Towards Better Advocacy for Software Patent Reform: Understanding How Value Differences Influence the Software Patent Debate

A Research Paper submitted to the Department of Engineering and Society

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science in Computer Science, School of Engineering

Nick Garrone

Fall 2024

Advisor

Kathryn A. Neeley, Associate Professor of STS, Department of Engineering and Society

I. Introduction

In 2022, 63.5% of patents granted in the United States were software-related, up from 26.5% in 1991 (Millien, 2023). Patents are temporary monopolies granted by the government that give the creator of an invention exclusive rights to use that invention for a limited period of time (Karakashian, 2015, p. 119). Patents are used to protect many different types of technologies, from engine advancements to vaccine manufacturing processes. While they remain largely uncontroversial in some fields, software patents have become the focus of intense discussion. Over 58% of software engineers wish to abolish the software patent system, according to one survey (Burton, 1996, p. 87).

The patent system was not designed with software in mind. From its inception in 1790, the system has undergone many changes relating to software patents (Griesbach and Camorara, 2016). Most recently, the 2014 supreme court case *Alice vs. CLS Bank International* established new guidelines about what kinds of software are patentable. This made software patents more difficult to obtain, but it left the system even more complicated than before (Zivojnovic, 2015). Despite their broad unpopularity among software engineers, software patents continue to be filed in the thousands by large, engineering-focused organizations (IPOA, 2024). Groups such as the Electronic Frontier Foundation argue against software patents, yet their efforts have been largely ineffective (Kamdar et al., 2015).

This lack of success can be attributed in part to a lack of effective communication on the part of software patent reformers. Pro- and anti- patent sides have different ways of talking about two key concepts in the software patent debate. The first is risk, where pro-patent parties see software patents as a way to mitigate risk while anti-patent parties see them as a source of risk. The second is innovation, where pro-patent parties see innovation as something that is cultivated from the top-down while anti-patent parties see it as something that grows from the ground-up. This paper analyzes discourse in order to support these assertions and to see how these concepts are actually expressed in the debate. It also explores the differing values of software patent reformers and pro-patent organizations. It then analyzes how these values influence the miscommunications that arise between the two sides. In order to perform this analysis, this paper draws upon moral conflict theory and analogical imagination.

II. Problem Definition: The Value Gap Between Software Engineers and Corporations

As established previously, software patents are widely disliked by software engineers. The disconnect arises where engineering-focused organizations, such as Microsoft and Meta, are strong proponents of the software patent system. In 2023 alone, Microsoft was granted 1927 patents and Meta 919 (IPOA, 2024, p. 7). A naive view would assume that the organization would take on the average view of its members; that is, that organizations composed primarily of an anti-software-patent group would also have an anti-software-patent stance. However, the way that these corporations are behaving as entities is disconnected from how their employees would act individually. The most obvious answer to explain the gap is that corporations benefit financially from patents in a way that individuals do not; the power that large corporations wield allows them to navigate the patent system more effectively than any individual engineer. However, this does not ring true in all cases: in some sense, patents serve as an equalizer, allowing small players to protect their technology and not get immediately outcompeted by the superior resources of a large group. Small firms patent less than large ones only as a function of size. The importance of the invention is rarely correlated (Athreye et al., 2020, p. 513). Looking deeper allows us to understand the values that arise in large engineering organizations and how those conflict with views common in software engineering culture on an individual level.

Values of Software Engineers Are in Opposition to the Software Patent System

The values of software engineers can be traced back to the early days of computing. Early computing pioneers, called “hardware hackers”, originated at the Massachusetts Institute of Technology, and later across the United States, in the 1960s and 1970s (Chandler, 2003, p. 230; Kirkpatrick, 2002, p. 167). These early pioneers worked out of an intrinsic love for computing rather than money or fame (Kirkpatrick, 2002). “Problems relating to programming arouse genuine curiosity in the hacker and make him eager to learn more.” (Himanen, 2001, p. 4). Hackers developed important advances in early computing, such as the Linux operating system (Himanen, 2001, p. 20). This group developed a distinct set of beliefs, called the “hacker ethic” by sociologist Pekka Himanen in his 2001 book *The Hacker Ethic and The Spirit of the Information Age.* It is important to clarify that “hacker” in this context does not refer to cyber-criminals who break into computer systems, but instead determined computer enthusiasts (Chandler, 2003, p. 230). The hacker ethic places high value on practical technical competency, diligence bordering on obsession in work, rationalism, the forgoing of material pleasure, freedom of information, and idealistic optimism about the impacts of computing (Himanen, 2001; Kirkpatrick, 2002; Brown, 2008). Hackers focused single-mindedly on their work, even to the exclusion of traditional values, forgoing the physical for the psychological. Hackers are described as

> …unwashed, disheveled, socially inept to the point of rudeness and singularly oblivious to other peoples' feelings…What pleasures they would allow themselves were of a purely cerebral nature; the thrill of getting 'the right solution' to a programming problem. (Kirkpatrick, 2003, p. 169)

As computers became more mainstream and easy to use, hacker culture changed, although the early hackers continued to exert influence (Kirkpatrick, 2002, p. 181).

The hacker emphasis on freedom of information provides especially relevant insight into the software patent issue. Despite their reputation for solitude, early computing pioneers did work together on large, collaborative projects. The development of the Linux kernel, a computer operating system, pioneered this style of working. Contributors worked on the project on their own initiative, with the founder of the project deciding which changes to accept or reject. However, due to the permissive licensing, any contributor could start their own version of the project if they disagreed with the direction of the original founder (Himanen, 2002, pp. 69-71). This represents a fundamental freedom not only to read and contribute to the source code but to take ownership of it. Patents function to stifle the freedom of information– while patents are indeed public, they prohibit others from using the invention for the term of the patent unless the patent holder gives permission. This desire for freedom puts the hacker ethic at odds with software patents.

While the early days of computing are long gone, the hacker ethic that arose during that time period has an enduring impact on software engineering culture today. The values proposed by the hacker ethic have been subsumed by the open source software movement and Silicon Valley business culture. Highly visible aspects of the hacker ethic such as techno-optimism are influential in Silicon Valley today: "...that early stream of idealistic thinking still runs strong: it remains a potent force in shaping the ethos of corporate giants like Facebook, Google, Amazon and Twitter, to name a few." (Baker, 2015, p. 1). Another stream of influence remains in the open-source software movement. " …the 'open source' movement, which involved the widespread sharing of programs via the Internet, perpetuates some of the ethos of the early hackers." (Kirkpatrick, 2002, p. 182). Both mainstays of contemporary software engineering culture, the influence of hackers in these spaces justifies the relevance of the hacker ethic as a lens for analysis. The values of the hacker ethic provide a lens into the mindset of contemporary anti-patent activists.

Organizations Develop a Different Set of Values

The dynamics of large corporations allow a different set of values to arise. Whereas the hackers are focused on relentless innovation and iteration, large organizations have a different set of needs. First, thinking of large organizations as having coherent values is not necessarily accurate: "The values etc., although referred to as 'corporate', are accepted rather than shared; they are labels which are communicated to employees and other stakeholders via a top-down process rather than the result of a participative process based on stakeholder dialogue." (Pruzan, 2001, p. 272). Values do not arise from the members of the organization and are not necessarily shared by them; instead, the needs of the organization as an entity are considered. Whereas hackers heavily exercise personal agency, the confines of a corporate environment de-emphasize agency. Freeman and Engel argue that "Agency revolves around the fact that what is good for the individual is not always what is good for the corporation." (Freeman & Engel, 2007, p. 97). This alienation of individual values from corporate values could explain the gap between software engineers and the corporations that employ them. Innovation is one area where values diverge– creativity and execution, the two ingredients of innovation, lie on a spectrum. Higher creativity makes it more difficult to execute on those ideas and vice-versa (Freeman & Engel, 2007, p. 96). This push-and-pull relationship means that large corporations face significant challenges around innovation. Meanwhile, hackers tend to ideate and execute quickly.

Understanding and Bridging the Disconnect

A failure to address this disconnect would allow the negative effects of software patents to continue damaging the economy. One major negative effect is the proliferation of patent trolls, also known as non-practicing entities, who file patents for the sole purpose of litigating them. These non-practicing entities do not use patents as a source of protection for genuine innovation, but rather for rent-seeking (Karakashian, 2015, p. 120). While their claims are sometimes valid, they also target small companies who would rather settle. 60% of patent lawsuits are brought by non-practicing entities (Karakashian, 2015, p. 121). The proliferation of these patent trolls has even led to a measurable decrease in startup employment. An analysis of anti-patent-troll laws, such as Vermont's prohibition of unreasonable demand letters, found that states adopting such measures had 4.4% growth in tech startup employment (Appel et al., 2019, p. 708). These negative impacts underscore the need for effective reform.

The value differences developed in this section are summarized in Table 1. Understanding how the value differences described previously actually arise in the software patent debate would allow for software patent reformers to communicate more effectively. This paper develops a research approach to investigate how these value differences manifest themselves in discourse.

	Hacker ethic	**Corporate ethic**
Practitioners	Software engineers, especially in open source and Silicon Valley startups	Large organizations, including large tech companies
Values	Technical excellence, diligence, rationalism, freedom, optimism	What is best for the organization
Innovation style	Rapid iteration and relentless execution	Balance between creativity and execution

Table 1. *Hacker values vs. corporate values.* Values developed by individual software engineers differ from the values of larger organizations (Created by Author).

III. Research Approach

In order to examine the value differences in discourse, this paper draws upon a synthesis of two research methods: first, a method of online discourse analysis proposed by Kristen Cole et al., and second, the analogical imagination approach proposed by Claudia Schwarz-Plaschg. These methods are used to analyze discourse on the two sides of the debate: on the pro-reform side, the arguments of software engineers on the pseudonymous forum *Hacker News*, and on the anti-reform side, statements from Microsoft and IBM on the topic of software patent reform.

Foundational Methods of Discourse Analysis

The first method, proposed by Cole et al., describes a concrete method of selecting discourse based on statistical methods and then analyzing that discourse. Cole et al. analyze discourse around gun control through the lenses of both moral conflict theory and actor-network theory. Moral conflict theory is concerned with facilitating better communication by resolving conflict around moral orders: "Moral orders are the 'knowledge, beliefs, and values' people use to determine their actions and ‘make judgements about the experiences and perspectives of others’.” (Cole et al., 2024, p. 3). The goal of moral conflict theory is to achieve transcendent communication, which is a way of reframing conversations in order to break an impasse in moral conflict (Cole et al., 2024, p. 4). The central method of the paper is introducing actor-network theory to moral conflict theory by using actor-networks to understand how different sides of a conflict understand certain shared concepts. These shared concepts, called boundary objects, are points in the discussion where either side of the conflict has a different understanding of what that concept represents (Cole et al., 2024, p. 8). By investigating the actor-networks underlying each side’s representation of the boundary objects, opportunities for transcendent communication can hopefully be achieved (Cole et al., 2024, pp. 8-9). Methodologically, a key part of the paper is the statistical procedure of choosing and analyzing discourse. In order to source the discourse, the authors chose comments from an online forum thread debating gun control. They only considered comments above a certain number of up votes and of a certain date range. Once they sourced the discourse, they performed a keyword frequency analysis to identify key actors, and from there did the analytical work of identifying boundary objects (Cole et al., 2024, pp. 6-7). This integration of statistics functioned to ground the analysis in facts, since it brought into consideration objects that were truly mentioned across many comments. In the software patent debate, this method can be used to identify boundary objects that inhibit communication due to differing underlying values.

The second method is described by Schwarz-Plaschg, who proposes a process called analogical imagination as a way to strengthen analysis of emerging technological trends. Schwarz-Plaschg defines analogical imagination as a process of generating analogies between an emerging trend and a historical one. These analogies are used to make predictions about the emerging trend. Important to the process of analogical imagination is the idea of generating disparate analogies. These analogies may even be contradictory (Schwarz-Plaschg, 2018, p. 141). Once analogies are identified, they are used to make predictions about the emerging technology while still grounding the discussion in historical fact. A key point that Schwarz-Plaschg notes is that analogy selection is inherently a rhetorical process because "... imagination is always already imbued with specific interests and framings." (Schwarz-Plaschg, 2018, p. 140). This underscores the need for the flexibility of the analogical imagination process which, while still carrying rhetorical weight, does not rigidly impose analogy as "...both the source and the target are co-created in the analogical process." (Schwarz-Plaschg, 2018, p. 141). In the software patent debate, this method can be applied by drawing an analogy between a historical reform movement and the software patent reform movement. Analyzing the debate with this analogy in mind would reveal the ways in which power influences how certain values are expressed in discourse.

Creating a Synthesized Approach to Discourse Analysis

To synthesize these two approaches, this paper uses Cole et al.'s method as a foundation and Schwarz-Plaschg's analogical imagination as a supplementary mode of analysis. Software patent discourse is analyzed using the boundary-object approach from Cole et al., and that analysis is framed through an analogy with the labor reform movement in the United States leading to the passage of the Fair Labor Standards Act of 1938. The purpose of the analogy is explicitly to investigate power in the debate– Cole et al.'s original method does not take power dynamics into consideration, but the software patent debate is asymmetrical. While this integrates both approaches, a modified version of each is employed due to the complexity and context of both original methods. Cole et al.'s method of finding boundary objects is used, but they are not analyzed using actor-network theory. Instead, they are analyzed by looking at how each side interprets them in relation to the hacker ethic or corporate ethic values developed earlier.

Schwarz-Plaschg's analogical imagination is used, but it does not form a core part of the analysis. Rather than exploring several analogies explicitly as a part of the research, this paper's analogical imagination process took place informally before the research was conducted. This paper focuses on a single analogy. The purpose of this is to allow for focus: the research method, being a synthesization of two methods, is already heavy; the exploration of multiple analogies, while beneficial, would lead to a lack of focus. Instead, using one analogy with an explicit focus on power allows for a coherent analysis. Figure 1 shows how both foundational methods contribute to a greater understanding of values in the software patent debate.

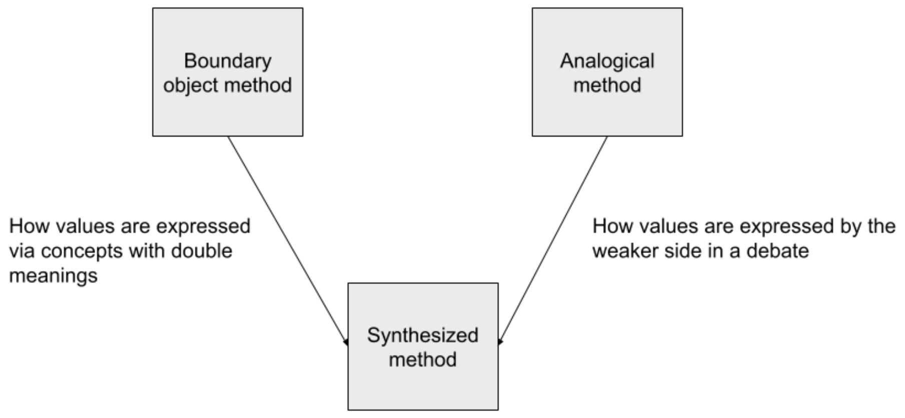
Boundary object method
Analogical method
How values are expressed via concepts with double meanings
How values are expressed by the weaker side in a debate
Synthesized method

Figure 1. *Synthesis of methods.* The boundary object method allows for an analysis of differing conceptions of key concepts in debate, while the analogical method allows for an understanding of how power dynamics change discourse (Created by Author).

The method used is as follows. First, software patent discourse is hand-picked. Second, the discourse is analyzed in order to identify boundary objects. Third, the boundary objects are analyzed by looking at how each side interprets them in relation to their underlying values. Fourth, discourse from the labor movement is gathered. Fifth, comparisons are drawn between the discourse of the labor movement and discourse of the software patent movement with regards to how the power differential is exposed through discourse. Finally, insights are synthesized from the analysis. This process is visualized in Figure 2.

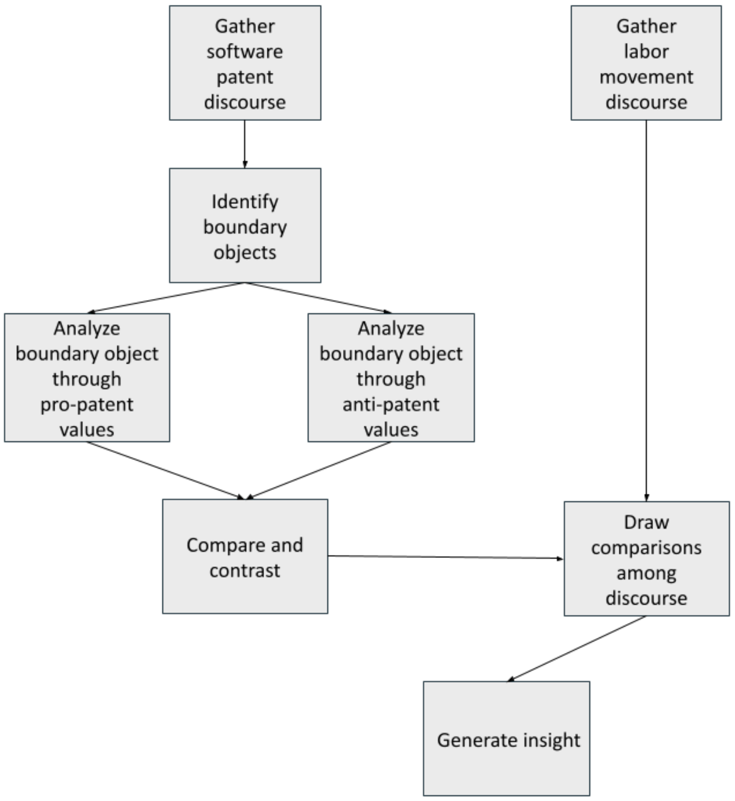
Gather software patent discourse
Gather labor movement discourse
Identify boundary objects
Analyze boundary object through pro-patent values
Analyze boundary object through anti-patent values
Compare and contrast
Draw comparisons among discourse
Generate insight

Figure 2. *Research method.* Selected software discourse undergoes boundary-object analysis to identify areas of miscommunication. Then, comparisons are drawn with the historical labor movement. Finally, insights are synthesized. (Created by Author)

Three sources of discourse are analyzed. The first is a comment thread on the forum *Hacker News* titled *Poll: Do you support software patents?*. *Hacker News* is a forum that is popular among software engineers. The second source is a written transcript of an oral statement from a Microsoft representative commenting on software patent reform at a 1994 US Patent Office public hearing. The third source is a statement from IBM at the same hearing. These sources together provide a representative look at software patent discourse from both sides.

IV. Results

Anti-patent and pro-patent parties have differing perceptions of risk and innovation. Where anti-patent activists see software patents as a source of risk, pro-patent parties see them as a way to mitigate risk. Where anti-patent activists see innovation as something that occurs naturally, pro-patent parties see it as something to be cultivated with patents. Differing interpretations of these boundary objects are represented in Table 2.

	Pro-patent	**Anti-patent**
Risk	Mitigation of risk, protection	Source of risk, danger
Sources of Innovation	Stimulated by top-down incentives	Naturally occuring

Table 2. *Boundary objects.* Risk and innovation are treated differently by pro- and anti- patent sides.

1. Risk: Patents as Cause versus Mitigation

One boundary object discovered in discourse is risk. Pro-patent corporations see software patents as a method of mitigating risk, while anti-patent software engineers see them as a source of risk. One forum user argues that the existence of software patents greatly heightens the exposure to risk of ordinary software engineers:

> You might have a better idea than the last guy, but chances are your idea overlaps with his a little. If your new idea has some similarities with an exiting [sic] patented idea, implementing it could be risky. If engineers did a patent search for every idea they had, they would find some overlap with existing patents every single time. (dustbyrn, 2011).

Here, the user is arguing that the existence of software patents create heightened legal risk for a software engineer, even if that engineer is not intending to infringe on patents. They see it as inevitable that any idea would conflict with some patent. Another user argues that "I think the worst practical thing about software patents is that they create uncertainty… I don't see how patents or any similar legal tools will ever be anything but a tool for those with vast resources to crush competitors…" (Silhouette, 2011). Again, software patents are viewed as a source of uncertainty, which causes risk. Programmers see it as inevitable that one's work will conflict with others, given the broad nature of some software patents. This clashes with the hacker ethic value of relentless iteration as software patents produce a looming shadow of risk that increases as the project grows.

Pro-patent parties instead see software patents as a way to mitigate risk, especially the risk of losing monetary investment into the development of a new technological capability. IBM states that

> The purpose of research and development in any technology is to gain an advantage over your competitor. But if your competitor can legitimately copy the fruits from your R&D and can create a product that can compete head-on with your product while you are still trying to build a market for the product, then you've lost. (Neukom, 1994)

Pro-patent parties see patents as a way to formalize and reinforce the competitive advantages that have been created by innovation. Rather than let some other company benefit from your invention, patents provide a way to legally capture that value. It de-risks their investments. This aligns with the value of corporate pragmatism developed earlier. Pro-patent parties see patents as just a tool to be wielded to reduce risk, while anti-patent parties do not see them as a tool but as an external threat.

Something that is clear in the differing interpretations of risk is that the power differential between the two parties is influencing them. Whereas anti-patent parties express hopelessness and futility in response to the current system, pro-patent parties look to how they can use the system to their advantage. The pro-patent advocates are representing an incumbent system, and the anti-patent activists are trying to reform the system. This creates an inherent power differential even beyond the economic power of large pro-patent companies. In order to understand this power differential, I draw an analogy to the passage of the Fair Labor Standards Act of 1938 (FLSA). The FLSA established many groundbreaking labor protections in the United States, including a minimum wage and the prohibition of child labor. The fight for the passage of the law was long and contentious. Industry leaders and some congressional representatives supported the existing lack of labor protections (Grossman, 1978). In some instances, the discourse of the reformers exhibited a notion of powerlessness. One note passed to Roosevelt, an advocate of the law, read:

> I wish you could do something to help us girls....We have been working in a sewing factory,... and up to a few months ago we were getting our minimum pay of $11 a week... Today the 200 of us girls have been cut down to $4 and $5 and $6 a week. (Roosevelt, 1936, pp. 624-625).

The way that the reformers express a sense of futility in regards to wages is similar to the way that some software engineers express a sense of futility about software patents. This similarity proves that the differing perceptions of risk in the patent debate are not solely a result of different underlying values. Rather, in any reform movement, the reformers have less power, and that will be expressed in the discourse they produce. Language around reform movements such as the software patent debate must be looked at through that lens. Reexamining the discourse around risk described previously, the feeling of helplessness that software developers describe when faced with software patents may not only be a representation of the hacker values of freedom or innovation, but instead inherent to every reform debate.

2. Sources of Innovation: Top-Down versus Bottom-Up

Another boundary object is understanding how innovation is cultivated. Both sides speak of wanting to increase innovation, yet they view the underlying sources of that innovation as different. The anti-patent parties see innovation as something that is inherent to the work of programmers; that is, that freedom from intervention allows innovation room to grow. They see innovation as a grassroots process cultivated from the bottom-up. They believe invention is the result of curiosity and problem solving for its own sake. One user describes this in a near-mythological fashion:

> We who invent things, we who solve problems in new ways, don't do so because we can get a patent. We solve the problem because when you put a problem in front of us, we can't stop ourselves from solving it. It haunts us. We dream about it. (dustbyrn, 2011)

This is an expression of the hacker ethic value of diligence in work for its own sake. Another user sees software patents as being in direct opposition to innovation because they take away from the creativity that drives it: “As we've seen, software patents stifle creativity and innovation; reward those with money rather than creative energy and ideas.” (Writersglen, 2011). Again, innovation is seen as something inherent. It is seen as something that exists in nature and is not created by some external power. The anti-patent view of risk encapsulates an understanding of innovation that is inextricably tied to the hacker ethic. Anti-patent concepts of innovation are modeled off of the tireless, passionate work style of the early computer pioneers.

Pro-patent parties see innovation as something that can be imposed from the top-down. Microsoft expresses that patents actually stimulate innovation: “...we believe that the existing system can mature in a fashion that effectively achieves the constitutional goals of stimulating and protecting innovation in a competitive context." (Neukom, 1994). They see patents as achieving a goal of promoting innovation by providing the inventor with protection for the technology, therefore providing monetary incentive for the invention. This is in line with the corporate value of pragmatism on the organizational level. Where hackers may sentimentalize or mythologize the process of innovation, businesses such as Microsoft view it in a more detached manner. IBM's statement also expresses this sentiment: "As the industry matures and competition from overseas increases, patents will be the key to protecting the most valuable US-originated innovations." (Siber, 1994). This statement also brings up the idea of patents as a protective force that incubates innovation. Patents are viewed as a pragmatic tool for promoting and protecting innovation.

V. Conclusion

Software patent reformers and software patent supporters communicate differently about how software patents influence risk and about where innovation comes from. The reformers draw upon the hacker ethic in interpreting these concepts, while supporters draw on the values fostered by large corporate structures. With these differences in communication identified, software patent reformers can adjust their discourse to better communicate their ideas. Instead of focusing on the idea of patents solely as sources of risk, they can acknowledge that they can be used to hedge against risk. They can then communicate the fact that those patents-as-hedges actually create downstream risk for small and large corporations alike. They can also draw upon the language of the patent supporters in regards to innovation. Instead of presupposing that innovation arises on its own, they can argue the viewpoint that innovation can be fostered from the top-down. They can point to examples of policies such as anti-patent-troll laws that have increased innovation via a top-down policy. By attempting to communicate reform ideas using the language and concepts of the pro-patent organizations, reformers can hopefully communicate their ideas more persuasively.

A limitation of this approach is that the persuasiveness of the reformers’ arguments may not matter because the cold economic realities of the situation could prevent reform through discourse alone. However, as seen with the Fair Labor Standards Act analogy, reform can still be achieved despite a power gap between the reformers and the incumbents. With a better understanding of the values of corporate proponents of patents, the reformers can hopefully make progress in reform efforts.

Works Cited

Appel, I., Farre-Mensa, J., Simintzi, E. (2019). Patent trolls and startup employment. *Journal of Financial Economics, 133*(3), 708-725. https://doi.org/10.1016/j.jfineco.2019.01.003.

Athreye, S.S., Fassio, C. & Roper, S. Small firms and patenting revisited. *Small Bus Econ* 57, 513–530 (2021). https://doi.org/10.1007/s11187-020-00323-1

Baker, B. D. (2020). Sin and the hacker ethic: The tragedy of techno-utopian ideology in cyberspace business cultures. *Journal of Religion and Business Ethics 4(2),* 1-28. https://via.library.depaul.edu/jrbe/vol4/iss2/1

Brown, J. J. (2008). From Friday to Sunday: The hacker ethic and shifting notions of labour, leisure and intellectual property. *Leisure Studies*, *27*(4), 395–409. https://doi.org/10.1080/02614360802334922

Burton, D. (1996). Software developers want changes in patent and copyright law. *Michigan Telecommunications and Technology Law Review,* 2(1), 87-91. https://repository.law.umich.edu/mttlr/vol2/iss1/4/

Chandler, A. (2003). "The changing definition of and image of hackers in popular discourse." *Cyberspace Crime*, edited by Wall, D.S., Routledge, 2003, 121-143.

Chesbrough, H., Lettl, C. and Ritter, T. (2018). Value creation and value capture in open innovation. *Journal of Product Innovation Management*, 35, 930-938. https://doi.org/10.1111/jpim.12471

Cole, K. L., Haverfield, M. C., & Choate, S. D. (2024). Actor-Networks in political moral conflict: a case study of an online gun control debate. *American Behavioral Scientist*. https://doi.org/10.1177/00027642241240333

dustbyrn. (2011, May 16). *We who invent things, we who solve problems in new ways, don't do so because we can get a patent* [Comment on the online forum post *Poll: Do you support software patents?*]. Hacker News. https://news.ycombinator.com/item?id=2552740

Freeman, J. & Engel, J. (2007). Models of innovation: startups and mature corporations. *California Management Review.* 50, 94-119. http://dx.doi.org/10.2307/41166418

Griesbach, R., Camarota, A. (2016). Putting down roots at the patent office. *United States Patent Office.* https://www.uspto.gov/learning-and-resources/newsletter/inventors-eye/putting-down-roots-patent-office

Grossman, J. (1978). Fair labor standards act of 1938: Maximum struggle for a minimum wage. *Department of Labor*. https://www.dol.gov/general/aboutdol/history/flsa1938

Himanen, P. (2001). *The Hacker Ethic and the Spirit of the Information Age*. Random House.

Intellectual Property Owners Association. (2024). Top 300 organizations granted U.S. patents in 2023. https://ipo.org/wp-content/uploads/2024/01/2024-Patent-300-IPO-Top-Patent-Owners-List.pdf

Kamdar A., Nazer D., Ranieri, V. (2015). Defend innovation how to fix our broken patent System. *Electronic Frontier Foundation.* https://www.eff.org/document/defend-innovation-how-fix-our-broken-patent-system

Karakashian, S. (2015). Software patent war, the effects of patent trolls on startup companies, innovation, and entrepreneurship. *Hastings Business Law Journal*, 11(1), 119-156. https://repository.uclawsf.edu/cgi/viewcontent.cgi?article=1035&context=hastings_business_law_journal

Kirkpatrick, G. (2002). The Hacker ethic and the spirit of the information age. *Max Weber Studies*, *2*(2), 163–185. http://www.jstor.org/stable/24579606

Millien, R. (2023). Software-related U.S. patent grants in 2022 remained steady while Chinese software patents rose 8%. *IPWatchdog.* https://ipwatchdog.com/2023/03/28/software-related-u-s-patent-grants-2022-remained-steady-chinese-software-patents-rose-8/id=158395/#

Neukom, W. (1995). IBM Statement on software patents [speech transcript]. James S. Huggins. https://web.archive.org/web/20040811200450/http://www.jamesshuggins.com/h/tek1/software_patent_ibm.htm

Pruzan, P. The question of organizational consciousness: Can organizations have values, virtues and visions?. *Journal of Business Ethics* 29, 271–284 (2001). https://doi.org/10.1023/A:1026577604845

Roosevelt, F.D. *Public Papers and Addresses*, Vol. V New York, Random House, 1936), pp. 624-25, quoted in Grossman, J. (1978). Fair Labor Standards Act of 1938: Maximum Struggle for a Minimum Wage. *Department of Labor*. https://www.dol.gov/general/aboutdol/history/flsa1938

Schwarz-Plaschg, C. (2018). The power of analogies for imagining and governing emerging technologies. *Nanoethics, 12(2),* 139-159.

Siber, V. (1994). Microsoft statement on software patents [speech transcript]. James S. Huggins. https://web.archive.org/web/20041027231007/http://www.jamesshuggins.com/h/tek1/software_patent_microsoft.htm

Silhouette. (2011, May 16). *I think the worst practical thing about software patents is that they create uncertainty. If there is one thing that* [Comment on the online forum post *Poll: Do you support software patents?*]. Hacker News. https://news.ycombinator.com/item?id=2552740

Writersglen. (2011, May 16). *I owned and managed a small software development company for nearly 25 years. We produced many creative educational and consumer* [Comment on the online forum post *Poll: Do you support software patents?*]. Hacker News. https://news.ycombinator.com/item?id=2552740

Zivojnovic, O. (2015). Patentable subject matter after Alice—distinguishing narrow software patents from overly broad business method patents. *Berkeley Technology Law Journal,* 30(4), 807–862. https://www.jstor.org/stable/26377742