NBER WORKING PAPER SERIES

THE PRIVATE VALUE OF SOFTWARE PATENTS

Bronwyn H. Hall
Megan MacGarvie

Working Paper 12195
http://www.nber.org/papers/w12195

NATIONAL BUREAU OF ECONOMIC RESEARCH
1050 Massachusetts Avenue
Cambridge, MA 02138
April 2006

Hall: 549 Evans Hall, Berkeley, CA 94720, bhhall@econ.berkeley.edu. MacGarvie: 595 Commonwealth Ave., Boston, MA 02215, mmacgarv@bu.edu. This is a revision of a paper written in April 2005. We thank Anne Layne-Farrar and LECG for the use of the Corptech data, and we thank Jim Bessen for supplying us with the list of software patents used in Bessen and Hunt (2003). We are also grateful to Stu Graham, Bala Iyer, Lucia Silva, numerous participants in the Empirical Patent Research Conference, St. Helena, California, February 2005, the IIOC, Atlanta, April 2005, the ZEW Conference on Patents and Innovation in Mannheim, September 2005 and seminars at UCSD, University of Kansas, and LMUMuenchen for extremely helpful comments. The views expressed herein are those of the author(s) and do not necessarily reflect the views of the National Bureau of Economic Research.

The Private Value of Software Patents
Bronwyn H. Hall and Megan MacGarvie
NBER Working Paper No. 12195
April 2006
JEL No. O34, L63, L86

ABSTRACT

We investigate the value creation or destruction associated with the introduction of software patents in the United States in two ways. The first looks at the cumulative abnormal returns to ICT firms around the time of important court decisions impacting software patents, and the second analyzes the relationship between firms' stock market value, the sector in which they operate, and their holdings of software patents cross-sectionally. We find that the extension of patentability to software was initially negative for software firms, especially for those producing application software or services. We also find that software patents are positively and significantly associated with Tobin's Q, and that the market's valuation of software patents increased following changes in the USPTO's treatment of software patents in 1995.

Bronwyn H. Hall
Department of Economics
549 Evans Hall, #3880
University of California-Berkeley
Berkeley, CA 94720-3880
and NBER
bhhall@econ.berkeley.edu

Megan MacGarvie
Boston University
School of Management
595 Commonwealth Avenue, Room 522H
Boston, MA 02215
and NBER
mmacgarv@bu.edu

The Private Value of Software Patents

Bronwyn H. Hall and Megan MacGarvie[2]

March 2006

1. Introduction

Ever since software became generally patentable in the United States in 1995, the wisdom of such a change has been widely debated. Proponents of the change argue that there is no statutory reason to exclude software (or computer-implemented) inventions from patentability, and also that patenting software has social benefits from disclosure and from the fact that it enables software components to be reused by others (Cohen and Lemley 2001). Critics, who are numerous, base their arguments on a series of considerations: that such patents have often been of low quality, that they discourage rather than encourage innovation, and that they have a negative effect on the growing open source model of innovation (Barton 2000, 2001, Kasdan 1994, Bakels and Hugenholtz 2002, Lunney 2001, Quillen 2001, Dreyfuss 2001, Meurer 2002). In a recent article in PC magazine, the columnist John Dvorak argued that software patenting is even bad for Microsoft, the largest patent holder among the pure software firms (Dvorak 2005).

The economic view of the patent system sees this debate as inevitable, given the nature of patents: with a patent grant we trade off short term exclusive (monopoly) rights

[2] UC Berkeley and NBER; Boston University and NBER.

to the use of an invention in return for two things: 1) an incentive to create the innovation; and 2) early publication of information about the innovation and its enablement. The argument is that without the patent system, fewer innovations would be produced, and those that were produced would be kept secret as much as possible to protect their returns from appropriation by others. In considering the economic impacts of a subject matter extension to software and business methods, the tradeoff between these benefits and the welfare cost of the grant of a monopoly right are at least as important as they are in any other technological arena.

Recent analysis also says that although ***competition*** may suffer when we grant a monopoly right to an inventor, it will benefit if this right facilitates entry into the industry by new and innovative firms (Hall 2004). Second, ***innovation*** will benefit from the incentive created by a patent but may suffer if patents discourage the combining and recombining of inventions to make new products and processes (Scotchmer 1996; Heller and Eisenberg 1998). There are several reasons to think that facilitating entry (the benefit of patents for competition) and discouraging recombination of elements (the cost of patents for innovation) may be particularly salient factors when considering the effects of patentability on the software industry.

Evaluating the tradeoff between the benefits and costs of the introduction of software patents is a formidable, perhaps impossible, task. It is not made easier by the fact that the most important expansion of software patentability occurred in 1994-1995, exactly coincident with the internet revolution and the beginnings of its impact on the

global economy's use of software for a wide range of new applications.[3] To evaluate the impact of software patentability properly would require an examination of a counterfactual world without software patents and measurement of the differential impact on welfare and innovative activity in existing firms, new entrants, downstream buyers, and consumers. In this paper we look at one ingredient of the overall problem: the private returns to established computer hardware and software firms from the expansion of software patentability. If these are not significantly positive, it is difficult to see how software patents could be beneficial for the economy on net. On the other hand, a finding that existing firms benefit financially from the introduction of software patents does not imply that the overall effect is benign.

Existing research on software patents reveals a dramatic increase in the propensity to patent software over the last two decades, and argues that this indicates that holding patents on software has become both easier and more valuable to firms (Bessen and Hunt 2003). In this paper we explore the relationship between firm value and the patenting of software in two different ways. First we conduct a series of event studies that look at the immediate market impact of changes in the patentability of software on firms in sectors where software patenting is prevalent.[4] Second, we use the methodology of Hall, Jaffe,

[3] It is possible that this timing is not a coincidence. The expansion of patentability has often come about as a result of the expansion of economic activity in a particular area. E.g., see Murmann (2003) on the chemical industry in Germany and Britain in the late 19th century. However the rapidity with which attempts to patent software followed on the introduction of the internet makes the causal explanation improbable. It seems more likely that the changes in software patenting in the mid-nineties were driven by the introduction of the personal computer in the mid-eighties.

[4] Based on a review of the industry sectors where firms obtain software patents described later in the paper, we identified the relevant two-digit industries as 35 (machinery including computing equipment), 36 (electrical machinery, 38 (instruments), 48 (telecommunications), and 73 (business services including software).

and Trajtenberg (2005) to estimate the market value of software patents using various patent-related measures and we explore the effect of changes in the patentability of software over time on the market value of publicly traded firms that hold such patents.[5]

In performing these analyses, we separate the software firms in the sample in those that are "upstream" and those that are "downstream," which corresponds to the position of their products in the computing stack, from the basic operating system layer to end user application software products (Raduchel 2006). We hypothesize that firms producing software that relies on hardware or software that is upstream from their software in order to operate may face more negative consequences than other firms from the introduction of software patents (at least initially), because they are more likely to need licenses for patented software technology in order to ensure that their products will operate successfully.

The paper is organized in the following way: the next section reviews the recent history of software patenting in the United States. This is followed by a discussion of our data, which consists of a panel of publicly traded firms in the ICT sector; we also spend some time evaluating a number of definitions of "software" patents that are based on keyword searches and patent classification systems. The next two sections present our empirical results: first a series of studies of the immediate stock market impact of software patentability decisions in the courts, and then the results for the relationship between firm market value and the ownership of patents. The latter results are presented

[5] In future, we may also be able to perform event studies on the grant of individual software patents (in the manner of Austin (1993)), which would allow us to estimate the value of software patents controlling for individual patent characteristics.

for two subperiods, before and after the general acceptance of software patenting by the USPTO guidelines. The final section concludes.

2. Background

The law concerning the patentability of software in the United States has evolved through a series of decisions following the passage of the 1952 Patent Act (which did not exclude any subject matter statutorily), to the point where algorithms may be patented if "there is practical application for the algorithm or if it is associated with a tangible medium."[6] Although a complete history of the use of intellectual property protection in the software industry is beyond the scope of this paper, we provide a brief summary here and refer the reader to Graham and Mowery (2003) for more detail.

In 1972, the Supreme Court's ruling in *Gottschalk v. Benson* held that because software is essentially a collection of algorithms, it could not be patented. In 1980, Congress confirmed that the appropriate intellectual property protection for software was copyright (Samuelson 1984). However, in 1981 the court allowed for patenting of software tied to physical or mechanical processes, such as the program implemented in the method for curing rubber at issue in *Diamond v. Diehr,* a decision which seems consistent with the present day European Patent Convention.[7] However, this and

[6] Sterne and Bugaisky, p. 221

[7] The European Patent Convention's (EPC) treatment of patentable subject matter differs markedly from that of the United States Patent Act. Whereas in the US, the 1952 Patent Act opened the door to judicial expansions of patentable subject matter by not including any explicit limits on statutory subject matter, Article 52 of the EPC expressly excludes several categories of inventions, among them mathematical methods, schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computers "to the extent to which a European patent application or European patent relates to such subject-matter or activities *as such*" (EPO 1989, emphasis added). In the case of software, the phrase "as such" has led to a number of software inventions actually being patented via their

subsequent decisions by the court during the next 15 years led the US Patent and Trademark Office (USPTO) to modify its position with respect to software patents, ultimately to one that allowed them even when not embodied in a physical process. As Figures 1 and 2 show, the growth of software patenting increased from a 5-10 per cent level prior to 1981 to a 15 per cent level afterwards, and then increased again in the mid-1990s, as the internet boom took off and the USPTO issued new guidelines for software patenting.

There matters rested until the early 1990s when the rise of the personal computer industry and the consequent vertical disintegration of the computing sector encouraged software and information producers to test the subject matter exclusions with respect to software again. In 1993 Compton Encyclopedia attempted to enforce a broad patent on the display of multi-media content on CD-ROMs against several firms; in response to complaints by these firms, the USPTO re-examined and revoked the patent in March 1994, which halted assertion of this particular software patent.

But then an important change occurred later in 1994 when the Court of Appeals of the Federal Circuit (CAFC) stated (*In re Alappat*) that unpatentable software was that which represented "a disembodied mathematical concept…which in essence represents nothing more than a 'law of nature,' 'natural phenomenon,' or 'abstract idea.'" Software that could be patented was "rather a specific machine to produce a useful, concrete, and tangible result."[8] A series of additional decisions, described in more detail in section 4 of

---

embodiment in hardware. See Bakels (2005) for a detailed discussion of the evolution of European policy in this area.

[8] In re *Alappat*, 33 F.3d 1526, 1544 (Fed. Cir. 1994), cited in Sterne and Bugaisky, p. 222.

this paper, culminated in 1995 with *In re Beauregard,* in which the CAFC ruled that the Patent Office's rejection of a software patent application by IBM was erroneous The Commissioner of Patents then issued new guidelines on the patenting of software, which allowed inventors to patent any software embodied in physical media (which essentially reversed the *Gottschalk* decision).[9] These decisions presumably lie behind the huge increases in software patenting during the 1996-1999 period shown in Figures 1 and 2.

A second important expansion of patentability took place in 1998 when Judge Rich issued the famous opinion in State Street Bank and Trust v Signature Financial Corporation.[10] The Signature patent at issue was a "pure" number-crunching type of application, which implemented financial accounting functions. The Federal Circuit Court decision stated clearly that Section 101 (the section of the patent code that deals with subject matter for patentability) is unambiguous - "any" means ALL, and it is improper to read limitations into 101 not intended by Congress. Therefore, mathematical algorithms are non-statutory only when "disembodied" and thus lacking a useful application. The court went on to make sure that the decision was precedent-setting by stating that with regard to the business method exception, "We take this opportunity to lay this ill-conceived exception to rest."

This decision was followed by an increase in applications for "business method" patents, most of which are arguably also software patents, because they describe the implementation of a business method on a computer or the internet. However, they are

[9] Sterne and Bugaisky, p. 223.

[10] State Street Bank and Trust Co., Inc. v. Signature Financial Group, Inc., 149 F.3d 1368 (Fed. Cir. 1998).

still a small fraction of all software patents (Hall 2003), at least using the relevant patent class to define them. Most of the analysis in this paper uses data that ends in 1999 or earlier, so these patents will play a relatively small role.

2.1 The critique of software patentability

The "quality" of a patent is a somewhat ill-defined catchall term for all the characteristics that the particular analyst would like patents to have. Hall (2003) and Hall et al (2003) review these criteria and argue why the failure of issued patents to satisfy them may be costly for firms and society. These criteria include satisfying the statutory definition of a patentable invention (novelty, non-obviousness, and utility),[11] sufficient disclosure to enable those "skilled in the art" to understand the invention, and relatively little uncertainty over the validity of the patent. A number of legal and technical scholars have critiqued the software patents issued after the series of court decisions in 1994-1995 on all of these grounds.

First, because software development took place over a long period before software was patentable, when the USPTO began to issue patents in this area, they did not have examiners with the relevant training and lacked adequate data bases with software prior art. For example, on February 7, 1995, the following exchange between a patent examiner and the editor concerning access to non-patent prior art took place in Aharonian's Patent Newsletter:[12]

[11] See Lunney (2001) for an argument that the non-obviousness test has been weakened since the creation of the Federal Circuit Court of Appeals in 1982.

[12] Aharonian (1995). http:/www.bustpatents.com/

> The examiner: "The problem with obviousness is evidence. When an examiner uses common sense, attorneys scream hindsight. Also, a problem with ordering non-patent publications or translations of foreign documents is the time it takes to process these requests. An examiner cannot simply call a company whose making, or is believed to have made, a product which appears to infringe on a claim. At best, the examiner could ask a librarian at our library to call a company to request info, but again that takes time. With ten hours to do a case, movement is paramount."
>
> Aharonian: "Additionally for subjects like software, the cost of purchasing copies of technical papers would exceed the application fee, so I doubt many examiners would get the authority to spend such sums. Since for most software patent applications, the most relevant prior art is non-patent materials, between the statistics I cited on citing prior non-patent prior art (an average of two out of about 30) plus the above and below comments, one could make a good case that it is impossible for the PTO to conduct adequate novelty analyses."

The result of lack of access to adequate prior art was that many poor quality software patents were issued (Barton 2001; Kasdan 1994, 1999; Lunney 2001).

Second, this sector is an extreme case of cumulative innovation, where one person's invention depends on those of many other people and the transactions costs associated with licensing in a large number of patents for any particular software product may exceed the profits attainable from its invention and discourage innovation altogether.

Third, some claim that the disclosure function of patents is particularly badly served by software patents, which rarely include the source code implementation and are often quite vaguely and broadly worded. Even in the case of patents in general, both Cohen et al. (2002) (using US survey data) and Arundel et al. (2002) (using European survey data) report that firms rank a number of other means of acquiring information such as customers, exhibits, conferences, journals, suppliers, competitors, and nonprofit institutions ahead of patent disclosures as technical information sources. Furthermore, in contrast to other industries, it has been argued that patents on software do not generally serve to diffuse information about the patented technology. Mann (2004) states that:

> "It is clear that [disclosure] is an important benefit of patents in some industries, although the software industry in its current form is probably not one of them…As others have noted, with software cases the Federal Circuit has interpreted the disclosure requirement in such a way as to minimize the likelihood that disclosures will require information that is directly useful to competing inventors. Moreover, given the rapid pace of innovation, it will not often be the case that information disclosure in a patent application filed years earlier will be of immediate value to competitors in the industry." [13]

Finally, the growing open source movement has been extremely critical of such patents, because of the obvious fact that coexistence of open source with patented software is problematical. When independent invention is not a defense against infringement (as in the case of patents), a developer that issues a GPL or copyleft license to others on code that he has developed cannot be sure that he has the right to do so. Thus

[13] Mann (2004), p. 49-50

the presence of patents on some innovations implemented via software may foreclose many avenues of open source development, unless low cost methods of licensing are developed.[14]

3. Data

To perform our study, we combine data on publicly traded firms and their market value for the 1975-2002 period with a version of the NBER patent and citations dataset that has been updated to 2002.[15] The patent data are matched to Compustat data on firms in the following SIC categories: 35 (machinery including computing equipment), 36 (electrical machinery), 38 (instruments), 48 (telecommunications), and 73 (business services including software). This match was done using a version of Hall's Compustat-assignee match updated to include firms that were not included in the previous (1995) version.[16] Using this updated match, we are able to identify patenting entities associated with 1,290 of the 2,379 Compustat firms in the SIC classes listed above. Our dataset is an unbalanced panel, and Appendix A3 lists the number of firms and the number of software firms in the dataset in each year of the sample. Both have grown during the period, but software firms have clearly become more important as a share of the ICT sector.

Table 2 contains summary statistics for the dataset. In order to identify the particular software area in which our firms operate, we have also incorporated

[14] For more on the debate on software patents and open source, see Evans and Layne-Farrar (2004).

[15] The update is available at http://emlab.berkeley.edu/users/bhhall

[16] Details on the matching algorithm can be found at http://emlab.berkeley.edu/users/bhhall/pat/namematch/namematch.html

information from the Corptech directory of technology companies on the type of software they produce (systems software, middleware, applications, or software-related services).[17]

3.1 Defining a software patent

One difficulty that all researchers in this area encounter is that the definition of a software patent is rather unclear (Layne-Farrar 2005). Many scholars in the area may feel that "we know one when we see one" but this is not a practical way to choose a particular set of patents out of the 3 million or so in our datasets. Although all patents are classified into a number of technology classes and simply choosing those associated with software might seem the desirable way to go, it is an unfortunate truth that many of the relevant classes are broad enough to contain both software and hardware patents, and some software patents end up classified in classes that do not appear to have anything to do with software at first glance.[18] For this reason, different researchers have taken a number of approaches to identify software patents. In early versions of this paper, we explored the use of several different definitions and finally settled on a combination of them.[19]

The results reported in the body of the paper were obtained using a definition of a software patent that combined three ways of defining software patents. The three definitions are that used by Graham and Mowery (2003), that used by Bessen and Hunt

[17] We gather this information from Corptech's SOF product codes and the classification developed in Gao (2005), which is listed in the appendix to this paper.

[18] The four US patent classes with the largest number of patents assigned to our software firms are 382 (Image Analysis), 345 (Selective Visual Display Systems), 341 (Coded Data Generation or Conversion), and 700 (Data Processing: Generic Control Systems or Specific Applications), all of which can hold hardware as well as software applications. On the other hand, there are over 1000 such patents in 435 (Molecular Biology) and over 500 in 84 (Music).

[19] For comparison, results using each of the definitions are found in an appendix.

(2003), and one that we constructed based on the patent classes and subclasses that contain patents assigned to fifteen of the largest software firms (which we call Hall-MacGarvie). Our combined definition was the union of the set of patents in all relevant IPC and US patent classes (the union of Graham-Mowery and Hall-MacGarvie) intersected with the set of patents found using a keyword search of title and abstract (Bessen-Hunt).

As a check, we compare the results of these automated classification systems with the sample of software and internet business method patents identified manually by John Allison via a reading of the claims and description in the patents (Allison and Lemley 2000; Allison and Tiller 2003). In the remainder of this section of the paper, we describe the definitions more fully and present some comparative statistics based on them.

Graham and Mowery (henceforth GM) identify as software patents those that fall in certain International Patent Classification (IPC) class/subclass/groups. Broadly defined, the class/subclasses are "Electric Digital Data Processing" (G06F), "Recognition of Data; Presentation of Data; Record Carriers; Handling Record Carriers" (G06K), and "Electric Communication Technique" (H04L).[20] Graham and Mowery selected these classes after examining the patents of the six largest producers of software in the U.S. (based on 1995 revenues) between 1984 and 1995. Patents in these classes account for 57% of the patents assigned to the hundred largest firms in the software industry.[21]

[20] The groups included are G06F: 3,5,7,9,11,12,13,15; G06K: 9,15; H04L: 9.

[21] Graham and Mowery, p. 232. The firms are Microsoft, Adobe, Novell, Autodesk, Intuit, and Symantec.

Bessen and Hunt (henceforth BH) define software patents as those that include the word "software," or the words "computer" and "program," in the description and/or specification. Patents that meet these criteria and also contain the words "semiconductor," "chip," "circuit," "circuitry" or "bus" in the title are excluded, as they are believed to refer to the technology used to execute software rather than the software itself.[22] Patents containing "antigen", "antigenic", or "chromatography" in the description/specification are also excluded.

Our third algorithm for defining software patents (which we label HM) is similar to that used by Graham and Mowery. We identified all the U.S. patent class-subclass combinations in which fifteen software firms (Microsoft, Adobe, Novell, Autodesk, Symantec, Macromedia, Borland, Wall Data, Phoenix, Informix, Starfish, Oracle, Veritas, RSA Security, and Peoplesoft) patented and then categorized patents falling in these class-subclass combinations as "software."[23] We refer to this definition of software patents as the Hall-MacGarvie definition.

As a check of the accuracy of these various definitions, we used a sample of manually identified software and business method patents compiled by John Allison.[25] Using a sample of 1000 patents issued between 1996 and 1998, Allison and Lemley

[22] Bessen and Hunt, p. 4.

[23] One complication in using the U.S. patent classification system is that patents are continually reclassified as new classes are opened up. Software has been particularly subject to this reclassification, because there were no specific software classes until the "700" classes were created after the issuance of the USPTO guidelines in 1996. We are using the 2002 classification of all the patents in our dataset, which is late enough so that software should be correctly classified.

[25] We are grateful to James Bessen for making these patents available to us in machine-readable form, and for updating the data to 2002.

(2000) identified 100 software patents by reading the claims and descriptions. Allison and Tiller (2003) augmented this sample with 230 internet business method patents, most of which are arguably members of the class of software patents. Table 1 shows the results of comparing this list with the patents selected by the three definitions, Bessen-Hunt, Graham-Mowery, and Hall-MacGarvie. From this procedure we can learn whether our samples include patents identified by Allison, although we are unable to ascertain how many patents in our sample would *not* have been labeled software by his procedure. That is, we can measure Type I error (missing a software patent we should have identified) but not Type II error (calling a patent a software patent when it is not).

The results are fairly clear: the keyword method of Bessen and Hunt is much more accurate in the sense of avoiding Type I error than either the International Patent Classification (IPC)-based classification or the US patent class-based classification, either of which identify only about half of these patents. Interestingly, Graham-Mowery is better at identifying software patents, whereas Hall-MacGarvie is much better at internet business method patents. This is probably because the US patent class system explicitly admits the existence of such patents, while the IPC does not, so they may end up classified in a wide range of IPC classes.

Layne-Farrar (2005) describes an attempt to determine the Type II error in these different definitions of software patents. She reports that when software experts read a random sample of patents from each dataset to see if they were truly for software, they classified about 5 per cent of the patents in the Graham-Mowery and Allison-Tiller sets as not software.[26] However, over half of the patents in the Bessen-Hunt sample were

[26] Layne-Farrar did not have access to the sample selected by our US patent class rule.

classified as not relating to software. This result suggests that using that definition alone may be problematic.

However, because the GM definition does well on Type II error, and because GM and HM seem to select different groups of patents, whereas BH seems to be more comprehensive (better at Type I error), our preferred definition of software patents combines these definitions in the following way: we take the union of the GM and HM samples, and then intersect this with the BH sample. On the assumption that the HM US class/subclass approach does something similar to the GM IPC class/subclass approach, this intersects a sample that should have approximately 5% type II error (that is, it rarely misidentifies patents as software that are not software) with an inclusive sample that covers almost all software patents plus many other computing patents. The result is shown in the final row of Table 1: our new definition captures over 80 per cent of the patents identified by Allison as software and internet patents, but not at the cost of being too inclusive. We therefore chose to use this combined definition as our preferred sample of software patents, and to present results using the other definitions in an appendix.

Figures 1 and 2 show the trends in software patent growth using the four definitions described above plus an aggregate software patenting series supplied by Aharonian (2001) in his email newsletter.[27] Figure 1 shows the absolute numbers and Figure 2 shows the share of software patents among all granted patents. The Aharonian definition appears to track more closely to the Bessen-Hunt keyword definition more closely than the others, whereas our combined definition is conservative and lies

[27] Greg Aharonian is a patent agent specializing in IT patenting who has been tracking software patents using his own subjective evaluation of subject matter for some time. We include his figures to give some idea of the view of an observer familiar with the sector and its patenting.

somewhere between the two patent class definitions (those using IPC and US patent classes). Using any of the definitions, there is substantial growth since 1976 that accelerates in around 1995-1996. The apparent jump from 1997 to 1998 using any of the definitions results from a jump in applications three years earlier, that is presumably due to the series of court decisions that are described in the next section of the paper.

4. Event studies

In 1994 and 1995, the Court of Appeals for the Federal Circuit (CAFC) handed down a number of decisions that affected the scope of software patents. These decisions include *In re Alappat,* 33 F.3d 1526 (1994, en banc); *In re Warmerdam*, 33 F.3d 1354 (1994); *In re Lowry*, 32 F.3d 1579 (1994), *In re Trovato*, 60 F.3d 805 (1995, en banc), and *In re Beauregard,* 53 F.3d 1583 (1995). In response to the confusion created by the court's determination that much of software was patentable, the USPTO proposed new guidelines for software patentability on May 12, 1995, and published these guidelines on March 29, 1996. In this section of the paper, we describe the decisions and then present the results of several studies of the effects of the decisions on the value of firms in the software industry or holding software patents. Table 3 presents a timeline for the decisions we consider, with the abnormal returns experienced by our firms around the date of each decision (these results are discussed in section 4.1)

In re Alappat

In 1988, Alappat filed an application for a patent on a means of smoothing the appearance of a waveform displayed by a digital oscilloscope. The PTO rejected the application's claims as non-patentable subject matter. Alappat appealed the decision to

the Board of Patent Appeals, which reversed the PTO's decision on June 26, 1991. The patent examiner then requested a reconsideration of the decision by an expanded panel of the Board of Appeals, which on April 22, 1992 reversed the decision of the original panel. On July 29, 1994, the CAFC handed down a decision stating that the invention was not an "abstract idea", but rather, "a specific machine to produce a useful, concrete, and tangible result."[28] This decision was interpreted as a clear expansion of the patentability of software.

In re Warmerdam

The Alappat decision was followed on August 11, 1994 by *In re Warmerdam*, partly affirming and partly reversing the decisions of the Board of Patent Appeals, which had upheld the PTO's rejection of an invention as non-patentable subject matter. The CAFC ruled that the invention, a mechanism for generating data structures for collision avoidance systems, was "nothing more than the manipulation of basic mathematical constructs, the paradigmatic 'abstract idea.'"[29] However, the court held that a machine containing in memory the data structure created by the mechanism would indeed be patentable.[30]

In re Lowry

Warmerdam was followed on August 26, 1994 by *In re Lowry,* which reversed the PTO's rejection of an application on the grounds that a data structure held in memory

[28] In re Alappat 33 F.3d 1526, 31 USPQ2d 1545

[29] In re Warmerdam, 3 F.3d 1354, 31 USPQ2d 1754

[30] Huttner and Strobert, *New York Law Journal*, September 25, 1995

was unpatentable under the "printed matter" doctrine. The "printed matter" doctrine provides that "an article of manufacture which consists of printed matter on a substrate cannot be statutory if it differs from other substrates only by the informational content of the printed matter except where there is some functional interaction between the printed matter and the substrate."[31] In this instance, the PTO had held that Lowry had not shown that the data structures were functionally related to the memory in which they were stored, but the CAFC ruled that the printed matter doctrine did not hold when the information in question was processed by a machine rather than by the human mind..

In re Trovato and In re Beauregard

In *In re Trovato* (December 19, 1994), the CAFC upheld the PTO's rejection of a software patent for containing non-statutory subject matter. The court ruled that "[p]utting Trovato's claims in their most favorable light, the most they provide is a systemic way in which to compute a number representing the shortest path. A new way to calculate a number cannot be recognized as statutory subject matter."[32] On May 12, 1995, the CAFC ruled in *In re Beauregard* that the PTO's rejection of a patent's claims under the printed matter doctrine was incorrect given the precedent set in *In re Lowry.* The patent office admitted its mistake in the case.[33]

New UPSTO Guidelines

---

[31] USPTO 1995. See also *In re Gulack*, 703 F.2d 1381, 217 USPQ 401.

[32] *In re Trovato*, 60 F.3d 805

[33] *The Computer Lawyer*, Vol. 12, No. 5; Pg. 30, May 1995

In response to the above decisions, the USPTO proposed new guidelines for software patentability on May 12, 1995, and published these guidelines on March 29, 1996. The guidelines expanded on *Warmerdam*'s finding with respect to data held in the memory of a machine, stating that the PTO would begin to allow claims related to software embedded in physical media. Claims should be considered processes unless they relate to some type of machine or physical apparatus. The guidelines clarified the definition of the following criteria to be used by examiners to distinguish between

> "a) a computer or other programmable apparatus controlled by software as a statutory 'machine';
>
> b) a computer-readable memory used to direct a computer, such as a memory device, a compact disc or a floppy disk as a statutory 'article of manufacture'; and
>
> c) a series of steps to be performed on or with the aid of a computer as a statutory 'process.'"[34]

The guidelines stated that inventions that were to be considered non-statutory included data compilations or structures independent of physical elements, encoded information representing creative or artistic expression, and processes that do "nothing more than manipulate abstract ideas or concepts."[35]

4.1 Null hypotheses on the effects of the patentability decisions

As stated previously, the expansion of the patentability of software may have had both positive and negative effects. Firms already holding patents at the time of the

[34] USPTO guidelines quoted in Huttner and Strobert

[35] op. cit.

decision may be positively affected due to an increased ability to exclude competitors from a market or collect licensing revenues. We will thus compare the effects of patentability on patent holders and non-patent holders as of the time of the event, with the expectation that the market reaction should be larger for patent holders. Note that another set of firms that may benefit from software patents are start-ups for whom patents may help secure financing. Because we are focusing on the market value of publicly traded firms, we are unable to study this effect.[36]

We also consider the potential negative effects of the decisions. For all firms, the decisions may have increased costs by creating the need to engage in licensing negotiations and by increasing the potential for hold-up. We hypothesize that the latter effect is especially relevant for firms in "downstream" market segments, i.e., firms producing software that must interact with or operate on top of other software or platforms. We thus test whether the market reacted negatively for firms specializing in applications software and software-related services relative to firms producing for "upstream" segments (middleware, systems software, and hardware).[37]

4.2 Results of the event studies

Table 3 contains the results of the event studies that identify the market's response to each of the above-mentioned decisions, for the entire sample of firms and for software firms only. We calculate the cumulative abnormal returns (CARs) associated with the events, and their standard errors, using the regression approach described by

[36] Cockburn and MacGarvie (2006) study the effects of patent "thickets" on entry in narrowly-defined software product areas.

[37] See the appendix for our classification of firms into these segments.

Salinger (1992).[38] We regress the stock returns for firms in our sample on the market return (using the CRSP value-weighted market index), and a series of dummies for the days in an event window from one day before to three days after the event in question (t = -1 to t = +3, where t is the date of the event). Our estimation window runs from t-90 days to t-30 days.

For each event we show the average CAR for all firms, for software firms only (firms whose primary SIC code is 73XX), and for firms specializing in applications software or software-related services, as defined by the Corptech directory.[39] We also include rank sum tests of the difference in the CAR distribution between the latter group and other firms, and those firms that do and do not hold software patents (according to the combined definition) at the time of the event. We group the CARs in this way because we expect the effect of the decisions to differ depending on whether firms can expect to take advantage of the increased strength of software patents. We expect that firms holding software patents at the time of decisions that broaden patentable subject matter to include some types of software will be positively affected relative to firms without patents. We also expect that firms in "downstream" sectors – like applications software and software-related services – will be negatively affected relative to upstream firms in hardware and middleware or systems software, because they may be forced to

[38] see p. 41-42.

[39] As of the data of our data (2002), Compustat classified IBM into software services. Because of the importance of this firm in software patenting in general and because for most of our time period IBM was primarily a hardware firm, we have reclassified it into computing machinery. However, throughout this paper, we found that estimates with and without including observations on IBM were substantially the same.

license in patented technology in order to ensure that their products will function on top of middleware or the operating systems.

The first event we consider is the Supreme Court's decision in *Diamond v. Diehr,* followed by the Compton patent grant and subsequent revocation. None of these events are associated with significant abnormal returns, with the exception of the Compton patent assertion, which was slightly negative (significant at the 10 per cent level). However, according to the rank sum test, the distribution of CARs for firms with no software patents at the time of this event was significantly to the right of that for firms with CARs when the patent was revoked, which suggests that they benefited from the decision, presumably because they would have been disadvantaged if such patents became widespread in the sector.

The *In re Alappat* decision is widely viewed as the groundbreaking one on software patentability and this is reflected in the abnormal returns at the time of the decision.[40] The software industry in general experienced negative abnormal returns, statistically significant at the 10 per cent level. Firms that specialized in applications software or services saw even more significant negative CARs of over 4 percent on average.[41] This is significantly lower than the CARs for other firms in the industry.

[40] In reference to *In re Alappat*, Evans and Layne-Farrar state that "This ruling cemented the statutory standing of software patents." (p. 6). Mann (2004) says Alappat "cleared the way for software patents" Cohen and Lemley (2001) state that "In 1994, the en banc Federal Circuit decided *In re Alappat*, opening a new era in software patent protection"(p. 10).

[41] When we refer to "applications/services firms" in this section, we mean firms operating only in these fields.

Furthermore, firms without patents also see a significantly negative market reaction relative to firms with patents.[42]

The next decision, *In re Warmerdam*, had mixed implications, partly expanding and partly restricting the patentability of software. The market reaction is accordingly mixed, with positive CARs for the industry as a whole, but a significantly negative difference between CARs for firms without and firms with software patents. *In re Lowry,* a clear expansion of patentability that followed *Warmerdam*, is associated with negative CARs for applications/services firms, and slightly negative for software firms in general, whereas *In re Trovato* had little impact.

The culmination of these decisions was the USPTO's announcement of proposed new guidelines for software patents, which is associated with CARs that are significantly lower for applications/services firms and non-patent holders. The final issuance of the guidelines is associated with yet another negative reaction for applications and services firms. However, there is a puzzling positive reaction for non-patent holders relative to patent holders when the guidelines were first proposed, which may reflect the resolution of uncertainty.[43]

5. Market value and software patenting

In this section of the paper we employ the Hall, Jaffe, Trajtenberg (HJT (2005)) methodology to estimate the contribution of software patents to Tobin's Q. We compare

[42] The comparisons of the mean CARs for applications/services firms vs. others, and non-patenting vs. patenting firms, are performed using a rank sum test, which is a non-parametric alternative to the t-test.

[43] We are in the process of further examining this result in a multiple regression model of the determinants of the CARs associated with this decision.

the value of these patents to patents in general, both before and after the changes in software patentability rules in 1995. In order to adjust for differences in the quality of the patents, we perform the same exercise with cite-weighted patents, and with cite-weighted patents excluding self-cites.

The HJT 2005 paper specifies a firm-level market-value equation that is linear and additively separable (following Griliches (1981)). We follow the HJT paper closely in the ensuing description of the model. The market value of firm *i* in year *t* is modeled as:

$$V_{it} = q_t(A_{it} + \gamma K_{it})^{\sigma_t} \tag{1}$$

where A_{it} stands for physical assets, and K_{it} the firm's knowledge assets. Taking logarithms yields the following equation:

$$\log V_{it} = \log q_t + \sigma_t \log A_{it} + \sigma_t \log[1 + \gamma(K_{it} / A_{it})] \tag{2}$$

Assuming constant returns to scale (that is, $\sigma_t = 1$) and moving A_{it} to the left hand side, we can express the model with Tobin's q as the dependent variable:

$$\log Q_{it} \equiv \log(V_{it} / A_{it}) = \log q_t + \log[1 + \gamma(K_{it} / A_{it})] \tag{3}$$

We specify the knowledge capital K_{it} as a function of the stock R&D spending and the stock of patents, both unweighted and weighted by citations. Given our specific interest in this paper, we further break *K* down according to whether the patents are in software-related fields. This results in the following estimating equation:

$$\log Q_{it} = \log q_t + \log[1 + \gamma_1(RD_{it} / A_{it}) + \gamma_2(P_{it} / RD_{it}) + \gamma_3(SP_{it} / RD_{it})] + \varepsilon_{it} \tag{4}$$

in which RD_{it} is firm *i*'s R&D capital stock in year *t*, P_{it} is a measure of patent stock in year *t*, and SP_{it} is a measure of the software patent stock in year *t*. The coefficient γ_2 measures the impact of patents above and beyond that of the R&D that produces them

and γ_3 measures the premium or discount associated with software patents. *P* and *SP* are based on patents dated by year of application, citation-weighted patents dated by year of application, or citations excluding self-citations to patents dated by year of application.

A full set of two-digit industry and year dummies for the sample period (1980-1999) are also included, and in some cases industry-year interactions.[44] The R&D and patent stock measures are constructed using the usual declining balance method with a depreciation rate of 15 per cent:

$$K_{it} = (1-\delta)K_{i,t-1} + R_{it} \tag{5}$$

where R_{it} is R&D spending, granted patents applied for in year *t*, or granted software patents applied for in year *t*. Patents are either simple counts, counts weighted by citations they subsequently received, or counts weighted by citations excluding self-citations.[45]

Interpretation of the coefficients in an equation like equation (4) can be difficult, since the variables are in a variety of units (dollars per dollar, counts per dollar, etc.). To enhance comparability, we computed the elasticity of Tobin's Q with respect to each of our variables using the following equation:

$$\frac{\partial \log Q_{it}}{\partial \log X_{it}^{j}} = \frac{\gamma_j X_{it}^{j}}{1+\gamma_1(RD_{it}/A_{it})+\gamma_2(P_{it}/RD_{it})+\gamma_3(SP_{it}/RD_{it})} \tag{6}$$

[44] We include these interactions to deal with the large increase in valuation for software firms at the end of our sample period, which was presumably due to the dotcom "bubble".

[45] Our patent data end in 2002 and our regressions in 1999. This means that we have three years to observe the issuance of the youngest patents in the sample, which is enough time to observe most of them. However, it is not sufficient time to observe forward citations for the most recent patents, so we have adjusted the number of citations received by each patent by the ratio of the total number expected to be received by a patent in that technology class to the average number such patents have received by the time of the relevant citation lag, using a methodology described in Hall, Jaffe, and Trajtenberg 2001.

where $j = 1,2,3$ and X^j_{it} is the corresponding right hand side variable. We then averaged these elasticities across the observations and reported them in the bottom panel of the table.

We estimate equation (4) using nonlinear least squares and report the results in Tables 4. Eicker-White standard errors are computed to ensure robustness to heteroskedasticity. We included a complete set of year dummies for each of the 5 two-digit industries in our sample (machinery, electrical machinery, instruments, telecommunications, and software). Although we were concerned that general trends in the valuation of software firms, especially during the late 1990s, might be contaminating our results, there was in fact almost no difference between the results in Table 4 and those using a single set of year dummies for all of the sectors. There are three sets of columns in the table, one set corresponding to the whole period, one set for the pre-guidelines period (1980-1994) and one for the post-1994 period, in order to focus on the question of whether the value of software patents increased following the *Beauregard* decision and the Commissioner of Patents' issuance of new guidelines on software patentability in 1995.

The overall results are similar to those in Hall, Jaffe, and Trajtenberg (2005), with R&D having a strong relationship with Tobin's Q, and either patents or citation-weighted patents having a somewhat weaker but still significant relationship in the presence of R&D. The most noteworthy result in the table is that software patents, whether or not they are weighted by forward citations, are valued at a significant premium by the market, relative to ordinary patents, but that this is entirely due to their value following the 1994/1995 changes in software patenting. The average elasticities suggest that

doubling the patent yield per R&D dollar increases market value by one per cent before 1995 and two per cent afterwards, whereas if the patents in question were software patents, the numbers would be the same before and 4 per cent after. The results for citation-weighted patents are similar, although the differential boost from patenting in software is slightly lower when the patents are weighted by citations. This suggests that, relative to other patents, owning software patents may be more important than the fact that they are cited. However we are reluctant to draw strong conclusions from this finding because there has been limited time to observe the citations during the later period.

This table also shows that removing self-citations from the citation weights makes little difference to the overall results for software patents, although it does reduce the impact of cites for non-software patents. That is, self-citations to non-software patents are valued more highly than ordinary citations, as in Hall Jaffe Trajtenberg (2005). The total effect of citations to software patents is also somewhat lower when self-citations are excluded, so the differential is about the same.

In Table 5 we split the sample into software and hardware (machinery, electrical machinery, instruments, and telecommunications) firms and show the results for the same estimation as in Table 4, but only for unweighted patent stocks and for cite-weighted patents excluding self-citations, and only for the pre-1995 period and the 1995-1999 period separately. This table reveals some interesting detail. First, weighted and unweighted software patents are valued in the same way as ordinary patents prior to 1995. Second, the patents held by software firms prior to 1995 are essentially not valued

at all by the market although their R&D clearly is valued.[46] In contrast, after the court decisions of 1994, software patents held by either hardware or software firms become more valuable, with a premium on the patent elasticity of 0.02 to 0.03. That is, if a hardware firm doubles its software patent yield per R&D dollar, this is associated with a two per cent increase in market value; for a software firm, it is associated with a three per cent increase in market value.

Third, citation weighted patents are valued very differently in the two sectors. For hardware firms, there is no premium for citations to software patents; they are valued in the same way as other patents, with an elasticity of about 0.02. For software firms, ordinary citation weighted patents are not valued at all, whereas citation-weighted software patents have an elasticity of about 0.02. One simple interpretation of these results is that patent rights are more valuable to the firm when they are rights to the main technology of their industry, so cite-weighted software patents are especially valuable to software firms, but not more valuable that other patents to hardware firms. This suggests that the patents are valuable for strategic reasons in hardware, and for intrinsic technological reasons in software.

6. Conclusions

We obtain two sets of conclusions based on our two approaches to measuring the private value of software patents. First, we conclude that, as measured by the stock market's reaction to legal decisions expanding the patentability of software, there is no

[46] Note that this is not simply because they do not have software patents. Prior to 1995, 23 per cent of the software firms hold software patents by our definition and 30 per cent have patents. After 1994, the numbers are 31 per cent and 38 per cent respectively.

evidence that the expansion of software patentability benefited firms in the software industry. There is limited evidence of relatively negative effects on some types of software producers – applications producers and providers of software-related services, and firms with no patenting experience at the time of the decisions.

Our second approach examines the relationship between Tobin's Q and firms' patent and citation stocks. Surprisingly, software patents seem to be more highly valued by the market than ordinary patents, even in the presence of controls for the overall increase in the market valuation of software firms. However, for hardware producers, this appears to reflect the strategic value of a software patent rather than the technological value of the patented invention (because only patents – not the citations to those patents – are valued by the market). The finding that forward citations to patents held by software firms are positively and significantly valued by the market suggests that these patents *do* reflect technologically valuable inventions and that the market recognizes this value *per se*.

Combining these two sets of findings, we conclude that the market evaluated software patents as unimportant *ex ante* and expected that the expansion of software patentability would negatively affect firms in downstream sectors and firms without patents. *Ex post*, of course, a greater number of firms in all ICT sectors invested in these patents. In addition, firms in the ICT sector that hold software patents are found to be valued at a significant premium relative to firms without software patents. This story paints a picture of firms adjusting to a new environment and set of rules, which although they were initially perceived as negative for the software sector, have provided some benefits, at least for firms large enough to be publicly traded on the market. Nevertheless

we should also emphasize that the majority of the software patents acquired during the past twenty years have been acquired by non-software firms in the ICT sector (in our sample the figure is almost 90 per cent), which suggests that the growth in these patents is driven to a great extent by the needs of hardware firms for large patent portfolios rather than by the needs of software firms to protect their inventions.

7. References

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

Allison, J. R. and E. H. Tiller, 2003, "Internet Business Method Patents," in W. M. Cohen and S. A. Merrill (eds.) *Patents in the Knowledge-Based Economy*, National Research Council, Washington: National Academies Press, pp. 259-84.

Aharonian, G. (2005). "Critiques of Software Patent Examination," on http://www.bustpatents.com/

Arundel, A., J. Cobbenhagen, and N. Schall. (2002). *The Acquisition and Protection of Competencies by Enterprises*. Report for EIMS project 98/180, DG Enterprise, European Union, Luxembourg.

Austin, D. (1993). "An Event-Study Approach to Measuring Innovative Output: The Case of Biotechnology" *American Economic Review* 83 (2): 253-258.

Bakels, R. (2005). "Current IPR Policies and Law Followed by Major Organisations in the Domain," in Ghosh et al, Report to the European Commission on The Effects of Allowing Patent Claims for Computer-Implemented Inventions, August.

Bakels, R. and P. B. Hugenholtz (2002). "The Patentability of Computer Programs," A Report to the European Parliament, Amsterdam: IViR.

Barton, J. H. (2002). "Non-Obviousness," *IDEA Journal of Law and Technology* 42.

_____. (2000). "Reforming the Patent System," *Science* 287: 1933-1934.

Bessen, J. and R. M. Hunt (2003). "An Empirical Look at Software Patents", Working Paper. http://www.researchoninnovation.org/swpat.pdf

Blind, K., J. Edler, and R. Nack (2001). "Micro- and Macro-economic Implications of the Patentability of Software Innovations. Intellectual Property Rights in Information Technologies between Innovation and Competition," Karlsruhe/Munich: Fraunhofer Institute and Max Planck Institute.

Cockburn, I. (2001). "Issues in Business Method Patents," Boston University and NBER.

Cockburn, I. and M. MacGarvie (2006) "Entry, Exit and Patenting in the Software Industry", Boston University Working Paper.

Cohen, J. E. and M. A. Lemley (2001). "Patent Scope and Innovation in the Software Industry," *California Law Review* 89(1): 1-57.

Cohen, W., A. Goto, A. Nagata, R. Nelson, and J. Walsh. (2002). "R&D Spillovers, Patents and the Incentives to Innovate in Japan and the United States." *Research Policy* 31:1349-1367.

Dreyfuss, R. C. (2001). "Examining State Street Bank: Developments in Business Method Patenting," *Computer und Recht International* 2001(1): 1-9.

_____. (2000). "State Street or Easy Street: Is Patenting Business Methods Good for Business?" In *Intellectual Property Law*, Volume 6, Chapter 14. London: Juris Publishing Company, Ltd.

Dvorak, John. (2005). "Software Patents: Microsoft's Fatal Error," *PC magazine*, April 18, 2005.

European Commission. (2002). "Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented Inventions," Brussels, Belgium.

European Patent Office (1989). European Patent Convention, Article 52. http://www.european-patent-office.org/legal/epc/e/ar52.html

Evans. D. and A. Layne-Farrar (2004) "Software Patents and Open Source: The Battle Over Intellectual Property Rights", *Virginia Journal of Law and Technology,* Vol. 9, No. 10, Summer 2004.

Gao, L. (2005) "Essays On Investments And Financing Decisions In Complementary Networks Systems", unpublished doctoral dissertation, Boston University.

Graham, S. J. H. and D. C. Mowery (2003). "Intellectual Property Protection in the U. S. Software Industry," in W. M. Cohen and S. A. Merrill (eds.), *Patents in the Knowledge-Based Economy*, Washington, DC: National Academies Press.

Graham, S. J. H. and D. C. Mowery (2005). "Software Patents: Good News or Bad News," in R. W. Hahn (ed.), *Intellectual Property Rights in Frontier Industries: Software and Biotechnology*. Washington, D.C.: AEI-Brookings Joint Center for Regulatory Studies.

Hall, B. H. (2005). "Exploring the Patent Explosion," *Journal of Technology Transfer* 30: 35-48.

Hall, B. H. (2003). "Business Method Patents, Innovation, and Policy," NBER Working Paper No. W 9717.

Hall, B. H., S. J. H. Graham, S. J. H., D. Harhoff, and D. C. Mowery. (2003). "Prospects for Improving U.S. Patent Quality via Post-grant Opposition," MIT Press: *Innovation Policy and the Economy* 4: 115-143.

Hall, B. H., A. Jaffe, and M. Trajtenberg (2005), "Market Value and Patent Citations," *Rand Journal of Economics* 36: 16-38.

Heller, M.A. and R.S. Eisenberg. (1998). "Can Patents Deter Innovation? The Anticommons in Biomedical Research," *Science* 280: 698-701.

Hunt, R. M. (2001). "You Can Patent That? Are Patents on Computer Programs and Business Methods Good for the New Economy?" *Philadelphia Federal Reserve Bank Business Review* 2001(Q1): 5-15.

_____. (1999). "Nonobviousness and the Incentive to Innovate: An Economic Analysis of Intellectual Property Reform," FRB-Philadelphia Research Department Working Paper No. 99-3.

Huttner, C. and A. Strobert (1995) "New Proposed Guidelines on Patentability Of Computer-Implemented Inventions", *New York Law Journal*, September 25, 1995.

Kasdan, J. (1999). "Obviousness and New Technologies," New York, NY: Columbia University Center for Law and Economics Studies Working Paper No. 146.

_____. (1994). "Fascinatin' Algorithm: Patent Protection for Computer Programs," Columbia University Center for Law and Economic Studies Working Paper No. 94.

Lerner, J. (2001). "Where Does State Street Lead? A First Look at Finance Patents, 1971-2000," *Journal of Finance* 57: 901-930.

_____. (1995). "Patenting in the Shadow of Competitors," *Journal of Law and Economics* 38(2): 463-496.

Layne-Farrar, A. (2005). "Defining Software Patents: A Research Field Guide," LECG, Chicago: manuscript.

Lunney, G. S., Jr. (2001). "e-Obviousness," *Michigan Telecommunications Technology Law Review* 7: 363-422.

Mann, R. (2004) "The Myth of the Software Patent Thicket", working paper.

Meurer, M. J. (2002). "Business Method Patents and Patent Floods," *Washington University Journal of Law and Policy* 8: 309.

Murmann, J. P. (2003). *Knowledge and Competitive Advantage: The Co-evolution of Firms, Technology, and National Institutions*, Cambridge: Cambridge University Press.

Raduchel, W. (2006). "The Economics of Software," in D. W. Jorgenson and C. W. Wessner (eds.), *Software, Growth, and the Future of the U.S. Economy, Report of a Workshop* Washington DC: National Academies Press.

Salinger, M. (1992), "Standard Errors in Event Studies," *The Journal of Financial and Quantitative Analysis,* Vol. 27, No. 1 (March 1992), 39-53.

Samuelson, P. (1984), "CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form," *Duke Law Journal*, pp. 663-702.

Scotchmer, S. (1996). "Protecting Early Innovators: Should Second Generation Products be Patentable?," *Rand Journal of Economics* 27: 322-331.

_____. (1991). "Standing on the Shoulders of Giants: Cumulative Research and the Patent Law," *Journal of Economic Perspectives* 5: 29-41.

Spindler, G. (2003). "The European Legal Framework for Software Patents," paper presented at the EPIP conference on New Challenges to the Patent System, EPO, Munich, Germany, April 24/25, 2003.

State Street Bank and Trust Co., Inc. v. Signature Financial Group, Inc., 149 F.3d 1368 (Fed. Cir. 1998).

Sterne, R. and L. Bugaisky (2004), "The Expansion of Statutory Subject Matter Under the 1952 Patent Act", *Akron Law Review* 37: 216-229.

USPTO (1995), "Proposed Guidelines for Computer-Implemented Inventions," http://www.uspto.gov, 1 June 1995.

USPTO. (1999). "White Paper on Automated Financial or Management Data Processing Methods (Business Methods)," http://www.uspto.gov/web/menu/busmethp/index2.htm

Appendix tables and figures

Differences between software patent definitions

We consider several different definitions of what constitutes a software patent (see p. 12-17 for a discussion of these definitions). Figures 1 and 2 plot the total number of software patents granted and the number of software patents as a percentage of all patents granted, for each of the alternative definitions.

Table A2 presents results from the market value model for the Bessen-Hunt, Graham-Mowery, Hall-MacGarvie, and "combined" (GM or HM intersected with BH) definitions. Each column presents estimates for one of the definitions, and the differences between them highlight the importance of the choice of definition.. For software firms, looking at the entire sample period (1990-1999), software patents and forward citations are positively and significantly associated with Tobin's Q across all the definitions. The magnitudes of the elasticities are also comparable across definitions.

However, there are differences between the definitions when we look at non-software firms and when we consider the change in valuation over time. Here, the BH definition diverges from the others, showing a positive and significant valuation of citation-weighted software patents for non-software firms, even in the pre-1994 period, as well as a *decrease* over time in the elasticity of market value with respect to software patents held by software firms. This may reflect the fact that the BH set of patents appears to contain a considerably greater number of hardware patents, at least according to the sample studied by Layne-Farrar.

Table A1

Category Definitions Based on CorpTech product codes (SOF codes)

Based on Gao (2005)

SYSTEMS SOFTWARE:

A. Utility systems software 1

MIDDLEWARE SOFTWARE:

A. Utility systems software 2
B. Communications systems software
C. Internet tools
D. Software development systems
E. Artificial intelligence software
F. Database/file management software
D. Software development systems

APPLICATIONS SOFTWARE:

A. Accounting, banking, financial, government, military, legal, real estate, insurance, health services, public utilities, transportation, sales/marketing and distribution software
B. Technical/scientific software
C. Construction, facilities and communications management software
D. Manufacturing software systems
E. Media and communications software
F. Office automation software
G. Educational and training software
H. Other applications software, not elsewhere specified

SERVICES:

A. Software related services

Table A2
Elasticities from market value equations, by software/non-software and time period

	Software firms				Non-software firms			
	Bessen-Hunt	**Graham-Mowery**	**Hall-MacGarvie**	**Combined**	**Bessen-Hunt**	**Graham-Mowery**	**Hall-MacGarvie**	**Combined**
1980-1999 (2173 software, 8717 non-software)								
R&D/Assets	***0.361***	***0.353***	***0.351***	***0.350***	***0.299***	***0.304***	***0.305***	***0.304***
Patents/R&D	-0.002	0.000	0.000	0.000	**0.010**	***0.018***	***0.020***	***0.019***
SW Pats/R&D	***0.013***	***0.016***	***0.023***	***0.024***	***0.026***	**0.006**	0.002	**0.008**
R&D/Assets	***0.360***	***0.352***	***0.353***	***0.351***	***0.297***	***0.301***	***0.301***	***0.301***
Citations/Patents	-0.004	0.001	0.001	0.001	***0.022***	***0.037***	***0.038***	***0.037***
SW Cites/Patents	***0.011***	***0.016***	***0.020***	***0.018***	***0.023***	0.003	0.002	0.004
R&D/Assets	***0.359***	***0.352***	***0.352***	***0.349***	***0.299***	***0.302***	***0.302***	***0.302***
Citations/Patents (excluding self-cites)	-0.002	0.001	0.001	0.000	***0.016***	***0.024***	***0.025***	***0.024***
SW citations/Patents (excluding self-cites)	***0.010***	***0.014***	***0.018***	***0.018***	***0.017***	0.001	0.000	0.001
1980-1994 (924 software, 5989 non-software)								
R&D/Assets	***0.413***	***0.428***	***0.408***	***0.415***	***0.279***	***0.286***	***0.286***	***0.286***
Patents/R&D	**-0.026**	-0.001	-0.001	0.000	**0.009**	***0.015***	***0.017***	***0.016***
SW Pats/R&D	**0.033**	**-0.018**	0.012	0.004	***0.021***	0.004	**-0.001**	0.002
R&D/Assets	***0.416***	***0.415***	***0.413***	***0.405***	***0.277***	***0.282***	***0.282***	***0.282***
Citations/Patents	**-0.013**	0.002	0.002	0.002	***0.021***	***0.034***	***0.036***	***0.034***
SW Cites/Patents	**0.018**	0.006	0.009	0.015	***0.019***	0.002	0.000	0.002
1995-1999 (1249 software, 2728 non-software)								
R&D/Assets	***0.300***	***0.294***	***0.292***	***0.293***	***0.338***	***0.333***	***0.333***	***0.335***
Patents/R&D	-0.002	0.000	0.000	0.000	0.019	***0.035***	***0.036***	***0.036***
SW Pats/R&D	**0.009**	***0.031***	**0.030**	***0.018***	***0.034***	***0.021***	***0.019***	***0.018***
R&D/Assets	***0.298***	***0.294***	***0.291***	***0.293***	***0.337***	***0.335***	***0.335***	***0.335***
Citations/Patents	-0.003	0.000	0.000	0.000	***0.029***	***0.044***	***0.046***	***0.045***
SW Cites/Patents	***0.012***	***0.018***	***0.025***	***0.018***	***0.030***	**0.013**	***0.011***	***0.012***
R&D/Assets	***0.298***	***0.293***	***0.290***	***0.292***	***0.338***	***0.338***	***0.336***	***0.338***
Citations/Patents (excluding self-cites)	-0.002	0.000	0.000	0.000	***0.024***	***0.021***	***0.021***	***0.021***
SW citations/Patents (excluding self-cites)	**0.011**	***0.019***	***0.026***	***0.019***	**0.027**	0.006	**0.008**	0.005

All regressions include a complete set of sector-year dummies and a dummy for missing R&D.

The quantities shown are the average elasticity of the market value with respect to the variable in the first column. The significance tests are based on the corresponding coefficient and its heteroskedastic-consistent standard error.

Bold text indicates significance at the 5% level; *Bold italics significance at the 1% level.*

Table A3: Total number of firms and number of software firms in the sample, by year

Year	All firms	Software firms
1980	303	15
1981	358	23
1982	377	25
1983	427	31
1984	442	35
1985	452	40
1986	459	45
1987	470	56
1988	453	62
1989	457	67
1990	459	72
1991	484	88
1992	528	104
1993	591	122
1994	653	148
1995	750	191
1996	811	226
1997	820	241
1998	774	243
1999	822	350

Figure 1
Software Patents Granted

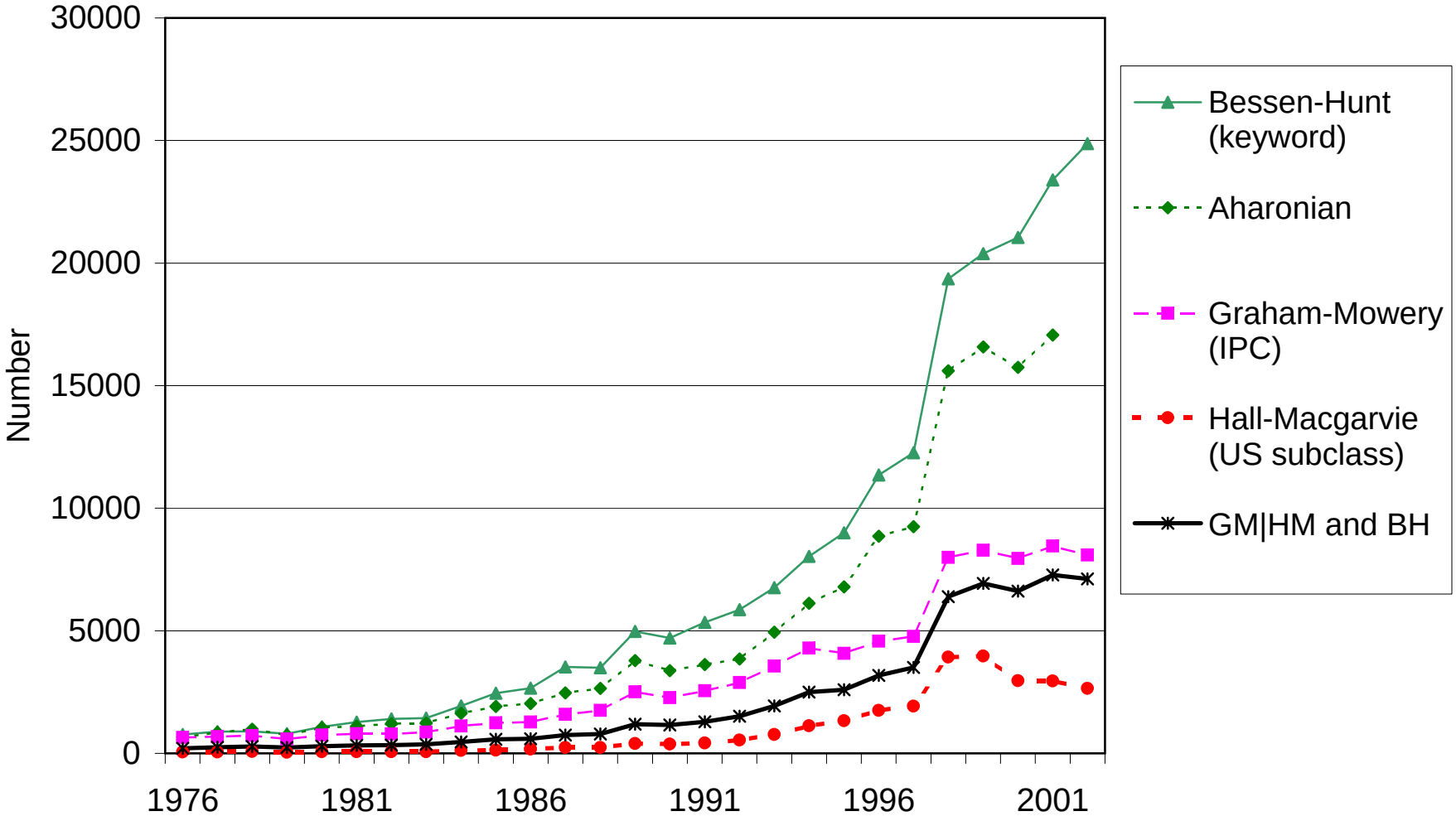

Figure 2
U.S. Software patents granted as share of all patents granted

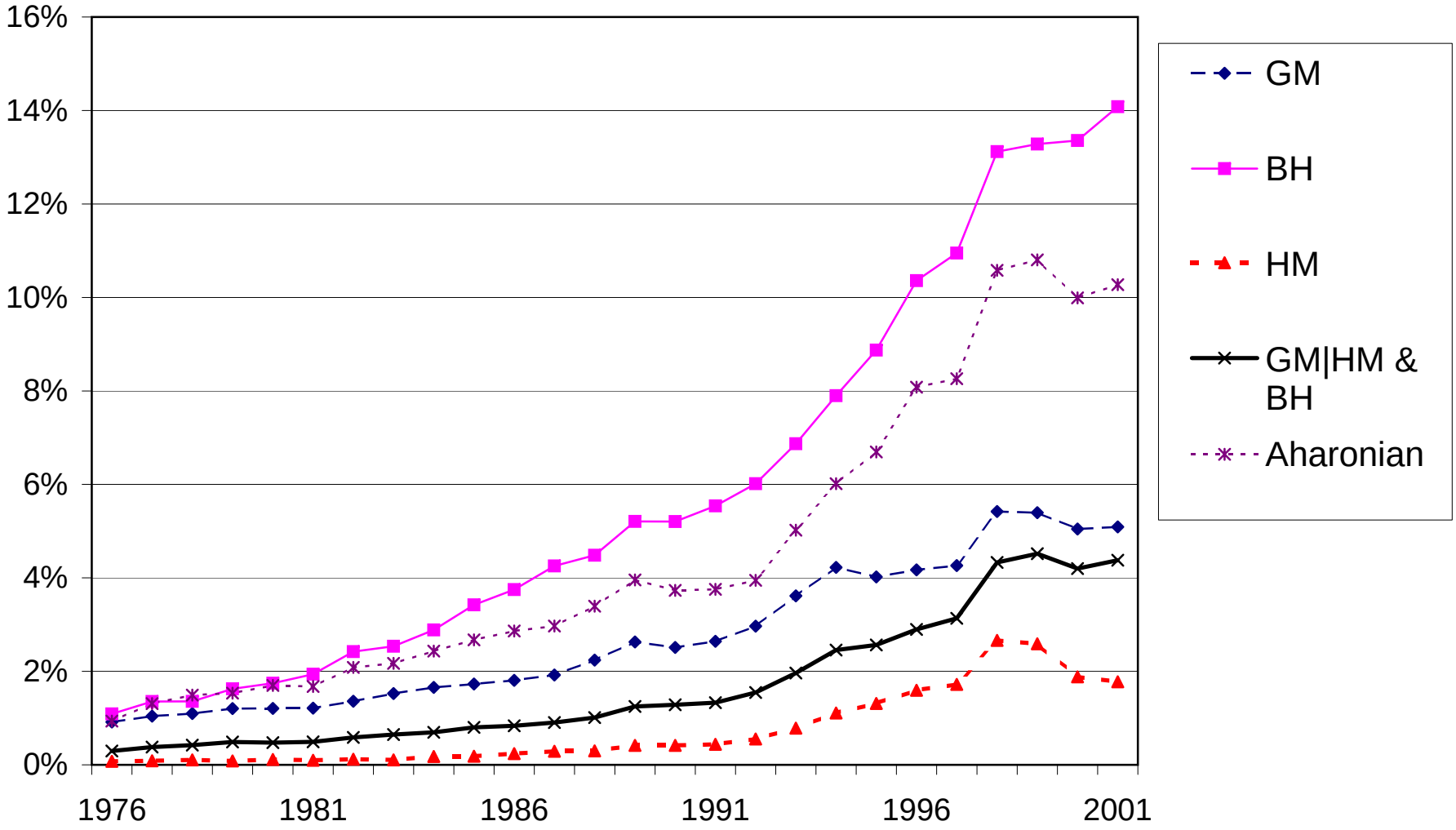

Table 1
Overlap with Allison's Samples

Definition	Original software patent list (2000)		Internet business method patent list		Union of the two lists	
	Number	Share	Number	Share	Number	Share
Allison+	**100**	**100%**	**230**	**100%**	**330**	**100%**
Bessen-Hunt (BH)	92	92%	211	92%	303	92%
Graham-Mowery (GM)	60	60%	89	39%	149	45%
Hall-Macgarvie (HM)	40	40%	150	65%	190	58%
Combined*	83	83%	183	80%	266	81%

The table shows the number of patents according to each definition that are found in the lists supplied by Allison.
+The original list is the set of patents given in Allison and Lemley (2000); the internet business method list is that from Allison and Tiller (2003).

*This set of patents is the union of GM and HM intersected with BH.

Table 2
Summary statistics for 1,594 firms 1980-1999 (10,890 observations)

Variable	Mean	Standard deviation	Median	Minimum	Maximum
Number of employees	5	21	0	0	406
Tobin's Q (market-to-book ratio)	5.49	6.16	2.59	0.016	20
R&D stock (millions of $96)	206.42	1324.46	11.84	0	29,510
R&D Stock / tangible (book) capital	1.18	1.38	0.66	0	5
D(no R&D this year)	0.13	0.33	0	0	1
Patents per year	12	90	0	0	2,711
Patent stock	53.43	408.58	0.65	0	10,526
D(no patent stock this year)	0.45	0.50	0	0	1
D(no software patent stock this year)	0.80	0.40	1	0	1
D(never patented)	0.36	0.48	0	0	1
SW patent stock (B-H)	11.63	113.52	0	0	4,617
SW patent stock (G-M)	8.63	89.87	0	0	3,517
SW patent stock (H-M)	2.80	37.90	0	0	1,724
SW patent stock (Combined)	6.13	74.33	0	0	3,294
Patent stock / real R&D stock	0.31	1.01	0.01	0	25
Cite-weighted patents per year*	224	1679	0	0	50,972
Cite-weighted patent stock*	1031	7906	6.17	0	220,358
Cite-wtd SW patent stock (B-H)*	297	2820	0	0	105,504
Cite-wtd SW patent stock (G-M)*	220	2329	0	0	81,729
Cite-wtd SW patent stock (H-M)*	76	1000	0	0	42,001
Cite-wtd SW patent stock (combined)*	170	1985	0	0	77,095
Cite-wtd patent stock / real R&D stock*	6.15	26.38	0.12	0	1,323
Cite-weighted patents per year**	98	695	0	0	18,482
Cite-weighted patent stock**	519	3684	3.68	0	86,708
Cite-wtd SW patent stock (B-H)**	142	1207	0	0	38,522
Cite-wtd SW patent stock (G-M)**	113	1062	0	0	31,999
Cite-wtd SW patent stock (H-M)**	34	391	0	0	14,051
Cite-wtd SW patent stock (combined)**	82	834	0	0	27,070
Cite-wtd patent stock / real R&D stock**	3.72	14.49	0.06	0.00	508

*Corrected for citation truncation after 2002

**Corrected for citation truncation after 2002 and self-citations excluded.

Table 3
Excess Returns [CAR(-1,+3)] to ICT Firms around Various Software Patenting Events

			CAR (-1,+3) in %				
						Rank sum test for differences in CAR	
Event	**Date**	**Description**	**All firms**	**SW firms**	**App/srv only**	**App/srv vs others**	**without SW pats vs with**
Diamond v Diehr	3-Mar-81	patenting allowed for software embodied in rubber-curing process (in a physical process)	0.1	2.02	2.02	1.00	0.07
Compton patent	16-Nov-93	assertion of multi-media display patent held by Compton	-0.89*	-0.81	-0.83	-0.04	-1.28
Compton revocation	28-Mar-94	Compton's multi-media patent re-examined by USPTO and revoked	0.59	1.61	1.62	0.25	-2.13**
In re Alappat	29-Jul-94	33 F.3d 1526 (Fed. Cir. 1994) (en banc). Narrowing mathematical algorithm limitation on patentable subject matter and thereby expanding patentable subject matter. - oscilloscope software	-0.48	-2.05*	-4.29***	-2.24**	-2.29**
In re Warmerdam	11-Aug-94	partly affirmed, partly reversed PTO's rejection of a data structure for collision avoidance systems as non-patentable subject matter - machine containing the data structure created by the mechanism patentable.	1.37***	3.47***	3.46**	1.63*	-1.88*
In re Lowry	26-Aug-94	data structures are patentable if held in a machine (printed matter doctrine does not apply) - confirming Warmerdam	-0.66	-1.92*	-3.69***	-1.96**	-0.38
In re Trovato	19-Dec-94	A new way to calculate a number cannot be recognized as statutory subject matter - countered growing trend of SW patentability	-0.05	-0.67	1.06	-0.64	0.54
In re Beauregard	12-May-95	IBM appealed a decision to exclude using printed matter doctrine, PTO admits mistake (had contradicted *in re Lowry*)	2.16***	-0.06	-2.15*	-2.23**	-1.49
USPTO new guidelines proposed	30-May-95	new guidelines allow patenting on software embodied in any hardware medium.	-1.32	-1.12	-0.55	0.52	2.59***
USPTO new guidelines final	29-Mar-96	new guidelines allow patenting on software embodied in any hardware medium.	-0.17	-1.05	-1.13	-1.75*	0.2

Each cell shows the average Cumulative Abnormal Return for a 5 day window (-1,+3) around the event date.
The number of firms is shown in parentheses. ***, **, and * denote significance at the 1, 5, and 10 per cent level respectively.

Table 4

Market Valuation of Software Patents (combined definition)

	All years (1980-1999)			1980-1994			1995-1999		
R&D stock/assets	0.568	0.576	0.572	0.676	0.683	0.683	0.447	0.451	0.445
	(0.031)	(0.031)	(0.031)	(0.052)	(0.051)	(0.051)	(0.036)	(0.036)	(0.036)
Pats/R&D stock	0.068			0.057			0.151		
	(0.018)			(0.019)			(0.059)		
SW pats/R&D stock	1.479			0.388			3.782		
	(0.385)			(0.358)			(0.765)		
Cite-wtd pats/		0.010			0.009			0.012	
R&D stock		(0.001)			(0.002)			(0.004)	
Cite-wtd SW pats/ R&D		0.034			0.018			0.064	
stock		(0.010)			(0.011)			(0.021)	
Cite-wtd pats excl. self-			0.012			0.012			0.021
cites/ R&D stock			(0.002)			(0.002)			(0.008)
Cite-wtd SW pats excl. self-			0.033			0.016			0.127
cites/ R&D stock			(0.012)			(0.013)			(0.040)
No R&D	0.395	0.425	0.411	0.520	0.561	0.555	0.225	0.228	0.211
	(0.051)	(0.053)	(0.052)	(0.069)	(0.071)	(0.071)	(0.077)	(0.076)	(0.075)
Log likelihood	-14259.2	-14212.7	-14232.7	-8932.2	-8893.9	-8902.9	-5300.2	-5299.8	-5307.6
Chow test (DF=4)*	***53.6***	***38.0***	***39.6***						
Observations (firms)		10890 (1594)			6913 (985)			3977 (1235)	

Elasticities

	Patents	Cite-weighted patents	Cite-wtd patents, excl. self-cites	Patents	Cite-weighted patents	Cite-wtd patents, excl. self-cites	Patents	Cite-weighted patents	Cite-wtd patents, excl. self-cites
R&D stock/assets	***0.298***	***0.297***	***0.298***	***0.295***	***0.292***	***0.292***	***0.309***	***0.312***	***0.312***
All pats/R&D stock	***0.013***	***0.028***	***0.023***	***0.013***	***0.029***	***0.027***	**0.018**	***0.027***	***0.021***
Software pats/R&D stock	***0.008***	***0.006***	***0.004***	0.002	0.003	0.002	***0.023***	***0.012***	***0.011***

*This is the likelihood ratio test for the hypothesis that the slopes are the same in the two subperiods.

Table 5
Market Valuation of Software Patents (combined definition)
Hardware and Software firms

	1980-1994				1995-1999			
	Hardware		Software		Hardware		Software	
R&D stock/assets	0.690	0.697	0.877	0.868	0.701	0.712	0.235	0.228
	(0.058)	(0.058)	(0.147)	(0.148)	(0.066)	(0.067)	(0.036)	(0.035)
Pats/R&D stock	0.064		-0.012		0.288		-0.004	
	(0.020)		(0.016)		(0.089)		(0.009)	
SW pats/R&D stock	0.343		1.178		4.283		3.048	
	(0.364)		(2.460)		(0.145)		(0.702)	
Cite-wtd pats excl. self-cites/ R&D stock		0.0099		0.0023		0.0220		-0.0003
		(0.0016)		(0.0030)		(0.0050)		(0.0009)
Cite-wtd SW pats excl. self-cites/ R&D stock		0.0079		0.1220		0.1078		0.1110
		(0.0128)		(0.0830)		(0.0800)		(0.0330)
No R&D	0.298	0.339	1.362	1.382	0.038	0.052	0.252	0.216
	(0.061)	(0.063)	(0.289)	(0.294)	(0.080)	(0.082)	(0.120)	(0.115)
Log likelihood	-7572.9	-7534.7	-1302.2	-1301.3	-3678.2	-3673.5	-1562.9	-1566.9
Observations (firms)	5989 (814)		924 (171)		2728 (786)		1249 (463)	

Elasticities

	Patents	Cite-wtd patents, x self-cites	Patents	Cite-wtd patents, x self-cites	Patents	Cite-wtd patents, x self-cites	Patents	Cite-wtd patents, x self-cites
R&D stock/assets	***0.286***	***0.282***	***0.415***	***0.408***	***0.335***	***0.338***	***0.293***	***0.292***
All pats/R&D stock	***0.016***	***0.025***	-0.001	0.002	***0.036***	***0.021***	0.000	0.000
Software pats/R&D stock	0.002	0.001	0.004	0.011	***0.018***	0.005	***0.032***	***0.019***

All regressions include a complete set of year dummies for each 2-digit sic (machinery, elec. machinery, instruments, telecomms, software).

The quantities shown are the average elasticity of the market value with respect to the variable in the first column. The significance tests are based on the corresponding coefficient and its heteroskedastic-consistent standard error.

Bold text indicates significance at the 5% level; *Bold italics significance at the 1% level.*