

Silicon Economics, Inc.

940 Lundy Lane  
Los Altos, CA 94024  
(650) 941-4083

April 30, 2009

Enlarged Board of Appeal  
European Patent Office  
80331 München

Case G 3/08: President's Referral relating to  
Patentability in the Field of Computing

Dear Sirs:

Thank you for the opportunity to submit this *amicus curiae*. I am a Silicon Valley-based American economist, focused on embedding economic thinking in computer software. I am founding what I hope will be a global company in which computer software patents will be strategically central. I have dealt with European companies, one of which acquired the first company that I founded. Before answering your four questions, I would like to address the matter of computer software patents, from both a philosophical and an economic perspective.

Some argue that computer software is too abstract to warrant being patentable subject matter. That argument is easily refuted by considering that computer software transforms a general-purpose computer into a specialized computer, to generate useful and concrete results. Since a device that generates useful and concrete results is patentable subject matter, computer software, coupled with a general-purpose computer, should also be considered patentable subject matter, with no issue of abstraction.

In my estimation, the matter of computer software patents has been needlessly confused by "inalienable rights," "state of nature," and "mathematical" considerations. Granted, everyone has certain inalienable rights, including the rights to think as one desires, to think mathematical thoughts, and to attempt to survive. Computer software patents, assuming that they are tied to a computer, do not encroach upon such inalienable rights.

If Defoe's Robinson Crusoe were on his island today, his inalienable rights would not be encroached by the existence of computer software patents. He could still think what he wanted, think mathematical thoughts, and attempt to survive. He would have no inalienable right to execute software ideas or concepts on a computer, simply because his "state of nature" would not provide him with the required computer. Likewise, the "state of nature" for the modern individual does not provide a computer; therefore the modern individual has no inalienable right to execute software ideas or concepts on a computer. In short, if having a computer is not an inalienable right, then software that executes on a computer is not an inalienable right.

The matter of computer software patents has also been needlessly confused by "business methods" considerations, possibly clouded by "inalienable rights" issues. This confusion can be resolved by making a distinction between two types of business-method patents – pure and improvement. One could legitimately argue that freedom of contract is an inalienable right, that each of today's 6.8 billion human beings has the fundamental right to contract with others to exchange valuably held rights and legal consideration, that the market economy is predicated upon such freedom, and that business-method patents undesirably interfere with the ability of people to contract and of the market to function. Accordingly, a patent application regarding the simple essential business function of buying low and selling high should not be allowed, because of blocking market operations, particularly competition, and because of denying people their rights to contract. I term this type of patent a pure business-method patent. In a subtle but distinct contrast, an improvement business-method patent should be allowed for advancing technology without blocking market function that could reasonably have occurred prior to the patent and without denying people their rights to contract.

As an example, a businessperson's new operation of buying flowers in one country and selling them in another country is a pure business-method patent matter, not allowable, for blocking possible-prior market operations, for denying people their contracting rights. The businessperson is essentially a competitor in a market that he might have created, not

entitled to any patent, monopoly power, or protection. In contrast, an inventor's/entrepreneur's custom improvement regarding refrigeration that makes possible buying flowers in one country and selling them in another country is an improvement to the businessperson's operation, allowable business-method subject matter, for not blocking prior market operations, for allowing people to retain their prior contracting rights. The inventor is developing and advancing technology, whatever the technology might be, and entitled to have the technology considered patentable subject matter. Clearly, if the invention regards only improved refrigeration, it is patentable subject matter. But it should also be patentable subject matter if it improves market functioning, such as would occur if the technology regards computer software for new pricing or buyer-seller matching which does not block anyone from using prior contracting rights and capabilities. Whether the invention employs computer software has no bearing on whether it is a pure, or an improvement, business-method matter. I would expect most business-method computer software matters today to be classified as improvement, rather than pure, matters.

Many assert that computer software should not be patentable, arguing that software is different from other technologies, but possibly simply promoting their own self-interest. Most large companies engage in only incremental innovation, preferring to focus on predictability and competing via marketing, size, etc. Patents are one of the few vehicles that allow new companies and individuals to challenge the status quo, and to trigger the entrepreneurial "creative destruction" that economist Joseph Schumpeter deemed necessary for societies and capitalism to advance. Today, the major global software companies have little to gain and much to lose from computer software patents, so naturally they can be expected to speak against patents. Along these lines, an associate general counsel of a major Silicon Valley-based global high tech company once told me, "The individual inventor seeking patents is the most deadly threat to corporate America." Complicating the matter are groups, possibly idealistic and utopia seeking, who assert that computer software should not be patentable. The most important of these groups is the open source Linux movement. While, like any successful enterprise, the Linux

movement is beneficial to society, the movement obviously benefits its members. To prohibit computer software patents on their account is to contort the market/legal system to their benefit. Given the nature of institutions to ossify, the Linux movement itself should be subjected to Schumpeter's creative destruction afforded by computer software patents.

Others assert that computer software should not be patentable because large companies would face holdout problems in which the lone inventor with a patent demands excessively high royalties, and because of patent trolls who purchase patents for economic rent extraction purposes, threatening litigation while never intending to otherwise commercialize the patents. Unfortunately, this is capitalism, with significant intellectual justification going back to Adam Smith in the eighteenth century, in which every large company plays a similar game when advantageous. With time, however, capitalism tends to self-correct because of competition and because of the development of general cultural understanding and expectations. Also, the computer software industry is not new, so instances analogous to the U.S.'s experience, in which major companies paid \$100,000 for URL addresses to people who happened to apply for the addresses weeks before, are likely to be rare.

Historically, patents have endured recurring periods of antagonism, only to reemerge recognized as being useful to society. I suspect we are currently in a period of antagonism toward computer software patents, but in due course, I believe the merit of allowing these patents will be recognized. In the meantime, if Europe continues to reject software as unpatentable, it might find itself leeching the IP (intellectual property) created by other countries, as Third-World countries have historically done with patented technologies and trademarks.

As an economist, I believe it preferable to let capitalism reign, without government microeconomic direction, controls, rules – i.e., economic distortions – but with government directed social safety nets and government supported judicial and criminal

prosecuting systems, on the watch for the next Madoff. Accordingly, I believe it preferable for Europe to address its broad issues resulting from globalization through broad public policy initiatives, rather than to contort its individual markets, such as the market for computer software patents.

I would also like to recognize some institutional difficulties of the recent past, namely the U.S. PTO (Patent and Trademark Office) having been overwhelmed by the flood of computer software patents, most of which, I believe should be rejected because of obviousness, including computerizing previously non-computerized processes. I believe that many computer software patent-related problems could be resolved by increasing the number of examiners, allowing examiners more time to do their job. Though a senior person at the USPTO recently rejected the idea of increasing the number of examiners, I believe the market mechanism can help address the problem by increasing both examiner pay and patent application fees. I would be glad to pay five times the current application fees for a faster and more efficient examination process.

In light of the above, to answer your four questions:

1. *Can a computer program only be excluded as a computer program as such if it is explicitly claimed as a computer program?*

Yes. As discussed above, I believe computer software to be patentable subject matter when both novel and nonobvious. Anything that reduces prima facie rejection of computer software patents is desirable.

2. *(a) Can a claim in the area of computer programs avoid exclusion under Art. 52(2)(c) and (3) merely by explicitly mentioning the use of a computer or a computer-readable data storage medium?*

Yes. As discussed above, neither Robinson Crusoe nor the modern individual has any inalienable rights regarding computer software and conversely, computer software patents do not impinge upon any inalienable rights. Consequently, merely mentioning a tie to physical hardware within a claim immediately

extinguishes any inalienable rights issues. The computer software tied to a computer transforms the computer into a specialized computer.

3. *(a) Must a claimed feature cause a technical effect on a physical entity in the real world in order to contribute to the technical character of the claim?*

No. As discussed above, there are no inalienable rights regarding computer software patents; consequently no technical effect on a physical entity is required. A computer and software jointly generating useful and concrete results justifies allowing computer software as patentable subject matter.

3. *(c) If question 3(a) is answered in the negative, can features contribute to the technical character of the claim if the only effects to which they contribute are independent of any particular hardware that may be used?*

Yes. The above arguments regarding computer software patents not impinging upon inalienable rights stand – regardless of whether a computer software program functions on one computer, some computers, or all computers.

4. *(a) Does the activity of programming a computer necessarily involve technical considerations?*

Today yes, tomorrow perhaps no. Today significant computer programming is an art and engineering field, requiring technical understanding of computer functions and knowledge of computer languages. Over time, writing or developing software programs will become increasingly automated and simplified, possibly to the point where anyone can develop a computer program. Such an eventuality, however, does not impeach computer software as being patentable subject matter, because the arguments presented above are independent of such an eventuality.

Joel Jameson  
Silicon Economics, Inc.  
April 30, 2009  
p. 7/7

4. *(b) If question 4(a) is answered in the positive, do all features resulting from programming thus contribute to the technical character of a claim?*

Yes. Arguably too, features include standard computer features that software engineers take for granted, such as the functioning of the operating system.

Respectfully submitted,

Joel Jameson  
President

jjameson@SiliconEconomics.com  
650-941-4083

**G 3/08**

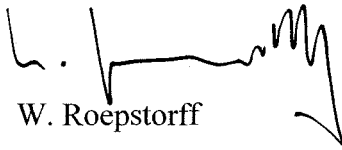
**Amicus curiae brief ' Silicon Economics, Inc' of 30.4.09**

**Note**

The authenticity of the document has been confirmed by a signed fax.

Munich, 6.5.09

The Registrar

  
W. Roepstorff