面向网构软件的操作系统:发展及现状

梅宏[1,2,3],郭耀[1,2]

1. 北京大学信息科学技术学院软件研究所,北京 100871
2. 高可信软件技术教育部重点实验室(北京大学),北京 100871
3. 上海交通大学计算机科学与工程系,上海 200240

摘要 计算机软件是计算机系统的“灵魂”,而操作系统是软件运行支撑技术的核心。在互联网时代,软件的运行环境和开发方法发生了重要变化,迫切需要一种新型的面向互联网计算的软件范型,中国学者将这种新范型命名为网构软件。网构软件对包括操作系统在内的软件技术体系带来一系列新挑战。本文关注“面向网构软件的操作系统”(简称“网构操作系统”),简要回顾了操作系统的发展历史,介绍了网构操作系统的概念、基本特征和关键支撑技术。在总结当前网构操作系统的发展现状的基础上,对操作系统的未来发展进行了展望。

关键词 操作系统;网构软件;网构操作系统;软件定义

计算机软件是计算机系统执行某项任务所需的程序、数据及文档的集合,是计算机系统的“灵魂”。作为计算机系统的一部分,软件已经逐渐渗入人类社会、经济、生活的方方面面。C++语言的设计者、著名计算机科学家Bjarne Stroustrup在演讲中多次提到“我们的文明运行在软件之上”。美国著名发明家和计算机科学家Ray Kurzweil在其《奇点临近:当计算机智能超越人类》[1]中也断言:“如果地球上所有智能软件都突然停止工作,那么人类现代文明也会戛然而止。”

计算机软件技术体系主要涉及四个方面(图1):软件范型、软件开发(构造)方法、软件运行支撑及软件质量度量与评估。软件范型是从软件工程师(或程序员)视角看到的软件模型及其构造原理,是软件技术体系的核心。每次软件范型的变迁,都会引发软件开发方法和运行支撑技术的相应变化,并导致新的软件质量度量和评估方法的出现。

随着计算平台从单机向多机、网络,以及开放互联网的演变,软件应用也从最初单纯的计算与数据处理拓展到各个行业的应用,乃至现在泛在的无所不在的应用。在过去60年中,软件范型经历了无结构、结构化、面向对象、面向构件/面向服务化等的变迁,每次软件范型的变迁,都会导致软件技术的一次螺旋上升。软件范型的演变,反映了推动计算机软件技术发展的几个基本动因——追求更具表达能力、更符合人类思维模式、易构造、易演化的软件模型,支持高效率和高质量的软件开发,支持高效能、高可靠和易管理的软件运行等[2]。

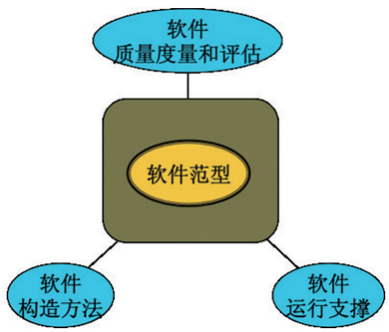

图1 软件技术体系

进入21世纪,在互联网计算环境下的软件形态展现出一系列新的特点。从软件范型的研究角度来看,发生了研究对象从“产生于相对封闭、静态、可控环境下的传统软件”到“运行于开放、动态、难控的网络环境下的复杂软件”的转变;质量目标的重心从“指标相对单一的系统内部和外部质量”到“指标比较综合的以可信度和服务质量为主的使用质量”的转变;构造方法从“满足功能需求并保障功能正确性”到“满足质量需求并保障可信度和服务质量”的转变;运行支撑从“凝练共性应用功能并保证软件正确运行”到“凝练共性管理功能并保证软件可信、高服务质量运行”的转变。

在对互联网计算环境下软件形态的分析和抽象的基础上,中国学者提出网构软件(Internetware)[3~5]的范型,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。通俗来说,网构软件是指在互联网上开发、在互联网上运行、呈现网

收稿日期:2016-05-30;修回日期:2016-07-06

基金项目:国家高技术研究发展计划(863计划)项目(2015AA01A202);国家自然科学基金项目(61421091)

作者简介:梅宏,教授,中国科学院院士,研究方向为系统软件和软件工程,电子信箱:meih@pku.edu.cn;郭耀(通信作者),副教授,研究方向为系统软件和低功耗计算,电子信箱:yaoguo@pku.edu.cn

引用格式:梅宏, 郭耀. 面向网构软件的操作系统:发展及现状[J]. 科技导报, 2016, 34(14): 33-41; doi: 10.3981/j.issn.1000-7857.2016.14.004

络化体系结构、通过互联网对外提供服务的一类软件。针对网络环境下的复杂网构软件,北京大学、南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[6~17]。

另一方面,在软件技术体系中,操作系统是软件运行支撑技术的核心。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[18]。操作系统运行在计算机上,向下管理计算机系统中的资源(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2所示的3个层次,分别是人机接口、系统调用和资源管理[19]。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。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。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。

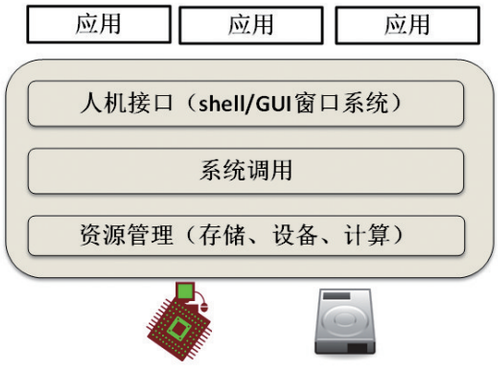

图2 操作系统结构的3个层次

操作系统发展的初期主要是单机操作系统,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。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,对网络提供更好的支持成为操作系统发展的一个重要目标。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)[20~22]的概念。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[23~25],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。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。

表1给出软件范型和操作系统在过去几十年的发展历程的对照。可以看到,软件范型和操作系统在发展过程中彼此促进、共同发展。在早期的单机时代,软件范型和操作系统都处于原始的无结构形态。随着软件范型的结构化,出现了以Unix为代表的结构化操作系统,并且它们一直到现在依然在流行。在面向对象软件范型的时代,出现了以IBM OS/2 2.0、JavaOS[26]、HaiKu(https://www.haiku-os.org/, BeOS[27])等为代表的面向对象操作系统。到了网络时代,随着软件范型向构件化、服务化等方向的演化,为了更好地支持网络功能,操作系统也提供了中间件、SOA等机制作为单机操作系统的补充。

进入网构软件时代,基于网构软件这种新的软件范型,软件技术体系面临新一轮的变化,其中作为软件运行平台的操作系统也面临一系列新挑战。本研究关注面向互联网的操作系统,也就是支持新型网构软件开发和运行的操作系统,即"面向网构软件的操作系统",并将其作为对当前不断涌现的各种面向不同互联网应用的网络化操作系统的一种统称,简称"网构操作系统"。

近年来,学术界和产业界都出现了面向不同领域的操作系统的概念和实现。虽然它们可能会采用不同的名字,例如云计算操作系统、物联网操作系统、机器人操作系统、数据中心操作系统等,但是它们本质上都是面向互联网的操作系统,而这些操作系统所支持的云计算、物联网、大数据等互联网应用都符合网构软件的一系列特征。因此,本文把这一类支持新型互联网应用的新型网络化操作系统都归入网构操作系统的范畴来进行讨论。

表1 软件范型与操作系统的发展历程对照

计算机硬件	软件范型	操作系统
单机时代	无结构	早期(硬件特定)的操作系统
	结构化	传统单机操作系统(Unix,Windows等)
	面向对象	面向对象的操作系统
网络(含互联网)时代	构件化/服务化	单机操作系统+中间件(含SOA)
	网构化	网构操作系统

1 操作系统的发展简史

1956年出现了历史上第一个实际可用的操作系统GM-NAA I/O(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计算机总共约有40台），主要是为了弥补处理器速度和I/O之间的差异，提供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。

第一个公认的现代操作系统是从20世纪70年代开始得到广泛应用的Unix系统[28]。Unix是第一个采用与机器无关语言（C语言）来编写的操作系统，从而可以支持更好的可移植性。采用高级语言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。

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。

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的移动计算设备，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。

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[29,30]，但目前均还未得到广泛的推广和应用。

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。

进入网络时代之后，在单机操作系统的发展主线之外，操作系统发展的另一个方向主要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[25]。

20世纪90年代出现了“网络操作系统（networking operating system）”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）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。

随着20年来互联网的快速发展，操作系统面向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。

在互联网流行之后，出现了互联网操作系统（Internet operating system）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（http://www.rebol.com/index-ios.html）。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。

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，一直都没有形成共识。Tim O'Reilly（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）在2010年发表了关于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构操作系统中的一部分。

近年来,面向不同的互联网计算与应用模式,国内外都提出了许多面向云计算和数据中心的云操作系统。根据Techopedia的定义,“云操作系统(cloud operating system)是设计来管理云计算和虚拟化环境的操作系统。”[31]一般来讲,云操作系统是指构架于服务器、存储、网络等基础硬件资源和单机操作系统、中间件、数据库等基础软件之上的、管理海量的基础硬件、软件资源的云平台综合管理系统。云操作系统管理的对象包括虚拟机的创建、执行和维护,虚拟服务器和虚拟基础设施,以及后台的软硬件资源。

除此之外,随着移动互联网和物联网的发展,出现了面向不同领域的操作系统的概念和实现,包括物联网操作系统、机器人操作系统、企业操作系统、城市操作系统、家庭操作系统等。究其本质,这些操作系统都是面向新型互联网应用而构建的支持这些应用的开发和运行的网络化操作系统,也均可以归入本文所提出的“网构操作系统”一类。

2 面向网构软件的操作系统

针对网络环境下的复杂网构软件,北京大学、南京大学等单位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。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)[19]。

图3给出了网构操作系统的概念结构图,依然遵守之前给出的操作系统的三层结构,但是每层结构的含义发生了较大的改变。网构操作系统的资源管理层管理的资源不再是单个计算机上的存储和计算资源,而是互联网上的云服务器(包括公有云和私有云)上的计算和存储资源以及拥有互联网介入能力的终端设备(PC、平板电脑、智能手机等)和传感器设备等。网构软件通过网构操作系统提供的一系列网构API来使用互联网上的资源。网构操作系统同时还提供了网构软件的开发和运行支撑。

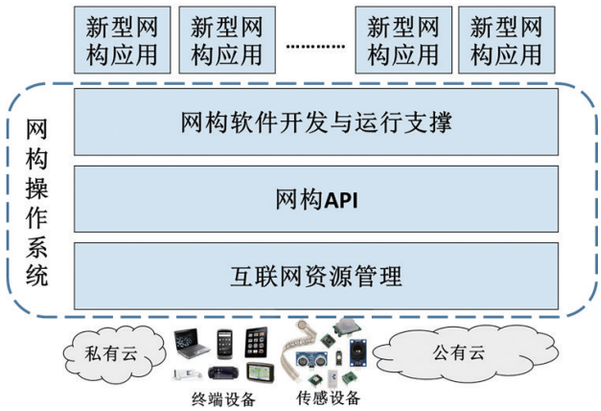

图3 网构操作系统的概念结构

本质上,网构操作系统是一种元操作系统(Meta OS),或者也可以归于系统的系统(systems of systems)。从这个角度来看,网构操作系统中可以包含一个或多个类似于Windows或Linux的传统操作系统,并在已有的操作系统之上构造一层新的系统软件,向下管理已有的计算机系统或者数据,向上为新型网构应用提供开发和运行支持。

2.1 操作系统的软件定义本质

随着“软件定义网络”的流行,近年来出现了各种各样不同的“软件定义”概念。“软件定义”的核心技术途径是硬件资源虚拟化和管理功能可编程[32]。所谓硬件资源虚拟化,是将硬件资源抽象为虚拟资源,然后由系统软件对虚拟资源进行管理和调度。常见的如操作系统中虚拟内存对物理内存的虚拟、伪终端对终端的虚拟、Socket对网络接口的虚拟、逻辑卷对物理存储设备的虚拟等。硬件资源虚拟化带来了如下好处:支持物理资源的共享,提高了资源利用率;屏蔽了不同硬件的复杂细节,简化了对资源的管理和调度;通过系统调用接口对上层应用提供统一的服务,方便进行程序设计;应用软件和物理资源在逻辑上分离,各自可分别进行独立的演化和扩展并保持整个系统的稳定。

管理功能可编程,则是应用对通用计算系统的核心需求。主要表现在访问资源所提供的服务以及改变资源的配置和行为两个方面。在硬件资源虚拟化的基础上,用户可编写应用程序,通过系统调用接口,访问资源所提供的服务,更重要的是能够灵活管理和调度资源,改变资源的行为,以满足应用对资源的多样需求。所有的硬件资源在功能上都应该是可以编程的,如此软件系统才可以对其实施管控,一方面发挥硬件资源的最佳性能,另一方面满足不同应用程序对硬件的不同需求。从程序设计的角度,管理功能可编程意味着计算系统的行为可以通过软件进行定义,成为所谓的“软件定义的系统”。

作为计算系统中最为重要的系统软件,操作系统一方面直接管理各种计算资源,另一方面作为“虚拟机”为应用程序提供运行环境,从操作系统的出现、发展和功能基本定型的过程中,可以看到操作系统实际上就是对计算系统进行“软件定义”的产物。相对于最早的硬件计算机,操作系统可视为一种“软件定义”的“虚拟计算机”,屏蔽了底层硬件细节,由软件对硬件资源进行管理,用户不再直接对硬件进行编程,而是通过应用编程接口(API)改变硬件行为,实现更优的灵活性、通用性和高效性。在此意义上,操作系统体现了“软件定义的系统”技术的集大成。

当前出现的所谓软件定义的网络、软件定义的存储等技术,如同设备互联技术、磁盘存储技术之于单机操作系统一样,本质上正反映了“网构操作系统”对网络化、分布式设备的管理技术诉求,也将成为“网构操作系统”核心的底层支撑技术,并在操作系统的整体协调

下,发挥最佳的功效。以云计算管理系统为例,作为一种互联网环境下的新型网构操作系统,通过“软件定义”技术对网络化、规模化的各种计算资源进行高效灵活的调度管理。云计算管理系统通过“软件定义”的途径,一方面实现资源虚拟化,达到物理资源的共享和虚拟资源的隔离;另一方面实现管理功能可编程,打破传统硬件有限配置能力的桎梏,为用户的业务需求提供高效灵活、随需应变的支撑。因此,云计算管理系统作为一种新兴的网构操作系统,是贯穿了硬件资源虚拟化、管理功能可编程特性的一个典型软件定义的系统。

2.2 网构操作系统的核心技术支撑

作为正在涌现和步入主流的新型操作系统,网构操作系统的核心技术支撑包括如下方面。

1)虚拟化。

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。网构操作系统以虚拟化的方式统一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。

虚拟化(virtualization)技术[33]最早出现在20世纪60年代的IBM大型机系统,在70年代的System 370系列中逐渐流行起来,这些机器通过一种叫虚拟机监控器(virtual machine monitor, VMM)的程序在物理硬件之上生成许多可以运行独立操作系统软件的虚拟机(virtual machine)实例。随着近年多核系统、集群、网格甚至云计算的广泛部署,虚拟化技术在商业应用上的优势日益体现,不仅降低了IT成本,而且还增强了系统安全性和可靠性,虚拟化的概念也逐渐深入到人们日常的工作与生活中。

在计算机科学领域中,目前广泛使用虚拟化来描述对各种计算资源的抽象,而不仅仅局限于虚拟机的概念。对于网构操作系统来说,虚拟化意味着把操作系统需要管理的底层软硬件资源都通过虚拟化的方式进行抽象,通过统一的编程接口(API)提供给应用软件和上层用户使用。这里的虚拟化是一种广义的虚拟化,它不仅仅指计算资源的虚拟化,同时也涵盖了存储资源、数据资源、传感器、网络设备、终端设备、甚至于用户信息等的虚拟化。

近年来,针对云计算和移动计算等新型计算模型,在虚拟化方面取得了很多研究进展,例如操作系统内核虚拟化技术[34]、数据中心的虚拟化[35]、支持多租户的嵌套虚拟化技术(nested virtualization)[36,37]、虚拟机监控器中的IO资源分配技术和优化算法[38,39]、智能手机设备上的轻量级虚拟化技术[40]等。为了对网构操作系统的运行提供更好的支撑,还需要研究针对不同设备和资源的更为广泛的虚拟化技术及相应的优化机制。

2)应用编程接口(API)。

应用编程接口(application programming interface, API)是软件系统不同组成部分衔接的约定,一般指操作系统或函数库向应用程序提供的编程接口。由于近年来软件的规模日益庞大,常常需要把复杂的系统划分成小的组成部分,因此编程接口的设计十分重要。在程序设计的实践中,编程接口的设计首先要使软件系统的职责得到合理划分。良好的接口设计可以降低系统各部分的相互依赖,提高组成单元的内聚性,降低组成单元间的耦合程度,从而提高系统的维护性和扩展性。

传统操作系统其实就是API的提供者。操作系统向应用软件提供各种不同的库(library),而应用程序需要通过API调用这些库的内容,使用计算机上的不同硬件与资源。到了互联网时代,网构软件更多地依赖API(或服务)来调用位于本地或者远程的存储和计算资源,以完成相应的功能。

随着API越来越重要,近年来出现了API经济(API economy)的概念。API不仅仅是计算机程序之间调用的接口,而是成为现代企业提供服务、应用和系统的数字纽带。API经济提倡的是一种数据和计算的开放性接口以及基于这些接口运行的数量庞大的新型应用。从这种角度来看,API经济推动的其实是提供API、支持新型应用开发和运行的一种网构操作系统,即API经济本身就是一种操作系统的体现形式。

3)系统安全和隐私保护。

在互联网时代,网构软件的开放性和涌现性带来了严重的安全和隐私威胁。越来越多的用户数据和计算都放到公有云平台上,会带来数据安全的问题和隐私泄漏的威胁,这主要包括以下核心技术。

(1)网构操作系统基础设施的安全机制:如何确保数据的保密性和完整性、如何在主机系统和应用层面保证基础设施的安全等。

(2)网构操作系统存储中的数据安全与隐私保护技术:如何降低数据的安全风险、如何规避数据隐私风险以及与隐私相关的法律法规等。

(3)网构操作系统的身份认证与访问控制:网构操作系统的信任边界、用户管理、身份认证机制、访问控制模型等。

(4)网构操作系统中应用执行的安全管理:包括云计算的安全性和可用性管理、访问控制、安全漏洞与保护机制等。

4)可定制。

集中计算模式和大量共性沉淀使得网构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,因此不可能都用到所有的资源。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,网构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。

构件化技术是以软件自动组装为最终目的的一种软件开发机制。早在20多年前,就有许多学者研究和开发了各种不同的构件化操作系统的原型[41~43],北京大学在国家高技术研究发展计划(863计划)支持下,也针对构件

化的嵌入式操作系统内核进行了长期的研究工作[44]。目前,许多单机操作系统其实都或多或少引入了构件化的概念,以增加操作系统的可定制性。

在互联网时代,网构软件的多样性必然会对网构操作系统的可定制能力提出更高的挑战,因此需要进一步研究包括构件化技术在内的各种可定制技术,以提高网构操作系统对于不同应用模式、不同使用环境的适应性。

3 网构操作系统发展现状

3.1 学术界研究现状

在学术界,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并进行优化,也在网构操作系统的相关研究方向上取得有价值的突破。

近年来,在网构操作系统方面的代表性研究工作包括:

1)面向数据中心的操作系统Arrakis[45,46]利用硬件的SR-IOV支持,使应用程序能安全、高效地直接访问硬件,减少了操作系统对I/O的干预,从而提升I/O性能。在系统实现上,Arrakis结合了multikernel和共享库操作系统的理念。在Arrakis上运行Redis,与Linux相比可以减少读操作65%的延时,写操作81%的延时。在吞吐率方面,读操作提升75%,写操作提升达8倍。

2)IX[47]是另一个面向数据中心的操作系统。IX利用硬件的VT-x支持,使应用程序能安全、高效地直接访问硬件,减少了操作系统对I/O的干预,从而提升I/O性能。在系统实现上,IX采用了共享库操作系统的理念,采取了一次执行完毕和选择性批处理两项措施进一步优化I/O性能。与Linux相比,IX可以提升Memcached吞吐率3.6倍,延时降低至三分之一。

3)Borg[48]是一个计算机集群管理系统,支持在上万台机器组成的多个集群中调度数千个应用程序和数十万个任务。Borg整合了准入控制、任务打包、过载使用(over-commit)和具有进程级性能隔离的机器共享,实现集群计算资源的高效利用。Borg还提供声明式任务描述语言以简化系统的使用。

4)Mesos[49]是一个分布式操作系统内核,支持像用一台电脑(一个资源池)一样使用整个数据中心。Mesos是以与Linux内核同样的原则而创建的,不同点仅在于抽象的层面。Mesos内核运行在每一个机器上,同时通过API为各种应用提供跨数据中心和云的资源管理调度能力。这些应用包括Hadoop、Spark、Kafka、Elastic Search等。Mesos还可配合Marathon框架管理大规模的Docker等容器化应用。

3.2 产业界现状

在产业界,主流的网构操作系统代表是云计算和数据中心操作系统,或者简称为"云操作系统"。美国VMware公司是虚拟化技术的全球领先者,他们提供的云管理系统vCloud Air[50]是一个基于VMware vSphere构建的混合云平台。该平台支持大量操作系统和应用程序不经修改直接在云上运行,提供与内部数据中心一样的可靠性和性能,支持计算资源的按需弹性伸缩。OpenStack是一种可扩展的开放式云操作系统,为大型和小型企业提供了封闭式云环境以外的另一种选择,可降低与专有平台相关的锁定风险。借助由超过6000人和包括戴尔、惠普、IBM、Red Hat在内的190多家公司组成的参与度极高的社区,OpenStack能够带来灵活性和丰富的选择。除此之外,业内领先的Amazon、Microsoft、IBM、Oracle等公司,也都有类似的云计算平台或者系统。

除了面向传统计算设备的网构操作系统之外,随着物联网、"互联网+"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"网构操作系统",其中比较有代表性的包括:

1)美国Microsoft公司在2012年推出了家庭操作系统HomeOS[51],为家庭智能应用的用户和开发者提供类似于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。

2)斯坦福大学最早研发的Robot Operating System(ROS)[52]是一个面向机器人编程的操作系统。ROS包含了一组工具、库以及约定,能够简化不同平台下机器人的开发过程。由于构建具有高鲁棒性的通用机器人软件是一件十分复杂的事情,对于人而言很平常的事情对于不同机器人来说却是十分不同的。没有单独的研究机构或个人愿意完全靠自己解决这个问题。ROS从设计的一开始就注重软件的协作开发,致力于消除不同机器人平台间的差异。

3)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(http://living-planit.com)。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个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。

4)随着传感网络的广泛部署,分布式计算平台推动了对于适用于传感网络的编程范型(programming paradigms)的研究。美国加州伯克利大学提出了面向传感网络的操作系统SwarmOS(https://www.terraswarm.org/swarmos),为上千个传感器提供编程抽象以及系统层面的支持。这些传感器具有有限的内存,无线网络的支持以及和环境的紧密接触。Swarm是一个基于对象的适用于传感网络的分布式系统。它的主要创新在于无缝地将现实时空中的物体集成到应用程序的计算环境中。

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,而网构化正是它们的共性特征之一。

3.3 国内发展现状

虽然中国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计划(973计划)项目、国家高技术研究发展计划(863计划)项目、核高基项目等科研计划的支持下,中国在操作系统和中间件领域已取得了长足进展。

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)[53]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[54]。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)。

北京大学研制的燕云管理平台是一个典型的网构操作系统(http://www.internetware.cn/),实现了对服务器、存储、网络、软件平台等基础软硬件资源的集成与配置管理,支持公有、私有与混合IaaS云的按需构造与管理。燕云提供了Power和x86混合IT架构的支持,并支持基于Power平台的PowerVM以及基于x86平台的KVM、VMware、Xen、Hyper-v。燕云不仅管理了北京大学、清华大学、南京大学、中国科学院等单位的数十台服务器并提供IaaS(infrastructure as a service)和PaaS(platform as a service)服务;而且通过OEM的方式陆续转化为联想和方正等多个IT企业的云管理产品,广泛应用于政务、交通、电信、医疗等多个行业领域的云计算平台,拥有逾百家大中型客户。另外,北京大学研究人员还提出了面向校园的网构操作系统CampusOS[55]。CampusOS主要用于管理大学校园里的联网资源的操作系统,可管理教师、学生、课程、组织机构数据以及用户设备上生成的数据。CampusOS为校园应用的开发提供SDK支持。同时,CampusOS的功能和SDK中的API也可以由开发者自由地添加。

国内云计算和数据中心提供商也提出和实现了多个面向云计算的操作系统。例如,华为FusionSphere[56]是华为公司面向多行业客户推出的云操作系统产品,基于OpenStack架构开发,整个系统专门为云设计和优化,提供强大的虚拟化功能和资源池管理、丰富的云基础服务组件和工具、开放的API接口等,可以帮助客户水平整合数据中心物理和虚拟资源,垂直优化业务平台。浪潮云海OS是浪潮云计算的软件包,包括Incloud OpenStack发行版、面向软件定义计算的InCloud Sphere、面向软件定义存储的InCloud Storage、面向软件定义网络的InCloud Network、数据中心运维管理平台InCloud Manager和面向软件定义安全的InCloud Security六大模块,用户可以灵活选择任意模块或整体解决方案部署到业务环境当中。

4 结论

在互联网时代,操作系统的概念进一步泛化,出现了各种各样不同的操作系统概念。这些操作系统的概念大多自成一体,各有各的解释,并没有人给出统一的界定。其实从操作系统的发展历史来看,早期的操作系统也有类似的情形:面向不同硬件开发了各种版本的形态各异的操作系统。直到Unix系统的出现,传统操作系统的形态才逐渐固化,呈现出现在所看到的Windows、Mac OS、Android等操作系统的形态。可以预见,随着技术的不断演化,未来的网络化操作系统也将会呈现出统一的态势,最终统一到一种通用的网络化操作系统的概念框架之下。

本研究从网构软件的角度,揭示操作系统的软件定义本质,给出了面向网构软件的操作系统的基本概念、框架以及现状和发展,在网构操作系统框架下对当前百花齐放的网络化操作系统现状进行了总结。期待对网络化操作系统的形态共识达成之后,可以总结和凝练互联网应用的共性,开发更高效的网络化操作系统的API、库和构件,使操作系统在互联网和"互联网+"的时代发挥更重要的作用。

参考文献(References)

[1] Kurzweil R. The singularity is near: When humans transcend biology[J]. Cryonics, 2006, 85(1): 160-160.

[2] 杨芙清, 梅宏, 吕建,等. 浅论软件技术发展[J]. 电子学报, 2002, 30(12A): 1901-1906.
Yang Fuqing, Mei Hong, Lv Jian, et al. Some discussion on the development of software technology[J]. Chinese Journal of Electronics, 2002, 30(12A): 1901-1906.

[3] Mei H, Huang G, Xie T. Internetware: A software paradigm for internet computing[J]. Computer. 2012, 45(6): 26-31.

[4] 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[J]. 中国科学(信息科学), 2008, 38(6): 818-828.
Yang F, Lv J, Mei H. Technical framework for Internetware: An architecture centric approach[J]. Science in China (Information Sciences), 2008, 38(6): 818-828.

[5] Hong M, Huang G, Zhao H, et al. A software architecture centric engineering approach for Internetware[J]. Science in China (Information Sciences), 2006, 49(6):702-730.

[6] Mei H, Huang G, Lan L, et al. A software architecture centric self-adaptation approach for Internetware[J]. Science in China (Information Sciences), 2008, 51(6): 722-742.

[7] Mei H, Liu X Z. Internetware: An emerging software paradigm for internet computing[J]. Journal of Computer Science & Technology, 2011, 26(4): 588-599.

[8] Tsai W T, Jin Z, Bai X. Internetware computing: Issues and perspective[C]//Proceedings of the First Asia-Pacific Symposium on Internetware´09. New York, NY: ACM, 2009: 415-438.

[9] Huang G, Song H, Hong M. SM@RT: Applying architecture-based runtime management into internetware systems[J/OL]. International Journal of Software and Information, 2009 [2016-04-28]. https://hal.archives-ouvertes.fr/inria-00459621/document.

[10] Yuan W, Jian L, Feng X U, et al. A trust measurement and evolution model for internetware[J]. Journal of Software, 2006, 17(4). Doi: 10.1360/jos170682.

[11] Lv J, Ma X X, Tao X P, et al. On environment-driven software model for Internetware[J]. Science in China (Information Sciences), 2008, 51(6): 683-721.

[12] Wang P, Sun C, Li L. Primary research on internetware reliability technology[C]//Interdisciplinary and Multidisciplinary Research in Computer Science, IEEE Cs Proceeding of the First International Multi-Symposium of Computer and Computational Sciences. 2006: 424-428.

[13] Tao H, Ding X, Wei J. An application-semantics-based relaxed transaction model for internetware[J]. Science in China (Information Sciences), 2006, 49 (6): 774-791.

[14] Chen X, Liu X, Fang F, et al. Management as a service: An empirical case study in the internetware cloud[C]//e-Business Engineering (ICEBE), 2010 IEEE 7th International Conference on. Shanghai: IEEE, 2010: 470-473.

[15] Chang X U, Liu Y P, Cheung S C, et al. Towards context consistency by concurrent checking for Internetware applications[J]. Sciece China Information Sciences, 2013, 56(8): 1-20.

[16] Zhao L N, Yin Z, Ye X Z, et al. Self-adaptability of internetware based on P2P network[J]. Journal of Zhejiang University, 2008, 42(8): 1316-1322.

[17] Liu Y, Xu C, Cheung S C. Diagnosing energy efficiency and performance for mobile internetware applications[J]. IEEE Software, 2015, 32(1): 67-75.

[18] 张效祥. 计算机科学技术百科全书[M]. 2版. 北京: 清华大学出版社, 2005.
Zhang Xiaoxiang. Encyclopedia of computer science and technology[M]. 2nd ed. Beijing: Tsinghua University Press, 2005.

[19] 梅宏, 郭耀. 面向网络的操作系统——现状和挑战[J]. 中国科学(信息科学), 2013, 43(3): 303-321.
Mei Hong, Guo Yao. Network-oriented operating systems: Status and challenges[J]. Science in China (Information Sciences), 2013, 43(3): 303-321.

[20] Ousterhout J K, Cherenson A R, Douglis F, et al. The sprite network operating system[J]. Computer, 1988, 21(2): 23-36.

[21] Rashid R F, Robertson G G. Accent: A communication oriented network operating system kernel[C]//Proceedings of the Eighth ACM Symposium on Operating Systems Principles. New York: ACM, 2010: 64-75.

[22] Gude N, Koponen T, Pettit J, et al. NOX: Towards an operating system for networks[J]. ACM Sigcomm Computer Communication Review, 2008, 38(3): 105-110.

[23] Emmerich W, Aoyama M, Sventek J. The impact of research on the development of middleware technology[J]. ACM Transactions on Software Engineering & Methodology, 2007, 17(4). Doi: 10.1145/13487689.13487692.

[24] Jenkins B. Developments in computer auditing[J]. Accountant. 1972.

[25] Bernstein P A. Middleware: A model for distributed system services[J]. Communications of the ACM, 1996, 39(2): 86-98.

[26] Mitchell J G. JavaOS: back to the future[J]. Stroke, 1994, 25(9): 1.

[27] Hacker S, Bortman H, Herborth C. The BeOS Bible[M]. 1st ed. London: Addison Wesley Longman Publishing House, 1999.

[28] Ritchie D M, Thompson K. The Unix time-sharing system†[J]. Bell System Technical Journal, 1978, 57(6): 1905-1929.

[29] Boyd-Wickizer S, Chen H, Chen R, et al. Corey: An operating system for many cores[C]. Usenix Symposium on Operating Systems Design and Implementation, OSDI 2008, San Diego, California, December 8-10, 2008.

[30] Wentzlaff D, Agarwal A. Factored operating systems (fos): The case for a scalable operating system for multicores[J]. ACM SIGOPS Operating Systems Review, 2009, 43(2): 76-85.

[31] Cloud operating system (Cloud OS)[EB/OL]. [2016-04-28]. https://www.techopedia.com/definition/26867/cloud-operating-system-cloud-os.

[32] 梅宏, 黄罡, 曹东刚, 等. 从软件研究者的视角认识“软件定义”[J]. 中国计算机学会通讯, 2015, 11(1): 68-71.
Mei Hong, Huang Gang, Cao Donggang, et al. Perspectives on “Software-defined” from software researchers[J]. Communications of CCCF, 2015, 11(1): 68-71.

[33] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization[J]. ACM Sigops Operating Systems Review, 2003, 37(5): 164-177.

[34] Nikolaev R, Back G. VirtuOS: An operating system with kernel virtualization[C]//Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles. New York: ACM, 2013: 116-132..

[35] Angel S, Ballani H, Karagiannis T, et al. End-to-end performance isolation through virtual datacenters[C]//Proceedings of the 11th USENIX conference on Operating Systems Design and Implementation. Berkeley, CA: USENIX Association, 2014: 233-248.

[36] Ben-Yehuda M, Day M D, Dubitzky Z, et al. The turtles project: Design and implementation of nested virtualization[C]//Proceedings of the 9th USENIX conference on Operating systems design and implementation. Berkeley, CA: USENIX Association,2010: 423-436..

[37] Zhang F, Chen J, Chen H, et al. CloudVisor: Retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization[C]//Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. New York: ACM, 2011: 203-216.

[38] Gulati A, Merchant A, Varman P J. mClock: Handling throughput variability for hypervisor IO scheduling[C]//Proceedings of the 9th USENIX conference

on Operating systems design and implementation. Berkeley, CA: USENIX Association, 2010: 437-450.
[39] Broomhead T, Cremean L, Ridoux J, et al. Virtualize everything but time[C]//Proceedings of the 9th USENIX conference on Operating systems design and implementation. Berkeley, CA: USENIX Association, 2010: 451-464.
[40] Andrus J, Dall C, Hof A V, et al. Cells: A virtual mobile smartphone architecture[C]//Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. New York: ACM, 2011: 173-187.
[41] Fassino J P, Stefani J B, Lawall J L, et al. Think: A software framework for component-based operating system kernels[C]//Proceedings of the General Track of the annual conference on USENIX Annual Technical Conference. Berkeley, CA: USENIX Association, 2002:73-86.
[42] Gabber E, Small C, Bruno J, et al. The pebble component-based operating system[C]//ATEC '99 Proceedings of the Annual Conference on USENIX Annual Technical Conference. Berkeley, CA: USENIX Association, 1999: 267-282.
[43] Lu X, Kon F, Singhai A, R, et al. 2k: A reflective, component-based operating system for rapidly changing environments[C]//ECOOP '98 Workshop ion on Object-Oriented Technology. London: Springer-Verlag, 1998: 64-64.
[44] 陈向群, 徐冬, 滕启明. JBEOS:一种构件化的嵌入式操作系统[J]. 南京大学学报(自然科学版), 2005, 41(B12): 2476-2480.
Chen Xiangqun, Xu Dong, Teng Qiming. JBEOS:A Component-based Embedded Operating System[J]. Journal of Nanjing University (Natural Sciences), 2005, 41(B12): 2476-2480.
[45] Simon P, Li J L, Zhang I, et al. Arrakis: the operating system is the control plane[C/OL]//Proceedings of the 11th USENIX conference on Operating Systems Design and Implementation. Berkeley, CA: USENIX Association, 2014[2016-04-28]. https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-peter_simon.pdf.
[46] Peter S, Li J, Zhang I, et al. Arrakis: The operating system is the control plane[J]. ACM Transactions on Computer Systems, 2016, 33(4): 26-34.
[47] Belay A, Prekas G, Klimovic A, et al. IX: A protected dataplane operating system for high throughput and low latency[C]//Usenix Conference on Operating Systems Design and Implementation. Berkeley, CA: USENIX Association, 2014: 49-65.
[48] Verma A, Pedrosa L, Korupolu M, et al. Large-scale cluster management at Google with Borg[C]//Proceedings of the Tenth European Conference on Computer Systems. New York: ACM, 2015. Doi: 10.1145/2741948.2741964.
[49] Hindman B, Konwinski A, Zaharia M, et al. Mesos: A platform for fine-grained resource sharing in the data center[C]//Proceedings of the 8th USENIX conference on Networked systems design and implementation. Berkeley, CA: USENIX Association, 2013: 295-308.
[50] VMware, Inc. Extend your workloads to hybrid cloud[EB/OL]. [2016-04-28]. http://www.vmware.com/cloud-services/infrastructure/.
[51] Dixon C, Mahajan R, Agarwal S, et al. An operating system for the home[C]//Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. Berkeley, CA: USENIX Association, 2012: 25.
[52] Quigley M, Conley K, Gerkey B, et al. ROS: An open-source Robot Operating System[C/OL]. [2016-04-28]. http://ai.stanford.edu/~ang/papers/icraoss09-ROS.pdf.
[53] 卢锡城, 王怀民, 王戟. 虚拟计算环境iVCE: 概念与体系结构[J]. 中国科学(信息科学), 2006, 36(10): 1081-1099.
Lu Xicheng, Wang Huaimin, Wang Ji. Virtualized computing environment iVCE: Concepts and architecture[J]. Science in China (Information Sciences), 2006, 36(10): 1081-1099.
[54] 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[J]. 电子学报, 2011, 39(5): 985-990.
Zhang Yaoxue, Zhou Yuezhi. A new cloud operating system: Design and implementation based on transparent computing[J]. Acta Electronica Sinica, 2011, 39(5): 985-990.
[55] Yuan P, Guo Y, Chen X. Towards an operating system for the campus[C]//Proceedings of the 5th Asia-Pacific Symposium on Internetware. New York: ACM, 2013. Doi: 10.1145/2532443.2532468.
[56] 华为技术有限公司. FusionSphere云操作系统 [EB/OL]. [2016-04-22]. http://e.huawei.com/cn/products/cloud-computing-dc/cloud-computing/fusionsphere/fusionsphere.
Huawei Technologies Co., Ltd. FusionSphere cloud operating system [EB/OL]. [2016-04-22]. http://e.huawei.com/cn/products/cloud-computing-dc/cloud-computing/fusionsphere/fusionsphere.

Development and present situation of Internetware operating systems

MEI Hong[1,2,3], GUO Yao[1,2]

1. School of Electronics Engineering and Computer Science, Peking University , Beijing 100871, China
2. Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education, Beijing 100871, China
3. Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240, China

Abstract Computer software is the soul of a computer system, while operating system is the core of software runtime supporting technology. In the Internet era, both software running environments and software development methods have been undergoing dramatic changes, thus we urgently need a new software paradigm for internet computing. Chinese researchers have named this new paradigm "internetware", which has brought a series of challenges to software technologies including operating systems. This article focuses on operating systems for internetware, or internetware operating systems. We first briefly review the history of operating systems, and then introduce the concepts, basic characteristics and key supporting technologies for internetware operating systems. After summarizing the current development status of internetware operating systems, we present a brief outlook and trends for future operating systems.

Keywords operating systems; internetware; internetware operating system; software-defined

(编辑 傅雪)