An Empirical Look at Software Patents

James Bessen*
Research on Innovation and MIT (visiting)

Robert M. Hunt**
Federal Reserve Bank of Philadelphia

May 2003

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 with other patents, software patents are more likely to be owned by large U.S. firms. Most are assigned to manufacturing firms; only 6% belong to software publishers. Our regression analysis finds that software patents have become a cheap form of appropriability. This cost advantage, not the profitability of software, accounts for most of their increased use. Also, software patents *substitute* for firm R&D rather than complement it. Their use is associated with substantially lower R&D intensity, 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. Also thanks to John Allison, Iain Cockburn, Bronwyn Hall, Joachim Henkel, Brian Kahin, Cecil Quillen, Leonard Nakamura, Eric von Hippel, Rosemarie Ziedonis and seminar participants at EPIP Munich, Federal Reserve Bank of Philadelphia, IDEI, MIT and the NBER. 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.

Introduction

The patentability of software-related inventions changed dramatically over the last three decades. During the 1970s, a number of federal courts described computer programs as mathematical algorithms, which are unpatentable subject matter under U.S. law.[1] Of course, software is always used as part of a system that includes hardware. Throughout the 1970s, such systems could be patented, but only if the novel aspects of the invention did not reside entirely in the software.[2]

This interpretation began to change after the Supreme Court decision in *Diamond v. Diehr* in 1981.[3] Thereafter, a series of court and administrative decisions—what we call regulatory changes—gradually relaxed the subject matter exception that restricted the patenting of software-related inventions (Hunt 2001). Any remaining uncertainty over the patentability of computer programs was eliminated with the publication of new examination guidelines by the U.S. Patent and Trademark Office in 1996 (USPTO 1996). During this same period, new legislation and 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 the last two decades. First, we explore the general characteristics of these patents and who obtains them. Using a broad definition of "software patent," we assemble a comprehensive database of all such patents. We find that over 20,000 software patents are now granted each year, comprising over 15% of all patents. Compared with other patents, software patents are more likely to be assigned to firms, especially larger U.S. firms, than to individuals. They are also more likely to have U.S. inventors. Surprisingly, most software patents are assigned to manufacturing firms and relatively few are actually assigned to

---

[1] See, for example, the Supreme Court decision in *Gottschalk v. Benson*, 409 U.S. 63 (1972).

[2] *Parker v. Flook* 437 U.S. 584 (1978).

[3] *Diamond v. Diehr* 450 U.S. 175 (1981).

firms in the software publishing industry (SIC 7372). It appears that most software patents are acquired by firms in industries that accumulate large patent portfolios.

In the second part of the paper, we examine 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 develop 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. This permits us to build on the empirical literature of "patent production functions" (including Scherer 1965, Bound et al.., 1984, Pakes and Griliches, 1984, Griliches, Hall and Hausman, 1986, Hall and Ziedonis, 2001).

As is well known, the number of patents a firm obtains need not correspond to 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. 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 or protect themselves from potential hold-ups (Bessen 2003, Hall and Ziedonis 2001). In general, firms can increase their appropriability—the share of potential monopoly rents they earn on inventions—by obtaining more patents. So there is a tradeoff between appropriability and patent cost that determines the optimal number of patents for each firm. In effect, patents are a kind of factor input for producing rents.

This economic treatment of patenting behavior permits us to use familiar econometric models. First, we use a derived factor demand equation for patents to explore the dramatic rise in software patents. Although general patenting rates have more than doubled over the last two decades, software patenting rates have risen much faster. We explore how much of this increase can be attributed to changing standards—the "regulatory" hypothesis—and how much can be attributed to more productive use of software in performing R&D—the "productivity" hypothesis. Using a demand equation for patents, we are able to estimate the effect of changes in the cost of patenting while controlling for profits, that is, while controlling the productivity effect.

Next, we look at the effect of software patents on R&D. It is sometimes argued that regulatory changes that facilitate patenting will increase investment in R&D. We call this the "incentive" hypothesis. For example, such arguments have been raised in support of extending patent coverage more directly to software in Europe.

We test this hypothesis by evaluating whether R&D and software patents are complements or substitutes. If the incentive hypothesis is correct, then software patents should complement R&D, and the substantial rise in software patenting should have significantly increased R&D investment. We use a familiar cost share equation to estimate the sign of the elasticity of substitution between software patents and R&D, after controlling for software use and other factor inputs, including IT.

The first section describes our data and provides descriptive statistics on software patents and who obtains them. The next section explores factors that may explain the increase in software patenting, and the third section examines whether software patents and R&D are complements or substitutes. The fourth section concludes.

I. Data and Descriptive Statistics

A. What Is a Software Patent?

How many software patents are being granted? Although the patent office maintains a system for classifying patents in terms of technology and its applications, this system does not classify inventions as software or something else. 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.

We do not use such a definition for two reasons. First, beginning with *Diamond v. Diehr*, attorneys have drafted software patents in such a way that they did not necessarily *appear* to be patents on software.[4] This makes the determination of a “pure” software patent somewhat arbitrary and impractical for a comprehensive database. Second, we do not necessarily assume that the subject matter exclusion was the *only* difference between software patents and other patents; in fact, we find both legal and empirical evidence below that software patents have been treated differently even when they are not necessarily “pure” software patents. So it is useful to study a somewhat broader range of patents in any case.

Our approach aims to include all patents for inventions that use software. Nevertheless, it appears that most of the patents we identify as software patents are “pure” software patents, at least during recent years. Allison and Tiller (forthcoming) performed a careful evaluation of “pure” software patents, examining 1,000 randomly selected patents from 1996-98 in detail. They found that 9.2% of these patents were totally embodied in software.[5] 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 consists mostly of “pure” software patents.[6]

B. Our Data

For this paper, 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. We are interested in the characteristics of these patents, so we match this list

---

[4] For instance, Cohen and Lemley argue (2001, p. 9), “The Diehr decision and its appellate progeny created what might be termed ‘the doctrine of the magic words.’ Under this approach, software was patentable subject matter, but only if the applicant recited the magic words and pretended that she was patenting something else entirely.”

[5] 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).

[6] Hunt (2001) explored several other methods of counting software patents and reports qualitatively similar patterns.

against patents contained in the NBER Patent Citations Data File (Hall, Jaffe, Trajtenberg, 2001) to obtain characteristics of each patent.

We are also interested in the characteristics of firms that obtain patents on software or hardware. The NBER data set links patent numbers to the assignees (usually the firm that first owned the patent). We use that match to obtain financial data from Compustat on publicly held U.S. firms that are also patent holders. In addition, the largest 25 public firms in the software publishing industry (SIC 7372) were added. (These firms obtain few software patents, so only one was included in the NBER match file.)

Our data set contains annual patent count data for 1,646 individual U.S. firms, including financial data on these firms for the years 1981-99. These matched firms accounted for 42% of all U.S. software patents and 38% of all U.S. patents issued during 1981-99.

C. The Number of Software Patents

Table 1 reports the number of software patents and other patents granted per year. As can be seen, their 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 alone 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).[7] 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. But as noted above, only about a quarter of the patents in our database for the late 90s do not qualify as "pure" software patents.

D. Characteristics of Software Patents and the Firms That Obtain Them

Table 2 shows characteristics of software patents compared with 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.

---

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

assignees and to have U.S. inventors.[8] After the U.S., the top countries ranked by inventors are Japan (18%), Germany (3%), Great Britain (2%), and Canada (2%). Consistent with the findings of Allison and Lemley (2000), software patents tend to receive a larger number of subsequent citations, and they have substantially more claims per patent, both measures sometimes interpreted as indicators of value. In other aspects they are similar to other patents.[9]

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.[10] 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 that obtain the most software patents.

To obtain more information about the firms that obtain software patents, we show means of firm characteristics weighted by the number of patents (software and total) the firm receives in Table 3. Relative to other patents, software patents tend to be obtained by firms 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%.[11]

---

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

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

[10] 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 seem to suggest that is so.

[11] The data set over-samples 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.

The distribution of software patents across industries appears to reflect something other than the *creation* of software. Table 4 also includes two measures that might better reflect software creation: the share of programmers and systems analysts employed in the industry and the ratio of total patents to R&D. The manufacturing sector acquires 69% of software patents, but employs only 10% of programmers and analysts; software publishing and services (including IBM) acquires only 16% of software patents, but employs 42% of programmers and analysts. Of course, it may be the case that engineers also write software, especially embedded software. If we add all types of engineers except civil engineers to our employment figures, then manufacturing accounts for 32% of potential software writers and software publishing and services account for 25%—manufacturing still acquires many more patents per potential software writer. This disparity also appears in the last column showing differences in patent propensity. Software publishing firms get only a quarter 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).[12]

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 all firms are equally likely to obtain software patents to protect individual software inventions. In fact, the firms that acquire the largest share of software patents appear to be just those firms that acquire large portfolios of patents for strategic purposes: large firms in electronics, computers, instruments industries, and IBM. As we argue below, software patents may be particularly useful for firms building strategic patent portfolios.

[12] Also, 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 this industry acquires only 6% of software patents.

II. Explaining the Rise in Software Patenting

A. Background: Patent Propensity

The rate of software patenting displayed in Table 1 has risen much faster than either R&D or the use of software in the economy. From the late 80s to the late 90s, annual software patent grants have increased nearly six-fold while software sales have only tripled as a share of GDP and real industry R&D investment has only gone up 35%.[13] This growth occurred against a general background of rising "patent propensity," that is, the ratio of all patents to real R&D grew about 40%.

To explore this trend, we build on the empirical work of Hall and Ziedonis (2001). They document a dramatic increase in patent propensity among U.S. semiconductor firms since the early 1980s. They estimate a Poisson regression that relates the log of patent applications for each firm to the firm's log employment, log R&D per employee, log capital per employee, and time dummies. Since the right-hand side variables capture R&D and firm size, the time dummies capture changes in patent propensity over time. These time dummies increased sharply beginning in the late 1980s.

To fix ideas and to verify the growth in patent propensity in our data, we performed a similar Poisson regression for all firms in our dataset, excluding the semiconductor industry. The normalized time dummies from this regression, together with those from Hall and Ziedonis, are shown in Figure 1.[14] The regression coefficients are in Table 5. These industries also exhibit a rise in patent propensity beginning in the late 1980s, but only about half as strong as for the semiconductor industry. This is not surprising; there is no reason to expect that all industries would respond elastically to changes in patent standards. One might expect that industries with complex technologies might respond more elastically than industries with "discrete" technologies, such as

[13] Total industry R&D increased from $121 billion in 1988 to $164 billion in 1998 in 1996 dollars (NSF, 2003). 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). 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.

chemicals. This would be the case if patent propensity rose as the result of strategic patent portfolio behavior encouraged by lower patenting standards. Since discrete technologies offer fewer opportunities for patent portfolio strategies, they might not show the same increase in patent propensity. Indeed, the chemical industry, SIC 28, is also shown in Figure 1 and it does not exhibit any rising patent propensity.[15]

What explains this general increase in patent propensity and the increase in software patenting in particular? Two major factors have been proposed: the productivity of R&D and legal/regulatory changes to the U.S. patent system. These factors affect both the general increase in patent propensity and also the *relative* increase in software patenting. Disentangling these two groups of factors is central to the economic analysis of the phenomenon.

The "productivity" hypothesis attributes most of the growth in patenting to greater productivity of R&D. 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. More productive R&D generates more inventions, hence more patents. Software patenting may be an instance of this broader trend, e.g., software may have been especially productive in the design of new products.

The "regulatory" hypothesis attributes most of the growth in patent propensity to the direct and indirect effects of regulatory changes. Evidence suggests that examination standards have fallen while it has become easier to enforce patents. Some researchers attribute changed standards to the creation of a unified appeals court for patent suits in 1982 (Merges, 1997 and Hunt 1999a). That court—the Court of Appeals for the Federal Circuit—relaxed the requirement that patents be granted only for inventions that are not obvious to "practitioners skilled in the art" (Coolley 1994, Dunner et al. 1995, Hunt 1999b, Lunney 2001). At the same time, the Federal Circuit *raised* post-grant standards in

---

[14] Our regressions use patent grant rates as the dependent variable while Hall and Ziedonis use patent application dates. Figure 1 lags their results two years to compensate. Also, we use firm fixed effects in the Poisson regression.

[15] Hall (2003) finds that the increase in patenting occurred across technology classes but was concentrated in industries broadly related to electronics, including semiconductors and computers. This is consistent with an interpretation that these industries experienced a shift to strategic portfolio patenting.

favor of patent holders. It also raised the evidentiary standards required to challenge patent validity and broadened the interpretation of patent scope (Rai 2000, Merges 1997). The Federal Circuit 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).[16] Plaintiff success rates have increased substantially in patent infringement suits (Lerner, 1995).

Other regulatory changes specifically affect software patents. First and foremost, the courts and the patent office gradually eliminated the subject matter exception to patents on mathematical algorithms, as described in the introduction. In addition, some researchers argue that the patent office was not prepared to examine applications for patents on software, and consequently many "bad" patents were granted (Jaffe 2000, Kesan 2002, Merges 1999).

Perhaps more important, courts reduced the "enablement" requirements for software patents. The enablement is the specification of detailed instructions explaining how the invention works. It is supposed to be a "best mode" example of all of the patent claims. For software patents and business methods, the courts have largely eliminated this requirement (Burk, 2002, Burk and Lemley, 2003 forthcoming).[17] This means, in the words of an IBM patent attorney, "[The patent standard] currently being applied in the U.S. invites the patenting of ideas that may have been visualized as desirable but have no foundation in terms of the research or development that may be required to enable their implementation" (Flynn, 2001). The new standards permit software and business method patents that may require relatively less R&D expense, that is, these are "cheap" patents.

---

[16] 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).*

[17] Reviewing case law, Burk and Lemley (2003, forthcoming, pp. 9-10) write "For software patents, however, a series of recent Federal Circuit decisions has all but eliminated the enablement and best mode requirements. In recent years, the Federal Circuit has held that software patents need not disclose source or object code, flow charts, or detailed descriptions of the patented program. Rather, the court has found high-level functional description sufficient to satisfy both the enablement and best mode doctrines." See also Cohen and Lemley (2001) on different treatment of software patents.

These differences are important not only because they affect patent propensity, but also because they reflect on whether the patent system is, in fact, a "one-size-fits-all" system that treats all technologies equally. Proponents of software patents have argued that different technologies should be offered the same intellectual property protections and, for this reason, software subject matter should not be excluded from patent protection. This argument is undermined if, in practice, software patents are given a significantly different treatment.

B. An Economics of Patenting

Our empirical strategy is to estimate firms' relative propensity to obtain software patents, after controlling for productivity and general changes in patent propensity. This allows us first to test the "one-size-fits-all" hypothesis and then to estimate the contribution of software-specific regulatory changes to the overall increase in patent propensity. To do this, we present a general economic model of patenting activity.

Most of the theoretical literature has little to say about the sort of regulatory changes described above.[18] In particular, patent standards have little role in patent race models that assume: 1) a patent delivers a certain monopoly right, and, 2) a single, unique patent covers an entire product market (i.e., there is a one-to-one correspondence between patents and product monopolies). These assumptions might reasonably approximate conditions in, say, the pharmaceutical industry, but software products are highly complex, they potentially involve hundreds of patents each, and patent scope and enforcement are uncertain. Under these conditions, it pays firms to build portfolios of patents.

Assuming that each additional patent increases the firm's joint probability of winning at litigation, a larger patent portfolio provides greater expected rents. It also provides a stronger bargaining position if the firm enters into cross-licensing negotiations. Either way, there is a monotonic relationship between a firm's patent portfolio size and its expected rents from innovation. For a given industry and regulatory environment,

[18] An exception is the literature on sequential innovation, which considers non-obviousness standards (Hunt, 2002, O'Donoghue, 1998). The theoretical discussion here is developed more fully in Bessen (2003).

temporarily ignoring any strategic interaction between firms, actual rents, V, can be written

(1) $V(n, \mathbf{X}) = m \cdot A(n) \cdot Q(\mathbf{X})$

where m is the monopoly markup, A is the appropriability function, Q is output (taking output price as *numeraire*), and $\mathbf{X} = \{K, L, M, R...\}$ is a vector of factor inputs, including R&D, R. If $A = 1$, the firm earns the entire monopoly rent, but if $A = 0$, competition dissipates those rents completely.[19] Also, $A = A(n)$, where n is the size of the firm's patent portfolio. Since firms will obtain patents first on those inventions that deliver the greatest marginal appropriability, we assume that A is increasing and concave in n.

But we know from survey evidence that firms do not, as a rule, obtain patents on all or even most of their inventions (Mansfield, 1986, Cohen et al., 2000). That is because the benefits of greater appropriability are sometimes less than the cost of obtaining a patent, which we denote c and assume to be a constant. Firms assemble patent portfolios, taking into account the tradeoff between increasing appropriability and the cost of obtaining patents. That is, firms maximize profits, $p = V - c \cdot n$, with respect to portfolio size, n, by solving a first order condition for optimal portfolio size, $\hat{n}$:

(2) $\hat{n} = n$ such that $\frac{\partial V}{\partial n} = c$ or $\frac{\partial \ln A}{\partial n} = \frac{c}{V}$.

In words, patenting itself is an economic activity distinct from, but related to, innovation.

Equation (1) can be thought of as a "rents production function," that is, a firm's function for generating rents using conventional factor inputs, R&D *and* patents. Then the solution to the first order condition on patents, (2), generates a derived patent demand function. With a convenient specification of functional form, the log demand for patents can be written (see Appendix):

(3) $\ln \hat{n} = \ln D(V, c) = a + b \ln V - b \ln c$.

This is, of course, quite similar to a factor demand equation for a Cobb-Douglas cost function. In fact, there is some evidence that patenting responds rather elastically to patenting costs. For example, in 19th century Britain, patent applications tripled or

[19] The latter outcome can happen only if imitation is costless and trade secret protection is infeasible.

quadrupled within a year or two in response to patent fee changes in 1852 and 1883 (Macleod et al., 2002).

This equation is also similar to the equation that Hall and Ziedonis use for their Poisson regression and which we use in Table 5. There are two main differences. First, (3) includes a measure of profits, *V*. Although one might argue that the combination of employment, capital per employee and R&D per employee is a good proxy for profits, the explicit identification of profits as a right-hand variable is important for capturing the productivity effect.[20] Specifically, if R&D has become more productive, that effect should be captured entirely in the firm's future profit stream. This means that the other terms capture any regulatory effects on patent propensity.

Second, this equation captures the effect of regulatory changes through the cost of patenting, *c*. Since *c* is unobserved, we might choose to capture these effects through time dummies. However, we can also capture the differential effect of software patents on patenting cost as developed in the next section.

C. Software Patents and the Demand for Patents

We can capture the relative effect of regulatory changes for software patents with an additional instrument, namely, the share of software patents in total patents, *s*. If software patents have a different cost of patenting than other patents, then the software share proxies for the relative cost of patents. Designate the costs of obtaining patents as c_s and c_o for software patents and other patents, respectively. Suppose also that the marginal appropriability of software patents changes across firms and over time. In the Appendix we show that as a first order approximation for small *s*, for firm *i* at time *t*, (3) becomes

[20] The Hall-Ziedonis paper is part of a larger literature that has estimated "patent production function" equations (Scherer 1965, Bound et al., 1984, Pakes and Griliches, 1984, Griliches, Hall and Hausman, 1986). These studies often used R&D on the right-hand side and then interpreted constant or dummy terms as measures of patent propensity. But it was recognized that this was a reduced form equation—R&D generates the rents (Pakes and Griliches, 1984, Griliches, 1990).

(4) $$\ln \hat{n}_{it} \approx m_t + b \ln V_{it} + b \frac{(c_o - c_s)}{c} \hat{s}_{it}$$

where c is now average patent cost, $c = (1-s)c_o + s c_s$, and $\hat{s}$ is the firm's optimal software patent share. The coefficient of $\hat{s}$ captures relative changes in the cost of software patenting. Changes in the cost of other patents are now captured in the time dummies, m_t.

Of course, the relative cost of software patents is unobserved. However, by interacting s with time dummies, we can measure changes in this coefficient over time:

(5) $$\ln \hat{n}_{it} = m_t + b \cdot \ln V_{it} + \sum_j d_t \cdot I(j=t) \cdot \hat{s}_{it}$$

where $I(j{=}t)$ is an indicator function (1 if $j{=}t$, otherwise 0). 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 coefficients d_t capture the relative effect of regulation on software patenting independently of the relative productivity of software, which is captured in the V term. If $d_t = 0$, then the "one-size-fits-all" hypothesis is true. That is, software patents are treated no differently than other patents and, consequently, the software share of patents has no effect on overall patent demand (optimal portfolio size). If $d_t < 0$, then software patents are more costly than other patents and greater use of software patents is associated with lower rates of patenting. This is the situation we might expect to have occurred during the early 80s when software patents were still restricted. On the other hand, if $d_t > 0$, then software patents cost less than other patents and their use is associated with a higher patent propensity. This might be the case if, say, the relaxed enablement requirement for software patents significantly reduced firms' cost of obtaining software patents.

However, the coefficients d_t may also pick up effects of strategic interaction between firms. The above analysis ignored strategic interaction in order to simplify the exposition. However, in a model of strategic interaction (see Bessen, 2003), firms may pursue patent portfolio strategies that reduce the appropriability of rivals' portfolios.

Moreover, there is reason to expect that such strategic behavior might be associated with low patent standards and low patenting cost. That is because low patenting costs may encourage firms to pursue aggressive patent thicket strategies (low cost makes it more affordable to build a patent thicket). Rivals may, in response, obtain more "defensive" patents.[21] To the extent that low cost software patents encourage strategic patenting (offensive or defensive), then software patents will be associated with strategic portfolio building. If and only if this correlation occurs, estimates of d_t will be larger (more positive) than in the non-strategic case. This means that the interpretation of the case where $d_t > 0$ could involve strategic patenting. However, if the "one-size-fits-all" hypothesis holds, then software patents will be treated just the same as other patents and there can be no specific correlation between software patents and strategic patenting. Then it still must be true that $d_t = 0$.Similarly, if software patents are more costly than other patents, strategic patenting will not be associated with software patents and it will still be true that $d_t < 0$.

To get a first look at the effect of software patents, we repeated the above Poisson regression adding interaction terms as in (5). The time dummies do not increase very much during the 1990s (Figure 1). It appears that software patents, either directly or indirectly, account for much of the increase in patent propensity outside of semiconductors.

Note also that the coefficients d_t change from being negative in the early 80s to large and positive in the late 90s. This implies that software patents went from being less advantageous than other patents to being much more advantageous. Indeed, the coefficient for the late 90s implies that each additional software patent was associated with an increase of almost *two* total patents. This suggests that not only do software-related regulatory changes explain the relative increase in software patenting, but that they also explain much of the *general* increase in patent propensity. This also suggests that software patents are associated with portfolio-building behavior.

---

[21] In other words, in such an equilibrium, patents are strategic complements.

However, it is possible that these estimates of d_t are overstated because this regression may not have properly captured firm profits or because of some estimation issues considered next.

D. The Effect of Software Patents on Patent Demand

To obtain more conservative estimates of d_t, we need to address several estimation issues. First, the model above is static while our estimation represents a single year in a dynamic real world. This means that our measure of profits should be the expected value of future profits and our measure of patents should be the flow of patents targeted to provide the optimal portfolio size when those profits are realized. 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* is 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.

To capture the expected stream of future profits, 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 cash flow for the following four years (measuring cash flow as operating income before depreciation plus R&D spending, if not missing). Both of these measures may differ from expected profits because of measurement error—"animal spirits" for market value or forecasting error for cash flow. To correct for this possibility, we perform an instrumental variable estimation (see below). Also, it is possible that these profit measures (and instruments based on current year variables) might be positively correlated with the error term—e.g., firms will adjust profit expectations upward upon receiving "windfall" patents. To prevent endogeneity problems, we use lagged instruments for $E[V]$.[22]

The variable s is measured as the simple share of software patents in the total patents granted to a firm in a given year. 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, $s_{i,t-1}$.[23] 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. To correct for this sampling variance, we weight the regressions by n.

Time dummies and broad industry dummies are included in the regressions to account for general trends in patenting and industry differences. 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 upward. Other possible control variables are included below.

Taking account of these considerations, the actual equation we estimate is

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

where I is the indicator function described above, T represents four different groups of years (1981-85, 1986-90, 1991-95, 1996-99), and all variables are deviations from means.

The regressions are shown in Table 6. The first column uses forward cash flow to proxy for expected profits (with a truncated sample period). As in the Poisson regression, the estimates of δ show a distinct upward trend: they are initially negative and become significantly positive by the early 90s.

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-

---

[22] We instrument with log sales, log employment lagged one year and a flag for new firms. 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.

[23] We also tested a two year lag, which produced similar coefficients but larger standard errors.

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, so we include subsequent citations as a measure of quality. And firms that cite their own patents heavily may be engaging in strategic "fence building" behavior, getting more patents to block competitors from using related technologies. The coefficient on patent citations received is consistent with the literature, suggesting that patent "quality" may substitute for quantity (Trajtenberg, 1990, Harhoff et al., 1999, Lanjouw and Schankerman, 1999, Hall et al., 2001). The coefficient on self-citations is positive and significant, which suggests that above average self-citation is related to portfolio-building behavior. Nevertheless, the estimates of δ are similar to those reported in the previous column.

The third and fourth columns repeat these estimations using market value as the proxy for profits. This permits estimates for the late 90s. The estimates of δ are similar. While some of the coefficients changed, the difference in coefficients, $d_{91-95} - d_{86-90}$, changed only slightly. Also, the estimates for the late 90s in Column 4 suggest that the cost advantage of software patents continued to increase.

These regressions include industry dummies, but do not correct for firm heterogeneity. Yet Hall and Ziedonis found important sources of firm heterogeneity. Since our regression is implicitly dynamic, firm fixed effects might be correlated with other regressors. So, to correct for firm heterogeneity, Column 5 shows the results after taking five-year differences. Note that the meaning of the coefficients of the interacted *s* terms is now different. For example, the coefficient of s for the period 1991-95 estimates $d_{91-95} - d_{86-90}$. The values of $d_{91-95} - d_{86-90}$ and $d_{96-99} - d_{86-90}$ are shown at the bottom of the table. Despite the larger standard errors introduced by differencing, the estimated changes in these parameters is somewhat larger than in Columns 3 and 4. The latter estimate implies that for each additional software patent obtained during the 90s, the total number of patents increased by *two*.

We also performed a Heckman two-stage regression to explore sample selection (not shown). First, we performed a probit where the dependent variable is one if the observation is included in the least squares regression and zero otherwise. An observation is 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 the firm went public during the last five years, and year dummies. A likelihood ratio test weakly rejects the null hypothesis that the disturbances of the two equations are uncorrelated (they are negatively correlated, but only at the 5% level). The resulting coefficients, however, are quite similar to those in Column 3. Finally, we also performed several regressions with longer lags; these did not change the coefficients substantially, but did increase the standard errors.

Thus the strong shifts in estimates of d_t over time appear to be robust. We draw three conclusions from these estimates. First, we can firmly reject the "one-size-fits-all" hypothesis; software *is* still treated differently. Software patents shifted from being relatively disadvantageous during the early 80s to being strongly advantageous during the late 90s. This concurs with the anecdotal evidence.

Second, because these estimates significantly exceed one during the late 90s, we infer that software patenting must be correlated with broader changes in patenting behavior, such as a shift to more strategic portfolio behavior. Note that in equation (4), representing the situation without strategic interaction, the coefficient of *s* cannot exceed β, as long as prices are non-negative. Since our estimates of this coefficient are much larger than β during the late 90s, this suggests that firms are using cheap software patents to build patent thickets. Indeed, each additional software patent is associated with an increase of the total portfolio of nearly *two* patents.

Third, the magnitude of the shift implies that regulatory change associated with software patents substantially affected not only the rate of software patenting, but the overall rise in patent propensity. We made some thumbnail calculations using the trends in total patenting and R&D cited above and the coefficients from Columns four or five of Table 6. Software patenting changes account for about half of the total rise in patent propensity.[24] The remainder, of course, can include both increases attributable to greater productivity of R&D and to regulatory changes not associated with software patents; these contributions are still unknown. Nevertheless, it is clear from the effect of software patents alone, that regulatory change played a very substantial role in the growth of patent propensity.

More generally, the productivity hypothesis implies that the rapid increase in patenting reflects growing returns from innovative activity and therefore the increase should be welcomed. The regulatory hypothesis, on the other hand, raises the possibility that regulatory changes may adversely affect innovative activity. We explore the effect of software patents on R&D in the next section.

III. Software Patents and R&D

A. Complements or Substitutes?

Our cost function approach provides a natural framework to explore the relationship between patenting and R&D. In particular, we can examine whether software patents and R&D are complements or substitutes.

Our analytical framework permits us to test a common argument about patents, which we call the "incentive" hypothesis. This is the hypothesis that regulatory changes that increase patenting (lower cost, relaxed standards, broader subject matter, etc.) provide stronger incentives for R&D and, therefore, patents serve to increase (complement) R&D. For instance, advocates for software patents in Europe argue that software patent subject matter extensions increased R&D incentives in the U.S.

Our empirical approach is simple. The dual to maximizing profits in (2) is a cost function that minimizes factor costs given a fixed level of rents. Assuming constant returns to scale and specifying a translog form, a standard transformation yields a series of estimable equations relating factor shares to log prices (see Greene, 1997, Chapter 15.6). For R&D, the factor share equation can be written:

---

[24] Using aggregate data, from 1988 to 1998, the log of patent propensity (patents/real R&D) increased 0.33 and the software share of patents increased .093. Applying the differences in Columns 4 and 5, 1.73 x .093 = 0.16 and 1.96 x .093 = .18, corresponding to 48% and 55% of the increase in log patent propensity.

(7) $$\frac{R}{Q} = a + g_0 \cdot \ln c + \sum_j g_j \cdot \ln p^j$$

where c is a measure of the costs of patenting and the p^j are factor prices. Since $c = (1-s)\,c_o + s\,c_s$, for small s, we can perform a similar substitution to the one in equations (4) and (5) above: $\ln c \approx \ln c_o - \frac{c_o - c_s}{c_o} s$. Assuming firm heterogeneity, and incorporating c_o in the time dummies, this yields an equation of the form

(8) $$\frac{R_{it}}{Q_{it}} = m_i + a_t + \sum_j f_t \cdot I(j=t) \cdot s_{it} + \sum_j g_j \cdot \ln p_t^j \ .$$

To control for firm heterogeneity, we take five-year differences. Five-year differences should reduce noise from measurement error and any biases associated with partial factor adjustment. All regressions include year dummies. The actual equation estimated is then (re-parameterized)

(9) $$\Delta \frac{R_{it}}{Q_{it}} = a_t + \sum_j f_t \cdot I(j=t) \cdot \Delta s_{it} + \sum_j g_j \cdot \Delta \ln p_t^j + e_{it}$$

We perform Weighted Least Squares to correct for sampling error in our key variable, the software share of patents.[25]

The price variables are industry price indices from the BLS (April 2001 release) at the two-digit level for manufacturing and for the private non-farm business sector for non-manufacturing firms. In addition to capital, labor, materials, energy, and purchased services, we include the index for the price of IT capital, to control for the influence of IT.[26]

Results for 1985-99 are shown in Table 7. The first column shows just the standard factor prices. The coefficient for labor is positive and significant and the coefficients for materials and purchased services are negative and significant. These

---

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

[26] Thanks to Bill Gullickson and Steve Rosenthal of BLS for providing data. We also used separate indices for the components of IT capital, but these did not change the non-price coefficients and they were highly multi-collinear.

coefficients, g_j, can be used to calculate the Allen partial elasticities of substitution with R&D. For the *j*th factor, the elasticity of factor substitution is (Greene, 1997, p. 696)

$$(10) \qquad q_{Rj} = 1 + \frac{g_j}{f_R f_j}$$

where f_R and f_j are the respective cost shares. Although we do not have a fully consistent set of cost share estimates (the R&D shares are "double counted"), we can obtain rough estimates using sample means. We find that labor substitutes for R&D ($q_{RL} > 0$), and materials and services complement R&D ($q_{RM}, q_{RS} < 0$, but weakly for materials). These results are similar to those obtained by Mohnen et al. (1986), who found that labor substitutes for R&D and that capital and materials complement R&D in the U.S.

Column 2 adds terms for the change in software share interacted with time dummies. These coefficients are positive and significant in the late 80s, becoming negative and significant in the late 90s. This implies that patents were a complement to R&D during the late 80s but became substitutes for R&D during the 90s with a very strong effect in the late 90s (the sign of the coefficient of *s* is opposite the sign of the coefficient of *c*). This pattern mirrors the changes in the cost of patenting found in Table 6. In simple terms, firms that increased their share of software patents also decreased their R&D intensity in the 90s.

Column 3 adds additional controls for firm size (log deflated sales), new firms, firm risk (standard deviation of annual stock price growth), and liquidity (change in long-term debt divided by capital). The last two coefficients are significant. Consistent with capital theory (in a dynamic model), riskier firms tended to decrease their R&D spending. Consistent with studies by Switzer (1984) and Hall (1990), liquidity may constrain R&D spending—firms that increased their relative debt levels also decreased their R&D. But these controls did not substantially alter the coefficients on software share.

The dependent variable in the first three columns is the change in the ratio of R&D to sales, effectively the output share of R&D. However, if there are non-constant returns to scale or a positive markup, then the output share may differ from the cost share. This might possibly introduce a bias to our results, especially if greater appropriability increased the realized markup. Thumbnail calculations suggest that this bias should not be large. Nevertheless, in the fourth column we use a measure of the cost share: R&D to (sales less pretax income). This is, of course, a somewhat noisy measure of operating costs and so our standard errors increase substantially. Nevertheless, our coefficient for the late 90s finds a highly significant and rather larger substitution effect, suggesting that this bias is not an issue.

An important issue for the interpretation of these results is the role of software use. We want to distinguish effects that arise from using software from effects that arise specifically from software patenting. Anecdotal evidence indicates that firms use software in designing products, both using existing tools, like computer-aided engineering software, and by using embedded software in hardware products that were formerly hard-wired. It is possible that firms that develop embedded software for use in new products may obtain more software patents. Also, such use of software in designing products may make R&D less costly to perform, so that firms spend less on R&D. Then an increase in software patenting could be associated with decreased R&D intensity. Note, however, that this will be true *only* if the demand for R&D is relatively inelastic. Otherwise, any reductions in the effective cost of performing R&D should *increase* R&D intensity—then if software makes R&D more productive, firms will want to spend even more on R&D.

However, we know from studies of the R&D tax credit that the demand for R&D is rather elastic with respect to its tax price. Berger (1993) finds that reductions in the tax price of R&D *increased* R&D intensity. In a survey of the literature, Hall and van Reenen (1999) find that the tax price elasticity of R&D is around -1; at this level, changes in the price of R&D should have little effect on R&D intensity. Thus, this evidence suggests that the greater use of software in the performance R&D cannot explain the measured decrease in R&D intensity.

Nevertheless, to further explore this issue, in Table 8 we perform regressions from 1991-99 without interacting the software share variable with time dummies. Column 1 presents a baseline regression. The software share coefficient again indicates a strong substitution effect midway between the previous estimates for the early and late 90s. The price coefficients are similar to those in Table 7. Labor substitutes for R&D and materials complement it, but now energy is a complement and not services.

Also, IT appears as a substitute for R&D. This is an important control—IT capital includes computers and software but also communications and other equipment. The IT price declines faster in industries that use more computers and software relative to communications equipment. So this variable captures the effect of the firm's total IT, especially software and computers. Clearly, the role of software patents is independent of this variable.[27] However, this captures the use of IT in the firm as a whole and this might be poorly correlated with the use of software within the R&D lab specifically.

Column 2 explores the role of software use another way. Perhaps the observed effect occurs only in software-related industries. This regression interacts the change in software patent share with a dummy variable if the firm is in SIC 35, 36, 38 or 73. The coefficient of the interaction variable is small and not statistically significant. It appears that the substitution effect is no stronger in software-related industries than in other industries. A variety of other regressions (not shown) confirm that the effect is observed in a wide range of industries.

Column 3 captures firms' software development effort more directly. This regression includes the change in the industry share of employment of programmers and systems analysts from the late 80s to 1998.[28] Firms in industries that increased this share should have increased their software development efforts. However, the regression shows that greater software development effort is associated with *increased* R&D intensity. It seems hard to argue that greater software development involved in new product R&D is responsible for decreasing R&D intensity. Moreover, note that the substitution effect is still large and highly significant, although slightly lower in magnitude, as might be expected, given that programmers may be included in R&D spending.

---

[27] If the IT price term is dropped from the regression, the coefficient on the change in software share changes only to -.052.

[28] Using data from the Occupational Employment Survey of the BLS we calculated the employment of computer programmers and systems analysts as a fraction of total employment of the 3-digit SIC industry. We then took the difference between this share for 1998 and 1987-89, depending on which year the industry was surveyed. Note that this change in employment shares is negatively correlated with the change in software patent shares, suggesting that these two variables measure rather different quantities

Thus, the measured substitution effect appears to occur independently of the use of software, and therefore, software patent share is not spuriously serving as a proxy for software use.

Column 4 further explores the effect of firm size by interacting the change in software share with a dummy that is one if the firm has more than 5,000 employees. Large firms do exhibit a stronger substitution effect, but smaller firms still have a substantial and statistically significant negative effect. This result is consistent with a model of strategic patenting behavior—incumbent firms are more likely to pursue aggressive patenting behavior that may reduce R&D incentives (Bessen, 2003).

We also performed several Heckman two-stage regressions similar to the one in the previous section (not shown). Here, however, we could not reject the hypothesis that the sample selection probit was independent of the least squares regression. Sample selection within the Compustat sample does not appear to have biased our results.

Finally, we considered the role of accounting changes on the relationship between software patents and R&D intensity. 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 80s and early 90s, yet our substitution effect is strongest in the late 90s. 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.

B. Interpretation

Thus, our analysis appears to decisively reject the incentive hypothesis during the 1990s. Software patents may have complemented R&D during the early 80s—when patenting standards were still relatively high—but they substituted for R&D during the 1990s.[29] Regulatory changes increased the amount of patenting, but they are also associated with lower R&D. We can reject naïve arguments that more patents, relaxed standards, or lower patenting costs lead to more R&D.

Moreover, the observed substitution effect is quite large. Thumbnail calculations imply that at the end of the 90s, R&D would have been about 10-15% higher without the substitution. Of course, what we observe here is *not* necessarily a causal relationship and our regression results do not indicate that the changes in patenting necessarily caused declines in R&D. Combining theoretical analysis with case studies of individual firms, we can identify at least four explanations for the substitution effect.

First, established firms in maturing technological industries might decrease their R&D intensity at the same time they strategically increase their patenting. For example, when Louis Gerstner Jr. became CEO of IBM in 1993, he slashed R&D by over $1 billion and also shifted the business much more toward services, so that the R&D-to-sales ratio has declined steadily. At the same time he initiated a much more aggressive patent licensing strategy—perhaps with declining profit opportunities from R&D, more aggressive patent use offered an alternative source of rents. IBM's patent royalties now approach $2 billion per year, nearly a quarter of income before taxes. Moreover, IBM's large patent portfolio allows it to access the fruits of other firms' R&D. According to IBM's Roger Smith, assistant general counsel, Intellectual Property, "The IBM patent portfolio gains us the freedom to do what we need to do through crosslicensing —it gives us access to the inventions of others that are key to rapid innovation" (1990). Thus, incumbent firms in industries facing declining R&D opportunities may substitute patents

[29] As above, software patents are likely associated with patterns of behavior affecting other patents as well, such as strategic portfolio behavior. In this section we attribute the substitution effect to "software patents" as a shorthand for the broader behavior associated with software patenting.

for R&D through an offensive patenting strategy. This is consistent with a theoretical model of patent thickets (Bessen, 2003).

Of course, this explanation raises the question of why IBM and other firms did not pursue an aggressive strategy earlier. Changing patenting standards provide an answer: lower standards in the 90s made patent thicket strategies profitable then but not earlier. In other words, patenting standards were a likely contributing factor, although changing technological opportunity may have also played a role.

Note that in this case, even though changes in patenting do not entirely cause the changes in R&D intensity, the effect of these changes is not salutary for R&D. IBM's actions necessarily decrease the R&D incentives for other firms that are forced to cross-license their innovations to IBM and/or pay royalties on them.

Indeed, the second explanation concerns the impediments that strategic use of patent portfolios—patent "thickets"—pose to innovators (see Gallini, 2002, for a review). Changes in patenting standards are seen as having made it easier to obtain patents and these patents have become easier to enforce. On the one hand, this appears to have substantially increased "defensive" patenting (Hall and Ziedonis, 2001). On the other hand, despite the greater likelihood that any patent will be upheld, there is greater uncertainty that any innovation may be subject to legal action by other firms. This is because relaxed patenting standards increase the attractiveness of aggressive patenting strategies (Bessen, 2003). And this greater risk raises the required rate of return on R&D investments. In the words of Stephen P. Fox, associate general counsel and director of Hewlett-Packard (2002), legal changes produced "pervasive uncertainty about legal rights, both in terms of ability to enforce one's own patents and ability to avoid rapidly escalating exposures to infringement claims by others. And that uncertainty heightens risks surrounding innovation investment decisions." According to Cecil D. Quillen, Jr., former general counsel at Kodak, "If the uncertainties are such that you cannot be confident that your products are free and clear of others' patents you will not commercialize them, or a higher return will be demanded if you do to compensate for the additional risk. And this probably means you will not do the R&D that might lead to low return (or no return) products" (2003). Thus changes in patenting standards may lead to both more defensive patents and to less R&D.

Third, more patents and less R&D is consistent with models of sequential innovation (O'Donoghue, 1998, Hunt, 2002). Lower standards for non-obviousness make it easier to obtain patents, so firms obtain more patents. But this also shortens the interval before a sequential innovation will become obsolete, so rents are less and R&D is less.

Finally, substitution is also consistent with the Cohen and Levinthal model (1989) of "absorptive R&D." If firms must invest substantial R&D in order to take advantage of R&D spillovers, then more patents may reduce spillovers and thus reduce R&D.

Thus, with all of these explanations, lower patenting standards and the associated patent flood are not beneficial to R&D and they may well be detrimental.

IV. Conclusions

Our results suggest that the rise in software patenting over the last 20 years, and particularly over the last decade, was driven substantially by regulatory changes—the gradual erosion of the subject matter exception for computer programs and other changes in patent law introduced in the 1980s. Lower standards for obtaining patents substantially reduced the cost of patenting inventions that use software and appear to have encouraged the strategic acquisition of large patent portfolios. At the beginning of our sample, during the early 80s, software patents were more costly relative to other patents; by the end of the 90s they were relatively "cheap." Our analysis firmly rejects the "one-size-fits-all" hypothesis that software patents are treated just like other patents.

Moreover, the rise in software patenting contributed significantly to the overall increase in patent propensity among U.S. firms. Cheaper patents enabled firms to accumulate large portfolios of patents at lower cost and this may have encouraged firms to pursue aggressive patent portfolio strategies (Bessen, 2003). This interpretation is consistent with the fact that larger U.S. firms in manufacturing industries (plus IBM) acquired a disproportionate share of software patents in the 1980s and 1990s. Many of these industries, including semiconductors and computers, have well-known patterns of strategic behavior using patent thickets (Hall and Ziedonis, 2001, Grindley and Teece, 1997). These industries also appear to exhibit increases in patent propensity, while industries that have not engaged in such strategic behavior, such as chemicals and pharmaceuticals, do not exhibit similar increases.

Most significant, we find that the increase in software patenting is associated with a *decrease* in R&D intensity; software patents substituted for R&D. Moreover, the magnitude of this substitution is substantial—about 10-15% of R&D. Our analysis rejects simple arguments that more patents encourage more R&D or that lower patent standards raise R&D incentives. Although we do *not* argue that regulatory changes caused the *entire* decrease in R&D, all of the explanations we explore do suggest that these changes had at least some negative effect on R&D spending.

Appendix

A. Derived Demand for Patents

Studies of patent values find that patents have highly skewed distributions, often modeled with either a Pareto or a lognormal distribution (Scherer and Harhoff, 2000). Since the marginal value of the *n*th patent is $A'(n)$, we may think of *A* as a cumulative distribution function. Here, we use a Pareto distribution to generate a demand equation with a flexible functional form. Let

(A1) $A(n) = 1 - n^{-g}, \qquad 1 \le n, \;\; 0 < g \;.$

We could, alternatively, use a Pareto distribution of n+1, but since we are mainly concerned with the region where n>>1, this complication can be ignored in our approximation. Then, solving the first order condition (2),

(A2)
$$\begin{aligned} \ln \hat{n} &= \frac{1}{1+g}\left[\ln g + \ln(mQ) - \ln c\right] = \frac{1}{1+g}\left[\ln g - \ln(1-n^{-g}) + \ln V - \ln c\right] \\ &\approx a + b \ln V - b \ln c, \qquad b = \frac{1}{1+g} \end{aligned}$$

The approximation to a constant will be valid in the range where n is sufficiently large. There may be some distortion at small values of n, but since our regressions place little weight on these, this approximation is suitable and it provides us a flexible functional form that reasonably approximates other skewed distributions of *A*.

B. Patent Demand with Qualitatively Different Software Patents

Now consider the situation with two different types of patents, software patents and other patents. Let the costs of obtaining patents be c_s and c_o for software patents and other patents, respectively. Let the numbers of each type of patent be n_s and n_o, respectively, so that $n = n_s + n_o$. Let *s* be the software share of patents. Then

$$\text{(A3)} \qquad c = (1-s)\,c_o + s\,c_s \qquad s = \frac{n_s}{n}.$$

Also, let $A = A(n,\ s;\ l)$, where λ is a parameter that differs across firms and over time. That is, *A* is a family of functions. Differences in this parameter produce different marginal appropriability for software patents so that firms have different optimal values of *n* and *s*.

The first order conditions are now

$$\text{(A4)} \qquad \left.\frac{\partial A}{\partial n_o}\right|_{n_s} = \frac{\partial A}{\partial n} - \frac{\partial A}{\partial s}\frac{s}{n} = \frac{c_o}{mQ}, \qquad \left.\frac{\partial A}{\partial n_s}\right|_{n_o} = \frac{\partial A}{\partial n} + \frac{\partial A}{\partial s}\frac{(1-s)}{n} = \frac{c_s}{mQ}$$

The corresponding optimal values can be written $\hat{s}(l)$ and $\hat{n}\left(\hat{s}(l)\right)$. We assume that the appropriability function is well-behaved so that the optimum is an interior solution in both variables. Note that these first order conditions imply that the marginal appropriability of each type of patent is proportional to its marginal cost. Firms compensate for any differences in the relative marginal appropriability arising from different values of λ by adjusting the relative number of patents they acquire of each type. E.g., if stronger enforcement increased the marginal appropriability of software patents, then firms would acquire more software patents until the first order condition was met.

With a little manipulation, these first order conditions can be re-written,

$$\text{(A5)} \quad \text{(a)} \quad \frac{\partial A}{\partial n} = \frac{c_o - (c_o - c_s)\,s}{mQ} = \frac{c}{mQ}, \qquad \text{(b)} \quad \frac{\partial A}{\partial s} = -\frac{(c_o - c_s)\,n}{mQ}.$$

Let us now assume that in the vicinity of $s \approx 0$, that $A(n,s;\ l) \approx A(n,0;\ l) = 1 - n^{-g}$, as above. This means that, designating partial derivatives with subscripts, $A_{ns}(n,0) \approx 0$, and the constraint imposed in (A5a) allows us to evaluate the implicit derivative, (A6) $$\frac{d\,\hat{n}(0)}{d\,\hat{s}} = -\frac{\partial\,(A_n - c/mQ)}{\partial\,s}\Big/ A_{nn} \;=\; \frac{(c_o - c_s)\,n}{(1+g)\,c}\,.$$

Then we can make a Taylor approximation,

(A7) $$\ln\hat{n}(\hat{s}) \;=\; \ln\hat{n}(0) \;+\; \frac{d\;\hat{n}(0)}{d\,\hat{s}}\cdot\frac{\hat{s}}{n} \;=\; \ln\hat{n}(0) \;+\; \frac{(c_o - c_s)}{(1+g)\,c}\cdot\hat{s}\,,$$

evaluating the implicit derivative using (A5a), , where subscripts designate partial derivatives. Then solving for $\ln\hat{n}(0)$ as above, we have

(A8) $$\ln\hat{n} \;\approx\; a \;+ b\ln V - b\ln c_o \;+\; b\,\frac{(c_o - c_s)}{c}\,\hat{s} \;=\; m_t \;+ b\ln V \;+\; b\,\frac{(c_o - c_s)}{c}\,\hat{s}$$

where changes in c_o have been absorbed into the time dummies, m_t.

Note that if software patents cost less than other patents, the coefficient of *s* is positive and greater *s* is associated with more total patents. If software patents are more costly, then the coefficient is negative and greater *s* is associated with smaller *n*.

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.”

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

Bessen, James. 2003. “Patent Thickets: Strategic Patenting of Complex Technologies,” ROI Working Paper.

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.

Burk, Dan L. 2002. Presentation to Federal Trade Commission, “Competition and Intellectual Property Law and Policy in the Knowledge-Based Economy,” July 10, 2002, pp. 155-6.

Burk, Dan L. and Mark A. Lemley. 2003, forthcoming. "Is Patent Law Technology Specific?” Berkeley Technology Law Journal, v. 17.

Cohen, Julie and Mark Lemley. 2001. “ Patent Scope and Innovation in the Software Industry,” *California Law Review*, v. 89, pp. 1-57.

Cohen, Wesley M., Akira Goto, Akiya Nagata, Richard R. Nelson, John P. Walsh. 2002. “R&D Spillovers, Patents And The Incentives to Innovate in Japan and the United States”, *Research Policy*, v. 31, pp. 1349-67.

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.

Flynn, John D. 2001. “Comments on the International Effort to Harmonize the Substantive Requirements of Patent Laws” IBM submission at http://www.uspto.gov/web/offices/dcom/olia/harmonization/TAB42.pdf

Fox, Stephen P. 2002. Presentation to Federal Trade Commission, “Competition and Intellectual Property Law and Policy in the Knowledge-Based Economy,” February 28, 2002.

Gallini, Nancy T. 2002. “The Economics of Patents: Lessons from Recent U.S. Patent Reform,” *Journal of Economic Perspectives*, v. 16, pp. 131-54.

Greene, William H. 1997. *Econometric Analysis*, 3rd edition. Upper Saddle River, N.J.: Prentice Hall.

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

Griliches, Zvi, Bronwyn H. Hall, and Jerry Hausman…

Grindley, P. and Teece, D. 1997. “Managing Intellectual Capital: Licensing and Cross-Licensing in Semiconductors and Electronics,” *California Management Review*, v. 39, pp. 8-41.

Hall, Bronwyn H. 1990. “The Impact of Corporate Restructuring on Industrial Research and Development,” *Brookings Papers on Economic Activity, Microeconomic*, v. 1990, pp. 85-124.

Hall, Bronwyn H. 2003. “Exploring the Patent Explosion,” invited lecture at the ZEW Workshop on Empirical Economics of Innovation and Patenting, Mannheim, Germany, March 14/15, 2003.

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.; Jaffe, Adam; and Trajtenberg, Manuel, "Market Value and Patent Citations: A First Look" (August 2, 2001). *Economics Department, University of California, Berkeley,* Working Paper E01-304.

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, pp. 101-128.

Harhoff, Dietmar, Francis Narin, F.M. Scherer and Katrin Vopel. 1999. “Citation Frequency and the Value of Patented Inventions,” *Review of Economics and Statistics*, v. 81, pp. 511-15.

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, pp. 909-38.

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.

Hunt, Robert M. 2002. "Patentability, Industry Structure, and Innovation," Working Paper 01-13R, Federal Reserve Bank of Philadelphia.

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, 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,* v. XLIV, pp. 573- 603.

Lanjouw, Jean and Mark Schankerman. 1999. “The Quality of Ideas: Measuring Innovation with Multiple Indicators,” NBER Working Paper No. 7345.

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,” *Brookings Papers on Economic Activity*, v. 3, pp. 783-820.

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

Macleod, Christine, Jennifer Tann, James Andrew, and Jeremy Stein. 2002. “The Price of Invention: counting the cost of patents in Victorian Britain, a case study from steam engineering,” working paper.

Mansfield, Edwin. 1986. “Patents and Innovation: An Empirical Study,” *Management Science*, v. 32, pp. 173-181.

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, pp. 577-615.

Mohnen, Pierre A., M. Ishaq Nadiri, and Ingmar R. Prucha. 1986. “R&D, Production Structure and Rates of Return in the U.S., Japanese and German Manufacturing Sectors: a Non-separable Dynamic Factor Demand Model,” *European Economic Review*, v. 30, pp. 749-771.

National Science Foundation. 2003. “Total (Company, Federal, and Other) Funds for Industrial R&D Performance, by Industry and by Size of Company: 1953-98,” Industrial Research and Development Information System.

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

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

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.

Quillen, Cecil D. 2003. E-mail 3/28/2003.

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, pp. 199-227.

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*, v. 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, pp. 101-21.

Switzer, Lorne. 1984. “The Determinants of Industrial R&D: A Funds Flow Simultaneous Equations Approach,” *Review of Economics and Statistics*, v. 66, pp. 163-8.

Trajtenberg, Manuel. 1990. “A Penny for Your Quotes: Patent Citations and the Value of Innovations,” *RAND Journal of Economics*, v. 21, pp. 172-87.

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

USPTO 1996. "Examination Guidelines for Computer Related Inventions," *Federal Register*, v. 61, pp. 7478-92.

Yoches, E. Robert. "The Compton's Reexamination – A Sign of the Times," *The Computer Lawyer*, v. 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
Number of claims	16.8	12.6	13.9
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 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	Programmer Employment	All patents	Patents/R&D
Manufacturing	69%	10%	85%	
Machinery (SIC 35)	27%	3%	17%	2.5
Electronics (SIC 36)	22%	2%	22%	2.8
Other	20%	5%	45%	1.8
Non-manufacturing	31%	90%	15%	
Software publishers (SIC 7372)	6%	42%	1%	0.7
Other software services (exc. IBM)	2%		1%	4.4
Other non-manufacturing	3%	48%	3%	2.8
Addendum: IBM	8%	--	3%	4.7

Note: covers 22,954 software patents and 109,509 total patents for firms the sample matched to Compustat. Programmer employment is the share of total employment accounted for by computer programmers and systems analysts in 1998 (from the Occupational Employment Survey of the BLS). Last column shows patents granted per $10 million of R&D in 96 dollars. The last row shows IBM's patents as a portion of all patents.

Table 5. Poisson Regressions

Dependent variable: Number of patents granted

	1		2		3		4		5	
Industries (SIC)	3674		All ex 3674		All ex 3674		28		All ex 3674	
	Hall-Ziedonis Poisson		Poisson		Poisson		Poisson		Poisson	
Ln(R&D / emp.)	.19	(.08)	.53	(.00)	.33	(.01)	.15	(.02)	.33	(.01)
No reported R&D	-1.69	(.83)	-1.77	(.01)	.08	(.03)	.47	(.12)	.08	(.03)
Ln(Employees)	.85	(.03)	.78	(.00)	.66	(.01)	.66	(.01)	.68	(.01)
Ln(P&E / emp.)	.60	(.11)	.17	(.00)	-.04	(.01)	.17	(.03)	.03	(.01)
d_{80-85}									-.68	(.06)
d_{86-90}									-.45	(.05)
d_{91-95}									.60	(.03)
d_{96-99}									1.90	(.02)
Year dummies	1979-95		1980-99		1980-99		1980-99		1980-99	
Firm fixed effects	Partial		--		Full		Full		Full	
N	946		21,782		20,972		2,352		20,972	

Note: Standard errors are in parentheses. The δ's are coefficients on terms where the independent variable is the software share of patents interacted with a time period indicator dummy. R&D and P&E are deflated. In column 1, observations are assigned to the year in which the patent application was filed; in the remaining columns, observations are assigned to the year of the patent grant. Columns 3-5 use the fixed effects estimation described in Hausman, Hall and Griliches (1984).

Table 6. Appropriability Regressions (1981-99)

Dependent variable: Ln(number of patents granted)

	1		2		3		4		5-yr Differences 5	
d_{81-85}	-.54	(.30)	-.24	(.26)	-.73	(.30)	-.71*	(.26)		
d_{86-90}	-.31	(.21)	-.44	(.18)	-.12	(.22)	-.63*	(.19)	.30	(.29)
d_{91-95}	.86*	(.16)	.76*	(.14)	1.10*	(.15)	.51*	(.14)	1.26*	(.21)
d_{96-99}					.90*	(.11)	1.11*	(.09)	.70*	(.17)
Ln(cash flow)	.70*	(.01)	.11*	(.02)						
Ln(market value)					.77*	(.01)	.24*	(.02)	.81*	(.03)
Ln(lagged R&D)			.53*	(.02)			.45*	(.02)		
Capital/employee			.58*	(.06)			.24*	(.05)		
Pct. domestic patents			-1.10*	(.07)			-.81*	(.07)		
Citations rec'd.			-.02*	(.00)			-.02*	(.00)		
Self-citations			2.66*	(.10)			2.56*	(.09)		
$d_{91-95} - d_{86-90}$	1.17*		1.19*		1.22*		1.14*		1.26*	
$d_{96-99} - d_{86-90}$					1.02*		1.73*		1.96*	
N	6,635		5,697		9,844		8,435		4,965	
Adjusted R-sq.	.669		.794		.655		.793			

Note: Standard errors are in parentheses and asterisk indicates significance at the 1% level. All regressions include year dummies and, except for column 5, industry dummies. The first four columns are weighted by number of patents granted; column 5 is weighted by $1/\left(1/n_{it}+1/n_{i,t-5}\right)$ where n is the number of patents granted. 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 (or difference in column 5). Market value is deflated millions. Cash flow is mean deflated cash flow (millions) for following four years. Log cash flow 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. Pct. domestic patents 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).

Table 7. R&D Cost Function Estimation, 1985-99

Dependent Variable (Col. 1-3): $\Delta \frac{R\&D}{Sales}$ (Col.4): $\Delta \frac{R\&D}{Cost}$

	1		2		3		4	
Δs x (1985-89)			0.026*	(0.009)	0.024*	(0.009)	0.026	(0.071)
Δs x (1990-94)			-0.018	(0.007)	-0.016	(0.007)	-0.019	(0.058)
Δs x (1995-99)			-0.072*	(0.006)	-0.072*	(0.006)	-0.138*	(0.047)
Lag log sales					0.000	(0.000)		
Lag new firm					-0.002	(0.003)		
Stock std. dev.					-0.013*	(0.004)		
Debt change					-0.002*	(0.001)		
$\Delta \ln p$ Capital	0.000	(0.001)	0.000	(0.001)	0.000	(0.001)	-0.001	(0.005)
$\Delta \ln p$ Labor	0.048*	(0.006)	0.052*	(0.006)	0.057*	(0.006)	0.019	(0.050)
$\Delta \ln p$ Energy	-0.008	(0.004)	-0.010	(0.004)	-0.006	(0.004)	-0.029	(0.033)
$\Delta \ln p$ Materials	-0.016*	(0.002)	-0.015*	(0.002)	-0.017*	(0.002)	0.019	(0.020)
$\Delta \ln p$ Services	-0.011*	(0.004)	-0.019*	(0.004)	-0.014*	(0.005)	0.001	(0.035)
$\Delta \ln p$ IT	0.007	(0.004)	0.002	(0.004)	-0.004	(0.005)	0.022	(0.035)
No. observations	5,613		5,613		5,467		5,612	
Adj. R squared	0.064		0.092		0.095		0.000	

Note: Standard errors in parentheses. All regressions include year dummies. Asterisk indicates significance at the 1% level. Time differences are five years. Sample includes firms matched to patent file with positive patents. All regressions are weighted by $1/\left(1/n_{it} + 1/n_{i,t-5}\right)$ where n is the number of patents granted. s is software share of patents granted, the p are factor prices for the 2-digit industry from the BLS, sales are deflated, “new firm” is a dummy for firms that entered the Compustat database within 5 years, “stock std. dev.” is the standard deviation of annual differences in year-end log stock prices, and “debt change” is the change in long term debt divided by the capital stock. Note that the coefficients of Δs in Column 5 have different meanings. For example, the coefficient for the period 1991-95 estimates $d_{91-95} - d_{86-90}$. Observations are excluded where R&D > ½ sales.

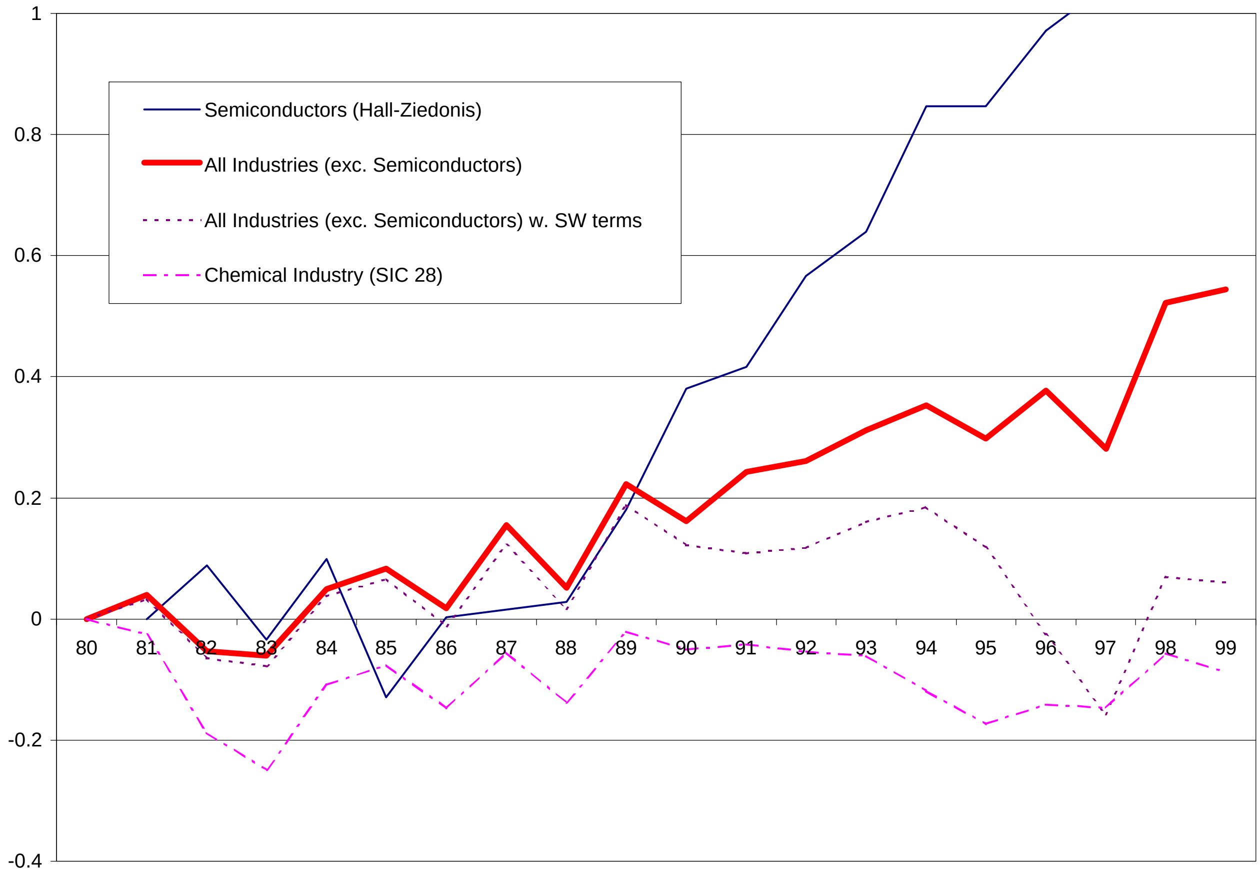
Table 8. R&D Cost Function Estimation, 1991-99

Dependent Variable: $\Delta \frac{R\&D}{Sales}$

	1		2		3		4	
Δs	-0.050*	(0.005)	-0.048*	(0.010)	-.042*	(.005)	-0.033*	(0.009)
Δs x Large							-0.025	(0.011)
Δs x SW related			-0.003	(0.011)				
Δ Programmers/Emp.					0.059*	(.010)		
$\Delta \ln p$ Capital	-0.001	(0.001)	-0.001	(0.001)	-.002	(.001)	-0.001	(0.001)
$\Delta \ln p$ Labor	0.057*	(0.011)	0.057*	(0.011)	0.047*	(.011)	0.057*	(0.011)
$\Delta \ln p$ Energy	-0.033*	(0.008)	-0.033*	(0.008)	-.031*	(.007)	-0.033*	(0.008)
$\Delta \ln p$ Materials	-0.030*	(0.005)	-0.030*	(0.005)	-.021*	(.005)	-0.029*	(0.005)
$\Delta \ln p$ Services	-0.001	(0.007)	-0.001	(0.007)	-.022*	(.008)	-0.003	(0.007)
$\Delta \ln p$ IT	0.022*	(0.006)	0.021*	(0.006)	0.015	(.006)	0.021*	(0.006)
No. observations	3,412		3,412		3,396		3,412	
Adj. R squared	0.091		0.091		0.101		0.093	

Note: Standard errors in parentheses. All regressions include year dummies. Asterisk indicates significance at the 1% level. Time differences are five years, except for Δprogrammers/total employment, which is the change in the ratio of employment of systems analysts and programmers to total employment for the 3-digit industry from 1987-89 to 1998 (depending on industry). Sample includes firms matched to patent file with positive patents. All regressions are weighted by $1/\left(1/n_{it} + 1/n_{i,t-5}\right)$ where n is the number of patents granted. s is software share of patents granted, the p are factor prices for the 2-digit industry from the BLS, "Large" signifies firms with more than 5,000 employees, and "SW related" signifies firms in SIC 35, 36, 38, and 73. Excludes observations where R&D > ½ sales.

Figure 1. Patent Propensities



Note: Data are time dummies from Poisson regressions (see Appendix) normalized to zero in 1980. Semiconductor data is from the preferred regression of Hall and Ziedonis (2001). Their regression is based on patent application dates; the other regressions are based on patent grant dates. To compensate, the Hall-Ziedonis data are lagged two years.