实时磁盘操作系统

结构分析

RDOS RDOS RDOS RDOS

国家电子计算机工业总局技术服务公司

前　言

本书是由中国科学院计算所、物理所和四机部1015所等单位组成的NOVA计算机操作系统分析小组对RDOS操作系统所作的分析报告。原文在《电子计算机参考资料》1976年第5、6期上刊登过。这次出版前，我们请原分析小组的同志对文章作了订正。

本书主要从操作系统的内部结构和如何实现这个角度对NOVA实时磁盘操作系统作了介绍。在叙述过程中，始终贯彻由功能到实现、由使用到结构、由粗到细、由浅入深的原则。对于从事DJS 100系列计算机系统软件的工作人员来说，可以作为掌握RDOS内部结构的自学读物。对于某些读者来说，也可以当作掌握操作系统的入门书。

由于出版的准备工作比较仓促，错误之处在所难免。希望读者批评、指正。

国家电子计算机工业总局技术服务公司

一九八〇年五月

NOVA 计算机操作系统分析报告

目　　录

NOVA计 算机操作系统分析报告

第一章　分析设计操作系统的几种观点

本章主要讨论在分析、设计操作系统时所用的几种观点，目的是在具体介绍 RDOS 系统的实现之前，先交待一下我们在分析 RDOS 系统时所用的基本观点和基本方法，同时还介绍了操作系统中经常使用的几个重要术语和概念。本章先简要地介绍一下操作系统及其类型；然后，再分别介绍操作系统的用户观点、资源管理观点、进程观点以及模块分层观点，以便使我们能够从操作系统的基本功能、组成部分，工作过程以及体系结构这四个方面去了解操作系统的功能设计及其实现方法。

第一节　操作系统及其类型

操作系统是计算机系统的一个重要组成部分，它是由一组有机联系的程序组成的。设计操作系统的主要着眼点是：合理地组织计算机的整个工作流程，有效地管理计算机的硬软资源，从而达到充分发挥资源利用率，方便用户操作和使用之目的。

就操作系统所提供的工作环境来说，可把操作系统分成三大类：批加工系统，分时系统和实时系统。

1）批加工系统

这种系统的基本思想，就是每次把一批作业通过输入机提交给操作系统，并由系统把它们暂时存入后备存贮器，等待运行，以后，当系统需要调入新的作业时，系统再根据当时情况和用户要求，按照一定的原则，从它们之中调进一个或几个作业参加运行；当某个作业工作完毕或执行不下去时，系统转去执行另一作业，如此重复，直至这一批作业全部做完时为止。这种系统的特点是在系统运行过程中不允许用户和机器之间发生交互作用。亦即，用户一旦把作业提交给操作系统之后，他就完全独立于他的作业，直至作业运行完毕后，他才能够根据输出的结果去分析作业的运行情况，确定下一次上机的任务。这样，就迫使用户改变了他们原来的工作方式，他们必须针对作业运行中可能出现的各种情况，事先规定好相应的措施，以便能够取得予计的效果。但是，由于系统工作时不允许用户插手干予他的作业，所以这种系统大大压缩了两次作业之间的交接时间，减少了手工操作次数，从而使系统有相对灵活的调度原则，便于实现整个计算机的工作流程自动化，充分发挥各种资源的利用率。

2）分时系统

分时系统有三个特点，即多路调制性，交互性和独占性。之所以要有这三个特点，是因为分时系统所要达到的目标，是使很多用户（多路调制性）能够同时与一台计算机进行“对话”(交互性)，但用户彼此之间却感觉不到别人也在使用计算机（独占性）。换言之，这种系统为每个用户提供的工作环境是一台可以交互会话的通用虚拟计算机。实现分时系统的基本思想如下：把计算机和很多终端设备联结起来，并令每台终端设备对应一个用户；于是，每个用户就可以通过终端设备向系统提出命令，请求完成某项工作；然后，系统分析从终端设

备发来的信息，或者完成用户提出的要求，或者查出用户的错误，并把运行结果再通过终端设备告诉相应的用户；以后，用户又根据系统提供的运行结果，向系统提出下一步请求，如修改程序、重新计算或执行其它工作等等；重复上述的交互会话过程，直至用户完成予计的工作为止。由于终端设备上的用户工作很慢，所以系统能够在较短的时间(几秒钟)内同时响应所有用户的要求，因而每个用户都感觉自己好象是独自使用计算机一样。分时系统的优点，就是在充分发挥机器效率的基础上，允许每个用户联机使用计算机，按照用户颇感自然的工作方式（交互会话式）去编写、调试、修改和运行自己的程序，从而能使用户在较短的时间内，实验、修改很多的设计思想，选出理想的解题方案，加快解题周期。其次，无论是本地用户或远地用户，只要和计算机连上一台终端设备，就可以简便地使用计算机。这将有利于计算机的普及推广以及扩大使用范围。第三，分时用户之间可以通过计算机的文件系统，彼此交流程序、信息和计算结果，便于多个用户协作完成共同关心的任务。

3）实时系统

实时系统主要是为联机实时任务服务的。实时系统有如下特点：

① 系统要对外部实时信号作出及时响应，响应的时间间隔要足以能够控制发出实时信号的那个环境。

② 系统的整体性强，实时系统所管理的联机设备和资源，要按一定的时间关系和逻辑关系，协调工作。

③ 实时系统没有分时系统那样强的交互式会话功能，通常不允许用户通过实时终端设备去编写新的程序或修改已有的程序。实时终端设备通常只是作为执行装置或询问装置使用的。

实时系统大部分是为特殊的实时控制任务设计的，如生产控制，票证予定等实时控制任务。这类实时控制任务对系统的可靠性和安全性要求很高，所以，系统的所有部分通常都是采用双工方式工作的。

上面，我们把操作系统大致分为三类，并讨论了各类系统的特点及其实现思想。但是，对一个具体的操作系统来说，它可能不属于其中任何一类，而是同时具备这三类或其中两类的特点，也可能以某类为主，兼有其它类的特点。不同类型的操作系统之间的差异，主要是由于系统所侧重的目标不同造成的。在批加工系统中，重点是放在机器的利用率和吞吐作业的能力上；在分时系统中，重点是放在交互作用和响应时间上；在实时系统中，重点是放在系统的完整性，响应时间和基本数据的保护上。

就实现操作系统的方法而言，大致可以概括为两种不同的方法，即多道程序法和程序交换法。

多道程序方法的基本思想，是在计算机的主存中同时存放几道彼此无关的程序，当正在运行的程序因某种原因工作不下去或规定给它的时间片用完时，系统就按照一定的原则去选择另一个可运行的程序，并将CPU控制转给它。当某道程序工作结束时，应从后备存贮器中挑选一个新的作业程序替代这个结束的作业程序。

另一种实现方法称为程序交换法。它的实现要点是在主存中只放一道可运行的作业程序，而其它的作业程序都放在后备存贮器中，当主存中的程序因某种原因工作不下去或规定给它的时间片用完时，系统就把主存中的程序交换到后备存贮器中，并从后备存贮器中把另一个可运行的作业程序交换进来，再将控制转给这个程序。实现时，应使程序交换工作和作业程序

的运行尽量重迭起来，以便压缩作业的转接时间，提高机器的使用效率。

具体实现一个操作系统时，可以根据实际情况和机器特点，选择上述两种方法之一加以实现。就一般的情况来说，结合采用这两种方法将会使系统的性能有显著地提高。

操作系统是一种大型复杂的系统，为了系统地研究、分析它的基本功能、组成部分、工作过程以及体系结构，我们通常要从不同的角度，采用不同的观点，去分析它的功能，剖析它的结构，研究它的实现方法。

下面，我们就分节介绍在分析设计操作系统时所用的各种观点。

第二节　操作系统的用户观点

这种观点主要是为了刻画操作系统的功能而引入的。从用户观点来看，配置了操作系统的计算机与原来真实的物理计算机是不相同的。因为，用户既不关心计算机的细节问题，也不关心操作系统的内部结构和实现方案，他们关心的只是系统提供的功能和系统服务的质量。所以，为了从功能角度为用户提供一个简单明了的操作系统，通常需要引入虚拟机概念去描述操作系统。

1　虚拟机概念

一个没有配置任何软件的计算机称之为裸机。因为没有软件协助用户解决他的问题，所以用户最不喜欢裸机这种工作环境。为了方便用户使用计算机，通常要为计算机配置各种软件去扩充机器的功能。抽象地说，每在原有计算机上增加一种软件，我们就可以认为，构造了一台比原有计算机功能更强的"计算机"，因为这种扩充后的计算机只是概念上的计算机，而不是真实的物理计算机，所以我们把它称为虚拟计算机（图 1.1）。

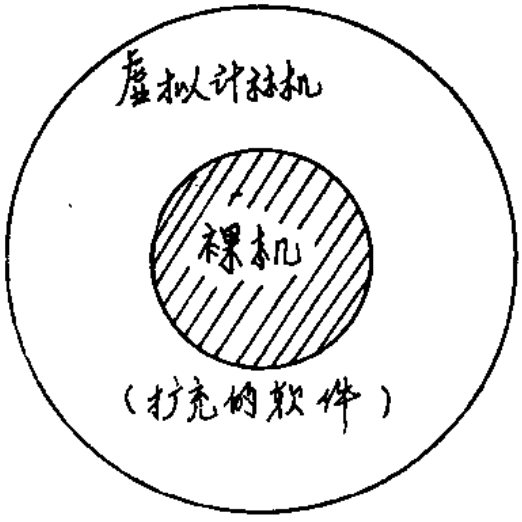

图 1.1　虚拟计算机

例如，在一个只有定点运算的计算机上，配置了浮点解释程序后，我们就可以认为，构造了一台可以执行浮点计算的虚拟计算机，

又如，在一台通用计算机上，如果我们给它配置了某种高级程序设计语言的编译程序，那末，就可以把它看作一台能够直接执行相应高级语言的虚拟机。

从上面这两个例子，我们不难看出：对于每一台虚拟机，我们除了需要配置一个完成相

应功能的软件外，还要为用户提供一种如何使用相应虚拟机的语言。如果不考虑虚拟机的性能和实现方法，那末，虚拟机的功能就完全由它的语言所决定。

从虚拟机的观点来看，操作系统当然也是一台虚拟机。下面，我们就介绍一下它的功能及其语言。

2 操作系统的功能和操作命令语言

操作系统对用户来说是一台虚拟机，是一台协助他们解决问题的装置。用户对这种装置的要求可概括为两个方面。一个方面是装置的性能，用户希望它能够简便灵活、安全可靠、经济有效地解决问题。另一个方面是装置的功能，用户希望操作系统为他提供如下的功能：

1）为用户创造一个适宜的工作环境(批加工的、分时的或实时的)，便于用户控制自己作业的运行。

2）为了简化用户编写、调试、修改和运行他的程序，操作系统应配置各种子系统(如：汇编程序、编译程序、编辑程序、装配程序和调试程序，等等)以及程序库(如：服务程序库和应用程序库，等等)，以便增加用户的解题能力。

3）为了简化输入输出操作，统一资源的分配管理，系统应提供广泛的数据管理操作。

用户是通过各种操作命令请求系统完成上述功能的。系统所提供的全部操作命令的总和称之为操作命令语言。操作命令语言是用户和系统的通讯手段和介面，按其使用方式来分，有三种不同的形式：

1）**作业命令语言**，脱机用户或批加工用户用这种语言编写作业说明书，组织作业的运行或完成某种操作。

2）**键盘命令语言**，这是为联机用户或交互工作用户通过控制台或终端设备，向系统提出请求而设计的命令语言。

3）**系统调用命令**，用户程序中可以直接使用这种命令，去调用系统提供的程序和子程序。

并非所有操作系统都要提供上述三类命令语言。例如，批加工系统就不一定要提供键盘命令语言；反之，分时系统也不一定要提供作业命令语言。

就操作命令的格式来看，虽然各种操作系统不尽相同，但作业命令和键盘命令一般都具有类似下面的格式：

〈命令〉〈参数表〉

其中，〈命令〉给出操作命令的名字，而〈参数表〉给出相应命令所需的全部参数或变元。一般地说，在操作命令语言中，通常都是以文件名作为参数或变元的。在较灵活的操作命令语言中，操作命令的前面可以加标号，后面可以加注解。加标号的目的是为了系统或其它命令根据作业的运行情况，按照指定的标号去引用相应命令；加注解的目的是为了用户在命令中加入必要的解说，以助自己或他人阅读之用。

因为系统调用命令要在用户程序中使用，所以这种命令的格式要随着写用户程序所用的语言之不同而有所不同。在用汇编语言写出的程序中，为了使操作命令和汇编指令协调一致，通常把系统调用命令表示成一组类似汇编指令的形式。但在提供了宏指令的汇编语言或高级程序设计语言中，亦可采用上面所谈的作业命令或键盘命令的格式。

虽然可以根据操作命令的功能，把系统提供的操作命令细分成很多的类，但是，归纳地说，可以把它们划分成两大类。一类是作业控制操作，另一类是数据管理操作。作业控制操

作的目的是组织作业的运行，控制作业的工作流程。这种操作命令有：控制作业步(如编译、编辑、调试、连结、装配和运行等)工作的操作，组织几个程序段并行工作的操作，以及并行程序段之间进行通讯和同步的操作。数据管理操作的目的是协助用户组织、保存和传输作业中的数据信息。这种操作命令包括有：资源(缓冲区、二级存贮器、输入输出设备等)的申请和释放操作，文件的读写和数据的加工操作等。

除了上面这些共同的操作命令外，作业命令语言和键盘命令语言还都有一些自己特有的操作命令。例如，在作业命令语言中，要有作业标识命令，它给出作业名、用户名、优先数、资源要求、运行时间等信息。为了使用户能在脱机工作环境下，根据作业运行情况，选择不同作业步工作，在作业命令语言中，还可以引入转移命令和条件转移命令。在键盘命令语言中，有时要提供与系统接通或切断的操作命令。有时为了对作业进行检查、修改，还要提供挂起或重新起动某作业或程序段运行的命令以及检查、修改数据内容和程序内容的命令。

如果把操作系统看作一台为用户定义的虚拟计算机，那末，操作命令语言就刻画了相应虚拟机的功能，并给出虚拟机所能执行的"指令"集合。于是，从用户所关心的功能角度来看，操作系统就是按下面两步循环工作的计算机(图 1.2)：

1）从作业说明书或键盘设备读进下一条命令S；

2）分析并执行命令S。

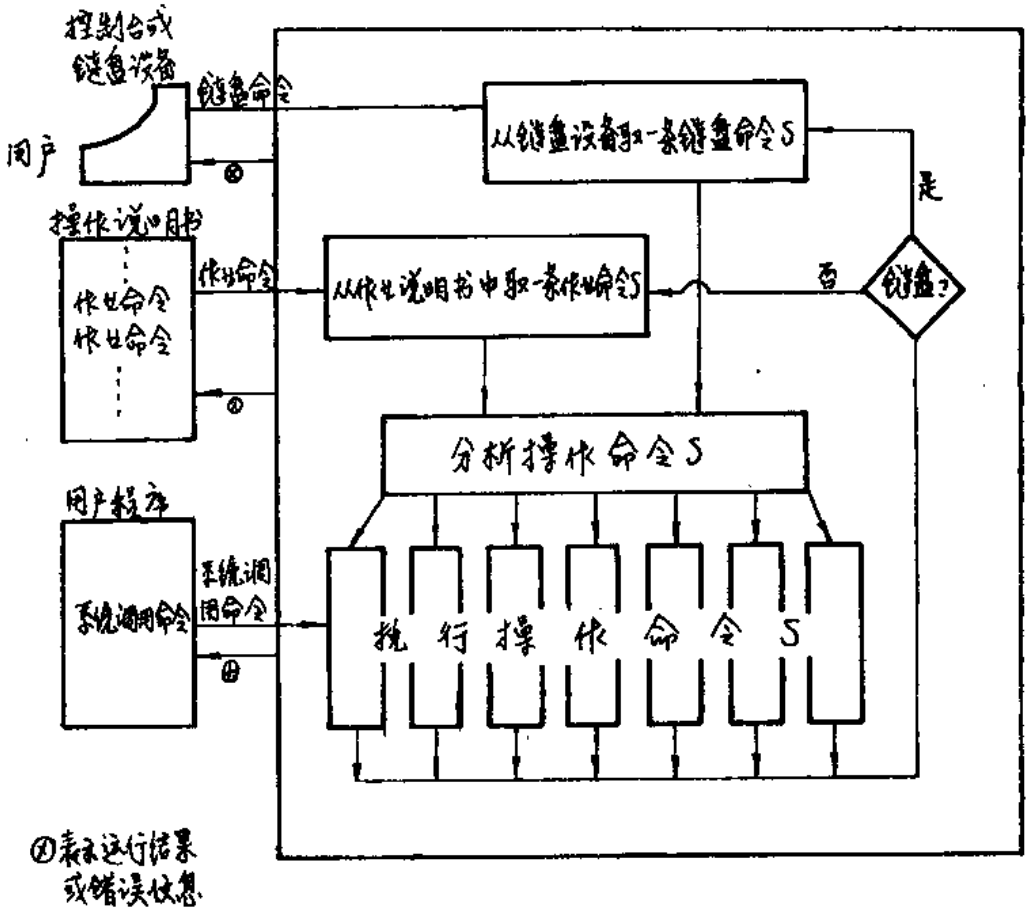

图 1.2 用户观点的操作系统工作流程示意图

当用户程序执行到系统调用命令时，则直接调用相应的系统程序；执行完毕时，返回到用户程序。

第三节　操作系统的资源管理观点

1．资源和资源管理

操作系统要管理很多的硬设备和软设备，这些硬软设备统称为资源。按其性质来分，可把资源归纳为四大类，即处理机、存贮器、外部设备以及信息(程序和数据)。这四类资源构成了操作系统本身和用户作业赖以活动的物质基础和工作环境。它们的使用方法和管理策略决定了整个操作系统的规模、类型、功能以及实现。所以，引入资源管理观点的重要作用之一，就在于我们能够使用这一观点去组织操作系统的有关内容。基于这一观点，我们可以把整个操作系统看成是一组资源管理程序所组成的。相应上述四类资源，可以把操作系统划分成处理机管理、存贮管理、设备管理和信息管理(即文件系统)这四大部分去加以分析和研究。

操作系统所管理的硬软设备愈来愈多，它们所管辖的范围也愈来愈大，这不仅促进了操作系统的发展，而且也产生了抽象研究"资源"的客观要求。尽管各种资源的性质极其不同，但是，从本质上看，它们除了具有"个性"之外，又都具有"共性"，这就促使人们去研究资源的统一概念，研究资源的使用方法和管理策略，以便寻求一种管理资源的普遍原则和系统方法。

当代操作系统的一个重要特点，就是内存中可以同时有几道作业处于可运行状态，而每道作业又往往划分成若干个彼此能够并行执行的程序段(通常称之为进程)。这些作业和进程就组成了系统中企图分配资源的分配对象。一般地说，作业和进程是按照"分配—使用—释放"这样的步骤去和资源发生联系的。作业所需的资源是在调度到这个作业时根据用户给出的信息进行分配的，并在作业运行完毕时释放所分到的全部资源。这种分配通常称为资源的静态分配。进程所需的资源是在进程运行中根据运行情况动态地分配、使用和释放的。这种分配通常称为资源的动态分配。依照资源的特点以及系统所采用的实现方法不同，每种资源的"分配""使用"和"释放"或者采用明显的调用命令加以指明，或者不用指明而由系统代为完成。

概括地说，研究资源管理的目的是：为用户提供一种简单、有效的使用资源的方法，充分发挥各种资源的利用率，防止资源分配中发生死锁现象。为此，对每种资源管理来说，要研究如下几方面的内容：

① 记住资源的使用状态。亦即，记住哪些资源未被使用，哪些资源已被使用，以及被谁使用等情况。

② 确定资源的分配原则和调度原则。亦即，根据系统的设计目标，确定一组原则，用以决定资源分配给谁？何时分配？分配多少？以及分配什么？等问题。

③ 根据②中所确定的原则以及用户的要求，执行资源分配。

④ 收回资源。当资源不再需要时，收回资源以便重新分配给其它作业或进程使用。

依据上述几点来看，一种较简单的资源管理方案，应包含如下的数据结构和程序。它们之间的联系如图 1.3 所示。

① 一张资源使用清单和一张作业或进程的排队表。资源使用清单记录了所有资源的使用情况，而作业或进程的排队表记录了所有等待分配此种资源的作业或进程。

② 为了便于检索、插入和取消上述两种数据结构中的登记项，资源管理中应分别提供

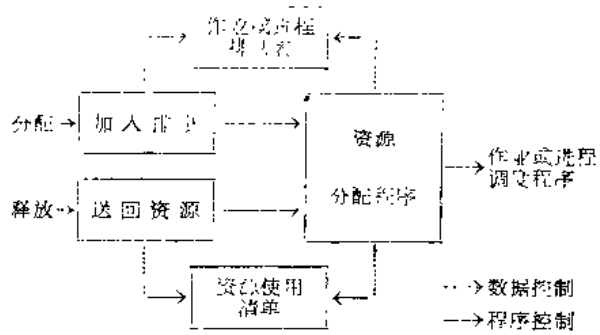

图 1.3 资源管理的程序模块和数据结构联系图

有关操作的程序。这里特别强调一下，当向排队表中插入作业或进程时，应把它们的状态改为"挂起"状态，表示等资源分配；反之，当从排队表中取消一个作业或进程时，应将其状态改为"就绪"状态，表示已分到资源，只等运行了。

③ 资源分配程序。它检索排队表和资源使用清单，并根据资源分配原则确定排队表中哪些作业或进程能分到所需的资源。对于能分到资源的作业或进程，则把它们从排队表中取下来(因而状态变为"就绪")，并从资源使用清单中取出它们所需的资源，分配给它们。然后，转到作业或进程的调度程序。

当正在运行的作业或进程执行分配资源命令时，则控制转到相应资源的加入排队模块。该模块把当前作业或进程的名字连同命令参数(如资源数量)合在一起造登记项，插入排队表中，等待分配资源。如果它们分到资源则处于就绪状态，否则处于挂起状态。

当正在运行的作业或进程执行释放资源命令时，则转到送回资源模块。它把释放的资源登入资源使用清单，然后在试着解挂等资源的作业或进程。如果解挂的作业或进程的优先数较高时，则调度程序将调度优先数最高的执行。否则还是调度原来发释放命令的进程。

2．处理机管理

处理机管理所关心的是处理机的分配问题。通常分为作业管理和进程管理两个阶段去实现处理机的分配。

作业管理中的主要组成部分是作业调度程序，其功能是按照一定的原则(如优先数、要求的资源，系统的均衡性)，从所有储备作业中选择一个即将投入运行的作业，并为它分配必要的资源(如主存空间，外部设备等)，建立一个用户作业进程(有的系统还要建立一些其它进程)，并把它们调入主存等待进程调度程序调度；当调度到用户作业进程时，它就依次解释执行用户指定的各个作业步(如输入、编译、装配和计算等)，每个作业步通常是由系统把它转化成若干个更小的进程来完成的；当作业完成或中途出现严重的错误时，则结束该作业，此时，用户作业进程将做一些善后处理工作，输出运行结果或错误信息，并根据作业运行情况，收集有关的统计资料，结算作业运行所需支付的费用，并分别把它们记入系统运行日记和会计文件中，然后通知作业调度程序此作业完成；最后，作业调度程序收回该作业的全部资源，注销该用户作业进程及其所有子进程，再去调度别的作业投入运行，重复上述过程，直至所有作业运行完毕为止。

从上面讨论中，不难看出：为了完成作业调度这一主要功能，作业管理还要具备另外一些功能。其中包括有：记住所有储备作业的状态；为选中的作业分配必要的资源；当作业结束后收回资源等功能。

进程管理的主要功能是把处理机分配给进程以及协调各进程之间的相互关系。它是由进程调度程序和运转控制程序这两个组成部分组成的。进程调度程序的功能是根据一定原则(如用户或系统给出的优先数，简单轮转等)，确定处理机分配给就绪状态进程中的哪一个进程、何时分配以及分配的时间长短等。运转控制程序的功能是记住进程的状态，并实现进程状态之间的转换。进程通常具有三种状态：运行状态、挂起状态和就绪状态。当进程要从运行状态变成挂起状态时，运转控制程序的工作是改变进程状态，保存进程的现场，并收回处理机以便分配给其它进程。当进程要从就绪状态变成运行状态时，除相应地改变状态外，应恢复进程的现场使之能够继续运行。

总括地说，作业调度又称高级调度或宏观调度，它的主要工作是确定某作业的进程可以作为候选进程(亦即有可能分到处理机的进程)；而进程调度称作低级调度或微观调度．它才最后确定哪个候选进程能得到处理机。有的系统在高级调度和低级调度之间又增加一级调度，称为中级调度。引入中级调度的目的在于减轻低级调度的负担，增加高级调度的灵活性。这是因为，我们可以把低级调度中的很多决策性的工作移到中级调度中去完成，从而加快低级调度的速度；另外，高级调度较难考虑作业运行中的情况，而中级调度程序可弥补这一缺点，增加调度的灵活性。

3．存贮管理

存贮管理所关心的问题是主存的分配问题和主存的扩充问题。存贮管理所要控制的物理装置是主存，有时也包括象磁盘、磁古等辅助存贮器。在这里，辅助存贮器只是为了补充主存容量之不足而引入的，它作为暂时保存信息之用，实际存取信息时应把相应信息调入主存中进行存取。存贮管理所要达到的目的，对用户来说，是要提供一个大容量的一级存贮器；对系统来说，是提高资源利用率，改善系统的整体性能。

存贮分配通常包括两个方面。一个方面是作业的存贮分配，另一个方面是进程的存贮分配。操作系统首先关心的是作业的存贮分配。这是因为，作业调度程序选择作业运行的首要条件，就是看一看主存中有没有足以保证作业运行的存贮空间。如果没有足够的空间，作业不能够运行，只好落选；如果有足够的空间，系统就以最有效、最经济的方式去为选中的作业分配所需空间。由于操作系统的重要特点之一是通常有几个作业同时在内存中处于可运行状态，所以存贮管理要解决如下几个问题：

① 存贮无关性。因为程序员无法予知存贮管理把他的作业分到主存中的什么地方，而且程序员也希望摆脱存贮地址、存贮大小等细节问题。为此，存贮管理应提供“再定位”能力，如“再定位装配程序”，“再定位硬件”、或“地址映象机构”等等。

② 存贮保护。因为主存中同时存放几个程序，所以，为了防止某个用户作业的程序干扰、破坏其它用户程序或系统程序，存贮管理必须保证：每个作业程序只能访问它自己存贮空间中的信息，不能存取任何其它信息。存贮保护必须由硬件提供措施，单纯由软件解决效率极差。硬件提供的存贮保护办法有：基址、限界寄存器方法和存贮键、锁方法。

③ 存贮扩充。主存空间是计算机资源中最为贫乏的资源，尤其当代操作系统要在主存中同时存放多个作业，这就使主存资源变得更加紧张。通常是用磁盘、磁古等辅助存贮器去扩充主存空间。硬件提供的办法是“虚拟存贮器”，而软件提供的办法有“交换、链接”和“复盖处理”等技术。

为了解决上述三个问题，操作系统提出并发展了多种存贮管理方案。其中包括有：分区

分配、再定位分区分配、页式分配、请页式分配以及段式分配，等等。这里，我们就不再一一介绍了。

存贮管理除了提供作业的存贮分配外，还要提供进程的存贮分配。用户进程工作时，除了自己所需的工作区外，还要为系统提供一定的表或工作区，以便套用系统命令完成所需的工作，如文件活动表(或称软通道)，进程控制块，等等。用户自己的工作区原则上是可以自己管理，但有的系统提供申请、释放工作存贮区的命令，以减轻用户管理存贮的麻烦。另外，系统进程工作时要频繁地使用各种表、链表、栈、排队、缓冲区等数据结构，存贮管理应提供有关操作和程序，以便管理、分配用户进程和系统进程所需的工作存贮区。

4. 设备管理

设备管理是操作系统中最庞杂、琐碎的部分。其原因是：① 这部分要涉及很多实际物理设备，它们品种繁多、用法各异；② 各种外部设备都能和主机并行工作，而且可被多个用户作业和进程所共享；③ 主机和外部设备以及各类外部设备之间的速度极不匹配，级差很大。基于这些原因，设备管理最为关心的问题是设备的无关性、设备分配、设备的传输控制以及设备性能的改善及其利用率的提高。

设备的无关性问题主要是通过文件系统解决的。从用户角度来看，各类外部设备均可统一在文件概念之下，设备的控制和传输是通过用户编写的文件命令和文件读写命令给出的。所有用户作业和用户进程发来的文件命令汇总到文件系统，文件系统再根据文件命令提出一系列I/O要求，并分发给不同的设备管理程序去完成。所以，设备管理是文件系统和实际物理设备之间的接口和介面。它所要解决的问题是上述其余三个问题。它的主要工作是：① 接收文件系统提出的I/O要求；② 把外部设备分配给各个I/O要求；③ 将I/O要求转换成I/O指令，启动设备进行传输，传输完毕后返回文件系统。

为了完成上述工作，设备管理应包括三个组成部分：I/O运转控制程序、I/O调度程序和I/O处理程序。

① I/O运转控制程序的主要功能是记住外部设备、控制器和通道的状态。这通常是通过相应的设备控制块、控制器控制块和通道控制块记录有关信息的。

② I/O调度程序的功能是根据一定的原则，确定把外部设备分配给哪一个I/O要求以及何时分配。I/O分配原则一方面取决于系统所要达到的目标(如吞吐能力、均衡性等)，另一方面取决于设备的特性。从资源分配角度来看，通常可把设备分成独占设备、共享设备以及虚拟设备这三种类型。独占设备如读纸带机、行印机。为了保证传输信息的连贯性，这类设备一经分配给某作业，那末就一直分定给该作业，直至作业完成或不再需要该设备为止。因为单个作业往往不能充分使用设备，而且又经常参与手工操作，所以这类设备的利用率很低。共享设备如磁盘、磁古等。这类设备的特点是允许多个用户同时交错地使用。因此，这类设备有相对灵活的调度原则、较高的设备利用率，但调度算法较为复杂。虚拟设备主要是为了改善独占设备的性能而引进的。其基本思想是，输入时，先把所有输入信息读入并复写到磁盘上，需要使用信息时再从磁盘读入信息；类似地，输出时，先把输出信息暂时有放在磁盘上，然后当设备空闲时，再把磁盘上的信息送到输出设备上输出。引入虚拟设备的好处是平衡了设备忙闲不均的工作要求，提高了主机和外部设备的并行工作能力，使独占设备转化成共享设备，使一台实际物理设备转化为多台虚拟设备。虚拟设备有较灵活的调度原则，这通常是采用假脱机处理技术实现的。

③ I/O处理程序的主要功能是把I/O要求翻译成相应的I/O指令，启动I/O设备进行信息传输；当I/O完成时，进行中断处理；改善设备性能，提高设备利用率。简言之，I/O处理程序的功能就是实现物理I/O传输。为了改善设备性能，提高设备利用率，通常所采用的办法是建立设备进程，采用灵活有效的调度原则，引入缓冲技术，提供假脱机处理等。

5．信息管理

在操作系统中，信息管理又称文件系统，它是由所有信息管理程序模块组成的。文件系统所关心的问题是为用户提供一种简便、统一的存取和管理信息的方法。

文件系统中的大部分工作是为了解决用户所需的信息结构及其操作与实际存贮介质结构及其I/O指令之间的差别而引入的。用户所希望的信息结构是按照简单的逻辑关系组织在一起的；他所希望的操作是一些只用名字就能存取所需信息的读写命令。然而，计算机只能使用各种I/O指令去存取相应介质上的信息，其信息结构又是按照存贮介质的各自特点组织的。如果没有文件系统，用户想要在各种存贮介质上传输信息，那是一项相当复杂、极为琐碎的工作。他必须把信息的简单逻辑结构映象到相应存贮介质上，确定所要传输的信息的位置，并把存取信息的操作翻译成相应设备的I/O指令。尤其是对那些可为多个用户所共享的磁盘、磁古、磁带等大容量存贮介质，用户不仅要记住自己信息的位置，而且要瞭解其它用户信息在介质上的分布情况。但是，这与用户信息的安全性和保密性又有所抵触。当考虑充分发挥主机和外部设备效率时，情况就更加复杂了。这通常要用到并行、中断、缓冲等较复杂的程序设计技术。总而言之，对某一个具体的用户来说，他是难于实现上述功能的。所以，在操作系统中，通常是提供文件系统协助用户去存取和管理信息。

文件系统的功能如下：

① 文件命令的解释和加工。每个文件系统都要为用户提供一组功能完善的文件命令，如建立、删除、打开、关闭、读写文件等命令，这些命令是通过文件命令加工程序实现的。

② 管理文件系统所用的资源。文件系统所使用的重要资源是文件目录和辅助存贮器。辅存是存放文件信息的场所，而目录则记录了相应文件的名字、位置、长度、属性等重要信息。

③ 为相应设备提出I/O要求。文件命令通常是转换成一系列I/O要求完成的。

相应上述功能，文件系统应包含如下组成部分：文件命令加工模块，目录管理模块、辅存管理模块以及设备管理接口模块。

概括地说，文件系统要用统一的观点、系统的方法为用户提供一组简便的、直接使用名字进行存取信息的操作，而文件系统所要解决的实质问题，就是按照充分发挥主机和设备效率的原则，把相应的文件操作转换成相应的I/O指令。转换中，使用的主要数据结构就是文件目录，根据它不仅可以把文件映象到实际存贮介质上，而且还可以确定信息所在的位置，进而把文件操作转换成相应的I/O指令。

第四节　操作系统的进程观点

1．为什么要引入进程

当代操作系统的一个重要特点是併发性。所谓併发性就是在操作系统自身程序或用户程序中，通常总是存在一些相对独立、但又能併发执行的程序段。这些程序段是一些"松散"联系的个体。它们在併发执行过程中，有很多时间是可以独立运行的；但它们又不是完全孤立、

彼此无关的，有时又要通过这样或那样的方式，彼此发生着相互依赖、相互制约的关系。概括地说，有两种相互制约方式。一种是**间接方式**，这是由于它们竞争相同资源产生的。得到资源的程序段可以继续运行，得不到资源的就要暂时挂起，等到有可利用的资源时再继续工作。一般地说，这种制约关系不仅可以发生在有逻辑关系的程序段之间，而且在那些逻辑上彼此毫无关系的程序段间，有时也会发生这种制约关系。在这种情况下，通常要由基础操作系统协调实现**并发控制**。另一种是**直接方式**，这通常是在那些逻辑上有一些关系的程序段之间发生的。一般是由于各程序段要求共享信息产生的。例如，要求信息通讯或同步时，就会发生这种制约关系。这时可以在程序段中使用系统提供的同步操作或信息通讯操作，调节各程序段之间的并发控制。

正是因为在这些可以并发执行的程序段之间，存在这样或那样的相互制约关系，所以每个程序段不能与外界隔绝，不能随心所欲地在处理机上运行。它不仅要受其它程序段的活动所制约，而且还要动态地依赖系统资源的分配情况。因此，每个可以并发执行的程序段，就会因外界条件的不同而处于不同的状态。有时处于可运行状态；有时就要因为等待某种资源或其它程序段的信号不能运行，而处于挂起状态。这样，对于这些可以并发执行的程序段，只用"程序"这一概念就不敷使用了。使用程序这个概念只能是简单、孤立、静止地研究分析它们，而不能深刻地揭示它们的活动联系及其状态变化。因此，我们必须从变化的角度，动态地分析研究这些可以并发执行的程序段。为此，在操作系统中就要引入"进程"的概念。

2．进程的状态变化和相互制约关系

进程有时又称作任务或活动。所谓进程就是每个处于并发执行中的程序段。为了在计算机中刻画这种动态变化的进程，通常把进程表示为两部分，一部分是程序，另一部分称做进程控制块 PCB(有时也称作任务控制块 TCB)。程序部分描述了进程本身所要完成的功能；而进程控制块则包含了进程的有关信息，它们表达了进程在当前时刻的状态以及它与其它进程和资源的关系。进程控制块所包括的信息有：进程的名字、基本状态、所需的资源和已分配的资源、调度信息、通讯信息、与其它进程控制块的连接字以及会计信息，等等。因此，每个进程可图示为

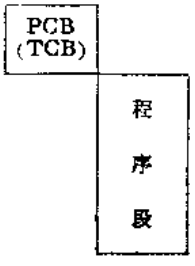

图 1.4 进程示意图

引入进程概念之后，我们就可以从变化的观点，动态地研究它们的状态变化和相互制约关系了。下面，我们先以一个简单的模型为例，讨论进程自身的状态变化；然后，再讨论进程之间的相互制约的形式；最后，讨论相互制约的实现方法。

2.1 进程的状态变化和状态变化图*

就最简单的模型来看，进程可以处于下面三种状态之一：

*) 注意：这里所谈及的"状态"的含义与前面不同。前面的含义较广，这里的含义较狭。

运行状态，该进程已分配到处理机，它的程序正在运行；

挂起状态，进程等某事件(例如，等输入输出完成)，此时，即使分配给它处理机，它也不能够运行；

就绪状态，进程已具备"运行条件"，但因处理机个数少于进程个数，所以该进程不能运行，而必须等待分配处理机。

进程的各个状态可依据一定的原因和条件发生变化。例如，在上述模型中，三种状态的变化可如图 1.5 所示。

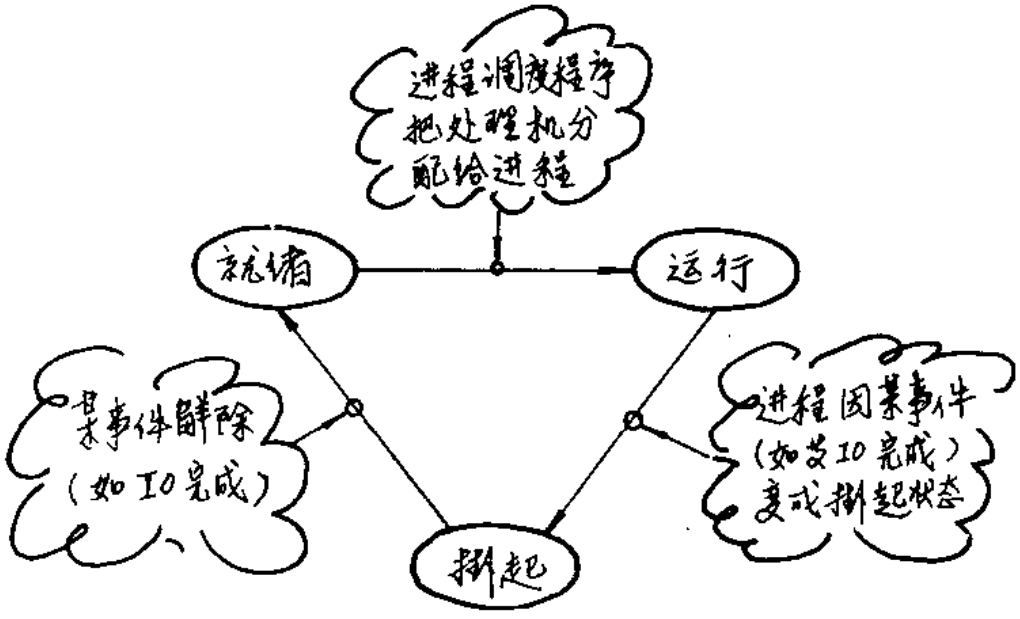

图 1.5 进程状态变化图

运行状态的进程可能因某事件（如，等输入输出完成）变成挂起状态；当相应事件解除(如，输入输出完成)时，这个进程又变成就绪状态；就绪状态的进程，当调度程序把处理机分配给它时，再次变回到运行状态。

为了在计算机中描述进程的状态变化，只要由有关程序修改相应进程的 PCB 内容(状态字、调度信息，等等)，就能动态地表达进程自身的状态变化以及它与外界环境的联系。一般地说，进程控制块的内容可由进程本身、其它进程、资源操作、中断程序以及调度程序进行修改和引用。对操作系统来说，所有的进程控制块将构成协调并发执行、维护系统工作的依据。

2.2 进程之间的相互制约形式

如前所述，进程的状态是依据一定的原因和条件发生变化的。然而，这些原因和条件，归根结蒂，又是由于进程之间的相互制约关系引起的。进程之间的相互制约关系或则是直接方式的，其图式为"进程—进程"；或则是间接方式的，其图式为"进程—资源—进程"。

① "进程—进程"关系

这种关系一般有两种表现形式。第一种是某进程 A 强制改变另一进程 B 的状态。这种情况通常是通过系统提供的改变状态命令实现的。例如，进程 A 执行改变状态命令

CHANG(B, S)

时，CHANG 命令的加工程序找出 B 的进程控制块，并将其中的状态字改成状态 S，然后再做一些相应的工作即可。第二种表现形式是进程之间进行信息通迅。这种情况一般是通过系统提供的信息通讯命令（如，SEND，REC）实现的。例如，如果进程 A 执行接收命令 REC (B, M)，准备在单元 M 中接收 B 之信息。此时，如果 B 尚未发送信息(M 为空)，则 REC 的加工程序应把进程 A 改为挂起状态，挂起原因是"等 B 向 M 中送信息"。如果 B 已发送了信息(M 非空)，则依系统实现方法不同，可返回调用者或转到调度程序重新进行调度。另一方面，当进程 B 执行发送命令 SEND(A, M, I)时，SEND 加工程序将信息 I 送入 M，然后查看 A 是否是因"等 B 向 M 中送信息"而挂起。如果是，则将 A 解挂，亦即，将 A 变成就绪状态。

② "进程—资源—进程"关系

这种制约关系通常是由基础操作系统中的各种资源管理程序协调控制的。当进程 A 向系统要求资源 R，但得不到满足时，则资源的分配-收回程序就把 A 挂起，挂起原因是"等资源 R"；反之，当另一个进程 B 释放资源 R 时，则资源分配—收回程序可把"等资源 R"的进程解挂。

2.3 进程之间相互制约关系的实现方法

为了协调进程之间的并发执行，每个操作系统都要根据自己的情况提供某种管理和控制进程状态变化的方法。这里，我们介绍两种常用的实现方法。一种是较简单的实现方法，另一种是应用较普遍的实现方法。

① 较简单的实现方法

这种方法的基本要点如下：在进程控制块 PCB 中开辟一栏名为"挂起原因"的信息区，用于记录相应进程被挂起的原因；另外，提供两条系统广义操作，即挂起操作 PEND 和解挂操作 UNPEND，两者的参数均为"挂起原因"。

当某段系统程序（如，资源 R 的分配-收回程序）需要把当前运行的进程挂起(如，没有资源 R）时，则执行 PEND (〈挂起原因〉)。此时，PEND 加工程序保存当前运行进程的现场，将状态字改为挂起状态，并填入〈挂起原因〉，然后，再将控制转给进程调度程序。

反之，当某段系统程序有可能解挂其它进程（如，有了资源 R）时，应当执行命令UNPEND (〈挂起原因〉)。此时，UNPEND 程序就检查全部被挂起的进程，如果某进程的"挂起原因"和解挂命令中的〈挂起原因〉相同时，则把相应进程解挂，亦即把它的状态改为就绪状态，然后，返回调用 UNPEND 命令的程序，继续工作。

② 应用较普遍的实现方法

这种方法是 Dijkstra 提出的，它的基本思想是引入两个以信号量作为参数的操作，称为 P、V 操作。所谓信号量 Sem 是一个由 S 和 Q 组成的二元组，即

$$Sem = (S, Q)$$

其中，S 是一个具有非负初值的整变量；而 Q 是一个初始状态为空的排队站。设 Sem 是一个信号量，则 P、V 操作可定义如下*)：

① P(Sem) S 值减 1。如果减一结果为负，则挂起当前运行的进程，并把它的 PCB 登

*) 注意，在P、V 操作定义中，"挂起某进程"的含义系指：把该进程的现场保存到相应进程的 PCB 中，并把其中的状态字改为挂起状态；"解挂某进程"的含义是指：把相应进程的 PCB 中的状态字改为就绪状态。