual mobile smartphone architecture. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 173–187
45 Das T, Padala P, Padmanabhan V N, et al. LiteGreen: saving energy in networked desktops using virtualization. In: Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference (USENIX ATC'10), Boston, 2010
46 Mickens J, Dhawan M. Atlantis: robust, extensible execution environments for web applications. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 217–231
47 Adya A, Cooper G, Myers D, et al. Thialfi: a client notification service for internet-scale applications. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, 2011. 129–142
48 Haridasan M, Mohomed I, Terry D, et al. StarTrack next generation: A scalable infrastructure for track-based applications. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 409–422
49 Douceur J R, Elson J, Howell J, et al. Leveraging legacy code to deploy desktop applications on the web. In: Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI'08), San Diego, 2008. 339–354
50 Greamo C, Ghosh A. Sandboxing and virtualization: modern tools for combating malware. IEEE Sec Privacy, 2011, 9: 79–82
51 Wang H J, Grier C, Moshchuk A, et al. The multi-principal OS construction of the Gazelle web browser. In: Proceedings of the 18th Conference on USENIX Security Symposium, Montreal, 2009. 417–432
52 Tang S, Mai H, King S T. Trust and protection in the Illinois browser operating system, In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 17–32
53 Giffin D B, Levy A, Stefan D, et al. Hails: Protecting data privacy in untrusted web applications. In: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI'12), Hollywood, 2012. 47–60
54 Enck W, Gilbert P, Chun B, et al. TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10), Vancouver, 2010. 393–407
55 Carroll A, Heiser G. An analysis of power consumption in a smartphone. In: Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference, Boston, 2010
56 Tang Y, Ames P, Bhamidipati S, et al. CleanOS: Limiting mobile data exposure with idle eviction. In: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI'12), Hollywood, 2012. 77–91

57 Gordon M S, Jamshidi D A, Mahlke S, et al. COMET: Code offload by migrating execution transparently. In: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI'12), Hollywood, 2012. 93–106

58 Mei H, Huang G, Xie T. Internetware: A software paradigm for internet computing. Computer, 2012, 45: 26–31

59 Lu X, Wang H, Wang J. Virtualized computing environment iVCE: concepts and architecture. Sci China Ser F-Inf Sci, 2006, 36: 1081–1099 [卢锡城, 王怀民, 王戟. 虚拟计算环境 iVCE: 概念与体系结构. 中国科学 F 辑: 信息科学, 2006, 36: 1081–1099]

60 Zhang Y, Zhou Y. A new cloud operating system: design and implementation based on transparent computing. Acta Electron Sin, 2011, 38: 985–990 [张尧学, 周悦芝. 一种云计算操作系统 TransOS: 基于透明计算的设计与实现. 电子学报, 2011, 38: 985–990]

61 Yang F, Lv J, Mei H. Technical framework for Internetware: An architecture centric approach. Sci China Ser E-Inf Sci, 2008, 38: 818–828 [杨芙清, 吕建, 梅宏. 网构软件技术体系: 一种以体系结构为中心的途径. 中国科学 E 辑: 信息科学, 2008, 38: 818–828]

62 Mei H, Huang G, Zhao H, et al. A software architecture centric engineering approach for Internetware. Sci China Press Ser E-Inf Sci, 2006, 49: 702–730

63 Guo Y, Zhang L, Kong J, et al. Jupiter: transparent augmentation of smartphone capabilities through cloud computing. In: Proceedings of the 3rd ACM SOSP Workshop on Networking, Systems, and Applications on Mobile Handhelds (MobiHeld'11), Cascais, 2011

64 Mei H, Huang G. PKUAS: An architecture-based reflective component operating platform. In: Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS 2004), Suzhou, 2004. 163–169

65 Liu X Z, Zhao Q, Huang G, et al. iMashup: assisting end-user programming for the service-oriented web. In: Proceedings of the IEEE/ACM International Conference on Automated Software Engineering (ASE'10), New York, 2010. 285–288

66 Zhao Q, Liu X, Huang G, et al. A browser-based middleware for service-oriented rich client. In: International Conference On Service Sciences (ICSS), Hangzhou, 2010. 22–27

Network-oriented operating systems: status and challenges

MEI Hong* & GUO Yao*

Key Laboratory of High Confidence Software Technologies, Ministry of Education, Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871*, China*
*E-mail: meih@pku.edu.cn, yaoguo@pku.edu.cn

Abstract The operating system represents a key system software layer in a computer system. The main thread of operating system development has been single-machine-oriented with the goal being to improve the performance of computation hardware, and provide friendlier and easier-to-use interfaces for applications and users. With the development of networking technology, networking support in operating systems has become an important auxiliary development thread. Owing to the rapid popularization of the Internet, network-oriented operating systems have attracted widespread attention and have become the main thread in operating system development. To manage the vast distributed data on the Internet and provide better support for new applications and services in the Internet era, operating system technology is undergoing significant evolution. This paper first provides a brief review of the development history of operating systems, including the history and development of operating systems for individual machines, as well as operating systems and middleware support in a networked environment. We also analyze the main challenges of operating systems in the Internet era. After summarizing

the research progress of existing network-oriented operating systems in both academia and industry, we discuss their main characteristics and future development trends. Finally, we introduce our proposal in this area focusing on Internetware.

Keywords operating system, Internet, middleware, Internet operating system, Internetware

MEI Hong was born in 1963. He received his Ph.D. degree in Computer Science from the Shanghai Jiaotong University, Shanghai in 1992. Currently, he is a Professor and the Dean of the School of Electronics Engineering and Computer Science at Peking University. He is also the Director of the Key Laboratory of High Confidence Software Technologies of the Ministry of Education. His research interests include software engineering and system software. Dr. Mei is an academician of the Chinese Academy of Sciences.

GUO Yao was born in 1976. He received his Ph.D. degree in Computer Engineering from the University of Massachusetts, Amherst in 2007. Currently, he is an Associate Professor in the School of Electronics Engineering and Computer Science at Peking University. His research interests include operating systems, low-power design, and mobile computing.