足够的内存可以存放多个程序，那么在内存中同时存放若干个程序的效率，比把所有内存都分给一个程序的效率要高得多，特别是，如果一个程序只需要整个内存的一小部分时，结果更是这样。当然，如此的做法会引起公平、保护等问题，这有赖于操作系统解决它们。有关空间复用的其他资源还有磁盘。在许多系统中，一个磁盘同时为许多用户保存文件。分配磁盘空间并记录谁正在使用哪个磁盘块，是操作系统资源管理的典型任务。

1.2 操作系统的历史

操作系统已经存在许多年了。在下面的小节中，我们将简要地分析一些操作系统历史上的重要之处。操作系统与其所运行的计算机体系结构的联系非常密切。我们将分析连续几代的计算机，看看它们的操作系统是什么样的。把操作系统的分代映射到计算机的分代上有些粗糙，但是这样做确实有某些作用，否则还没有其他好办法能够说清楚操作系统的历史。

下面给出的有关操作系统的发展主要是按照时间线索叙述的，且在时间上是有重叠的。每个发展并不是等到先前一种发展完成后才开始。存在着大量的重叠，不用说还存在有不少虚假的开始和终结时间。请读者把这里的文字叙述看成是一种指引，而不是盖棺论定。

第一台真正的数字计算机是英国数学家Charles Babbage （1792—1871）设计的。尽管Babbage花费了他几乎一生的时间和财产，试图建造他的“分析机”，但是他始终未能让机器正常的运转，因为它是一台纯机械的数字计算机，他所在时代的技术不能生产出他所需要的高精度轮子、齿轮和轮牙。毫无疑问，这台分析机没有操作系统。

有一段有趣的历史花絮，Babbage认识到他的分析机需要软件，所以他雇佣了一个名为Ada Lovelace的年轻妇女，作为世界上第一个程序员，而她是著名的英国诗人Lord Byron的女儿。程序设计语言Ada则是以她命名的。

1.2.1 第一代（1945～1955）：真空管和穿孔卡片

从Babbage失败之后一直到第二次世界大战，数字计算机的建造几乎没有什么进展，第二次世界大战刺激了有关计算机研究的爆炸性开展。Iowa州立大学的John Atanasoff教授和他的学生Clifford Berry建造了据认为是第一台可工作的数字计算机。该机器使用了300个真空管。大约在同时，Konrad Zuse在柏林用继电器构建了Z3计算机，英格兰布莱切利园的一个小组在1944年构建了Colossus，Howard Aiken在哈佛大学建造了Mark 1，宾夕法尼亚大学的William Mauchley和他的学生J.Presper Eckert建造了ENIAC。这些机器有的是二进制的，有的使用真空管，有的是可编程的，但是都非常原始，甚至需要花费数秒时间才能完成最简单的运算。

在那个早期年代里，同一个小组的人（通常是工程师们）设计、建造、编程、操作并维护一台机器。所有的程序设计是用纯粹的机器语言编写的，甚至更糟糕，需要通过将上千根电缆接到插件板上连接成电路，以便控制机器的基本功能。没有程序设计语言（甚至汇编语言也没有），操作系统则从来没有听说过。使用机器的一般方式是，程序员在墙上的机时表上预约一段时间，然后到机房中将他的插件板接到计算机里，在接下来的几小时里，期盼正在运行中的两万多个真空管不会烧坏。那时，所有的计算问题实际都只是简单的数字运算，如制作正弦、余弦以及对数表等。

到了20世纪50年代早期有了改进，出现了穿孔卡片，这时就可以将程序写在卡片上，然后读入计算机而不用插件板，但其他过程则依然如旧。

1.2.2 第二代（1955～1965）：晶体管和批处理系统

20世纪50年代晶体管的发明极大地改变了整个状况。计算机已经很可靠，厂商可以成批地生产并销售计算机给用户，用户可以指望计算机长时间运行，完成一些有用的工作。此时，设计人员、生产人员、操作人员、程序人员和维护人员之间第一次有了明确的分工。

这些机器，现在被称作大型机（mainframe），锁在有专用空调的房间中，由专业操作人员运行。只有少数大公司、重要的政府部门或大学才接受数百万美元的标价。要运行一个作业（job，即一个或一组程序），程序员首先将程序写在纸上（用FORTRAN语言或汇编语言），然后穿孔成卡片，再将卡片盒带到输入室，交给操作员，接着就喝咖啡直到输出完成。