

SUBMISSION ON PATENTABLE SUBJECT MATTER -- SOFTWARE

As a software engineer, I would like to specifically comment on the patenting of software. Software should NOT be patentable because software patents stifle innovation. A clear and explicit exclusion for software would add clarity to many cases. Other mechanisms to limit patentable subject matter may also be appropriate.

Software is about building systems, not individual pieces. There are vast numbers of pieces in a software system, and there is a large amount of re-invention. Most ideas are relatively simple, and can be reinvented with a relatively small amount of effort.

For this reason, patents tend to protect questions rather than answers. Patenters think about what problems we might need to solve in ten years time. Then, ten years later, other people solve these problems easily independently. Only to find that someone else had a patent on it. This in no way improves innovation.

The patenting process is also severely broken. Most patents are for trivial inventions. They are described in obscure and arcane language that takes considerable effort to decipher. It is impractical for engineers to read the millions of software patents to look for possible infringements. And any such search would result in numerous possible conflicts, and possible triple damages (in the USA, but that affects patent searches here). Litigation over any aspect of patents is also very expensive and unpredictable. There is no obvious way to address these issues. And patent terms are far too long in the dynamic software industry.

In practice large companies mainly build a software patent portfolio so that they can sign non-aggression pacts with other large companies. This is very wasteful. On the other hand, the patent software trolls do not develop software but simply wait for larger companies to reinvent their patented ideas.

Fortunately, most patents are never enforced. But no one would ever write a line of code if they knew all the patents that could potentially attack them. I have personally worked on innovative projects that have been cancelled due to fear of unknown patents. On the other hand I worked for an Australian company that flourished because a US patent was not applicable here (the infamous RSA patent).

Good techniques are often avoided due to patent issues. For example, the weaker RSA cryptographic algorithm is used instead of the more secure Elliptic Curve public keys because of patents. Awkward key agreement algorithms such as JPAKE are developed solely to avoid patents on PAKE. Engineers do not like dealing with patents, and will go a long way to avoid them. Licensing numerous patents is just too hard, and they resent paying the tax.

There can be little doubt that the algorithms would have been independently invented when needed. Some like PAKE are obvious extensions to previous schemes. The patent holders simply got to the question first, and then blocked others from developing the obvious answer.

It is also clear that other motivations are sufficient to develop technology. For example, the more complex JPAKE algorithm was developed specifically so that it could be placed in the public domain, avoiding patent issues with PAKE. Likewise the Vorbis video Codec was developed to avoid patent issues with MPeg standards. It is much more difficult to develop solutions that do not use the obvious (and hence patented) approach. Interestingly, corporations are reluctant to use Vorbis because of potential patent issues, even though the developers of Vorbis explicitly renounce any patent rights. This is because patent law suites are extremely unpredictable. Many expensive and often frivolous suites have been processed for MPeg, whereas Vorbis has not been tested in court.

Much software today is open source, i.e. given away without any revenue. This means that it cannot license patents. That highlights the difference between software and other endeavors. Software is intangible. Further, patent revenue is a very small contributor to the funding of software academic research.

Software patentability (in the USA) is relatively recent. The software industry was very innovative before their introduction. There is no tangible economic evidence that software patents improve innovation.

There may well be other areas that should also not be patented, but I am only an expert in software. Any Patents are bureaucratic burden. The onus needs to be on the patent/legal industry to show that patents really encourage innovation, and not on software engineers to show why they are harmful.

I would refer you to the USA Trade commission report on software patents, summarized here:
<http://eupat.ffii.org/papri/ftc03/index.en.html>. It takes an economic rather than a legal perspective.

As an engineer, I am focused software, not legal issues. People that do focus on software legal issues tend to be part of the patent industry, and so will naturally be very keen to increase its scope and potential for litigation. Likewise, if you write to large companies about patents you will get replies from their patent departments. I would therefor suggest that your sample of responses may be very biased.

For these reasons this review has just been noticed by the anti software patent community.

"If people had understood how (software) patents would be granted when most of today's ideas were invented and had taken out patents, the industry would be at a complete standstill today." Bill Gates, Microsoft. Many other quotes can be found at
<http://eupat.ffii.org/vreji/quotes/index.en.html>

Yours Sincerely,

Dr Anthony Berglas
Southern Cross Software Queensland.