‘The granting of software and e-commerce business method patents is inconsistent with open and co-operative development of the internet. The grant of such patents will inhibit the future development of new and innovative web technologies and threatens to destroy the Open Source Software movement’.

Do you agree? Discuss with reference to specific issues and examples.

I. Introduction

The internet is profoundly reshaping the manner in which we are interacting with technology in our world. It has transformed not only the way in which we share and communicate information but also how we locate, learn and explore different types of media online. Fundamental to the development of the internet is the principle of collaboration that exists between millions of different people scattered across the globe - who are able to connect, deliberate, ponder and solve complex problems which would have otherwise remained unsolvable. It is this organic and multi-lateral cooperative effort which has allowed researchers and entrepreneurs to innovate and create commercially viable applications which have not only increased the dissemination of media on the internet, but also improved the manner in which the internet is evolving.

It is apparent that in order for the continual expansion of the internet to continue there must be persistent and increased collaboration and development between all users. It was collaboration which lead to the formation of the internet during its initial development as far back as the 1950’s[1], and it was collaboration which became fundamentally important during the early 1980’s and 1990’s when people began to realise the increased need to find and organise file and information structures. It was also at this point that the view of software development changed radically. Software was no longer being viewed as a pure mathematical algorithm but rather as a medium that was critically important to business success, and many technology companies began to question why software was not been offered the same legal protection as other industries. Companies such as Microsoft and Intuit were pushing for increased protection of intellectual property rights on software in order to protect their commercial business strategies and facilitate innovation. During the same era, Richard Stallman created the Free Software Foundation (FSF) to openly oppose this drastically changing trend. Stallman despised both the concept of proprietary software and that of intellectual property rights over software, rather proposing that all software be ‘free’ – a definition he characterized as *‘free to use, read, modify and redistribute without any legal repercussions’*[2]. Stallman’s arguments were almost entirely dismissed by many software companies at the time simply because ‘free’ was not ‘commercially viable’.

Thus, this paper endeavours to explore the changing attitudes between proprietary and open source software[3] and its evolution in relation to the internet. It will seek to focus on the development of patent law in the United States primarily because it was the first legal system in the world to allow increased intellectual property rights over software and in doing so, ultimately opened the floodgates for the rest of the world to follow. It will consider the

[1] Wikipedia, History of the Internet (2007), http://en.wikipedia.org/wiki/History_of_the_Internet Viewed 2nd Oct 2007
[2] Dwheeler, History of Unix, Linux and Open Source/Free Software, http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/history.html, Viewed 2nd Oct 2007
[3] For the purposes of this paper, proprietary software is software which is sold as closed source and for-profit. Open-source software is software that is made readily available and its source code is fully published.

implications of open source software and discuss whether the attitudes in relation to open source software have changed since the early development of the internet. The paper will also attempt to explore the juxtaposition between protecting and rewarding inventors with patent protection versus unfairly restricting other innovators from developing and advancing technological processes through open source software. Finally, it will conclude by discussing whether a rational solution can be discovered which protects inventors while also allowing the open source community to freely evolve.

II. Patents

The fundamental purpose of a patent is an impartial grant of an exclusive set of rights which provide the patent holder with a monopoly over the related subject matter. The majority of patent law revolves around the issues of what is patentable, whether the specification provided in the original filing is adequate, whether the monopoly claimed is supported by the relevant and appropriate documentation and whether the supposed invention is in fact novel and inventive when measured against the 'prior art' – that which was already known in the relevant field at the time the patent was sought[4]. While these fundamental aspects are standard feature of all non-software related patents, the development and approval of software patent law has only very recently been established and the requirements differ to some extent. The precedent set by the 1981 US Supreme Court in the case of *Diamond v Diehr*[5] was the first instance in which the US courts had instructed the United States Patent and Trademark Office to grant a patent for an invention in which computer software was utilised. The decision set out in this case created confusion for computer programmers because there was no clear definition provided by the court as to whether software related inventions were actually patentable. The resulting accepted interpretation at the time was that software in isolation was not patentable but innovations which used software as part of larger process were.

The US law remained stagnant with this definition for almost a decade, and it wasn't until the mid 1990s that a clearer ruling on software patentability evolved. In 1994, the Court of Appeals for the Federal Circuit (CAFC) built on the decision from *Diamond v Diehr*[6] and stated in *In re Alappat*[7] that *"programming creates a new machine, because a general purpose computer in effect becomes a special purpose computer once it is programmed to perform particular functions pursuant to instructions from the program software"*[8]. This decision set an entirely new legislative decree for software patents and the intellectual protection of software in general. Despite the significance of this decision, it was not until 1998 that CAFC made the single most important ruling in relation to software patents today, by removing the archaic *'business method exception'*. The *'business method exception'* was created in *Hotel Security Checking Co. Vlorraine Co* in 1908[9] and was implemented in order to prevent 'methods of doing business' being patentable subject matter. It was consequently invalidated by CAFC in 1998 when ruling on the *State Street Bank & Trust v Signature Financial Group*[10] case. *State Street* attempted to have *Signatures* patent invalidated on the basis that it was a mathematical algorithm and was also a business method. CAFC rejected

[4] IP Australia, Patents, http://www.ipaustralia.gov.au/patents/what_index.shtml, Viewed 2nd Oct 2007
[5] *Diamond v. Diehr*, 450 U.S. 175, 185 (1981)
[6] Ibid 17
[7] *In re Alappat*, 33 F.3d 1526, 1537 (1994)
[8] Ibid 18
[9] *Hotel Security Checking Co. v. Lorraine Co.*, 160 F. 467 (2d Cir 1908)
[10] *State Street Bank v. Signature Financial Group*, 149 F.3d 1368, 1370 (1998)

State Streets argument and invalidation claim, which it suggested was based entirely on the contention of '*unpatentable abstract ideas*'[11] and subsequently negated the *'business method exception'* entirely. Following the *State Street* decision, it became clearly apparent that computer software and data structures were now considered patentable subject matter in the United States. The US Patent and Trademark Office attempted to outline some *'computer related examination guidelines'*[12] for software patents with the basis of these guidelines being that the invention must produce a *'useful, concrete and tangible'*[13] outcome. Unfortunately, these guidelines were, and still are, easily fulfilled during the drafting process of software patents and a flood of new software patents emerged as a result.

Although there is ample evidence available to suggest that most computer software programs are statutory in nature, the typical requirements for patentability still remain. The most important prerequisite for a software patent is that the invention defined within, or by, the software program is non-obvious – a determination made by assessing whether the invention would have been obvious to a person of ordinary skill and knowledge in computer programming at the time of filing[14]. This fundamental prerequisite of software patents is also the greatest limitation in the patentability process, as most patent examiners are unable to effectively research all of the prior art, and subsequently validate patents which should not have been certified. As a result, the continuous flow of pre-existing and recently filed 'valid invalid software patents' are causing serious problems and concerns for all of the free source, open source and even proprietary software communities since those accused of patent infringement must conduct their own prior art research in order to determine whether or not the patent is invalid. While it is argued by proponents of the patent system that owners of software patents are disadvantaged because of the ease at which the validity of patents can be questioned, the entire process obviously requires substantial review and improvement.

Many supporters of open source software have actually suggested that the US Patent legislation should replicate laws implemented by the European Patent System[15]. The European system specifically states that computer related inventions are not patentable[16], although it does allow member states to interpret and dictate this directive. As a result, many member states have simply allowed software patents to be valid which has resulted in thousands of patents being accepted by the European system, and enraged many enterprise software companies[17]. The European Patent Office (EPO) argues that the European Patent Convention[18] does not ban software programs completely but rather specifically defines what is, and is not, patentable. They indicate that algorithms by themselves are definitively not patentable for example, but an algorithm contained within a computer program which solves a technical problem is. The ambiguous definition and lack of clarity in the European Patent

[11] *Idid 22* at 1374.
[12] United States Patent & Trademark Office, Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility, http://www.bitlaw.com/source/uspto/StatutoryGuidelines.pdf, Viewed 4th Oct 2007
[13] Ibid 10 at 1373-74
[14] Ibid 12
[15] Tom Yager, *Software Patents Set Sail, http://www.infoworld.com/article/03/10/03/39OPcurve_1.html*, Viewed 4th Oct 2007
[16] Ibid 16
[17] *EU Bows to Business, Postpones Software Patent Vote*, http://www.zdnet.co.uk/tsearch/court+eu+patent.htm, December 17 2004, Viewed 4th Oct 2007
[18] Art. 52(2)(c) & 52(3), *Convention on the Grant of European Patents (European Patent Convention),* http://www3.european-patent-office.org/dwld/epc/epc_2002_v1_bm.pdf, Viewed 5th Oct 2007

Convention is not unlike the historical decision from *Diamond v Diehr*[19] in the United States, where many European software developers are now simply confused as to what is, and what is not patentable – a suggestion the EPO discounts.

Despite these flaws, there are increasing numbers of people[20] who are of the view that the European method of handling patents would assist the US in reducing the flow of 'valid invalid software patents', and reward and recognise truly novel inventions with patent protection. Members of the proprietary software industry are actively dismissing this suggestion. They claim that the US System only just provides the minimum requirements to protect innovation[21] and anything less would severely reduce their competitive advantage in industry, and render the fundamental purpose of the patent system – the granting of a temporary monopoly - useless. While it is unclear whether the European Patent System is, or would be, more effective than the US System in reviewing software patents, it does illustrate that there are more systemic problems in the US System compared to the European one, due to common law rulings on software patents.

III. Open Source Software

Open source software is predominately a method of creating and distributing software[22]. According to the Open Source Initiative[23], open source doesn't just mean that a person has fundamental access to the code. The distribution terms of open source software can also include many different criteria ranging from free distribution initiatives to source lock down terms, where any modifications made to the source code must always be openly provided. Thus it is clear open source software is not rule-free software, but rather software in which certain restrictions are imposed on users of the code. Founder of the Free Software Foundation, Richard Stallman, has consistently dismissed the fundamental premise of 'open source' as he believes software should be entirely 'free'. He suggests that *"free software is motivated by an idealistic goal: spreading freedom and cooperation. I want to encourage free software to spread, replacing proprietary software that forbids cooperation, and thus makes our society better"*[24], a view that hinges on the extremist end of open source collaboration and is perhaps only effective in certain development environments.

While Stallman may want all software to be free, the success of open source software from both a commercially viable and development model cannot be denied[25]. It is often acknowledged by even proponents of open source software models, that there is immense value in the open source system, by being able to generate results and facilitate collaboration in order produce and develop software. This was illustrated clearly by the release of the

[19] Ibid 5
[20] Open Letter, *An Open Letter to the European Parliament Concerning the Proposed 'Directive on the Patentability of Computer-Implemented Inventions 2003'*, http://www.researchineurope.org/policy/patentdirltr.htm, Viewed 6th Oct 2007
[21] Andy Reinhardt, *Inventing a Better Patent Law; Can Europe spur software innovation while safeguarding intellectual property?*, http://www.businessweek.com/magazine/content/03_48/b3860058_mz054.htm, Viewed 6th Oct 2007
[22] David S. Evans, Open-Source Software Poses Challenges for Legal and Public Policy, http://www.wlf.org/upload/013103LBEvans.pdf, Viewed 8th Oct 2007
[23] Open Source Initiative, The Open Source Definition, http://www.opensource.org, Viewed 8th Oct 2007
[24] Richard Stallman, Copyleft: Pragmatic Idealism, http://www.fsf.org/philosophy/pragmatic.html, Viewed 8th Oct 2007
[25] S Weber, The Success of Open Source Software, Harvard University Press, 2004

Hallow Documents[26] - a set of highly confidential documents which were leaked to the press which outlined Microsoft's plan to disrupt open-source software entirely, and also its strategy to *"deny open source initiatives such as Linux entry into the market"*[27]. Other highly publicised illustrations of open source successes included the response by the Open Source Initiative[28] foundation to AOL letter regarding the Mozilla project and its future.

While these examples provide illustration that open source systems and applications are becoming increasingly viewed as a cost-effective, secure and reliable way of implementing systems in the corporate environment[29], there are also a number of challenges that open source software have yet to overcome. The most obvious challenge comes from Microsoft and its so-called "FUD" tactics, or Fear-Uncertainty-Doubt strategies, which attempt to undermine the popularity of open source systems and their safety online and in corporate environments. This is particularly evident through Microsoft's continued argument that open-sourced systems allow hackers to study weaknesses in source code structure, thereby gaining valuable insight into the organization of code and consequently infiltrating companies who use the software to gain access to their data. Conversely, open source commentators argue that since the software is 'open', any programmer can review and fix security flaws much faster than proprietary developers, and thereby plug security holes quickly and efficiently. A strong example of open source collaboration is the internet based Mozilla browser project, which has attracted contributions from millions of software developers around the world and is now the 2nd most popular browser on the planet after Internet Explorer[30]. Despite this example, the critical challenge to open source software is not defending its manner of operation but rather the increased threat from software patents that could potentially ruin the open source movement and halt the innovative process.

The issue of software patents has become a critical problem for the continued development and expansion of the open source software community, predominately due to the fact that open source software relies heavily only on copyright protection and licensing. The United States Patent and Trademark Office have been issuing patents for software development at unprecedented rates[31]. A possible explanation for this substantial increase in patent applications stems from computer programmers not wanting to rely solely on copyright law as their only means of intellectual property protection, particularly because US Copyright law requires copyright owners register their works in order to litigate[32]. Additionally, software developers are wary of the subjective nature of copyright interpretation in respect of the US lead idea-expression dichotomy[33] contained within the US Copyright Act[34]. The idea-expression dichotomy suggests that *"ideas that are the fruit of an author's labours go into the public domain, while only the author's particular expression remains the author's to*

[26] Wikipedia, Halloween Documents, http://en.wikipedia.org/wiki/Halloween_documents, Viewed 8th Oct 2007
[27] Ibid 26
[28] Open Source Initiative, An Open Letter to AOL, http://opensource.org/pressreleases/aol-letter.php, Viewed 8th Oct 2007
[29] Ibid 2
[30] Mozilla, Mozilla Foundation, http://www.mozilla.org, Viewed 10th Oct 2007
[31] Kenneth Nichols, The Age of Software Patents Abstract, http://doi.ieeecomputersociety.org/10.1109/2.755002, Viewed 10th Oct 2007
[32] Copyright Law, Registering your Work, http://www.copyright.gov/help/faq/faq-register.html#register, Viewed 10th Oct 2007
[33] Edward Samuels, The idea-expression dichotomy in copyright law, http://www.edwardsamuels.com/copyright/beyond/articles/ideapt1-20.htm, Viewed 10th Oct 2007
[34] The United States Copyright Act of 1976

control"[35]. The application of the idea-expression dichotomy is particularly prevalent in relation to open source software programs in the instance that the abstraction-filtration-comparison test is applied. The *"abstraction, filtration and comparison"* test was developed during the *Computer Associates International Inc v Altai Inc*[36] case. The US Copyright Act[37] clearly indicates that computer programs are to be protected as literary works, and non-literal structures of computer programs are to be covered analogously in 'other literary' works. The test attempts to separate the expression of an idea from the idea itself, and it does this by identifying protectable elements of the expression from the unprotectable elements. Many have commented that the *"the abstraction-filtration-comparison test eliminates protection for computer programs by entirely filtering out not only the individual elements of computer programs such as software objects but also the compilation of selection and arrangement expression that is the program's structure, since both are designed with efficiency in mind"*[38]. The difficultly for software developers to rely on this test during copyright litigation is that it is difficult to interpret, and only provides protection for the literal component of the program – a potential problem in open source software development. Consequently, the outcome of this case has encouraged some software developers to seek patent protection as a more secure way of defending their underlying software programs as opposed to any reliance on copyright law, and thereby actually encouraging, perhaps unknowingly, proprietary software development.

Additionally, the other obvious problem created by the application of this law, is that any businesses attempting to develop a commercial open source application may be unknowingly infringing patents and risk consequential litigation. As many computer programmers are shying away from a reliance on copyright protection and filing patents, it makes it exceeding difficult for any commercial enterprise to effectively develop open source applications without becoming embroiled in a patent litigation suit. It has been commented[39] numerous times in the US that the patent system is entirely too broad in its approach of issuing software patents for concepts which have existed in some prior art form. A recent review[40] of the US Patent and Trademark office has suggested it has systemic internal problems and is unable to handle the sheer volume of patent pending applications being filed, the complex and technical nature of the work and the lack of experienced staff examining and validating software patents. In addition, there has also been increasing observation that Patent Attorneys are technically rewording prior art in order to achieve patentable subject matter which has already existed in some prior form[41]. A clear illustration of the problem can be identified via U.S. Patent No. 6,330,551 issued in 2001 which was granted to protect '*automated online dispute settlement systems*'. The patent is effectively for an internet based computerized system of dispute resolution which allows each party to resolve their disputes electronically by entering in a monetary sum to settle the claim; a computer generated algorithm then automatically calculates the dispute payout to each party. The difficultly with this patent is

[35] Edward Samuels, The Idea-Expression Dichotomy in Copyright Law, http://www.edwardsamuels.com/copyright/beyond/articles/ideapt1-20.htm, View 11th Oct 2007
[36] *Computer Associates International Inc v Altai Inc 23 USPQ2d 1241 (1992)*
[37] Ibid 32
[38] John W.L. Ogilvie, Defining Computer Program Parts Under Learned Hand's Abstractions Test in Software Copyright Infringement, http://digital-law-online.info/misc/ogilvie.htm, Viewed 11th Oct 2007
[39] The Washington Post, Backlog, Quotas Overwhelm Patent Examiners, http://www.washingtonpost.com/wp-dyn/content/article/2007/10/07/AR2007100701199_pf.html, Viewed 11th Oct 2007
[40] Ibid 37
[41] A Duffus, The Proposal for a Directive on the Patentability of Computer-implemented Inventions, http://en.wikipedia.org/wiki/Directive_on_the_patentability_of_computer-implemented_inventions, Viewed 14th Oct 2007

that there is definitively obvious prior art documented by a paper written by RM Issaac on *"Theories and Tests of 'Blind-Bidding' Dispute Resolution"* which was published in 1989 in the Journal of Economics.

There are countless other examples similar to the one provided above and it is evident that the issuing of such patents, which have a broad and unqualified concept specification are definitively negative to the continual expansion of the open source software movement on the internet. The difficulty stems from the fact that the inherent nature of open source software is its intrinsic ability to facilitate collaboration and take free forms of expression from any contributor that wants to devote their time to the project. The internet has accelerated the rate at which programmers can collaborate and build software across distributed networks and decentralised development boundaries. If a patent exists for a fundamental idea behind an open source software project being created, then the collaborative process on this project is damaged. Regardless of the project owner's legal right to the copyright of the code, the project is rendered useless without a valid licence from the patent holder. Evidently, this causes many developers to stop contributing to the project for their own indemnity purposes and the open source initiative breaks down entirely. Furthermore, while open source copyright holders could attempt to have a patent invalidated in court, such processes are generally extremely expensive and developers generally do not have the funds or resources to undertake such a process. Thus in this instance, the collaborative and collective efforts of software developers in the open source community are severely disrupted by the existence of patents.

However, while it is obvious from the previous example that patents are hindering the open source software effort, the alternative to patent legislation could be worse. Currently, many patent holders offer free licences to open source collaboration projects and provide free licensing and distribution rights in respect of their IP portfolios[42]. Open source and free software proponents often forget that patents spur innovation forward because the invention needs to be fully disclosed to the public. The ability to review and improve upon existing technologies then further spur forward the creative process, and are positive for competition. Additionally, the removal of software patents in their entirely would result in an increased reliance on copyright law and trade secrets - both of which would be significantly worse for the software industry and open source community in general[43]. Furthermore, if patents were eliminated entirely, it would make it extremely difficult for small businesses to gather external funding, particularly when venture capital financing look heavily to patent applications before funding small business.

IV. Filling the Gap & Conclusion

It is clear that there is a gaping divide between proponents of free and open source software and patent law. Open source commentators are of the firm belief that patent law is hindering the innovative development processes behind software applications and the internet in general. There has been extensive discussion in this paper that open source software cannot coexist with patent litigation simply because of the legal differentiation between the protection of copyright and expression, and the monopoly rights associated with patent law over the relevant subject matter. In some regards, it is acceptable to conclude that free and

[42] Notably IBM and Microsoft are freely licensing their IP to legitimate users.
[43] William M. Landes & Richard A., The Economic Structure of Intellectual Property Law, Harvard College, Page 294, 2003

open source proponents are hypocritical in their views on patents since they express their desire to preserve collaboration and freedom of expression, yet limit the abilities of an individual inventor to protect his or her idea. Equivalently, many of their arguments illustrate critical weaknesses in the US Patent and Trademark system. It is clear that the current system is not working effectively and 'valid invalid software patents' are consistently being approved by patent examiners. However, to suggest that the abolishment of the patent system as a whole is required to ensure that the internet is protected and innovation can continue - is not an effective solution to the problem either. Many academics have written about systemic problems associated with the US patent office and a plethora of plausible reform structures which would work[44]. Reforms of the patent system would benefit all sections of patent law and not just software and business methods. In addition, encouraging enterprise business to licence their IP portfolios to open source initiatives are actually beneficial, since there are economic benefits if the software is used commercially.

It is important to remember that the foundations of the internet were built on the collaborative efforts of many, and it is acceptable conclusion that only with this continued cohesion can the internet continue to develop into the future. The benefits of open source software have been well documented throughout this paper, and equivalently, so have those of software patents. While it is accepted that fundamental problems exist with the continuing trend of validating 'invalid' patents, the system is capable of being reformed with external input, database improvement and increased information dissemination. Open source software can and will continue prosper into the future despite looming patent issues, as it has stubbornly proven that it can even in the current software patent environment. Additionally, large scale open source projects such as Mozilla and Linux only promote the efforts of open source development and future open source initiatives. Thus, while it is accepted that patent reform is necessarily in order for open source initiatives to continue to proposer, the abolishment of the patent system in it's entirely is nonsensical and does not seek to promote, harbour or extend innovation in any form – whether it be on the desktop or the internet.

[44] Robert Merges, As Many as Six Impossible Patents Before Breakfast: Property Rights for Business Concepts and Patent System Reform, http://papers.ssrn.com/sol3/papers.cfm?abstract_id=180748, Viewed 16th Oct 2007

Bibliography

1. Wikipedia, History of the Internet (2007), http://en.wikipedia.org/wiki/History_of_the_Internet Viewed 2nd Oct 2007

2. Dwheeler, History of Unix, Linux and Open Source/Free Software, http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/history.html, Viewed 2nd Oct 2007

3. IP Australia, Patents, http://www.ipaustralia.gov.au/patents/what_index.shtml, Viewed 2nd Oct 2007

4. *Diamond v. Diehr*, 450 U.S. 175, 185 (1981)

5. *In re Alappat*, 33 F.3d 1526, 1537 (1994)

6. *Hotel Security Checking Co. v. Lorraine Co.*, 160 F. 467 (2d Cir 1908)

7. *State Street Bank v. Signature Financial Group*, 149 F.3d 1368, 1370 (1998)

8. United States Patent & Trademark Office, Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility, http://www.bitlaw.com/source/uspto/StatutoryGuidelines.pdf, Viewed 4th Oct 2007

9. Tom Yager, *Software Patents Set Sail*, *http://www.infoworld.com/article/03/10/03/39OPcurve_1.html*, Viewed 4th Oct 2007

10. *EU Bows to Business, Postpones Software Patent Vote*, http://www.zdnet.co.uk/tsearch/court+eu+patent.htm, December 17 2004, Viewed 4th Oct 2007

11. Art. 52(2)(c) & 52(3), *Convention on the Grant of European Patents (European Patent Convention)*, http://www3.european-patent-office.org/dwld/epc/epc_2002_v1_bm.pdf, Viewed 5th Oct 2007

12. Open Letter, *An Open Letter to the European Parliament Concerning the Proposed 'Directive on the Patentability of Computer-Implemented Inventions 2003'*, http://www.researchineurope.org/policy/patentdirltr.htm, Viewed 6th Oct 2007

13. Andy Reinhardt, *Inventing a Better Patent Law; Can Europe spur software innovation while safeguarding intellectual property?*, http://www.businessweek.com/magazine/content/03_48/b3860058_mz054.htm, Viewed 6th Oct 2007

14. David S. Evans, Open-Source Software Poses Challenges for Legal and Public Policy, http://www.wlf.org/upload/013103LBEvans.pdf, Viewed 8th Oct 2007
15. Open Source Initiative, The Open Source Definition, http://www.opensource.org, Viewed 8th Oct 2007

16. Richard Stallman, Copyleft: Pragmatic Idealism, http://www.fsf.org/philosophy/pragmatic.html, Viewed 8th Oct 2007

17. S Weber, The Success of Open Source Software, Harvard University Press, 2004

18. Wikipedia, Halloween Documents, http://en.wikipedia.org/wiki/Halloween_documents, Viewed 8th Oct 2007

19. Open Source Initiative, An Open Letter to AOL, http://opensource.org/pressreleases/aol-letter.php, Viewed 8th Oct 2007

20. Mozilla, Mozilla Foundation, http://www.mozilla.org, Viewed 10th Oct 2007

21. Kenneth Nichols, The Age of Software Patents Abstract, http://doi.ieeecomputersociety.org/10.1109/2.755002, Viewed 10th Oct 2007

22. Copyright Law, Registering your Work, http://www.copyright.gov/help/faq/faq-register.html#register, Viewed 10th Oct 2007

23. Edward Samuels, The idea-expression dichotomy in copyright law, http://www.edwardsamuels.com/copyright/beyond/articles/ideapt1-20.htm, Viewed 11th Oct 2007

24. The United States Copyright Act of 1976

25. Edward Samuels, The Idea-Expression Dichotomy in Copyright Law, http://www.edwardsamuels.com/copyright/beyond/articles/ideapt1-20.htm, View 11th Oct 2007

26. *Computer Associates International Inc v Altai Inc 23 USPQ2d 1241 (1992)*

27. John W.L. Ogilvie, Defining Computer Program Parts Under Learned Hand's Abstractions Test in Software Copyright Infringement, http://digital-law-online.info/misc/ogilvie.htm, Viewed 11th Oct 2007

28. The Washington Post, Backlog, Quotas Overwhelm Patent Examiners, http://www.washingtonpost.com/wp-dyn/content/article/2007/10/07/AR2007100701199_pf.html, Viewed 11th Oct 2007

29. A Duffus, The Proposal for a Directive on the Patentability of Computer-implemented Inventions, http://en.wikipedia.org/wiki/Directive_on_the_patentability_of_computer-implemented_inventions, Viewed 14th Oct 2007

30. William M. Landes & Richard A., The Economic Structure of Intellectual Property Law, Harvard College, Page 294, 2003

31. Robert Merges, As Many as Six Impossible Patents Before Breakfast: Property Rights for Business Concepts and Patent System Reform, http://papers.ssrn.com/sol3/papers.cfm?abstract_id=180748, Viewed 16th Oct 2007