01 操作系统引论

1.1 目标、作用与发展动力

> *操作系统的目标与应用环境有关。*

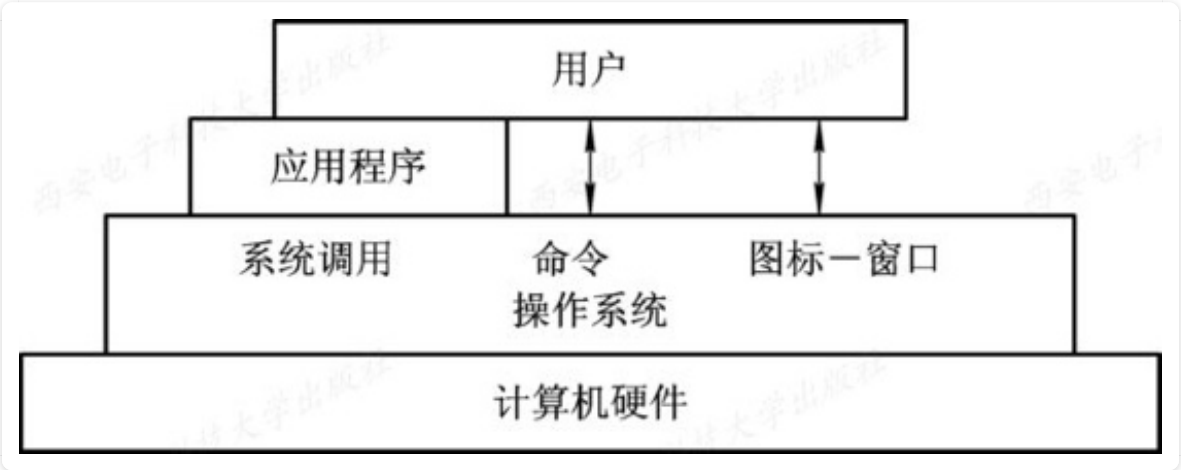
人机交互性：查询系统……
实时性：工业控制、武器控制以及多媒体环境……
方便性：微机……

1.1.1 目标

- 方便性
- 有效性：提高系统的资源利用率和吞吐量
- 可扩充性：无结构–>模块化结构–>层次化结构–>微内核结构（便于添加新功能和模块）
- 开放性：遵循开放系统互连OSI国际标准

1.1.2 作用

1.OS作为用户与计算机硬件系统之间的接口



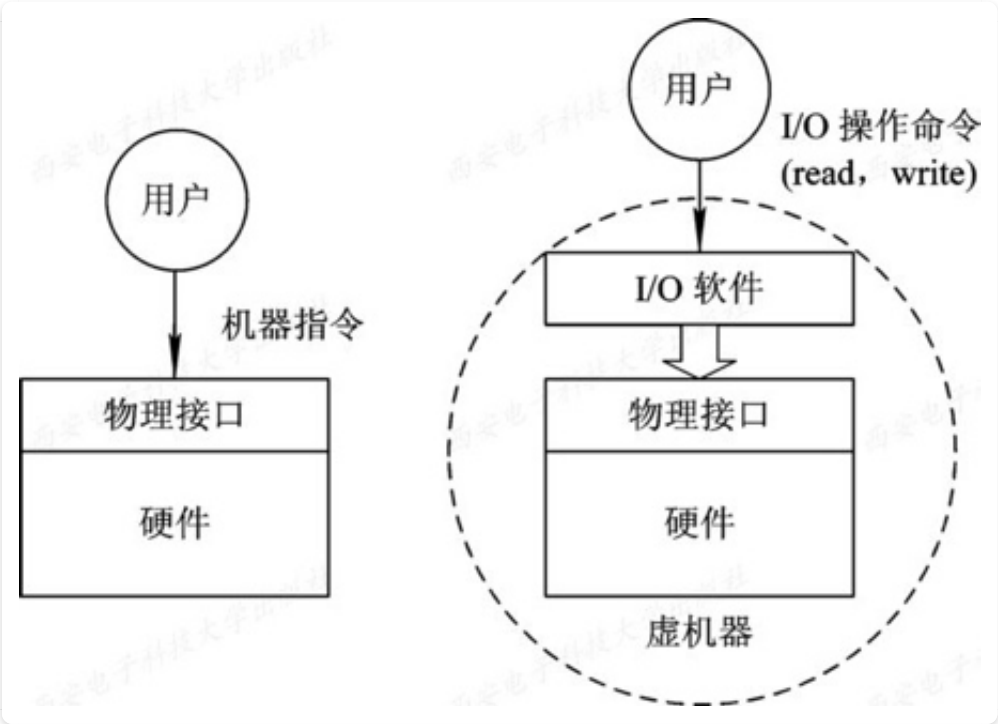
用户通过三种方式使用计算机：系统调用、命令、图标——窗口

2.OS作为计算机系统资源的管理者

- 处理机：用于分配和控制处理机
- 存储器：负责内存的分配与回收
- I/O设备：负责I/O设备的分配(回收)与操纵
- 文件(数据和程序)：用于实现对文件的存取、共享和保护

3.OS实现了对计算机资源的抽象

对于一台完全无软件的计算机系统(即裸机)，由于它向用户提供的仅是硬件接口(物理接口)，因此，用户必须对物理接口的实现细节有充分的了解，这就致使该物理机器难于广泛使用。为了方便用户使用I/O设备，人们在裸机上覆盖上一层I/O设备管理软件。



如图所示,由它来实现对I/O设备操作的细节，并向上将I/O设备**抽象为一组数据结构以及一组I/O操作命令**，如read和write命令，这样用户即可利用这些数据结构及操作命令来进行数据输入或输出，而无需关心I/O是如何具体实现的。通常把覆盖了上述软件的机器称为 扩充机器 或 虚机器 。它向用户提供了一个对硬件操作的抽象模型。用户可利用该模型提供的接口使用计算机，无需了解物理接口实现的细节，从而使用户更容易地使用计算机硬件资源。亦即，I / O设备管理软件实现了对计算机硬件换 作的第一个层次的抽象。
同理，为了方便用户使用文件系统，又可在第一层软件(I / O管理软件)上再覆盖一层用于文件管理的软件，由它来实现对文件操作的细节，并向上层提供一组实现对文件进行存 取操作的数据结构及命令。这样，用户可利用该软件提供的数据结构及命令对文件进行存取。此时用户所看到的是一台功能更强、使用

更方便的虚机器。亦即，文件管理软件实现了对硬件资源操作的第二个层次的抽象。
依此类推，如果在文件管理软件上再覆盖一层面向用户的窗口软件，则用户便可在窗口环境下方便地使用一机，从而形成一台功能更强的虑机器。
由此可知，os是铺设在计算机硬件上的多层软件的集合，它们不仅增强了系统的功能，还隐藏了对硬件操作的具体细节，实现了对计算机硬件操作的多个层次的抽象模型。值得说明的是，不仅可在底层对一个硬件资源加以抽象，还可以在高层对该资源底层已抽象的模型再次进行抽象，成为更高层的抽象模型。随着抽象层次的提高，抽象接口所提供的功能就越强，用户使用起来也越方便。

1.1.3 推动操作系统发展的主要动力

- 不断提高计算机资源利用率
- 方便用户
- 器件的不断更新换代
- 计算机体系结构的不断发展
- 不断提出新的应用需求

1.2 发展过程

> *50年代中期，出现了第一个简单的批处理OS；*
>
> *60年代中期，开发出多道程序批处理系统；不久又推出分时OS，与此同时，用于工业和武器控制的实时OS也相继问世；*
>
> *70到90年代，是VLSI和计算机体系结构大发展的年代，导致了微型机、多处理机和计算机网络的诞生和发展，与此相应地，也相继开发出了微机OS、多处理机OS和网络OS，并得到极为迅猛的发展。*

1.2.1 未配置操作系统的计算机系统

1.人工操作方式

> *早期的操作方式是由程序员将事先已穿孔的纸带(或卡片)，装入纸带输入机(或卡片输入机)，再启动它们将纸带(或卡片)上的程序和数*

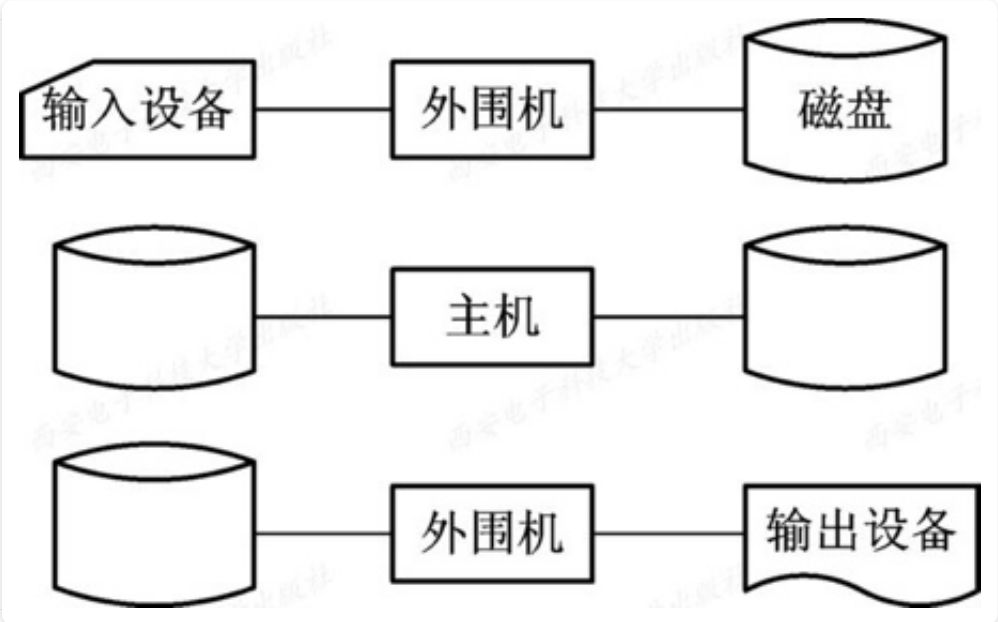
> *据输入计算机，然后启动计算机运行。仅当程序运行完毕并取走计算结果后，才允许下一个用户上机。*

缺点：

- 用户独占全机：一台计算机的全部资源由上机用户所独占。
- CPU等待人工操作：当用户进行装带(卡)、卸带(卡)等人工操作时，CPU及内存等资源是空闲的。

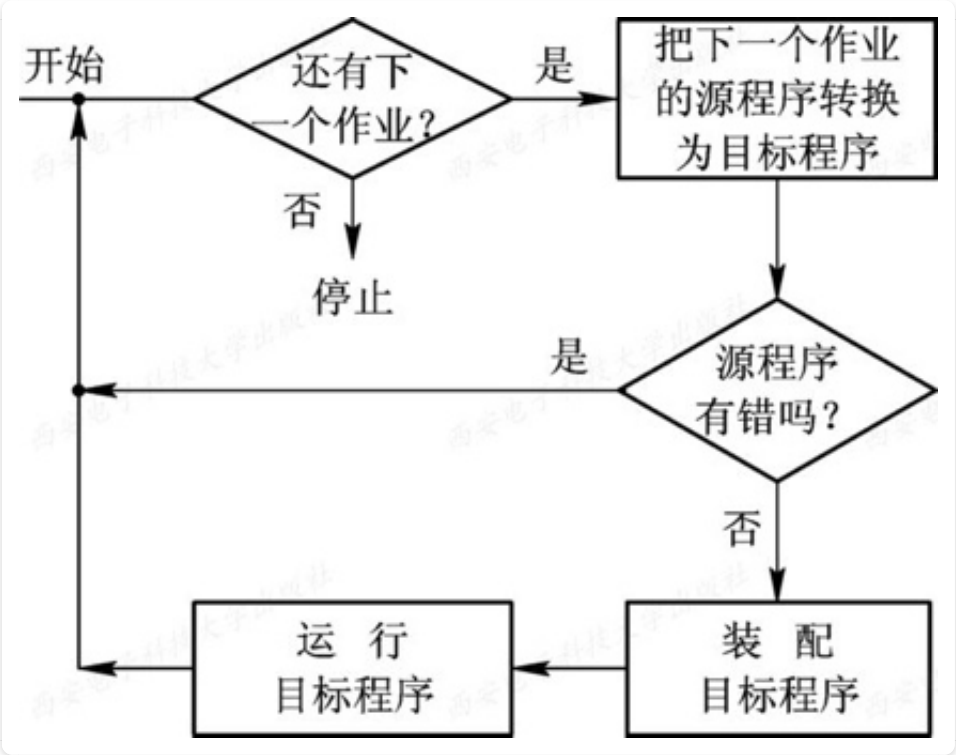
2.脱机输入/输出(Off-Line I/O)方式

> *为了解决人机矛盾及CPU和I/O设备之间速度不匹配的矛盾，20世纪50年代末出现了脱机I/O技术。*
> *该技术是事先将装有用户程序和数据的纸带装入纸带输入机，在一台外围机的控制下，把纸带(卡片)上的数据(程序)输入到磁带上。当CPU需要这些程序和数据时，再从磁带上高速地调入内存。*



1.2.2 单道批处理系统(Simple Batch Processing System)

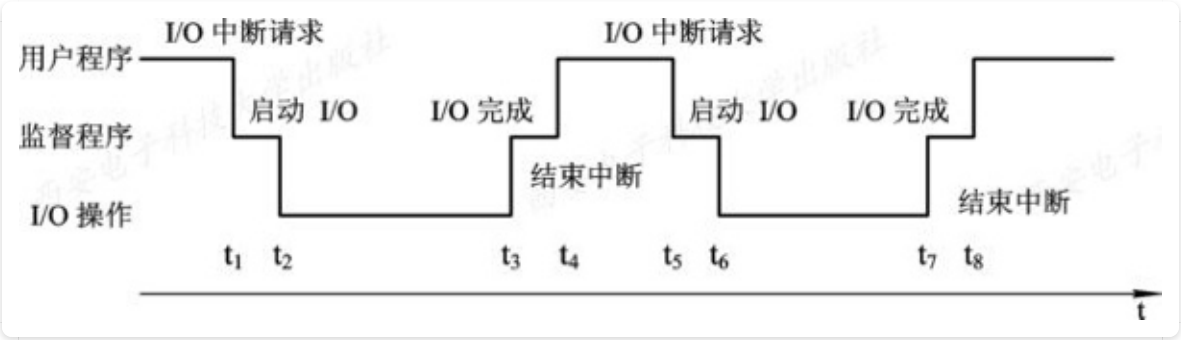
> *为实现对作业的连续处理，需要先把一批作业以脱机方式输入到磁带上，并在系统中配上监督程序(Monitor)，在它的控制下，使这批作业能一个接一个地连续处理。*



单道批处理系统最主要的缺点：系统中的资源得不到充分的利用。

因为在内存中仅有一道程序，每逢该程序在运行中发出I/O请求后，CPU便处于等待状态，必须在其I/O完成后才继续运行。又因I/O设备的低速性，更使CPU的利用率显著降低。

单道程序的运行情况：



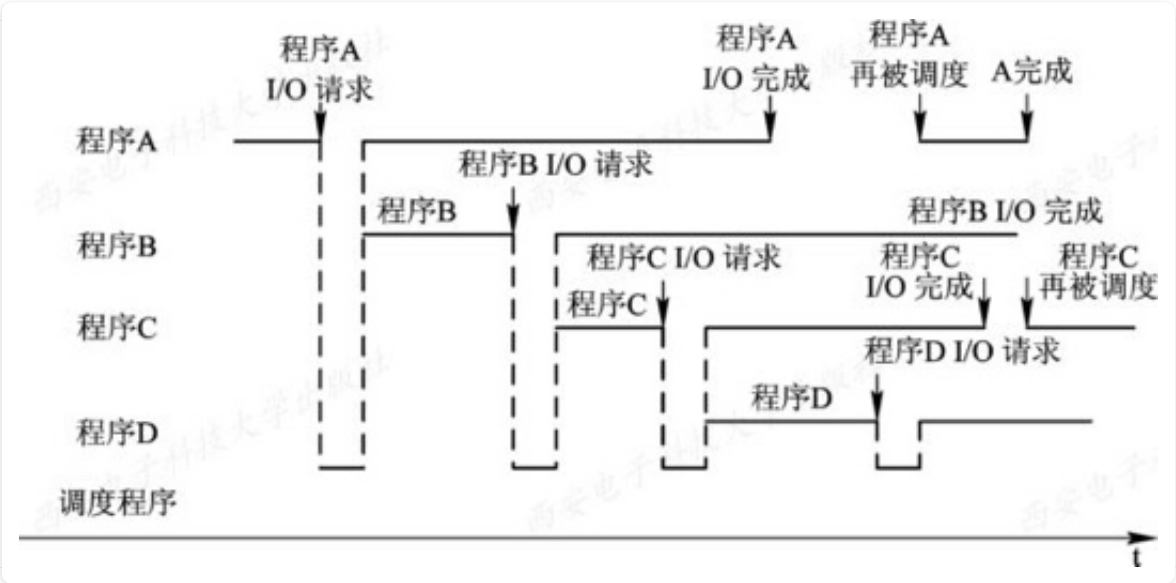
可以看出：在t2～t3、t6～t7时间间隔内CPU空闲。

1.2.3 多道批处理系统(Multiprogrammed Batch Processing System)

> *20世纪60年代中期，IBM公司生产了第一台小规模集成电路计算机旧IBM360（第三代计算机系统）。由于它较之于晶体管计算机无论在体积、功耗、速度和可靠性上都有了显著的改善，因而获得了*

> *极大的成功。IBM公司为该机开发的OS/360操作系统是第一个能运行多道程序的批处理系统。*
> *为了进一步提高资源的利用率和系统吞吐量，在20世纪60年代中期引入了多道程序设计技术，由此形成了多道批处理系统。*

四道程序时的运行情况：



优点

- 资源利用率高
  - 引入多道批处理能使多道程序交替运行，以保持CPU处于忙碌状态
  - 在内存中装入多道程序可提高内存的利用率；此外还可以提高I/O设备的利用率。
- 系统吞吐量大
  - CPU和其它资源保持“忙碌”状态；
  - 仅当作业完成时或运行不下去时才进行切换，系统开销小。

缺点

- 平均周转时间长：由于作业要排队依次进行处理，因而作业的周转时间较长，通常需几个小时，甚至几天。
- 无交互能力：用户一旦把作业提交给系统后，直至作业完成，用户都不能与自己的作业进行交互，修改和调试程序极不方便。

多道批处理系统需要解决的问题：

- 处理机争用问题：既要能满足各道程序运行的需要，又要能提高处理机的利用率。
- 内存分配和保护问题：系统应能为每道程序分配必要的内存空间，使它

们“各得其所”，且不会因某道程序出现异常情况而破坏其它程序。
- I/O设备分配问题：系统应采取适当的策略来分配系统中的I/O设备，以达到既能方便用户对设备的使用，又能提高设备利用率的目的。
- 文件的组织和管理问题：系统应能有效地组织存放在系统中的大量的程序和数据，使它们既便于用户使用，又能保证数据的安全性。
- 作业管理问题：系统中存在着各种作业(应用程序)，系统应能对系统中所有的作业进行合理的组织，以满足这些作业用户的不同要求。
- 用户与系统的接口问题：为使用户能方便的使用操作系统，OS还应提供用户与OS之间的接口。

1.2.4 分时系统(Time Sharing System)

1.引入

> *推动多道批处理系统形成和发展的主要动力是提高资源利用率和系统吞吐量，而推动分时系统形成和发展的主要动力，则是为了满足用户对人机交互的需求，由此形成了一种新型OS。*

分时系统是指在一台主机上连接了多个配有显示器和键盘的终端并出此所组成的系统，该系统允许多个用户同时通过自己的终端，以交互方式使用计算机，共享主机中的资源。

用户的需求具体表现在：

- 人机交互
- 共享主机

2.分时系统实现中的关键问题

> *在多道批处理系统中，用户无法与自己的作业进行交互的主要原因是：作业都先驻留在外存上，即使以后被调入内存，也要经过较长时间的等待后方能运行，用户无法与自己的作业进行交互。*

- 及时接收：给系统配置多路卡，给终端配置一个缓冲区
- 及时处理
    - 作业直接进入内存
    - 采用轮转运行方式：引入作业片

3.特征

- 多路性
- 独立性
- 及时性
- 交互性

1.2.5 实时系统(Real Time System)

1.实时系统的类型

- 工业(武器)控制系统
- 信息查询系统
- 多媒体系统
- 嵌入式系统

2.实时任务的类型

- 周期性实时任务和非周期性实时任务 非周期性实时任务必须联系一个截止时间（Deadline），进一步分为开始截止时间和完成截止时间。
- 硬实时任务（Hard Real-time Task）和软实时任务（Soft Real-time Task）：对截止时间是否严格。

3.实时系统与分时系统特征的比较

- 多路性
- 独立性
- 及时性
- 交互性
- 可靠性：实时系统要求高度可靠

1.2.6 微机操作系统的发展

1.单用户单任务操作系统

- CP/M（8位）
- MS-DOS（16位）

2.单用户多任务操作系统

> *只允许一个用户上机，但允许用户把程序分为若干个任务，使它们**并发**执行，从而有效地改善了系统的性能。*

3.多用户多任务操作系统

> *允许多个用户通过各自的终端，使用同一台机器，共享主机系统中的各种资源，而每个用户程序又可进一步分为几个任务，使它们能并发执行，从而可进一步提高资源利用率和系统吞吐量。在大、中和小型机中所配置的大多是多用户多任务操作系统，而在32位微机上，也有不少配置的是多用户多任务操作系统，其中最有代表性的是UNIX OS。*

1.3 基本特性

> *批处理系统有着高的资源利用率和系统吞吐量；分时系统能获得及时响应；实时系统具有实时特征*
> *除此之外，它们还共同具有并发、共享、虚拟和异步四个基本特征。*

1.3.1 并发(Concurrence)

使得OS能有效地提高系统中的资源利用率，增加系统的吞吐量。

1.并行与并发

> 并行性是指两个或多个事件在同一时刻发生。
> 并发性是指两个或多个事件在同一时间间隔内发生。

2.引入进程

> *在一个未引入进程的系统中，在属于同一个应用程序的计算程序和I/O程序之间只能是顺序执行，即只有在计算程序执行告一段落后，才允许I/O程序执行；反之，在程序执行I/O操作时，计算程序也不能执行。*

但在为计算程序和I/O程序分别建立一个进程(Process)后，这两个进程便可并发执行。

若对内存中的多个程序都分别建立一个进程，它们就可以并发执行，这样便能极大地提高系统资源的利用率，增加系统的吞吐量。

1.3.2 共享(Sharing)

> *一般情况下的共享与操作系统环境下的共享其含义并不完全相同。*

1.互斥共享方式

> *系统中的某些资源，如打印机、磁带机等，虽然可以提供给多个进程(线程)使用，但应规定在一段时间内，只允许一个进程访问该资源。为此，在系统中应建立一种机制，以保证多个进程对这类资源的互斥访问。*

2.同时访问方式

系统中还有另一类资源，允许在一段时间内由多个进程“同时”对它们进行访问。这里所谓的“同时”，在单处理机环境下是宏观意义上的，而在微观上，这些进程对该资源的访问是交替进行的。典型的可供多个进程“同时”访问的资源是磁盘设备。一些用重入码编写的文件也可以被“同时”共享，即允许若干个用户同时访问该文件。

1.3.3 虚拟(Virtual)

1.时分复用技术

- 虚拟处理机技术
- 虚拟设备技术

2.空分复用技术

在计算机中也把空分复用技术用于对存储空间的管理，用以提高存储空间的利用率。

本质上是实现内存的分时复用。

> *虚拟的实现，如果是采用分时复用的方法，即对某一物理设备进行分时使用，设N是某物理设备所对应的虚拟的逻辑设备数，则每台虚拟设备的平均速度必然 等于或低于物理设备速度的 1/N。类似地，如果是利用空分复用方法来实现虚拟，此时一台虚拟设备平均占用空间必然也等于或低于物理设备所拥有空间的1/N。*

1.3.4 异步(Asynchronism)

在多道程序环境下，系统允许多个进程并发执行。在单处理机环境下，由于系统中只有一台处理机，因而每次只允许一个进程执行，其余进程只能等待。当正在执行的进程提出某种资源要求时，如打印请求，而此时打印机正在为其它进程打印，由于打印机属于临界资源，因此正在执行的进程必须等待，并释放出处理机，直到打印机空闲，并再次获得处理机时，该进程方能继续执行。可见，由于资源等因素的限制，使进程的执行通常都不可能“一气呵成”，而是以“停停走走”的方式运行。

1.4 操作系统的主要功能

1.4.1 处理机管理

- 进程控制
- 进程同步
- 进程通信
- 调度
    - 作业调度
    - 进程调度

1.4.2 存储器管理

1.内存分配

- 为每道程序分配内存空间，使它们“各得其所”
- 提高存储器的利用率，尽量减少不可用的内存空间(碎片)。
- 允许正在运行的程序申请附加的内存空间，以适应程序和数据动态增长的需要。

OS在实现内存分配时，可采取静态和动态两种方式：

- 静态分配方式 每个作业的内存空间是在作业装入时确定的，在作业装入后的整个运行期间不允许该作业再申请新的内存空间，也不允许作业在内存中“移动”。
- 动态分配方式： 每个作业所要求的基本内存空间虽然也是在装入时确定的，但允许作业在运行过程中继续申请新的附加内存空间，以适应程序和数据的动态增长，也允许作业在内存中“移动”。

2.内存保护

- 确保每道用户程序都仅在自己的内存空间内运行，彼此互不干扰
- 绝不允许用户程序访问操作系统的程序和数据，也不允许用户程序转移到非共享的其它用户程序中去执行。

3.地址映射

在多道程序环境下，由于每道程序经编译和链接后所形成的可装入程序其地址都是从0开始的，但不可能将它们从“0”地址(物理)开始装入内存，致使(各程序段的)地址空间内的逻辑地址与其在内存空间中的物理地址并不相一致。为保证程序能正确运行，存储器管理必须提供地址映射功能，即能够将地址空间中的逻辑地址转换为内存空间中与之对应的物理地址。该功能应在硬件的支持下完成。

4.内存扩充

> *内存扩充并非是从物理上去扩大内存的容量，而是借助于虚拟存储技术，从逻辑上扩充内存容量，使用户所感觉到的内存容量比实际内存容量大得多，以便让更多的用户程序能并发运行。这样既满足了用户的需要，又改善了系统的性能。*

为了能在逻辑上扩充内存，系统必须设置内存扩充机制(包含少量的硬件)，用于实现下述各功能：

- 请求调入功能
- 置换功能

1.4.3 设备管理

- 完成用户进程提出的I/O请求，为用户进程分配所需的I/O设备，并完成指定的I/O操作。

- 提高CPU和I/O设备的利用率，提高I/O速度，方便用户使用I/O设备。

为实现上述任务，设备管理应具有:

- 缓冲管理
- 设备分配
- 设备处理

1.4.4 文件管理

1.文件存储空间的管理

2.目录管理

3.文件的读/写管理和保护

- 文件的读/写管理
- 文件保护

1.4.5 操作系统与用户之间的接口

1.用户接口

- 联机用户接口
- 脱机用户接口
- 图形用户接口

2.程序接口

程序接口是为用户程序在执行中访问系统资源而设置的，是用户程序取得操作系统服务的唯一途径。它是由一组系统调用组成的，每一个系统调用都是一个能完成特定功能的子程序。每当应用程序要求OS提供某种服务(功能)时，便调用具有相应功能的系统调用(子程序)。早期的系统调用都是用汇编语言提供的，只有在用汇编语言书写的程序中才能直接使用系统调用。

1.4.6 现代操作系统的新功能

> *现代操作系统是在传统操作系统基础上发展起来的，它除了具有传统操作系统的功能外，还增加了面向安全、面向网络和面向多媒体等功能。*

1.系统安全

- 认证技术
- 密码技术
- 访问控制技术
- 反病毒技术

2.网络的功能和服务

- 网络通信
- 资源管理
- 应用互操作

3.支持多媒体

- 接纳控制功能
- 实时调度
- 多媒体文件的存储

1.5 OS结构设计

> *早期OS的规模很小，如只有几十KB，完全可以由一个人以手工方式，用几个月的时间编制出来。*
>
> *此时，编制程序基本上是一种技巧，OS是否是有结构的并不那么重要，重要的是程序员的程序设计技巧。*
>
> *但随着OS规模的愈来愈大，其所具有的代码也愈来愈多，往往需要由数十人或数百人甚至更多的人参与，分工合作，共同来完成操作系统的设计。这意味着，应采用工程化的开发方法对大型软件进行开发。由此产生了“软件工程学”。*

1.5.1 传统操作系统结构

1.无结构操作系统

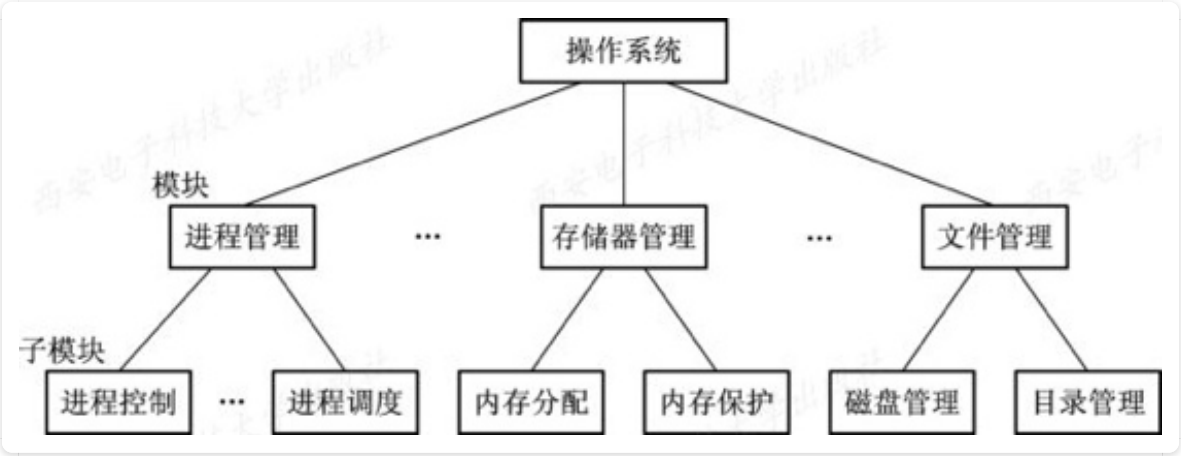
> *在早期开发操作系统时，设计者只是把他的注意力放在功能的实现和获得高的效率上，缺乏首尾一致的设计思想。*
> *此时的OS是为数众多的一组过程的集合，每个过程可以任意地相互调用其它过程，致使操作系统内部既复杂又混乱，因此，这种OS是无结构的，也有人把它称为整体系统结构。*

2.模块化结构OS

(1)模块化程序设计技术的基本概念

> *模块化程序设计技术是20世纪60年代出现的一种结构化程序设计技术。该技术基于“分解”和“模块化”的原则来控制大型软件的复杂度。为使OS具有较清晰的结构，OS不再是由众多的过程直接构成的，而是按其功能精心地划分为若干个具有一定独立性和大小的模块。*

由模块、子模块等组成的模块化OS结构：



(2)模块独立性

> *在模块–接口法中，关键问题是模块的划分和规定好模块之间的接口。如果我们在划分模块时将模块划分得太小，虽然可以降低模块本身的复杂性，但会引起模块之间的联系过多，从而会造成系统比较混乱；如果将模块划分得过大，又会增加模块内部的复杂性，使内部的联系增加，因此在划分模块时，应在两者间进行权衡。*

(3)模块接口法的优缺点

优点：

- 提高OS设计的正确性、可理解性和可维护性
- 增强OS的可适应性
- 加速OS的开发过程

缺点：

- 在OS设计时，对各模块间的接口规定很难满足在模块设计完成后对接口的实际需求。
- 在OS设计阶段，设计者必须做出一系列的决定(决策)，每一个决定必须建立在上一个决定的基础上，但模块化结构设计中，各模块的设计齐头并进，无法寻找一个可靠的决定顺序，造成各种决定的“无序性”，这将使程序人员很难做到“设计中的每一步决定”都是建立在可靠的基础上，因此模块–接口法又被称为“无序模块法”。

3.分层式结构OS

(1)分层式结构的基本概念

> *为了将模块–接口法中“决定顺序”的无序性变为有序性，引入了有序分层法，分层法的设计任务是，在目标系统An和裸机系统(又称宿主系统)A0之间，铺设若干个层次的软件A1、A2、A3、…、An–1，使An通过An–1、An–2、…、A2、A1层，最终能在A0上运行。在操作系统中，常采用自底向上法来铺设这些中间层。*

(2)分层结构的优缺点

优点

- 易保证系统的正确性
- 易扩充和易维护性

缺点

- 系统效率降低 由于层次结构是分层单向依赖的，必须在每层之间都建立层次间的通信机制，OS每执行一个功能，通常要自上而下地穿越多个层次，这无疑会增加系统的通信开销，从而导致系统效率的降低。

1.5.2 客户/服务器模式(Client/Server Model)简介

1.客户/服务器模式的由来、组成和类型

(1)客户/服务器系统主要由三部分组成

1. 客户机
2. 服务器
3. 网络系统

(2)客户/服务器之间的交互

- 客户发送请求消息
- 服务器接收消息
- 服务器回送消息
- 客户机接收消息

(3)客户/服务器模式的优点

- 数据的分布处理和存储
- 便于集中管理
- 灵活性和可扩充性
- 易于改编应用软件

1.5.3 面向对象的程序设计(Object-Orientated Programming)技术简介

1.面向对象技术的基本概念

> *面向对象技术是20世纪80年代初提出并很快流行起来的。*

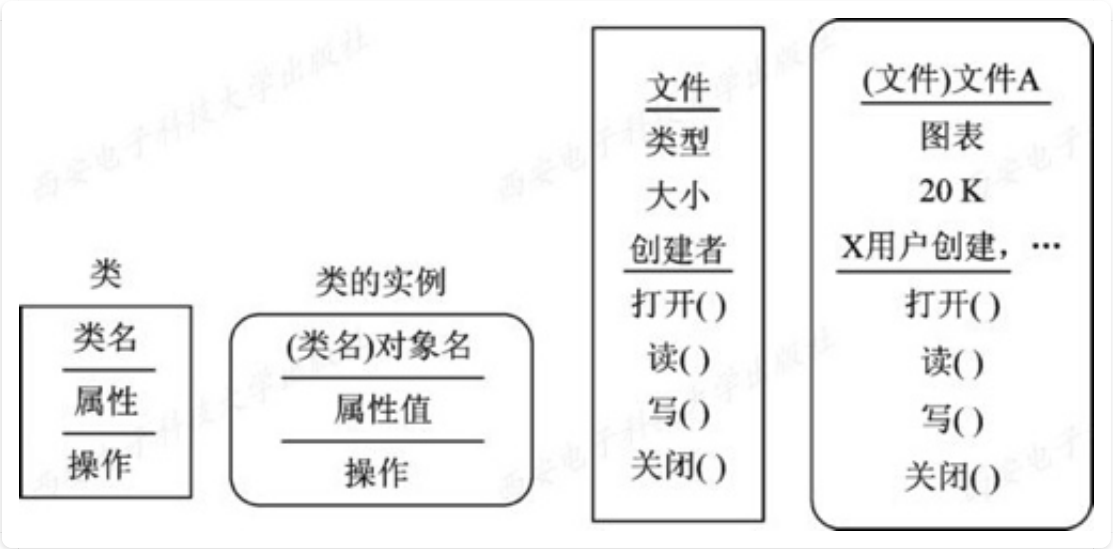
(1)对象

> *在面向对象的技术中，是利用被封装的数据结构(变量)和一组对它进行操作的过程(方法)来表示系统中的某个对象的，如下图。对象中的变量(数据)也称为属性，它可以是单个标量或一张表。面向对象中的方法是用于执行某种功能的过程，它可以改变对象的状态，更新对象中的某些数据值或作用于对象所要访问的外部资源。*

> ***一个对象的示意图：***
>
> 
>

如果把一个文件作为一个对象，该对象的变量便是文件类型、文件大小、文件的创建者等。对象中的方法包含对文件的操作，如创建文件、打开文件、读文件、写文件、关闭文件等。

类和对象的关系：



(2)对象类

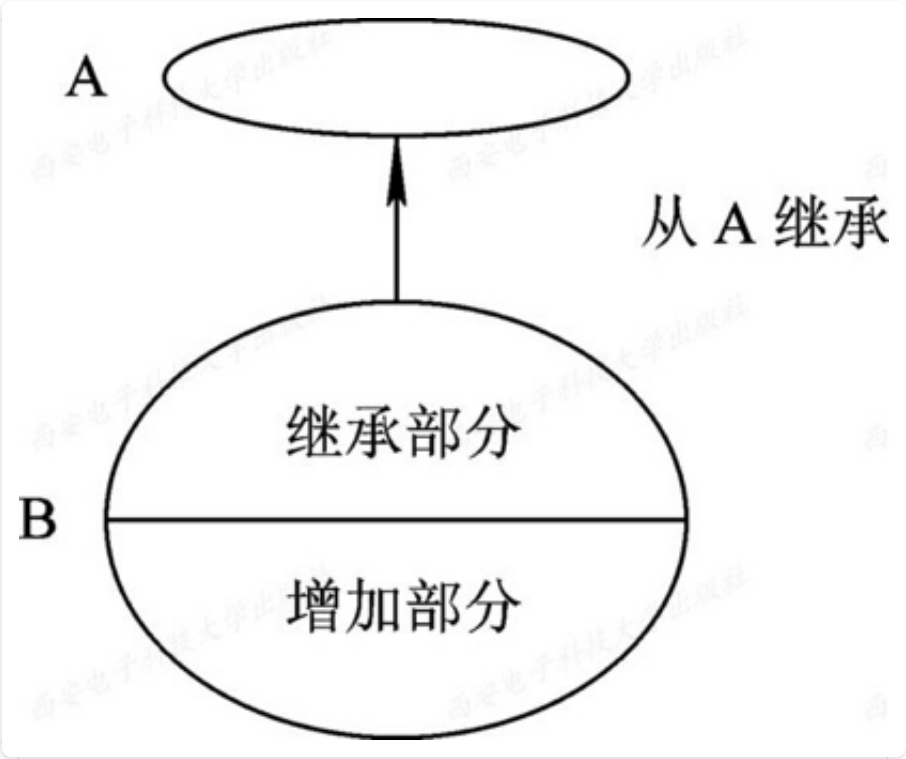
> *在实践中，有许多对象可能表示的是同一类事物，每个对象具有自*

> *己的变量集合，而它们所具有的方法是相同的。如果为每一个相似的对象都定义一组变量和方法，显然是低效的，由此产生了“对象类”的概念，利用“对象类”来定义一组大体相似的对象。一个类同样定义了一组变量和针对该变量的一组方法，用它们来描述一组对象的共同属性和行为。类是在对象上的抽象，对象则是类的实例。对象类中所定义的变量在实例中均有具体的值。*

(3)继承

在面向对象的技术中，可以根据已有类来定义一个新的类，新类被称为子类(B)，原来的类被称为父类(A)

类的继承关系：



(4)面向对象技术的优点

在操作系统设计时，将计算机中的实体作为对象来处理，可带来如下好处：

- 通过“重用”提高产品质量和生产率
- 使系统具有更好的易修改性和易扩展性
- 更易于保证系统的“正确性”和“可靠性”

1.5.4 微内核OS结构

1.微内核操作系统的基本概念

(1)足够小的内核

> *在微内核操作系统中，内核是指精心设计的、能实现现代OS最基本核心功能的小型内核，微内核并非是一个完整的OS，而只是将操作系统中最基本的部分放入微内核。*

通常包含有：

- 与硬件处理紧密相关的部分
- 一些较基本的功能
- 客户和服务器之间的通信。

这些OS最基本的部分只是为构建通用OS提供一个重要基础，这样就可以确保把操作系统内核做得很小。

（2）基于客户/服务器模式

由于客户/服务器模式具有非常多的优点，故在单机微内核操作系统中几乎无一例外地都采用客户/服务器模式，将操作系统中最基本的部分放入内核中，而把操作系统的绝大部分功能都放在微内核外面的一组服务器(进程)中实现，如用于提供对进程(线程)进行管理的进程(线程)服务器、提供虚拟存储器管理功能的虚拟存储器服务器、提供I/O设备管理的I/O设备管理服务器等，它们都是被作为进程来实现的，运行在用户态，客户与服务器之间是借助微内核提供的消息传递机制来实现信息交互的。

在单机环境下的客户/服务器模式:

(3)应用“机制与策略分离”原理

> *在现在操作系统的结构设计中，经常利用“机制与策略分离”的原理*

> *来构造OS结构。*

所谓机制，是指实现某一功能的具体执行机构。而策略，则是在机制的基础上借助于某些参数和算法来实现该功能的优化，或达到不同的功能目标。

(4)采用面向对象技术

操作系统是一个极其复杂的大型软件系统，我们不仅可以通过结构设计来分解操作系统的复杂度，还可以基于面向对象技术中的“抽象”和“隐蔽”原则控制系统的复杂性，再进一步利用“对象”、“封装”和“继承”等概念来确保操作系统的“正确性”、“可靠性”、“易修改性”、“易扩展性”等，并提高操作系统的设计速度。正因为面向对象技术能带来如此多的好处，故面向对象技术被广泛应用于现代操作系统的设计中。

2.微内核的基本功能

> *现在一般都采用“机制与策略分离”的原理，将机制部分以及与硬件紧密相关的部分放入微内核中。*

由此可知微内核通常具有如下几方面的功能：

- 进程(线程)管理
- 低级存储器管理
- 中断和陷入处理

3.微内核操作系统的优点

> *由于微内核OS结构是建立在模块化、层次化结构的基础上的，并采用了客户/服务器模式和面向对象的程序设计技术*

因此，微内核结构的操作系统是集各种技术优点之大成，因而使之具有如下优点：

- 提高了系统的可扩展性
- 增强了系统的可靠性
- 可移植性强
- 提供了对分布式系统的支持
- 融入了面向对象技术

4.微内核操作系统存在的问题

微内核OS存在着潜在缺点，其中最主要的是，较之早期的操作系统，微内核操作系统的运行效率有所降低。

实际情况是往往还会引起更多的上下文切换。

例如，当某个服务器自身尚无能力完成客户请求而需要其它服务器的帮助时，如图所示，其中的文件服务器还需要磁盘服务器的帮助，这时就需要进行8次上下文的切换。