1.5.8 Recycling of Concepts

Computer science, like many fields, is largely technology driven. The reason the ancient Romans lacked cars is not that they liked walking so much. It is because they did not know how to build cars. Personal computers exist *not* because millions of people had some long pent-up desire to own a computer, but because it is now possible to manufacture them cheaply. We often forget how much technology affects our view of systems and it is worth reflecting on this point from time to time.

In particular, it frequently happens that a change in technology renders some idea obsolete and it quickly vanishes. However, another change in technology could revive it again. This is especially true when the change has to do with the relative performance of different parts of the system. For example, when CPUs became much faster than memories, caches became important to speed up the "slow" memory. If new memory technology some day makes memories much faster than CPUs, caches will vanish. And if a new CPU technology makes them faster than memories again, caches will reappear. In biology, extinction is forever, but in computer science, it is sometimes only for a few years.

As a consequence of this impermanence, in this book we will from time to time look at "obsolete" concepts, that is, ideas that are not optimal with current technology. However, changes in the technology may bring back some of the so-called "obsolete concepts." For this reason, it is important to understand why a concept is obsolete and what changes in the environment might bring it back again.

To make this point clearer, let us consider a few examples. Early computers had hardwired instruction sets. The instructions were executed directly by hardware and could not be changed. Then came microprogramming, in which an underlying interpreter carried out the instructions in software. Hardwired execution became obsolete. Then RISC computers were invented, and microprogramming (i.e., interpreted execution) became obsolete because direct execution was faster. Now we are seeing the resurgence of interpretation in the form of Java applets that are sent over the Internet and interpreted upon arrival. Execution speed is not always crucial because network delays are so great that they tend to dominate. But that could change, too, some day.

Early operating systems allocated files on the disk by just placing them in contiguous sectors, one after another. Although this scheme was easy to implement, it was not flexible because when a file grew, there was not enough room to store it any more. Thus the concept of contiguously allocated files was discarded as obsolete. Until CD-ROMs came around. There the problem of growing files did not exist. All of a sudden, the simplicity of contiguous file allocation was seen as a great idea and CD-ROM file systems are now based on it.

As our final idea, consider dynamic linking. The MULTICS system was designed to run day and night without ever stopping. To fix bugs in software, it

was necessary to have a way to replace library procedures while they were being used. The concept of dynamic linking was invented for this purpose. After MULTICS died, the concept was forgotten for a while. However, it was rediscovered when modern operating systems needed a way to allow many programs to share the same library procedures without having their own private copies (because graphics libraries had grown so large). Most systems now support some form of dynamic linking once again. The list goes on, but these examples should make the point: an idea that is obsolete today may be the star of the party tomorrow.

Technology is not the only factor that drives systems and software. Economics plays a big role too. In the 1960s and 1970s, most terminals were mechanical printing terminals or 25×80 character-oriented CRTs rather than bitmap graphics terminals. This choice was not a question of technology. Bit-map graphics terminals were in use before 1960. It is just that they cost many tens of thousands of dollars each. Only when the price came down enormously could people (other than the military) think of dedicating one terminal to an individual user.