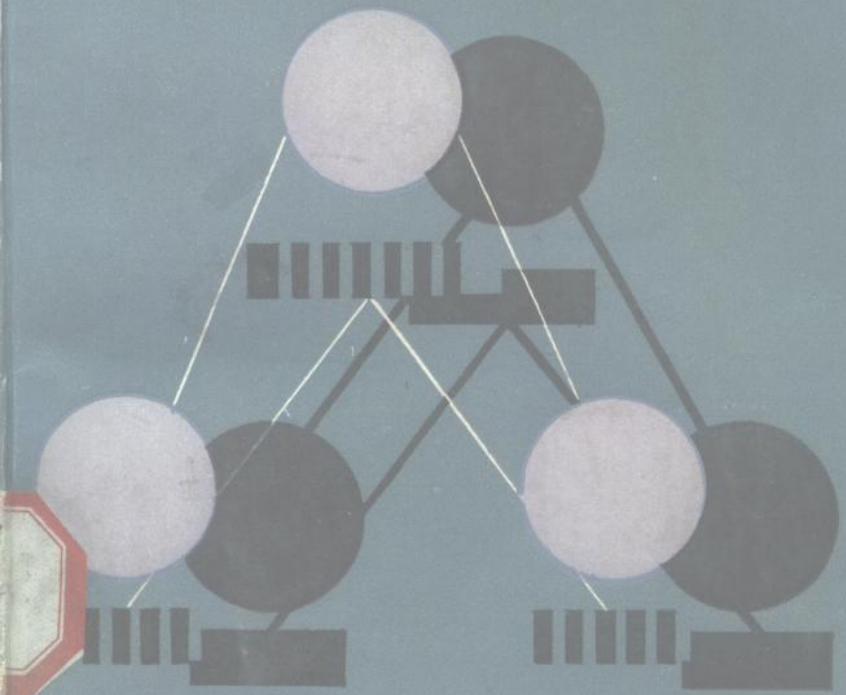
操作系统结构分析

杨芙清　俞士汶　著



北京大学出版社

操作系统结构分析

杨芙清　俞士汶　著

北京大学出版社

操作系统结构分析

杨芙清　俞士汶　著

责任编辑：邱淑清

*

北京大学出版社出版
（北京大学校内）

北京大学印刷厂印刷

新华书店北京发行所发行　各地新华书店经售

850×1168毫米 32开本 10.75印张 270千字

1986年5月第一版　1986年5月第一次印刷

印数：1—32,000册

*

统一书号：13209·100　定价：2.10元

内 容 简 介

本书以作者亲自参加设计的三个实际系统（150机系统、计算机激光编排系统和DJS 200/XT2系统）为模型，分析了操作系统的三种结构，并且介绍了操作系统设计的三个部分，即功能设计、算法设计和结构设计。全书共分五章。第一章是总纲，第二章介绍模块组合结构，第三章介绍进程分层结构，第四章是系统程序设计语言XCY的导引，第五章讲述了层次管程结构。全书是一个完整的体系，各章又具有相对的独立性。

本书可作为大学计算机科学系高年级学生和研究生的教材或教学参考书，也可供计算机科学技术人员参考。

序　言

操作系统是当代计算机系统的必要组成部分，是最重要的系统软件之一。无论是计算机的研制生产单位，还是用戶，都十分重视操作系统的开发。国际上自六十年代中期开始发展，直到现在这个领域的研究工作仍很活跃。国內从六十年代末也开始了这方面的研究。随着现代化建设事业的发展，自己研制的和从国外引进的计算机系统正在迅速增加，应用范围也在迅速拓广。因此，越来越多的科技人员迫切要求掌握操作系统。

在操作系统的发展前期，人们关心的是系统的功能和效率。随着硬、软件技术的进步，人们日益认识到系统结构直接影响到系统的性能，因此，近年来普遍重视操作系统结构和结构设计方法的研究。可以将操作系统的结构大致划分为三种类型：模块组合结构、进程分层结构和层次管程结构。这种划分与操作系统的发展阶段也是相适应的。探索操作系统结构发展的规律性，无疑是一项很有意义的工作。

本书以作者亲自参加设计的三个实际系统为模型，分析了操作系统的三种结构，并对操作系统这门新兴学科中的若干重要概念，如多道程序、进程、系统程序设计语言、管程等作了深入浅出的详细的论述。全书共分五章。第一章是总纲，扼要地叙述了三种结构的主要特征、优缺点及适用范围。第二章介绍了模块组合结构。第三章介绍了进程分层结构。第四章介绍了 XCY 语言，这一章既是第五章的预备知识，也可作为系统程序设计语言的导引。第五章介绍了层次管程结构。每章后面都列了参考文献。全书是一个完整的体系，前后照应，但各章又具有相对的独立性，读者可以根据需要，分开来学习，也可以在读了后面各章后再读

第一章，理解会更深刻。作者希望本书对读者掌握现有的系统，设计新的系统，研究新的结构设计方法，都将有所裨益。

本书虽然立足于三个具体的系统，但并不是这三个系统的说明书。材料的取舍与加工服从全书体系的需要。书中使用的术语力求前后一致，力求与当前通用的一致，这样与原系统使用的也就有所不同。

杨芙清、徐联舫、俞士汶、陈成森、李玉珍、谢呈奇等同志参加了150系统的研制。CL 系统由陈堃銶负责总体设计，俞士汶、付国泰参加了研制工作。杨芙清主持了 DJS200/XT2 的研制，先后参加工作的还有徐联舫、金德鑫、朱慧眞、邹悦、方裕、柳纯录、邵维中、苏渭珍、刘永进、沈琼华、沈承凤等同志。实际的系统都是集体劳动的成果。对这些同志的辛勤劳动，作者谨致深切的谢意。这些同志中有的原来就是协作单位的，有的已经调离北大，作者不会忘记他们对本书所作出的贡献。

在本书写作过程中，方裕做了大量的整理工作。南京大学徐家福教授提出过不少宝贵意见。科学院计算所仲萃豪副研究员在百忙中审阅了全书。作者对他们致以衷心的谢意。

书中的内容曾由作者对研究生、大学生和科技人员等不同对象进行过多次讲授，本书是在讲稿的基础上反复修改和补充而写成的。

由于水平所限，书中肯定会有不少缺点错误，作者眞诚地希望广大读者批评指正。

作　者

1983年9月于北京大学

目　录

第一章　操作系统结构设计

操作系统是一门新兴的学科，是随着大型计算机系统的问世而逐步发展起来的，国外是六十年代中期发展的，国内是六十年代末开始发展的。

§1.1　操作系统概述

1.1.1　计算机的早期使用方式

为了说清楚操作系统是干什么的，我们先回顾一下计算机的早期使用方式。

在第一代的计算机中，人们是采用手工方式使用计算机的。用户按事前分配好的上机时间一个接一个地进入机房，并通过以下步骤使用计算机：

(1) 将纸带装在输入机上，按输入键，将纸带上的程序和数据输入内存。

(2) 在控制台上拨好启动地址，按启动键，程序开始执行。

(3) 当程序执行到需要输出中间结果和最后结果时，则启动打印机。只有当打印机工作结束后，程序才可继续执行。

(4) 若程序执行过程中发生故障需要用户干预时，则停机显示，由用户在控制台上进行处理。

(5) 计算完毕后取下纸带，取走结果。

这样的使用方式主要有以下两个特点：

1. 独占

使用计算机的用户独占全部计算机资源，包括中央处理

机CPU、内存贮器、外存贮器和所有外部设备，而其中部分资源甚至大部分资源往往是空闲的。

2. 串行

各个用户是串行工作的，用户和计算机之间是串行工作的，计算机的各个部件之间也是串行工作的。

这种独占和串行的工作方式造成了计算机资源的利用很不充分，计算机的使用效率很低。如果说，由于早期计算机的速度还比较低，这种矛盾尚不突出的话，那么，当计算机进入第二代后，早期的手工使用方式造成的计算机资源严重浪费的问题就相当严重了。

例如，有一个作业在运算速度为每秒1万次的计算机上需计算1小时，若手工操作需用3分钟，则操作时间与机器有效运行时间之比为1:20。但是，当计算机的运算速度提高到每秒60万次时，计算时间就缩短为1分钟，而操作时间仍然需用3分钟，则操作时间与机器有效运行时间之比变成了3:1，即相当于计算机每工作1分钟，因手工操作却要停顿3分钟，机器经常处于空闲状态。

同时，由于主机和外部设备之间速度相差悬殊以及计算机各部件的串行工作方式也造成了主机长期空闲等待现象，从而也造成计算机资源的严重浪费。

1.1.2 操作系统的概念

计算机的早期使用方式说明了计算机硬件给用户提供的接口（又叫界面，即使用环境）是不便使用的。为了驯服这种"裸机"，人们研制了操作系统，它管理基本的硬件资源，不仅提高了计算机的使用效率，而且为用户提供了一个方便的操作环境。在这种意义上说，操作系统是一个资源管理程序，它按照某种预定的目标(例如，计算机各部件的利用率要高，作业的吞吐量要大等)，运用某些策略和机构来管理这些资源。有了操作系统，用户就不

再直接使用计算机硬件，而是通过这个新的接口来使用计算机系统，用户可以不了解计算机的具体细节而放心大胆地使用计算机系统的各种资源。

概括地说，操作系统是一套管理和控制计算机的程序，是为了发挥计算机的潜力、提高计算机的使用效率、方便用户使用而设计的。因此，操作系统是用户程序和计算机硬件之间的接口，是给裸机穿上的第一件外衣，它既改善了用户使用计算机的环境，又为进一步改善提供了可靠的基础。配置了操作系统的计算机成为一台功能更强、使用更方便、效率提高了的虚拟计算机，这种关系如图1-1所示。

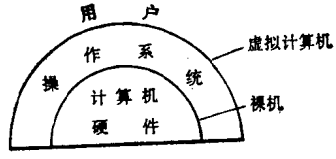


图1-1　裸机和虚拟机

1.1.3　操作系统的主要特征

为充分发挥计算机各部件平行工作的能力，操作系统采用了并发程序设计技术，设计多道程序系统。简言之，所谓并发程序指的是这样一类程序，它们在其执行过程中，在时间上有交迭现象。例如，在单处理机情况下，在内存中存放多个用户程序，当一个程序由于某种原因，如等待某种资源或某种事件的发生而不能继续执行时，系统就调度另一程序运行。这样一来，一个程序的头一个操作在另一个程序的最后一个操作完成之前就开始了，这种现象就是时间上的交迭现象。并发程序设计技术的应用使计算机的各种资源得到充分的利用，计算机的各个部件也就有节奏有规律地活动。由于操作系统本身具有并发性，并发程序设计技术也可被运用于操作系统本身的设计，从而使操作系统成为一种

具有**并发结构**的大型软件，它的主要目的是使多个程序能有效地**共享**计算机资源，而且这些程序对计算机资源的需求情况是不可预知的，是动态产生的。并发和共享是现代操作系统的主要特征，是使操作系统能够提高计算机使用效率的主要技术手段，也是使操作系统成为复杂程度高、研制周期长、正确性难以保证的大型软件的典型实例的主要原因。

1.1.4 操作系统的设计

1. 目标和要求

操作系统是计算机系统的重要组成部分，是计算机系统工作时经常起作用的程序。同时，它又是一种复杂程度高的大型程序，为使计算机系统可靠而有效地工作，必须配置一个高质量的操作系统。

一个高质量的操作系统应具有简明性、可靠性、有效性、可适应性和可移植性。无简明性，人们无法去了解一个大型程序的设计目的和细节。无可靠性，将严重影响使用效果。有效性指的是系统效率高，即用于非生产性的时间与空间的开销少。可适应性指的是一种特定计算机系统环境中的软件对于另一种计算机系统环境的适应能力。研制一个大型软件的费用十分昂贵，而使用要求又在不断变化，经常要求对系统作些修改，以适应环境和要求的变化，如果一个系统没有可适应性，它将是一个僵死的系统，是无生命力的。可移植性指的是把一个程序从一个计算机系统环境中搬到另一个计算机系统环境中能正常运行的特性。由于编制大型程序的工作量很大，所以，可移植性就成为软件设计中的一个重要准则，而影响可移植性的最大因素就是和机器有关部分的处理。

具有简明性、可靠性、可适应性的系统称为可维护的系统，它是容易管理的。可适应性和可移植性合称为灵活性。

2. 设计阶段

设计一个操作系统，一般可分为三个阶段：功能设计、算法设计和结构设计。

功能设计指的是根据系统的设计目标和使用要求，确定所设计的操作系统应具备哪些功能。

算法设计是根据计算机的性能和操作系统的功能，来选择和设计达到系统功能的算法和策略，并分析和估算其效能。

结构设计则是按照系统的功能和特性要求，选择合适的结构，使用相应的结构设计方法将系统逐步地分解、抽象和综合，使操作系统结构清晰、简明、可靠、易读、易修改，而且使用方便，适应性强。

操作系统的三个设计阶段不是截然分开的，而是互相渗透的。总的目的是要求能够设计出一个具有好结构、高功效，而又具有所需要的功能的系统。

§1.2 操作系统的结构

1.2.1 程序结构

程序的可靠性和程序结构密切相关。所谓程序结构有两层含义。一是指程序的整体结构，即由程序的成分构造程序的方式，如 PASCAL 语言程序是分程序结构，FORTRAN 语言程序是块结构，EUCLID 语言程序是模块结构等等；二是指程序的局部结构，即程序的数据结构和控制结构。本书主要讨论前者。

我们的目标是设计一个能正确实现功能要求的程序，也就是说，要设计一个可靠的程序，运行的结果能正确地反映设计要求。为此，我们必须构造出一个结构良好的程序。所谓程序的结构良好性，指的是程序的结构清晰、易读、易维护、可移植，当然这样的程序也要易验证、易调试和易修改。

结构化程序设计就是为了使程序有一个合理的结构，以便于保证和验证其正确性而规定的一套如何进行程序设计的准则和方法。按照这样一套准则和方法设计出来的程序就是结构化程序。

模块化是一种结构程序设计方法，它指的是把一个程序按功能分解成若干个彼此具有一定独立性，同时也具有一定联系的组成部分，这些组成部分就称作“模块”，每个程序由一个或多个模块组成。对大型程序来说，模块化是一种必然趋势。近年来的研究表明，在分析的基础上设计出一组“基本模块”和一组“构成法则”，然后由这些基本模块出发，通过有关的构成法则，构成所需的程序。如果这些模块是正确的，而且这些构成法则也是正确的，则得到的程序也是易于验证其正确性的。

1.2.2 软件结构

软件结构通常指大型程序系统的结构，与小规模程序结构具有本质的差别，后者主要研究程序的结构良好性，如易读性、易验证性等。大型程序是由小规模程序组成，因此，研究程序结构包括两个方面：首先是小规模程序怎样设计，然后如何把小规模程序组成大型程序。前者是局部结构，后者的结构只有在小规模程序能保证正确后，才能使组成的大型程序的正确性有保证。本书讨论的是如何把小规模程序组成大型程序问题，这些问题在研究小规模程序的结构设计时是不存在的，至少也是不严重的。

一个大型程序系统总是由一些模块组成。模块之间的接口指的是一个模块中的程序访问另一个模块内的程序或数据的方式，也可以说，模块间的接口就是指模块间传递和交换信息的关系。设计大型程序系统，对于接口必须十分重视。

1.2.3 操作系统结构

操作系统是一种大型软件。为了研制操作系统，必须分析它的结构。也就是要弄清楚如何把这一大型软件划分成若干较小的

模块以及这些模块间有着怎样的接口。在操作系统中，有些模块需要使用另一些模块中的程序，有些模块需要使用另一些模块内的数据，而系统的某些功能又需要若干模块协同工作来实现。

例如，有两个模块，一个是命令接收模块 A，另一个是命令处理模块 B。当命令接收模块 A 接收到来自操作员的命令后，就要把命令交给命令处理模块 B 去分析执行，于是，模块 A 和模块 B 之间就有了接口。在这个例子中，模块 A 要调用模块 B 的命令分析程序，并要将接收到的命令传送给模块 B 的命令分析程序。

操作系统还是一个具有并发特性的大型程序，模块间的接口是相当复杂的，信息交换也是十分频繁的，因而对结构的研究就显得更加重要了。

在操作系统的早期设计中，由于计算机结构还比较简单，系统规模也比较小，逻辑关系简单，人们关心的是系统的功能设计和效率。随着计算机结构的复杂化，应用范围的不断扩大，使用要求也不断提高，不仅要求有较强的系统功能，而且要求有较强的可适应性和可靠性。后来，又提出了容错的概念，即要设计一种在某些硬、软件失效的情况下仍能正常工作的系统。从而使人们日益认识到，系统结构直接影响到系统的性能。因此，近年来普遍重视了操作系统的结构和结构设计方法的研究，它已成为软件工程界的一个重要的研究领域。

§1.3 操作系统结构设计简述

在操作系统的发展过程中，产生了多种多样的系统结构，几乎每一个操作系统在结构上都有自己的特点。但从总体上看，到目前为止，大致可划分为三种类型，并且这种划分又和操作系统的发展阶段相一致。但是，这并不意味着后来发展的结构已经取代了早期的结构。目前这三种结构的系统都在实际运行中，各有其适用范围。本节将简述这三种结构的设计方法并加以比较。由

第二章开始将以具体的系统为背景，来详细分析这三种结构和它们的设计方法。

1.3.1 模块组合结构

操作系统是一个有多种功能的系统程序，可以看成是一个整体模块，也可看成是由若干个模块按一定的结构方式组成的。

早期的操作系统采用模块组合法（或称无序模块法、模块接口法等），系统中的模块不是根据程序和数据本身的特性而是根据它们完成的功能来划分的，数据基本上作为全程量使用。在系统内部，不同模块的程序之间可以不加控制地互相调用和转移，信息的传递方式也可以根据需要随意约定，因而造成模块间的循环调用，如图1-2所示。我们把这种操作系统结构称之为模块组合结构。它的主要优点是，结构紧密，接口简单直接，系统效率较高。它的缺点有以下三点：第一，模块间转接随便，各模块互相牵连，独立性差，系统结构不清晰；第二，数据基本上作为全程量处理，系统内所有模块的任一程序均可对其进行存取和修改，从而造成了各模块间有着更为隐蔽的关系。要更换一个模块，或修改、增加一个模块都比较困难，因为要弄清各模块间的接口，按

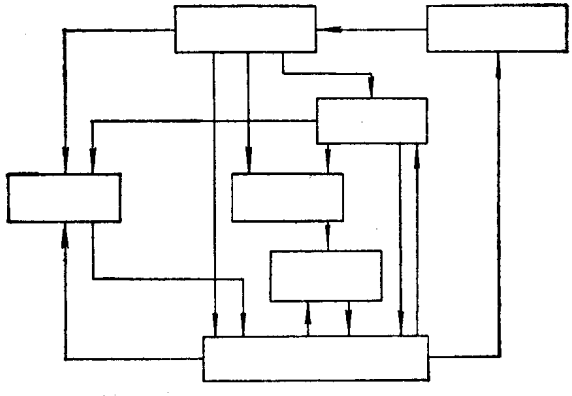


图1-2 模块组合结构示意图

当初设计时随意约定的格式来给信息，这是一件相当复杂的事；第三，由于模块组合结构常以大型表格为中心，为保证数据完整性，往往采用全局封中断的办法，从而限制了系统的并发性。对系统中实际存在的并发性也未能抽象出明确的概念，缺乏规格化的描述方法。所以，这种结构的可适应性比较差。它只适用于规模比较小、使用环境比较稳定却要求效率比较高的系统。

随着系统规模的不断增大，采用这种结构构造的系统的复杂性迅速增长，以致人们难以驾驭，这就促使人们去寻求新的结构概念和新的结构设计方法。

1.3.2 进程分层结构

并发性是操作系统的一个重要特征，在并发程序设计中，往往产生与时间有关的错误。为了描述操作系统的这种并发的动态的性质，在美国麻省理工学院设计的兼容分时系统 CTSS(Compatible Time-Sharing System) 中首先提出了描述并发性的结构概念——进程 (process)，或者称作任务 (task)，采用这个概念，把含有并发活动的系统划分为若干异步运行的，与时间无关的顺序程序模块。这样一来，操作系统的基本任务就是协调这些异步运行的进程，使它们能够协同工作。六十年代后期，研究的焦点是如何使共享系统资源的诸进程安全同步，即解决进程间的通讯问题。1968年 E.W.Dijkstra 提出的 T.H.E 系统发展了许多有关进程通讯的概念和技术，如临界区、信号灯及 P、V 操作等，实现了并发执行的诸进程间的同步，避免了与时间有关的错误的产生，并在此基础上发展了功能更强的通讯工具。

进程概念的引入使并发程序的结构更为清晰，同时也增加了系统的安全性。每个进程有自己的专用数据，共享数据则通过临界区使用。进程间传送信息可以利用系统提供的统一的消息机构实现，进程间控制的转移均由进程调度程序(又叫低级调度程序)统一管理，这个程序属于系统中不同于进程的一个特殊成分，即