第6章 I/O系统

I/O系统概述

操作系统中负责管理输入输出设备的部分称为I/O系统，完成设备管理功能。

本章要点

- I/O系统概述
- I/O软件的组成
- Windows 2000/XP I/O系统结构
- Windows 2000/XP I/O系统的数据结构
- Windows 2000/XP的设备驱动程序
- Windows 2000/XP的I/O处理

设备管理的重要性

- 设备管理是对硬件资源中除CPU、存储器之外的所有设备进行管理，而外设的投资通常占整个系统投资的50~80，管理好整个系统的设备，使其高效地发挥功用是操作系统的重要任务。本章主要介绍设备的1/0控制方式，设备缓冲技术，磁盘调度以及设备管理子系统。
- 设备管理的重要性表现在以下方面：

  输入输出设备的性能经常成为系统性能的瓶颈。

  输入输出设备千差万别，对它们要实现统一管理

  及时传送信息对实时处理和控制系统非常重要。

设备的分类

1 实用特性

存储设备：磁盘、磁带

I/O设备：穿孔卡片、键盘、鼠标、显示器、打印机

终端设备：交互设备

2 信息组织方式：

1 字符设备：字符为单位组织和处理信息的设备，如键盘、终端、打印机

2 块设备：字符块为单位组织和处理信息的设备，如磁盘、磁带

设备一次操作的数据传送单位

通常输入输出类设备都是字符设备

存储设备都是块设备

3 按照设备使用可共享性分类

独占设备：指在一段时间内只允许一个用户进程使用的设备。多数低速I/O设备都属于独占设备，如打印机

共享设备：在一段时间内允许多个进程使用的设备。如磁盘，若干个进程可以交替从磁盘上读写信息

虚拟设备：通过虚拟技术将一台独占设备变换成为若干逻辑设备，供若干个进程同时使用

SOOLING技术

I/O系统的功能和目标

设备管理的主要任务：

完成用户提出的I/O请求

为用户分配I/O设备

提高I/O设备的利用率

方便用户使用I/O设备

为了完成上述任务，设备管理应具备以下功能：

1 设备分配：

2 设备处理：

3 缓冲管理：

4 设备独立性：

5 提高设备与设备、cpu与设备间并行操作

设备分配：

按照设备类型和相应的分配算法决定I/O设备分配给哪一个要求使用该设备的进程。在I/O设备与CUP之间还需分配相应控制器和通道。未分配到所需设备的进程应放入一个等待队列。

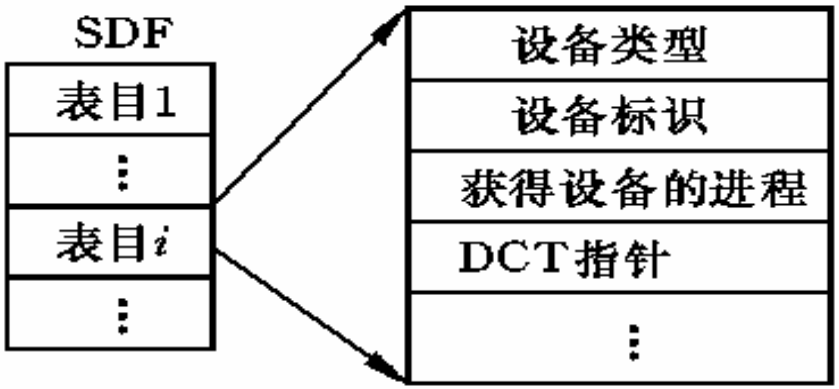
不是每个进程随时都能得到资源（设备、通道、控制器）

1 设备控制表DCT:反映设备的特性

2 系统设备表SDT:反映设备的资源状态

3 控制器控制表COCT：I/O控制器状态

4 通道控制表 CHCT：通道状态



DCT

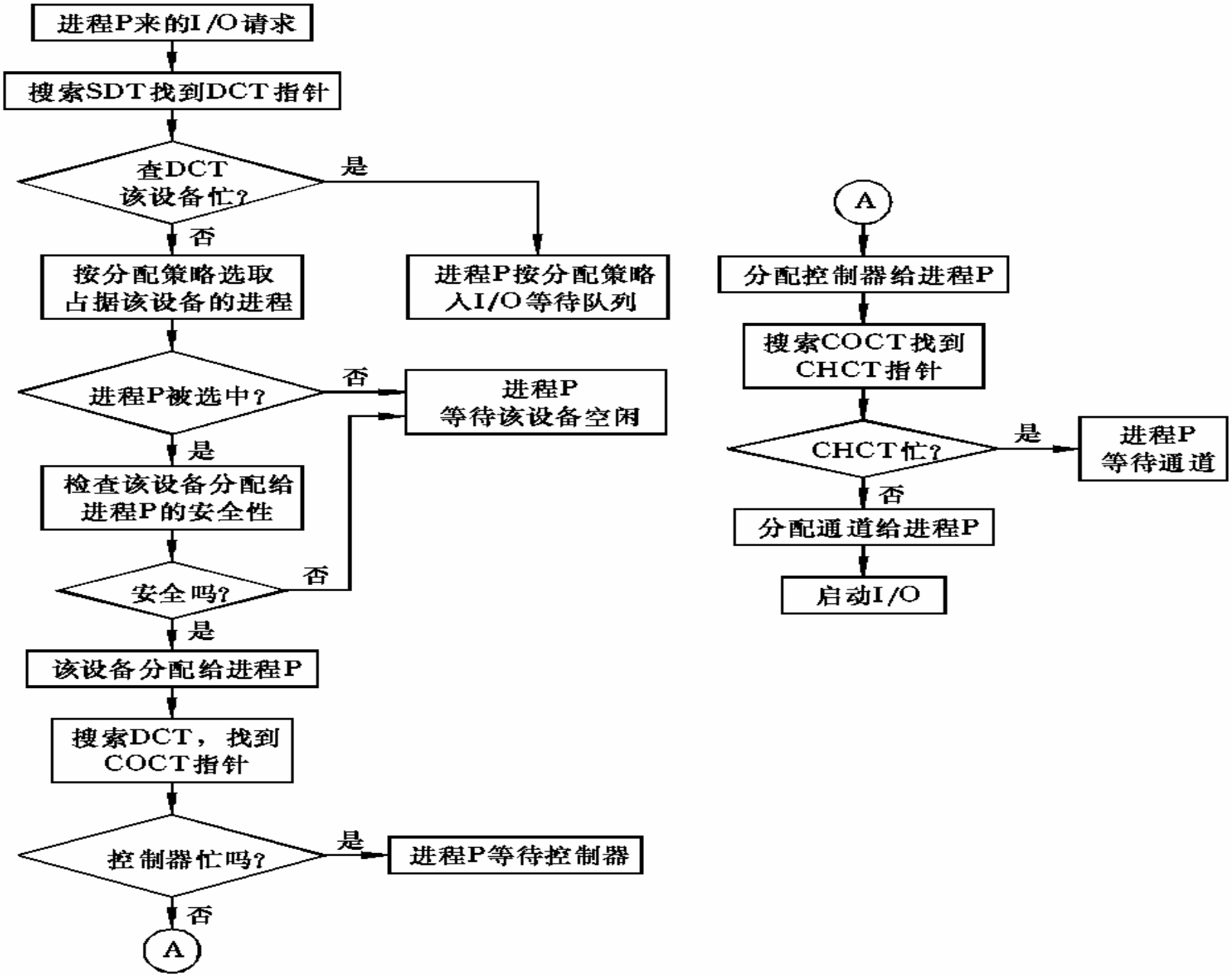
设备类型
设备标识
设备忙／闲标记
COCT指针
控制器等待队列首
控制器等待队列尾
⋮

CDCT

控制器标识
控制忙／闲标记
CHCT指针
控制器等待队列首
控制器等待队列尾
⋮

CHCT

通道标识
通道忙／闲标记
通道等待队列首
控制器等待队列尾
⋮


进程P来的I/O请求
搜索SDT找到DCT指针
查DCT
该设备忙?
是
否
进程P按分配策略
入I/O等待队列
按分配策略选取
占据该设备的进程
进程P被选中?
否
进程P
等待该设备空闲
是
检查该设备分配给
进程P的安全性
安全吗?
否
是
该设备分配给进程P
搜索DCT，找到
COCT指针
控制器忙吗?
是
进程P等待控制器
否
A
A
分配控制器给进程P
搜索COCT找到
CHCT指针
CHCT忙?
是
进程P
等待通道
否
分配通道给进程P
启动I/O

设备分配原则

1 设备忙，避免进程死锁

2 设备独立性

设备分配方式

1 静态：建立进程时一次性分配

2 动态：进程运行过程中申请和释放

设备分配策略

1 先来先服务

2 优先级高者

数据传送控制方式有4种

(1) 程序直接控制方式；
(2) 中断控制方式；
(3) DMA方式；
(4) 通道方式。

程序直接控制方式

1 当用户进程需要数据时，CPU发出启动设备准备数据的启动命令“Start”

2 用户进程进入测试等待状态

3 等待时间内，CPU循执行测试指令检查设备状态寄存器。

4 当设备状态寄存器的值显示为输入完成（ CPU检测）。

5 寄存器发出“Done”信号之后，设备开始往内存或CPU传送数据。

I/O控制器中还有一类称为数据缓冲寄存器的寄存器。在CPU与外围设备之间传送数据时，

中断方式

为了减少程序直接控制方式中CPU等待时间以及提高系统的并行工作程度，中断(interrupt)方式被用来控制外围设备和内存与CPU之间的数据传送。在设备控制器的控制状态寄存器的相应的中断允许位。

中断的分类

根据申断信号的含义和功能分为以下五类:

- 机器故障中断:因机器发生错误（电源故障，内存读数错误等)而产生的中断，用以反映硬件故障，以便进入诊断程序。
- I/O中断:由输人入输出设备引起的中断，用以反映通道或外部设备工作状态。
- 外中断:由各种外部事件引起的中断，用以反映外部的要求。
- 程序性中断:因程序中错误使用指令或数据引起的中断，用以反映程序执行过程中发生的例外情况。
- 访管中断:由于程序执行了"访管"指令（系统调用)而产生的中断，用于反映用户程序所请求操作系统为其完成某项工作。

DMA方式

- DMA控制器可用来代替CPU控制内存和设备之间进行成批的数据交换。
- DMA方式仍存在着一定的局限性。

DMA方式对外围设备的管理和某些操作仍由CPU控制。在大中型计算机中，系统所配置的外设种类越来越多，数量也越来越大，因而，对外围设备的管理的控制也就愈来愈复杂。多个DMA控制器的同时使用显然会引起内存地址的冲突并使得控制过程进一步复杂化。

通道控制方式

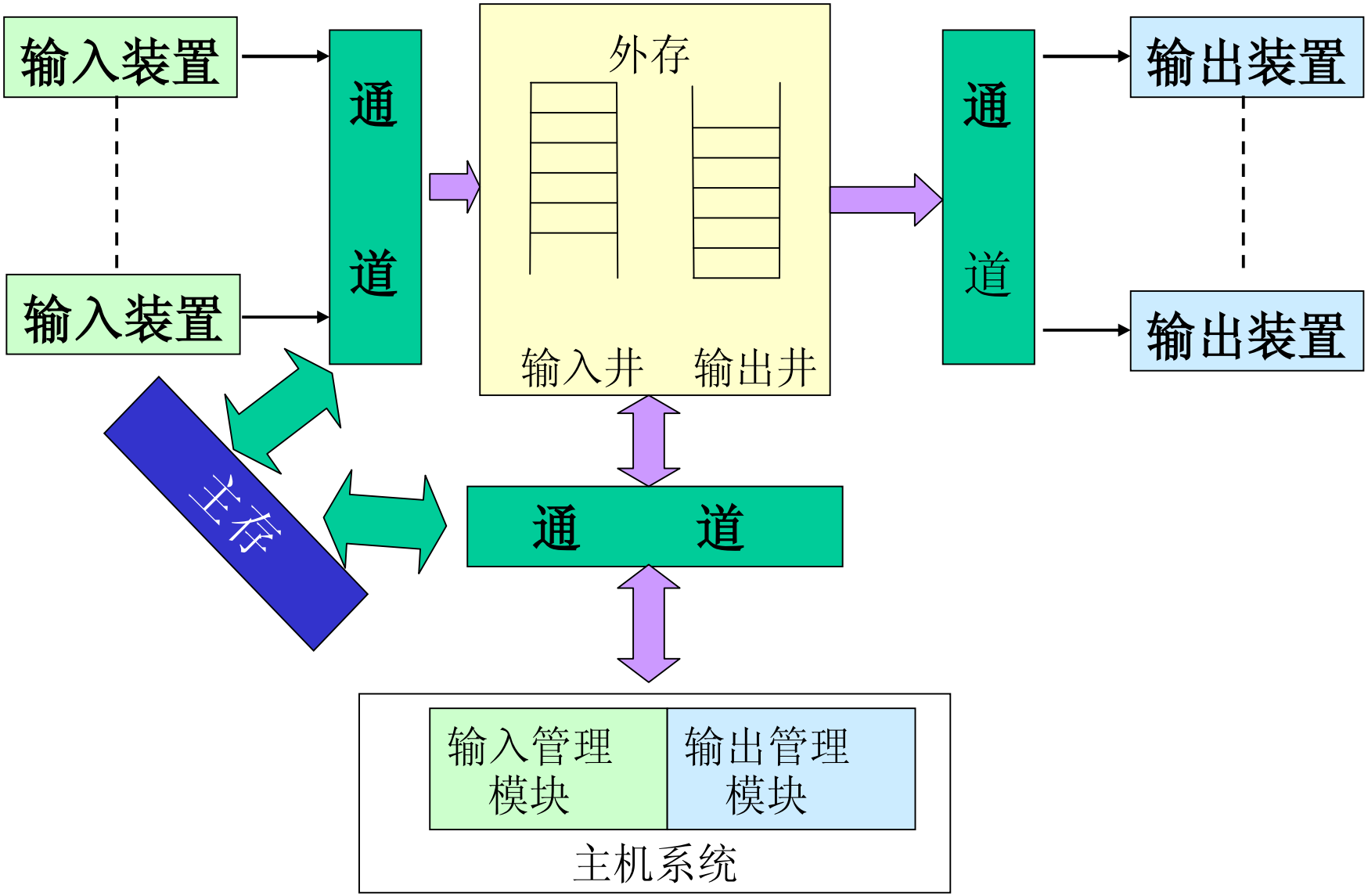
在DMA方式中，数据的传送方向、存放数据的内存始址以及传送的数据块长度等都由CPU控制，而在通道方式中，这些都由专管输入输出的硬件——通道来进行控制。

按照信息交换方式不同，一个系统中可设立三种类型的通道，即字节多路通道、数组多路通道和选择通道。

I/O软件的层次

SPOOLING系统

- SPOOLING又可译作外围设备同时联机操作。多台外围设备通过通道或DMA器件和主机与外存连接起来。



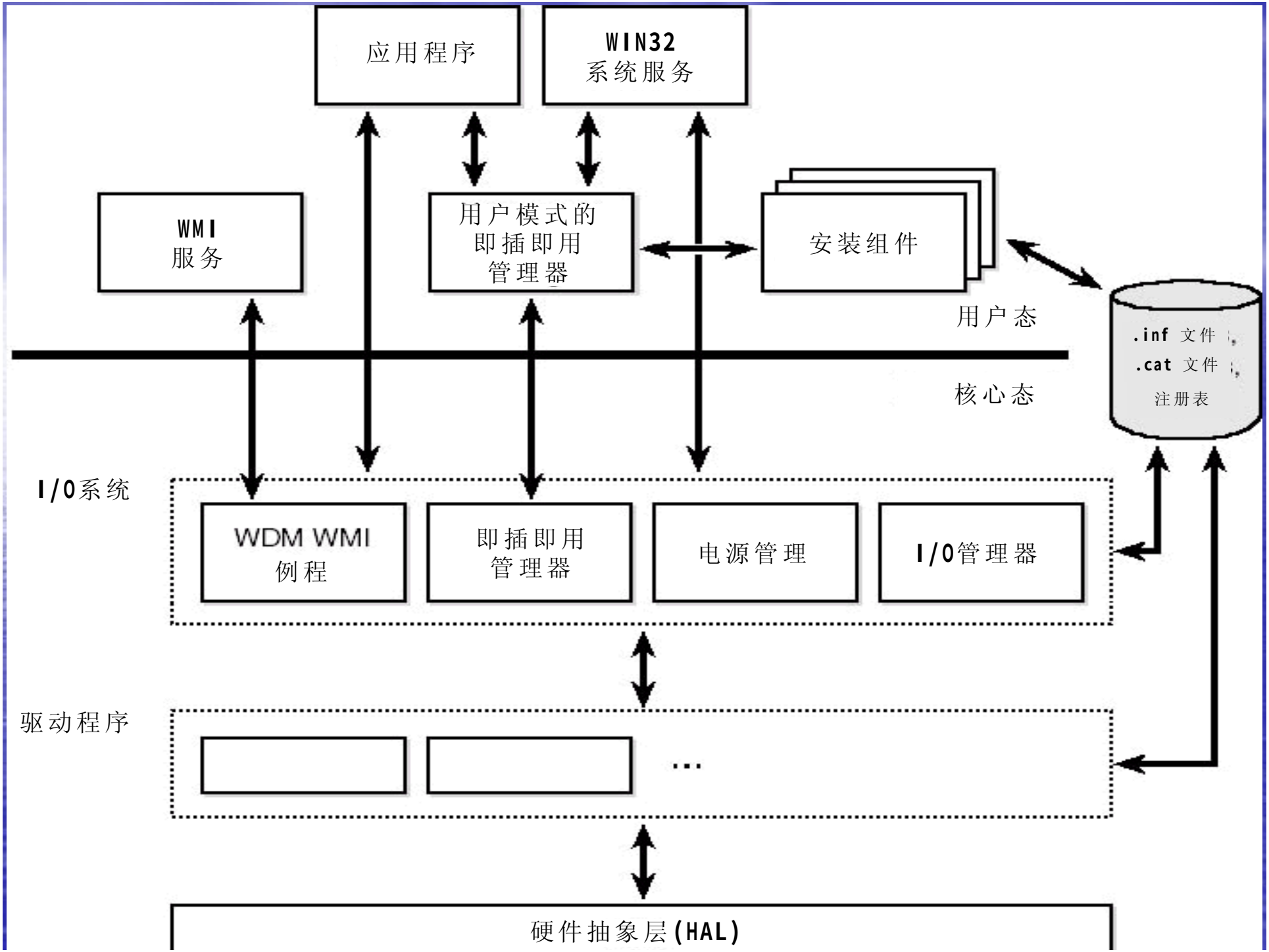
SPOOLING系统


输入井
作业4
作业3
作业5
作业2
输出井
作业3
作业2
作业4
作业1
写过程将缓冲数据写入磁盘
读过程将磁盘数据读入缓冲
作业5
输入设备
存输入
取输入
存输出
取输出
作业1
打印机
作业2
作业3
作业4
主　机
读过程从设备读数到缓冲区
写过程将缓冲数据送打印机

Windows 2000/XP的I/O系统结构

- Windows 2000/XP的I/O系统是重要的执行体组件
- 设计目标
  - 在单处理器或多处理器系统中都可以快速进行I/O处理。
  - 使用标准的Windows 2000/XP安全机制保护共享的资源。
  - 满足Microsoft Win32、OS/2和POSIX子系统指定的I/O服务的需要。
  - 提供服务，使设备驱动程序的开发尽可能地简单，并且允许用高级语言编写驱动程序。

- 根据用户的配置或者系统中硬件设备的添加和删除，允许在系统中动态地添加或删除相应的设备驱动程序。
- 通过添加驱动程序透明地修改其他驱动程序或设备的行为。
- 为包括FAT、CD－ROM文件系统（CDFS）、UDF（Universal Disk Format）文件系统和Windows 2000/XP文件系统（NTFS）的多种可安排的文件系统提供支持。
- 允许整个系统或者单个硬件设备进入和离开低功耗状态，这样可以节约能源。


应用程序
WIN32
系统服务
WMI
服务
用户模式的
即插即用
管理器
安装组件
用户态
核心态
.inf 文件
.cat 文件
注册表
I/O系统
WDM WMI
例程
即插即用
管理器
电源管理
I/O管理器
驱动程序
...
硬件抽象层(HAL)

即插即用系统组成

- 1 注册表：维护已经安装的硬件和PNP设备软件的数据库，帮助驱动程序和其他组件识别和定位设备使用的资源。
- 2 INF文件：设备驱动程序安装使用的一个安装信息文件。
- 3 PNP管理程序：内核模式部分与硬件交互，管理硬件资源的正确配置和检测；用户模式组件与用户接口组件交互.。
- 4 I/O管理程序：将设备驱动程序的请求创建成一个I/O请求包，将此数据结构传递给驱动程序。
- 5 插即用驱动程序： 驱动程序的实现建立在WDM 设备驱动程序分层处理的设备栈管理之上。

- I/O管理器把应用程序和系统组件连接到各种虚拟的、逻辑的和物理的设备上，并且定义了一个支持设备驱动程序的基本构架。
- 设备驱动程序为某种类型的设备提供一个I/O接口。设备驱动程序从I/O管理器接受处理命令，当处理完毕后通知I/O管理器。设备驱动程序之间的协同工作也通过I/O管理器进行。
- PnP（即插即用，plug and play）管理器通过与I/O管理器和总线驱动程序的协同工作检测硬件资源的分配，并且检测相应硬件设备的添加和删除
- 电源管理器通过与I/O管理器的协同工作检测整个系统和单个硬件设备，完成不同电源状态的转换。

- WMI（Windows Management Instrumentation）支持例程，也叫做Windows驱动程序模型（WDM，Windows Driver Model）WMI提供者，允许驱动程序使用这些支持例程作为媒介，与用户态运行的WMI服务通讯。
- 注册表作为一个数据库，存储基本硬件设备的描述信息以及驱动程序的初始化和配置信息。
- 硬件抽象层（HAL）I/O访问例程把设备驱动程序与多种多样的硬件平台隔离开来，使它们在给定的体系结构中是二进制可移植的，并在Windows 2000/XP支持的硬件体系结构中是源代码可移植的。

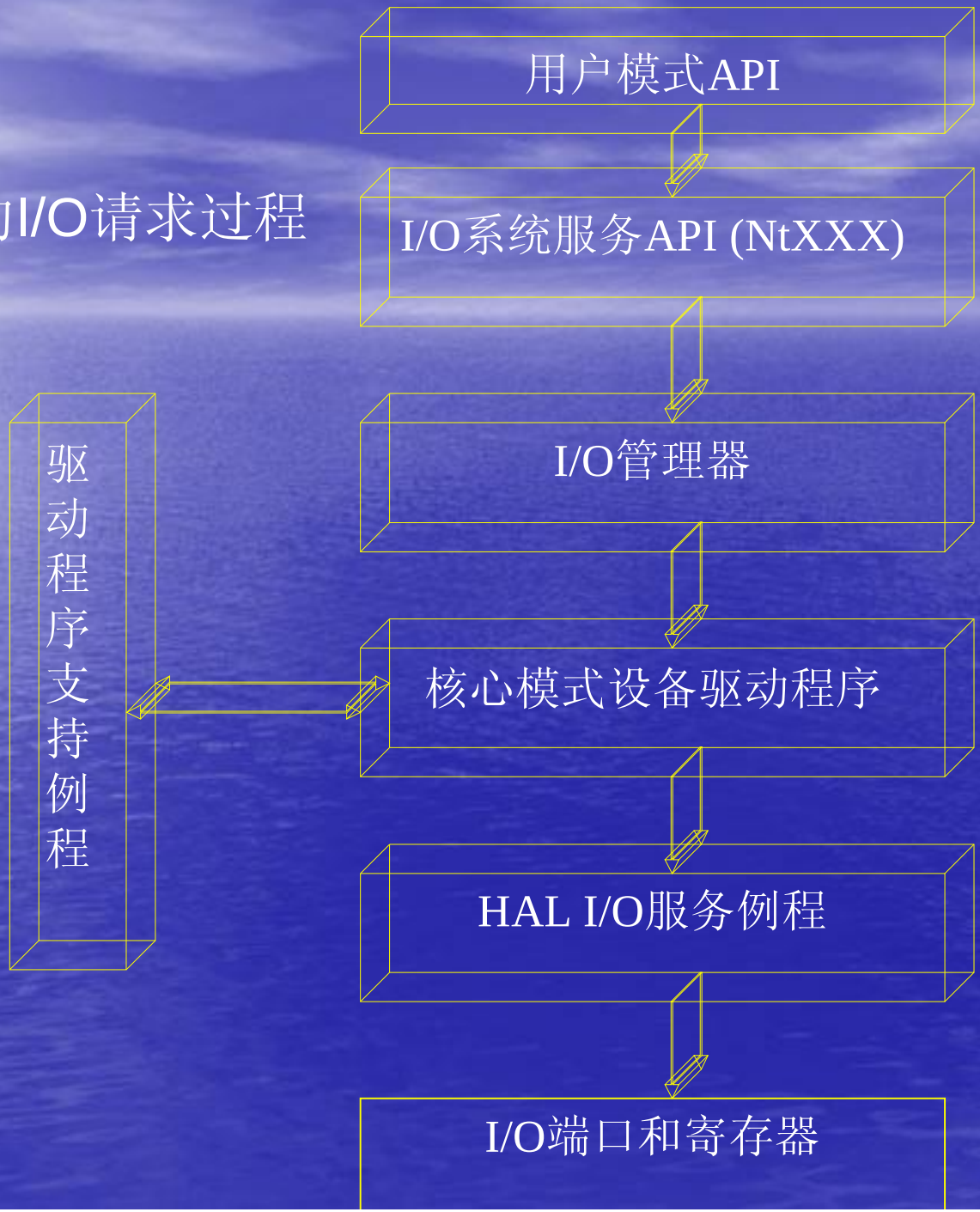
Windows 2000/XP的I/O类型

- 同步I/O和异步I/O
  - “同步”：设备执行数据传输并在I/O完成时返回一个状态码，然后程序就可以立即访问被传输的数据
  - “异步”：应用程序发布I/O请求，然后当设备传输数据的同时，应用程序继续执行
- 快速I/O：允许I/O系统不产生IRP而直接到文件系统驱动程序或高速缓存管理器去执行I/O请求
- 映射文件I/O和文件高速缓存：把磁盘中的文件视为进程的虚拟内存的一部分，程序可以把文件作为一个大的数组来访问，而无需做缓冲数据或执行磁盘I/O的工作
- 分散/集中I/O：应用程序执行一个读取或写入操作，从虚拟内存中的多个缓冲区读取数据并写到磁盘上文件的一个连续区域里

Windows 2000/XP I/O特点

- 在Windows 2000/XP中，所有的I/O操作都通过虚拟文件执行，隐藏了I/O操作目标的实现细节，为应用程序提供了一个统一的到设备的接口界面
- 用户态应用程序调用文档化的函数，这些函数再依次地调用内部I/O子系统函数来从文件中读取、对文件写入和执行其他的操作。I/O管理器动态地把这些虚拟文件请求指向适当的设备驱动程序

一个典型的I/O请求过程



I/O管理器（I/O manager）

- I/O管理器定义有序的工作框架，在该框架里，I/O请求被提交给设备驱动程序
- 大多数I/O请求用“I/O请求包（IRP）”表示，I/O系统是由“包”驱动的，这些包它从一个I/O系统组件移动到另一个I/O系统组件
- I/O管理器创建代表每个I/O操作的IRP，传递IRP给正确的驱动程序，并且当此I/O操作完成后，处理这个数据包
- I/O管理器还为不同的驱动程序提供了公共的代码，驱动程序调用这些代码来执行它们的I/O处理

PnP管理器

- PnP管理器自动识别所有已经安装的硬件设备。
- PnP管理器通过一个名为资源仲裁（resource arbitrating）的进程收集硬件资源需求（中断，I/O地址等）来实现硬件资源的优化分配，满足系统中的每一个硬件设备的资源需求。
- PnP管理器通过硬件标识选择应该加载的设备驱动程序。
- PnP管理器也为检测硬件配置变化提供了应用程序和驱动程序的接口，因此在Windows 2000/XP中，在硬件配置发生变化的时候，相应的应用程序和驱动程序也会得到通知。

电源管理器

- 电源管理需要底层硬件符合ACPI标准
- ACPI为系统和设备定义了不同的能耗状态，从S0（正常工作）到S5（完全关闭）
  - 电源消耗：计算机系统消耗的能源
  - 软件运行恢复：计算机系统回复到正常工作状态时软件能否恢复运行
  - 硬件延迟：计算机系统回复到正常工作状态的时间延迟

- Windows 2000/XP电源管理策略
  - 电源管理器是系统电源策略的所有者，因此整个系统的能耗状态转换由电源管理器决定，并调用相应设备的驱动程序完成，电源管理器根据以下因素决定当前的能耗状态
    - 系统活动状况
    - 系统电源状况
    - 应用程序的关机、休眠请求
    - 用户的操作，例如用户按电源按钮
    - 控制面板的电源设置
  - 设备驱动程序可以独立地控制设备的能耗状态

Windows2000/xp I/O系统数据结构

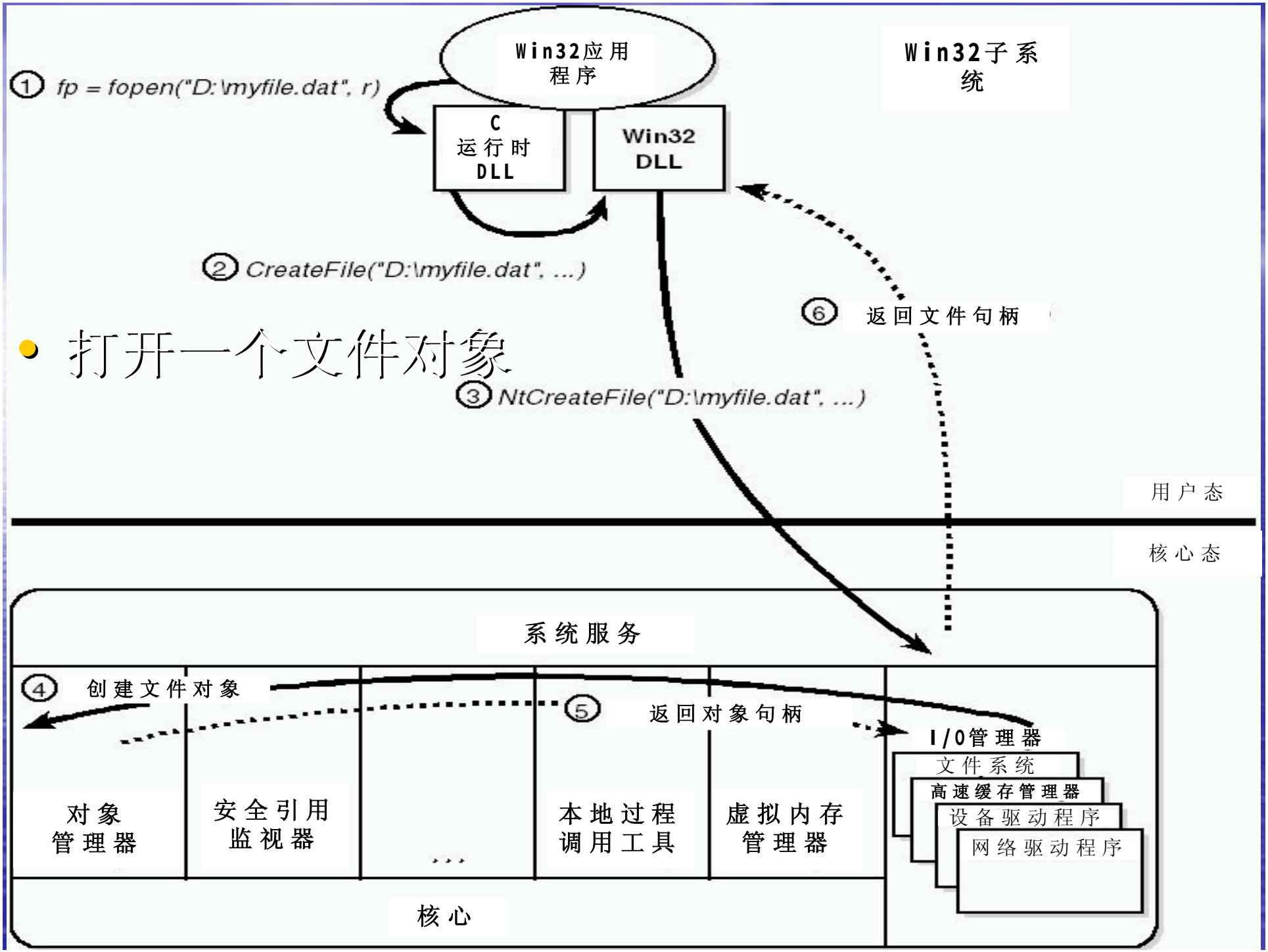
- 文件对象
- 驱动程序对象
- 设备对象
- I/O请求包（IRP）

文件对象

- 提供了基于内存的共享物理资源的表示法

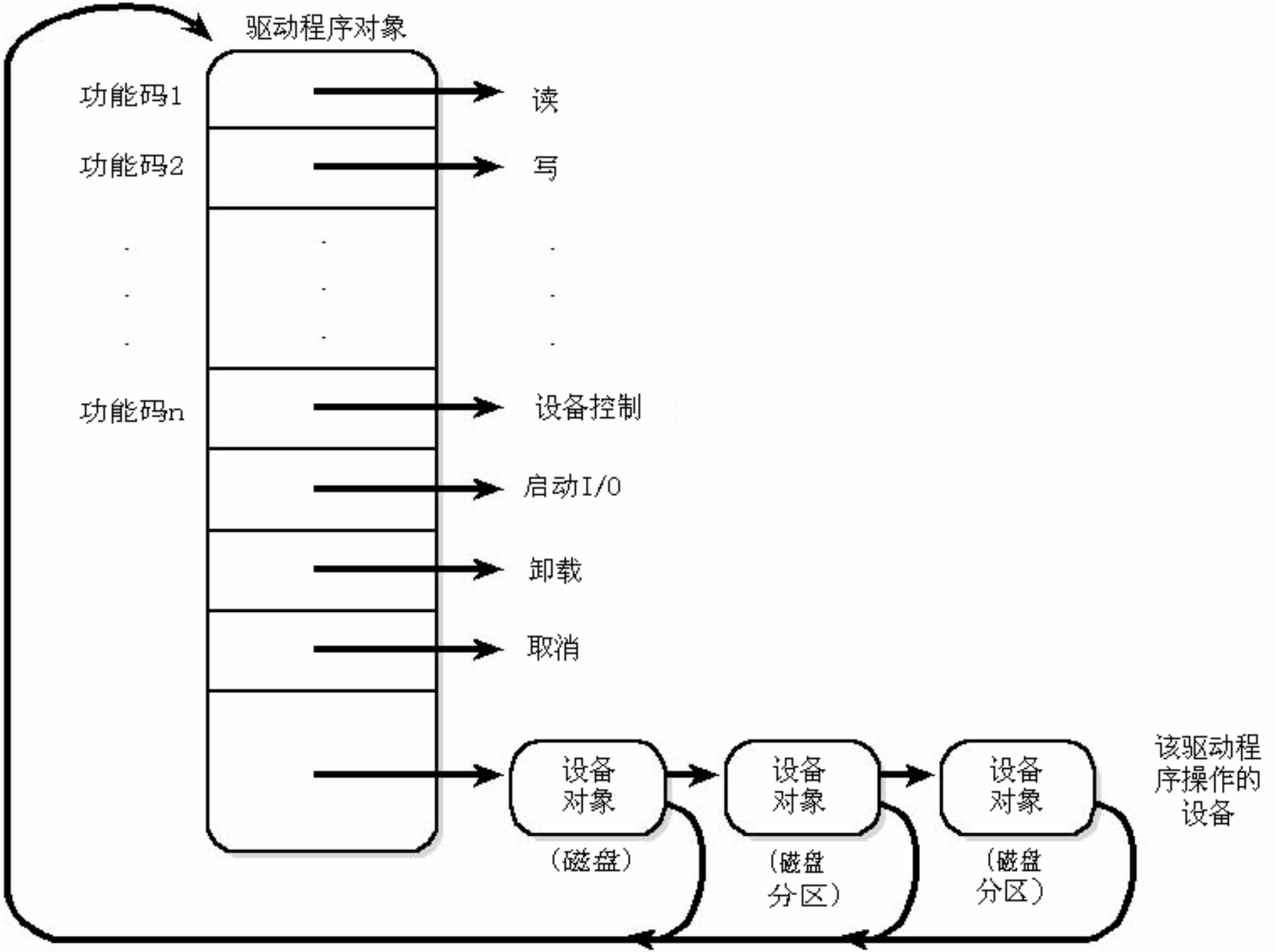
属性	目的
文件名	标识文件对象指向的物理文件。
字节偏移量	在文件中标识当前位置（只对同步I/O有效）。
共享模式	表示当调用者正在使用文件时，其他的调用者是否可以打开文件来做读取、写入或删除操作。
打开模式	表示I/O是否将被同步或异步、高速缓存或不高速缓存、连续或随机等等。
指向设备对象的指针	表示文件在其上驻留的设备的类型。
指向卷参数块的指针	表示文件在其上驻留的卷或分区。
指向区域对象指针的指针	表示描述一个映射文件的根结构
指向专用高速缓存映射的指针	表示文件的哪一部分由高速缓存管理器管理高速缓存，以及它们驻留在高速缓存的什么地方。

- 打开一个文件对象


Win32应用程序
Win32子系统
① fp = fopen("D:\myfile.dat", r)
C运行时DLL
Win32 DLL
② CreateFile("D:\myfile.dat", ...)
⑥ 返回文件句柄
③ NtCreateFile("D:\myfile.dat", ...)
用户态
核心态
系统服务
④ 创建文件对象
⑤ 返回对象句柄
I/O管理器
文件系统
高速缓存管理器
设备驱动程序
网络驱动程序
对象管理器
安全引用监视器
...
本地过程调用工具
虚拟内存管理器
核心

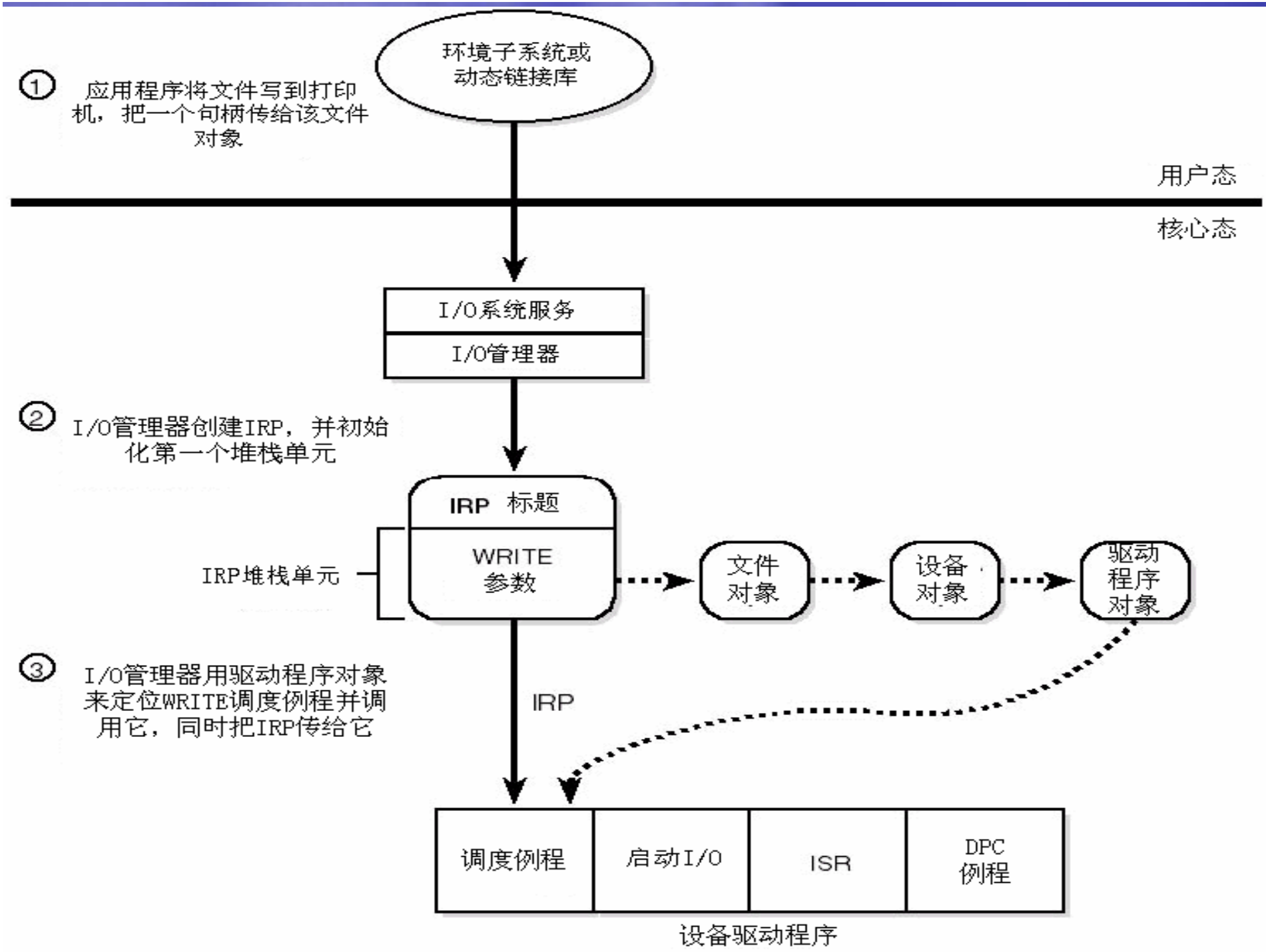
驱动程序对象和设备对象

- 驱动程序对象在系统中代表一个独立的驱动程序，并且为I/O记录每个驱动程序的调度例程的地址（入口点）。
- 设备对象在系统中代表一个物理的、逻辑的或虚拟的设备并描述了它的特征。


驱动程序对象
功能码1
功能码2
功能码n
读
写
设备控制
启动I/O
卸载
取消
设备
对象
设备
对象
设备
对象
（磁盘）
（磁盘
分区）
（磁盘
分区）
该驱动程
序操作的
设备

I/O请求包

- 存储处理I/O请求所需信息
- 线程调用I/O服务时，I/O管理器就构造一个IRP来表示在整个系统I/O进展中要进行的操作
- I/O管理器在IRP中保存一个指向调用者文件对象的指针
- 两部分组成：
  - 固定部分（称作标题）：请求的类型和大小、是同步请求还是异步请求、用于缓冲I/O的指向缓冲区的指针和随着请求的进展而变化的状态信息
  - 一个或多个堆栈单元：一个功能码、功能特定的参数和一个指向调用者文件对象的指针


环境子系统或
动态链接库
① 应用程序将文件写到打印机，把一个句柄传给该文件对象
用户态
核心态
I/O系统服务
I/O管理器
② I/O管理器创建IRP，并初始化第一个堆栈单元
IRP 标题
IRP堆栈单元
WRITE
参数
文件
对象
设备
对象
驱动
程序
对象
③ I/O管理器用驱动程序对象来定位WRITE调度例程并调用它，同时把IRP传给它
IRP
调度例程
启动I/O
ISR
DPC
例程
设备驱动程序

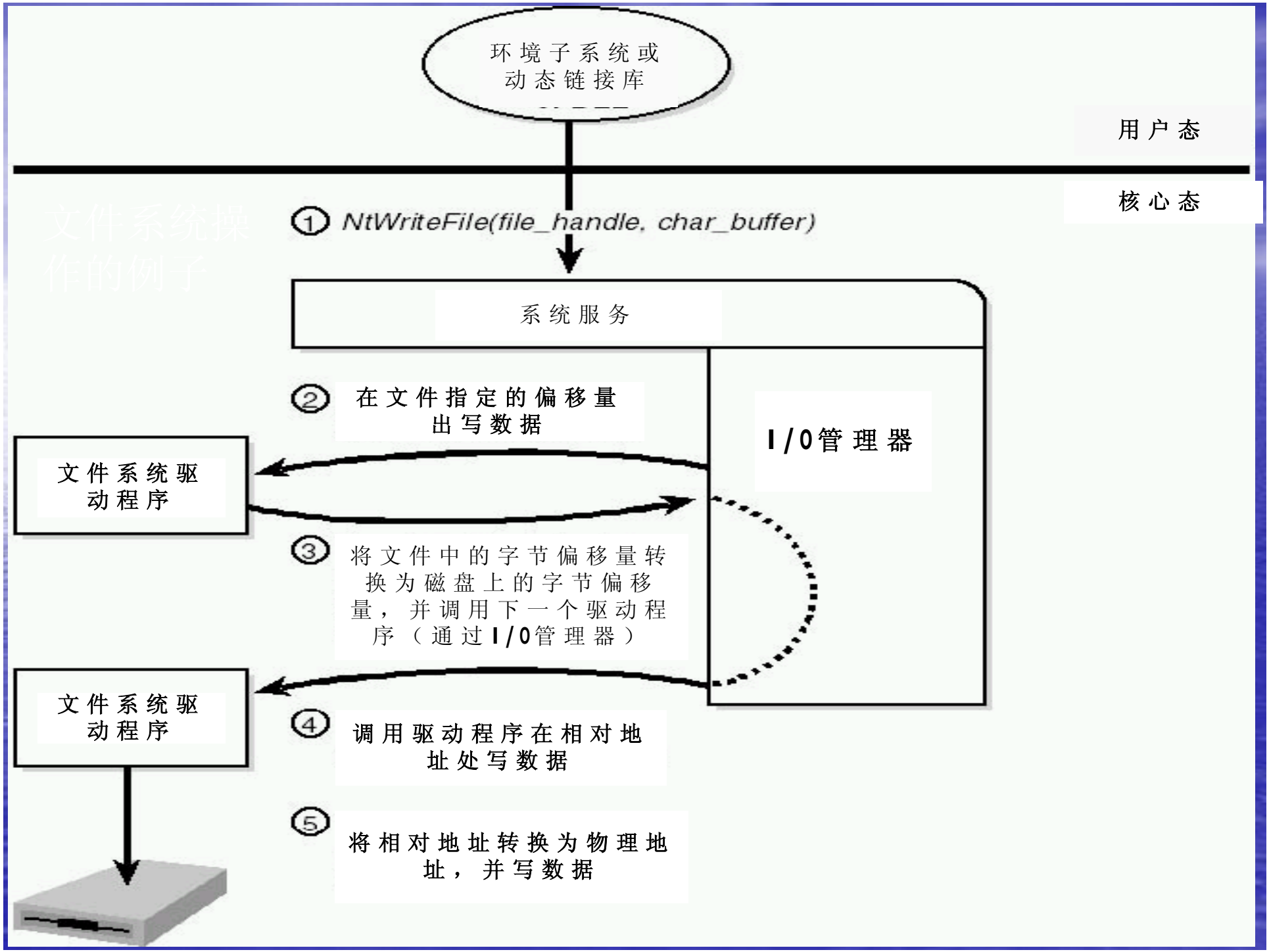
Windows 2000/XP设备驱动程序

- 支持多种类型的设备驱动程序和编程环境
- 核心驱动程序的分类
  - 文件系统驱动接受访问文件的I/O请求。
  - 同Windows 2000/XP的PnP管理器和电源管理器有关的设备驱动程序。
  - 为Windows NT编写的设备驱动程序。
  - Win32子系统显示驱动程序和打印驱动程序将把设备无关的图形（GDI）请求转换为设备专用请求。
  - 符合Windows驱动程序模型（WDM，Windows Drivers Model）的WDM驱动程序，包括对PnP，电源管理和WMI的支持。

- 有三种类型的WDM驱动程序
  - 总线驱动程序（**bus driver**）管理逻辑的或物理的总线，例如PCMCIA， PCI， USB， IEEE 1394， 和ISA，总线驱动程序需要检测并向PnP管理器通知总线上的设备，并且能够管理电源。
  - 功能驱动程序（**function driver**）管理具体的一种设备，对硬件设备进行的操作都是通过功能驱动程序进行的。
  - 过滤驱动程序（**filter driver**）与功能驱动程序协同工作，用于增加或改变功能驱动程序的行为。

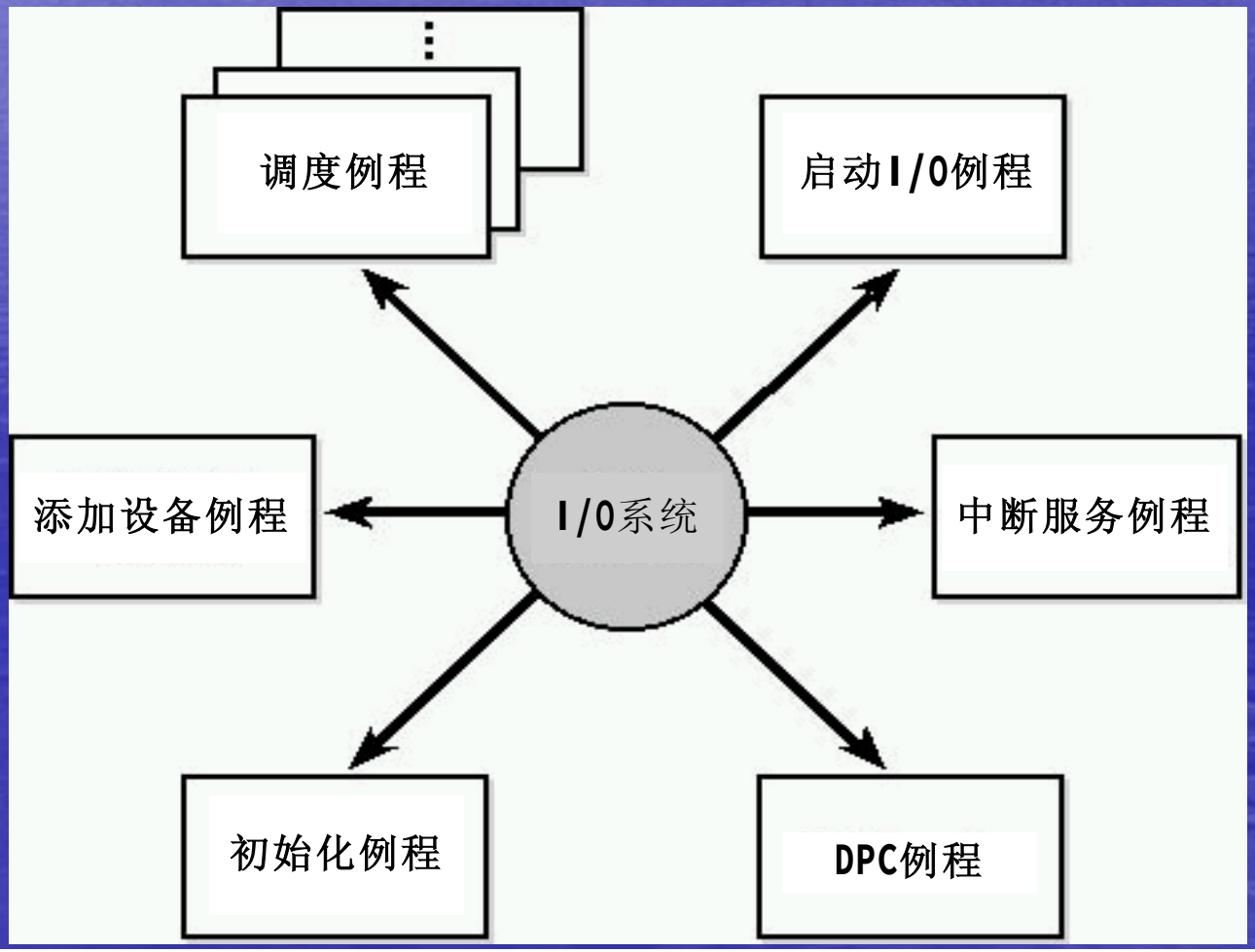
- 用户态的驱动程序
  - 虚拟设备驱动程序（VDD）通常用于模拟16位MS－DOS应用程序。它们捕获MS－DOS应用程序对I/O端口的引用，并将其转化为本机Win32 I/O函数。Windows 2000/XP中用户态MS－DOS应用程序不能直接访问硬件，而必须通过一个真正的核心设备驱动程序。
  - Win32子系统的打印驱动程序将与设备无关的图形请求转换为打印机相关的命令，这些命令再发给核心模式的驱动程序例如并口驱动(Parport.sys)、USB打印机驱动(Usbprint.sys)等

- 硬件支持驱动可以分类如下
  - 类驱动程序（class drivers）为某一类设备执行I/O处理，例如磁盘、磁带或光盘。
  - 端口驱动程序（port drivers）实现了对特定于某一种类型的I/O端口的I/O请求的处理，例如SCSI。
  - 小端口驱动程序把对端口类型的一般的I/O请求映射到适配器类型。例如，一个特定的SCSI适配器。


环境子系统或
动态链接库
用户态
核心态
文件系统操作的例子
① NtWriteFile(file_handle, char_buffer)
系统服务
② 在文件指定的偏移量出写数据
I/O管理器
文件系统驱动程序
③ 将文件中的字节偏移量转换为磁盘上的字节偏移量，并调用下一个驱动程序（通过I/O管理器）
文件系统驱动程序
④ 调用驱动程序在相对地址处写数据
⑤ 将相对地址转换为物理地址，并写数据

驱动程序的结构

- 设备驱动程序包括一组被调用处理I/O请求不同阶段的例程



- 初始化例程，当I/O管理器把驱动程序加载到操作系统中时，它执行驱动程序的初始化例程。
- 添加设备例程，用于支持PnP管理器的操作
- 一系列调度例程，调度例程是设备驱动程序提供的主要函数。
- 启动I/O例程，驱动程序可以使用启动I/O例程来初始化与设备之间的数据传输。
- 中断服务例程（ISR），当一个设备中断时，内核的中断调度程序把控制转交给这个例程。ISR运行在高级的设备中断请求级（IRQL）上，越简单越好，以避免对低优先级中断产生不希望的阻塞。
- 中断服务DPC例程，DPC例程执行在ISR执行以后的大部分设备中断处理工作。DPC例程在低于ISR的IRQ的时候执行，从而避免对其他中断产生不希望的阻塞。DPC例程初始化I/O完成并启动关于设备的下一个队列的I/O操作。

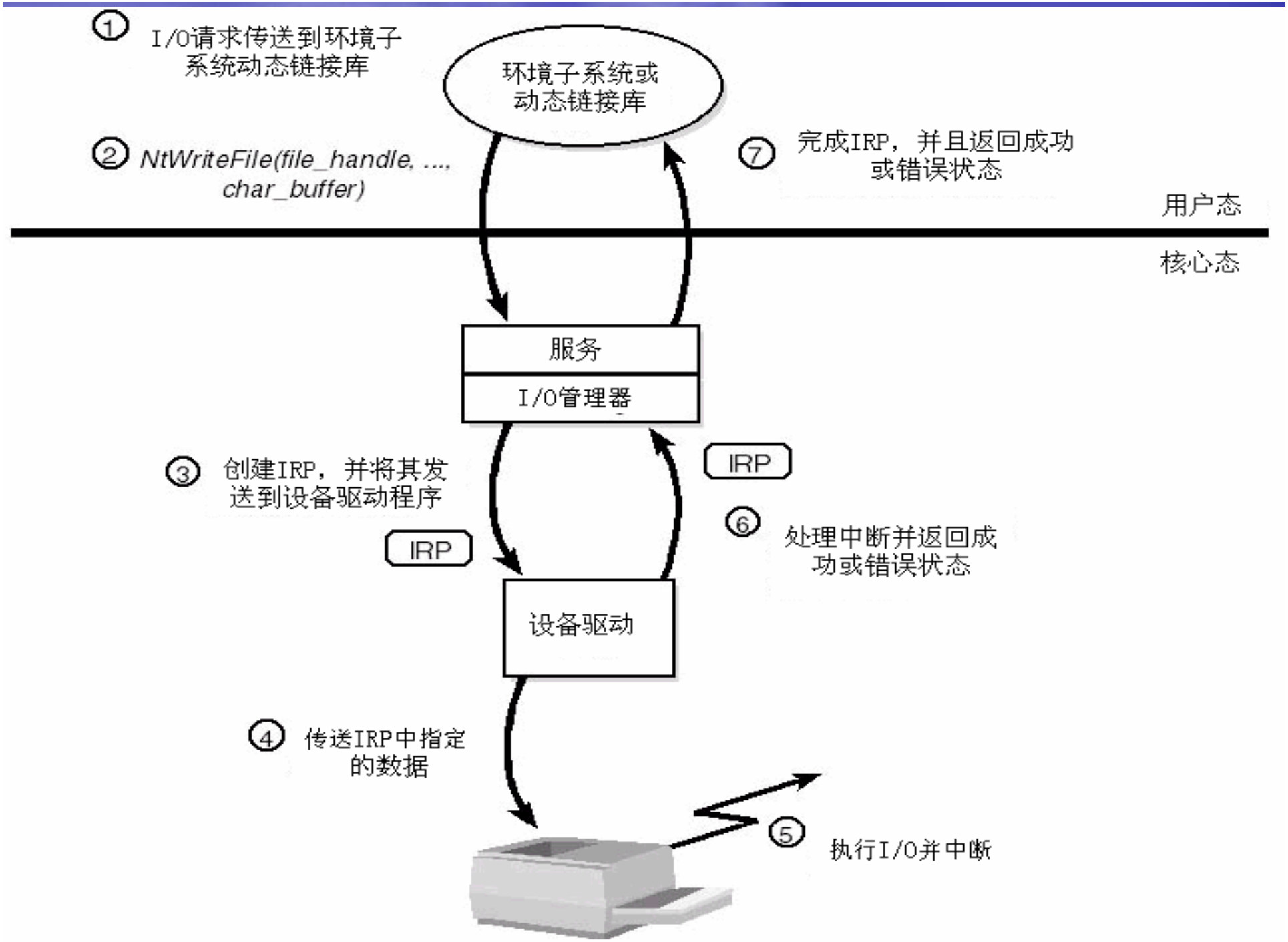
- 此外，还经常有如下部分
  - 一个或多个完成例程，通过一个较低层的驱动程序确定何时完成对一个IRP的处理。
  - 取消I/O例程，如果某个I/O操作可以被取消，驱动程序就可以定义一个或多个取消I/O例程。
  - 卸载例程，卸载例程释放任何驱动程序正在使用的系统资源，以使I/O管理器能从内存中删除它们。
  - 系统关闭通知例程，这个例程允许驱动程序在系统关闭的做清理工作。
  - 错误记录例程，当意外错误发生时，驱动程序的错误记录例程将记录发生的事情，并通知I/O管理器。I/O管理器把这个信息写入错误记录文件。

同步

- 必须同步执行它们对全局驱动程序数据的访问
  - 驱动程序的执行可以被高优先级的线程抢先，或时间片（或时间段）到时被中断，或被其他中断所中断
  - 在多处理器系统中，Windows 2000/XP能够同时在多个处理器上运行驱动程序代码

Windows2000/xp的I/O处理

- 对单层驱动程序的I/O请求处理
- I/O请求经过子系统DLL。
- 子系统DLL调用I/O管理器的服务。
- I/O管理器以IRP的形式给驱动程序（这里指设备驱动程序）发送请求。
- 驱动程序启动I/O操作。
- 在设备完成了操作并且中断CPU时，设备驱动程序服务于中断。
- I/O管理器完成I/O请求。


①
I/O请求传送到环境子系统动态链接库
环境子系统或动态链接库
②
NtWriteFile(file_handle, ..., char_buffer)
⑦
完成IRP，并且返回成功或错误状态
用户态
核心态
服务
I/O管理器
③
创建IRP，并将其发送到设备驱动程序
IRP
IRP
⑥
处理中断并返回成功或错误状态
设备驱动
④
传送IRP中指定的数据
⑤
执行I/O并中断

对多层驱动程序的I/O请求

- 在单层I/O处理的基础上变化而来
- I/O管理器调用顶层设备驱动程序
- 然后上层设备驱动程序调用低一级的驱动程序，形成I/O请求的转换、传递和嵌套。最终形成对设备的操作
- 分层驱动程序多用于几个设备的协作或者像文件、网络这样的复杂功能实体中

处理一个中断

- I/O设备中断发生
- 处理器将控制转交给内核陷阱处理程序
- 内核陷阱处理程序将在它的中断向量表中搜索定位用于设备的ISR
- ISR被首次调用时，它通常只获得设备状态，后响应设备的中断。然后它使一个DPC排队，清除中断并退出
- 过一段时间，在DPC例程被调用时，设备完成对中断的处理

完成I/O请求

- 设备驱动程序的DPC例程执行完以后，I/O处理结束之前还可以有一些工作做，这一阶段称为I/O完成阶段，它因I/O操作的不同而不同
- I/O管理器通过在线程中执行一个核心态的异步过程调用（APC）来完成这个操作
- APC例程将数据和返回的状态复制到最初调用者的地址空间，释放代表I/O操作的IRP，并将调用者的文件句柄设置为有信号状态
- 最初调用者从它们的等待状态中被唤醒