Software Patents: A Replication Study

Germán Poo-Caamaño
University of Victoria
Victoria, BC, Canada
gpoo@uvic.ca

Daniel M. German
University of Victoria
Victoria, BC, Canada
dmg@uvic.ca

ABSTRACT

Previous research has documented the legal and economic aspects of software patents. To study the evolution in the granting of software patents we reproduced and extended part of the empirical study on software patents conducted by Bessen and Hunt. The original study established a criteria to identify software patents, and provided a look at the evolution of patents granted until 2002. We present a simple approach to retrieve patents from the full text database provided by the United States Patent and Trademark Office (USPTO), which is freely accessible. We also present the evolution of software patents since the original study, and which we also present separated by major technological firms. Our research shows a continuous increase in the number of software patents granted higher, both in number of patents granted (in absolute numbers) and in proportion of overall patents (in relative terms). The relevance of studying the evolution of software patents relies in the challenges to find prior-art, either for practitioners looking for patenting as well as for examiners evaluating granting a new patent.

1. INTRODUCTION

There are four main types of protection for an intellectual property: copyright, trade secrets, patents and trademarks. Today, the software industry takes advantage of all of them. Source code is copyrighted, often protected as a trade secret (when developers sign non-disclosure agreements), companies trademark names and logos of the software they develop, and patents are used to protect the algorithms and processes that the software implements.

Historically, copyright was the most common way to protect software, mostly due to its simplicity. The moment a person writes code, this person (or its employer) obtains copyright protection on it. Registration is simple and inexpensive, and it is only necessary if one party is to sue another party for copyright infringement. In 1980, the United States amended its copyright law to add software programs are copyrightable material [4]. The European Union did the same in 1991 with the Computer Programs Directive [5].

In the United States *Apple Computer, Inc. v. Franklin Computer Corp* [11] demonstrated that copyright was strong enough to protect a software. Legal cases, however, started to push the limits of this protection. In *Lotus Development Corp. v. Borland International*, Inc. [6] Lotus had sued Borland for copying the menu hierarchy of Lotus 1-2-3. The major issue at hand was whether copyright protection extended to the "look and feel" of the program (the menus and their contents). The Supreme Court of the United States ruled that the menu hierarchy of the Borland's program was a "method of operation" which cannot be protected with copyright. One outcome of this case was that software companies started to protect their intellectual property through patents [7].

As reported by Bessen and Hunt [3] there is no official definition of software patent. The system to classify patents does not distinguish if the underlying technology is software or not. However, it is important to set the research scope of the patents related to software. Thus, Bessen and Hunt [3] define software patent as:

> *"Software patent involves a logic algorithm for processing data that is implemented via stored instructions; that is, the logic is not "hard-wired". These instructions could reside on a disk or other storage medium or they could be stored in "firmware," that is, a read-only memory, as is typical of embedded software. But we want to exclude inventions that do not use software as part of the invention. For example, some patents reference off-the-shelf software used to determine key parameters of the invention; such uses do not make the patent a software patent."*

This definition is claimed to be reasonably accurate in comparison with the literature, and more accurate in terms of false positives than the definition based on International Patent Classification [2].

Over time, the patents granted related to software have increased. The timing seems consistent with legal changes that made easier to obtain such patents. Software patents are dominated by industries in the fields of computers, electronics, and instrumentation [3,7].

Software patents are difficult to find because the lack of a standardized and comprehensive scheme to classify them [10]. As a consequence, software patents can be found classified in multiple categories. The increase of patents granted—plus the complexity of searching patents using keywords—makes the correct classification of a software patent an interesting problem to tackle.

OpenSym '15 August 19–21, 2015, San Francisco, CA, USA
© 2015 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-3666-6/15/08…15.00
DOI: http://dx.doi.org/10.1145/2788993.2789833

2. RELATED WORK

Bessen and Hunt [3] performed an empirical study on software patents. The study explored the effects on the patenting behavior of public firms as a consequence of the shift experimented by the federal courts in United States and the US Patent and Trademark Office (USPTO) with respect to software patents. In the 1970s, software algorithms were not patentable [7]. However, such position started to shift gradually after the Supreme Court permitted the the patenting of software algorithms in 1981, when ruled the cases *Diamond v. Bradley* [12] and *Diamond v. Diehr* [13].

Graham and Mowery [7] studied the evolution of the intellectual protection in United States software industry since software companies started to appear in 1978. They explain the use of copyright in the beginning, and then the increase of software patents once they were allowed by the US Supreme Court in different cases during the 1980s and 1990s.

Mulligan and Lee [10] wrote about the scalability issues on software patents. They claim that some industries are unable to respect other's patent rights because it is not feasible to do so. Software patents are not indexed or organized, which makes unfeasible to look all patents in a particular technology. To look for prior-art, patents lawyers have to search across all patents by using specific keywords, inventors, patents assignees, or citation between patents. One challenge is to guess the right keywords to find prior-art. Another challenge is to examine all patents that match the keywords, which might not be feasable if the number of patents is big enough, presenting scalability issues.

Several studies [7,9,10] state that one of the weaknesses of the United States patent system regarding to software patents is the granting of *low quality* patents, particularly when the access to prior-art is limited, including access by the examiners [7,9]. A *Low quality* patent is a patent that does not truly satisfy the definition of patentable invention, that is, novel, non-obvious, and utilitarian; hence a patent that should not have been granted.

Patents are considered a threat for software projects, particularly for small companies and Free and Open Source Software (FOSS) projects that might not have the economical nor legal support to negotiate in case of patent infringement. The existance of low quality patents has lead to initiatives like *Defensive Publications* [1], to encourage individuals and organizations to publish known inventions that have not been patented, so they can become prior–art, and therefore, it would prevent infringement claims for patents granted afterwards [1].

The goal of this study is to replicate the work by Bessen and Hunt [3] to better understand the evolution of software patents over time and to bring their study up to date (by adding data from 2002 to 2014).

Thus, in this paper we address the following research questions:

RQ_1. Can we reproduce the results of Bessen and Hunt's?

RQ_2. What is the evolution of software patents since Bessen and Hunt's study?

To answer RQ_1, we searched software patents in the United States Patent and Trademark Office (USPTO) database issued between 1976 and 2002, and compared our results against the original study of Bessen and Hunt [3]. To answer RQ_2, we extended the search from 2002 to 2014.

3. STUDY DESIGN

3.1 Data Sources

Figure 1 depicts different data sources available to retrieve patents from, and which were used by Bessen and Hunt [3]. NBER (National Bureau of Economic Research) Patent Citation Data File [8] is a public domain data set made available for research purposes. It has been reported to have a reasonable match with Compustat data source [3]. It contains patents granted from 1975 to 1999. Compustat is a proprietary database of financial and statistical information, it was used to collect information of firms investing in Research and Development. CHI Research is a proprietary database that contains matches between firms and patents [3]. Finally, USPTO database is the canonical database of patents in United States. In this replication study we used the public data sources.

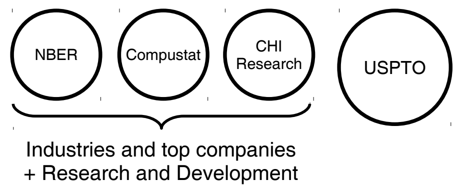

Figure 1: Alternatives data sources used by Bessen and Hunt [3] to retrieve data of patents.

3.2 Data collection

Bessen and Hunt describe an *algorithm* that allows the retrieval of software patents from the USPTO database. The algorithm is no other than a criteria to perform a keyword search in the USPTO database. To develop the algorithm, Bessen and Hunt performed a manual classification of 400 patents, and then refined—by trial and error—a search criteria that matched the classification. The final search criteria is shown in Listing 1. It contains the search to utility patents (*APT/1*), whose description contains either the keyword *software* or the keywords *computer* and *program*. It excludes patents likely to be related with electronic or electricity, whose patent title (*TTL*) contains at least one of the keywords *chip*, *semiconductor*, *bus*, *circuit*, or *circuitry*. It also excludes those patents whose description contains at least one of the keywords *antigen*, *antigenic*, or *chromatography*.

```
SPEC/(software OR (computer AND program))
AND    APT/1
ANDNOT TTL/(chip OR semiconductor OR bus OR
            circuit OR circuitry)
ANDNOT SPEC/(antigen OR antigenic OR
             chromatography)
```

Listing 1: Search criteria used to query the USPTO patents database.

In addition, we retrieved by year, using the filter issue date (*IDT*). For example, *ISD/($/$/2014)* to retrieve the patents granted in 2014.

We used a script to query the USPTO Web site[1] using as input the content in Listing 1, plus additional filters as needed, like year (*IDT/*) or company (*AN/*). and then to scrape the Web page with the results of our queries[2]. As the search engine did not provide the

[1]http://patft.uspto.gov/netacgi/nph-Parser?%s

[2]The script used and data retrieved are available at http://calcifer.org/research/software-patents

number of matches per query, the script performed the pagination to retrieve and count the results per page. To overcome the rate limit per request of USPTO Web site, the script simulated an user session, waiting a random number of seconds between queries.

4. RESULTS AND DISCUSSION

In this section we present and discuss the results with respect to the research questions.

4.1 *RQ$_1$*: Can we reproduce the results of Bessen and Hunt's?

Because software patents are a subset of utility patents, we collected the query results for both software and utility patents, and tabulate them as shown in Table 1. The table is split between the results reported by Bessen and Hunt (from 2nd to 4th columns) and this replication study (last 3 columns).

	Bessen and Hunt (2007)			Replication Study		
Year	Software Patents	Utility Patents	%	Software Patents	Utility Patents	%
1976	765	70,226	1.1%	725	70,225	1.0%
1977	884	65,269	1.4%	838	65,269	1.3%
1978	897	66,102	1.4%	845	66,102	1.3%
1979	795	48,854	1.6%	746	48,854	1.5%
1980	1,080	61,819	1.7%	1,019	61,819	1.6%
1981	1,275	65,771	1.9%	1,210	65,771	1.8%
1982	1,402	57,888	2.4%	1,319	57,888	2.3%
1983	1,443	56,860	2.5%	1,358	56,860	2.4%
1984	1,939	67,200	2.9%	1,844	67,200	2.7%
1985	2,453	71,661	3.4%	2,357	71,661	3.3%
1986	2,657	70,860	3.7%	2,530	70,860	3.6%
1987	3,530	82,952	4.3%	3,384	82,952	4.1%
1988	3,495	77,924	4.5%	3,326	77,924	4.3%
1989	4,974	95,537	5.2%	4,751	95,537	5.0%
1990	4,704	90,364	5.2%	4,481	90,365	5.0%
1991	5,347	96,513	5.5%	5,080	96,511	5.3%
1992	5,862	97,444	6.0%	5,579	97,444	5.7%
1993	6,756	98,342	6.9%	6,425	98,342	6.5%
1994	8,031	101,676	7.9%	7,579	101,676	7.5%
1995	9,000	101,419	8.9%	8,571	101,419	8.5%
1996	11,359	109,645	10.4%	10,818	109,645	9.9%
1997	12,262	111,983	10.9%	11,534	111,984	10.3%
1998	19,355	147,519	13.1%	18,414	147,520	12.5%
1999	20,385	153,486	13.3%	19,456	153,488	12.7%
2000	21,065	157,595	13.4%	20,730	157,493	13.2%
2001	23,406	166,158	14.1%	23,073	166,034	13.9%
2002	24,891	167,438	14.9%	24,550	167,325	14.7%
2003				27,362	169,022	16.2%
2004				30,113	164,289	18.3%
2005				28,901	143,806	20.1%
2006				40,871	173,772	23.5%
2007				38,356	157,282	24.4%
2008				41,958	157,772	26.6%
2009				47,740	167,349	28.5%
2010				64,480	219,614	29.4%
2011				69,029	224,504	30.7%
2012				84,243	253,155	33.3%
2013				96,634	277,835	34.8%
2014				109,281	300,713	36.3%

Table 1: Software patents and utility patents between 1976 and 2014, a comparison of results from Bessen and Hunt's work and our replication study.

The results were close, although we did not obtain exactly the same figures. The differences in software patents vary from 40 (in 1976, 765 versus 725) to 941 per year (in 1998, 19,355 versus 18,414). In spite of the difference, the tendencies in both results look similar. We hypothesized that some patents might have been reclassified between the time that both studies were performed. Further inspection is needed to understand the differences.

4.2 *RQ$_2$*: What is the evolution of software patents since Bessen and Hunt's study?

To better understand the evolution of software patents since 2003, we present Figure 2, which depicts both studies between 1976 and 2002, and from 1976 to 2014. Since early 1990s, there is a steady increase of software patents granted.

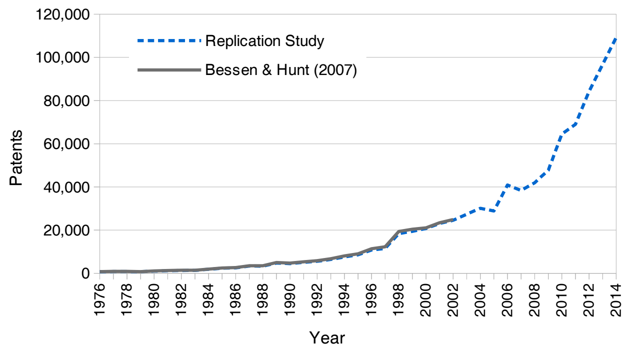

Figure 2: Comparison of results of software patents collected from 1976 to 2014.

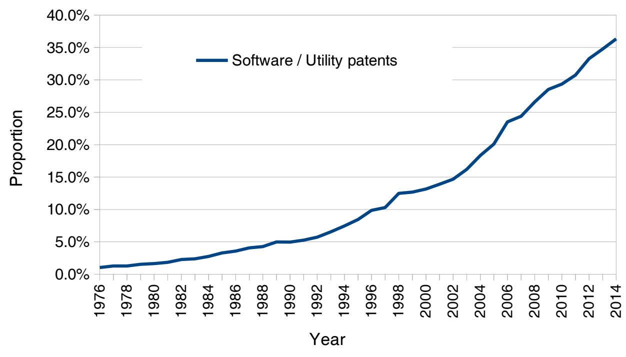

Figure 3: Evolution of the proportion of software patents with respect to utility patents from 1976 to 2014.

Bessen and Hunt [2,3] reported that early in the 1990s the process for applying to utility patents had became more cost–efective. However, the software industry has increased since then as well. Figure 3 shows the share of software patents within utility patents. A high proportion of utility patents correspond to software patents. In 2014, 36.3% of the utility patents were software patents.

The share of software patents by large firms is another aspect to consider. Figure 4 shows the evolution of new software patents in a selected group of firms. We observe a steady increment of software patents granted by company, with a peak in 2012. The major firms obtaining software patents per year are IBM, followed by Microsoft, Qualcomm, Google, and Canon. Further research is needed to understand the slope in 2013, or high peak in 2012.

These findings open new research problems, such as, improving the algorithm to identify software patents, assessing the quality of the patents, identifying possible prior-art.

5. THREATS TO VALIDITY

In general, the following threats to validity exists for the described approach.

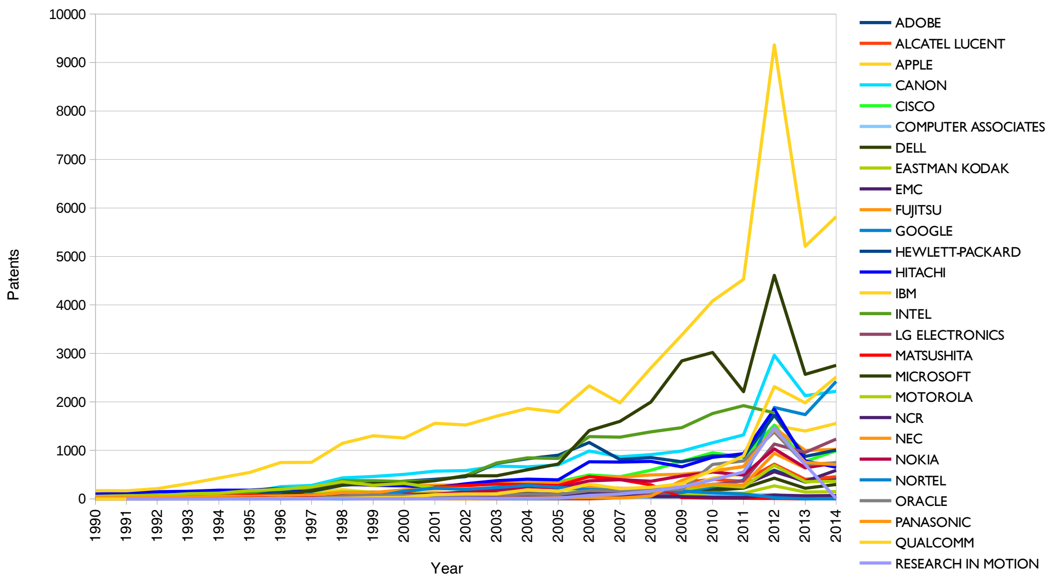

Figure 4: Evolution of new software patents per year in (selected) firms from 1990 to 2014.

Internal validity. As stated by Bessen and Hunt [3], there is no official definition of software patent by the USPTO. The definitions have been built by researchers. For the purpose of this study, we rely on the same definition provided by Bessen and Hunt.

Construct validity. The original work reproduced in this study consisted of a manual classification of patents of 400 patents, that lead to a refined search criteria that matches the classification. To our knowledge, the resulting set of patents categorized are not publicly available and, therefore, cannot be reproduced. Although an important step, it goes beyond the purpose of this replication study, which consisted in the automatic retrieval of potential software patents. In addition, the increasing number of software patents may as well stand for the fact that the query defined in the original work is becoming less precise as time passes. Future work should involve experts on patents to revalidate the classification of patents.

External validity. This study only applies to patents granted in United States and the criteria used for searching patents depend on the classification provided by the USPTO. Although other countries have patent systems, they might have a different regime to grant them. These factors prevent the generalization of the results to other regimes different than United States patent system.

6. CONCLUSIONS

We studied the same data source that a previous study, although our results were not exactly the same, they were close enough to conclude that we were able to reproduce the tendency with similar results. Considering these results, we can know retrieve the content of each patent to apply different techniques to classify patents.

The increase of software patents grants can be interpreted to mean that copyright might not be enough to protect software. Or at least, it might be a perception among firms applying for software patents.

However, the number of patents does not say anything about quality of the patents. For this, manual inspection might be needed. Data mining techniques can be applied to assist the process.

7. REFERENCES

[1] Defensive Publications. http://defensivepublications.org/. Visited on 2014-02-03.

[2] J. E. Bessen and R. M. Hunt. A Reply to Hahn and Wallsten. Technical report, 2004.

[3] J. E. Bessen and R. M. Hunt. An Empirical Look at Software Patents. *Journal of Economics & Management Strategy*, 16(1):157–189, Mar. 2007.

[4] B. Bordoloi, P. Ilami, P. P. M. Jr., and K. Mykytyn. Copyrighting computer software: the "look and feel" controversy and beyond. *Information & Management*, 30(5):211 – 221, 1996.

[5] European Council. Computer Programs Directive, 1991.

[6] First Circuit. Lotus Development Corp. v. Borland International, Inc., 1995.

[7] S. J. H. Graham and D. C. Mowery. Intellectual Property Protection in the U.S. Software Industry. In *Patents in the Knowledge-Based Economy*, chapter 7, pages 219–258. The National Academies Press, 2003.

[8] B. H. Hall, A. B. Jaffe, and M. Trajtenberg. The NBER Patent Citation Data File: Lessons, Insights and Methodological Tools. Working Paper 8498, National Bureau of Economic Research, October 2001.

[9] B. H. Hall and M. MacGarvie. The private value of software patents. *Research Policy*, 39(7):994–1009, Sept. 2010.

[10] C. Mulligan and T. B. Lee. Scaling the Patent System. *NYU Annual Survey of American Law, Forthcoming*, pages 1–28, 2012.

[11] Third Circuit. Apple Computer Inc v. Franklin Computer Corporation, 1983.

[12] U.S. Supreme Court. Diamond v. Bradley, 1981.

[13] U.S. Supreme Court. Diamond v. Diehr, 1981.