INFORMATION WANTS TO BE FREE, BUT THE PACKAGING IS GOING TO COST YOU!†

*Gregory A. Stobbs**

Cite As: Gregory A. Stobbs, *Information Wants To Be Free, But The Packaging Is Going to Cost You!*,
2 MICH. TELECOMM. TECH. L. REV. 75 (1996)
available at <http://www.mttlr.org/voltwo/stobbs.pdf>

I. INTRODUCTION

To charge up my batteries before writing *Software Patents* for John Wiley & Sons, I attended four days of hearings on software patents, officiated by Bruce Lehman, Commissioner of the U.S. Patent and Trademark Office. The hearings were split equally between West coast and East, two days in San Jose, California and two days in Washington, D.C. It should come as no surprise that the attendees in San Jose and Washington, D.C. were as diverse as was the weather in those two cities that winter of 1993/94.

The San Jose crowd was a mixture of big business, small business, and what appeared to be folk heroes of the cyberpunk movement. I infer the folk hero status of the latter by the pockets of rowdy cheers and applause that would erupt sporadically from the audience as these persons testified. Big business did little cheering. Also present in San Jose was the computer trade press. The reporters seemed to appear, *en mass*, just in time to hear testimony of their favorite folk heroes and then disappeared. Perhaps the reporters retired to a nearby watering hole to work on their notes in the true spirit of Hunter S. Thompson. In any event, I find it interesting that the just-in-time technique of the automotive manufacturing industry has spread to the trade press industry.

The Washington crowd was a mixture of big business (mostly East coast and international companies), small business, and, of course, Washington, D.C. law firms. If the trade press was present, it was far less obvious. The crowd was considerably more subdued in Washington. The weather may have had a lot to do with it. Much of Washington was closed due to a freezing rain storm, and the audience

† Originally submitted as a position paper for an online panel discussion.

* Gregory A. Stobbs is a principal in the patent law firm of Harness, Dickey & Pierce P.L.C. where he handles patent, trademark, and copyright matters involving complex computer software, networks, database, and speech processing. Mr. Stobbs holds a B.S. in electrical engineering and a J.D.

seemed to comprise mostly persons waiting to testify or waiting to catch the next flight home.

What drew all of these people to testify to Patent Commissioner Lehman? It seems that something being termed the "software patent" is about to explode in the face of the fastest growing industry in America—the software industry. The sides are drawn: one side urging the Commissioner to abolish software patents; the other side deeply fearful that he just might do it. Each side argues forcefully—equities, economics, technology, tradition—and yet, neither side seems to comprehend that the issue they debate is a reincarnation of an issue that has been debated for the last eight hundred years. Queen Elizabeth first debated it when the Crown's right to grant a patent on playing cards was tried in 1602; Thomas Jefferson and James Madison debated it when the Bill of Rights was drafted in 1788; and we debate it now.

The question is this: where do we draw the line between private ownership and the public domain? It is not a question of choosing between copyright and patent, of choosing between hardware and software, or of choosing between implementation and algorithm. It is a more fundamental question that reaches back to ancient human values and transcends our current fixation on computers and software.

It helps to put things in perspective. When debating where we and the law are headed (as we are now), it helps to know where we have been. In this regard, do not assume that software patents are new-trodden soil. In fact, software patents have been around longer than you might think. What was the first software patent? The answer may surprise you.

II. IN SEARCH OF THE FIRST SOFTWARE PATENT

Those who learn about law from the computer trade press treat the software patent as a new disease—as a rash of sorts that must be stopped quickly before it spreads. Perhaps the computer trade press is just writing what its readers want to hear, but they have it all wrong. The software patent is not new. While it is noble to curb a rash of bad decisions, it is lunacy to amputate an arm of law that has served us well for centuries.

Perhaps finding the first software patent will help. Nothing quiets the new disease argument quite as well as giving examples of healthy software patents that flourished long before the recent software patent controversy arose.

Before we can begin identifying the first software patent, we should agree on what software is. That may not be easy. Software is like life.

We can all agree when we see it, but we may not agree on what makes it so. Here are some possible definitions to work with:

> (1) coded instructions that cause a machine to operate in a certain way;
>
> (2) that which adapts a general purpose computer to special purpose;
>
> (3) a computer-implemented method of performing any task.

So what is the very first software patent? Here are two candidates:

> (1) 3,633,l76—Kaiser Aluminum's January 4, 1972 patent for its "Recursive Kopy Program for Remote Input Management System";
>
> (2) 2,552,629—Bell Labs' May 15, 1951 patent for its "Error-Detecting and Correcting System"

The Bell Labs' patent certainly qualifies as early. It was applied for only four years after Bell Labs had finished building its first computer, the General Purpose Relay Calculator. However, there is another software patent candidate that predates the Bell Labs' patent by one hundred years. The patent belongs to Samuel Morse.

You are no doubt familiar with Samuel Morse—the man who was awarded a series of patents in the 1840's for his invention of the telegraph. Did you know that Morse was awarded (and the Supreme Court upheld) a claim to his Morse code? Here is that claim:

I claim, as my invention, the system of signs, consisting of dots and spaces, and of dots, spaces, and horizontal lines, for numerals, letters, words or sentences, substantially as herein set forth and illustrated, for telegraphic purposes.[1]

Could this be the first software patent? Morse code certainly qualifies as "coded instructions that cause a machine to operate in a certain way." Today's digital computers use coded instructions (machine code) based on a binary system (l's and O's). Morse's "machine code" is based on a ternary system (dots, dashes, and spaces). Other than that, there is no real difference.

Before you deny Morse the title of first software patentee (or urge that Morse code is not software because it is only for the telegraphic purpose of transporting information), consider this. A digital computer equipped with modem and a communications program also transports

1. O'Reilly v. Morse, 56 U.S. 62, 86 (1853).

information for telegraphic purposes using its binary system of signs (1's and O's). The computer's communication program is software. Is the message?

III. INFORMATION WANTS TO BE FREE

Another notion that often crops up in debates over software patents is that there is something inherently unique about software and that we are blazing a new trail in legally defining it. Perhaps it is that software is as close as we have come to extending the human brain beyond our bodies. In one sense, software is the transference of knowledge from the human brain to the computer brain, from the neural net to the Internet. Software is a vision sculpted in silicon; it is a master plan written down in Fortran; it is order squeezed from chaos and bottled up in a CD-ROM.

Software is information—information is software. And software wants to be free. "Software wants to be free" is the credo of the cyberpunk movement that traces is roots to William Gibson's science fiction novel *Neuromancer*.[2] *Neuromancer* is about a futuristic world ruled by multinational corporations in which cyberspace cowboys directly wire their brains through computer terminal to the Net in order to hack their way through layers of encryption "ice" to gain access to the information they seek.

Some opponents of software patents take the cyberpunk credo to heart and argue that software, or at least the underlying ideas embodied in the software, should not be subject to private ownership. Software, or at least the underlying ideas embodied in the software, should be free.

The cyberpunks are in good company, at least insofar as ownership of ideas is concerned. Thomas Jefferson, it turns out, was a cyberpunk. Jefferson believed that ideas are and should be free. Here is how Jefferson put it:

> If nature has made any one thing less susceptible than all others of exclusive property, it is the action of the thinking power called an idea, which an individual may exclusively possess as long as he keeps it to himself; but the moment it is divulged, it forces itself into the possession of every one, and the receiver cannot dispossess himself of it. Its peculiar character, too, is that no one possesses the less, because every other possesses the whole of it. He who receives an idea from me, receives

2. WILLIAM GIBSON, NEUROMANCER (1984).

> instruction himself without lessening mine; as he who lights his taper at mine, receives light without darkening me. That ideas would freely spread from one to another over the globe, for the moral and mutual instruction of man, and improvement of his condition, seems to have been peculiarly and benevolently designed by nature, when she made them, like fire, expansible over all space, without lessening their density in any point, and like the air in which we breathe, move, and have our physical being, incapable of confinement or exclusive appropriation.[3]

Do not be mistaken to assume that Jefferson was in any way "antipatent." Jefferson formulated our nation's first patent laws. (Many of Jefferson's concepts are still used today.) Jefferson also served as the first Commissioner of the Patent Office (the position Bruce Lehman holds today), and Jefferson personally reviewed and examined each and every patent that issued through his Patent Office.

The patent law, as Jefferson devised it and as it exists today, draws a line between abstract ideas and patentable inventions. Ideas (ranging from lofty theories about gravity to the punch line of a joke) are not patentable as they are not something that anyone can own. Gravity and the ability to laugh are gifts of nature; they are simply not property that one can own to the exclusion of everyone else.

Inventions are different. Inventions are artificial, human-created things. We humans control our inventions, making them and unmaking them as we see fit. Inventions are patentable if they are novel and not obvious. It has been very difficult for the legal system to articulate the difference between ideas (which Jefferson says should remain free) and inventions (from which society benefits through grant of a patent).

The difficulty in articulating the difference between idea and invention is not unique to software. The U.S. Supreme Court has been grappling with this issue since 1852 (and the English courts before that). The issue is what is patentable subject matter—a separating of raw ideas which are incapable of being patented from the creations of humanity which (if new) are capable of being patented. Here is the Supreme Court's record on this issue—see if you can find a common thread:

> (1) method of making pipe that capitalizes on lead's melting property—held not patentable subject matter;[4]

3. Letter from Thomas Jefferson to Isaac McPhereson (Aug. 13, 1813).
4. Le Roy v. Tatham, 55 U.S. 156 (1852).

(2) method of using electromagnetism, however developed, for remotely printing intelligible characters—held not patentable subject matter;[5]

(3) the rubber tipped pencil eraser—held not patentable subject matter;[6]

(4) method of making flour—held was patentable subject matter;[7]

(5) mechanical process of making metal lath by cutting slits in sheet metal and expanding it—held was patentable subject matter;[8]

(6) a V-shaped radio antenna angled according to a mathematical formula—held was patentable subject matter;[9]

(7) improving the nitrogen fixing ability of leguminous plants by mixing certain naturally occurring strains of different species of root-nodule bacteria and inoculating the plants with the mixture—held not patentable subject matter;[10]

(8) method of converting binary coded decimal (BCD) numbers into pure binary numbers—held not patentable subject matter;[11]

(9) method of updating the value of an alarm limit used in the catalytic chemical conversion of hydrocarbons—held not patentable subject matter;[12]

(10) a bacterium from the genus Pseudomonas containing at least two stable energy-generating plasmids, each providing separate hydrocarbon degradative pathways (the bacteria eats oil spills)—held was patentable subject matter;[13]

5. O'Reilly v. Morse, 56 U.S. 62 (1853).
6. Rubber-Tip Pencil v. Howard, 87 U.S. 498 (1874).
7. Cochrane v. Deener, 94 U.S. 780 (1876).
8. Expanded Metal Co. v. Bradford, 214 U.S. 366 (1909).
9. MacKay Radio & Tel. v. Radio Corp. of Am., 306 U.S. 86 (1939).
10. Funk Bros. Seed v. Kalo Inoculant, 333 U.S. 127 (1948).
11. Gottschalk v. Benson, 409 U.S. 63 (1972).
12. Parker v. Flook, 437 U.S. 584 (1978).
13. Diamond v. Chakrabarty, 447 U.S. 303 (1980).

(11) method of operating a rubber mold in which a digital computer determines when to open the mold—held was patentable subject matter.[14]

Is there a common thread that binds the Court's reasoning? If there is (and I believe there is), it certainly is not immediately apparent. If you will indulge me in a little side trip, I'll show you how I unravel the thread.

IV. THE COMMON THREAD TO PATENTABLE SUBJECT MATTER

You've followed every twist and turn; explored every promising way out, every dead end. Now, just as you think you have conquered the labyrinth, you find yourself standing smack dab in front of a thirty-foot slab of granite that is blocking your way. That's how a software attorney feels each time a new decision is rendered on what is patentable software subject matter. It's little comfort, but the justices probably feel the same way as you.

To be sure, it is an important question. What discoveries should society reward with a patent property right. If answered correctly, society benefits and inventing proliferates. If answered incorrectly, progress pays the ransom. Fortunately, as with many important questions, society has grappled with this property right question before.

If you have had the pleasure of attending law school, perhaps you remember *Pierson v. Post*[15] from first year law school. If not, don't feel bad. It was property law 101. Here is a refresher. Post was held not to own the fox he was chasing across public land; hence Post could not recover against Pierson—a man who leaped from a bushy hiding place and shot the fox first with Post still in hot pursuit.[16] Possession is nine-tenths of the law, and Post did not own the wild fox he was pursuing.

The Supreme Court has rendered three software patent decisions in a long line of decisions on what is patentable subject matter. More recently, the Federal Circuit Court of Appeals (that hears all patent appeals) has rendered four significant software patent decisions in the last year: *In re Alappat*;[17] *In re Warmerdam*;[18] *In re Lowry*;[19] and *In re*

14. Diamond v. Diehr, 450 U.S. 175 (1981).
15. 3 Cai. R. 175 (N.Y. Sup. Ct. 1805).
16. *Id.*
17. 980 F.2d 1439 (Fed. Cir. 1992).
18. 33 F.3d 1354 (Fed. Cir. 1994).
19. 32 F.3d 1579 (Fed. Cir.), *reh'g denied*, 1994 U.S. App. LEXIS 36805 (1994).

Trovato.[20] Many see these cases as irreconcilable. Perhaps there is a common thread in *Pierson v. Post*.

Since early times, the patent has been an inducement to exchange technical know-how for a property right. Share your new and useful discovery with the public, and the patent is yours—that's the deal. If you don't own the discovery you offer, then the deal is off. *Pierson v. Post* teaches us that ownership requires human control. If you do not control the fox, you cannot claim to own it. The public does not grant patents for untamed laws of nature, abstract ideas, and natural phenomena.

What does it mean to be in control? That can be a difficult question. Often proving the absence of control is easier. Take Isaac Newton for example. Newton discovered that apples fall from trees according to precise rules. Newton even devised a branch of mathematics (calculus) to describe these rules. Can Newton claim to own gravity? Well ask yourself, "Does Newton control gravity?" Certainly not. The proof: Newton cannot prevent apples from falling from trees. Newton's discovery, while brilliant, lacks the element of human control. Newton does not own gravity. Gravity remains a wild fox.

V. THE THREAD UNRAVELED

If you're with me so far and want a few examples, read on. I believe you can find the presence or absence of human control at the core of every patentable subject matter question. A few examples from the more recent Federal Circuit decisions in *In re Alappat*,[21] *In re Warmerdam*,[22] *In re Lowry*,[23] and *In re Trovato*[24] help make this point. Let us start with *In re Alappat*.[25] Alappat discovered a rasterizer for making a diagonal line of a digital oscilloscope trace look less like a stair step by partially dimming the pixels that fall on the stair step corners. Human control is present because one can make the rasterizer exist or cease to exist by assembling or disassembling the claimed component parts. The claimed combination is patentable subject matter.

In re Trovato[26] is the other side of the coin. Trovato discovered that the least costly route from point A to point B can be modeled by assigning a cost to each leg of every route and by then trying all possible

20. 60 F.3d 807 (Fed. Cir. 1995).
21. 980 F.2d 1439 (Fed. Cir. 1992).
22. 33 F.3d 1354 (Fed. Cir. 1994).
23. 32 F.3d 1579 (Fed. Cir.), *reh'g denied*, 1994 U.S. App. LEXIS 36805 (1994).
24. 60 F.3d 807 (Fed. Cir. 1995).
25. 980 F.2d 1439 (Fed. Cir. 1992).
26. 60 F.3d 807 (Fed. Cir. 1995).

combinations. However, Trovato's least-cost relationship is true—by definition. The least costly route is always the sum of the least costly segments. One is powerless to change this relationship; hence human control is absent, and the claimed relationship is not patentable subject matter.

In re Warmerdam[27] and *In re Lowry*[28] provide another two-sided coin to examine. Interestingly, both cases involve claims to a computer data structure. Warmerdam claims a data structure for representing a physical object, and Lowry claims a general purpose data structure not tied to any specific physical object. The relation object renders one patentable and the other not but not in the way you might first think.

Warmerdam discovered that abstract "bubbles" lined up along the medial axis of an object (like a snowman along a stick figure) can model the space occupied by the object.[29] Warmerdam's discovery amounts to an abstract observation—true of all physical objects—that a physical object always fits within a series of boundaries lined up along the object's medial axis. One cannot make this relationship cease to exist, hence human control is absent. Warmerdam's data structure is not patentable subject matter.

Lowry devised a general purpose computer data structure to hold and organize information into different categories, like different compartments of digital suitcase.[30] Lowry selected his compartments from an infinite universe of possible data compartments. Human control is present because of this human selection. Lowry's data structure is patentable subject matter.

The next time you face a patentable subject matter issue, give the human control test a try. First, identify what the applicant claims to have invented or discovered, and, then, see if you can find the element of human control. If you find human control and weave it into the claims, you should have no trouble with the patentable subject matter issue. If you cannot find human control, you are probably chasing a wild fox.

VI. SO WHAT IS THE PATENT OFFICE DOING NOW THAT THE COURTS HAVE RULED SOFTWARE IS PATENTABLE?

In response to the recent Federal Circuit decisions, the Patent Office promulgated new Guidelines that the patent examiners use in evaluating

27. 33 F.3d 1354 (Fed. Cir. 1994).
28. 32 F.3d 1579 (Fed. Cir.), *reh'g denied*, 1994 U.S. App. LEXIS 36805 (1994).
29. *Warmerdam*, 33 F.3d at 1357.
30. *Lowry*, 32 F.3d at 1580.

software patent inventions.[31] Technically, the Guidelines are not law, but rather they are the Patent Office's interpretation of the law. Nevertheless, the Guidelines are quite important because they dictate what the software patent applicant must do to satisfy the Patent Office.

So what's new? For one thing, the Patent Office has dropped its fixation on the prohibition against all mathematical algorithms. Instead, the Patent Office now offers several suggestions on how certain computer-related subject matter may be presumed statutory. You will undoubtedly find that many software inventions fall into one of these three categories:

> (1) computer or programmable apparatus (a statutory "machine");[32]
>
> (2) computer-readable memory for directing a computer to function in a particular manner (a statutory "article of manufacture");[33]
>
> (3) computer-implemented or computer-aided process (a statutory "process").[34]

Take a good look at the second category. To claim the algorithm, you have to claim the memory. To claim the genie, you have to claim the bottle. That appears to be what the Patent Office is saying.

Does that mean that any genie you can put into memory is statutory subject matter? No. Prerecorded music stored in CD-ROM memory, whether the earliest Beethoven or the latest Body Bags, is one genie that is not patentable subject matter. We all know this instinctively. The challenge is to explain why—something which the Guidelines do not do. Thus, for the time being, a technological genie on a disk is patentable; music, literature, and art on a disk is not.

Maybe this will not present a problem for most of us, but if the Jefferson Airplane will sue a screen saver manufacturer over flying toasters,[35] someone out there is bound to force the Federal Circuit to answer this thorny software patent question: what is art?

---

31. Examination Guidelines for Computer Related Inventions, 61 Fed. Reg. 7478 (1996).

32. *Id*. at 7482.

33. *Id*.

34. *Id*. at 7483.

35. *See* Jefferson Airplane v. Berkeley Systems, 886 F. Supp. 713 (N.D. Cal. 1994).

VII. WHAT IS IN STORE FOR THE FUTURE?

In the future, we may have to adjust the cyberpunk's credo: "Information wants to be free, but the information packaging and delivery system is going to cost you!" I envision a layered intellectual property protection model, something like the ISO/OSI layered model used to describe software. Both patent and copyright protection will factor into the layered model.

If you are not familiar with the ISO/OSI layered model, here is a brief introduction. ISO/OSI stands for International Organization for Standardization/Open Systems Interconnection. It is a layered architectural model or plan for how computers communicate with each other over a communications network. The model can also be applied to software itself. The plan divides software communication into seven layers, each layer building on the layer below it, with the most primitive functions handled at the lowest layer (called the physical layer) and with the most advanced functions handled at the highest layer (called the application layer). By analogy to a human-to-human telephone conversation, the physical layer concerns whether the phone is plugged in properly, and the application layer concerns whether both parties understand the semantic meaning of the words they are speaking.

If you think about it, much software developed today is a form of information packaging and delivery. Certainly, World Wide Web browsers and interactive television software fall into this category. So does the lowly spreadsheet program that packages the data you enter and supplies it back to you in a tabular or graphical form that is easier to understand. It's all information packaging.

Viewed as an information packaging and delivery system, software fits a layered ISO/OSI model quite well. At least the layered model allows us to separate the information or content from the software systems employed to package and deliver this content. Take a software system for delivering prerecorded music over the Internet in an interactive, user-controllable form. To the teenage end user the product is music—all that matters is the content. To the software engineer, the product may be a new form of Java applet that allows music to be compressed, delivered, and decompressed faster than it would take to listen to the song when played—all that matters is the compression technology; the content is irrelevant. To the communication engineer, the product may be a new form of software controlled fiber optic switching system that allows any piece of music to be delivered to the teenager on demand—again the software switching technology is all that matters; the content of the information and how it is compressed is irrelevant.

As you can see, the software-enabled technology, which is on the horizon, involves many different layers of abstraction. Different blends of intellectual property protection may be required. For the highest level of abstraction (the information content itself), I believe the copyright will continue to be the most suitable form of protection. For the lowest level of abstraction (the physical packaging and delivery details), I believe the patent will continue to be the most suitable form of protection. In the middle is where it gets interesting.