An Empirical Look at Software Patents

James Bessen*
Research on Innovation and MIT (visiting)

Robert M. Hunt**
Federal Reserve Bank of Philadelphia

PRELIMINARY VERSION. NOT FOR CITATION.

December 15, 2002

Abstract: U.S. legal changes have made it easier to obtain patents on inventions that use software. Software patents now comprise 15% of all patents. Compared to other patents, software patents are more likely to be owned by large U.S. firms. Most are assigned to manufacturing firms; only 6% to software publishers. Our regression analysis finds that software patents have become a cheap form of appropriability. This cost advantage, not "technological opportunity," accounts for their increased use. Also, software patents *substitute* for firm R&D rather than complement it. Their growth is associated with lower R&D, consistent with strategic "patent thicket" behavior.

Keywords: Software, Patents, Innovation, Technological Change

JEL classification: O34, D23, L86

* Corresponding author. jbessen@researchoninnovation.org

** Ten Independence Mall, Philadelphia, PA 19106. Phone: (215) 574-3806. Email: bob.hunt@phil.frb.org

Thanks to Peter Bessen of May8Software for providing a software agent to acquire our patent database and Annette Fratantaro for her work with the Compustat data set. The views expressed here are those of the authors and do not necessarily represent the views of the Federal Reserve Bank of Philadelphia or the Federal Reserve System.

"The IBM patent portfolio gains us the freedom to do what we need to do through cross-licensing—it gives us access to the inventions of others that are key to rapid innovation. Access is far more valuable to IBM than the fees it receives from its 9,000 active patents. There's no direct calculation of this value, but it's many times larger than the fee income, perhaps an order of magnitude larger." --Roger Smith, IBM assistant general counsel, Intellectual Property (IBM, 1990)

Introduction

The patentability of software-related inventions changed dramatically over the twenty years ending in 2001. Prior to 1981, court decisions excluded mathematical algorithms as patentable subject matter. Although systems that included software could be patented, inventions where the novelty lay exclusively in the software component could not be patented. This interpretation began to change after the Supreme Court decision in *Diamond v. Diehr* in 1981. Thereafter, a series of court decisions and administrative decisions gradually lowered the requirements for obtaining patents on software-related inventions (Hunt 2001). Additionally, other court decisions lowered standards for obtaining patents in general, while strengthening aspects of patent enforcement (Hunt 1999a, 1999b). Although these changes affected all patents, they may have had a disproportionate impact on software inventions.

This paper explores two aspects of the changes in software patenting during this period. First, we explore general characteristics of these patents. Using a broad definition of "software patent," we assemble a comprehensive database of all such patents to explore their numbers and characteristics. We find that over 20,000 software patents are now granted each year, comprising over 15% of all patents. Compared to other patents, software patents are more likely to be assigned to firms than to individuals, especially larger U.S. firms. They are also more likely to have U.S. inventors and they receive more citations from subsequent patents. Software patents are assigned to firms in a wide variety of industries. Most are assigned to manufacturing firms and relatively few are actually assigned to firms in the software publishing industry (SIC 7372).

The second part of the analysis explores the economics behind these changes. We look at the economics behind the rise in software patenting and we look at the relationship between this rise and firm R&D spending.

To do this we use an analytical framework based on the cost of appropriability. A large literature has used patents as a measure of inventions (see Griliches, 1990, Jaffe and Trajtenberg, 2002). We take a somewhat different approach, formally modeling the firm's decision to obtain patents as an economic optimization problem. As is well-known, the number of patents a firm obtains does not exactly equal the number of inventions embodied in its products. Firms often choose to protect some inventions with trade secrecy alone rather than using patents (Levin et al, 1987, Cohen et al, 2000). This occurs because patents are costly—patents incur application and renewal fees, legal and search costs, and they may often involve R&D beyond what is required just to develop a product. On the other hand, firms sometimes obtain more patents than they actually use; for instance, firms may build a "thicket" of patents on similar technologies to limit the ability of competitors to enter the market (Bessen, 2002, Hall and Ziedonis, 2001). In general, firms can increase their appropriability—the percent of potential profits that the firm actually earns—by obtaining more patents.[1] So there is a tradeoff between appropriability and patent cost that determines the optimal number of patents for each firm.

In this framework, the changes in software patenting can be viewed as changing the quality-adjusted cost of obtaining a patent—lower standards for patenting reduced the legal costs and the amount of additional R&D needed specifically to obtain a patent. At the same time, broader patent scope and stronger enforcement increased the appropriability each patent delivered, reducing the quality-adjusted cost.[2] If software patents cost less to obtain, say, after 1990, then this would have reduced the average cost of acquiring a portfolio containing software patents, increasing optimal portfolio size. Moreover, firms able to obtain relatively more software patents would have benefited from a relatively lower average cost of patenting.

Using this framework, we first examine the degree to which the relative increase in software patenting can be attributed to reductions in the price of patenting as opposed to

[1] By potential profits we mean all the rents a firm could earn if it could perfectly protect its inventions.

[2] We mean quality-adjusted in a particular sense here - the amount of appropriability that can be obtained from a given expenditure on patents. We are agnostic about the quality of patent examination or of the underlying invention.

greater "technological opportunity" for software-related inventions. By technological opportunity, we mean that firms may have found it more profitable to use software in designing new products. They may have thus obtained more software patents not because the cost of patenting declined, but because they were resorting more often to software, rather than hardware, to solve technical problems.

In a regression analysis, greater technological opportunity is associated with a larger profit stream, while a decline in the cost of patenting increases the rate of patenting independently of profits. We find that prior to 1990, software patents tended to cost more than other patents and firms obtained patents on software-related inventions in spite of this cost (i.e., for reasons of technological opportunity). After 1990, however, we find that software patents cost less than other patents and they are associated with proportionately larger patent portfolios for any given level of profits. In other words, changes in the cost of patenting software largely explain the widespread use of these patents after 1990.

The second question we examine using this framework is whether R&D and software patents are complements or substitutes. In a simple model of patents, patents should complement R&D. That is, a drop in the "price" of patenting should increase the optimal level of appropriability; this should, in turn, increase the incentives to invest in R&D, resulting in greater R&D investment.[3]

However, if firms pursue patent portfolio strategies, then patents may *substitute* for R&D. This occurs for two reasons. First, firms can benefit from "knowledge spillovers" from other firms' R&D (Spence, 1984). Firms with large patent portfolios can bargain for access to other firms' technology—as IBM claims in the quote at the beginning of this paper. This technology may then be used in lieu of R&D spending. A drop in the price of patenting permits large portfolio holders to increase their spillover access, effectively substituting other firms' R&D for their own. On the other hand, holders of small portfolios may choose to obtain more patents (defensively), but nevertheless see a decline in net appropriability (Bessen, 2002). With lower potential profits, they will also

[3] In dynamic models of innovation this intuition does not necessarily hold. The reason is that it is possible that grant more patents can actually dissipate more rents than they create. See Bessen and Maskin 2001, Hunt 2001, or O'Donoghue 1998.

invest less in R&D. Thus strategic patent portfolio behavior may cause firms to substitute software patents for R&D.

Using the software share of a firm's patents as a proxy for the cost of patenting, we find that, in fact, software patents do substitute for R&D. Those firms that increased the share of software patents in their patent portfolios tended to reduce their R&D spending relative to sales in the 1990's. The magnitude of the substitution effect is significant: at sample means, continuing firms would have spent 10% more on R&D if patenting standards had not been changed for software. It is possible that R&D by new firms might offset this decline, but our evidence suggests this is unlikely.[4]

Our results suggest that an economically important reason why firms acquired software patents in the 1990's was related to patent thicket strategies. In effect, the change in patent standards made it cheaper to acquire a patent thicket. This explanation is consistent with the fact that large firms, and manufacturing firms, acquired many more software patents than were acquired by software publishers or other small firms and independent inventors typically associated with software innovation. Regression analysis suggests that most of these patents were acquired without a corresponding increase in expected profits, that is, they were acquired to improve appropriability of existing technology rather than to protect new innovations. And firms adjusted their R&D spending consistent with a model of strategic patenting.

The first section presents a model of appropriability. The second section describes our data and variables. The third section presents our empirical results and the fourth section concludes.

[4] A Heckman regression indicates that the exclusion of new public entrants did not skew our results, so it appears they too would have followed a similar pattern. Moreover, if strategic patent portfolio behavior caused these changes, then cheaper patents would reduce entrants' R&D incentives. Finally, new firms simply do not conduct a large portion of industry R&D.

I. A Model of Appropriability

A. Background

The rise in the number of software patents should be viewed as a multi-factored phenomenon. The first and most basic factor is the reversal of the subject matter exception that precluded patents on software, as described in the introduction.

A second possible factor is the greater use of software in the design of new inventions. Software has grown rapidly in economic importance. The BEA estimates that the share of real (chain weighted) GDP accounted for by final sales of software tripled (from 0.6% to 2.1%) between 1987 and 1998 (Parker and Grimm 2000). Much of this growth may have consisted of new products using software.[5] To the extent that these new products involved new *inventions*, they may have generated new patents. In other words, software may have been especially productive in the design of new products. Kortum and Lerner (1999) examine the rise of patenting generally and attribute it primarily to an increase in the productivity of firms' R&D programs.[6] Software patenting may be an instance of this broader trend. We call this the "productivity" hypothesis.

But other factors may have contributed to the rise in software patenting as well. One hypothesis is the "bad patent" theory. Some researchers argue that the patent office was not prepared to examine applications for patents on software, and consequently many more patents were granted than should have been (Jaffe 2000, Kesan 2002, Merges 1999). They suggest that this is the result of the subject matter extension—patent examiners lack the skills for analyzing software and they are unfamiliar with prior art in the field, thus they grant patents for software inventions that are not novel.[7] This view is supported by anecdotal claims that software patents insufficiently cite non-patent prior art (Aharonian). An implication of this view is that examination quality may improve as the patent office

[5] These estimates include software developed for firms' own use and custom software in addition to packaged software products. Packaged products comprise somewhat less than a third of total software.

[6] Hall and Ziedonis (2001), however, specifically reject this argument for the semiconductor industry.

[7] For a particularly egregious example--the Compton's multimedia patent, see Yoches 1995.

gets more experience and/or the funding to hire examiners with software experience. Kesan and Merges suggest additional reforms to address this concern.

A fourth set of factors includes broader regulatory changes that affect patents in all subject matter categories. Some researchers attribute broad changes in legal standards to the creation of a unified appeals court for patents suits in 1982 (Merges, 1997 and Hunt 1999a). The court raised the evidentiary standards required to challenge patent validity and broadened the interpretation of patent scope (Rai, Merges, 1997). The court relaxed the requirement that patents be granted only for inventions that are not obvious to "practitioners skilled in the art" (Cooley 1994, Dunner et al 1995, Hunt 1999b, Lunney 2001). The court is more willing to grant preliminary injunctions to patentees during infringement suits (Cunningham 1995, Lanjouw and Lerner 2001) and to sustain large damage awards (Merges, 1997, Kortum and Lerner, 1999). And plaintiff success rates have increased substantially in patent infringement suits (Lerner, 1995).[8]

Although these changes affect all patent subject matter areas, some of these factors may have outsized effects on software patenting. For example, an invention that uses software, rather than hard-wiring the logic, may be more easily adapted to specific applications. Lower patenting standards and broadening patent scope permit patents to claim a broader set of applications. American software patents tend to have abstract claims and little in the way of detailed specifications of the claimed inventions (hardware patents tend to have much more detailed specifications). Software may be particularly useful for writing patents that claim many applications or broader applications—the cost of performing the R&D necessary to obtain a patent may be substantially less if the invention incorporates software. We call this the "cost of patenting" hypothesis.

Three of these factors affect the cost of appropriability: exclusion of software as a subject matter for patents meant that they had a very high cost of appropriability (infinitely high) that was later reduced. When the patent office grants too many bad patents for

[8] Case law for these changes includes the following. On evidentiary standards: *Medtronics Inc. v. Intermedics, Inc.* 799 F.2d 734 (1986) and *Hybritech Inc. v. Monoclonal Antibodies Inc.* 802 F.2d 1367 (1986). On nonobviousness: *Stratoflex, Inc. v. Aeroquip Corp*., 713 F.2d 1530 (1983), and *Simmons Fastener Corporation v. Illinois Tool Works*, 739 F.2d 1573 (1984). On preliminary injunctions: *Atlas Powder Co. v. Litton Systems, Inc., 773 F.2d 1230 (1985).*

software, then the average cost of obtaining a patent is lower. Lower standards and broader scope for software patents also lowers the cost of appropriability. On the other hand, changes in enforcement standards (such as interpretation of the doctrine of equivalents or the extent to which patents are invalidated for reasons of obviousness) affect the appropriability each patent achieves. If a given number of patents convey a greater degree of appropriability because of enforcement changes, then the *quality-adjusted* cost of appropriability also declines.[9]

Only the productivity hypothesis explains the rise in software patenting independently of a drop in the cost of appropriability. Thus the cost of appropriability may help us distinguish this hypothesis from the others. Moreover, as we develop below, the specific pattern of changes in the relative cost of appropriability for software patents provides information helpful to distinguish between the relative roles of patent subject matter extension, bad patents and general cost measures in the rise of software patenting. We begin by formalizing a model of the cost of appropriability.

B. Patents, R&D and Appropriability

We base our analysis of software patents on an economic model of patenting; specifically we model the tradeoff between appropriability and the cost of obtaining patent protection.

To formalize this notion, define appropriability, A, as the portion of the potential profits that a firm actually collects. Let v be actual profits. Since the potential profits are greatest when the firm has a monopoly, actual operating profits (ignoring the cost of patenting and the cost of R&D) are $v = m\,A\,Q$, where m is the monopoly markup and Q is output (taking output price as *numeraire*). Then if $A = 1$, the firm earns monopoly rents, but if $A = 0$, competition dissipates all rents and there is no other form of appropriability such as trade secrets. Let $A = A(n, N)$, where n is the size of the firm's own patent portfolio and N represents the sizes of the patent portfolios of all other firms in the

[9] We adjust the cost only for the quality of standards of enforcement. Stronger enforcement means that appropriability is greater for a given cost of obtaining a patent, so the quality-adjusted cost of appropriability is less. As in footnote 2, we mean the cost is "quality-adjusted" in the sense that it corresponds to a given level of appropriability.

industry. To keep notation simple, we consider only a single other firm, so *N* is a scalar rather than a vector.

It is helpful to consider a simple example of this appropriability function. Suppose a firm has multiple inventions and it can protect each of them as a trade secret (including lead time and learning advantages) or with a patent. Suppose a patent achieves a higher fixed level of appropriability than trade secrecy, but a patent costs more to obtain. The firm's average level of appropriability will increase with the number of inventions the firm chooses to patent. That is, $\frac{\partial A}{\partial n} > 0$. Also, the firm will choose to obtain patents for those inventions that are sufficiently valuable so that the additional appropriability of a patent offsets the additional cost. So the firm will patent the most valuable inventions first. Consequently, $\frac{\partial^2 A}{\partial n^2} < 0$. Now, if firms pursue patent portfolio strategies then it is also possible that $\frac{\partial A}{\partial N} \neq 0$. Specifically, with patent portfolio strategies, a firm's appropriability decreases with the size of other firms' portfolios (Bessen, 2002). The concavity of the appropriability function with regard to *n* also applies in the case of patent portfolio strategies, so we assume this to be a general property.

We assume that the production function has constant returns to scale in conventional input factors, and diminishing returns in the firm's knowledge stock, *K*. Factoring out the conventional input factors, we may write

$Q = Q(K),\ \frac{\partial Q}{\partial K} > 0,\ \frac{\partial^2 Q}{\partial K^2} < 0$. The firm's knowledge stock consists of the depreciated stock of the firm's own R&D, *r*, plus spillovers from the inventive activity of other firms. Assuming a static equilibrium for simplicity, stocks will be proportional to flows, so, ignoring a possible constant, $K = r + x$, where *x* represents spillover knowledge.[10] We

[10] This specification makes spillovers and own-R&D substitutes. Cohen and Levinthal (1989) propose that some R&D may complement spillovers. However, our purpose here is to explore a possible substitution effect (see below), so we use this specification.

assume, $x = x(N)$ and $\frac{\partial x}{\partial N} \leq 0$, that is, patents (weakly) reduce spillovers.

Let *c* be the quality-adjusted cost of obtaining a patent. Total profits are then

(1) $$p(n, N, r) = m \cdot A(n, N) \cdot Q\big(r + x(N)\big) - c \cdot n - r.$$

This equation captures two basic tradeoffs: greater appropriability versus the cost of patents and greater output (productivity) versus the cost of R&D. We assume that firms optimize with respect to *n* and *r*, taking *N* as given. For simplicity, let $Q(K) = K^g$, $0 < g < 1$. Then the first order maximizing conditions are

$$\frac{\partial p}{\partial n} = 0 = m\frac{\partial A}{\partial n}Q - c, \qquad \frac{\partial p}{\partial r} = 0 = mA\frac{\partial Q}{\partial K} - 1$$

or

(2) (a.) $$\frac{\partial \ln A}{\partial n} = \frac{c}{v}, \qquad \text{(b.)} \quad \frac{r}{Q} = mAg - \frac{x}{Q},$$

recalling that $v = mAQ$.

The first equation expresses the intuition that, given diminishing returns to appropriability with *n*, an increase in the cost of patenting raises the marginal appropriability, reducing the firm's optimal patent portfolio size, all else equal. The second equation expresses the intuition that, all else equal, R&D intensity will increase with appropriability, but decrease with spillover intensity.[11] Given an equilibrium in the industry, conditions (2) applied to all firms in the industry yield optimal values $\hat{n}$, $\hat{r}$, and $\hat{N}$.

The total interaction between R&D intensity and patent cost varies depending on the nature of the industry equilibrium, specifically whether firms pursue patent portfolio strategies. This can be seen by taking the total derivative of (2b) subject to (2a) and the industry equilibrium condition,

[11] As noted above, spillovers might increase own-firm R&D, so the sign of this term is not really determined theoretically. Our regression strategy, in fact, allows this term to have either sign.

$$(3) \qquad \frac{d\,\frac{\hat{f}}{Q}}{d\,c} = mg\,\frac{d\,\hat{A}}{d\,c} - \frac{d\,\frac{\hat{x}}{Q}}{d\,c}.$$

The sign of this derivative determines whether patent cost acts to increase or decrease R&D intensity. It is often assumed that cheaper patents will increase R&D incentives and therefore increase R&D intensity. In this case, the derivative would be negative and patents would complement R&D. However, if firms pursue patent portfolio strategies this might not be the case.

First, consider the case with no strategic portfolio behavior. In this case, a higher cost of patents will reduce R&D intensity. Without strategic portfolio effects, $\frac{\partial A}{\partial N} = 0$. Then it is easy to show that the first term in (3) is negative. An increase in patent cost induces firms to obtain fewer patents, hence they have lower appropriability. Also, with fewer patents, one would expect spillovers to be greater.[12] Then the net sign of the second term in (3) will also be negative. Thus (3) tells us that when firms do not use patent portfolios strategically, $\frac{d\,\frac{\hat{f}}{Q}}{d\,c} < 0$. In other words, R&D and patents are *complements* in this case; a higher patent cost reduces R&D intensity.

On the other hand, if firms pursue portfolio strategies, then patents may *substitute* for R&D. This occurs through both terms in (3). First, when firms' portfolio sizes are asymmetric and they pursue strategic portfolio behavior, $\frac{\partial A}{\partial N} < 0$. As shown in Bessen (2002), a firm's appropriability declines with the size of its competitors' patent portfolios—competitors are able to gain greater bargaining leverage with larger portfolios. Then, under general conditions, it can be shown that a decrease in patent cost actually *decreases* appropriability for small portfolio holders. This occurs despite an increase in portfolio size for all firms, i.e., small firms acquire "defensive" patent portfolios, but face

[12] This might not be the case if patents disclosed much valuable technical information. Survey evidence and anecdotal evidence suggest that although some useful information is revealed in patent disclosures (mostly information about competitors' technical direction), firms gain relatively little information that is useful for innovation from them, except, perhaps, in Japan (Cohen et al, 2002, Macdonald, 1998, Oppenheim, 1998, Tang et al, 2001).

reduced appropriability nevertheless. In this case, $\frac{d A}{d c} > 0$, for small portfolio holders.

Second, under general conditions, cheaper patents mean that more firms will engage in strategic portfolio bargaining and cross-licensing (Bessen, 2002). But cross-licenses give firms access to other firms' technology, in other words, there may be greater spillovers with cheaper patents.[13] In effect, cross-licensing permits firms to literally substitute other firms' R&D for their own as in the IBM quote above. This means that the net sign of the second term in (3) will also be positive a hence, $\frac{d \frac{\hat{r}}{Q}}{d c} > 0$. Then patents and R&D are substitutes. Section III explores whether R&D intensity increases or decreases with patent cost.

C. The Cost of Appropriability and Software Patents

The role of software patents in the average cost of appropriability can be modeled as follows. Designate the quality-adjusted costs of obtaining patents as c_s and c_o for software patents and other patents, respectively. Let *s* be the software share of patents. Then

$$(4) \qquad c = (1-s)c_o + s c_s = c_o(1-s u), \qquad u \equiv \frac{c_o - c_s}{c_o}$$

where *u* is the relative cost advantage of software patents (positive or negative).

This equation relates the cost of patenting to the software share of patents. Since the cost of patenting is unobserved, our estimation strategy is to use *s* as a proxy. Consider how this may be done in equation (2a). Give *A* a functional form such that $\frac{\partial \ln A}{\partial n} \propto \hat{n}^{-1/b}$, $0 < b < 1$.[14] Then (2a) can be written

[13] Cross-licensing agreements typically do not provide for transfer of technical know-how. Usually they only provide use of the other firm's patents. However, even this may reduce the amount of R&D devoted to "inventing around" those patents, so cross-licensing may substitute for own firm R&D.

[14] Formally, ln *A* is an instance of a Pareto cumulative distribution function, $\ln A = 1 - n^{-(1-b)/b}$. This specification meets the concavity requirement and it provides a skewed distribution of the marginal appropriability of patents, corresponding to a skewed distribution of patent values (Scherer and Harhoff 2000)

$$
\begin{aligned}
\ln \hat{n}_{it} &= k + b \cdot \ln \hat{v}_{it} - b \cdot \ln c_{it} \\
&\approx m_t + b \cdot \ln \hat{v}_{it} + b \cdot u_t \cdot s_{it} \\
&= m_t + b \cdot \ln \hat{v}_{it} + \sum_j d_t \cdot I(j = t) \cdot s_{it}
\end{aligned} \tag{5}
$$

where $I(j=t)$ is an indicator function (1 if $j=t$, otherwise 0), v is profits, i designates firm and t designates year. This specification assumes that firms face similar costs of patenting for each type of patent (within an industry at any time) but that firms are heterogeneous in the degree to which they can use software in their products. The approximation is valid as long as the software share of patent costs is not too large.

Equation (5) is similar in form to "propensity to patent" regressions estimated in the patent literature (Scherer 1965, Bound et al, 1984, Pakes and Griliches, 1984, Griliches, Hall and Hausman, 1986, Hall and Ziedonis, 2001). Those regressions place the log of R&D or log sales on the right in place of log profits. In theory, R&D generate potential profits and potential profits are protected with patents. Here we focus on just the latter part of this relationship, the link between profits and patents, although we, too, include log R&D in some of our regressions. In any case, our main interest is in the s interaction terms that correspond to the constant in the patent propensity regressions. In Section III we estimate this equation, although with some different econometric considerations than this literature.

II. Description of Data

A. Definition of Software Patent

Software is always used as part of a system with hardware. However, patent law has changed the extent to which software innovations could be patented and the degree to which those innovations had to be related to specific hardware systems. In the 70's, software could be included in the embodiment of an invention, but the hardware configuration had to be novel. Through a series of court and administrative decisions during the 80's and 90's, innovations in the software components could themselves get patent protection if the software controlled a physical process or even if it merely produced meaningful data.

Some observers have sought to distinguish "pure" software patents from those patents that simply included software as part of a hardware-software system (Aharonian, Allison and Lemley, 2000). These former patents presumably are ones where the novelty lies in the software component (entirely or partially) and/or the hardware involved is a general purpose technology (PC, network, etc.). Some of these distinctions are necessarily arbitrary, however, because this determination depends substantially on the manner in which the patent is drafted and may not reflect a real difference in the scope or enforceability of the patent.

The approach here is simpler. We aim to include all patents for inventions that use software. Since we are concerned with multiple regulatory changes that may have affected software patenting, we need not restrict ourselves to those patents that would have been excluded under old subject matter rules. The changes in the cost of appropriability affect all software/hardware systems, although this cost reduction may have been greater for those systems that use general purpose hardware.

B. Data Sources

A list of utility patents (excluding re-issues) was obtained from the database of the U.S. Patent and Trademark Office meeting the following conditions:

1. Uses the word "software" in the specification OR
2. Uses the words "computer" AND "program" in the specification AND
3. Does NOT use the words "semiconductor," "chip," "circuit," "circuitry" or "bus" in the title (these patents more often execute software than use it).

This procedure generated a list of 134,690 software patents granted between 1976 and 1999. This list was then merged with the NBER Patent Citations Data File (Hall, Jaffe, Trajtenberg, 2001) to obtain characteristics of each patent.

In addition, patent assignees in this dataset were matched with firm data from Compustat for a limited number of U.S. firms. This allowed annual patent count data for 1,646 individual firms to be combined with detailed financial information from 1981-99. The match used U.S. firms in the NBER match file; in addition, the largest 25 public firms in the software publishing industry (SIC 7372) were specifically included. (These firms

obtain few software patents, so only one was included in the NBER match file.) These matched firms accounted for 42% of all US software patents and 38% of all US patents issued during 1981-99.

III. Empirical Analysis

A. Numbers of Software Patents

Table 1 reports the number of software patents and other patents granted per year. As can be seen, the numbers have grown dramatically in absolute terms and also relative to other patents. Today over 15% of all patents granted are software patents. The growth in software patents accounts for over 25% of the total growth in the number of patents between 1976 and 2001.

Table 1 also shows estimates of the number of "true" software patents published by Gregory Aharonian (PATNEWS).[15] The overall trends are quite similar and the numbers in recent years are also quite close. Clearly, our definition of software patents is more inclusive, especially during the early years.

Allison and Tiller (forthcoming) performed a more careful evaluation of "pure" software patents, examining 1,000 randomly selected patents from 1996-8 in detail. They found that 9.2% of these patents were totally embodied in software.[16] For our sample, we classified 12.2% of all patents as software patents for these years. Hence, it is fair to say that our measure largely comprises "pure" software patents in any case.

B. Characteristics of Software Patents

Table 2 shows characteristics of software patents compared to other patents using data from the NBER patent database. Software patents are more likely to be owned by firms than by individuals or government. They are also more likely to be owned by U.S.

---

[15] Aharonian used the "I know one when I see one" criterion (private communication).

[16] An initial analysis identified only 76 of the patents as software patents (Allison and Lemley, 2000). However, subsequent analysis based on better knowledge of the drafting of software claims increased the number of software patents in the sample to 92 (private communication from John Allison).

assignees and to have U.S. inventors.[17] They tend to receive a larger number of subsequent citations, but in other aspects they are similar to other patents.[18]

One might want to know the extent to which some of these differences are explained by *who* gets the software patents. The third column of Table 2 shows means for non-software patents weighted by the total number of software patents each assignee received.[19] These means are thus representative of the heavy software patenters. As can be seen, much (but not all) of the difference between software patents and other patents can be explained by the patenting behavior of the firms who obtain the most software patents.

To obtain more information about the firms who obtain software patents, Table 3 shows means of firm characteristics weighted by the number of patents (software and total) the firm receives. Relative to other patents, software patents tend to be obtained by firm with larger market value, sales and R&D budgets and are less likely to be obtained by newly public firms. Allison and Lemley (2000) also find that software patents are more likely to be obtained by larger entities as classified by the patent office.

Table 4 shows the industries of the firms obtaining software patents in the sample matched to Compustat. Most of the software patents are obtained by manufacturing firms. Software publishers (SIC 7372) acquire only 6% of the patents in this sample and other software service firms excluding IBM account for 2%.[20] These numbers are highlighted by the last column showing differences in patent propensity. Software publishing firms get only a third of the number of patents per dollar of R&D that other firms obtain. This corresponds to the views expressed by software publishing executives that software patents are of little value to them (USPTO, 1994).

---

[17] Allison and Lemley (2000) find that their sample of software patents has about the average portion of U.S. inventors, although they use a somewhat different method to classify inventors' national origins.

[18] See Hall, Jaffe, and Trajtenberg (2001) for a description of the citation data.

[19] This is an appropriate weighting under a null hypothesis that the process that generates patents in general is also the process that generates software patents. Of course that hypothesis may be incorrect, and the significant differences we see in the table seems to suggest that is so.

[20] The data set oversamples SIC 7372 somewhat. If each industry in the sample is weighted to match the employment for that industry in the 1997 economic census, then SIC 7372 would account for 5% of software patents. IBM itself accounts for 20% of the software patents in our sample. IBM is consistently the largest software patenter and we break it out separately because it is not representative of the software services industry overall.

Overall, software patents are more likely to be obtained by larger firms, established firms, U.S. firms, and firms in manufacturing (and IBM); they are less likely to be obtained by individuals, small firms, newly public firms, foreign firms and software publishers. They are also more likely to be subsequently cited in other patents.

This pattern conflicts with the simple view that firms are equally likely to obtain software patents to protect individual software inventions. BEA analysis of software investment (Parker and Grimm, 2000) implies that about 30% of software is produced as packaged software, the primary product of firms in SIC 7372. Yet the software publishing industry acquires a much smaller portion of software patents. Anecdotal evidence suggests that, compared to other technologies, relatively more software innovations are developed by individuals and small firms. Yet large firms acquire proportionately more software patents. In fact, the firms that acquire the largest share of software patents appear to be just those firms that pursue patent thicket strategies: large firms in electronics, computers, instruments, transportation and other manufacturing industries and IBM. As we argue below, this is no coincidence, but, in fact, indicates that software patents (because they are a cheap form of appropriability) are particularly useful for firms building strategic patent portfolios.

C. Exploring the Rise in Software Patenting

The increase in the rate of software patenting is part of a broader trend as seen in Table 1, although the number of software patents has increased an order of magnitude more rapidly than other patents. As discussed above, concomitant regulatory changes did not necessarily cause this increase. It is possible that firms acquired more software patents because they found it more profitable to use software in developing their products (the "productivity" hypothesis). They may have acquired more software patents simply because they used more software in product design. This section explores how much of the rise in software patenting can be attributed to a lower quality-adjusted cost of software patents or whether, instead, software patents have been acquired because technological opportunities for using software have become more profitable.

Each factor is captured by a different term in equation (5), which we estimate in this section. This equation relates the log of the target number of patents to: a.) a measure of target profits, and b.) the software share of patents, *s*, interacted with time dummies. If firms found it more profitable to use software in products, then greater use of software patents would be captured in the profits term. If, on the other hand, firms could obtain software patents at a lower cost for a given year, then this would be captured in the *s* term for that year.

The coefficients of the *s* terms, d_t, provide information about the relative cost of obtaining software patents compared to other patents:

If $d_t < 0$, then software patents are more costly than other patents and firms will only obtain software patents if they need to protect software that is used for technological reasons.

If $0 < d_t$, then there is a cost advantage to using software patents. For a given level of profits, firms may obtain software patents because appropriability costs less.

To implement equation (5), we assume that the target number of patents can be approximated by the actual number of patents granted, *n*, plus a stochastic error term, $\ln \hat{n}_{it} = \ln n_{it} + e_{it}$. We use patents granted as opposed to patent applications under the assumption that corporate intellectual property departments anticipate appropriability needs and control the patenting process to meet those needs. Any errors in this process give rise to the stochastic disturbance term. Since neither ln *n* nor *s* are meaningful when *n* = 0, these observations are excluded. Below we explore the effect of this sample selection (and the selection imposed by the matching of firms to patents) on parameter estimates. Also, this approach does not require a Poisson estimation, although sampling variance does introduce heteroscedasticity.

The variable *s* is measured as the simple share of software patents in total patents granted. It is possible that the error term is correlated (negatively) with the current year measure of *s*. For this reason, we use the lagged value, $s_{i,t-1}$. We interact *s* with dummies for four periods (1981-85, 1986-90, 1991-95, 1996-99) to yield coefficients, d_t, for each of these periods. Note that this measure of *s* will have a large variance for small values of

n—the observed value will have a sampling variance that is proportional to 1/*n*. This measurement error will tend to bias the estimates of d_t toward zero.

The exposition of (5) assumes a static equilibrium for simplicity and it uses a current measure of profits accordingly. In a dynamic setting, firms will adjust their target patent portfolio size based on their expected stream of future profits. To capture this variable, we use two measures: the market value of the firm (long term debt plus the carrying value of preferred stock plus the end-of-year value of common stock) and the mean cashflow for the following four years (measuring cashflow as operating income before depreciation plus R&D spending, if not missing). Under rational expectations, both of these measures may differ from expected profits because of measurement error—"animal spirits" for market value or forecasting error for cashflow. To correct for this possibility, we perform an instrumental variable estimation below. Also, it is possible that these profit measures (and instruments based on current year variables) might be positively correlated with the error term—that is, firms will adjust profit expectations upwards upon receiving "windfall" patents. To prevent endogeneity problems, we use lagged instruments for $E[v]$.[21]

Other factors may also affect the cost of patenting and the calculation of optimal patent portfolio size. Time dummies and firm fixed effects are also included in the regressions to accounts for general trends in patenting and unobserved firm characteristics. Patent acquisition costs might also vary according to firm size—larger firms may realize economies of scale in the patenting process, e.g., with an in-house legal department. Since *v* is already included in the regression, positive economies of scale will tend to bias estimates of β upwards. Other possible control variables are included below.

Taking account into these considerations, the actual equation we estimate is then

[21] In the regressions shown below, we instrument with log sales and log employment lagged one year. We also performed these regressions with longer lags. The results were similar (although with larger errors). Moreover, the estimates of β increased with longer lags, suggesting that endogeneity was not a major issue.

$$(6) \qquad \ln n_{it} = a_i + m_t + b \cdot \ln v_{it} + \sum_j d_T \cdot I(j = T) \cdot s_{i,t-1} + e_{it}$$

where *I* is an indicator function as above and *T* represents four different groups of years (1981-85, 1986-90, 1991-95, 1996-99).

The regressions are shown in Table 5. The first column uses forward cashflow to proxy for expected profits (with a truncated sample period). The estimates of δ show a distinct upward trend, beginning strongly and significantly negative and becoming strongly and significantly positive during the early 90's. This implies that software patents were at a substantial cost disadvantage during the early 80's, but they had a large cost advantage during the 90's.

The second column adds additional controls. We add the log of lagged R&D to proxy changes that might not be captured in the profit measure. Firms with capital intensive technologies may more easily acquire patents, so we add a measure of capital intensity. Also, differences in the global dispersion of R&D and profits may influence patenting, so we include the portion of inventors from the U.S. Also, firms may choose to substitute patent quality for quantity; we include subsequent citations as a measure of quality. And firms who cite their own patents heavily may be engaging in strategic "fence building" behavior, getting more patents to block competitors from using related technologies. These additional variables do appear to have some effect, but the overall picture remains the same although the estimate of δ is slightly less for the early 90's..

The third and fourth columns repeat these estimations using market value as the proxy for profits. This permits estimates for the late 90's. The estimates of δ are similar, although slightly more negative and with a slightly poorer fit. However, the differences between the coefficients for the early 90's and late 80's are close. Also, the estimates for the late 90's suggest that the cost advantage of software patents continued to increase.

The fifth column shows the coefficients from a two stage "Heckit" estimation to explore the effects of sample selection. First a probit is performed where the dependent variable is one if the observation is included in the least squares regression and zero otherwise. An observation will be included if the firm is matched to the patent file and has at least one patent for the observation year. The total sample includes all Compustat

observations for U.S. firms with non-missing data. The probit regressors are log sales, log employment, a flag if firm went public during last 5 years, and year dummies. A likelihood ratio test weakly rejects the null hypothesis that the disturbances of the two equation are uncorrelated (they are negatively correlated at the 5% level) and the reported coefficients are adjusted for this correlation. This procedure corrects for Compustat firms that are not included in the regression sample, but it does not address the selection of firms for the Compustat sample, i.e., publicly owned firms. The resulting coefficients are similar.

To interpret the shift in coefficients, consider what would happen if a firm were to increase the number of software patents it obtains from 0 to n_s, with no change to its profits. Then its optimal number of patents would also increase. It is easy to show that $\Delta\hat{n}_{it} \approx d_t \cdot n_s$ as long as *s* is not large. This means that if we observe a firm with n_s software patents at time *t*, we can say that d_t percent of these software patents increased the firm's patent portfolio without any increase in profits (assuming d_t is positive). In other words, d_t percent of these software patents can be attributed solely to the cost advantage of software patents.[22] So 31%-42% of the software patents obtained during the early 90's can be attributed to the cost advantages of software patents as can 56%-60% of the software patents acquired during the late 90's.

To evaluate the change in software patents over time, the difference in $d_t \cdot n_s$ between two periods reflects the combined effect of changes in d_t and changes in *s*. Using the aggregate counts in Table 1 averaged over 5 year periods, the combined changes account for an increase of 2,400 - 3,100 patents/year (using Columns 1 and 2) between the late 80's and early 90's; actual software patents/year grew by about 3,200 during this period. Changes in the cost of appropriability of software patents account for an increase of 10,800 - 11,500 patents/year between the late 80's and late 90's (using Columns 3 and 4); actual software patents/year grew by about 14,100. In other words,

[22] If d_t were negative, then $d_t \cdot n_s$ represents the number of patents the firm would choose *not* to obtain because of the cost disadvantage of software patents. These patents are presumably obtained in spite of the cost disadvantage for reasons of technological opportunity.

changes in the relative cost of appropriability account for about 80% of the increase in software patenting during the 90's.

Software patents have become cheap. Considering that the estimates of δ are likely attenuated because of sampling variance, our results indicate that the productivity hypothesis has little explanatory power, most likely accounting for less than 20% of the increase in software patents.

Moreover, the extension of subject matter to software can only explain part of the observed trend. If this were the only factor involved, then one would expect that software patents would cost about the same as other patents, once the exclusion was removed. This would imply δ should be near zero for the 90's. However, the actual estimates are significantly positive. This implies that software patents have a substantial cost advantage, either because many bad software patents are granted and/or software patents are particularly useful for taking advantage of lower patenting standards.

Finally, the magnitude of the shift in δ suggests that bad patents alone cannot explain the shift unless one is willing to posit that *most* software patents are invalid (and that this is not true for other patents). If, say, only 10-20% of software patents were granted improperly, this could not explain the large estimated shift in the cost of appropriability that accounts for 80% of the increase in software patents. Moreover, it is not clear that the examination process is all that different for software patents. On average, software patents actually spend a longer time in prosecution than other patents, and, despite anecdotal comments (Aharonian), software patents actually cite *more* non-patent prior art than other patents (Allison and Lemley, 2000). Of course, perhaps the "right" prior art is not cited, nevertheless, the gross statistics suggest that the quality of software patent examination is roughly similar to the quality of examination for other patents. And now that 150,000 software patents have been granted, it seems difficult to argue that patent examiners are any less familiar with prior art in this area than in any other. Yet our results indicate a substantially lower cost of appropriability for software patents.[23]

[23] The issue we address is the *relative* quality of patent examination. The quality of *all* patent examination may be poor, in which case, some of the reforms suggested may be appropriate (Kesan, 2002, Merges, 1999).

Thus we conclude that the fall in the cost of appropriability of software patents was largely caused by more basic factors than examination quality, although examination quality likely contributed. Perhaps software patents are particularly well-suited to take advantage of lower patenting standards and broader patent scope because of the abstract nature of their claims. Indeed, Allison and Lemley (2000) find that software patents contain more claims and more independent claims than other patents on average. Surveying 14 technology areas, software patents had more claims per patent than any other technology except acoustical patents. This suggests that general regulatory changes *combined* with the extension of patents to software inventions allowed firms to inexpensively build patent portfolios. And this drove the acquisition of many software patents.

D. Software Patents and R&D

These results imply that during the 1990's, Δs should be negatively correlated with Δc —those firms that increased their software share of patents should have decreased their average cost of patenting relative to other firms. Accepting this interpretation, Δs can be used to explore whether software patents tended to complement R&D or substitute for it. Following the analysis in Section II, if software patents complement R&D, then R&D intensity (R&D to sales ratio) should increase with Δs; if software patents substitute for R&D, the relationship should be negative.

We do find a significant negative association below. However, this does not necessarily mean that software patents substitute for R&D. For instance, it is possible that greater use of software decreased the cost of performing R&D and R&D demand was sufficiently inelastic that the R&D share of output decreased. We first perform the estimation and then explore alternative explanations below.

Table 6 shows regressions for the 1990's using five year differences. The first column shows a simple regression with year and industry dummies. All the regressions are weighted to correct for sampling variance in Δs.[24] The coefficient of Δs is significantly

[24] The weight used is $1/(1/n_t + 1/n_{t-5})$ since the sampling variance for *s* is proportional to *n*.

negative, suggesting a substitution effect. It is possible that some third factor might be correlated with both the change in R&D intensity and the change in software share (negatively), producing a spurious correlation in this regression. The second column adds variables to control for possible size effects, effects related to new firms, and risk (the year-to-year standard deviation of the stock price). These additional variables do not significantly alter the coefficient of Δs.

The third column explores how this substitution effect differs between firms with large number of patents and other firms. As discussed above, the mechanism of substitution may be somewhat different for large portfolio holders. This appears to be the case; the coefficient is significantly larger for Δs interacted with an indicator of more than 100 patents per year. However, the coefficient for the remaining firms is still significantly negative.

The fourth column repeats the basic regression with firm fixed effects to capture firm specific factors that may affect the growth of R&D intensity. The coefficient of Δs is still negative and significant, although slightly smaller in magnitude. The fifth column explores whether the substitution effect changed during the 90's by interacting Δs with time indicator dummies. The effect appears substantially stronger during the later 90's and the coefficient for the early 90's is not statistically significant. This suggests that the cost advantage was greater during the later 90's (there is evidence of this in Table 5) and/or the substitution effect lags the cost changes. This trend also holds if we account for large patenters separately (regression not shown). When we interacted Δs with both time period (early and late 90's) and the firm's patenting level (more or less than 100 patents/year), both groups of firms had statistically significant coefficients during the late 90's but not during the early 90's. Moreover, the difference between the groups was not as great during the late 90's as indicated in column three.

One possible estimation problem may arise because we measure r/Q as the ratio of R&D to sales. However, the sales deflator does not accurately correct for changes in the markup that should accompany greater appropriability, so there may be a slight negative bias in the coefficient of Δs. Thumbnail calculations suggest that this bias should not be large. Nevertheless, in the sixth column we use a dependent variable that should

not be affected by changes in the markup, namely, the change in R&D per employee. Here the coefficient is again negative and significant.

Two other regressions are not shown. We also performed a Heckman two stage regression similar to the one in the previous section. Here, however, the disturbance of the probit equation was not significantly correlated with the least squares disturbance. And we also performed a three stage least squares regression for a system of equations including two appropriability equations as in the previous section (one for the current year and one for the lagged year). These results were also similar to the regressions shown.

Finally, Table 7 shows the first regression repeated for five industry groups. The coefficients are all negative and significant except for business services (SIC 73), which includes software services, and machinery (SIC 35), which includes computer equipment. The substitution effect is strongest in electronics (SIC 36). This is consistent with results found in Hunt (1996).

Thus there does appear to be a robust negative association between the growth in software patent share and the growth in R&D intensity. Moreover, this effect appears stronger among large portfolio holders and among firms *outside* computer and software-related industries.

One possible explanation for this association is based on the productivity of software in R&D. If greater use of software reduced the cost of performing R&D and if the demand for R&D were sufficiently inelastic, then greater use of software would be associated with lower R&D intensity. For two reasons, however, this effect is unlikely to explain the relatively large effect measured in Table 6. First, the results of the previous section suggest that most of the software patenting in the 90's was not driven by the productivity of software. This makes it seem unlikely that the growth of software patents proxy for changes that have a large effect on the cost of performing R&D. Second, evidence from studies of the effect of the R&D tax credit on R&D spending suggests that demand for R&D is elastic. Berger (1993) found that reductions in the tax price of R&D *increased* R&D intensity. More generally, Hall and Reenen (1999) survey a number of recent empirical studies examining the response of R&D spending to changes in the tax treatment of R&D. They find a tax price elasticity approximately equal to one in absolute

magnitude, and possibly larger. Assuming that R&D demand responds to changes in the cost of performing R&D the same way it responds to the tax cost of R&D, this elasticity implies that decreases in the cost of R&D should not decrease R&D intensity and may well increase it somewhat.[25] Hence any effect of software on R&D productivity is an unlikely explanation for the decrease in R&D intensity associated with increased use of software patents.

Another possible explanation for a link between software patents and R&D intensity involves accounting changes. Beginning in 1985, FASB required firms to capitalize software development expenses (but not research or maintenance expense, which usually account for most software cost). Reported software R&D includes directly expensed items plus the amortization of capitalized software. Typically, software is amortized over 30 months. This means that a rapid increase in software development might take two years to fully appear in R&D measures. However, this is unlikely to explain our results for two reasons. First, the change in accounting practice occurred during the late 80's and early 90's, yet our substitution effect is strongest in the late 90's. Second, while this might weaken a positive association between changes in software patents and R&D intensity, it is unlikely to explain a *negative* relationship. Moreover, the amortization lag is still shorter than the average lag between application and grant for software patents (3.15 years according to Allison and Lemley 2000), so the effect of amortization is likely to be quite weak over five year differences.

Thus, finding no suitable alternative explanation, we conclude that software patents have substituted for R&D in the 1990s, an outcome that is consistent with a model where firms pursue patent thicket strategies.

The magnitude of this substitution effect is substantial. Assigning Δs a value of 10% over the 90's (see Table 1), R&D intensity would have been about 0.35% higher had the quality-adjusted cost of software patents remained the same over 90's. The mean R&D intensity of the regression sample was 3.6% in 1990, suggesting that relative R&D

[25] Assuming a cost function with constant returns to scale, changes in the price of R&D will have no effect on the R&D share of output if the absolute magnitude of the price elasticity of demand for R&D is 1-*S*, where *S* is the R&D share of output. If demand is more elastic than this, the R&D share will increase with a price decrease.

spending would have been about 10% higher among continuing firms by the end of the decade if patenting regulation had not changed for software. It is possible that the net effect of entering and exiting firms may have offset this reduction in R&D. However, entering firms do not perform a large share of R&D. Furthermore, we interpret our results as evidence of strategic portfolio behavior; such strategic behavior would have reduced R&D incentives for entrants.

IV. Conclusion

We conclude that the rise in software patenting has been driven largely by patent portfolio strategies. Lower standards for obtaining patents (plus tougher enforcement of those patents) substantially reduced the cost of patenting inventions that use software. Cheap patents enabled firms to accumulate large portfolios of patents at low cost.

This explanation is consistent with the fact that larger U.S. firms in manufacturing industries (plus IBM) have acquired a disproportionate share of software patents. Many of these industries, such as semiconductors and computers, have well-known patterns of strategic behavior using patent thickets (Hall and Ziedonis, 2001, Grindley and Teece, 1997). Large firms in these industries also have experienced in-house counsel or established relationship with law firms specializing in intellectual property.

The effect of software patents on R&D is also consistent with strategic portfolio patenting. Firms that acquired software patents performed relatively *less* R&D. Patents appear to substitute for R&D because large portfolio holders gain access to other firms' R&D and/or portfolio strategies reduce incentives to invest in R&D.

The social welfare effects of this substitution are ambiguous. In some models, it may actually be socially beneficial to reduce R&D. For example, in some patent race models, firms may duplicate each other's R&D and this duplication is socially wasteful. In such cases, if patents lead to less R&D, this decrease may reduce duplicative R&D and thus benefit society. On the other hand, society may benefit when multiple firms pursue parallel R&D paths (Bessen and Maskin, 2000).

However, policy analysts usually assume that greater R&D incentives are desirable, perhaps recognizing from Arrow (1962) that there may be a general tendency to under-invest in R&D. Indeed, advocates for broadening patent coverage for software in

Europe have argued that such changes are needed in order to increase R&D incentives (e.g., Hart et al, 2000). To the contrary, our results indicate that cheaper software patents, as implemented in the U.S., may have significantly reduced R&D investments.

References

Aharonian, G. *Patent News Service*.

Allison, John R. and Mark A. Lemley. 2000. "Who's Patenting What? An Empirical Exploration of Patent Prosecution," *Vanderbilt Law Review*, v. 58, pp. 2099-2148.

Allison, John R. and Emerson H. Tiller. Forthcoming. "An Empirical Analysis of Internet Business Method Patents."

Arrow, K. 1962. "Economic Welfare and the Allocation of Resources for Inventions," *The Rate and Direction of Inventive Activity*, Princeton: Princeton University Press.

Berger, Philip. 1993. "Explicit and Implicit Effects of the R&D Tax Credit," *Journal of Accounting Research*, v. 31, pp. 131-71.

Bessen, James. 2002. "Patent Thickets: Strategic Patenting of Complex Technologies," ROI Working Paper (December, 2002 version).

Bessen, James and Eric Maskin . 2000. "Sequential Innovation, Patents and Imitation," MIT Department Economics Working Paper #00-01.

Bound, John, Clint Cummins, Zvi Griliches, Bronwyn H. Hall, and Adam Jaffe. 1984. "Who Does R&D and Who Patents?" in Griliches, Zvi ed., *R&D Patents, and Productivity*, Chicago: University of Chicago Press for the NBER.

Cohen, Wesley M., Akira Goto, Akiya Nagata, Richard R. Nelson, John P. Walsh, (2002, forthcoming) "R&D spillovers, patents and the incentives to innovate in Japan and the United States", *Research Policy*.

Cohen, Wesley M. and Daniel Levinthal. 1989. "Innovation and Learning: The Two Faces of R&D," *Economic Journal*, v. 99, pp. 569-96.

Cohen, Wesley M., Richard R. Nelson, and John P. Walsh. 2000. "Protecting Their Intellectual Assets: Appropriability Conditions and why U.S. Manufacturing Firms Patent (or not)," National Bureau of Economic Research Working Paper 7552.

Coolley, Ronald B. 1994. "The Status of Obviousness and How to Assert It as a Defense,"*Journal of the Patent and Trademark Office Society*, v. 76, pp. 625-44.

Cunningham, M.A. 1995. "Preliminary Injunctive Relief in Patent Litigation," *IDEA*, v. 35, pp. 213-59.

Dunner, Donald R., J. Michael Jakes, and Jeffrey D. Karceski. 1995. "A Statistical Look at the Federal Circuit's Patent Decisions; 1982-1994," *Federal Circuit Bar Journal*, v. 5, pp. 151-80.

Griliches, Zvi. 1990. "Patent Statistics as Economic Indicators: A Survey," *Journal of Economic Literature*, v. 28, pp. 1661-1707.

Grindley, P. and Teece, D. 1997. "Managing intellectual capital: licensing and cross-licensing in semiconductors and electronics," *California Management Review*, v. 39, no. 2, p. 8.

Hall, Bronwyn, H. and John van Reenen. 1999. "How Effective are Fiscal Incentives for R&D? A Review of the Evidence," NBER Working Paper no. 7098.

Hall, Bronwyn H., Adam B. Jaffe, and Manuel Trajtenberg. 2001. "The NBER Patent Citations Data File: Lessons, Insights and Methodological Tools," NBER Working Paper No. 8498.

Hall, Bronwyn H. and Rosemary Ham Ziedonis. 2001. "The Patent Paradox Revisited: An Empirical Study of Patenting in the U.S. Semiconductor Industry, 1979-1995" *RAND Journal of Economics*. V. 32:1, pp. 101-128.

Hart, Robert, Peter Holmes, and John Reid. 2000. "The Economic Impact of Patentability of Computer Programs -Report to the European Commission DG Internal Market," London Intellectual Property Institute.

Hausman, J., Hall, B., and Griliches, Z. 1984. "Econometric models for count data with an application to the patents-R&D relationship," *Econometrica*, v. 52, no. 4, p. 909.

Hunt, Robert M. 1996. "The Value of R&D in the U.S. Semiconductor Industry: What Happened in the 1980s?"in Three Essays in Law and Economics, Ph.D. Dissertation, University of Pennsylvania.

Hunt, Robert M. 1999a. "Nonobviousness and the Incentive to Innovate: An Economic Analysis Of Intellectual Property Reform," Working Paper 99-3, Federal Reserve Bank of Philadelphia.

Hunt, Robert M. 1999b. "Patent Reform: A Mixed Blessing for the U.S. Economy?" Federal Reserve Bank of Philadelphia *Business Review*, (November/December), pp. 15-29.

Hunt, Robert M. 2001. "You Can Patent That? Are Patents on Computer Programs and Business Methods Good for the New Economy?" Federal Reserve Bank of Philadelphia *Business Review*, 1st Quarter, pp. 5-15.

Jaffe, Adam B. 2000. "The U.S. Patent System in Transition: Policy Innovation and the Innovation Process," *Research Policy*, v. 29, pp. 531-557.

Jaffe, Adam B. and Manuel Trajtenberg. 2002. *Patents, Citations And Innovations: A Window On The Knowledge Economy*. Cambridge, Mass.: MIT Press.

Kesan, Jay P. 2002. "Carrots and Sticks to Create a Better Patent System," *Berkeley Technology Law Journal*, v. 17, No. 2, pp. 763-797.

Kortum, Samuel and Josh Lerner. 1999. "What is Behind the Recent Surge in Patenting?" *Research Policy*, v. 28, pp.1-22.

Lanjouw, Jean and Josh Lerner. 2001. "Tilting the Table? The Use of Preliminary Injunctions" *Journal of Law and Economics, Vol.* XLIV , pp. 573- 603.

Lerner, Josh. 1995. "Patenting in the Shadow of Competitors," *Journal of Law and Economics*, v. 38, pp. 463-495.

Levin, Richard C., Alvin K. Klevorick, Richard R. Nelson, and Sidney G. Winter. 1987. "Appropriating the Returns from Industrial Research and Development," *Brooking Papers on Economic Activity*, 3, pp. 783-820.

Lunney, Glynn S. Jr. 2001. "E-Obviousness," *Michigan Telecommunications and Technology Law Review*, v. 7, pp. 363-421..

Macdonald, Stuart (1998), "What the patent system offers the small firm," Summary Report, prepared for the ESRC.

Merges, Robert P. 1997. *Patent Law and Policy: Cases and Materials*, 2nd ed., Virginia: Michie Company.

Merges, Robert P. 1999. "As Many As Six Impossible Patents Before Breakfast: Property Rights For Business Concepts And Patent System Reform," *Berkeley Technology Law Journal*, v. 14.

Nichols, Kenneth. *Inventing Software: The Rise of Computer Related Patents.* London: Quorum Books, 1998.

O'Donoghue, Ted. "A Patentability Requirement for Sequential Innovation," *RAND Journal of Economics,* Vol. 29 (1998), pp. 654-79.

Oppenheim, Charles (1998), "How SMEs use the patent literature," Summary Report for the UK Economic and Social Research Council.

Pakes, Ariel and Zvi Griliches. 1984. "Patents and R&D at the Firm Level: A First Look," *Economic Letters*, v. 5, pp. 377-81.

Parker, Robert and Bruce Grimm. 2000. “Recognition of Business and Government Expenditures for Software as Investment: Methodology and Quantitative Impacts, 1959-98,” Bureau of Economic Analysis.

Rai, Arti K. 2000. “Addressing the Patent Gold Rush: The Role of Deference to PTO Patent Denials,” *Washington University Journal of Law and Economic Policy*, v.2, p. 199.

Scherer, F. M. 1965. “Firm size, market structure, opportunity, and the output of patented inventions,” *American Economic Review*, v. 55, pp. 1097-1125.

Scherer, Frederic. M. and Dietmar Harhoff. 2000. "Technology Policy in a World of Skew Distributed Outcomes," *Research Policy*, Vol. 29, pp. 559-66.

Smith, Roger. 1990. quoted in Boyer, Chuck, “The Power of the Patent Portfolio,” *Think* (IBM), No. 5, pp. 10-11.

Spence, M. 1984. “Cost reduction, competition, and industry performance,” *Econometrica*, v. 52, no. 1, p. 101.

Tang, Puay, John Adams and Daniel Pare, (2001), “Patent protection of computer programmes” Submitted to European Commission, Directorate-General Enterprise.

USPTO 1994. “Hearings on Software Patent Protection,” United States Patent and Trademark Office, January-February 1994.

Yoches, E. Robert. “The Compton’s Reexamination – A Sign of the Times,” *The Computer Lawyer*, Vol. 12, No. 3 (March 1995), pp. 14-18.

Tables

Table 1. Number of Software Patents Granted

	Software Patents	Aharonian Estimates	Other Utility Patents	Software/ Total
1976	766	100	69,460	1.1%
1977	885	100	64,384	1.4%
1978	902	150	65,200	1.4%
1979	800	200	48,054	1.6%
1980	1,080	250	60,739	1.7%
1981	1,281	300	64,490	1.9%
1982	1,404	300	56,484	2.4%
1983	1,444	350	55,416	2.5%
1984	1,941	400	65,259	2.9%
1985	2,460	500	69,201	3.4%
1986	2,666	600	68,194	3.8%
1987	3,549	800	79,403	4.3%
1988	3,507	800	74,417	4.5%
1989	5,002	1,600	90,535	5.2%
1990	4,738	1,300	85,626	5.2%
1991	5,401	1,500	91,112	5.6%
1992	5,938	1,624	91,506	6.1%
1993	6,902	2,400	91,440	7.0%
1994	8,183	4,569	93,493	8.0%
1995	9,186	6,142	92,233	9.1%
1996	11,664	9,000	97,981	10.6%
1997	12,810	13,000	99,173	11.4%
1998	20,411	17,500	127,108	13.8%
1999	21,770	21,000	131,716	14.2%
2000	23,141	--	134,454	14.7%
2001	25,973	--	140,185	15.6%

Note: Excludes re-issues.

Table 2. Characteristics of Software Patents (1990-95)

	Software Patents	Other Patents	Other Patents, weighted mean
Assignee type			
Non-gov't. org. (firm)	88%	80%	
Individual/ unassigned	11%	18%	
Government	2%	2%	
U.S. assignee (if assigned)	70%	51%	
U.S. inventor	70%	53%	66%
Mean citations received	9.7	4.6	6.1
Percent of self-citations	12%	13%	13%
Percent of patents owned by top 5% of assignees	63%	64%	

Note: Total patents: 40,348 software, 545,410 other. Self-citations is average of upper and lower bounds (see Hall, Jaffe, and Trajtenberg, 2001). Differences between the means in the first two columns are all significant at the 1% level. Third column weights means by the number of total number of software patents the assignee received.

Table 3. Firm Characteristics by Patent Type (1990-95)

Weight	Software	Total Patents
Ln(firm market value) (million $96)	9.36	9.12
Ln(firm sales) (million $96)	9.38	9.03
Ln(R&D) (million $96)	6.58	5.96
Newly public firm	1.5%	2.1%

Note: Table shows firm means weighted by patent numbers for firm for each year 1990-95 from the sample matched to Compustat. Covers 1,170 firms with 112,041 patents of which 12,518 were software patents. Newly public firms first appeared in the Compustat file within the last 5 years.

Table 4. Software Patents by Industry (1995-99)

	Software patents	All patents	Patents/R&D
Manufacturing	69%	85%	
Machinery (SIC 35)	27%	17%	2.5
Electronics (SIC 36)	22%	22%	2.8
Other	20%	45%	1.8
Non-manufacturing	31%	15%	
Software publishers (SIC 7372)	6%	1%	0.7
Other software services (exc. IBM)	2%	1%	4.4
Other non-manufacturing	3%	3%	2.8
Addendum: IBM	20%	9%	4.7

Note: covers 22,954 software patents and 109,509 total patents for firms the sample matched to Compustat. Last column shows patents granted per $10 million of R&D in 96 dollars.

Table 5. Appropriability Regressions (1981-99)

Dependent variable: Ln(number of patents granted)

	1		2		3		4		5	
	IV		IV		IV		IV		Heckit	
d_{81-85}	-.71*	(.15)	-.71*	(.14)	-.82*	(.13)	-.81*	(.13)	-.77*	(.11)
d_{86-90}	-.04	(.10)	-.05	(.10)	-.18	(.10)	-.20	(.09)	-.18	(.08)
d_{91-95}	.42*	(.11)	.31*	(.11)	.24*	(.09)	.12	(.10)	.10	(.08)
d_{96-99}					.60*	(.09)	.56*	(.09)	.73*	(.08)
Ln(cashflow)	.95*	(.06)	.52*	(.07)						
Ln(market value)					.80*	(.03)	.50*	(.05)	.19*	(.01)
Ln(lagged R&D)			.29*	(.03)			.23*	(.03)		
Capital/employee			.28	(.22)			-.13	(.14)		
U.S. inventors			-.21	(.11)			-.32*	(.09)		
Citations rec'd.			-.00	(.00)			-.01*	(.00)		
Self-citations			.24	(.09)			.36*	(.08)		
$d_{91-95} - d_{86-90}$	.46*		.36*		.42*		.32*		.28*	
$d_{96-99} - d_{86-90}$					.77*		.76*		.91*	
N	6,635		5,697		9,844		8,435		102,110	
Adjusted R-sq.	.813		.859		.784		.825			

Note: Standard errors are in parentheses and asterisk indicates significance at the 1% level. All regressions include firm fixed effects and year dummies. The δ's are coefficients on terms where the independent variable is the lagged software share of patents interacted with a time period indicator dummy. Market value is deflated millions. Cashflow is mean deflated cashflow (millions) for following four years. In the IV regressions, log cashflow and log market value are instrumented with lagged log deflated sales, lagged log employment and a dummy variable that equals one if the firm went public within the last five years. Capital per employee is deflated gross plant and equipment. U.S. inventors is the percent of patents with U.S. inventors. Citations received is the mean number of subsequent citations each patent receives; self-citations is the mean of upper and lower bound percentage of patents citing the firm's own patents (see Hall, Jaffe and Trajtenberg, 2002). The "Heckit" regression performs a Probit on whether the observation was included in the sample (whether the firm was matched to the patent database and had positive patents). Probit variables are log sales, log employment, a flag if firm went public during last 5 years, and year dummies. A likelihood ratio test rejects the hypothesis that the OLS and Probit equations are independent at the 5% level (the disturbances are weakly negatively correlated).

Table 6. R&D and Software Patents: Complements or Substitutes?

Dependent Variable: $\Delta \frac{R\&D}{sales}$ (columns 1-5)

	1	2	3	4	5	6
Δs	-.035* (.005)	-.036* (.005)	-.019* (.006)	-.030* (.005)		-5.97* (1.09)
Δs x (n >100)			-.047* (.010)			
Δs x (1991-95)					-.010 (.007)	
Δs x (1996-99)					-.048* (.007)	
Ln(sales)		.001 (.000)				
New public firm		-.001 (.004)				
Stock std. dev.		-.003 (.004)				
Industry f.e.	✓	✓	✓			
Firm f.e.				✓	✓	✓
N	3,464	3,416	3,464	3,464	3,464	3,417
Adj R-sq	.087	.124	.092	.579	.580	.550

Note: Standard errors in parentheses. All regressions include year dummies and either firm fixed effects or 2-digit industry dummies. Asterisk indicates significance at the 1% level. Dependent variable is change in R&D/Sales ratio for columns 1-5. For column 6, dependent variable is change in R&D/Employment. Time differences are five years; observations from 1991-99. Sample includes firms matched to patent file with positive patents. Covers 611 firms in an unbalanced panel. All regressions are weighted by $1/(1/n_t + 1/n_{t-5})$ where n is the number of patents granted. s is software share of patents granted. Excludes observations where R&D > ½ sales.

Table 7. Regression of R&D Intensity by Industry

Industry	SIC 35	SIC 36	Other manu.	SIC 73	Other Non-manu.
Δs	-0.009 (0.006)	-0.060* (0.014)	-0.028* (0.008)	.008 (.014)	-.067* (.015)
N	629	618	2,008	91	122
Adj R-sq	0.743	0.480	0.563	0.972	0.945

Note: Regressions same as previous table column 4, but for specific industries. Asterisk designates significance at the 1% level.