为了把这个观点叙述得更透彻，我们考虑一些例子。早期计算机采用了硬连线指令集。这种指令可由硬件直接执行，且不能改变。然后出现了微程序设计（首先在IBM 360上大规模引入），其中的解释器执行软件中的指令。于是硬连线执行过时了，因为不够灵活。接着发明了RISC计算机，微程序设计（即解释执行）过时了，这是因为直接执行更快。而在通过Internet发送并且到达时才解释的Java小程序形式中，我们又看到了解释执行的复苏。执行速度并不总是关键因素，但由于网络的时间延迟是如此之大，以至于它成了主要因素。这样，钟摆在直接执行和解释之间已经晃动了好几个周期，也许在未来还会再次晃动。

1. 大型内存

现在来分析硬件的某些历史发展过程，并看看硬件是如何重复地影响软件的。第一代大型机内存有限。在1959年至1964年之间，称为“山寨王”的IBM 7090或7094满载也只有128KB多的内存。该机器多数用汇编语言编程，为了节省内存，其操作系统用汇编语言编写。

随着时代的前进，在汇编语言宣告过时时，FORTRAN和COBOL一类语言的编译器已经足够好了。但是在第一个商用小型计算机（PDP-1）发布时，却只有4096个18位字的内存，而且令人吃惊的是，汇编语言又回来了。最终，小型计算机获得了更多的内存，而且高级语言也在小型机上盛行起来。

在20世纪80年代早期，微型计算机出现时，第一批机器只有4 KB内存，汇编语言又复活了。嵌入式计算机经常使用和微型计算机一样的CPU芯片（8080、Z80、后来的8086）而且一开始也使用汇编编程。现在，它们的后代，个人计算机拥有大量的内存，使用C、C++、Java和其他高级语言编程。智能卡正在走着类似的发展道路，而且除了确定的大小之外，智能卡通常使用Java解释器，解释执行Java程序，而不是将Java编译成为智能卡的机器语言。

2. 保护硬件

早期的IBM 7090/7094一类大型机，没有保护硬件，所以这些机器一次只运行一个程序。一个有问题的程序就可能毁掉操作系统，并且很容易使机器崩溃。在IBM 360发布时，提供了保护硬件的原型，这些机器可以在内存中同时保持若干程序，并让它们轮流运行（多道程序处理）。于是单道程序处理宣告过时。

至少是到了第一个小型计算机出现时——还没有保护硬件——所以多道程序处理也不可能有。尽管PDP-1和PDP-8没有保护硬件，但是PDP-11型机器有了保护硬件，这一特点导致了多道程序处理的应用，并且最终导致UNIX操作系统的诞生。

在建造第一代微型计算机时，使用了Intel 8080 CPU芯片，但是没有保护硬件，这样我们又回到了单道程序处理。直到Intel 80286才增加了保护硬件，于是有了多道程序处理。直到现在，许多嵌入式系统仍旧没有保护硬件，而且只运行单个程序。

现在来考察操作系统。第一代大型机原本没有保护硬件，也不支持多道程序处理，所以这些机器只运行简单的操作系统，一次手工只能装载一个程序。后来，大型机有了保护硬件，操作系统可以同时支持运行多个程序，接着系统拥有了全功能的分时能力。

在小型计算机刚出现时，也没有保护硬件，一次只运行一个手工装载的程序。逐渐地，小型机有了保护硬件，有了同时运行两个或更多程序的能力。第一代微型计算机也只有一次运行一个程序的能力，但是随后具有了多道程序的能力。掌上计算机和智能卡也走着类似的发展之路。

在所有这些案例中，软件的发展是受制于技术的。例如，第一代微型计算机有约4KB内存，没有保护硬件。高级语言和多道程序处理对于这种小系统而言，无法获得支持。随着微型计算机演化成为现代个人计算机，拥有了必要的硬件，从而有了必须的软件处理以支持多种先进的功能。这种演化过程看来还要继续多年。其他的领域也有类似的这种轮回现象，但是在计算机行业中，这种轮回现象似乎变化得更快。

3. 硬盘

早期大型机主要是基于磁带的。机器从磁带上读入程序、编译、运行，并把结果写到另一个磁带上。那时没有磁盘也没有文件系统的概念。在IBM于1956年引入第一个磁盘——RAMAC（RAndoM ACcess）之后，事情开始变化。这个磁盘占据4平方米空间，可以存储5百万7位长的字符，这足够存储一张中等分辨率的数字照片。但是其年租金高达35 000美元，比存储占据同样空间数量的胶卷还要贵。不过这个磁盘的价格终于还是下降了，并开始出现了原始的文件系统。