Patent Reform Is Not Enough[1]

RICHARD M. STALLMAN

When people first learn about the problem of software patents, their attention is often drawn to the egregious examples: patents that cover techniques already widely known. These techniques include sorting a collection of formulae so that no variable is used before it is calculated (called "natural order recalculation" in spreadsheets), and the use of exclusive-or to modify the contents of a bit-map display.

Focusing on these examples can lead some people to ignore the rest of the problem. They are attracted to the position that the patent system is basically correct and needs only "reforms" to carry out its own rules properly.

But would correct implementation really solve the problem of software patents? Let's consider an example.

In April 1991, software developer Ross Williams began publishing a series of data compression programs using new algorithms of his own devising. Their superior speed and compression quality soon attracted users.

The following September, when the FSF was about a week away from releasing one of them as the new choice for compressing our distribution files, use of these programs in the United States was halted by a newly issued patent, number 5,049,881.

Under the current patent rules, whether the public is allowed to use these programs (i.e., whether the patent is invalid) depends on whether there is "prior art": whether the basic idea was published before the patent application, which was on June 18, 1990. Williams' publication in April 1991 came after that date, so it does not count.

A student described a similar algorithm in 1988–1989 in a class paper at the University of San Francisco, but the paper was not published. So it does not count as prior art under the current rules.

Reforms to make the patent system work "properly" would be no help here. Under the rules of the patent system, this patent seems valid. There is no prior art for it. It is not close to obvious, as the patent system interprets the term. (Like most patents, it is neither worldshaking nor trivial, but somewhere in between.) The fault is in the rules themselves, not their execution.

In the US legal system, patents are intended as a bargain between society and individuals; society is supposed to gain through the disclosure of techniques that would otherwise never be available. It is clear that society has gained nothing by issuing patent number 5,049,881.

Under current rules, our ability to use Williams's programs depends on whether anyone happened to publish the same idea before June 18, 1990. That is to say, it depends on luck. This system is good for promoting the practice of law, but not progress in software.

Teaching the Patent Office to look at more of the existing prior art might prevent some outrageous mistakes. It will not cure the greater problem, which is the patenting of every *new* wrinkle in the use of computers, like the one that Williams and others independently developed.

[1] Originally published in GNU's Bulletin, vol. 1 no. 13.

This will turn software into a quagmire. Even an innovative program typically uses dozens of not-quite-new techniques and features, each of which might have been patented. Our ability to use each wrinkle will depend on luck, and if we are unlucky half the time, few programs will escape infringing a large number of patents. Navigating the maze of patents will be harder than writing software. As *The Economist* says, software patents are simply bad for business.

If you'd like to do something, the easiest thing to do is to join the League for Programming Freedom.