4) 一个批处理作业的初始化。

启动操作系统时，通常会创建若干个进程。其中有些是前台进程，也就是同用户（人类）交互并且替他们完成工作的那些进程。其他的是后台进程，这些进程与特定的用户没有关系，相反，却具有某些专门的功能。例如，设计一个后台进程来接收发来的电子邮件，这个进程在一天的大部分时间都在睡眠，但是当电子邮件到达时就突然被唤醒了。也可以设计另一个后台进程来接收对该机器中Web页面的访问请求，在请求到达时唤醒该进程以便服务该请求。停留在后台处理诸如电子邮件、Web页面、新闻、打印之类活动的进程称为守护进程（daemon）。在大型系统中通常有很多守护进程。在UNIX中，可以用ps程序列出正在运行的进程；在Windows中，可使用任务管理器。

除了在启动阶段创建进程之外，新的进程也可以以后创建。一个正在运行的进程经常发出系统调用，以便创建一个或多个新进程协助其工作。在所要从事的工作可以容易地划分成若干相关的但没有相互作用的进程时，创建新的进程就特别有效果。例如，如果有大量的数据要通过网络调取并进行顺序处理，那么创建一个进程取数据，并把数据放入共享缓冲区中，而让第二个进程取走数据项并处理之，应该比较容易。在多处理机中，让每个进程在不同的CPU上运行会使整个作业运行得更快。

在交互式系统中，键入一个命令或者点（双）击一个图标就可以启动一个程序。这两个动作中的任何一个都会开始一个新的进程，并在其中运行所选择的程序。在基于命令行的UNIX系统中运行程序X，新的进程会从该进程接管开启它的窗口。在Microsoft Windows中，多数情形都是这样的，在一个进程开始时，它并没有窗口，但是它可以创建一个（或多个）窗口。在UNIX和Windows系统中，用户可以同时打开多个窗口，每个窗口都运行一个进程。通过鼠标用户可以选择一个窗口并且与该进程交互，例如，在需要时提供输入。

最后一种创建进程的情形仅在大型机的批处理系统中应用。用户在这种系统中（可能是远程地）提交批处理作业。在操作系统认为有资源可运行另一个作业时，它创建一个新的进程，并运行其输入队列中的下一个作业。

从技术上看，在所有这些情形中，新进程都是由于一个已存在的进程执行了一个用于创建进程的系统调用而创建的。这个进程可以是一个运行的用户进程，一个由键盘或鼠标启动的系统进程或者一个批处理管理进程。这个进程所做的工作是，执行一个用来创建新进程的系统调用。这个系统调用通知操作系统创建一个新进程，并且直接或间接地指定在该进程中运行的程序。

在UNIX系统中，只有一个系统调用可以用来创建新进程：fork。这个系统调用会创建一个与调用进程相同的副本。在调用了fork后，这两个进程（父进程和子进程）拥有相同的存储映像、同样的环境字符串和同样的打开文件。这就是全部情形。通常，子进程接着执行execve或一个类似的系统调用，以修改其存储映像并运行一个新的程序。例如，当一个用户在shell中键入命令sort时，shell就创建一个子进程，然后，这个子进程执行sort。之所以要安排两步建立进程，是为了在fork之后但在execve之前允许该子进程处理其文件描述符，这样可以完成对标准输入、标准输出和标准出错的重定向。

在Windows中，情形正相反，一个Win32函数调用CreateProcess既处理进程的创建，也负责把正确的程序装入新的进程。该调用有10个参数，其中包括要执行的程序、输入给该程序的命令行参数、各种安全属性、有关打开的文件是否继承的控制位、优先级信息、为该进程（若有的话）所需要创建的窗口规格以及指向一个结构的指针，在该结构中新创建进程的信息被返回给调用者。除了CreateProcess，Win32中有大约100个其他的函数用于处理进程的管理、同步以及相关的事务。

在UNIX和Windows中，进程创建之后，父进程和子进程有各自不同的地址空间。如果其中某个进程在其地址空间中修改了一个字，这个修改对其他进程而言是不可见的。在UNIX中，子进程的初始地址空间是父进程的一个副本，但是这里涉及两个不同的地址空间，不可写的内存区是共享的（某些UNIX的实现使程序正文在两者间共享，因为它不能被修改）。但是，对于一个新创建的进程而言，确实有可能共享其创建者的其他资源，诸如打开的文件等。在Windows中，从一开始父进程的地址空间和子进程的地址空间就是不同的。

2.1.3 进程的终止

进程在创建之后，它开始运行，完成其工作。但永恒是不存在的，进程也一样。迟早这个新的进程