***Software and Patent Scope: A Report from the Middle Innings***

**Robert P. Merges**

**UC Berkeley**

## I. Introduction

I started studying and teaching[1] patent law in 1988. At that time, software patents were just beginning to seep out from the back currents and ease into the mainstream. I will not bore the reader with a case-by-case account of this long process. Suffice it to say: with the persistence of rabbits in a field or termites in a fallen tree, software patents just kept showing up. Finally, perhaps exhausted with the business of drawing lines around "real" software inventions, in a futile attempt to keep them out of the domain of patents, a hapless Federal Circuit threw in the towel in 1998. And, as so many predicted, après *State Street*, le deluge. Applications, and then of course issued patents, claiming software increased dramatically, and today they are commonplace.

This brief (I promise) article surveys the post-deluge scene in the software industry. Though there are many – and I mean *many* – interesting predictions and provocative hypotheses in the vast literature on the topic, I will limit myself (in Part II) to two observations about how patents have affected the industry. For good measure, or just to stir things up, I will add a conjecture about the future. Observation number one is deceptively straightforward: the software industry is thriving, rather than dying. Whatever else patents have done or not done to or for this important industry, they have not killed it. Observation two explains why: legal issues are of secondary importance. The major driving forces in the industry, technological change and widespread capital availability, simply swamp whatever marginal effects law might be exerting.

Viewed from the perspective of legal doctrine, the trends I survey in Part II add up to one simple conclusion: history has largely answered the "section 101" question, viz., is software patentable subject matter, and should it be? There is no doubt now that it is, and the normative question is dropping away in importance. Thus the interesting questions now concern the contours and details of software protection doctrine. As always with patent law, many of the important questions come down to issues of *scope*: how broad will software patents be? How many potential software programs will be covered by a given patent? Already, the caselaw is moving beyond section 101 to address this fundamental question of patent scope. And of the doctrines that collectively determine scope, none is more au courant than the law of disclosure – in particular, the knotty and troublesome "written description" doctrine.

---

[1] Actually, as a beloved older colleague used to remind me, I merely "met my classes regularly." Only sporadically, then and now, do I actually teach anyone anything.

A recent Federal Circuit software patent case, *Lizardtech v. Resources Mapping, Inc.*,[2] turned decisively on application of this doctrine. *Lizardtech* thus provides an excellent vehicle for discussing current developments in the law of software patent scope. This is the task I take up in Part III of this paper. In that Part I briefly survey the rapid ascent of written description doctrine, and explain why it has emerged as an adjunct – if not a complete successor – to the traditional law of enablement. The *Lizardtech* opinion itself reveals both the appeal and the shortcomings of this prominent new doctrine. On the positive side, the impetus behind the written description revolution is well illustrated by the facts in *Lizardtech*: the patent claim at issue appears to have been significantly broader than what the teachings of the specification could reasonably support. Yet I also argue in Part III that the Federal Circuit could have reached the same result using the reliable tools of traditional enablement doctrine. The thrust of my comments, then, is to praise the outcome of some of the recent written description cases, while advocating a more parsimonious doctrinal approach: namely, reliance on the traditional "undue experimentation" standard of section 112. I admit that this standard might be improved with some timely renovations. But the court need not have buried it altogether under the confusing rubble of a new and untested doctrine, as it seems to have done.

In the end, though, *Lizardtech* is important more for its outcome than its reasoning. As I describe in Part IV, if it is followed by other cases, *Lizardtech* will signal an important trend. Together with other recent developments, particularly the hobbling of the formerly robust doctrine of equivalents, it may presage a cautiously narrow approach to software patent scope. If so, an important theoretical issue will be joined. Loosely speaking, there have been two major schools of thought on the topic of patent scope. The "big, wide, early" school, which often rallies under the banner of Edmund Kitch's prospect theory, favors broad initial property grants to pioneers. The "narrower, pro-competition" school favors narrower patents as a first principle, with varying degrees of deviation permitted based on the nature of the technology or industry in question. The consensus among commentators on software patent scope, and a fair inference from other work in the literature, is that software is a good candidate for this "narrower, pro-competition" treatment. So if these theoreticians are correct, *Lizardtech* and any progeny it might spawn would seem to be a step in the evolutionary right direction.

But another branch of theory – the more recent "anticomons" theory – would point out that too many narrow patents could coalesce into a dark cloud on the software industry's horizon. The reason, in short, is transaction costs. Too many individual property rights in the hands of too many individual owners can – theoretically at least – wreak havoc with the creation and distribution of workable software products, which often encompass many potentially patentable components. So Part IV describes how the anticommons logic may play out in the software industry. In this Part, I put my credentials as a "transaction cost optimist" to work, describing various means and mechanisms by which at least the worst version of the anticommons is likely to be avoided. In addition, I describe judicial tools that can be deployed to prevent unfair rent-seeking via rogue patent owners, in particular the newly rediscovered powers of equity in the granting of injunctionc post-*eBay v. MercExchange*.

---

[2] 424 F.3d 1336 (Fed. Cir. 2005).

In the conclusion, I wrap all this up, trying to sound wise and prophetic, without being redundant. If you keep faith with me until the end of the article, you can judge whether I succeeded. Or not.

## II. The Software Industry: Yesterday and Today

I begin with some history. Fifteen or so years ago, when it was becoming apparent that patents were becoming a fact of life in the software industry, many feared for the industry's future. An MIT forum in 1989, for example, brought together a fair cross-section of people involved with the software industry in the Boston area. At that forum, Dan Bricklin – co-founder of the company that created VisiCalc (an early "hit" product in the Apple II era) – spoke up about his concerns. The official notes of the Forum read as follows:

> Bricklin emphasized that a sophisticated applications program may involve 10-10,000 patentable processes. He noted that if companies began spending money to obtain software patents for these processes then the current royalty structure would have to change in order for companies to remain profitable. Bricklin noted that there is an important distinction to be made [between] ... most patents [and] ... software patents. He explained that there is usually one patent that covers the whole product in the case of plant or chemical products. In contrast, a software product can easily involve hundreds of patents for a single product.
>
> Bricklin characterized the software industry as inherently cottage-based. He explained how most of the major advances in the PC industry seem to come out of small shops or out of small development teams. Some examples include WordPerfect Corp., Lotus Corp., and Software Arts. Bricklin noted that with even better tools today one programmer can do even more than he accomplished in the past. He believes that some products should be written by individuals or small groups to achieve better cohesiveness while there is still demand for large companies to handle the larger scale projects. In some cases, Bricklin notes that it is cheaper for a company to go outside and buy a software product rather than develop [it] themselves. He believes that if the industry had the copyright protection just on the source code, it would be cheaper to buy than to make.
>
> Bricklin commented that many people feel software must be "protectable" because it is a product of someone's hard work. In his opinion, "craftsmanship" is not protectable and he does not feel that just because you work hard on something, e.g., software, that it should be protectable. He believes we should have patents because patents advance technology, not because patents are inherently good.
>
> Bricklin also cited the problem of "mine fields" in that a software developer often finds out about a related patent after the product has been shipped. . . . As a

> developer of software he is also uneasy about patents since he admits having limited knowledge about intellectual property. In reaction to the increase in software patents, Bricklin noted he has been working on lower tech products which involve using information in the public domain.[3]

Bricklin was by no means alone; many others shared the same fear. In a 1994 manifesto alarmingly entitled "Software Patents: An Industry at Risk," the League for Programming Freedom argued for "the special properties of software that make the application of the patent system inappropriate," including "the complexity of software" and unprecedented rapid change and development.[4] (Irlam and Ross, 1994, at section 2). Their prediction for a future with patents was bleak:

> A vision of patents entrenched in the software industry is a vision of stagnation. A vision of IBM once again calling the shots. A vision of companies like Xerox and AT&T who have proven incapable of bringing innovative products to market stealing profits from those companies [that] can.[5]

## A. Fears of the Forefathers

It is safe to say that software professionals such as Dan Bricklin and the members of the League for Programming Freedom were mostly, but not completely, wrong about the prospects for software patents. Patents have not killed the software industry; they have not led to a slowdown in entry; and they do not appear to have had much if any effect on industry structure. In addition, despite the predictions of the League for Programming Freedom, the industry has stagnated. Early leaders in acquiring patents, such as the legacy hardware-oriented firms such as Xerox and AT&T, have not been able to prevent entry, slow innovation, or even slow down the evolution of the industry to any measurable degree. (Indeed, some of these legacy firms are struggling to survive, in part due to the dynamism of software and other sectors of the technology-intensive industries in which they operate.) In particular, patents do not seem to have influenced overall industry concentration, nor do they appear to have affected the minimum efficient scale of firms in the industry. In addition, I will present data showing that successful software firms are doing more than simply cynically stockpiling patents; they are putting real effort into seeking and obtaining high-quality patents, patents that demonstrate significant earmarks of quality.

The early patent-era predictions about the software forecast an industry where entry stagnated. The idea was that patents would entrench established companies, slowing the pace of change and ultimately putting a damper on firm entry. In this section of the paper, I discuss broad trends in firm entry in the software industry since the dawn

---

[3] MIT Communications Forum, "Software Patents: A Horrible Mistake," March 23, 1989, Seminar Notes. 1989 (statement of Daniel Bricklin, President of Software Garden, Inc.).

[4] See, e.g., Gordon Irlam and Ross Williams, League for Programming Freedom, "Software Patents: An Industry at Risk," 1994, avail. at http://lpf.ai.mit.edu/Patents/industry-at-risk.html.

[5] Irlam and Ross, supra note 4, at section 4.3.

of the patent era (1988-1994). As suggested earlier, I conclude that entry continues to be robust, and therefore that the predictions just mentioned have turned out to be wrong.

So my observation amounts to this: patents have not killed software. Now I am the first to admit that this is hardly a ringing endorsement for the new regime of software patents. After all, who would buy something from a salesperson whose only claim was that the product would not kill you? However, given the history of debate in this area, this is a nontrivial point. For there was a time, and there were people (sometimes, almost, myself included) who thought patents would very seriously harm the software industry. For them (including me, or at least, some past version of me) it is good news indeed that they (we) were wrong. By almost any measure, the software industry in the U.S. is doing really quite well. Whether this is because software patents are really in the end *good* for the industry, or whether the industry has just learned to get by with them, and maybe at times put them to useful ends, no one really knows, though I make some guesses in the Conclusion (section IV). But the simple point is that the industry has survived the onslaught of patents, at least reasonably well and at least so far.

## B. General Industry Structure: Predictions and Reality

Another prediction from the 1990s was that patents would make the software industry become more concentrated over time. Indeed, it was not uncommon for scholars to suggest that legal protection for software – at the time, usually copyright protection – would contribute to this trend. The idea was that strong ownership over software, in particular "backbone" software having network effect implications, would tend to "lock in" a dominant product, and hence the company that owned it. What this view ignored was the dynamic sources of growth in the industry. The antitrust aphorism had it that in software and other network industries, there is competition *for* markets rather than competition *in* markets. The view from the 1990s underestimated how real and effective this competition was in the fields where it operated. But it also overestimated its pervasiveness. For every "backbone" product – such as an operating system program – there are many applications and ancillary products that connect to the backbone. For these products, ownership rights do not appear to create "lock-in" conditions on anything like the scale envisioned in the early 1990s.

Industry concentration statistics tell the story here. The conventional measure of concentration, the Herfindahl-Hirschman index (HHI) ranges from 0 to 10,000; the HHI for the software industry as a whole is less than 244 for software, compared to an average of 334 for U.S. manufacturing industries. What this means is that the top 20 sellers of pre-packaged software generate 61% of total industry revenues. This compares very favorably to other industries, many of which are considered quite competitive: autos, airlines, and personal computers, for example. There is evidence of significant turnover over time as well – a key indicator of a dynamic industry. Of the top ten software companies in 1990, five did not make the list in 2000, either because they went out of business or were acquired. This is remarkable turnover compared to some industries, such as pharmaceuticals, where similar comparisons from 1990 and 2000 show that eight of

ten firms made both lists (and the ones that did not were acquired by others that did) (Evans 2002:36).

There is solid evidence that changes in the industry were driven by innovation. Evans (2002: 36) cites estimates that in 1986, research and development (R&D) by publicly traded software firms was 1% of total domestic R&D; by 2000, that had grown to 10%. The transition in the industry from stand-alone enterprise and desktop computing to a fully-networked, Internet-based computing environment, was fueled by this massive R&D spending. Many new firms took advantage of the opportunities created by this transition. Changes in the composition of industry leadership thus belie the predictions of a stagnant industry.

## 1. Patents and Innovation: Some Comparative Data

How do patents relate to these trends? Consider some evidence from U.S. patents obtained by foreign-based software companies. In general, researchers find that as a foreign country moves up the learning curve in the software industry, inventors in the software industry from that country receive more patents. For example, in data through 2002, inventors from Israeli software firms received far more patents than did those from Ireland and India. The latter two countries have sizeable software industries; Irish software firms earned $ 18 billion in 2003, while Indian firms earned $1.5 billion in the same year. (Arora and Gambardella, 2005: 9, 41). The Israeli industry has revenues of $4.1 billion. Nonetheless, Israeli inventors receive far more U.S. patents.

This comparative data tracks qualitative assessments regarding the nature of the software industries in the three countries. Software firms in India and Ireland are less innovative than elsewhere. Routine service-type programming is the norm in India, while in Ireland, the industry is dominated by subsidiaries of foreign firms who add minor value to the parent company software, and who are located in Ireland partly for tax reasons. Israeli firms, which are perceived as being the source of more innovative software, patent much more heavily than their Indian and Irish counterparts. (Arora and Gambardella, 2005: 45). While a number of reasons might explain this pattern, it is certainly consistent with the view that patents correlate closely with R&D and innovation – which would tend to refute the early 1990s argument that patents are anathema to software innovation. In addition, it can be said that the Israeli software industry is in no sense highly concentrated. So the comparative data once again supports the notion that predictions concerning the concentration-increasing effects of software patents have failed to materialize.

## B. Focus on Entry

Different industries reflect different sources of innovation. In some, a few large, established firms contribute significant innovations. In others, much innovation comes from small startups. The software industry shows some elements of both patterns.

Established firms often contribute significantly to innovation, while new startups have also been a major source of new products and other innovations. While a serious decline in the volume of startup activity would not therefore necessarily represent evidence of industry stagnation, a steady flow of new entrants would be in keeping with the industry's traditional pattern of innovation.

As Figure 1 shows, new firms have entered the software industry at a healthy clip since 1970. Most importantly for purposes of this paper, none of the major signposts along the road toward software patentability appear to have had any impact on the volume of startup activity. After the earliest years in the chart,[6] several important patent-related events occurred.

**Figure 1: Software Startups**

| Year | Number of new firms formed |
|---|---|
| 1970 | 102 |
| 1971 | 81 |
| 1972 | 123 |
| 1973 | 121 |
| 1974 | 119 |
| 1975 | 208 |
| 1976 | 224 |
| 1977 | 231 |
| 1978 | 342 |
| 1979 | 428 |
| 1980 | 402 |
| 1981 | 480 |
| 1982 | 504 |
| 1983 | 530 |
| 1984 | 528 |
| 1985 | 551 |
| 1986 | 500 |
| 1987 | 430 |
| 1988 | 412 |
| 1989 | 436 |
| 1990 | 358 |
| 1991 | 379 |

[6] The early data coincide with the 1972 Supreme Court decision in *Gottschalk v. Benson*. The *Benson* decision was decidedly "anti-patent"; it cast serious doubt on whether software would be patentable. But it in many ways simply confirmed what the long-standing assumption had been: that computer programs – or "algorithm inventions, as they were then referred to – were not a proper subject matter for patents. Under a crude "inverse relationship" hypothesis, the Benson opinion might be expected to have led to greater startup activity. Yet *Benson* was not enough of a "policy shock" to have any impact on startup activity.

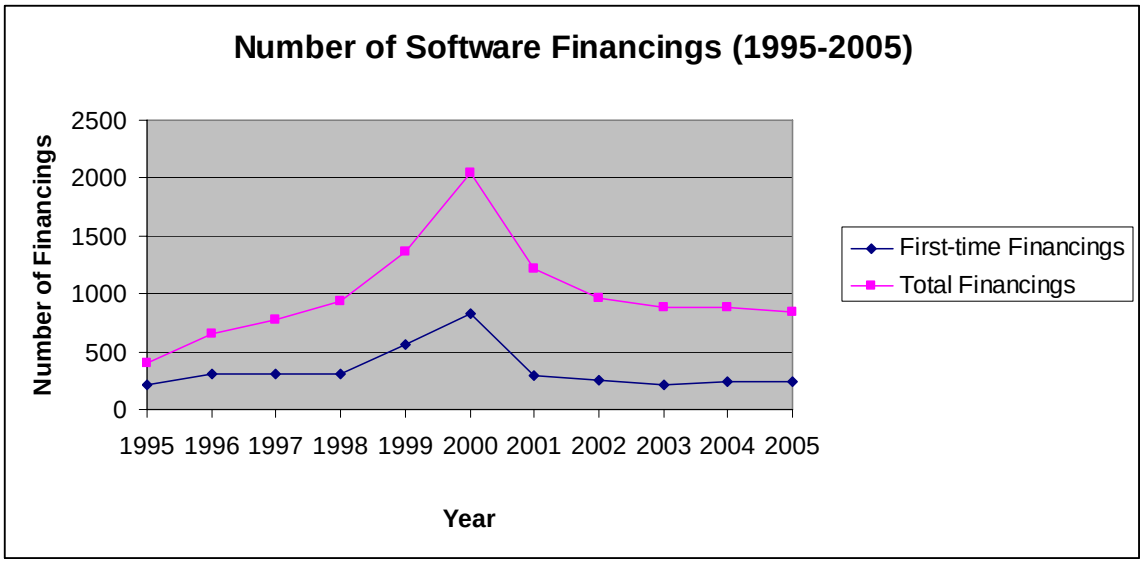
| | |
|---|---|
| 1992 | 360 |
| 1993 | 337 |
| 1994 | 416 |
| 1995 | 469 |
| 1996 | 497 |
| 1997 | 353 |

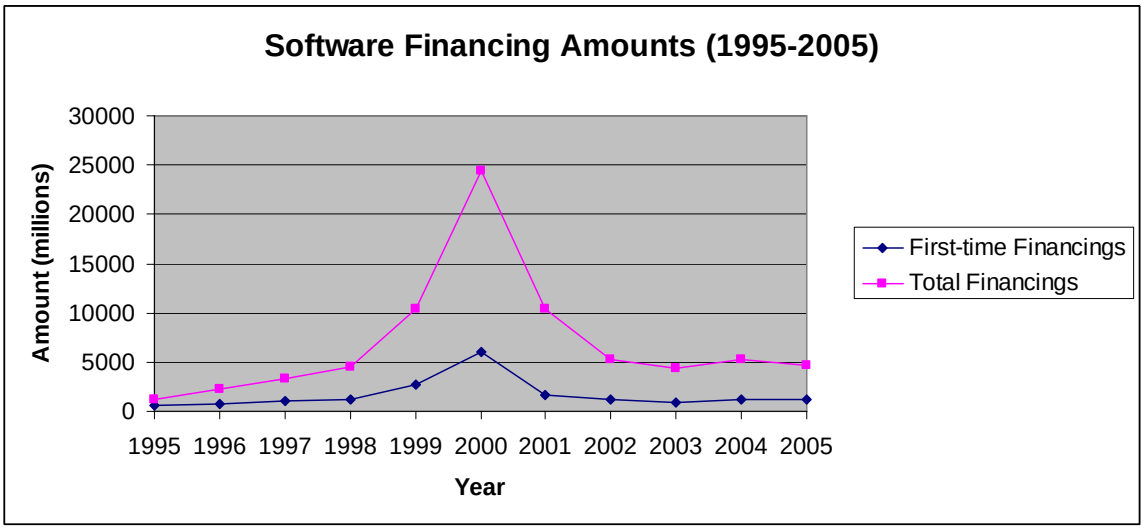
More detailed data on startup activity, including venture capital funding amounts, is shown in Figure 2:

**Figure 2**

| | First time software financings | First time software financing amounts | Total software financings | Total software financing amounts |
|---|---|---|---|---|
| 1995 | 210 | $553M | 401 | 1157 |
| 1996 | 305 | $822M | 655 | 2283 |
| 1997 | 311 | $1018 | 780 | 3369 |
| 1998 | 309 | $1181M | 931 | 4493 |
| 1999 | 562 | $2697 | 1361 | $10466M |
| 2000 | 829 | $6023 | 2047 | 24435 |
| 2001 | 292 | $1643 | 1217 | 10408 |
| 2002 | 260 | $1178 | 959 | 5306 |
| 2003 | 214 | $892 | 880 | 4432 |
| 2004 | 234 | $1180 | 887 | 5247 |
| 2005 | 238 | $1180 | 840 | 4704 |

As the following graphs show, putting aside the 1999-2000 Internet financing "bubble," software startups and financing activity have remained robust throughout the period of interest:




Indeed, a recent book by industry expert Michael Cusumano, aimed directly at software entrepreneurs, MIT's Michael Cusumano points out that currently, as has been true historically, only about 6% of the software company business plans pitched to venture capitalists receive funding (Cusumano, 2004: 196). Cusumano lays out guidelines and suggestions for the budding software startup founder, and sprinkles cautionary tales of failure throughout – solid evidence that entry is still robust in this important industry.

## C. Putting Legal Issues in Context

To review, patents have now become common in the software industry.[7] Despite earlier predictions,[8] the industry has not slowed to a crawl.[9] Why not?

This leads to my second observation. This one is just as prosaic, though maybe a bit more novel for law review readers: it is that developments in technology and business have proven to be far more important than legal developments in the software industry since 1988. The almost miraculous increase in computing power (as predicted in "Moore's law") that began in the 1970s has continued up to the present, creating the technological equivalent of the "permanent revolution" associated with a leading Mexican political party. The internet is only the most recent manifestation of this. Earlier waves of innovation emanated from the minicomputer, the personal computer, video games, and artificial intelligence. Ever-increasing computing power, and fatter broadband "pipes" for content, are the revolutionary technological frontiers of today. This too shall pass – but another one will surely come, as long as computing power continues to grow.

On the business side, this breakneck innovation has, predictably, drawn the interest of people with a lot of money to invest. This investment push has exerted a huge influence on the software industry. External sources of finance mean that the industry's growth is not tied to its own current revenue. Independent entrepreneurs (or those who hope to be soon) pitch new product concepts and ideas for new companies to people with a lot of money to invest – typically principals of venture capital investment funds, or VC's for short. The steady influx of VC money frees entrepreneurs from having to move up the chain of command in a large company, waiting for enough seniority and a rosy enough capital market inside the firm to pitch a new product idea. It is easy to overlook how important this is. Not only does it allow talented and driven people people to create new companies early in their careers. The constant flow of startup company activity keeps the pressure on the established firms – the incumbents – to keep up with new trends and continue to innovate. While a large "installed base" can be an advantage, having to maintain "legacy products" while trying to keep pace with new entrants can also pose a serious challenge to large software companies.

Taken together, technological change and VC financing have largely shaped the U.S. software industry. The legal system, and specific legal rules in particular, have played only a modest, background, role. Perhaps out of hubris, those who predicted bad

---

[7] See generally Stuart J.H. Graham & David C. Mowery, Intellectual Property Protection in the U.S. Software Industry, in Patents in the Knowledge-Based Economy 219 (Wesley M. Cohen & Stephen A. Merrill eds., 2003).

[8] See note 2 supra. Some of these dire predictions have a more recent vintage. See, e.g., James Bessen and Robert M. Hunt, An Empirical Look at Software Patents, Research on Innovation Working Paper 03-17R, avail. at www.researchoninnovation.org. For a response to Bessen and Hunt, see Robert Hahn & Scott Wallstein, A Review of Bessen and Hunt's Analysis of Software Patents, Working Paper: IE-Brookings Joint Center (November 2003).

[9] For less pessimistic assessments of the quality and impact of software patents, see Martin Campbell-Kelly, "Not All Bad: An Historical Perspective on Software Patents", 11 Michigan Telecommunications and Technology Law Review, 191 (2005); Bronwyn H. Hall and Megan MacGarvie, The Private Value of Software Patents, Nat'l Bureau of Economic Research Working Paper 12195 (April, 2006), avail. www.NBER.org (calculating "abnormal" returns and stock market valuations for firms holding software patents, and finding that patents are often associated with increased firm value).

things for software with the coming of patents really missed this essential truth. What it means for present debates is just this and nothing more: that we are talking about tinkering at the margins. Patent law may exert some small force on the overall shape and direction of software. But in the retrospective vector analysis that historians might construct, it will be a distinctly minor force indeed. Changing technological paradigms and capital market conditions, in particular within the VC industry, will be far more important.

And so we in the legal community should be reassured: there is little chance we will mess up this tremendous progress in any fundamental way. Yet of course we are not off the hook completely. For even if it is only at the margin, the legal rules do matter. It is incumbent upon us to do as well as we can to foster, promote and support the development of this important industry. This we can do by designing and implementing intelligent, workable legal rules for the efficient operation of the industry. My point in emphasizing the predominance of technological and business forces in the industry is merely to remind us that whatever legal rules we design will play out in the context of a dynamic, expanding industry. We would be wise to keep this context in mind when designing the rules (e.g., regarding patent scope) and in predicting their impact, as I do in the next section.

## III. Enablement and "Written Description" in the Software Industry

So legal developments exert only a marginal effect on the direction of the software industry. But this is not the same as saying they are irrelevant. By definition, in a close situation, with other factors roughly in equipoise, legal rules can make a difference. Of the various doctrines that collectively determine patent scope, those involving disclosure are highly important. Of particular contemporary importance are doctrines governing the relationship between the disclosure portion of a patent (the specification) and the legal claims that follow it.[10] These matters are governed by section 112 of the Patent Act.[11] I consider these doctrines in this section.

---

[10] Technically, as the statute makes clear, the claims are not distinct from the disclosure portion of the patent. But by common usage, practitioners often refer to the disclosure portion as the "specification," to distinguish it from the claims at the end, so I use that terminology here. See 35 U.S.C. § 112 ("The specification shall conclude with one or more claims . . . ."). The Federal Circuit has consistently in recent years used "written description," or, more verbosely, "written description portion of the specification," in its opinions, but I find this confusing and so resist it. The shorter phrase "written description" is easily mistaken for the written description *doctrine*, doubling the confusion associated with that term; while "written description portion of the specification" uses six words where one ("specification") would serve. I am all for slavish consistency when it will serve a useful purpose, but it becomes for me a small-minded defense against a mythical hobgoblin when it takes the form of pedantic repetition.

[11] This section reads in part:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same, and shall set forth the best mode contemplated by the inventor of carrying out his invention.

35 U.S.C. § 112.

## A. A Very Rapid History of Disclosure Law: Emergence of the "Classical" Enablement Standard

Early in their history, patents were special royal favors or privileges handed out to those who the sovereign wanted to reward. Because of this, the relevant state authorities did not consider it important to publish detailed disclosures of patented inventions. Inventions were valued for the contribution they made to the state, in the form of practical benefits that flowed from their construction and operation. The idea "letters patent" being valuable for their *information content* did not arise until the eighteenth century in Britain. And when the idea did emerge, it was as part of a general effort on the part of the British courts to limit and restrain the rights of patentees – a hangover, so to speak, from the "anti-monopoly" era of the seventeenth century that put such a distinctive mark on early British patent law. Thus British patent cases through the early nineteenth century imposed harsh disclosure rules on patentees, whose overall effect was to invalidate a good number of the patents that those courts were called upon to review. One oft-cited doctrine provided, for example, that a patent was invalid if an inventor disclosed multiple means of carrying out the invention, but one of those means was in fact inoperative. The courts were quick in these cases to impute moral culpability to inventors in such circumstances; they did not inquire into the benefits of the inventions at issue, or practical problems of construction or implementation that might have been present in carrying out the invention. Although in the latter part of this period the courts sometimes apologized for these results, restrictive disclosure rules were an established part of the law.

American jurists, sensitive to the different conditions they faced in the new world, diverged from the British doctrine. For them, patents were not a vestige of decrepit and corrupt political maneuvers, but instead represented the flourishing of a vital, energetic young nation. Perhaps in this they were influenced by the fact that machinery was widely perceived as a solution to the chronic labor shortage facing the newly-independent nation. Whatever their ideological foundation, Justice William Story and other notable judges in the early federal period fashioned a much more liberal disclosure standard in patent law. They looked for a real inventive contribution, and if they found it, they did not scrutinize the inventor's specification for technical defects or minor deficiencies. This is not to say that they were unreservedly "pro-patentee," however. When the facts demanded, they were quite willing to invalidate a patent whose claims appeared to far outrun the value of the information disclosed in its specification. This was the case, for example, with an 1825 case involving one of many patents on steamboat technology.

This liberal but balanced view of the law of disclosure prevailed throughout the nineteenth century. By the late nineteenth century, it had come to be stated as the "undue experimentation" standard, invoked famously in *The Incandescent Lamp Patent* case of 1895. Under this standard, a skilled artisan must be able to construct all or most of the embodiments covered by a patent claim without the need for excessive or "undue"

---

35 U.S.C. § 112, ¶ 1 (2000).

experimentation. (Claims themselves by this time had taken on their modern form, i.e., in full text, describing the outer boundaries of the invention, placed at the end of the specification.) Undue experimentation carried forward the flexible test of the early years. It admitted of some experimentation, again in distinction to the early British cases. Yet it also provided that a claim was invalid if it would take too much additional work for someone in a patent's field to move from the description in the specification to the actual making or use of the claimed invention. Sometimes, the then-emerging law of "invention" (today, nonobviousness) was invoked in service of this test: if the research require to span the gap from specification to working embodiments was so great that it constituted patentable invention, then the specification was *a fortiori* deficient.

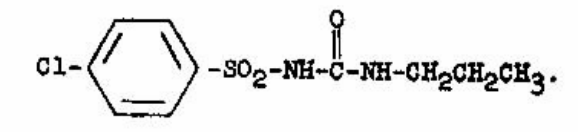
The story of the disclosure requirement in the twentieth century is one of gradual refinement and adaptation. Doctrinal nuance was introduced in the application of basic disclosure principles to emerging new scientific and technical fields, even as the foundational principle of 'undue experimentation" remained fixed in place. So for example the very large families of compounds that came to be claimed (using the Markush claim format) in chemical patent practice gave rise to some important cases in the law of "chemical enablement"; but the basic contours of the doctrine did not change.[12] This was true also for biotechnology, which emerged later in the century. And the few early cases that applied disclosure law in the area of software patents recited similar principles as well, though these have been criticized in the academic literature.

Beginning in the 1960s, a small number of patent opinions were premised on a written description requirement distinct from "classical" enablement. But these were few in number, and limited to a certain narrow fact situation. Typically, an inventor filing an application overseas would file a counterpart application in the U.S., adding or modifying one or more claims. If the added claims lacked solid support in the specification, courts would say that they had not been adequately described in the original specification, and were therefore invalid. Many of these cases involved chemical inventions. [13] It seems evident that only a minor adjustment would have been necessary to characterize this as an

---

[12] See, e.g., *In re* Gardner, 427 F.2d 786, 789 (C.C.P.A. 1970). In *Gardner*, the inventor claimed to have discovered antidepressant activity in 2-aminomethyl-1, 3-benzodioxole compounds. The PTO rejected the patent application because, in part, the specification failed to disclose how to use the invention, i.e. what dosages of the drug are effective. The court found that one skilled in the art could, after "*a great deal of work,*" eventually find out how to use the invention. The court held that "the law requires that the disclosure in the application shall *inform* them *how* to use, not how to find out how to use for themselves." *Id.* See generally, David J. Weitz, The Biological Deposit Requirement: A Means of Assuring Adequate Disclosure, 18 Berkeley Tech. L.J. (2004).

[13] The Ruschig case involved a family of compounds described by the following structure:



enablement issue.[14] After all, from an early time, the enablement requirement has been measured from the date a patent is filed; hence the later-filed claims in these cases might well have been said not to have been enabled by the earlier application.[15] But this seemingly sensible solution was foreclosed because the opinions in the cases insisted that the earlier application did enable the later claims. The defect in disclosure was characterized as a separate matter: that the specification did not properly identify or focus on the later-claimed subject matter. I will have more to say on this complex issue later; the important point here is just to note that it entered the caselaw through these cases.

This additional disclosure standard– identified in later cases as the "written description" requirement – lay dormant for a good while. It was not until the 1990s that this requirement was dusted off and applied with a vengeance, giving the settled law of enablement a violent twist. First in cases that followed the older pattern (initial specification, later, broader claim), and then in a series of biotechnology cases, patents were found enabled but invalid for failure to meet the written description requirement. The doctrine has grown so quickly, and been applied so widely, that enablement rarely decides disclosure cases in the Federal Circuit today. The "written description" requirement now dominates the scene.

## 1. Infatuation With Written Description

In the next section, I describe a subset of today's written description cases that really do present novel issues for traditional enablement law. Unfortunately, the Federal Circuit has now gone well beyond these challenging cases in fashioning written description doctrine. That court has chosen to extend written description principles into many cases best decided under traditional enablement doctrine. For example, consider *University of Rochester v. G.D. Searle & Co. Inc.*[16] In this case, scientists had discovered a novel physiological mechanism for blocking the action of a common receptor, which held the promise of delivering a new family of painkilling drugs devoid of some negative side effects that had plagued the prior art. The difficulty with the patent application in the case was that no actual painkilling compounds were identified; only the mechanism by

---

[14] Indeed, the examiner's rejection at issue in the *Ruschig* case might well have been for lack of enablement: the application, he states, "is rejected as having no specific support in this disclosure." 379 F.2d at 992. He went on to state:

> The compound of claim 13 is not named or identified by formula and it can find support only as choices made between the several variables involved. This is not regarded as adequate support for a specific compound never named or otherwise exemplified in the specification as filed.

Id. The examiner then cited In re Fried, 312 F.2d 930 (C.C.P.A. 1963), a similar chemical case holding that a parent application which suggested but did not disclose the making of specific compounds, failed to provide support for those compounds as claimed in a later application. According to the court, "the invention disclosed in the appealed application was not in fact sufficiently disclosed in the parent application 'in the manner provided in the first paragraph of section 112 of this title'," and thus the claim was invalid.

[15] See, e.g., XX

[16] 358 F.3d 916 (Fed. Cir. 2004).

which the as-yet hypothetical drugs would operate, claimed in process terms. The Federal Circuit rightly affirmed the invalidity of the claims at issue. It wrongfully based its decision on the written description requirement. This case, easily decided under conventional enablement doctrine, stands as a fine example of unnecessary doctrinal proliferation.

*University of Rochester* is in no sense unique. Another case, *Enzo Biochem, Inc. v. Gen-Probe Inc.*,[17] deals directly with the issue of the deposit of microbiological materials claimed in a patent – an issue litigated, fleshed out, and settled in a series of cases in the 1970s and 1980s dealing with deposit under the enablement standard.[18] Another example is *Moba v. Diamond Automation, Inc.*,[19] a written description case that might easily be described in terms of conventional enablement law. To the same effect is *Reiffin v. Microsoft Corp.*, [20] a software patent case, in which the Federal Circuit stated that the purpose of the written description requirement is to "ensure that the scope of the right to exclude, as set forth in the claims, does not overreach the scope of the inventor's contribution to the field of art as described in the patent specification."[21] One familiar with the older enablement cases[22] could be forgiven for thinking, upon reading this passage, "I thought that's what enablement was all about."[23]

---

[17] 323 F.3d 956 (Fed.Cir. 2002). See also Regents of the University of California v. Eli Lilly & Co., 119 F.3d 1559 (Fed. Cir. 1997) (requiring a precise definition of a DNA sequence in the patent specification, for it to meet written description requirement). I would argue that (1) *Lilly* was actually an enablement case, and (2) it was wrongly decided under whichever doctrine governed it. See Janice Mueller, Berkeley Tech. L.J. (1998).

[18] See, e.g., In re Argoudelis, 434 F.2d 1390 (CCPA 1970) (microorganism claim). Such a deposit has been considered adequate to satisfy the enablement requirement of 35 U.S.C. § 112, when a written description alone would not place the invention in the hands of the public and physical possession of a unique biological material is required. See, e.g., In re Wands, 858 F.2d 731, 735-36, 8 USPQ2d 1400 (Fed.Cir.1988) ("Where an invention depends on the use of living materials ... it may be impossible to enable the public to make the invention (i.e., to obtain these living materials) solely by means of written disclosure."); In re Lundak, 773 F.2d 1216, 1220 (Fed.Cir.1985) ("When an invention relates to a new biological material, the material may not be reproducible even when detailed procedures and a complete taxonomic description are included in the specification."); see generally Berge Hampar, Patenting of Recombinant DNA Technology: The Deposit Requirement, 67 J. Pat. & Trademark Off. Soc'y 569, 607 (1985) ("The deposit requirement is a nonstatutory mechanism for ensuring compliance with the 'enabling' provision under 35 U.S.C. § 112.").

[19] 325 F.3d 1306 (Fed. Cir. 2003) (per curiam), cert. denied, 124 S. Ct. 464 (2003) (no written descriptin problem in specification, because each element of the claim was described fully in the specification in terms intelligible to one skilled in the art). Note that Judge Randall Rader has consistently dissented from recent developments in written description doctrine.See, e.g., Id. at 1322-27 (Rader, J., concurring) (arguing that "[b]y making written description a free-standing disclosure doctrine, [the Federal Circuit] produces numerous unintended and deleterious consequences").

[20] 214 F.3d 1342 (Fed. Cir. 2000).

[21] Id., at 1345.

[22] See, e.g., In re Fisher, 427 F.2d 833, 839 (CCPA 1970) (Disclosure standard "requires that the scope of the claims must bear a reasonable correlation to the scope of enablement provided by the specification to persons of ordinary skill in the art.").

[23] See also Union Oil Co. of Cal. v. Atlantic Richfield Co., 208 F.3d 989, 997 (Fed. Cir. 2000) ("The written description requirement does not require the applicant 'to describe exactly the subject matter claimed, [instead] the description must clearly allow persons of ordinary skill in the art to recognize that [he or she] invented what is claimed.'").

In deciding cases such as these, the Federal Circuit has performed a double disservice. First, it has violated the ancient principle of Occam's Razor, or intellectual parsimony: do not invoke two concepts when one will do. Second, by introducing a new doctrine into a settled area of law, it has opened the door to creative arguments and novel strategies that threaten basic, established principles. A lawyer handed a case that is a sure loser under established enablement principles only needs to reframe the issue as a written description question. Appealing to this novel and unformed body of law just might get the desired result. In the process, the meandering process of common law evolution – so prized when new issues demand case by case legal development – is unleashed unnecessarily. The resulting destabilization of established principles comes at a steep cost: the loss of certainty and predictability.

The second disservice is this: in its zeal to defend and expand written description doctrine, the Federal Circuit has diverted attention from an important issue. The fact is that some of the early written description cases – in particular, *Gentry Gallery* – did present facts that posed a challenge to traditional enablement doctrine. Here was a unique issue that really did call for some sort of doctrinal innovation. But the challenging issues presented by this new of type of case are now lost in the haze of confusion caused by applying written description virtually every time a section 112 issue is raised in litigation. The unique content of these new cases – the legitimate subject of new doctrinal expansion – is obscured in a much wider (and unnecessary) battle over what written description means. There is no reason for this to be so. As I describe in the next section, the novel issues raised by cases such as *Gentry Gallery* can be dealt with separately, without the need for doctrinal clutter. To see this, we have to now take a look at the nature of the novel issues these cases present.

## 2. Non-enablement Written Description Cases: "Misappropriation by Amendment"

The nub of the issue is how written description differs from what I have called classical enablement. To be frank, the courts have not been especially helpful in providing an answer to this crucial question. The perceived need for something beyond enablement originated in the early chemical cases; perhaps it was a function of the very liberal chemical enablement standard.[24] The various standards that have been announced

[24] In *In re Fried*, 312 F.2d 930 (C.A.F.C. 1963), for example, the parent specification whose content was the key to the case claimed a large family of chemical compounds in the usual manner, by virtue of a structural chemical formula and a "Markush claim." This type of claim, named after an old Patent Board opinion, recites that individual elements are "selected from the group consisting of" a list of elements. Given the large number of constituents in complex chemical structural formula, and the ability to claim each constituent as one among a list of variables, such a claim can easily encompass thousands of embodiments. The defect in cases such as *In re Fried* was a lack of guidance about how to choose among the long list of claimed variables:

> [W]hile appellant's parent application indicates that the 17-keto group of the steroid "compounds of the invention" therein may be converted to the corresponding steroids, it is clear, as pointed out by the examiner, that there is no disclosure of a specific method of preparation of the specific compounds claimed here and, as pointed out by the Board of Appeals, that there is no disclosure of a specific working example for preparing one compound here claimed. Since compounds here

all suffer from a lack of analytical rigor. The most common one centers on the notion of "possession." A claim will fail under the written description requirement if the inventor cannot show in the specification that the claimed subject matter was within his or her possession when the patent was filed. As distinct from enablement, the possession requirement seems to suggest that the inventor must have a firm grip on the claimed subject matter, as evidenced by the specification. This notion of "having hold of" is distinct from the traditional enablement test, which is concerned with what the specification teaches. When one reads the cases, it is apparent that the Federal Circuit has become convinced that an inventor can teach a great deal in a specification that he or she does not have a firm grasp of, or does not "consider as [his or her] invention." There are embodiments, in other words, that fall within the scope of a specification's teachings that cannot be said to be fair game for the patentee to claim. It is this notion – of the shortcomings of the enablement standard in maintaining a commensurate relationship between specification and claims – that has given rise to the written description revolution.

In some cases, a patentee has filed a specification that hones in on a particular set of embodiments, which are claimed in an initial application. Then one or more claims is amended to cover either a competitor's product or an item suggested by the prior art. In each of these, the original application, by failing to claim initially the technology later claimed in an amendment, signals that these embodiments are not particularly important or even relevant to the inventor. It is the actions of a third party that give them salience. When a third party introduces the disclosed but unclaimed variant, it suddenly acquires salience for the applicant. This seems unfair to the courts involved. By virtue of a claim amendment, a patentee attempts to encompass embodiments they did not envision as belonging to the real heart of the invention.

This type of unfairness might be described as "misappropriation by amendment." The patentee attempts to appropriate the effort of a competitor, or the contributions of a prior art reference such as an earlier technical article or patent. The unfairness is straightforward: these embodiments are more properly attributed to the labor of others. They are not rightfully within the ambit of an inventor's patent right.

---

> claimed are not named or identified by formula in the parent application, *they can find support only as choices are made between the several variables involved.*

312 F.2d at 936 (emphasis added). Note the conceptual similarity between this thought and the statement from the seminal *enablement* case, *The Incandescent Lamp Patent*, 165 U.S. 465 (1895):

> If, as before observed, there were some general quality, running through the whole fibrous and textile kingdom, which distinguished it from every other, and gave it a peculiar fitness for the particular purpose, the man who discovered such quality might justly be entitled to a patent; but that is not the case here. An examination of materials of this class carried on for months revealed nothing that seemed to be adapted to the purpose . . . .

165 U.S. at 475.

The doctrinal difficulty in these cases stems from a perhaps overgenerous view of enablement. It is arguable, on the basis of current understanding of the "undue experimentation" standard, that the later-claimed embodiments were in fact "taught" by the initial specification. In *Gentry Gallery*, for example, one might argue that a skilled furniture designer could easily deduce from the original disclosure that the seatback controls could be moved off the center console, as easily as they were moved in the initial design from their conventional location on the armrests. Traditional enablement law thus presents a deficiency: it cannot deal with cases such as this, where a general set of teachings enables a host of embodiments, but does not specifically mention or suggest particular variants which later come to light through the efforts of others. To guard against claim amendments that effectively misappropriate these others' efforts, courts apply the written description doctrine.[25]

The impetus behind these cases is surely right. Misappropriation by amendment is not to be condoned or encouraged. The filing of a patent application ought not to be a fishing license, enabling an applicant to troll the waters for promising variants painstakingly developed by others. This practice suffers from the same defect as the now-condemned practice of "submarine patents," made famous by the *Lemelson* case. There is simply no place in a self-respecting system for patents which permit this sort of blatant, rent-seeking gamesmanship.

But does that conclusion inevitably lead to the written description requirement as currently conceived and applied? In my opinion, not necessarily. I would argue that there is more suppleness in the fabric of conventional enablement doctrine than the Federal Circuit has so far appreciated. Judges were not condemned, in other words, by any ineluctable logic built into the "undue experimentation" standard, to search for an additional requirement. There was no need, I do not think, to identify a subset of the embodiments taught or enabled in a patent specification that could satisfy an additional, more stringent requirement. Rather than carve out a new requirement, courts could have recalibrated the old requirement.

### C. A Case Study of Software Patent Scope: *Lizardtech v. Earth Resources Mapping, Inc.*

After so much aggregate data and doctrinal description, some concrete details about patents and the software industry are in order. A recent case on software patent scope, *Lizardtech v. Earth Resources Mapping, Inc.*, provides an excellent vehicle. The decision in *Lizardtech* deals with the application of the written description requirement (described in Part II) to a software patent. In addition, I discuss the companies involved in the litigation, focusing on their business strategies and the way patents fit into those

[25] It is evident that the real culprit here is the overgenerous – and perhaps even dysfunctional –rules that allow virtually endless prosecution of patent applications. Others have taken aim at these rules, and some sort of reform is likely in the near future. See, e.g., Mark A Lemley & Kimberly A. Moore, *Ending Abuse of Patent Continuations*, 84 B.U. L. Rev. 63 (2004).

strategies. This will flesh out some of the points made earlier about the impact of patents on the software industry.

To anticipate a bit, companies now acquire at least a few patents as a matter of course, for various purposes. They may help attract financing. They may pave the way for new lines of business. They may be strictly defensive, providing only "freedom to operate." In any event, the ability of these companies to attract venture capital demonstrates once again that patents have not fundamentally harmed the software industry. In addition, the *Lizardtech* case demonstrates, consistent with recent research, that patents are used differently by different types of firms in the software industry. This should make courts and other policymakers hesitate to formulate software patent policy on the basis of assumptions about patents' impact on a monolithic software industry. Since patents mean different things to different firms, patent policy will affect different firms differently. Finally, since the patent was invalidated in *Lizardtech* as claiming too much given what was disclosed, it ought to assuage at least somewhat fears that software patents will inevitably be overbroad and therefore deleterious to the industry.

## 1. Background: The Companies Involved

In addition, patents may play affect a large industry, such as software, in disparate ways, because of the differing business strategies of firms. The parties in the *Lizardtech* case provide an example.

### a. Lizardtech

Plaintiff Lizardtech is a small software company that develops and sells a number of data compression programs for customers in several industries. Some of their customers must view and analyze large, complex maps, in an effort to locate and develop natural resources such as oil and gas. These maps are stored on computers in massive data files, occupying multiple terabytes. (A terabyte is one thousand billion bytes, or one thousand gigabytes. Most computer hard drives hold forty to one hundred gigabytes.) Companies that use this sort of data have workers scattered all over the globe; the ability to send files to each other makes them much more productive. But to store and transmit these large files, they must first be drastically compressed. Lizardtech has developed a number of computer programs for data compression that is both efficient and accurate, i.e., that loses little or no detail after it is transmitted and decompressed. The defendant in Lizardtech's patent suit, Earth Resource Mapping, sells competing products in the same industry.

Lizardtech also sells compression software to the publishing industry. Its most prominent customer in this area is *The New Yorker* magazine, which adopted Lizardtech's technology when it wanted to make its entire historical publishing output – eighty years of weekly magazines – available on DVD disks.[26] Marvel Comics, home of superheroes such as Spiderman and Thor, recently used Lizardtech technology to make available early issues of many of its most popular comic books. As with the mapping

[26] See http://www.lizardtech.com/files/doc/casestudies/The_New_Yorker_Case_Study.pdf.

applications, the appeal of the Lizardtech software is that it can compress the relevant files into small enough space to be manageable, yet restore essentially all of the detail to the stored images when they are decompressed for viewing.

Lizardtech was founded in 1992 on the basis of research performed at Los Alamos National Laboratory in New Mexico.[27] In 1996, the company was moved to Seattle "to raise venture capital."[28] The move seems to have paid off. By February 2000, LizardTech had already received two early rounds of financing; in that it month, it received a third round consisting of $15 million.[29] At this point, LizardTech had raised a total of approximately $20 million.[30] This round of financing was intended to "enable LizardTech to develop its wavelet-based imaging language, called MrSID(TM) (Multiresolution Seamless Image Database) for new platforms and applications."[31]

LizardTech received $25 million of venture capital financing, primarily from Mitsubishi Corp., during the week of 11/20/2000.[32] "Other investors in this round include Oak Investment Partners, Encompass Ventures, SeaPoint Ventures, Digital Partners, Summit Ventures, Kirlan Ventures and Zeron Group."[33] At this point, CEO John Grizz Deal "refused to disclose sales figures," he reported quadrupled revenues, with "revenues of $2 million for fiscal year 1999."[34] The number of employees also grew from 50 to 200. However, the same report said, "LizardTech, which has yet to turn a profit, has raised $45 million to date."[35] Most recently, LizardTech was then acquired by Celartem Technology USA, Inc., the U.S. arm of a Japanese software holding company, for $11.25 million in cash.[36]

On the product side, early industry response to LizardTech's DjVu technology seems to have been positive, as indicated by the tone of a technical discussion website

---

[27] Archives.seattletimes.nwsource.com, Files Slither Down in Side with LizardTech Product, http://archives.seattletimes.nwsource.com/cgi-bin/texis.cgi/web/vortex/display?slug=btinterface19&date=20050919&query=lizardtech.
[28] Id.
[29] Internetnews.com, VC Buzz $275.4+ Million in Today's Deals, http://www.internetnews.com/bus-news/article.php/298351#lizard (February 2, 2000). The primary investor in February 2000 was a venture capital firm known as Oak Investment Partners, but two others – SeaPoint Ventures and Encompass Ventures – also participated. "Encompass Ventures, along with Kirlan Ventures, Summit Ventures and Staenberg Private Capital, led earlier private investments in LizardTech," i.e., the first two rounds, prior to February 2000. Id. These earlier rounds totaled roughly $5 million. Id.
[30] *Id*.
[31] *Id*.
[32] Seattlepi.com, Seattle Post-Intelligencer: Venture Capital Notebook, http://seattlepi.nwsource.com/venture/funding.asp?company=LizardTech.
[33] *Id*.
[34] *Id*.
[35] Id. Given the timing, it might be accurate describe Lizardtech as one of many companies caught up in the "dot com" boom or bubble. A subsequent article states that LizardTech laid off 40 people from its workforce of 80 in May 2002 "in an attempt to bring head count in line with revenue." Seattlepi.com, Seattle Post-Intelligencer: Venture Capital Notebook, http://seattlepi.nwsource.com/venture/layoff.asp?id=439 (April 2, 2002).
[36] GISmontor.com, Newsletter Archive, http://www.gismonitor.com/news/newsletter/archive/062603.php (June 26, 2003).

from the year 2000.[37] Over the next few years, the company seems to have focused on developing its image compression technology but did not actually begin commercialization until 2004.[38] Then it began entering into license agreements with various "value added" resellers of its geospatial imaging technology as well as integration with various piece of imaging and archival software.[39]

In 2004, LizardTech stated that it had its strongest fiscal year "for its geospatial line of products," primarily a program called "GeoExpress." It reported a 34 percent increase in sales over 2003 (but gave no dollar figure).[40] In 2005, LizardTech launched several new versions of software and had some measure of commercial success in deals with magazine companies and distributors, such as The New Yorker and North American Publishing Company.[41] This is still a fairly small company however; it had $29 million in sales in 2005, a 52 percent increase in compound annual growth since 2001.[42] As of 2005, the company had 170 employees in Seattle, Portland, New York, San Rafael (CA), and England.[43] LizardTech has continued to create new versions of its GeoExpress software, including a new version in August 2006.[44]

### b. Earth Resources Mapping

Earth Resource Mapping Pty. Ltd. received a $1,139,450 grant from Australia's Industry Research and Development Board to help finance its world-first Image Web Server product.[45] "The R&D Start Program, administered by AusIndustry, provides funding for Australian companies to undertake R&D and related activities and aims to increase the number of R&D projects with high commercial potential, foster innovation in Australian business and encourage greater commercialization of outcomes from R&D projects."[46]

Whereas LizardTech seems to deal broadly in digital image storage and distribution, ERM seems focused specifically on geoimagery.[47] "The company goal has

---

[37] Slashdot.org, A New Web Image Format, http://slashdot.org/articles/00/11/21/2312220.shtml.
[38] LizardTech.com, Press Room – Press Releases, http://www.lizardtech.com/press/news.php?archive=2004 (2004).
[39] *Id*.
[40] LizardTech.com, LizardTech Reports Strongest Fiscal Year for its Geospatial Solutions in 12-Year Company History, http://www.lizardtech.com/press/news.php?item=08-03-2004 (2004).
[41] LizardTech.com, Press Room – Press Releases, http://www.lizardtech.com/press/news.php?archive=2005 (2005).
[42] Archives.seattletimes.nwsource.com, Files Slither Down in Side with LizardTech Product, http://archives.seattletimes.nwsource.com/cgi-bin/texis.cgi/web/vortex/display?slug=btinterface19&date=20050919&query=lizardtech.
[43] *Id*.
[44] LizardTech.com, Press Room Homepage, http://www.lizardtech.com/press/.
[45] AusIndustry.gov.au, Perth company's R&D Start project a big hit with image processing, http://www.ausindustry.gov.au/content/content.cfm?ObjectID=CF0D93E3-36B6-4678-8164B6F174505356&L3Keyword=benefit (July 28, 1999).
[46] *Id*.
[47] *See* Giscafe.com, Earth Resource Mapping – Corporate Listing, http://ecat.giscafe.com/corpprofile.php?vendor_id=3000512. (GISCafe.com is a professional association

always been to make image processing easier to use as a tool, so that professionals of all skill levels and disciplines can effectively utilize the power of geoprocessing and remote sensing technologies."[48] In addition to its image processing tool, ER Mapper, the company offers "ER Radar for radar and SAR data processing."[49]

ER Mapper is used by professionals in a wide range of industries including oil and gas, mining, forestry, defense, agriculture, environmental, state and local government, and telecommunications. Anyone managing the earth's natural resources or the urban infrastructure has an application.

ER Mapper is available only from resellers who market to vertical industry types and geographic regions on a non-exclusive basis. These resellers are experts in the application of remote sensing to their industry and provide local support and training. Please contact us if you have an interest in becoming an ER Mapper reseller.[50]

An ERM press release following the "demise of Mapping Science resulting from litigation by LizardTech" gives some sense of the state of the market at the time.[51] According to ERM CEO Stuart Nixon, ERM intended to continue a push for open standards in GIS imagery, a stance that Mapping Science had taken as well.[52] ERM substantiated this claim by supplying a range of licenses for their ECW JPEG 2000 SDK, two of which were free. "Only strictly commercial ventures who want unlimited compression in commercial products have to pay a once-off and royalty free payment for the SDK, and when they do they get the complete source code."[53] The press release also stated that pricing schemes differed drastically between the companies in that LizardTech charged a 'per megabyte' price for image compression and management, while Mapping Science and ERM did not.[54]

ERM reported 95% growth in 1994.[55] In 2006, it added such customers as Shell Exploration and Production, The Sidwell Company, Terralink International, and several universities.[56]

## 2. File Formats, Business Strategy, and Patents: A Quick Case Study

---

for companies in the Geographic Information Systems market and lists both Earth Resource Mapping and LizardTech as members.)

[48] *Id*.
[49] *Id*.
[50] *Id*.
[51] Geoplace.com, Earth Resource Mapping – Press Release, http://geoplace.com/pressrelease/detail.asp?id=6413.
[52] *Id*.
[53] *Id*.
[54] *Id*.
[55] *Id*.
[56] ERmapper.com, Press Releases.

Lizardtech's business strategy parallels that of Adobe, Inc., sponsor of the popular "personal document format" (pdf) file format. Adobe gives away, at no cost, many, many copies of its basic Acrobat document viewing software. The company does this to "seed" the market for its profitable document design software, which is optimized to work well with pdf format files. Adobe has understood the value of selling software that works seamlessly with files stored in a popular format. In economic terms, it seeks to take advantage and profit from the "network externalities" effect: the fact that as more people use pdf documents, pdf-compatible software becomes more valuable.[57]

Like Adobe, Lizardtech sponsors a software file format which, in its basic version, is free to all comers. Its format, designed to compress and store images and scanned documents more efficiently and with less degradation that the pdf format, is called DjVu ("déjà vu"). As with Adobe, Lizardtech sells sophisticated software designed to work seamlessly with files stored in the DjVu format.[58] Its business model is based on the same idea as Adobe's: the more people that store files in the DjVu format, the more valuable Lizardtech's software will be.

ERM has also introduced its own competing file format standard, called ECW JPEG 2000, a variant on the widely-used JPEG image storage format.[59] Although it is difficult to obtain data, the DjVu standard seems to be more popular. This may stem in part from the fact that it is older, having been developed at AT&T in the late 1990s and licensed and maintained by AT&T until Lizardtech took over in the year 2000.[60] (Interestingly, Lizardtech took assignment of this AT&T patent in the same month it received a large infusion of venture capital.)

### 3. Patent Portfolios of the Two Firms

The following table shows the patents held by the two firms involved in the *Lizardtech* case. All the patents listed under "Lizardtech" were eventually assigned to Lizardtech; but as the table shows, a number were first assigned to other entities.

***LizardTech***

| Pat. No. | Inventor | Original Assignee | Filed | Issued |
|---|---|---|---|---|

[57] See, e.g., Carl Shapiro and Hal R. Varian, Information Rules: A Strategic Guide to the Network Economy 189 (1998) (describing free distribution of Adobe Acrobat, i.e., pdf, software, to promote network externalities). See also Robert M. Grant, Contemporary Strategy Analysis: Concepts, Techniques and Applications 351 (2002) (in chart describing companies that control industry standards, listing Adobe's pdf document format as an example).

[58] See http://www.lizardtech.com/products/doc/professional.php (describing "professional" version of DjVu image handling software); http://www.lizardtech.com/products/doc/enterprise.php (describing "enterprise" -- more advanced and more expensive – version of "professional" version).

[59] Specification Sheet, ECW JPEG 2000 SDK [Software Developer Kit] 3.1, 21 January 2005, available (ironically, perhaps, in pdf format) at http://www.ermapper.com/ecw.

[60] See uspto.gov, assignment search, reel number 011089/0349, Recorded: 11/09/2000 (recording assignment of U.S. Patent 5,900,953).

| | | | | |
|---|---|---|---|---|
| 5,130,701 | James M. White, Vance Faber, Jeffrey S. Saltzman (Los Alamos) | US Dept. of Energy → UC Regents | 05/12/89 | 07/14/92 |
| Digital color representation | | | | |
| *This patent has undergone numerous assignments, mergers, and security agreements.* | | | | |
| 5,467,110 | James M. White, Vance Faber, Jeffrey S. Saltzman (Los Alamos) | US Dept. of Energy → UC Regents | 09/28/90 | 11/14/95 |
| Population attribute compression | | | | |
| *This patent has undergone numerous assignments, mergers, and security agreements.* | | | | |
| 5,710,835 | Jonathan Bradley (Los Alamos) | UC Regents | 11/14/95 | 01/20/98 |
| Storage and retrieval of large digital images | | | | |
| 5,900,953 | Leon Bottou, Yann Andre Lecun | AT&T | 06/17/97 | 05/04/99 |
| Method and apparatus for extracting a foreground image and a background image from a color document image | | | | |
| App # 100069999 | Vance Faber, Randall L. Dougherty | LizardTech, Inc. | 12/03/01 | |
| Method for lossless encoding of image data by approximating linear transforms and preserving selected properties for image processing | | | | |

***Earth Resource Mapping***

| Pat. No. | Inventor | Assignee | Filed | Issued |
|---|---|---|---|---|
| 6,201,897 | Stuart Nixon (San Diego) | Earth Resource Mapping (San Diego | 11/09/98 | 03/13/01 |
| Transformation and selective inverse transformation of large digital images | | | | |
| 6,442,298 | Stuart Nixon (San Diego | Earth Resource Mapping, Ltd. (West Perth, AU) | 10/26/00 | 08/27/02 |
| Transformation and selective inverse transformation of large digital images | | | | |
| *This is a continuation of the '897 patent. Together, these cover the allegedly infringing technology at issue in the* Lizardtech *suit.* | | | | |
| 6,633,688 | Stuart Nixon, Simon Cope (Marangaroo, AU), Mark Sheridan (Kewdale, AU) | Earth Resource Mapping, Inc. (San Diego) | 04/28/00 | 10/14/03 |
| Method system and apparatus for providing image data in client/server systems | | | | |

## 4. Patents and Business Developments

Lizardtech's February, 2000 venture capital infusion followed by two years the issuance of the '835 patent it later deployed in litigation against ERM. The general pattern of patent issuance and later-round venture financing is consistent with the recent aggregate empirical research of Professor Ronald Mann.

There is some direct evidence that the patents involved in this case facilitated financing. Security interests in several Lizardtech patents were recorded in 1999; and a release form this security interest was recorded later, in 2003.[61]

## 5. The Court's Decision in *Lizardtech*

The court in the *Lizardtech* case invalidated claim 21 in Lizardtech's '835 patent that was asserted against ERM.[62] The claim, according to the court, failed to comply with the written description requirement, because although a specific algorithm was recited in the patent specification, the asserted claim had been broadened (by dropping a limiting feature present in the algorithm described in the specification). To see why, it is important to understand Lizardtech's algorithm, and how it differed from the one used by ERM.

### a. The Technology: Data Compression Algorithms

Computers represent graphic images – such as maps – as long strings of numbers. Each number represents a distinct value; for example, the color of an individual computer screen "picture element" or "pixel." One way to transmit an image is to simply assign a number to each pixel location and put it in a large table. Because there are so many pixels on a screen, this can quickly use up the available memory in a computer. The problem is exacerbated considerably when one want to store a huge image – say, a detailed, high-resolution map of an entire county or state. Very few standard computers could hold a single file that stored all this data.

To address this problem, software engineers use various types of compression algorithms. These are mathematical operations that take raw data (such as pixel colors) as their input, and produce as their output a compressed version of the data. There are a number of ways to do this mathematically. One is to take all pixel color data that is very close to zero (e.g., close to white in color), and simply set it equal to zero. Then, all the non-zero data can be stored in a table, along with information about the total number of

---

[61] Reel/frame 009958/0719, Recorded: 05/21/1999 (recording of security interest taken by Silicon Valley Bank in US Patents 5,130,701; 5,467,110; and 5,710,835 (the patent involved in the *Lizardtech v. ERM* case). A "release" from this security interest was later recorded on the same three patents. See Reel/Frame 014409/0528 (recorded August 22, 2003).

[62] Claim 21 of U.S. Patent 835 reads as follows

Claim 1 reads as follows, in part:

pixels. When this table is “unpacked” – decompressed – any value in the table that has no data associated with it can be treated as a zero. In this fashion, a great deal of storage space is saved. In effect, every zero value gets stored at very low “cost” in terms of space in the table. The decompressed image will not be a perfect, exact copy of the original, but it will be “close enough.”

This is a very simple example of data compression. Lizardtech’s algorithm was much more sophisticated. It was based on the idea of a mathematical transform. Transforms work by repeatedly performing a mathematical operation on a string of data, converting it into a format that can be stored in a smaller amount of space. For example, in a very short string of only two numbers, one can take their average, and then record the difference between them. (These numbers, the result of the mathematical transform operation, are referred to as coefficients.) These coefficients can typically be stored in a smaller amount of space. The average of two numbers just one number; and the difference between two numbers is in practice often smaller than the original numbers in the string. (Smaller number, less storage). The original data can be re-generated from the compressed, stored version, by in effect performing the inverse of the mathematical operation used in the compression transform.

Sophisticated techniques start from this simple concept. Increased compression can be achieved a number of ways. One common technique is to perform an operation – such as the “take the average and compute the difference” operation just described – and then perform it again on the first result. That is, in the parlance of the field, perform the operation *recursively*. This can result in a very significant degree of compression. Of course, each time the operation is performed, a new set of coefficients results. And to re-generate the initial numbers, the process must be recursively “undone.”

Transforms can employ another mathematical concept called a “filter.” Filters are themselves strings of numbers in a specific format – technically, arrays – that perform operations on a series of numbers in a data string that is being compressed. A filter can be “moved down” a string of numbers, performing an operation on every second, third, fourth, etc., number in the string. The operation can be to multiply each “sampled” value in the string by a certain number, and then multiply the “neighboring” numbers by other numbers in the filter array. This will produce a smaller array that in effect samples various points in the data to be compressed, but also captures mathematically a rough sense of the value of neighboring numbers as well.

The specific transforms that Lizardtech and ERM use in their algorithms are called Discrete Wavelet Transforms, or DWTs. DWT works by splitting data using two filters. The low pass filter uses filtering values and techniques that retain the low frequency data, i.e., rough data about large groupings of values. In a large image file, this might represent broad areas of color. The high pass filter retains the high frequency data, i.e., high-variation data with a lot of relatively large differences between adjacent values. In image data, this might represent data about the edges of map features, or textures such as different elevations.

All of the data in a map file necessarily falls into one of these categories, so nothing is lost in the splitting process. First, the filters are each run over the array in the row direction to create two new arrays of high band and low band coefficients. Because this effectively doubles the amount of data, creating two new arrays from the original array, half of this data can be deleted with no loss. This is called "downsampling." After downsampling, the filters are each run over each of the arrays calculated in the previous step, but in the column direction. This creates a total of four sub-bands: low-low, low-high, high-low, and high-high. The arrays are downsampled once again to keep the total amount of data equal to that in the original image. At this point, no data has been lost. Rather, the image has been transformed into a different encoding.

The data compression in DWT comes when this reconfigured array is recursively transformed, and data with values close to zero are eliminated.

One nice property of DWT is the ability to display the image in lower resolution. In order to do this, the algorithm can display one of the low-low decompositions instead of reassembling the image. While some of the details stored in the other decompositions will not appear, the broader contours of the image will be visible.[63] Lower and lower resolutions can be displayed by showing the low-low decomposition of successive DWT iterations.[64]

A major problem with the DWT is that it requires storing the entire array in computer memory (such as RAM) at one time, which is impossible for large images. The obvious solution to this problem is to split the image into "tiles" and calculate the DWT on each of those tiles. However, this creates a new problem. Because there is no access to the rest of the image when calculating DWT on a tile, the values outside the tile are set to 0. That is, the "filter" can only incorporate the data in a given tile into its calculations because it does not "see" the entire row or column as it would if the whole image were present. This creates edge artifacts throughout the image – false representations of the data caused when the array of coefficients is decompressed.

Lizardtech's patented method deals with this problem by using the fact that DWT is linear to split the calculation of the DWT coefficients. The method first splits the image into tiles, as above. Then it calculates the DWT for these tiles starting from one corner, progressing in diagonal stripes. This part of the process is taught by the prior art.

---

[63] This also provides a good metaphor for understanding conceptually what DWT is doing. In effect, an iteration of DWT separates the data of an image into three categories. First, the broad contours, the "most important" data, is stored in the low-low decomposition. Second, the fine-grained details, the "less important" data, is stored in the other three decompositions. Third, the finest-grained details, the "least important" data, which consists of the numbers very close to 0, is deleted for the purpose of compression.

[64] This is an extremely useful technique, the best example of which is the program Google Earth. I'm not certain what transform they use, but the idea is the same. When the user wants to see a map, Google only needs to send the coefficients for the low-low decomposition at that resolution for that area rather than sending the data for the entire map, thus yielding a large increase in speed and a large decrease in bandwidth. Moreover, "zooming in" merely requires sending another set of coefficients. Another benefit is that the client computer, rather than the server computer, does the calculation to reassemble the image. *See* http://earth.google.com/.

The "trick" is what it does in addition to this. The prior art centers the filter on each element in the tile. The Lizardtech method also centers the filter on elements *outside the tile* that have been set to 0. This is a means of calculating not only the coefficients for the tile loaded into memory, but also the effect of that tile on later calculated coefficients. In effect, the sums are broken up and calculated in pieces. This results in a seamless transform that is exactly the same as if DWT had been run on the entire, unbroken image. The basic DWT calculation, centering the filter on a pixel and calculating the sum of products, is a linear operation. It is the sum of products. Therefore, it doesn't matter whether we add the values in that sum in the same order or calculate part of the sum now and part of the sum later. The resulting coefficient will be the same as long as all parts of the sum are included at some point in the algorithm. Thus, the Lizardtech method is effective because it breaks up the sums but still manages to include all of the parts of the sum.

**b. The Court's Decision**

Lizardtech claimed a number of embodiments of its data compression process. Claim 1 is both representative and important in the court's decision. It reads:

> 1. A method for selectively viewing areas of an image at multiple resolutions in a computer ... comprising the steps of:
>
> > storing a complete set of image data array I(x,y) representing said image ...;
> >
> > defining a plurality of discrete tile image data $T_{ij}(x,y)$ subsets, where said complete set of image data I(x,y) is formed by superposition of said discrete tile image data $T_{ij}(x,y)$;
> >
> > performing one or more discrete wavelet transformation (DWT)-based compression processes on each said tile image data $T_{ij}(x,y)$ in a selected sequence to output each said discrete tile image data $T_{ij}(x,y)$ as a succession of DWT coefficients ...;
> >
> > ***maintaining updated sums*** of said DWT coefficients from said discrete tile image $T_{ij}(x,y)$ to form a seamless DWT of said image and storing said sums in a first primary memory location of said computer;
> >
> > ***periodically compressing said sums*** and transferring said compressed sums to a second secondary memory ...;
> >
> > selecting a viewing set of said image data array I(x,y) to be viewed at a desired resolution; ....[65]

[65] '835 Patent, quoted at 424 F.3d 1336, 1340.

The highlighted terms figure prominently in the ‘835 patent’s specification. They represent the incremental, “creeping” calculation of coefficient values as arrays for individual “tiles” are transformed. They are central to Lizardtech’s approach, as they represent the key step differentiating Lizardtech’s algorithm from prior art DWT techniques.

Without doubt, the accused infringer ERM did not infringe claim 1. ERM had a completely different technique for using DWT to transform data; the ERM technique does not break the data into tiles. Thus it does not need to “maintain updated sums” for tiles, or to “periodically compress[] said sums.” The ERM technique was described in general terms in a declaration at trial by company founder and chief researcher Stuart Nixon:

> ERM Uses a Continuous Sliding Window Approach: This technique never breaks up or tiles the image. so it does not introduce any edge artifacts. Instead, it relies on the critical observation that contrary to what was previously thought, the DWT process does not need to generate the entire intermediate images before generating the output sub-band images. The newly-patented ERM method uses this observation to perform a standard prior art DWT technique, but does so by structuring the data flow to ensure that only the minimum amount of data required is stored in memory at any one time.[66]

But Lizardtech had another claim – claim 21 – to fall back on in its infringement suit. According to the court, “[c]laim 21 of the '835 patent is identical to claim 1 except that it does not contain the ‘maintaining updated sums’ and ‘periodically compressing said sums’ limitations [of that claim].”[67] That is, it eliminated the key phrases that placed ERM’s technique outside the bounds of the patent claim. This led the trial court to find the patent invalid under the written description requirement. And it led Lizardtech to appeal.

The Federal Circuit focused on the omission of the “maintain updated sums” limitation as critical:[68]

> The trouble with allowing claim 21 to cover all ways of performing DWT-based compression processes that lead to a seamless DWT is that there is no support for such a broad claim in the specification. The specification provides only a single way of creating a seamless DWT, which is by maintaining updated sums of DWT coefficients. There is no evidence that the specification contemplates a more generic way of creating a seamless array of DWT coefficients.

This seems absolutely correct on the facts. The idea of “maintaining updated sums” refers to the idea that calculation of the sum is split up. Thus, the Lizardtech

---

[66] Declaration of Stuart Nixon at 14, Lizardtech, Inc. v. Earth Resource Mapping, Inc., No. C99-1602C (W.D. Wash. 2000).
[67] 424 F.3d at 1343.
[68] Id.

method must maintain updated sums for each coefficient until all parts of the sum have been calculated. Because Lizardtech did not describe all ways to obtain a seamless DWT, this claim could cover a host of methods that are not supported by the specification. The only example in the specification maintains updated sums, and there is no indication of how the algorithm might be performed without doing so.

The court admitted in framing its holding that – consistent with the argument earlier in this paper – enablement might just as well explain this outcome as the written description requirement:

> Those two requirements [enablement and written description] usually rise and fall together. That is, a recitation of how to make and use the invention across the full breadth of the claim is ordinarily sufficient to demonstrate that the inventor possesses the full scope of the invention, and vice versa. This case is no exception. Whether the flaw in the specification is regarded as a failure to demonstrate that the patentee possessed the full scope of the invention recited in claim 21 or a failure to enable the full breadth of that claim, the specification provides inadequate support for the claim under section 112, paragraph one.[69]

The question recurs: why two requirements, instead of one? This aside, the decision seems quite apt. It also represents an answer, in part, to many of the early fears about software patents.

## 6. Reprise: The Role of Patents in Competition Between Lizardtech and ERM

The opinion in *Lizardtech* does not state why Lizardtech brought suit against ERM. What is clear from the business press and the other available evidence is that the firms compete vigorously in the market for geospatial data handling software. It appears that the patent suit was just another front in a multi-dimensional battle for market share and corporate survival – business as usual, as it were.

The very unremarkable nature of the patent infringement suit in this case points up how routine patents and all their trappings (including the occasional infringement suit) have become in the software industry. Quite contrary to all those predictions in earlier years, patents are evidently not strangling either of these companies, or the industry sector in which they operate. Quite the opposite. Patents may have played a role in helping one or both of these firms attract investment capital. (The record on ERM is sketchy is in this regard, but the evidence presented earlier for Lizardtech is convincing.) And they are surely playing a role in the age-old battle over "shelf space" in this competitive industry. But, contrary to predictions early and late, neither company is asserting its patents willy-nilly throughout the industry; they do not seem to represent a massive transaction cost burden on the industry. And neither firm is a behemoth either; both are on the small side, by U.S. corporate standards. This belies early predictions that the "patent overhead" – the costs of acquiring and administering patents – would drive

[69] 424 F.3d 1345.

small firms out of the software industry. Whatever else patents may have done, they have not shut down these two small, innovative companies. Nor, to judge by concentration statistics and other data, have they done so in other sectors of the software industry.

### a. Patents and "Open Standards"

One interesting feature of the two firms' strategies is the widespread licensing of competing data compression protocols or file formats: DjVu, in the case of Lizardtech, and ECW JPEG 2000 for ERM. As mentioned, the strategy here is dictated by the realities of network externality-driven businesses. The companies reason, on the basis of examples like Adobe's pdf format, that making its compressed file format a standard will help it make more money in the long term.

The interesting point to note is that both formats appear to be covered by patents. What this means is that both firms are choosing to *give away* to many people free copies of patented software. This is surely not what the early critics of software patents predicted. And indeed, many contemporary devotees of free, open source software are quite wary of software patents as well. This is due in part to fears that software patents will clog the arteries of commerce and innovation. Critics point out that the software field grew up, after all, without the specter of patents, and that the higher transaction costs that presumably accompany the advent of patents for software inventions can only cause harm.

This may yet come to pass. But the fact remains that many customers are receiving free copies of these firms' software, despite the fact that they are patented. This business strategy suggests that dichotomous thinking among open source fans and other software patent critics may be misguided. The decision to patent, for these firms at least, is separate and distinct from the question of whether to adopt an open software licensing strategy. The automatic pairing of "patented" with "proprietary/closed" licensing or dissemination strategies does not apply here.

A careful look around reveals that these firms are not unique in this respect. Obtaining patents represents a strategic choice quite distinct from whether to make a technology available to some or all users. There are many reasons why a firm might patent software and still license freely to many users. Much of the thinking is driven by the dynamics of network industries. Patents may be held in reserve, to be deployed (if at all) against direct competitors, while being essentially "waived" via open licensing to other users.

The table below tries to capture the difference between patent strategies and open vs. closed licensed strategies. Most of the examples will be familiar. The fourth quadrant, with the example of the "Encase" forensic disk analysis standard, may not be. Encase is a software technique for making a precise copy of a hard drive whose contents are to be studied for "forensic" purposes (i.e., to obtain evidence of the disk's contents in a legal proceeding). The creator of the standard, Guidance software, uses the Encase protocol in its own products, but does not license it to others. This file format has not been patented

(as yet, anyway);[70] the company seems to maintain it as a trade secret. It thus serves as an example of an unpatented technology whose owner has chosen a closed licensing strategy.

**Table 1: Strategy Grid**

| | **OPEN** | **CLOSED/PROPRIETARY** |
|---|---|---|
| **Patents** | Adobe Acrobat; Lizardtech DjVu format | Apple iTunes music format |
| **No Patents** | Open source software, e.g., Linux Operating System | Encase Forensic Disk Analysis software (http://www.guidancesoftware.com/products/ef_index.asp) |

The point of the table is simple: patents can and often do coexist with open licensing strategies. Thus there is no direct link between obtaining software patents and locking up technology against all comers. Indeed, a single firm can both obtain and enforce patents (against some), and freely license those patents (to others). It might even be said that the choice to "selectively waive" the property right is one of the key advantages of obtaining patents.

## IV. Conclusion

Entry and competition are robust in the software industry. Firms are obtaining patents to assist in financing, and to use as strategic weapons in ongoing battles over market share. Courts are limiting software patents in ways calibrated to adjust the value of the property right to the quantum of technical contribution represented in the patent's specification.

Many features of the system can no doubt be improved. (Patent trolls and poor quality patents come to mind immediately). It is surely not the best of all possible worlds, but this picture does not anything like a fiasco to me. And that is a reassuring thought.

[70] Guidance does have one patent, but it is unrelated. See McCreight et al., "Enterprise computer investigation system," U.S. Patent 6,792,545, issued Sept. 14, 2004, assigned to Guidance Software, Inc.