Minimizing IPR Infringement Risks in Open Source Projects

Mikko Välimäki and Ville Oksanen
Helsinki University of Technology, 02015 HUT, Finland
mikko.valimaki@hut.fi, ville.oksanen@hiit.fi

Abstract. Software engineering and legal studies have had traditionally very little interaction. Unfortunately, the legislation sets increasingly restricting parameters for the possible design choices and actual implementation of the software projects. If these limitations are not taken into account, the outcomes may be illegal in the worst case. One example of this is different privacy and data protection legislations, which have typically very strict penalties including even criminal sanctions.

In this article we concentrate on mitigating the intellectual property rights infringement risks in open source software projects. Open Source highlights the risks of potential third party copyright and patent infringements. If anyone can contribute to the source code, it is possible that the code infringes third party copyrights without the knowledge of a subsequent developer and the end-user. Further, there may be hidden patents, which are not known until the software has become popular.

We present three defensive risk management practices for open source projects: (1) liability allocation in software license, (2) liability insurance, (3) acquiring patents either directly by patenting or indirectly by gaining access to competitor patents, and (4) risk avoidance. Then, we present evidence on actual risk management practices. We find that the sale of liability warranties for open source is already a growing business for developers, but risk management practices could be significantly improved. For example, insurance markets are not well formed and only large IT enterprises can use patents as a defensive tool as patent pools have been a neglected option so far. We suggest that further empirical research is needed to study the actual infringement risks and risk management alternatives in open source development.

Keywords. Open source, risk management, product liability, liability insurance, patent strategy.

1. Introduction

> *"An engineer, a physicist, and a lawyer were being interviewed for a position as chief executive officer of a large corporation. The engineer was interviewed first, and was asked a long list of questions, ending with "How much is two plus two?" The engineer excused himself, and made a series of calculations before returning to the board room and announcing, "Four."*
>
> *The physicist was next interviewed, and was asked the same questions. Again, the last question was, "How much is two plus two?" After a consultation with the United States Bureau of Standards and many calculations, he also announced, "Four."*
>
> *The lawyer was interviewed last, and again the final question was, "How much is two plus two?" The lawyer drew all the shades in the room, looked outside to see if anyone was there, checked the telephone for listening devices, and then whispered, "How much do you want it to be?"*

This joke and many other similar ones describe too well the attitudes between lawyers and engineers. These groups don't generally speaking interact very well. This is unfortunate,

because close co-operation could help reducing the legal and thus also business risk of projects of all kind. This is even more imperative for software development because the area is basically mined with different liabilities and very high failure rate.

In a typical software project the lawyers are – if at all – involved at the beginning while the contract is negotiated. If the project is a bad failure, the lawyers may get involved again in the end. During a typical implementation phase the lawyers are kept as far away as possible.

The same division can be found from the scientific literature. Most of legal literature about software deals with the contract negotiation and different licensing options. On technical literature the legal aspects are typically at most in a footnote. As matter of fact, a search to IEEE's article database with a term "legal risk" brings up only 6 articles. With "legal liability" 22 hits are found, but majority of these articles deals with the liability questions pertaining the expert systems, not development process.

Table 1. illustrates the different areas of law, which may apply to a software project in its different stages.

Project phase		Applicable laws	Level of Involvement
Negotiation	Design	Contract, patent, copyright, fair trade practice, trade secret, competition	High on negotiations, low on design
Implementation		Labor, contract, patent, copyright, fair trade practice, trade secret, competition	Low
Delivery	Support	Contract, liability, fair trade practice, insolvency	Low unless a failure

Table 1.

The recently much debated legal cases initiated by SCO[1] against large Linux users are perhaps the best examples of how intellectual property rights (IPR)[2] can be – in theory – used as a strategic tool against particular software users. Open source, and distributed components based software development methodology in general, highlights the risks of potential third party copyright and patent infringements.

While SCO case is not a good example of a successful IPR offensive because SCO has been unable to prove actual infringement, it has forced many companies to check and tune-up their intellectual property defense strategies. It is very well possible that some companies have hidden and more easily verifiable rights to existing open source products, which may prove troublesome in the future. For example, a leaked memo from Hewlett Packard shows how seriously a large IT enterprise takes the potential risk of patent infringement lawsuits against open source projects. (Barr 2004)

[1] SCO's case is a long and interesting saga, which contains lots of history of Unix, mergers, license agreements and name changes. Due to space requirements, we are not able to explain it here in the details it would require. Instead, see e.g. http://en.wikipedia.org/wiki/SCO_Group

[2] In this article, we refer with IPRs to both patents and copyrights. To be precise, IPRs also include trademarks and trade secrets, which are not discussed here.

We aim to build a practical framework for the management of IPR infringement risks in open source development. Theoretically, the framework is mainly based on the economic theories of product liability (e.g. McKean 1970, Calabresi 1970), liability insurance (e.g. Shavell 1982) and patent strategy (e.g. Teece 2000, Shapiro 2001). We also go on to test the framework presenting evidence on actual risk management practices gathered from publicly available market news, interviews and patent statistics.

For the purpose of our risk management analysis, we identify three types of significantly different organizations who develop open source: IT enterprises, open source companies, and community projects. With *IT enterprise* we refer to established big companies, which have numerous product offerings in the computer and software markets including significant interest and involvement in open source software. Typical IT enterprises are for example Apple, HP and IBM but also Microsoft because of their role as a kind of counterpart to open source. Second, with *open source company* we refer to any small or medium size company that develops and markets software products based on open source components. Pioneering in this sense are those companies, which are build around well-known open source products such as Red Hat Inc. (GNU/Linux) and MySQL AB (MySQL). Third, with *open source community project* we refer to non-commercial but significant open source development projects such as Linux or Apache.

The article is organized as follows. First we discuss the nature of potential third party copyright and patent infringements in open source software. We see them as accidents towards the users caused, which are caused by the actions of the right holder. Then, we introduce four different defensive risk management alternatives: (1) liability allocation in software license, (2) liability insurance, (3) acquiring patents either directly by patenting or indirectly by gaining access to competitor patents, and (4) risk avoidance. To test the relevance of our framework, we review actual risk management policies. We find that the risk management strategies of established IT enterprises, pioneering open source companies and open source community projects each differ from the others. Obviously, an optimal strategy depends on the actual infringement risk, which may be significantly different for various types of developers. Finally, we conclude the article by discussing implications to long-term risk management and licensing policy.

2. Nature of Third Party IPR Infringements

2.1 Infringements as Accidents

Basic relationships between involved parties in an infringement situation are characterized in the figure 1:

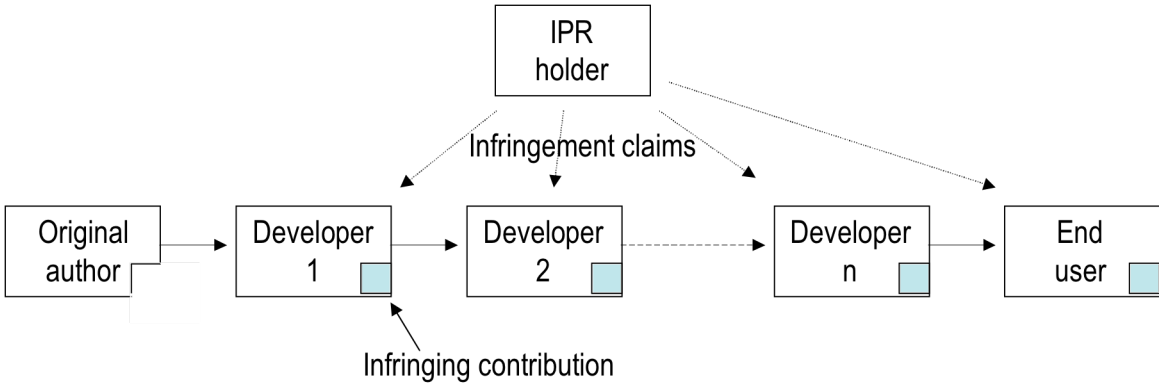

Figure 1. Developer-chain and IPR infringement.

Assume there is an open source product, where $Developer_1$ (acting in good faith) adds a contribution, which infringes IPR holder's copyright or patent. Now, every subsequent developers from $Developer_2$ to $Developer_n$ can be liable according to copyright and patent laws even if they did not know that the software infringes a third party right. Developers can violate copyright by e.g. copying, modifying and distributing the infringing source code contribution. Developers can also infringe patents by e.g. making, marketing and selling patented inventions embodied in the infringing contribution. In essence, such direct intellectual property infringements are under a strict liability rule.

First, we believe that the situation of an individual developer towards users (any subsequent developer or the end-user) is significantly analogical to a product liability setting. Therefore, we start by analyzing *IPR infringement as an accident* resulting in a negative externality for all open source developers and their users. Our question is, following Calabresi's (1970) seminal treatise on the economics of accident law, what are the costs of such accidents and how open source developers can minimize them within the developer-chain.

Second, we open up the "black-box" of infringement claim and ask how the infringement risk can be managed towards the *cause of the accident*, namely the IPR holder. The question then becomes, what means an individual developer has to protect itself from unwanted actions from the side of IPR holders.

Before our analysis of risk management alternatives, a more detailed explanation of the nature of IPR infringements as accidents is in place. We aim to demonstrate that since the infringements are hardly avoidable *ex ante*, the accident analogy is indeed applicable.

2.2 Copyright

It is possible to think of two types of copyright infringements. Either source code has been copied in ways not permitted in the copyright law or license terms have not been honored.[3] Infringements may be in both in-house or third party produced components. Problems with the latter are naturally more difficult to identify.

Literal source code copying can be avoided to some extent in advance. However, it requires that the suspected copied source code is available, which typically is not the case. Also

3 In fact licenses can be also based on other rights than copyright but for the purposes of this article it is sufficient to address licenses simply as based on copyright.

technical analysis of source code copying does not go very deep. There are clear limitations ,for example, in identifying structural copying.

Also license term infringements can be avoided to some limited extent in advance. A prerequisite is that all licenses in a given product can be listed from external files, source code etc. Further, the known licenses can be evaluated for possible third party rights reservations and other non-standard terms. Unfortunately, the interpretation of license terms may not be clear especially if individual program authors have written them. There is also disagreement in the interpretation of many standard open source licenses (Rosen 2004). Not surprisingly, Free Software Foundation enforces over fifty GNU GPL license violations annually and estimates that they are becoming more commons (Cohen 2003). Also, projects like GPL-violations.org are taking the enforcement to totally new level by using courts to get injunctions against infringing products.

From the above, it follows that an open source software developer or user who utilizes third party components has *only very limited means to avoid copyright infringement risks in advance*. If the source code of the component is not fully available, it is practically impossible to evaluate the risk. The availability of the source helps in theory infringement analysis according to the rough lines explained above, but in any case the analysis will be costly, time-consuming and far from complete. Source code availability may also actually raise the infringement risk since it makes easier for offensive rights owners to identify infringements.

2.3 Patents

In addition to copyright, there may be hidden patents, which are not known until the software has become popular and well known. The risk of patent infringement grows with the geographical market area and is obviously greatest in the United States markets. It is however *difficult to estimate the real risk of patent infringement*. Obviously many commonly used open source software components already infringe patents. For example Open Source Risk Management Inc., who sells IPR insurances, claims that Linux kernel could infringe almost 300 different patents. – Earlier they declared Linux does not infringe copyrights. (OSRM 2004a, OSRM 2004b)

As noted, the identification and evaluation of patent infringement risk is in practice close to impossible. Since most patent claims do not include source code, it is not possible to do technical searches based on available patent data and open source code. In practice it may be possible just to avoid infringing well-known patents and follow the patent portfolios of close competitors. Although, even keeping up with the competitors may prove both prohibitively costly and difficult due to the dynamic nature of software industry.

Already thousands of software patents have been granted in different parts of the world. It would be unrealistic to claim that even the largest IT enterprises could know with precision whether their large products infringe third party patents. However, large enterprises have by definition better means to defense themselves against possible infringement claims. They typically own patent portfolios that can be used strategically for counterclaims and cross-licensing. Little open source companies, which lack extensive patent portfolios, need other defense options.

In short, *even careful developers and users regardless of their size have more limited possibilities to avoid patent than copyright infringements*. The risk is essentially bigger for the

developers of products with large market share. Paradoxically, these are also the developers and programs that are perhaps socially most beneficial. Again, open source code does not help in defense but merely increases the infringement risk since patent owners may more easily identify infringements.

3. Alternatives to Manage Risks

3.1 License Disclaimers and Warranties

According to the cheapest cost avoider principle, the costs of accidents should be allocated to the party who can avoid them with comparatively least costs (Calabresi 1970, McKean 1970). For example, if we assume that developer has superior knowledge of the risks involved in his software, he should also bear the most risks from accidents resulting from its use. Having some, but not excessive, liability increases investments in new product development and also the overall product quality. In software development, however, the standard liability rule has been *caveat emptor*, i.e. the subsequent developer or finally end-user is liable.

Historically, liability limitations were needed in software licenses mainly because developers wanted to avoid the risks of damage caused by technical errors or bugs. It has been usually argued that it is not practically possible to write complex bug-free programs (Brooks 1975) In addition, since each program copy is identical to the other, any error will multiply, and potential liability burden for developer would be huge. Requiring stricter liability would therefore imply significantly higher software prices without much increase in quality. However, in some cases its possible to minimize the effects of bugs that way that the more or less inevitable malfunctioning does not lead to any major loss of life or property. In these cases strict liability would offer great incentive.

Also open source license disclaim any kind of warranty for programming faults or bugs. These risks are shifted down from the original developers to subsequent developers and end-users. Most popular open source licenses limit also the developer's liability from legal errors including copyright and patent infringements to the maximum extend permitted by the law. (Rosen 2004)

However, especially some later open source licenses make difference between technical and legal errors. They include more specific infringement warranty (indemnification) clauses that do shift more liability towards original developers.[4] In practice, this means that if a subsequent open source developer intends to warrant itself from IPR risks against the original developer, an additional indemnification clause need not be negotiated. It most be noted that such a clause is efficient only to the extent the original developer has resources to defend the user in court. In the event the original developer is unknown or bankrupt, the clause has no effect whatsoever. This leads us to discuss the third party insurance alternative.

3.2 Additional Insurance

Independent of the liability allocation, insurance is arguably a desirable alternative from social policy perspective (Shavell 1982). Ideally, when compensation maximum has been agreed on, that can be used further as a liability limitation in license and contractual

4 Currently for instance Open Software License and Real Network's License include copyright or other intellectual property warranty. Open source licenses are listed at http://www.opensource.org/.

obligations. If developers have the liability, they would carry only the risk of paying deductible in the event of a proven infringement. Users would participate in insurance payments by paying slightly higher prices for their software.

An extreme option would be a *public insurance system* much like the car insurance systems commonly used. However, that would also imply high administrative costs and obviously software development would have to be monitored by further government rules. Perhaps some development methods would be banned altogether. (McKean 1970)

Compared to the liability limitation clause in a license, insurance surely costs more and may not be available to non-commercial community developers. In theory, insurance is economically rational choice in those cases, in which accidents are rare and difficult to estimate but when the event finally occurs, the results are often financially devastating. Insurance is also a mean to price IPR infringement risk more objectively on the markets.

Pricing such insurances is however problematic in practice. Since e.g. patent infringement cases are rare, there is little precedent to efficiently price IPR liability insurances. Another problem is that the developers have superior knowledge about the risks, which is costly to acquire independently by the insurers. In such a new market, insurance providers also limit their responsibility and it may be impossible to get an insurance that would cover all types of IPR risks in all possible markets. This also seems to be case: for example patent litigation liability insurance has not been an option mainly because the *supply of such insurances has been low*. There is however some evidence, that the supply would be increasing (Betterle and Davison-Jenkins 2001, CJA Consultants 2003).

3.3 Strategic Positioning in Patents Race

Many economists have compared software patenting to an arms race where all market participants try to patent as much as they can. The result has been an extensive *patent thicket*, whose effects are further amplified by the sequential and cumulative nature of software development[5] (Bessen 2004). The effects of patent thicket on innovation and licensing are not clear but it has been suggested that they can be in fact negative (Shapiro 2001, Bessen and Hunt 2003).

Bednarek and Ineichen (2003) argue that in such an environment the developer has, besides extensive own patenting, basically three options: (1) acquire adequate patent portfolio by using mergers and acquisitions (2) purchase required amounts of patents from competitors, or (3) license the required patents either directly or from a patent pool. Of course, the developer can also decide to do nothing and deal with the situation only if it ever emerges.

The first two options are typically very expensive and thus not even in theory available for typical open source developers. These options are never the less feasible for large IT enterprises, which often use either of them to capture and protect their position in different areas of software development. The last option is more common but not very viable for most of open source developers, which rely on GNU GPL as their license of choice. The reason is obvious: under GPL the licensee has to get a license for all future uses including derivate works. Patent pools do not normally offer that kind of licensing option, at least by default.

---

[5] The concept of patent thicket is not new. A similar development has been seen with many other emerging technologies such as semiconductors (Lemley 2002) and copying machines (Melamed and Stoeppelwerth 2002).

But how threatening are patents after all? It has been commonly argued that a mere infringement claims do not yet mean that open source could not be used. It has been estimated, that in the United States only 1,5% of all patents are ever litigated and as few as 0,1% are litigated until trial (Lanjouw and Schankermann 2001 and Lemley 2001). Further, as many as 46% of all patents litigated to trial are finally held invalid (Allison and Lemley 1998). This has lead Lemley and Shapiro (2004) to characterize patents not as exclusive but rather as "probabilistic" property rights bearing similarities to lottery tickets. When we also take into account that the most potential court case targets have patents of their own for defending purposes (cross-licensing and counter suits), then perhaps *ignorance of the whole patents race may be a reasonable strategy* for smaller companies and community projects.

3.4 Risk Avoidance

Finally, one may try to avoid IPR risks as much as possible. Within the development process, risk management policies can be improved by using formal written copyright assignments and prior patent searches and assurances from contributing developers. To be sure, an "avoidance at-all-cost" strategy can't be effective because, as we noted, many infringement risks simple can't be avoided in advance. Also, if open source products become standard, as has already happened in e.g. Internet technology, avoidance would be socially inefficient.

A long-term "environmental" avoidance strategy is to influence the legal policy to limit the reach of IPRs. Such *lobbying* is typically done through industry pressure groups or by helping grass-root activists. It makes economically sense to invest in lobbying as long as the marginal benefits (in this case lower IPR-risk) exceed the marginal costs (Landes and Posner 2003). There are some problems, though. One is that the political system is sometimes resilient towards lobbying and it is hard to predict the outcome. Also, in the case of IPRs there are typically other participants, whose interest and lobbying are diametrically opposite and thus may neutralize the effort.

3.5 Summary

Table 1 below summarizes the scope, effectiveness and costs of the discussed risk management alternatives.

	Scope	**Effectivity**	**Price**
Disclaimer	Licensees	Low	Low
Insurance	Market	Relative	Relative
Patenting	Market	High	High
Avoidance	Market	Low	Relative

Table 1. Comparison of different IPR defense options for open source developers.

While a liability disclaimer in license is easy to add, it does not protect the developer towards IPR holder. Insurance is relatively more effective and can, at least in theory, protect developers against both licensees and IPR holders. However, in practice the liability insurance market is not very well formed yet. Patenting is a seen as a highly effective risk management option though it costs undoubtedly more than any other alternative. Also, the real infringement risk of patents can be questioned. Finally, the rationality of avoidance should be judged against the lost benefits from using the risky software.

4. Actual Management Practices

4.1 Background

In order to understand the actual management practices we collected qualitative information from publicly press releases and interviews on the Internet and patent statistics. As noted, we divided open source developers into three categories: (1) large IT enterprises who sell ”solutions” and large open source installations, (2) small pioneering open source companies who typically have an innovative open source product, and (3) community-lead open source projects without a company background. Next, we discuss the risk management practices of these three different types of organizations towards users (costs of accidents) and third parties (causes of accidents). Our aim is to preliminary test the feasibility of the theoretical framework.

4.2 IT Enterprises

Towards users. When Utah-based software company SCO claimed in spring 2003 that Linux infringes their intellectual property, large IT enterprises selling and supporting Linux systems didn't respond immediately. Soon, however Hewlett-Packard started a new era by offering a limited warranty to its customers in fall 2003 (HP 2003). Novell and Red Hat followed in the beginning of 2004 (Novell 2004, Red Hat 2004). This way intellectual property warranties had become a kind of *additional warranty business* for large IT enterprises.

Admittedly, these warranties are far from perfect: HP indemnifies only claims (both copyright and patents) in SCO case, Novell has limited indemnification to copyright claims, and Red Hat only promises to change infringing components without additional costs to the user. In addition, according to a recent survey, most corporate Linux users have not expressed interest in these programs. (Fichera 2004). It seems evident that such warranty policies only reflect how risk-averse IT enterprises are towards IPR infringements.

Some IT-enterprises have been using also Open Source Development Labs to create a joint Linux legal defence fund for Linux end users. The fund is not tied to any single vendor's Linux-distribution, but only supports end-users against SCO's suits, which confines its scope significantly. (Open Source Development Labs 2003).

Towards third parties. The approximate size of certain interesting companies' patent portfolios is presented in the table 2 below:

IBM	33665
HP	13898
SUN	4062
Microsoft	3359
Apple	1776
Novell	444
Red Hat	1

Table 2. Total number of US patents granted from 1984 to early August 2004. Source: USPTO.[6]

[6] The number of those patents whose annual fees have been paid, and are an active part of the company portfolios, is obviously lower (Lanjouw and Pakes 1998).

Known open source promoters IBM and HP have clearly the biggest patent portfolios though most of their software patents are from the late 1990s. Many early patents cover only hardware. It is also interesting to note that Novell has just fewer than 500 patents total and Red Hat, which should be counted as an open source company, was granted its first patent in 2004. Still, these two companies feel confident to offer IPR insurances for their Linux customers.

It seems evident that *patenting will increase in the future*. For example Microsoft has announced to apply for over 3000 patents, which means multiplying its current patenting activity a few times (Fried 2004). It obviously aims to reach the level of IBM and HP.

However, there seems to be no immediate risk of patent infringement claims by the biggest patent owners. All of them subscribe to the *mutual defense policy*. IBM has said to use its patents only to defend itself in potential open source cases. For instance, after SCO had sued IBM for Linux infringements, IBM filed a countersuit in terms of three patent infringements (Shankland 2004). Also HP has committed to open source and has prepared to defense itself against possible patent offensives (Barr 2004). Finally, Microsoft has publicly said they look only for licensing possibilities, not attacks (Stone 2004). Of course, these oral statements have no legal effect.

4.3 Open Source Companies

Towards users. We found interestingly that many open source companies *offer extensive warranties* for their products. For copyright infringements, warranties sound justified, since source code rewriting is common (Välimäki 2003). Also license infringement seems unlikely since these companies should have better knowledge of open source licensing. But for third party patents, infringement risk is harder to manage.

Towards third parties. According to a recent European Union study, patent infringement insurance has become a viable patent defense option to European small and medium size companies. The availability and terms of such insurances vary however greatly country by country and in practice such insurances may not cover litigation in the United States, where the coverage would be most relevant. The patent infringement insurance market within the United States is even less developed (CJA Consultants 2003).

Red Hat was the first open source company to publicly announce it files for defensive patents in 2002 (Red Hat 2004b). The lesson from the patenting arms race is that it takes years to build a credible patent portfolio by filing patents. The first patent was granted to Red Hat in 2004, four years after its actual filing date. Also, using the patents is relatively more costly to small companies as they are targets of patent lawsuits more often than enterprises having large patent portfolios (Lanjouw and Schankermann 2004). – Open source companies also typically oppose patents strongly at the policy level (Farber 2003).

4.4 Community Projects

The best option to community projects seems to be to keep liability disclaimers intact and avoid patented and proprietary technology with reasonable means. As a practical measure, for instance, the Linux development process was recently modified to better document the origin of the source code and any subsequent changes to it. The aim is to avoid legal uncertainties in

advance. Now all Linux developers must click through a statement saying that to their best knowledge the new source code is home made, not copied from anyone, and that its distribution in Linux is allowed with GNU GPL (OSDL 2004). – It is worth noting that this ”assurance” does not carry any kind of legal effect though.

Open source and free software activists have been circulating the idea of creating a patent pool for GPL-software for some time now. For example, Richard Stallman has been supportive for the idea, but so far Free Software Foundation has not made any formal steps to create a “GNU-patent pool” (Kelly 2000).

Open source community developers *campaign visibly against stricter intellectual property legislation*. Many developers have warned that software patents and increasing proprietary control threat the functionality of the open source model. These warnings have a long history in developer communities: there has been campaigning against software patentability since the early 1990s (League for Programming Freedom 1991).

Recent political campaigning around the Europe’s software patent directive proposal has increased fears that distributed open development is particularly vulnerable to patent monopolies.[7] Well-known open source developers including Linus Torvalds and Alan Cox have expressed critical public opinions on software patents in general and the directive proposal in particular (Torvalds and Cox 2003).

5. Concluding Remarks

We believe our approach of analyzing IPR infringement risks to open source developers as product accidents is workable. Future research efforts should be focused to study in more detail how different risk management alternatives are actually used and what is the real level of infringement risks for both copyright and patents.

In this study, we found that the use of license disclaimers continues to be the standard in the industry. However, we also found that both IT enterprises and open source companies have started to sell liability warranties to their users. The growing warranty business proves that more research on risk management would be relevant to both developers and business managers.

The insurance alternative is still in large part an untouched ground. Open source companies are only slightly but perhaps increasingly interested in defensive insurance plans. The biggest challenge may however be non-commercial community projects, which are currently not able to gain directly from expensive insurances.

We suggest the defense options to patent infringements should be studied further. It is clear that IT enterprises have collected large patent portfolios consisting of typically thousands of patents during the last ten years. They have publicly announced to use them only defensively towards open source. Also open source companies are investigating defensive patenting possibility. While a mutual defense policy seems to be accepted by almost all companies, we did not found evidence of concrete IPR defense alliances in open source development.

[7] Proposal for a directive of the European Parliament and of the Council on the patentability of computer-implemented inventions, Brussels, 20.02.2002, COM(2002) 92 final.

Perhaps patent pools have been an overlooked option. If pools would be constructed taking into board all parties, including non-commercial open source developers, they would surely lower the social costs of patent infringement risks.

To summarize, in addition to liability allocation *inter partes* and patenting, the risk management game in open source development includes third party liability insurances, careful risk avoidance, and even political influence on the development of intellectual property laws. Like Calabresi (1970) noted in his analysis of accidents, it is plausible to assume that an optimal IPR risk management strategy for each type of organization is a mix of available alternatives.

Finally, what could be done better to minimize the social costs of IPR infringement risk for open source in the future? In short term, different IPR risk management techniques should be developed further from the level of source code integrity up until political activity. In the long term, the intellectual property rights legislation would need a revision. The mere existence of socially undesirable accidents, as we call them, is based on the strict infringement liability defined intellectual property laws.

References

Allison J. R., M. Lemley. 1998. Empirical Analysis of the Validity of Litigated Patents. *American Intellectual Property Law Association Quarterly Journal*. **26**(3) 185-275.

Barr, J. 2004. HP memo forecasts MS patent attacks on free software. *NewsForge*, 19th July 2004.

Bednarek, M., M. Ineichen. 2004. Patent Pools as an Alternative to Patent Wars in Emergent Sectors. *Intellectual Property & Technology Law Journal* **16**(7).

Bessen, J., R. M. Hunt. 2003. An Empirical Look at Software Patents. *National Bureau of Economic Research Working Paper* #2003-17

Betterle, R. S. , D. J. Davison-Jenkins. 2001. Coverage concepts for intellectual property. *Risk Management* **48**(2) 17-21.

Brooks, F. P. 1975. *The Mythical Man-Month*, Addison-Wesley.

Calabresi, G. 1970. *The Costs of Accidents. A Legal and Economic Analysis.* Yale University Press.

CJA Consultants. 2003. *Patent Litigation Insurance. A Study for the European Commission on possible insurance schemes against patent litigation risks.* Final Report. European Commission, Brussels.

Cohen, N. 2003. What's GNU? GPL v3 and ASPs. *Open Magazine*. 3rd July 2003. http://www.open-mag.com/features/Vol_66/GNU/GNU.htm

Farber, D. 2003. Unplugged: Mårten Mickos, CEO MySQL AB. *ZDNet Tech Update*. 31st July 2003.

Fichera, R. 2004. Linux IP Litigation: Users Largely Unconcerned About SCO Suit, Indifferent To Indemnification. *Forrester Research* 14th May 2004, summary available at http://www.forrester.com/

Fried, I. 2004. Gates wants patent power. *News.com* 29th July 2004.

Hewlett-Packard. 2003. HP indemnifies Linux customers against SCO lawsuit. 24th September 2003, available at http://www.hp.com/

Kelly, J. S. 2000. An interview with Richard Stallman. *LinuxWorld.com* 29th March 2000. http://www.itworld.com/Man/2687/LWD000329rms/

Landes, W. M., R. A. Posner. 2003. *The Economic Structure of Intellectual Property Law*. Harvard University Press.

Lanjouw, J. O., A. Pakes, J. Putnam. 1998 How to Count Patents and Value Intellectual Property: The Uses of Patent Renewal and Application Data. *Journal of Industrial Economics*. **46**(4) 405-432.
Lanjouw, J. O., M. Schankermann. 2001. Characteristics of Patent Litigation: A Window on Competition. *RAND Journal of Economics*. **32**(1) 129-51.
Lanjouw, J. O., M. Schankermann. 2004. Protecting Intellectual Property Rights: Are Small Firms Handicapped. *Journal of Law and Economics*. **47**(1) 45-74.
The League for Programming Freedom. 1991. Against Software Patents, 28th February 1991, http://lpf.ai.mit.edu/Patents/against-software-patents.html
Lemley M. A. 2001. Rational Ignorance at the Patent Office. *Northwestern University Law Review*. **95**(4) 1497-1532.
Lemley, M. A. 2002. Intellectual Property Rights And Standard-Setting Organizations, California Law Review **90** (6)1889-1981.
Lemley, M. A., Shapiro, C. 2004. Probabilistic Patents. University of California, working paper.
McKean, R. M. 1970. Products Liability: Implications of Some Changing Property Rights. *The Quarterly Journal of Economics* **84**(4) 611-626.
Melamed, D., Stoeppelwerth, A. M. 2002. The CSU Case: Facts, Formalism And The Intersection Of Antitrust And Intellectual Property Law. *George Mason Law Review* **10**(3) 407-429.
Novell. 2004. NovellSupports Enterprise Linux Customers With New Linux Indemnification Program. 13th January 2004, available at http://www.novell.com/
Open Source Development Labs. 2003. Linux Legal Defense Fund FAQ. Available at http://www.osdl.org/
Open Source Development Labs. 2004. OSDL to Support Enhancements to Linux Kernel Development Process. 24th May 2004. Available at http://www.osdl.org/
Open Source Risk Management. 2004a. OSRM Certifies Linux Kernel Free of Copyright Infringement. 19th April 2004. Available at http://www.osriskmanagement.com/
Open Source Risk Management. 2004b. Results of First-Ever Linux Patent Review Announced. 2nd August 2004. Available at http://www.osriskmanagement.com/
Red Hat. 2004a. Red Hat Announces Open Source Assurance to Safeguard Customer Investment. 20th January 2004. Available at http://www.redhat.com/.
Red Hat. 2004b. Red Hat, Inc. Statement of Position and Our Promise on Software Patents. Available at http://www.redhat.com/legal/patent_policy.html
Rosen L. E. 2004. *Open Source Licensing: Software Freedom and Intellectual Property Law.* Prentice Hall.
Shankland, S. IBM pledges no patent attacks against Linux. *News.com*. 4th August 2004.
Shavell, S. 1982. On Liability and Insurance. *The Bell Journal of Economics*. **13**(1) 120-132.
Shapiro, C. 2001. Navigating the Patent Thicket: Cross Licenses, Patent Pools, and Standard Setting. In Jaffe, A., J. Lerner, S. Stern (eds.): *Innovation Policy and the Economy*, MIT Press.
Stone, B. 2004. Nickels, Dimes, Billions. *Newsweek*, web exclusive, 2nd August 2004.
Teece, D. 2000. *Managing Intellectual Capital*. Oxford University Press.
Torvalds, L., A. Cox. 2003. An Open Letter to the European Parliament. Available at http://www.effi.org/patentit/patents_torvalds_cox.txt
Välimäki, M. 2003. Dual Licensing in Open Source Software Industry. *Systemes d´Information et Management*. **8**(1) 63-75.