每个过程可以自由调用其他过程，只要后者提供了前者所需要的一些有用的计算工作。这些可以不受限制彼此调用的成千个过程，常常导致出现一个笨拙和难于理解的系统。

在使用这种处理方式构造实际的目标程序时，首先编译所有单个的过程，或者编译包含过程的文件，然后通过系统链接程序将它们链接成单一的目标文件。依靠对信息的隐藏处理，不过在这里实际上是不存在的，每个过程对其他过程都是可见的（相反的构造中有模块或包，其中多数信息隐藏在模块之中，而且只能通过正式设计的入口点实现模块的外部调用）。

但是，即使在单体系统中，也可能有一些结构存在。可以将参数放置在良好定义的位置（如，栈），通过这种方式，向操作系统请求所能提供的服务（系统调用），然后执行一个陷阱指令。这个指令将机器从用户态切换到内核态并把控制传递给操作系统，如图1-17中第6步所示。然后，操作系统取出参数并且确定应该执行哪一个系统调用。随后，它在一个表格中检索，在该表格的k槽中存放着指向执行系统调用k过程的指针（图1-17中第7步）。

对于这类操作系统的基本结构，有着如下结构上的建议：

1) 需要一个主程序，用来处理服务过程请求。

2) 需要一套服务过程，用来执行系统调用。

3) 需要一套实用过程，用来辅助服务过程。

在该模型中，每一个系统调用都通过一个服务过程为其工作并运行之。要有一组实用程序来完成一些服务过程所需要用到的功能，如从用户程序取数据等。可将各种过程划分为一个三层的模型，如图1-24所示。

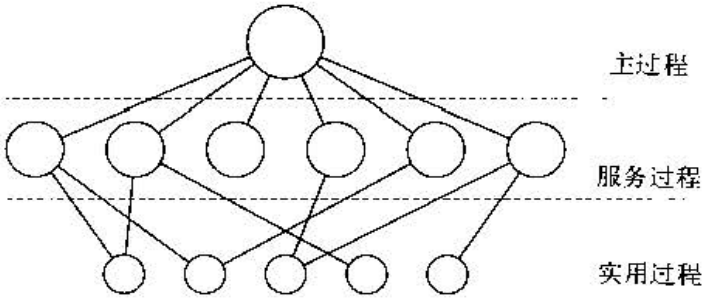

图1-24 简单的单体系统结构模型

除了在计算机初启时所装载的核心操作系统外，许多操作系统支持可装载的扩展，诸如I/O设备驱动和文件系统。这些部件可以按照需要载入。

1.7.2 层次式系统

把图1-24中的系统进一步通用化，就变成一个层次式结构的操作系统，它的上层软件都是在下一层软件的基础之上构建的。E. W. Dijkstra和他的学生在荷兰的Eindhoven技术学院所开发的THE系统（1968），是按此模型构造的第一个操作系统。THE系统是为荷兰的一种计算机，Electrologica X8，配备的一个简单的批处理系统，其内存只有32K个字，每字27位（二进制位在那时是很昂贵的）。

层号	功 能
5	操作员
4	用户程序
3	输入/输出管理
2	操作员-进程通信
1	存储器和磁鼓管理
0	处理器分配和多道程序设计

图1-25 THE操作系统的结构

该系统共分为六层，如图1-25所示。处理器分配在第0层中进行，当中断发生或定时器到期时，由该层进行进程切换。在第0层之上，系统由一些连续的进程所组成，编写这些进程时不用再考虑在单处理器上多进程运行的细节。也就是说，在第0层中提供了基本的CPU多道程序功能。

内存管理在第1层中进行，它分配进程的主存空间，当内存用完时则在一个512K字的磁鼓上保留进程的一部分（页面）。在第1层上，进程不用考虑它是在磁鼓上还是在内存中运行。第1层软件保证一旦需要访问某一页面时，该页面必定已在内存中。

第2层处理进程与操作员控制台（即用户）之间的通信。在这层的上部，可以认为每个进程都有自己的操作员控制台。第3层管理I/O设备和相关的信息流缓冲区。在第3层上，每个进程都与有良好特性的抽象I/O设备打交道，而不必考虑外部设备的物理细节。第4层是用户程序层。用户程序不用考虑进程、内存、控制台或I/O设备管理等细节。系统操作员进程位于第5层中。

在MULTICS系统中采用了更进一步的通用层次化概念。MULTICS由许多的同心环构造而成，而不