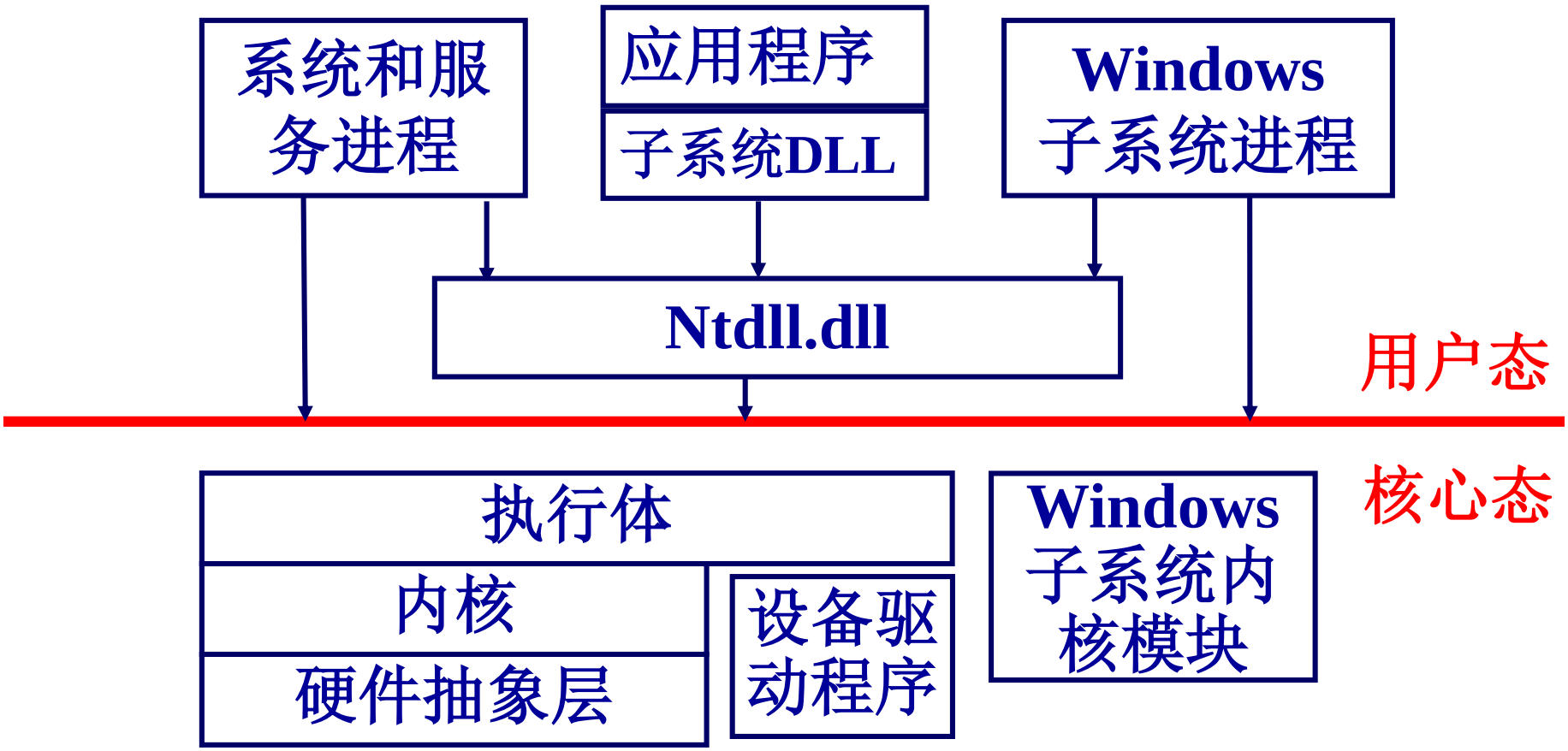
第14章　Windows操作系统模型

Windows 不仅可在单机上运行，也支持多机网络和对称多处理。界面友好、功能强、可扩充性、可靠性和兼容性好。

14.1 Windows 体系结构



Windows 系统文件

1. **Ntoskrnl.exe**：执行体和内核
2. **Hal.dll**：硬件抽象层
3. **Ntdll.dll**：对于内核提供的每一个系统服务，该DLL都提供一个以**Nt**作为前缀的存根函数。另外，还提供系统级支持函数。
4. **Win32k.sys**：Windows子系统的内核部分
5. **Kernel32.dll**，**Advapi32.dll**，**User32.dll**，**Gdi32.dll**：Windows子系统DLL
6. **NTFS.sys**：ntfs驱动程序
7. 设备驱动程序：可动态加载的模块(.sys)

系统进程

- idle进程：每个CPU一个线程。
- system进程：包含大多数内核系统线程

1. smss.exe：会话管理器。负责创建环境变量和启动csrss.exe和winlogon.exe。建立会话空间
2. csrss.exe：Windows子系统进程
3. winlogon.exe：用户登录进程
4. services.exe：系统服务管理器。系统有很多功能组件是以服务的方式实现的，如事件日志、任务调度器和各种网络组件等。该系统服务是一些特殊的进程。
5. svchost.exe：系统提供的通用服务宿主进程
6. lsass.exe：本地安全认证子系统进程
7. Explorer.exe：shell进程

环境子系统

- 有三个环境子系统：Windows、POSIX、OS/2。 Windows子系统是主子系统，其他两个是无用的。
- Windows 有一系列的系统调用，但从未公开过，而且每次发布都会改变。解决办法：定义一系列的函数调用，称作Windows API，不随新的Windows的发布而改变，向应用程序提供运行环境的调用接口。

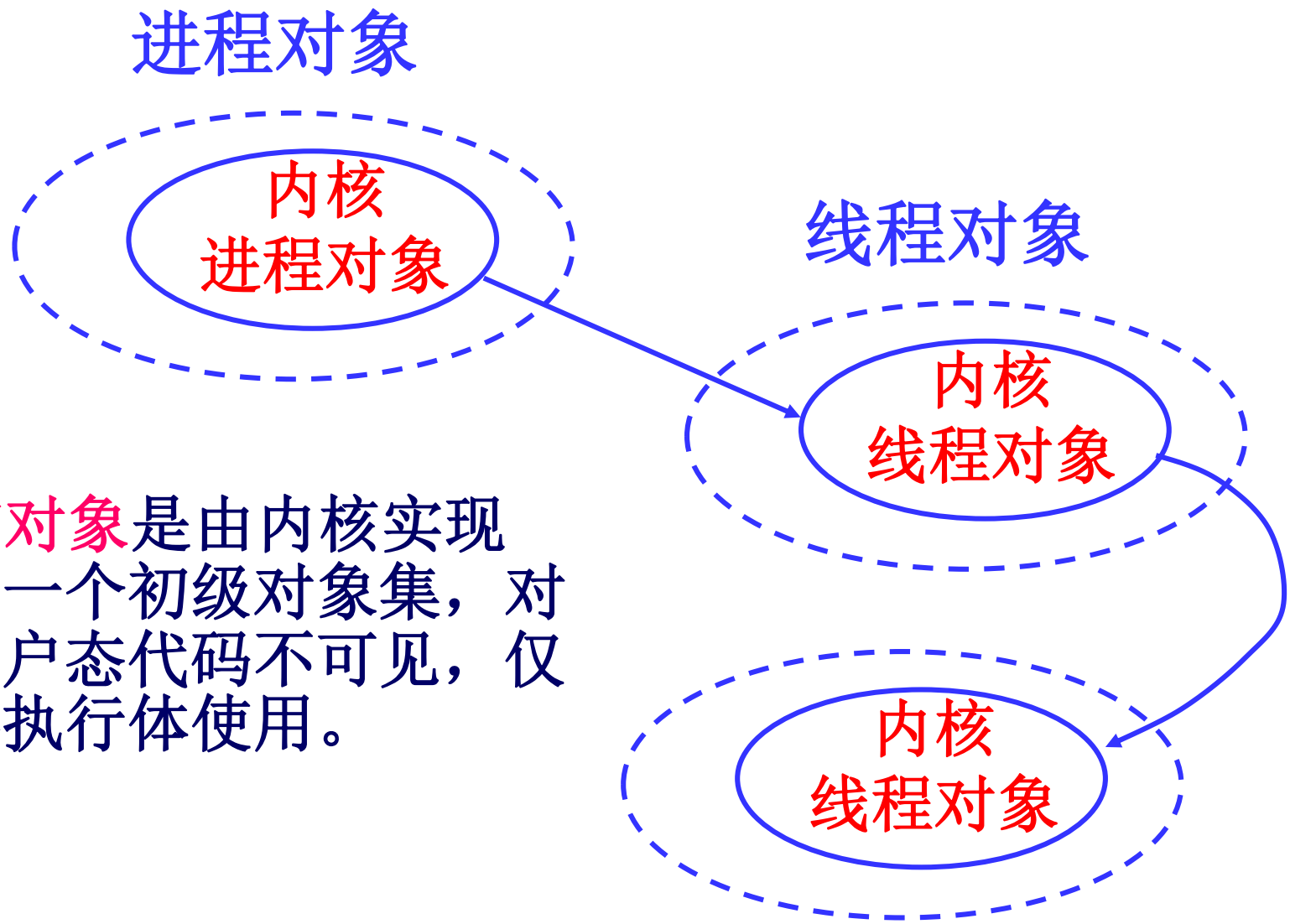
Windows子系统

- Windows子系统和系统内核一起构成了用户应用程序的执行环境。

1. 内核部分win32k.sys包括：窗口管理、图形设备接口。窗口管理负责收集和分发消息、控制窗口显示、管理屏幕输出。
2. 用户部分包括：子系统进程csrss.exe、子系统DLL。 csrss.exe负责控制台窗口的功能，以及创建进程和线程等。用户通过子系统DLL来发起系统调用。

Windows 内核结构

- **ntoskrnl.exe**：执行体、内核
- 内核层实现操作系统的基本机制，而所有的策略决定则留给执行体。惟一例外的策略决定是线程调度和分发。
- 执行体中的对象绝大多数封装了一个或者多个内核对象，并且通过某种方式（如对象句柄）暴露给应用程序。



内核对象是由内核实现的一个初级对象集，对用户态代码不可见，仅供执行体使用。

NT执行体

- 内部组件：进程和线程管理器、内存管理器、I/O管理器、缓存管理器、对象管理器、配置管理器、即插即用管理器、电源管理器、安全监视器、本地过程调用、一组运行时库函数、支持例程
- 每个组件都是一些过程的集合，组件之间没有固定的界限。
- 同层的各个组件之间可以互相调用。

内核

- 内核对象包括：控制对象、调度程序对象（dispatcher object）。
- 控制对象被用于控制内核的操作，但是不影响线程的调度。包括：APC、DPC、中断对象等。
- 调度程序对象：事件、互斥体、信号量、进程、线程、队列、门、定时器。实现同步，对象的状态会影响线程的调度。

硬件抽象层(HAL)

- 直接操纵硬件。
- **HAL.dll** 是一个可加载的核心态模块。
- **HAL**隐藏各种与硬件有关的细节。使内核、设备驱动程序和执行体免受特殊硬件平台差异的影响。 系统可移植性好。

14.2 Windows 特点

1. 核心态组件使用了面向对象的设计原则，但不是一个面向对象的操作系统。
2. 代码完全可重入，并可以被抢先。
3. 融合了分层模型和客户/服务器模型。客户进程和服务器进程通过执行体中提供的消息传递工具进行通信。

14.3 Windows的系统机制

① 陷阱调度。属于内核的功能。包括中断、DPC 、APC 、异常调度、系统服务调度。

② 执行体对象管理器。

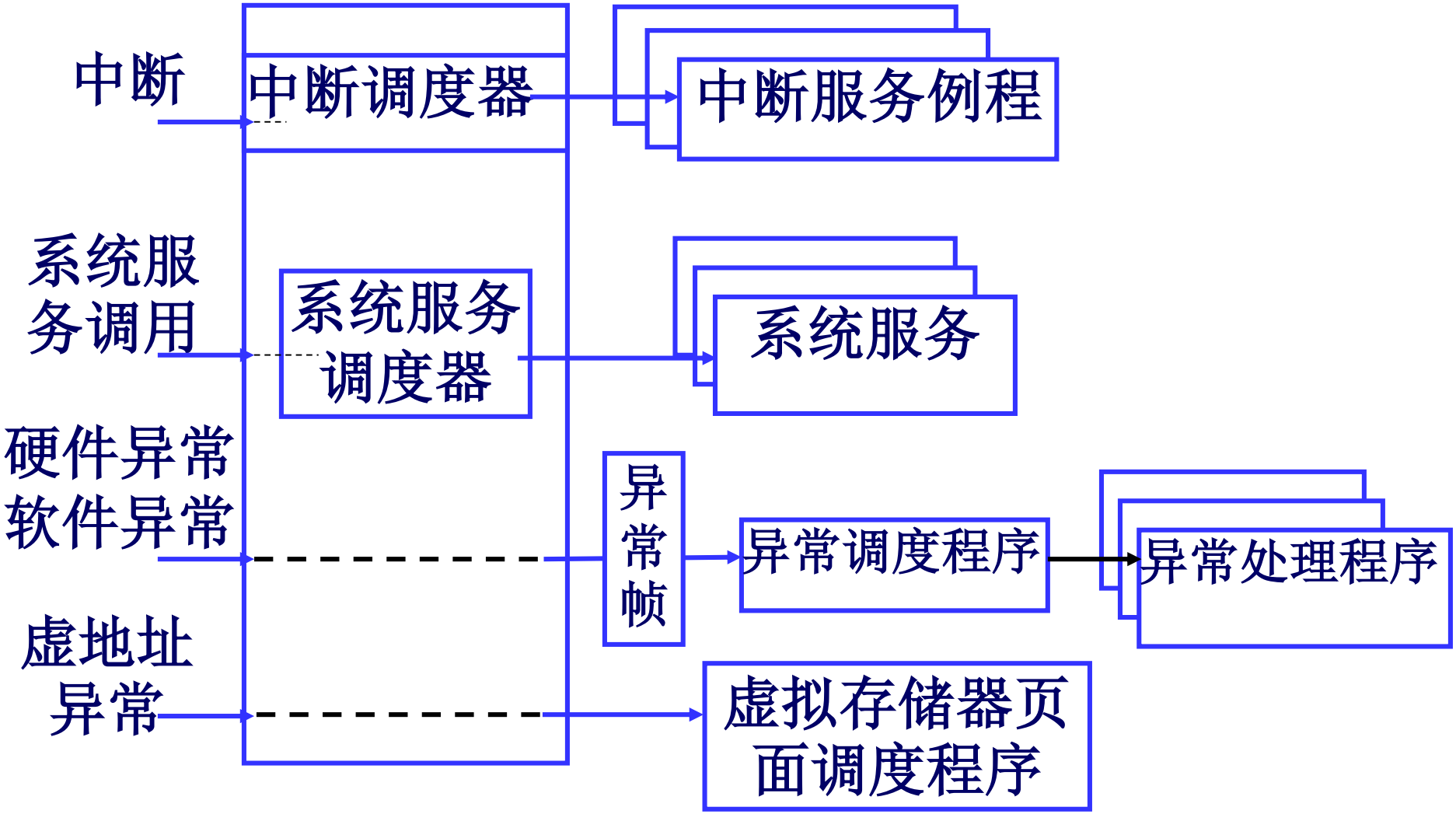
③ 同步。自旋锁、内核调度程序对象。

④ 本地过程调用LPC。

陷阱处理程序

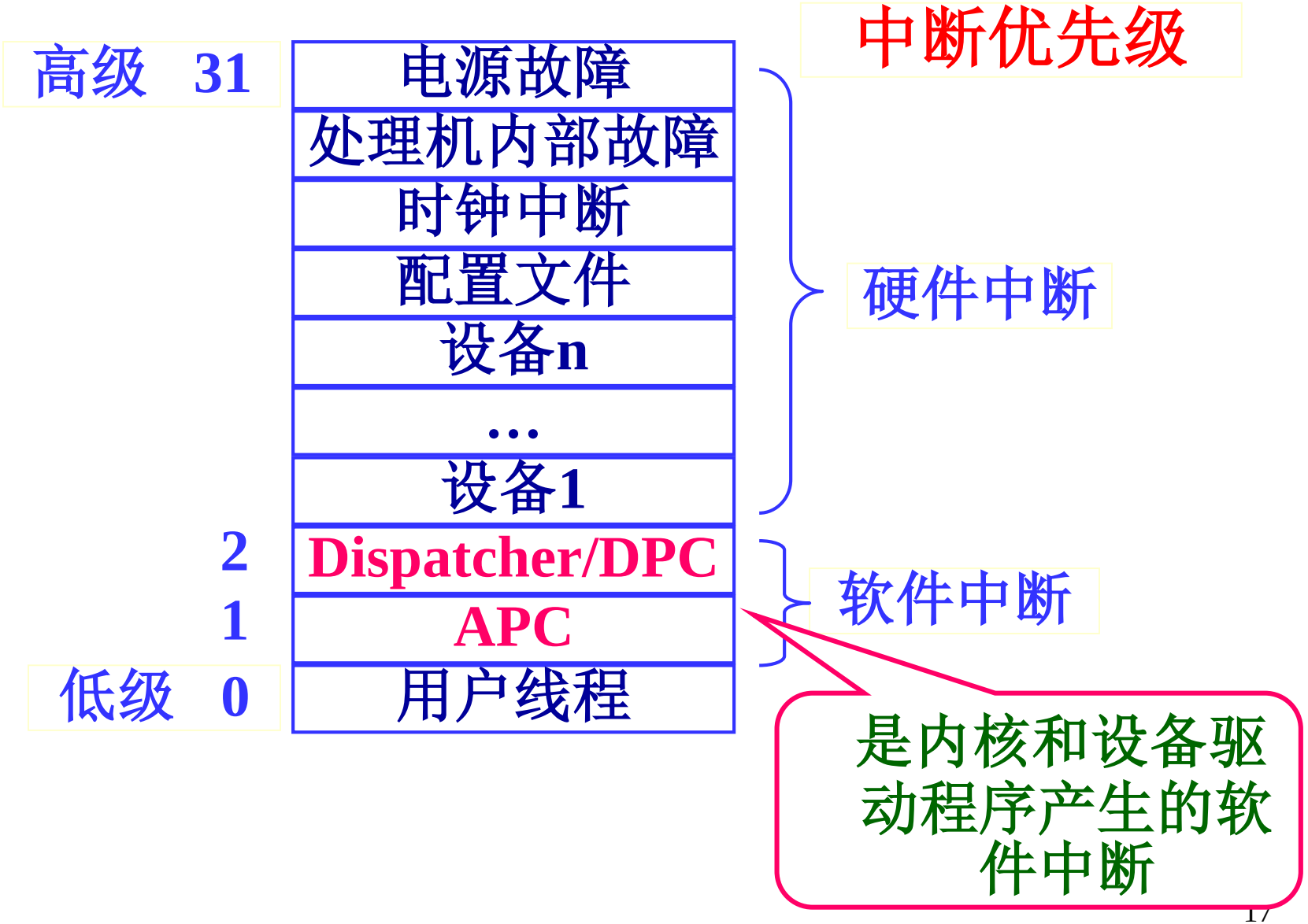
- 是操作系统处理意外事件的硬件机制。
- 当硬件或软件检测到异常或中断发生时，将暂停正在处理的事情，把控制转交给内核的陷阱处理程序。陷阱处理程序检测异常和中断的类型，并将控制转交给相应的处理程序。
- 中断是异步事件；异常是同步事件，系统服务调用被视为异常。

陷阱处理程序


中断
中断调度器
中断服务例程
系统服务调用
系统服务调度器
系统服务
硬件异常
软件异常
异常帧
异常调度程序
异常处理程序
虚地址异常
虚拟存储器页面调度程序

中断调度

- **Windows没有使用中断控制器的优先级，而是规定了一套软件中断优先级，称为中断请求级别IRQL。**
- **IRQL与线程的调度优先级不同。调度优先级是线程的属性，而IRQL是中断源的属性。**
- **运行于核心态的线程可以提高或降低它正在运行的处理机的IRQL，以屏蔽低级中断。**


中断优先级
高级 31
电源故障
处理机内部故障
时钟中断
配置文件
设备n
…
设备1
硬件中断
2
Dispatcher/DPC
1
APC
软件中断
低级 0
用户线程
是内核和设备驱动程序产生的软件中断

- **Windows 使用中断分派表(IDT, Interrupt Dispatch Table)查找处理特定中断的例程，以中断源的IRQL作为表的索引，找到中断处理例程的入口地址。**
- **在多处理机系统中，每个处理机都有一个独立的IDT，以便不同的处理机运行不同的中断服务例程ISR。**

延迟过程调用 DPC

- DPC被用来执行一些相对于当前高优先级的任务来说不那么紧急的任务。
- 有时内核在进行系统嵌套调用时，检测到应该进行重调度。为了保证调度的正确性，内核用DPC来延迟请求调度的产生。
- 硬件中断服务例程可以把一些相对不紧急的事情放到一个DPC对象中处理，从而缩短处理机停留在高IRQL的时间。

- DPC队列是一种机制，它能记住有哪些工作尚未处理。
- 当IRQL降低到DPC/Dispatcher级别以下时，DPC中断就产生。调度程序依次执行DPC队列中的每个例程，直至DPC队列为空。
- DPC队列是系统范围的。

异步过程调用 APC

- 为用户程序和系统代码提供了一种在特定用户线程环境中执行代码的方法。
- 如果需要从内核空间复制一个缓冲区到某一用户进程地址空间缓冲区，那么复制过程需要在用户进程上下文运行，这样页表才能包含内核缓冲区和用户缓冲区。
- 每个线程都有自己的**APC**队列。 **APC**队列也由内核管理。

设备中断处理例子

1. 当设备中断产生时，中断处理例程ISR被调用，它停留在设备IRQL上的时间通常只够捕获住该设备的状态并停止设备中断，之后把一个DPC放入队列中。
2. 当DPC例程被调用时，再完成此设备的中断处理。
3. 需要APC在某一进程地址空间中完成数据的复制。

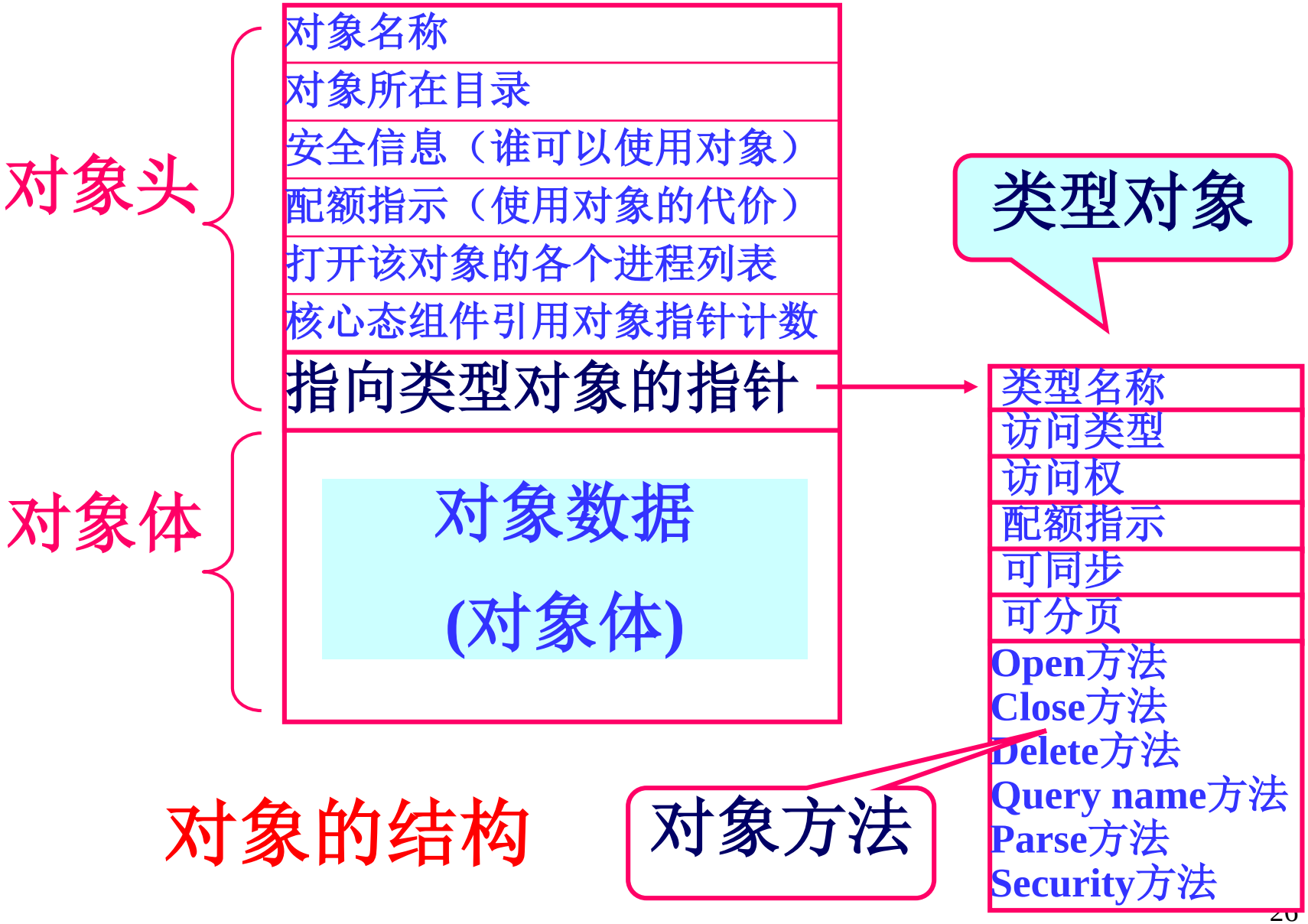
14.4 对象管理器

- 管理操作系统内的所有对象。
- 有两种类型的对象：执行体对象、内核对象
- 对象管理器的工作就是跟踪所有对象，便于对象访问的安全检测。

- 对象是一种内核数据结构。当系统刚启动时，并没有任何对象，对象是后来创建的。
- 空闲进程和系统进程中的对象被固化到**ntoskrnl.exe**文件中。
- 创建对象时，就从内核地址空间（执行体内存池）分配一块虚拟内存给对象。
- 执行体内存池使用快查表（类似伙伴系统）来管理小粒度的内存块。

14.4.1　对象结构

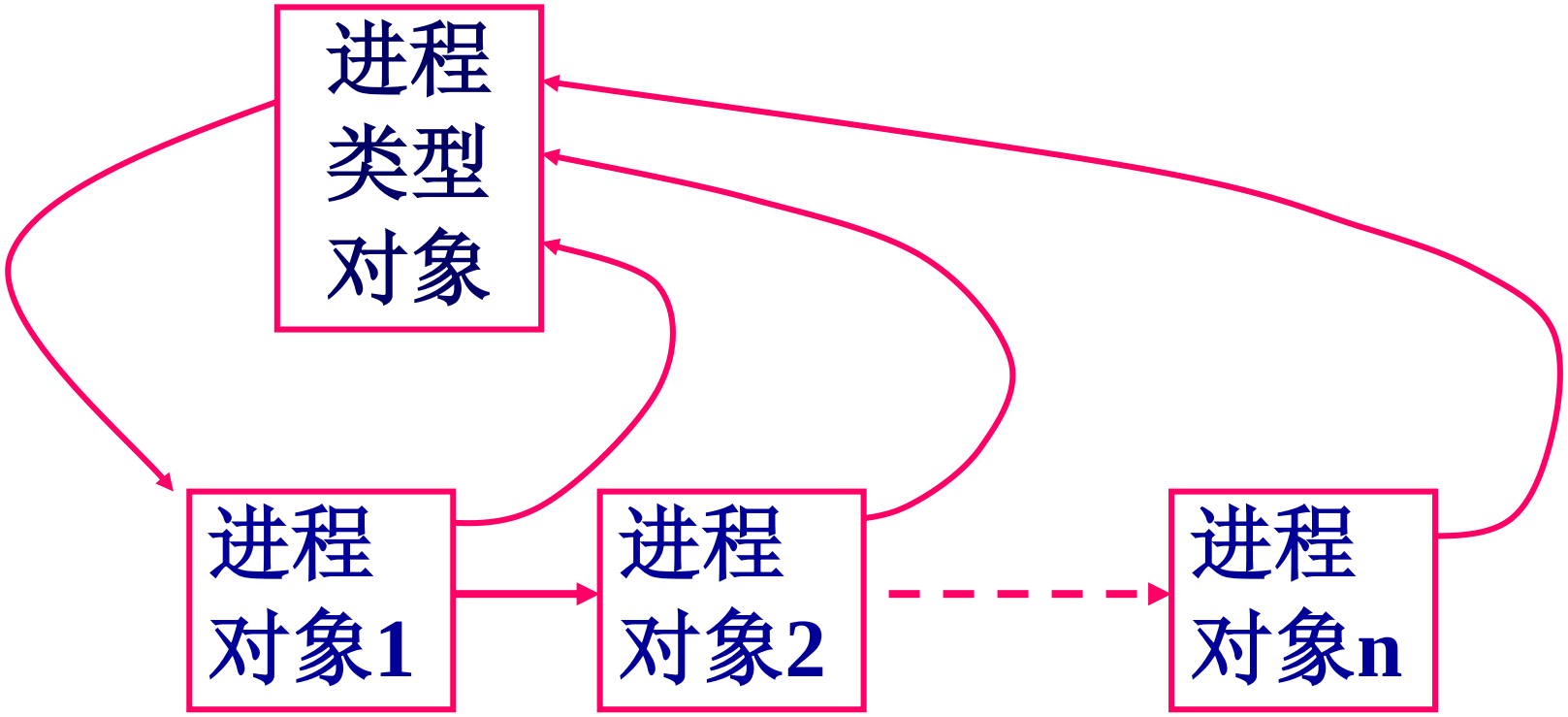
- 对象：对象头、对象体
- 对象管理器控制对象头，使用对象头中的数据管理对象。
- 各执行体组件控制其创建对象的对象体。


对象头
对象名称
对象所在目录
安全信息（谁可以使用对象）
配额指示（使用对象的代价）
打开该对象的各个进程列表
核心态组件引用对象指针计数
指向类型对象的指针
对象体
对象数据
(对象体)
类型对象
类型名称
访问类型
访问权
配额指示
可同步
可分页
Open方法
Close方法
Delete方法
Query name方法
Parse方法
Security方法
对象方法

对象的结构

类型对象

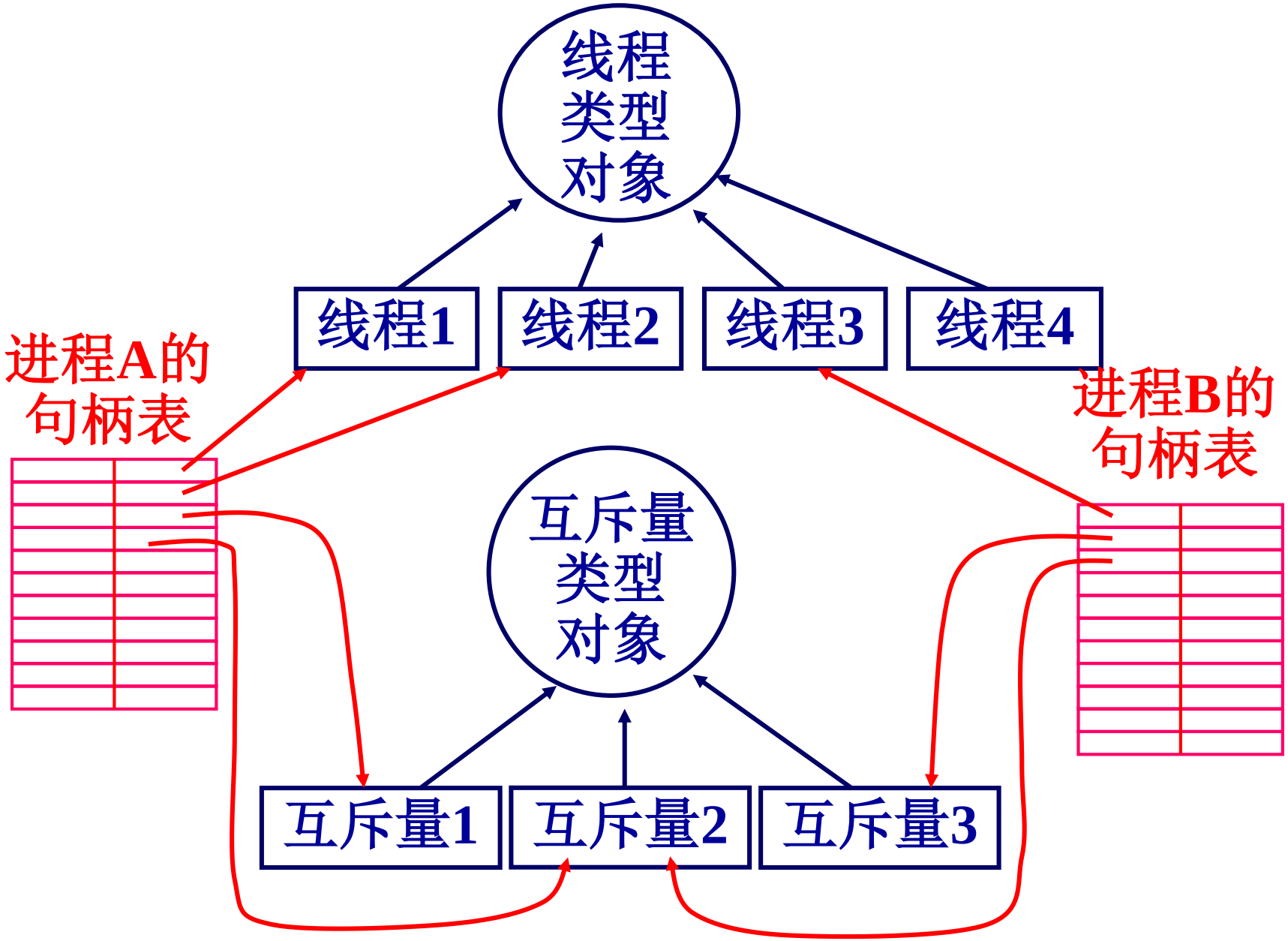
- 每个对象都有一个对象类型。
- 类型决定对象包含的数据和能够应用于对象的服务（方法）。
- 对象管理器存储特定类型的所有对象都有的公共属性。存储这些公共属性数据的对象叫做类型对象。
- 类型对象将所有同类对象链接在一起。



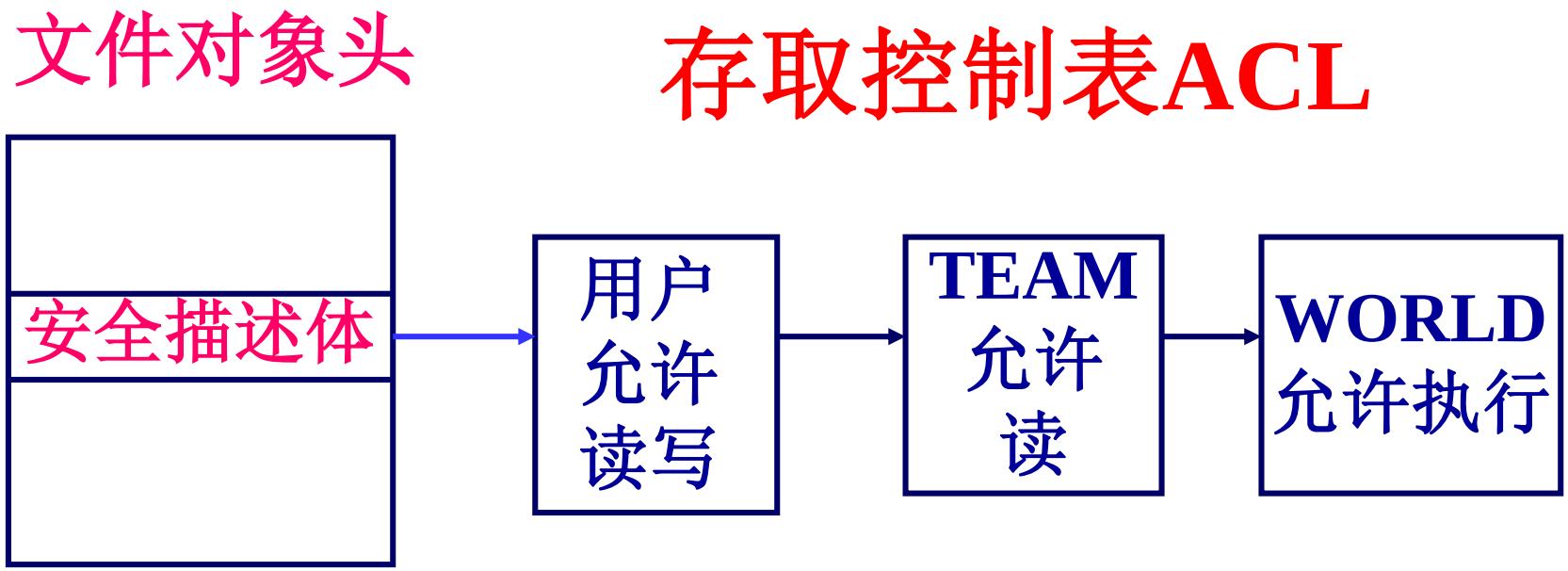
进程类型对象与进程对象之间的关系

14.4.2 管理对象

- 对象有对象名。
- 通过对象名创建/打开对象时，返回对象句柄。
- 每个进程将所有打开对象的指针放入进程打开对象句柄表中。进程打开对象句柄表存放在EPROCESS中


线程
类型
对象
线程1
线程2
线程3
线程4
进程A的
句柄表
进程B的
句柄表
互斥量
类型
对象
互斥量1
互斥量2
互斥量3

- 访问令牌：用来识别一个用户并辨别用户的特殊权限。是系统创建的控制用户正确使用系统资源的标识卡。
- 安全描述体用于控制进程和线程访问对象的存取控制表**ACL**



某文件的存取控制表

14.5 对象之间的同步

要正确地共享主存中的某些数据，多线程必须同步执行。

14.5.1 内核对象的同步

14.5.2 执行体对象的同步

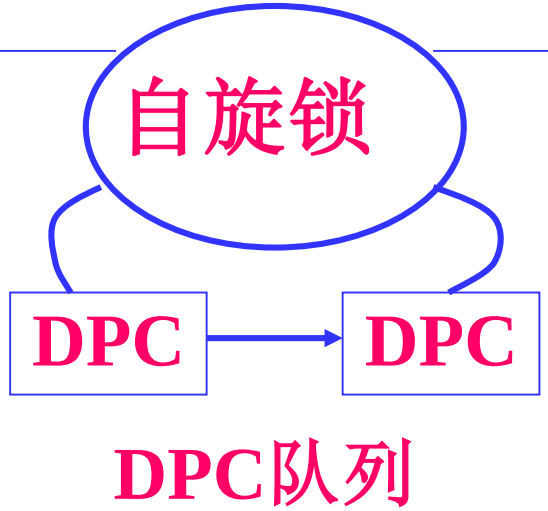
14.5.1 内核同步

- 互斥访问内核的临界区。
- 内核引入自旋锁（Spin lock）实现多处理机互斥机制。自旋锁是一个与公用数据结构有关的锁定机制。
- 在Intel处理机上，自旋锁是使用一条硬件支持的“测试与设置”指令来实现的。

用自旋锁保护全局数据结构DPC队列

处理机A

处理机B



Do尝试获得DPC队列的自旋锁Until成功

Begin—临界区

从队列中移出一个DPC

End

释放DPC队列的自旋锁

Do尝试获得DPC队列的自旋锁Until成功

Begin—临界区

向队列中加入一个DPC

End

释放DPC队列的自旋锁

- 如果自旋锁不空闲，内核将一直尝试得到锁。因为内核被保持在过渡状态“旋转”，直到获得锁，所以自旋锁由此得名。
- 在自旋锁上等待，实际是使处理机暂停。
- 拥有自旋锁的线程不被剥夺处理机。拥有自旋锁时，执行的指令数尽可能少，以提高系统效率。

14.5.2 执行体同步

- 内核以内核对象的形式给执行体提供其他的同步机构，称为“调度程序对象”。
- 每个同步对象有两种状态：“有信号”，“无信号”。线程可以等待一个或多个同步对象变为有信号状态，实现同步。

（1）等待调度程序对象

- 对象管理器提供了两个系统服务程序，使线程与调度程序对象同步：

  WaitForSingleObject( )

  WaitForMultipleObjects( )

（2）调度程序对象
被置为有信号状态的条件

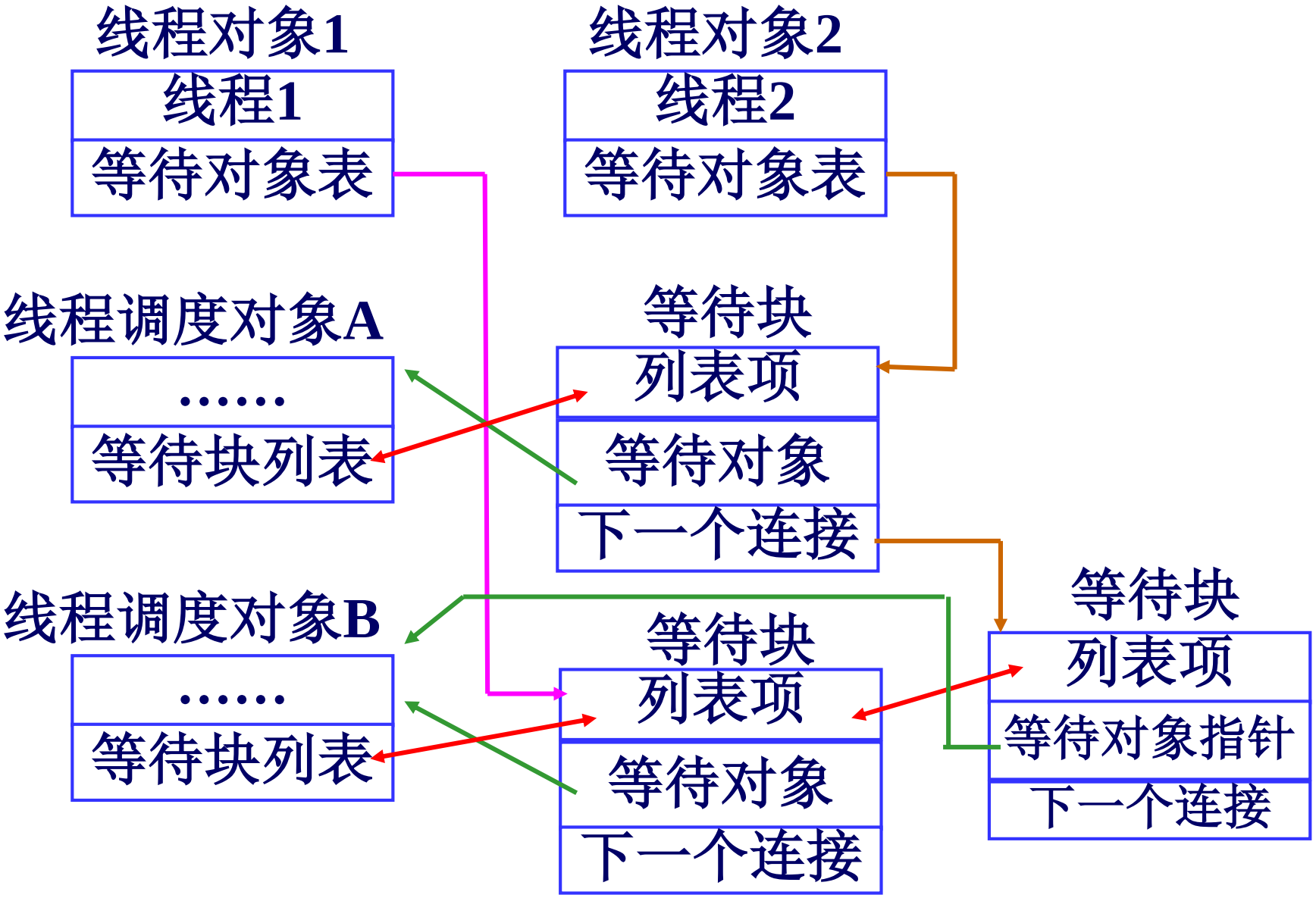
- 一个线程对象在其生存期内处于无信号状态；终止时，被置为有信号状态。
- 当进程的最后一个线程终止时，内核把进程对象设置为有信号状态。
- 定时器对象在一个确定的时间被设置为“发声”。当它的时间期满后，内核设置定时器对象为有信号状态。

（3） 同步涉及的数据结构

调度程序对象头和等待块是系统设置的两个数据结构，用于跟踪谁在等待，等待什么。

- ✓调度程序头：包含了对象所属类型、对象当前的状态和正在等待该同步对象的线程队列。
- ✓等待块：包含正在等待对象句柄的线程与被等待对象之间的联系。

- ✓每个处于等待状态的线程都有等待块列表，记录该线程正在等待的对象的情况。
- ✓每个调度程序头都有一个等待块列表头指针，将等待该对象的所有线程的等待块连接在一起。


线程对象1
线程1
等待对象表
线程对象2
线程2
等待对象表
线程调度对象A
……
等待块列表
等待块
列表项
等待对象
下一个连接
线程调度对象B
……
等待块列表
等待块
列表项
等待对象
下一个连接
等待块
列表项
等待对象指针
下一个连接

14.6 小结

1. 融合了分层和客户/服务器系统模型。
2. 系统机制：包括陷阱调度、执行体对象管理器、各种同步对象以及本地过程调用等。
3. 对象管理器。它将系统公共的资源作为对象来对待，以控制进程使用对象。有两种类型的对象：执行体对象和内核对象。
4. 对象之间的同步。