Chapter 9

The Use of Intellectual Property in Software: Implications for Open Innovation[1]

Stuart J.H. Graham
Georgia Institute of Technology
stuart.graham@mgt.gatech.edu

David C. Mowery
University of California, Berkeley
mowery@haas.berkeley.edu

To appear in *Open Innovation: Researching a New Paradigm* (Oxford, 2006)

Revised: October 26, 2005

1 Introduction

The sine qua non of the "open innovation" paradigm is the use by firms of both internal and external sources of knowledge in their quest for competitive advantage in the marketplace (Chesbrough, 2003a). The movement from the "closed" to the "open" innovation paradigm has spanned decades.[2] Chesbrough catalogues changes in firms' strategies during the 1980s and 1990s for sourcing knowledge and managing their innovation processes, with firms moving increasingly from a "closed" to an "open" paradigm. This shift to an "open" paradigm encompasses change in industry structure, firm strategy, and managers' perception and uses of intellectual property.

In this chapter we analyze management of intellectual property in the software industry during the 1980s and 1990s. Our analysis of the development of intellectual property protection in software highlights the tension between open innovation and changes in U.S. intellectual property policy that have strengthened the rights of patentholders and have raised the economic value of patents in many sectors, including software.

Software technology has been one of the main drivers of dynamism and controversy in the "new" intellectual property regime of the post-1980 period. Software is a good candidate for investigating how intellectual property rights (IPRs) have created an environment for "open innovation." First, software is an important element of the information technology (IT) "general purpose technology" that is used across many diverse economic sectors. Second, the introduction and adoption of software technologies and the growth of an industry to support and supply software products have occurred during a period of growing emphasis by corporations on open innovation. In fact, many of the firms that Chesbrough highlights as practitioners of "open innovation" are among the most significant users and producers of software technologies.[3]

Intellectual property protection creates a platform for the transfer of knowledge assets that spring from inventive activity. The external sourcing of knowledge is crucial to the operation of Chesbrough's system of "open innovation" and frequently relies on the "markets for intellectual capital" that may benefit from stronger formal intellectual property rights. Transactions in knowledge are complicated by the characteristics of knowledge, an intangible asset characterized by substantial uncertainty. Uncertainty over the boundaries and uses of knowledge assets limits the recognition by buyers and sellers of opportunities and increases transaction costs in the transfer of these assets. Transaction costs rise because heterogeneity in these assets is relatively high, their boundaries are often fuzzy, and the disclosure of their attributes is relatively difficult (Teece, 2000). IPRs often support the development of markets for the sale and licensing of knowledge.

Computer software is a technological field in which patenting, particularly in the United States, has grown rapidly in recent years (Graham and Mowery 2003) despite the fact that the underlying "technology" is several decades old. As we note below, much of this increased patenting reflects changes in industry structure and in U.S. policy toward software patents that have increased the economic value of patents. This chapter builds on our earlier studies of software patenting (Graham and Mowery 2003; Graham, Hall, Harhoff, and Mowery 2003; Graham and Mowery 2005) that analyzed trends in such patenting during the 1980s and 1990s. With our analysis in this chapter, we hope to shed additional light on the causes of the growth of U.S. software patenting while investigating the environment in which "open innovation" occurs in this and other sectors.

2 Development of the U.S. Software Industry: Vertical Disintegration

The growth of the global computer software industry has been marked by at least 4 distinct eras spanning the post-1945 period (Graham and Mowery, 2003). During the early years of the first era (1945-65), covering the development and early commercialization of the computer, software as it is currently known did not exist. Even after the development of the concept of a stored program, software was largely custom-developed for individual computers. During the 1950s, however, the commercialization and widespread adoption of "standard" computer architectures supported the emergence of software that could operate on more than one type of computer or in more than one computer installation. In the United States, the development of the IBM 650, followed by the even more dominant IBM 360, provided a large market for standard operating systems and application programs. The emergence of a large installed base of a single mainframe architecture occurred first and to the greatest extent in the United States. During this period, however, most of the software for early mainframe computers was produced by their manufacturers and users.

During the second era (1965-78), independent software vendors (ISVs) began to appear. During the late 1960s, producers of mainframe computers "unbundled" their software product offerings from their hardware products, separating the pricing and distribution of hardware and software. This development provided opportunities for entry by independent producers of standard and custom operating systems, as well as independent suppliers of applications software for mainframes. The changing structure of the industry created a supply of innovation from newer, smaller entrants. Unbundling occurred first in the United States and has progressed further in the United States and Western Europe than in the Japanese software industry, and is consistent with Chesbrough's description of an increasingly disintegrated innovation environment.

Although independent suppliers of software began to enter in significant numbers in the early 1970s, computer manufacturers and users remained important sources of both custom and standard software in Japan, Western Europe, and the United States during this period. Some computer "service bureaus" that had provided users with operating services and programming solutions began to unbundle their services from their software, providing yet another cohort of entrants into the independent development and supply of traded software. Sophisticated users of computer systems, especially users of mainframe computers, also created solutions for their applications and operating system needs. A number of leading suppliers of traded software in Japan, Western Europe, and the United States were founded by computer specialists formerly employed by major mainframe users.

During the third era (1978-93), the development and diffusion of the desktop computer produced explosive growth in the traded software industry. Once again, the United States was the "first mover" in this transformation, and the U.S. market quickly emerged as the largest single one for such packaged software. Rapid adoption of the desktop computer in the United States supported the early emergence of a few "dominant designs" in desktop computer architecture, creating the first mass market for packaged software. The independent vendors that entered the desktop software industry in the United States were largely new to the industry. Few of the major suppliers of desktop software came from the ranks of the leading independent producers of mainframe and minicomputer software, and mainframe and minicomputer ISVs are still minor factors in desktop software.

Rapid diffusion of low-cost desktop computer hardware, combined with the emergence of a few "dominant designs" for this architecture, eroded vertical integration between hardware and software producers and opened up great opportunities for ISVs in the marketplace, and supported a distributed supply base of innovation in software technologies. Declines in the costs of computing technology have continually expanded the array of potential applications for computers and supported a vibrant environment for innovation; many of these applications rely on software solutions for their realization. A growing installed base of ever-cheaper computers has been an important source of dynamism and entry into the traded software industry, because the expansion of market niches in applications has outrun the ability of established computer manufacturers and major producers of packaged software to supply them.[4]

The packaged computer software industry now has a cost structure that resembles that of the publishing and entertainment industries much more than that of custom software--the returns to a product that is a "hit" are enormous and production costs are low. And like these other industries, the growth of a mass market for software elevated the importance of formal intellectual property rights, especially copyright and patent protection.

An important contrast between software and the publishing and entertainment industries, however, is the importance of product standards and consumption externalities in the software market. Users in the mass software market often resist switching among operating systems or well-established applications because of the high costs of learning new skills, as well as their concern over the availability of an abundant library of applications software that complements an operating system. These switching costs typically are higher for the users who dominate mass markets for software and support the development of "bandwagons" and the creation through market forces of product standards. As the widespread adoption of desktop computers created a mass market for software during the 1980s, these de facto product standards in hardware and software became even more important to the commercial fortunes of software producers than was true during the 1960s and 1970. In some instances strong intellectual property protection aided firms in establishing control of a proprietary standard (Simcoe, Chapter 8), further raising the value of formal instruments for IP protection.

The fourth era in the development of the software industry (1994-present) has been dominated by the growth of networking among desktop computers, both within enterprises through local area networks linked to a server and among millions of users through the Internet. Networking has opened opportunities for the emergence of new software market segments (for example, the operating system software that is currently installed in desktop computers may reside on the network or the server), the emergence of new "dominant designs," and potentially, the erosion of currently dominant software firms' positions. Some network applications that are growing rapidly, such as the Worldwide Web, use code (html) that operates equally effectively on all platforms, rather than being "locked into" a single architecture. Like the previous eras of this industry's development, the growth of network users and applications has been more rapid in the United States than in other industrial economies, and U.S. firms have maintained dominant positions in these markets (See Mowery and Simcoe, 2002b).

The Internet has provided a new impetus to the diffusion and rapid growth of open source software. Although so-called "shareware" has been an important form of software in all of the eras of the software industry, the Internet's ability to support rapid, low-cost distribution of new software and the centralized collection and incorporation into that software of improvements from users has made possible such widely used software as Linux and Apache (See Kuan 2001; Lerner and Tirole 2002; and Lee and Cole 2003).

This brief overview of the development of the U.S. software industry's structure illustrates the ways in which the emergence of independent software vendors and the growth of "mass markets" for standard (packaged) software, have elevated the importance of formal intellectual property rights for firms in the industry. At the same time, however, change in the policy regime governing overall U.S. intellectual property protection, as well as shifts in the coverage of software by such formal instruments for intellectual property protection as patents and copyright, have important implications for the future evolution of industry structure. The next section describes the evolution of this formal IPR regime for software.

3 The evolution of U.S. patent protection for software

This section and the next present a descriptive summary of the growth of software patenting during the 1987-2003 period. Although this discussion focuses on patents, other forms of intellectual property protection, including copyright,[5] trade secret,[6] and trademark[7] remain important in software.

Federal court decisions during the past decades have consistently broadened and strengthened the economic value of software patents.[8] Although early cases during the 1970s supported the stance of the U.S. Patent and Trademark Office (USPTO) in stating that software algorithms were not patentable,[9] the landscape has significantly shifted since then (Samuelson, 1990).[10] In the cases of *Diamond v. Diehr*[11] and *Diamond v. Bradley,*[12] both decided in 1981, the Supreme Court announced a liberal rule that permitted the patenting of software algorithms. Both the courts and the U.S. Patent and Trademark Office have supported this policy, strengthening patent protection for software (Merges, 1996). A vivid example of the effects of this stronger patent regime is the 1994 court decision that found Microsoft guilty of patent infringement and awarded $120 million in damages to Stac Electronics, the plaintiff. The damages award was hardly a crippling blow to Microsoft, but the firm's infringing product had to be withdrawn from the market temporarily, compounding the financial and commercial consequences of the decision (Merges, 1996). Sony Corporation faced a similar ruling in 2005 affecting its Playstation user-control joysticks, after a U.S. federal court determined that Sony had infringed two patents held by Immersion Corporation. Sony was ordered to pay $91 million and cease U.S. sales of its Playstation consoles and 47 of its software titles (Williams, 2005).

4 Patenting trends in the U.S. software industry, 1987-2003

In order to better illuminate the intellectual property environment in which "open innovation" in software occurs, we examine trends in U.S. software patenting during 1987-2003, focusing on the product areas we believe have been most affected by changes in the legal treatment of software patents—packaged software. As our previous work emphasized (Graham and Mowery, 2003), no widely accepted definition of "software patent" exists, and the problem of identifying software patents is not made easier by USPTO changes in patent classification schemes. Other researchers have chosen to define a "software patent" by reference to certain key words in the patent disclosure (Bessen and Hunt, 2004), but we rely instead on the classification decisions of USPTO patent examiners. Empirical support for the comparative accuracy of our "classification" method is forthcoming (Layne-Farrar, 2005).

One difficulty that arises when using these two patent office classifications is the rapid growth in the number of software-related USPTO patents during the period of this analysis. Because we are interested in analyzing changes over time in the number of software patents, we seek to insulate our sample from any "reclassifications" of patents from "all other" to a "software-related" category. Such an analysis is made more difficult by the USPTO's unannounced reclassifications. In order to eliminate the impact of reclassifications, we use a "snapshot" in time of the U.S. classifications for U.S.-issued patents. All patents in our analysis reflect their classifications on the same date.

4.1 Method for Defining a "Software" Patent

Using the USPTO's "Cassis" database, we collected data on all patents issued in the U.S. 1987-2003, relying on the USPTO classification scheme as of December 31, 2003, with patents issued on previous dates updated to reflect current USPTO classifications, thus allowing us to compare trends in patenting over time. These classes, disclosed in Table 1, were identified by examining U.S. patenting in the years 1987-2003 by the six largest U.S. producers of personal computer software, based on their calendar 2000 revenues as reported by Softletter (2001).[13] These 12 USPTO classes account for 67.9% of the patenting of the "top 6" firms, while these same 6 firms account for 88.4% of the patenting of the Softletter "top 100." Table 9.1 contains data on the distribution of the patents in this sample among these 12 USPTO classes.[14]

Table 9.1: Patenting by the Softletter 100 (2001), by USPTO patent class, 1987-2003 (total patents=3,891)

U.S. Patent Class	Patent Count	Share of All Firm Patents	Cumulative Total	U.S. Patent Class	Patent Count	Share of All Firm Patents	Cumulative Total
345	730	18.8%	18.8%	717	118	3.0%	58.0%
707	624	16.0%	34.8%	714	85	2.2%	60.2%
709	363	9.3%	44.1%	711	80	2.1%	62.3%
382	157	4.0%	48.2%	710	59	1.5%	63.8%
713	141	3.6%	51.8%	358	52	1.3%	65.1%
704	125	3.2%	55.0%	715	25	0.6%	65.8%

Our definitional scheme does not cover all software patents, but it does provide longitudinal coverage of a particularly dynamic and important segment of the overall software industry, inasmuch as IDC estimated that global packaged software revenues amounted to $179 billion in 2004.[15] The data in Figure 9.1 indicate that the share of all U.S. patents accounted for by software patents grew from 2.1% to 7.4% of all issued U.S. patents between 1987 and 1998, and the share of patents in these 12 U.S. classes has remained between 6.9 and 7.5% of overall patenting during 1999-2003. This slowing in the rate of growth in "software" patenting as a share of total U.S. patenting occurs in virtually every USPTO class included in our definition of software patents, and may reflect the effects of the post-2001 downturn in the IT industry.[16]

FIGURE 9.1 HERE

There are several potential explanations for the slowdown in software patent growth, relative to overall U.S. patenting, after 1999. In other work (Graham and Mowery, 2004), we noted that 1995 changes in the legal patent term of protection (changing the term of protection from 17 years from date of application to 20 years from date of issue) created strong incentives for patent applicants to pursue "continuations" in their applications, which (among other advantages) enabled applicants to extend the length of application secrecy. Because we showed that software-patent applicants made extensive use of continuations, it is possible that the 1995 changes in patent term may have reduced incentives for software inventors to seek patents. Moreover, the negative effects on applicants' incentives to pursue continuations have been intensified by more recent requirements to publish many patents within 18 months of application.

It is also possible that the accumulation of experience by USPTO examiners in dealing with software-patent applications, as well as the expanding body of patent-based prior art on which examiners rely in part, have led to lower rates of issue for software patent applications. The fluctuations in growth in software patents, however, do not appear to be associated with fluctuations in the "pendency" of patent applications (the length of time required to review and grant or deny patent applications), since the average pendency of applications for issued software patents, which is greater than the average for all patents, has increased steadily through the 1995-2003 period.[17]

4.2 Software-related patenting by packaged software and electronic systems firms, 1987-2003.

In this section, we analyze patenting by U.S. software firms during 1987-2003, focusing on leading U.S. packaged software firms identified by Softletter in their 2001 tabulation of the 100 largest U.S. packaged software firms (based on revenues). We focus on these firms because, unlike the electronics systems firms, inventive output is more likely to be purely software-related. Figure 9.2 displays trends during 1987-2003 in the share of all U.S. software patents held by the 100 largest U.S. packaged software firms, comparing trends that both include and exclude the largest player in the industry, Microsoft. Figure 9.2 demonstrates that these firms increased their share of overall software patenting during the 1987-2003 period, from less than .06% in of all software patents in 1988 to nearly 4.75% in 2002, declining to 4.13% of software patents in 2003. Eliminating Microsoft from the figure reveals more modest growth, with shares growing from less than .06% in 1987 to 1.35% in 2000 and declining to 1.0% in 2003. Similarly to Figure 9.1, the data in Figure 9.2 suggest rapid growth in software patenting through the late 1990s, followed by no growth or declines after 2000.

FIGURE 9.2 HERE

Although patenting by large packaged-software firms has grown since the late 1980s, it is interesting and surprising to note that electronic systems firms account for a larger share of software patenting as we define it. Both our USPTO and IPC classification methods show that the share of overall "software" patents accounted for by large electronic systems firms (IBM, Intel, Hewlett-Packard, Motorola, National Semiconductor, NEC, Digital Equipment Corporation, Compaq, Hitachi, Fujitsu, Texas Instruments, and Toshiba) considerably exceeds the share of "software" patents assigned to specialist packaged-software firms. Our data analysis demonstrates that the share of "software" patents assigned to our sample of 12 "electronics systems" firms fluctuates between a low of 21% in 1990 and a high of 28% in 1994 before falling to 21-23% of all software patenting for 1998-2003.[18]

We calculate the share of all patents issued to these firms that we classify as "software" patents during 1987-2003. Figure 9.3 displays the time trend for the share of these patents within these 12 firms' patent portfolios during 1987-2003. Software patents' share of overall firm patents increases during the 1987-2003 period for all of these firms, from roughly 14% in 1987 to 25% of their overall patent portfolios by 2003. Even more striking, however, is the level and growth of software patenting by IBM, the largest U.S. patenter,[19] which increases its software patenting from 27% of its overall patenting in 1987 to 42% in 2003. In contrast to the software patenting of the other eleven systems firms, IBM's share of "software patents" in its annual patenting increases through 2003.

FIGURE 9.3 HERE

Inasmuch as electronic systems firms appear to account for a larger share of patenting during the 1987-2003 period than do packaged-software specialists, a comparison of patenting propensities between systems and software-specialist firms would be very interesting. Unfortunately, the absence of detailed line-of-business reporting of their R&D investments means that we have data on software-related R&D spending for only one of the 12 systems firms included in Figure 9.3, IBM.

FIGURE 9.4 HERE

Figure 9.4 compares the patent propensities of IBM and Microsoft for the 1992-2002 period. The figure is presented on a log scale, and shows that IBM's software patenting per software R&D dollar spent is substantially greater than Microsoft's, dominating Microsoft's propensity by a factor approaching or exceeding an order of magnitude (a factor of 10) in every 3-year interval. Furthermore, Microsoft's patent propensity has "plateaued" at 0.10-0.12 patents per $100 million during the 1996-2003 period, but IBM's has continued to grow, climbing from 0.7 patents per $100 million R&D during 1997-1999 to nearly 1.0 patents per $100 million R&D during 2001-2003.[20] Some of the reported growth in IBM's patent/R&D ratio reflects shrinkage in the firm's reported software R&D budget during 1997-2002, a period of growth for Microsoft R&D investments. Nevertheless, the figure suggests considerable contrast between the patenting behavior of the largest packaged-software specialist and the largest software producer among U.S. electronic systems firms.

In earlier work (Graham and Mowery 2003), we showed that the growth in software patenting by both packaged-software "specialists" and electronics system firms in the United States was associated with a decline in the use of copyright protection for software. Is this growth in software patenting that we document consistent with a shift toward "open innovation" in this technology? IBM's release of over 500 patents to the "open source" community (discussed below) suggests that patents can support the creation of an intellectual property "bazaar" that itself advances open-source software development, although the ultimate significance of IBM's recent initiative remains to be seen.

There are several potential explanations for the rapid growth in software patenting during this period. First, software patenting may have grown along with overall patenting in the United States simply because the returns to investment in innovation had increased or because of the broader strengthening of patentee rights that resulted from Congressional actions and judicial decisions during the 1980s and 1990s (Kortum and Lerner 1999). But this explanation cannot account for the fact that software patenting grew as a share of overall U.S. patenting during this period, more than doubling its share from 1.7% in 1987 to 3.9% in 1997 (Graham and Mowery 2003).

Another explanation for the growth in software patenting argues that increased patenting, especially by large firms such as Microsoft and IBM, reflects the growing importance of "defensive patenting" in software. Competing firms may seek patents less to support the commercial development of specific invention than as a means of avoiding costly litigation (See Hall and Ziedonis 2001 for a discussion of "defensive patenting" in semiconductors). "Defensive patenters" apply for a large number of patents for exchange in cross-licensing agreements, thus preserving their freedom to innovate.

Growth in software patents alternatively might reflect a decline in the rigor of USPTO review of the increased number of software-related patent applications that followed the changes in the legal treatment of software patents during the 1980s. Lacking patent-based prior art to guide their evaluation of a much larger flow of applications, USPTO examiners may issue low-quality patents. Such explanations suggest that the "quality" of software patents should have declined during the 1980s and 1990s, reflected in declining rates of citation to these patents in subsequent patents. But the "defensive patenting" explanation predicts that the patents assigned to large software firms should exhibit particularly significant declines in quality, whereas the "weakened review" explanation predicts an across-the-board decline in the quality of all issued software patents. In fact, evidence presented by us elsewhere (Graham and Mowery,

2005) cast doubt on any such fall in patent software quality, at least in the patents issuing to software-specialist firms and electronics systems firms.

4.3 Open Source Software and "Open Innovation"

Because in this chapter we focus on software, we are able to examine another mechanism by which firms are accessing external knowledge: open source. While other chapters in this book examine external pathways exploited by firms' open innovation strategies (Fabrizio, Chapter 7; Simcoe, Chapter 8), the "open source" mechanism has not been covered in detail. The "open source" development model--essentially one in which developers are liberated to access and build upon the efforts of others--is being increasingly exploited by leading firms in their quest to "open" corporate innovation.

As an exemplar, consider International Business Machines Corp. (IBM). The giant computer systems, software, and services firm has taken a proactive stance toward the "open source" model, both creating chief officers with responsibility over the firm's open source strategies, and investing in "open source" research centers around the globe. In April, 2005, IBM's Vice President of Worldwide LINUX Business Strategy summed up the company's use of "open source" in its larger open innovation strategy: "Linux is not really about being free, it's about freedom--freedom to collaborate and innovate" (Kerner, 2005). IBM has responded with investments, creating in Bangalore, India a Linux Solution Centre (one of 7 worldwide) and an IBM Linux Competency Centre (one of 4 in Asia). Moreover, in January 2005 IBM released 500 of its patents to the open source community, allowing software engineers to freely use the ideas embodied in the patents without paying royalties to the company.

The finding that IBM is patenting more heavily within software relative to its R&D investment, while simultaneously "opening up" a portion of its software-related patent portfolio, raises some questions about the applicability to IBM of the open innovation framework. Alternatively, the role of patent strategy within an "open innovation" strategy remains to be developed. It is likely that these IBM patents are not among the firm's most valuable software patents, although that hypothesis remains to be explored. Nevertheless, IBM's recent action suggests that patents are not incompatible with open-source software development, although the recent litigation between IBM and Santa Cruz Operation (SCO) centered on claims that IBM's use of "open source" software had run afoul of pre-existing IPRs held by SCO.

As open-source software has sparked increased interest by developers, firms, and academics alike, a recurrent theme is the need to limit the strength and improve the clarity of proprietary IPRs in order to ensure open access and design freedom. The open-source development model is undermined when the development community is blocked from using "fountainhead" innovations, and when developers are uncertain about the extent or strength of protection of such key innovations. Indeed, the software innovation process may be unusually sensitive to IP roadblocks due to software's character as a "cumulative innovation" technology, meaning that innovation is closely linked to and builds upon prior generations of the technology. Moreover, the attractiveness for would-be adopters of open-source software could be severely reduced if the open-source code, once adopted, were subject to threats of litigation from third-party owners of IPRs implicated in the open-source software. Moreover, the lack of procedures within the U.S.

patent system for administrative (as opposed to litigation-based) procedures for challenging the validity of patents once issued means that the quality of many software patents is uncertain, which can have a chilling effect on both development and deployment of open-source software (OSS) innovations.

Open-source licenses create a legal relationship between the creator of the software and its voluntary users, but the open-source license cannot preclude the existence of various IPRs in the software. The successful General Public License (GPL or "copy left") employed by the Free Software Foundation rewarded developers' collaboration while limiting the disincentives created by commercial expropriation (Lerner and Tirole 2002). Nevertheless, the GPL binds only those parties to the agreement—it does not apply to innovators who are not party to the GPL or have developed the same technologies independently. GPL-compliant users may relinquish certain IP rights in their derivative works under the terms of the license, but property rights, insofar as they define a relationship between the property holder and the world, cannot be eliminated by a bilateral license. In fact, the restrictive character of IPRs create the foundation for the operation of the open-source license: copyright and patent rights, to the extent that the latter have been sought, are held by the inventor of the open-source software who then "passes" these rights on to other developers, allowing these voluntary adopters to use the rights under the terms of the open-source license.

The innovation environment in software is complicated by significant variation among open-source licenses in the treatment of IPRs to works derived from the original open-source software. Licenses run the gamut from restrictive to permissive in their accommodation of creation by adopters of IP rights in derivative innovations. The popular General Public License (GPL) is relatively restrictive, reflecting the license document's expressions of hostility to software patents in its preamble.[21] The terms of the GPL largely ignore software patents, however, with the exception of restrictions in Section 7 that prohibit the distribution of any OSS subject to a patent infringement action initiated by a third party or a court order.[22] By contrast, the license offered under the Berkeley Software Distribution (BSD) is a bare-bones OSS license, including terms that require a notice of copyright and disclaimer of warranties (U.C. Regents, 1999), but otherwise allowing the commercialization of derivative works with no restriction on patent rights *per se* (Simon 2003). An "intermediate" variant is the Mozilla Public License (MPL), more permissive than the GPL but more restrictive than the BSD license in its treatment of IP rights. The MPL treats the patent rights of the originators explicitly in the terms of the license,[23] recognizing that the licensing of patent rights to complementary or build-on propriety applications may be necessary.[24] The MPL has been used by at least one software company to support "taking the code private" into its own proprietary software.[25] While "taking open-source code private" strikes at the heart of the bargain that OSS adopters make with the open-source community, the texts of the GPL, MPL, and BSD license demonstrate that proprietary innovations arising from or built upon the core OS software are nevertheless anticipated.

This variation in the legal treatment of "open source" innovations demonstrates the uncertain environment in which development under this model occurs. For software firms engaged in the type of "open innovation" that Chesbrough describes, stronger patent rights may have offsetting effects. On the one hand, strong IPRs may create more efficient markets for intellectual property, thereby facilitating the purchase by firms of innovations from external sources. At the same time, however, strong intellectual property rights may impede firms' access to "open-source" software innovations from software developers outside the firm.

5 Conclusion

Spurred by favorable judicial decisions, software patenting has grown significantly in the United States since the 1980s, although the available data suggest that growth in software patents' share of overall U.S. patenting has slowed since approximately 2000. Scholars have produced little evidence to suggest that increased patenting has been associated with higher levels of innovation in the U.S. software industry, although virtually no evidence has likewise been raised to suggest that increased patenting has proven harmful to innovation in this important sector of the "post-industrial" economy. The vertically specialized structure of the U.S. software industry, populated by firms specializing in software only, is a dramatic shift from the vertically integrated structure that characterized the U.S. and global computer industries in the 1960s. But stronger patent protection for software emerged in the 1980s, well after the transformation of this industry structure that began in the late 1960s. The links between stronger formal protection for intellectual property in this industry and the development of its vertically specialized structure thus are weak. In this sector, the connections between the increasing proliferation of innovators suggested by Chesbrough and the role of patents as transactional mechanisms requires further study.

Electronic systems firms appear to account for a larger share of overall software patenting, in our definition, than do the packaged-software specialist firms during the 1987-2003 period. It is possible, although we have no direct evidence to support this argument, that systems firms are patenting their software-related intellectual property for strategic reasons, e.g., to support complex cross-licensing agreements similar to those in the semiconductor industry that are discussed in Hall and Ziedonis (2001). There is less evidence of such cross-licensing agreements among software specialists, although the recent agreement between Microsoft and Sun Microsystems (Guth and Clark, 2004) provides one such example. As Hall and Ziedonis note, much of the cross-licensing that provides incentives for extensive patenting by firms is motivated by the prospect or the reality of litigation. Evidence from software patent litigation cited in Graham (2004) indicates that packaged-software specialist firms account for a small fraction of software patent litigation, by comparison with computer hardware firms and firms from a diverse array of other industries. Thus, the strategic motives of firms' patenting, the function of defensive cross-licensing and litigation, and the role these play in Chesbrough's observations of changing innovation remain open questions.

Chesbrough's "open innovation" paradigm raises many questions for researchers, including the manner and mechanisms of structural change in industries, and the role played by the transactional environment for knowledge. This chapter offered both a case study in the development of the software industry, and an analysis of patenting in an important source of innovation in the new economy, software. With it, we intended to offer a view into the system of innovation in one important sector as a means of raising questions about the generalizability of Chesbrough's "open innovation" paradigm. We leave the development of these to further research, and researchers.

6 References

Bessen, James, and Robert M. Hunt, 2004. An Empirical Look at Software Patents, Federal Reserve Bank of Philadelphia, Working Paper 03-17, http://ideas.repec.org/p/fip/fedpwp/03-17.html

Bresnahan, Timothy, and Shane Greenstein, 1996. "The Competitive Crash in Large Scale Commercial Computing." in (eds) R. Landau, T. Taylor, and G. Wright, *The Mosaic of Economic Growth*. Stanford, CA: Stanford University Press.

Chesbrough, Henry, 2003a. *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Boston: Harvard Business School Press

Child, Simon., 2003. "Open Source and Intellectual Property." MyITAdvisor. National Computing Center, March 22.

Free Software Foundation. 1991. "GNU General Public License Version 2," http://www.fsf.org/licensing/licenses/gpl.txt

Graham, Stuart J.H., 2004. Continuation, Complementarity, and Capturing Value: Three Studies Exploring Firms' Complementary Uses of Appropriability Mechanisms in Technological Innovation, unpublished Ph.D. dissertation, Haas School of Business, U.C. Berkeley.

Graham, Stuart J.H. and David C. Mowery, 2005. "Software Patents: Good News or Bad News?." In Robert W. Hahn, ed., *Intellectual Property Rights in Frontier Industries: Software and Biotechnology,* Washington: AEI-Brookings Joint Center.

Graham, Stuart J.H., and David C. Mowery, 2004. "Submarines in Software? Continuations in U.S. Software Patenting in the 1980s and 1990s," *Economics of Innovation and New Technology*, 13/5: 443-456.

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

Graham, Stuart J.H., Bronwyn H. Hall, Dietmar Harhoff, and David C. Mowery, 2003. "Post-issue Patent 'Quality Control:' A Comparative Study of U.S. Patent Reexaminations and European Patent Oppositions." In Wesley M. Cohen and Stephen A. Merrill, eds., *Patents in the Knowledge-Based Economy.* Washington, D.C.: National Academies Press.

Guth, Robert A., and Don Clark, 2004. Behind Secret Settlement Talks: New Power of Tech Customers, *Wall Street Journal,* April 5, p. 1.

Hall, Bronwyn H. and Rosemarie Ham Ziedonis. 2001. "The Determinants of Patenting in the U.S. Semiconductor Industry, 1980-1994." *Rand Journal of Economics,* 32/1: 101-128.

Kerner, Sean Michael, 2005. "IBM: Open Source is more than just Linux," *InternetNews*, April 20.

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

Kuan, Jennifer W. 2001. "Open Source Software as Consumer Integration into Production." SSRN working paper 259648.

Layne-Farrar, Anne, 2005. "Defining Software Patents: A Research Field Guide." AEI-Brookings Joint Center Working Paper 05-14, http://aei-brookings.org/admin/authorpdfs/page.php?id=1177

Lerner, Josh and Jean Tirole, 2002. "Some Simple Economics of Open Source." *Journal of Industrial Economics* 50(2), pp. 197-234.

Lee, Gwendolyn K. and Robert E. Cole, 2003. "From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development." Organization Science 14(6), pp. 633-649.

LizardTech. 2001. "LizardTech's Statement on DjVu Open Source Licensing" http://djvulibre.djvuzone.org/lti-licensing.html

Merges, Robert P., 1996. "A Comparative Look at Intellectual Property Rights and the Software Industry." In David C. Mowery, ed., *The International Computer Software Industry: A Comparative Study of Industry Evolution and Structure*. New York: Oxford University Press.

Mowery, David C., and Timothy Simcoe, 2002b. The Origins and Evolution of the Internet, in B. Steil, R. Nelson and D. Victor, eds., *Technological Innovation and National Economic Performance* (Princeton, NJ: Princeton University Press).

Mozilla.org. 1999. "Mozilla Public License Version 1.1", http://www.mozilla.org/MPL/MPL-1.1.html

Samuelson, Pamela, 1990. "Benson Revisited: The Case Against Patent Protection for Algorithms and Other Computer Program-Related Inventions." *Emory Law Journal,* 39, pp. 1025-1122.

Child, Simon, 2003. "Open Source and Intellectual Property." *MyITAdvisor*. National Computing Center, March 22.

Softletter, 2001. The 2001 Softletter 100. *Softletter Trends & Strategies in Software Publishing,* 3, pp. 1-8.

Software & Information Industry Alliance, 2005. "Packaged Software Industry Revenue and Growth." www.siia.net/software/pubs/growth_software05.pdf

Teece, David J., 2000. *Managing Intellectual Capital.* New York: Oxford University Press.

U.C. Regents. 1999. "The BSD License," http://www.opensource.org/licenses/bsd-license.php

U.S. Patent and Trademark Office, 2002. "Patenting by Organizations, 2001." Report. http://www.uspto.gov/go/taf/topo_01.pdf

Williams, Martyn, 2005. "Sony loses Playstation patent case, must pay $91 million." *PC World,* March.

Figure 1: Software patent's share of all issued U.S. patents, 1987-2003

(Comparing two definitions: U.S. Classification and International Patent Classification)

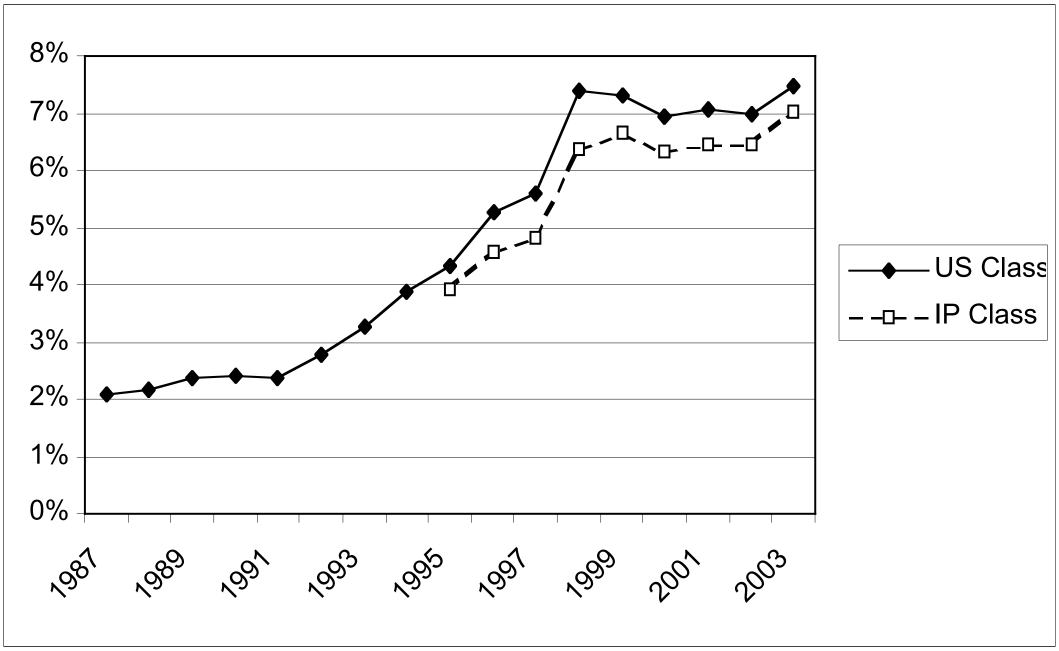


Figure 2: Large packaged-software firms' software patents, as a share of all US issued software patents, 1987-2003

(Comparison: US-class defined "software" patenting by 100 largest "packaged-software" firms as share of all "software" patents issued, *including and excluding* Microsoft)

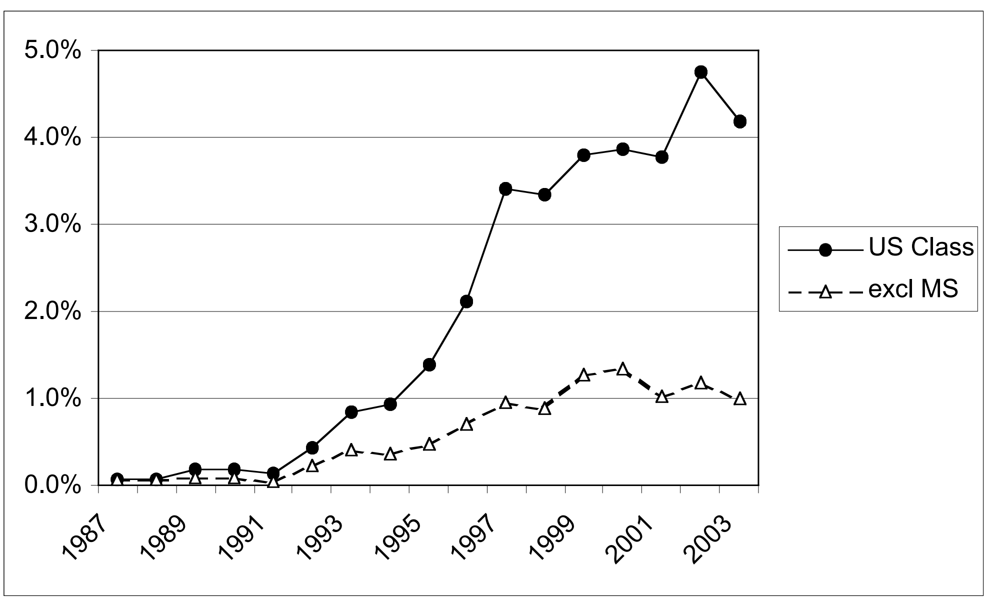


Figure 3: Large systems firms ' software patents, as a share of firm patents, 1995-2003

(Weighted Average: Comparing US -class defined "software" patenting by IBM, Intel, Hewlett-Packard, Motorolla, National Semiconductor, NEC, Digital Equipment, Compaq Computer, Hitachi, Fujitsu, Texas Instruments, and Toshiba)

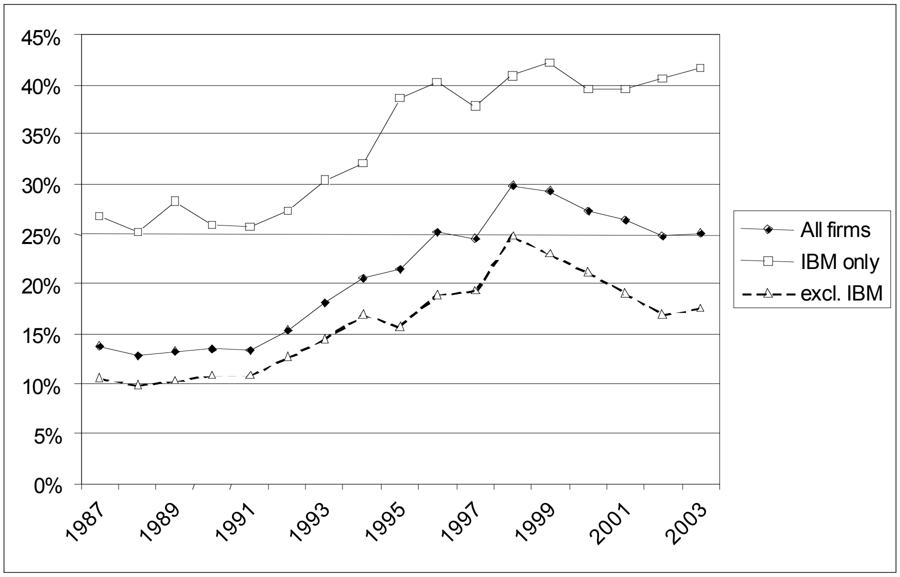
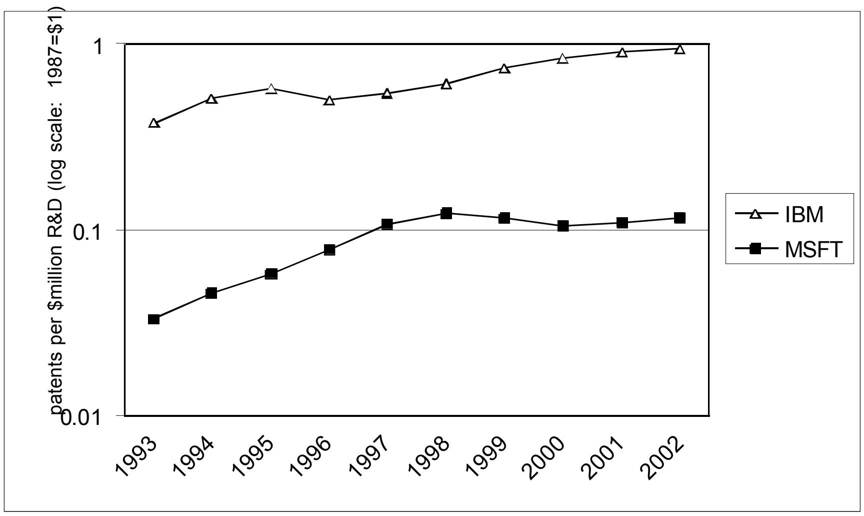


Figure 4: Comparison of IBM and Microsoft 's software patent propensity, firms ' "software" patents per "software" R&D expenditures, 1987-2002

(3-year moving averages; Patenting limited to each firms ' defined software patents)



7 End Notes

[1] We acknowledge the helpful comments made on an earlier version by conferees at the American Enterprise Institute/Brookings Joint Center for Regulatory Studies "Intellectual Property Rights in Frontier Industries" Conference, held April 30, 2004, with special thanks to Starling Hunter. Research for this chapter was supported by the National Research Council, the Andrew W. Mellon Foundation, the Alfred P. Sloan Foundation, the Kauffman Foundation, and the Tokyo Foundation.

[2] Chesbrough suggests that the 1970s were characterized by a closed paradigm (p. 45) while the 1980s and 1990s were decades of change in which the open innovation paradigm was evolving (p. 45-49).

[3] Chesbrough highlights Xerox, Intel, Lucent, and IBM as adopters of the "open" innovation paradigm.

[4] Bresnahan and Greenstein (1995) point out that a similar erosion of multiproduct economies of scope appears to have occurred among computer hardware manufacturers with the introduction of the microcomputer.

[5] Copyright offers protection from the moment of authorship, and remains an important protection for software "writings." *Computer Associates Int'l v. Altai, Inc.,* 982 F.2d 693 (2d Cir. 1992); *Whelan Associates v. Jaslow Dental Laboratory,* 797 F.2d 1222 (3rd Cir. 1986).

[6] A trade secret is formally some information used in a business which, when secret, gives one an advantage over competitors. The secret must be both novel and valuable.

Metallurgical Industries, Inc. v. Fourtek, Inc. 790 F.2d 1195 (1986).

[7] A trademark protects names, words, and symbols used to identify or distinguish goods and to identify the producer. *Zatrains, Inc. v. Oak Grove Smokehouse, Inc.* 698 F.2d 786 (5th Cir., 1983).

[8] The European Union currently is debating the extent to which "software" will be afforded patent protection, an issue that has been reasonably well-settled in the United States since the mid-1980s.

[9] *Gottschalk v. Benson,* 409 U.S. 63 (1972).

[10] Samuelson (1990) argues that the Patent and Trademark Office was at odds with the Court of Customs and Patent Appeals (CCPA) throughout the 1970s over the patentability of software and concludes that the CCPA's views in favor of patentability ultimately triumphed.

[11] 450 U.S. 175 (1981).

[12] 450 U.S. 381 (1981).

[13] Reported in *Softletter* (2001) this group of 100 companies includes Microsoft, Novell, Adobe Systems, Autodesk, Intuit, and Symantec. We are grateful to *Softletter* for permission to use these data. The 2001 tabulation was the last year to date in which *Softletter* produced this report.

[14] We also constructed a sample of software patents that was based on the International Patent Classes associated with patenting by large packaged software specialists during this period of time. Overall, the IPC-based definition yields similar results and trends,

and we omit discussion of it because of space limitations.

[15] Estimated by International Data Corp. and reported in Software & Information Industry Alliance (2005). In the U.S., 2003 software-only industry revenues were surveyed by the Census Bureau to be $70 billion, with an average 12% growth during the 1990s, and employment growth showing 5%. *Ibid.*

[16] The only class that did not show a reasonably flat growth trajectory after 1998 was the IPC group “G06F 17” which is the newest “software” class, and showed steady growth as a share of all patenting 1995-2003.

[17] Pendency for software applications at issue for all software patents compared to all non-software patents from 1995-2003 are:

Year	1995	1996	1997	1998	1999	2000	2001	2002	2003
Pendency	130%	127%	122%	120%	122%	128%	136%	145%	146%

[18] Data analysis using the IPC-definition method found substantially the same trend in these firms’ patenting shares.

[19] The U.S. Patent and Trademark Office (2002) shows that IBM (International Business Machine) is the largest patenting organization in the U.S., having been assigned 3,411 and 2,886 patents in 2001 and 2000, respectively. These numbers were 75% and 43% larger, respectively, than the next largest patentholder in these years (NEC).

[20] Graham and Mowery (2003) point out two problems with these data in comparing the “software patent propensities” of IBM and Microsoft: The R&D data reported by these two firms may not be strictly comparable, since a portion of Microsoft’s total

reported R&D investment may cover some fixed costs of maintaining an R&D facility that are not included in IBM's reported software-related R&D investment (Although IBM does maintain "software only" research facilities around the globe; Its Bangalore, India LINUX facilities are but one example). In addition, an unknown portion of Microsoft's reported R&D spending includes development programs for hardware, and these data therefore may understate the Microsoft software-related patent propensity and overstate that for IBM.

[21] Free Software Foundation (1991), Preamble: "[A]ny free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all."

[22] Free Software Foundation (1991), Section 7: "If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy

both it and this License would be to refrain entirely from distribution of the Program."

[23] Mozilla.org (1999), Section 2.1 "The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims … (b) under Patents Claims infringed by the making, using or selling of Original Code, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Code (or portions thereof)."

[24] Mozilla.org (1999), Section 3.4(b) "If Contributor's Modifications include an application programming interface and Contributor has knowledge of patent licenses which are reasonably necessary to implement that API, Contributor must also include this information in the LEGAL file."

[25] "A Word About the GNU GPL and Patent Rights: LizardTech has chosen to use the GNU GPL because this license is accepted in the Open Source community…. At the same time, LizardTech does not endorse or agree with all of the underlying assumptions or biases of the Free Software Foundation that are reflected in the GNU GPL. In particular, the GNU GPL is generally antagonistic to patent protection for software. … On the issue of patents, LizardTech believes the approach taken in the Mozilla Public License … is preferable to that in the GNU GPL, because the MPL explicitly provides for contributors to grant patent rights to users of the open source software …." (LizardTech 2001).