CHAPTER 5 OPERATING SYSTEMS (操作系统)

5.1 INTRODUCTION(简介)

Without its software, a computer is basically a useless lump of metal. With its software, a computer can store, process, and retrieve information; display multimedia documents; search the Internet; and engage in many other valuable activities to earn its keep. Computer software can be divided roughly into two kinds: system programs, which manage the operation of the computer itself, and application programs, which perform the actual work the user wants. The most fundamental system program is the operating system, which controls all the computer's resources and provides the base upon which the application programs can be written[1].

A modern computer system consists of one or more processors, some main memory (often known as RAM—Random Access Memory), disks, printers, network interfaces, and other input/output devices. All in all, a complex system. Writing programs that keep track of all these components and use them correctly, let alone optimally, is an extremely difficult job. If every programmer had to be concerned with how disk drives work, and with all the dozens of things that could go wrong when reading a disk block, it is unlikely that many programs could be written at all.

An operating system is software that manages the overall operation of the computer system. Its primary purpose is to support application programs. The parts of an operating system can be grouped into four broad functional categories. One set of parts forms the shell or user interface; another set of parts is responsible for coordinating multiple computers in a network; a third set coordinates multiple tasks or basic units of work within a single computer; and, finally, the kernel of the operating system is software that ties the hardware to the software and performs such tasks as keeping track of everything in memory and managing the flow of information to and from disks, the keyboard, and the display screen.

Operating systems have developed over past thirty years for two main purposes. First, they provide a convenient environment for the development and execution of programs. Second, operating systems attempt to schedule computational activities to ensure good performance of the computing system.

The operating system must ensure correct operation of the computer system. To prevent user programs from interfering with the proper operation of the system, the hardware was modified to create two modes: user mode and monitor mode. Various

instructions (such as I/O instructions and halt instructions) are privileged and can only be executed in monitor mode. The memory in which the monitor resides must also be protected from modification by the user. A timer prevents infinite loops. Once these changes (dual mode, privileged instructions, memory protection, timer interrupt) have been made to the basic computer architecture, it is possible to write a correct operating system.

NOTES

[1] which 引导非限制性定语从句,修饰 operating system。

KEYWORDS

system program	系统程序	timer	定时器
processor	处理器	keyboard	键盘
display screen	显示屏	instruction	指令

第5章 操作系统

5.1 简 介

没有软件,计算机基本上是一堆废铁。安装了软件的计算机可以储存、处理、检索信息,显示多媒体文档,在因特网上冲浪,从事其他更有意义的活动等。计算机软件可以分为两类:管理计算机本身操作的系统软件和执行用户工作的应用软件。最基本的系统程序即是操作系统,它控制着所有计算机的资源,并提供编写应用程序的基础。

一个现代的计算机系统包括一个或多个处理器,主存(经常被称为 RAM,即随机存储器)、磁盘、打印机、网络接口和其他输入/输出设备。所有这些,构成了一个复杂的计算机系统。编写程序来跟踪所有这些组件并且正确地使用它们,是一项十分复杂的工作。如果每个程序员都必须去关心磁盘是怎样工作的,并且当读取磁盘块时所有都能查找出来的错误也都要面对的话,许多程序根本无法编写。

一个操作系统是能够处理计算机系统中的所有操作的软件。它的主要目的是支持应用程序。一个操作系统可以划分为 4 个功能模块。一部分形成用户接口;另一部分负责协调网络中的多台计算机;第三部分是在一台计算机内部协调多个任务或基本的工作组件;最后一部分是操作系统的核心,用于绑定硬件和软件,并且执行诸如跟踪内存的每一份工作和处理磁盘、键盘和显示器之间的信息流。

操作系统在过去的 30 年里主要在两个方面发展。第一,它们为程序的开发和执行提供了方便的环境。第二,操作系统通过强大的计算能力来确保计算机系统的良好性能。

操作系统必须确保计算机系统的正确操作。为了防止用户干扰系统的正确操作,硬件被修改为两种工作模式:用户模式和监控模式。各种各样的指令(例如 I/O 指令和暂停指令)仅仅能在监控模式下执行。监视器所处位置的内存也必须保护。计时器防止无限制地

循环。一旦这些变化(双重模式、特权指令、内存保护、计时中断)成为基本的计算机体系结构,就可以编写正确的操作系统。

5.2 HISTORY OF OPERATING SYSTEMS (操作系统的历史)

Operating systems have been evolving through the years. In the following sections we will briefly look at this development. Since operating systems historically have been closely tied to the architecture of the computers on which they run, we will look at successive generations of computers to see what their operating systems were like[1]. This mapping of operating systems generations to computer generations is crude, but it does provide some structure where there would otherwise be none.

The first true digital computer was designed by the English mathematician Charles Babbage (1792—1871). Although Babbage spent most of his life and fortune trying to build his "analytical engine", he never got it working properly because it was purely mechanical, and the technology of his day could not produce the required wheels, gears, and cogs to the high precision that he needed. Needless to say, the analytical engine did not have an operating system.

As an interesting historical aside, Babbage realized that he would need software for his analytical engine, so he hired a young woman, named Ada Lovelace, who was the daughter of the famed British poet, Lord Byron, as the world's first programmer. The programming language Ada is named after her.

The Zeroth Generation (1940s) Early computing systems had no operating system. Users had complete access to the machine language. They hand-coded all instructions.

The First Generation (1950s) The operating systems of the 1950s were designed to smooth the transition between jobs. Before the systems were developed, a great deal of time was lost between the completion of one job and the initiation of the next. This was the beginning of batch processing systems in which jobs were gathered in groups or batches. Once a job was running, it had total control of the machine. As each job terminated (either normally or abnormally), control was returned to the operating system that "cleaned up after the job" and read in and initiated the next job.

The Second Generation (Early 1960s) The second generation of operating systems was characterized by the development of shared systems with multiprogramming, and the beginnings of multiprocessing. In multiprogramming systems several user programs are in main storage at once and the processor is switched rapidly between the jobs. In multiprocessing systems, several processors are used on a single computer system to increase the processing power of the machine.

Device independence began to appear. In first generation systems, a user wishing to write data on tape had to reference a particular tape drive specifically. In second generation

systems, the user program specified only that a file was to be written on a tape device with a certain number of tracks and a certain density. The operating system located an available tape device with the desired characteristics and instructed the operator to mount a tape in that drive.

Timesharing systems were developed in which users could interface directly with the computer through terminals. Timesharing systems operate in an interactive or conversational mode with users. The user types a request to the computer, the computer processed the request as soon as it can (often within a second or less), and a response (if any) is typed on the user's terminal. Conversational computing made possible great strides in the program development process. A timesharing user could locate and correct errors in seconds or minutes, rather than suffering the delays, often hours of days, in batch processing environments.

The Third Generation (Mid 1960s to Mid 1970s) The third generation of operating systems effectively began with the introduction of the IBM System/360 family of computers in 1964. Third generation computers were designed to be general-purpose systems. The concept sold a lot of computers, but users running particular applications that did not require this kind of power paid heavily in increased run-time overhead, learning time, debugging time, maintenance, etc.

Third generation operating systems were multimode systems. Some of them simultaneously supported batch processing, timesharing, real-time processing, and multiprocessing. They were large and expensive. Nothing like them had ever been constructed before, and many of the development efforts finished well over budget and long after scheduled completion. A notable exception to this is the UNIX system developed at Bell Laboratories.

These systems introduced to computer environments a greater complexity. The systems interposed a software layer between the user and the hardware. This software layer was often so thick that a user lost sight of the hardware, and saw only the view created by the software. To get one of these systems to perform the simplest useful task, users had to become familiar with complex job control languages to specify the jobs and their resource requirements.

The Fourth Generation (Mid 1970s to Present) The fourth generation systems are the current state of the art. Many designers and users are still smarting from their experiences with third generation operation systems and are careful before getting involved with complex operating systems. With the widespread use of computer networking and online processing, users gain access to networks of geographically dispersed computers through various types of terminals. The microprocessor has made possible the development of the personal computer, which is one of the most important developments of social consequence in the last several decades. Personal computers are often equipped with data communications interfaces, and also serve as terminals. The user of a fourth generation

systems no longer may communicate with geographically dispersed systems. Security problems have increased greatly with information now passing over various types of communications lines. Encryption is receiving much attention—it has become necessary to encode highly proprietary data.

The percentage of the population with access to computers in the 1980s is far greater than ever before and growing rapidly. It is common to hear the term user friendly denoting systems that give users of average intelligence easy access to computer power. The highly symbolic, mnemonic, acronym-oriented user environments of the 1960s and 1970s are being replaced in the 1980s by menu-driven systems that guide the user through various options expressed in simple English.

An interesting development that began taking place during the mid-1980s is the growth of networks of personal computers running network operating systems and distributed operating systems. In a network operating system, the users are aware of the existence of multiple computers and can log in to remote machines and copy files from one machine to another. Each machine runs its own local operating system and has its own local user (or users).

Network operating systems are nor fundamentally different from single-processor operating systems. They obviously need a network interface controller and some low-level software to drive it, as well as programs to achieve remote login and remote file access, but these additions do not change the essential structure of the operating systems.

A distributed operating system, in contrast, is one that appears to its users as a traditional uniprocessor system, even though it is actually composed of multiple processors. The users should not be aware of where their programs are being run or where their files are located; that should all be handled automatically and efficiently by the operating system.

True distributed operating systems require more than just adding a little code to a uniprocessor operating system, because distributed and centralized systems differ in critical ways. Distributed systems, for example, often allow applications to run on several processors at the same time, thus requiring more complex processor scheduling algorithms in order to optimize the amount of parallelism.

NOTES

[1] since 在这里表原因。

[2] that 引导宾语从句。

KEYWORDS

programmer	程序员	generation	代
multiprogramming	多道程序设计	time-sharing	时间共享
real-time processing	实时处理	encryption	加密,密码术
remote machine	远程计算机	mnemonic	便于记忆的

acronym-oriented	只取首字母的	interface	接口
distributed operating system	分布式操作系统		

5.2 操作系统的历史

操作系统这些年有了很大的发展。下面简单回顾一下它的发展。由于历史上操作系统在运行时紧紧地与计算机的系统结构结合在一起，我们将通过计算机的更新换代来看看其操作系统的发展。以下操作系统与计算机发展的描述是粗糙的，但它提供了基本的框架。

最早的真正的数字计算机是由英国的数学家 Charles Babbage（1792—1871 年）设计的。尽管 Babbage 花费了大量的精力和物力去尝试建立他的“解析机”，但该机器从未正确工作，因为它是纯机械的，并且他那个年代的这个技术还不能保证生产的车轮、齿轮和嵌齿到所需要的精度。不管怎么说，这个解析机没有一个操作系统。

作为一个有趣的历史，Babbage 认识到他需要解析机的软件，因此他聘请了名叫 Ada Lovelace 的年轻人，她是著名的英国诗人 Lord Byron 的女儿，世界上首位程序员。程序设计语言 Ada 就是以她命名。

萌芽时期的计算机（20 世纪 40 年代） 最早的没有操作系统的计算机。用户对机器语言具有完全的访问权限，他们处理所有的指令代码。

第一代计算机（20 世纪 50 年代） 20 世纪 50 年代的操作系统是为了工作之间的协调而设计的。在系统开发之前，大量的时间浪费在一个工作的结束和另一个工作的开始之间的衔接上。这就是把工作聚集在一起或成批工作，即批处理系统的开始。一旦一个任务正在运行，它拥有对机器的全部控制。当一个工作终止（正常或不正常），控制都返回到操作系统，任务结束之后清空并为下一个任务进行读取数据及初始化的工作。

第二代计算机（20 世纪 60 年代初期） 操作系统的第二代是通过多道程序设计和多处理系统的开发而实现的。在多道程序设计系统中许多用户程序立刻调入主存，处理器在任务之间快速交换。在多处理系统中，许多处理器在单个计算机系统中使用以提高机器的处理能力。

设备的独立性开始出现。在第一代系统中，用户希望把数据写入磁带中必须使用特定的磁带驱动设备。在第二代计算机中，用户程序仅仅在一个文件必须被写入带有一系列磁道和确定的磁道密度的磁带设备时指定。操作系统对使用的磁带进行定位并且指示操作者在驱动器上放置磁带。

时间共享系统是在用户能直接通过终端与计算机接口基础上发展的。时间共享系统的操作建立在与用户的交互式或会话式的模式上。用户向计算机发出请求，计算机在能够处理时立刻处理，并且在用户终端上返回响应。会话式计算使程序发展过程的大幅度提高成为可能。一个时间共享用户在批处理环境下能在几秒或几分之内定位和纠正错误，而不会耽误几个小时或几天。

第三代计算机（20 世纪 60 年代中期到 20 世纪 70 年代中期） 第三代操作系统是伴随着 1964 年 IBM 的 System/360 系列的计算机出现而发展的。第三代计算机是为了通用系统而设计的。在这个理念下售出了许多台计算机，而且特殊应用的用户不需要为这种不断增长的运行负载、学习时间、调试时间、维护时间而付出高昂的费用。

第三代操作系统是多模式系统。它们中的一些同时支持批处理、时间共享、实时处理和多处理。它们是庞大和昂贵的。以前还从来没有像它们那样的构造，并且许多开发项目远远超出了预算，计划的时间表也一超再超。不过由贝尔实验室开发的 UNIX 系统是一个值得一提的例外。

这些系统给计算机环境营造了一个更为复杂的环境。系统在用户和硬件之间提出了一个软件层。这个软件层经常很拥挤，以至于用户看不到硬件，只能看到软件部分。使用这些系统之一来执行一个简单的有用的任务，用户必须对复杂工作的控制语言熟练掌握并指定这些工作任务级和其资源需求。

第四代计算机（20 世纪 70 年代中期到现在） 第四代计算机体现了当今工艺的发展水平。许多设计者和用户仍然借鉴第三代操作系统的经验，在接受复杂的操作系统之前小心翼翼。随着计算机网络和在线处理的广泛应用，用户通过各种各样的终端访问分散在各地的计算机网络。微处理器使个人计算机的发展成为可能，这是在过去的几十年来最重要的社会发展之一。个人计算机经常配备数据通信接口，同时也为终端服务。第四代计算机操作系统的用户不再可能与地理上分散的系统通信，信息的安全性现在成为不同类型的通信线路中日益增长的问题，加密得到更多的重视——它成为那些高度保密的私人数据的必需的方法。

到 20 世纪 80 年代使用计算机的比例远远超过以前并且在快速地增长。普通的大众可以很轻松地驾驭计算机。60 年代和 70 年代所使用的符号化的、便于记忆的、只取首字母的用户环境在被 20 世纪 80 年代菜单驱动系统所取代，它是一种引导用户用简单英语表达的系统。

80 年代中期一个有意义的发展是个人计算机运行网络操作系统和分布式操作系统的网络的成长。在网络操作系统中，用户会意识到多个计算机的存在，而且能够登录到远程计算机上从一台机器向另外的机器拷贝文件。每台计算机运行自己的本地操作系统并且有自己的用户。

网络操作系统也并不是根本上不同于单处理器操作系统，但它们显然需要一个网络接口控制器和一些底层的软件来驱动，就像程序实现远程登录和远程文件访问一样，但是这些加起来也不能改变操作系统的重要结构。

相反地，分布式操作系统以传统的单用户系统的身份出现，即使它实际上是由多处理器组成的也是如此。用户用不着知道他们的程序运行或文件被访问的位置，一切都会被自动且有效地被操作系统所处理。

分布式操作系统不仅仅需要增加一小段代码到一个单处理器操作系统，因为分布式和中心系统关键部分是不同的。例如，分布式系统经常允许同时在许多处理器上运行应用，这样就需要更多的复杂处理算法来优化并行工作的数量。

5.3 TYPES OF OPERATING SYSTEMS（操作系统的类型）

The types of operating systems include single program, multiprogramming, multiprocessing, and virtual machine operating systems. These operating systems can be classified by two criteria: (1) whether they allow more than one user to use the computer

at the same time and (2) whether they allow more than one program to run at the same time.

Single program operating systems allow only a single user to run a single program at one time. This was the first type of operating systems developed. For example, if you are working on a personal computer with a single program operating system you can load only one application, such as a spreadsheet, into main memory. If you want to work on another application, such as word processing, you must exit the spreadsheet application and load the word processing program into memory.

Multiprogramming operating systems, also called multitasking operating systems, allow more than one program to be run at the same time on one computer. Even though the CPU is only capable of working on one program instruction at a time, its capability to switch back and forth between programs makes it appear that all programs are running at the same time. For example, with a multiprogramming operating system the computer could be performing a complex spreadsheet calculation and at the same time be downloading a file from another computer while the user is writing a memo with the word processing program.

Multiprogramming operating systems on personal computers can usually support a single user running multiple programs. Multiprogramming operating systems on some personal computers and most minicomputers and mainframes can support more than one user running more than one program. This version of a multiprogramming operating system is sometimes called a multiuser-multiprogramming operating system. Most of these operating systems also allow more than one user to be running the same program. For example, a wholesale distributor may have dozens of terminal operators entering sales orders using the same order entry program on the same computer. [1]

Computers that have more than one CPU are called multiprocessors. A multiprocessing operating system coordinates the operations of computers with more than one CPU. Because each CPU in a multiprocessor computer can be executing one program instruction, more than one instruction can be executed simultaneously. Besides providing an increase in performance, most multiprocessors offer another advantage. If one CPU fails, work can be shifted to the remaining CPUs. In addition to an extra CPU, some systems, called fault-tolerant computers, are built with redundant components such as memory, input and output controllers, and disk drivers. If any one of the components fails, the system can continue to operate with the duplicate component. Fault-tolerant systems are used for airline reservation systems, communication networks, bank teller machines, and other applications where it is important to keep the computer operating at all times.

A virtual machine (VM) operating system allows a single computer to run two or more different operating systems. The VM operating system allocates system resources such as memory and processing time to each operating system. To users, it appears that

they are working on separate systems, hence the term virtual machine[2]. The advantage of this approach is that an organization can run different operating systems (at the same time) that are best suited to different tasks. For example, some operating systems are best for interactive processing and others are best for batch processing. With a VM operating system, both types of operating systems can be run concurrently.

NOTES

[1] wholesale distributor 的含义是“批发商”。

[2] appears 在此处的含义是“看起来”。

KEYWORDS

virtual machine operating system	虚拟机器操作系统	criteria	标准
batch processing	批处理	download	下载
wholesale distributor	批发商	fault-tolerant	容错
virtual machine (VM)	虚拟机		

5.3 操作系统的类型

操作系统的类型包括单程序操作系统、多程序操作系统、多处理器操作系统和虚拟机操作系统。这些操作系统可以用两个标准来划分：

(1) 它们是否允许多个用户同时使用计算机。

(2) 它们是否允许多个程序同时运行。

单程序操作系统在同一时刻内只允许一个用户运行一个程序，这是最初开发的操作系统类型。例如，如果你正在使用单程序操作系统的个人计算机，那么你只能将一个应用程序(如电子表格)装入主存储器。如果想使用另一个应用程序(如字处理应用程序)，就必须退出电子表格应用程序，然后再将字处理应用程序装入主存储器。

多程序操作系统，也称为多任务操作系统，它允许在一台计算机上同时运行多个程序。CPU 同时只能对一条程序指令进行处理，但是它的前后台切换功能使它看起来似乎是同时运行了多个程序。例如，使用多任务操作系统，用户在使用字处理程序书写一个便函的同时，计算机还可以进行复杂的电子表格计算，以及从另一台计算机下载文件等。

个人计算机上的多程序操作系统通常能支持单个用户运行多个程序。一些个人计算机、大多数小型计算机和大型计算机上使用的多程序操作系统允许多个用户运行多个程序，多程序操作系统的这个版本有时也称为多用户-多程序操作系统。这些操作系统中的绝大多数也允许多个用户同时运行同一个程序。例如，批发商可能有多个终端操作者，他们在同一台计算机上使用同一个订单输入程序，输入销售订单。

有多个 CPU 的计算机称为多处理器计算机，多处理操作系统协调有多个 CPU 的计算机的操作。由于多处理器计算机中的每一个 CPU 都能执行一条程序指令，因此这种计算机能同时执行多条指令。除了增强处理能力以外，大多数多处理器计算机还具有另外一个优点。如果一个 CPU 出现故障，它的工作可以转交给其他 CPU 执行，除了额外的 CPU，一些称为容错计算机的系统，还有一些冗余部件，如存储器、输入输出控制器以及磁盘驱动器

等。如果任何一个部件出现故障,系统还可以通过另外的部件继续工作。容错系统被用于民航订票系统、通信网络、银行取款机以及其他需要计算机始终保持正常工作的场合。

虚拟机(VM)操作系统允许一台计算机运行两个或多个操作系统。虚拟机操作系统为每一个操作系统分配系统资源,如内存和处理时间等。对用户而言,它们看起来似乎是在不同的系统上工作,因而得名"虚拟机器"。这种方法的优点是:计算机能同时运行,并能最好地适应不同任务的不同操作系统。例如,有些操作系统适合于交互操作,而另一些则更适合于批处理。通过虚拟机操作系统,就能同时运行这两种操作系统。

5.4 FUNCTIONS OF OPERATING SYSTEMS (操作系统的功能)

All application programs share some tasks in common. They include accepting characters typed at the keyboard, displaying information on the screen, managing information on a disk, and managing information in memory. The operating system takes care of the details of these tasks. A most important example of how operating systems support application programs is the task of managing files. A file is a named collection of information. Whether your application is general or special purpose, your program needs to store information in files.

The operating system takes care of:

- Formatting the disk, which involves electronically preparing the disk to be able to store files.
- Managing the location of information on the disk.
- Checking to make sure that errors do not occur when reading to and writing from the disk.
- Performing the input and output necessary to retrieve and store information on the disk.

Errors may occur in the CPU and memory hardware (such as a memory error or a power failure), in I/O devices (such as a parity error on tape, a card jam in the card reader, or the printer out of paper), or in the user program (such as an arithmetic overflow, an attempt to access illegal memory location, or using too much CPU time). For each type of error, the operating system should take the appropriate action to assure correct and consistent computing.

Operating systems also manage the other components of a computer system. They support programs, called device drivers, which control the various hardware devices, such as the keyboard, display screen, and printer. The device driver translates instructions from the application program wants to print something, it simply sends the information and the appropriate instructions to the operating system, which, in turn, calls upon the printer device driver to manipulate the printer to perform the desired task.

KEYWORDS

format 格式化 retrieve 检索