我们将研究许多关于抽象的内容，因为这是理解操作系统的关键。

上述观点是非常重要的，所以值得用不同的表述语句来再次叙述。怀着对设计Macintosh机器的工业设计师的尊重，作者这里不得不说，硬件是丑陋的。真实的处理器、内存条、磁盘和其他装置都是非常复杂的，对于那些为使用某个硬件而不得不编写软件的人们而言，他们使用的是困难、可怕、特殊和不一致的接口。有时这是由于需要兼容旧的硬件，有时是为了节省成本，但是，有时硬件设计师们并没有意识到（或在意）他们给软件设计带来了多大的麻烦。操作系统的一个主要任务是隐藏硬件，呈现给程序（以及程序员）良好、清晰、优雅、一致的抽象。如图1-2所示，操作系统将丑陋转变为美丽。

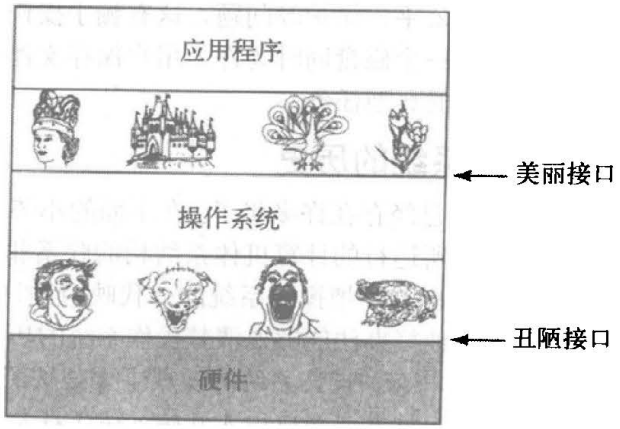

图1-2 操作系统将丑陋的硬件转变为美丽的抽象

需要指出，操作系统的实际客户是应用程序（当然是通过应用程序员）。它们直接与操作系统及其抽象打交道。相反，最终用户与用户接口所提供的抽象打交道，或者是命令行shell或者是图形接口。而用户接口的抽象可以与操作系统提供的抽象类似，但也不总是这样。为了更清晰地说明这一点，请读者考虑普通的Windows桌面以及面向行的命令提示符。两者都是运行在Windows操作系统上的程序，并使用了Windows提供的抽象，但是它们提供了非常不同的用户接口。类似地，运行Gnome或者KDE的Linux用户与直接在X Window系统（面向文本）顶部工作的Linux用户看到的是非常不同的界面，但是在这两种情形中，操作系统下面的抽象是相同的。

在本书中，我们将具体讨论提供给应用程序的抽象，不过很少涉及用户界面。尽管用户界面是一个巨大和重要的课题，但是它们毕竟只和操作系统的外围相关。

1.1.2 作为资源管理者的操作系统

把操作系统看作是向应用程序提供基本抽象的概念，是一种自顶向下的观点。按照另一种自底向上的观点，操作系统则用来管理一个复杂系统的各个部分。现代计算机包含处理器、存储器、时钟、磁盘、鼠标、网络接口、打印机以及许多其他设备。从这个角度看，操作系统的任务是在相互竞争的程序之间有序地控制对处理器、存储器以及其他I/O接口设备的分配。

现代操作系统允许同时运行多道程序。假设在一台计算机上运行的三个程序试图同时在同一台打印机上输出计算结果，那么开始的几行可能是程序1的输出，接着几行是程序2的输出，然后又是程序3的输出等，最终结果将是一团糟。采用将打印结果送到磁盘上缓冲区的方法，操作系统可以把潜在的混乱有序化。在一个程序结束后，操作系统可以将暂存在磁盘上的文件送到打印机输出，同时其他程序可以继续产生更多的输出结果，很明显，这些程序的输出还没有真正送至打印机。

当一个计算机（或网络）有多个用户时，管理和保护存储器、I/O设备以及其他资源的需求变得强烈起来，因为用户间可能会互相干扰。另外，用户通常不仅共享硬件，还要共享信息（文件、数据库等）。简而言之，操作系统的这一种观点认为，操作系统的主要任务是记录哪个程序在使用什么资源，对资源请求进行分配，评估使用代价，并且为不同的程序和用户调解互相冲突的资源请求。

资源管理包括用以下两种不同方式实现多路复用（共享）资源：在时间上复用和在空间上复用。当一种资源在时间上复用时，不同的程序或用户轮流使用它。先是第一个获得资源的使用，然后下一个，以此类推。例如，若在系统中只有一个CPU，而多个程序需要在该CPU上运行，操作系统则首先把该CPU分配给某一个程序，在它运行了足够长的时间之后，另一个程序得到CPU，然后是下一个，如此进行下去，最终，轮到第一个程序再次运行。至于资源是如何实现时间复用的——谁应该是下一个以及运行多长时间等——则是操作系统的任务。还有一个有关时间复用的例子是打印机的共享。当多个打印作业在一台打印机上排队等待打印时，必须决定将轮到打印的是哪个作业。

另一类复用是空间复用。每个客户都得到资源的一部分，从而取代了客户排队。例如，通常在若干运行程序之间分割内存，这样每一个运行程序都可同时入住内存（例如，为了轮流使用CPU）。假设有