International Free and Open Source Software Law Review

Volume 01} Issue 01} July 2009

# Contents



ISSN: 1877-6922

Available online at: http://www.ifosslr.org

# Impressum

## Editorial Committee

This publication is managed by a rotating Editorial Committee. The membership of the Committee for this issue is as follows:

### Amanda Brock

Amanda Brock is General Counsel of Canonical, the commercial sponsor of the Ubuntu project. Having graduated with Honours from Glasgow University, Amanda went on to obtain a Masters in Comparative Jurisprudence from New York University Law School and a LLM in IT and IP law from Queen Mary and Westfield, University of London. She is author of E:Business; The Practical Guide to the Laws and has written extensively on commercial and IT law.

### Daniel Berlin

Daniel Berlin works in Washington, D.C., for Google as a member of its technical staff and as a lawyer. Prior to his work with Google, Berlin was an advisory engineer with IBM Research in New York, where he worked on open source projects such as Subversion and GCC. Berlin earned a law degree from the George Washington University School of Law and a bachelor of science degree in computer science from the University of Rochester.

### Michel Jaccard

Partner, BCCC Avocats, Geneva / Lausanne. Expertise in IT, IP and corporate transactions.

### Andrew Katz

Andrew Katz studied Natural Sciences and Law at Cambridge University where he graduated with honours in 1989. In 1991 he was called to the Bar, and in 1993 requalified as a solicitor. He moonlighted as a programmer during his studies at Bar School, programming in Turbo Pascal. He has released software under the GPL. He is currently a partner at Moorcrofts LLP, a boutique law firm in England's Thames Valley and advises a wide range of businesses on free and open source related issues. He has lectured and published widely on the subject.

### Florence Tung-Mei Ko

In 2005, Florence Tung-Mei Ko joined the Open Source Software Foundry (OSSF) at Academia Sinica as a Legal Specialist. Since then, she has been engaged in research of Free/Open Source Software licensing. She writes the "Legal Article" column for the OSSF Newsletter, and has published many articles related to Free/Open Source Software licensing issues in Chinese.

### Thomas Margoni

Thomas Margoni's current role is as a post-doctoral researcher at the University of Western Ontario. He previously read for a PhD at Trento Law School (Italy) and researched at UOC (Spain) on copyright related issues, especially reutilization of contents (contracts, exceptions and DRM). His background includes engagement with FOSS and Open Content issues. He has collaborated with Italian and Spanish Creative Commons legal teams on the drafting of licenses and on aspects as the relationship with collecting societies.

### Iain G. Mitchell QC

Chairman, Scottish Society for Computers and Law; Chairman, Scottish Lawyers' European Group; Chairman, Faculty of Advocates IT Group; Lecturer, Honorary Board of Lecturers, Institut für Informations-, Telekommunikations- und Medienrecht, Westfälische Wilhelms -Universtät, Münster; Freeman, Worshipful Company of Information Technologists.

### Carlo Piana

Independent lawyer specialising in Information Technology and Telecommunication Law and Free Software Advocate. Started by using GNU/Linux and became intrigued by the legal and philosophical implications of it. Serves as Counsel to the Free Software Foundation Europe and advises projects and companies active in Free and Open Source Software.

### Tomasz Rychlicki

Tomasz Rychlicki graduated from the University of Gdańsk, the Faculty of Law, Center of European Law. He also studied at Chicago-Kent College of Law in the LL.M. Program in International Intellectual Property Law. Tomasz currently works for PATPOL, Warsaw. He is a member of the Editorial Board (Copyright, Related Rights and Designs, including sui generis database right) at the Journal of Intellectual Property Law & Practice published by Oxford University Press, Oxford.

### Brendan Scott

Brendan runs a legal practice based in Sydney, Australia. Brendan is a founding member and a director of Open Source Industry Australia Limited. He is a past president of the NSW Society for Computers and the Law and a past editor of its journal. He has over 15 years of experience in Technology and Telecommunications law and has a special interest in open source and the law related to it.

### Ywein Van den Brande

Ywein Van den Brande (b. 1977) studied law, information science and economics at the Catholic University of Leuven and Groep T (Leuven, Belgium), the Université de Rouen (Rouen, France) and U.C.L.A. (Los Angeles, USA).
Ywein is an active member of the Free Software Foundation Europe and is a member of workgroups on information architecture and usability. Ywein is the author of several legal contributions on law and technology, and regularly speaks at conferences. He is a guest lecturer at Groep T and blogs on the CIF blog of the journal www.tijd.be.

### Mark Webbink

Mark Webbink is a Visiting Professor of Law and Executive Director of the Center for Patent Innovations at New York Law School. Webbink is also a Senior Lecturing Fellow at Duke Law School and has served as an Adjunct Professor at NCCU Law School. From 2000 to 2007 Webbink served in various capacities with Red Hat, Inc., including General Counsel, Deputy General Counsel for Intellectual Property, Senior Vice President and Secretary. Webbink presently serves on the board of directors of the Software Freedom Law Center.

**The editors wish to thank the Editorial Coordinators for their hard work and contribution to making the Review happen.**

## Editorial Coordinators

Graeme R. West

graeme@heliocentrik.net

Shane Martin Coughlan

shane@opendawn.com

## Contact

The IFOSS L. Rev. Editorial Committee can be contacted via email at:

ed-com@ifosslr.org

# Policies and informations

## Copyright and licensing statement



## Graphic design

The Editorial Committee wishes to thank Tomasz Politański Design for its logo and associated graphic design work.

http://tomaszpolitanski.com

## Publisher & sponsorship

IFOSS L. Rev. is published by its Editorial Committee, with financial and administrative assistance from NLnet Foundation (Science Park 140, 1098 XG Amsterdam,The Netherlands) and Mozilla Foundation (650 Castro Street, Suite 300, Mountain View, CA, 94041-2021, USA)

Please note that neither NLnet Foundation nor Mozilla Foundation accept correspondence on behalf of this publication. All correspondence should be directed to the Editorial Committee via email (see above).

## Editorial policies

IFOSS L. Rev. accepts articles for publication from qualified personnel based on the criteria available to view on its web site ( http://www.ifosslr.org ). Submissions are welcome from all, and authors are strongly encouraged to read the style and content guidelines available on the web site. The review operates an anonymous peer review system for articles as appropriate, and expects all authors to meet the highest standards of scholarship and integrity.

## Bibliographic information

The authors explicitly encourage libraries, archives and educational institutions to hold copies of IFOSS L. Rev. in their collections, in electronic and/or printed form. All users are advised that articles may occasionally be updated after publication, and as such, regular metadata harvesting is encouraged. Linking back to original copies on the IFOSSLR web site, where authoritative versions are archived, is strongly recommended. Please contact the Editorial Committee for further information on best practices.

ISSN: 1877-6922

## Second edition

The Second edition of IFLOSSLR is planned for December 2009. Suggestions for contributions will be welcomed by the Editorial committee and should be sent to:

ed-com@iflosslr.org

# Foreword and statement of purpose: an introduction to IFOSS L. Rev.

*by The Editorial Committee, coordinated by Iain G. Mitchell Q.C.*[a]

*(a) Iain G. Mitchell QC is Chairman of the Scottish Society for Computers and Law; Chairman of the Faculty of Advocates Information Technology Group; Chairman of the Scottish Lawyers' European Group; a visiting lecturer at the Institut für Informations- Telekommunikations- und Medienrecht at the Westfälische Wilhelms-Universität Münster, Germany; former Joint Editor of the E-Law Review; and a Freeman of the Worshipful Company of Information Technologists.*

**Abstract**
This editorial article by the Editorial Committee of the International Free and Open Source Software Law Review introduces the rationale behind the publication's establishment, its objectives and operating methods. The editorial also presents an overview of the articles which comprise the first issue.

**Keywords**
Law; information technology; Free and Open Source Software; International Free and Open Source Software Law Review

## Purpose

The world is changing. Old structures, old business models and old ways of thinking are breaking down. Such a world is, at once, full of challenges or, depending on your point of view, opportunities.

In his book, The Media Lab: Inventing the Future at MIT, Stewart Brand wrote:

> "Information wants to be free. Information also wants to be expensive. Information wants to be free because it has become so cheap to distribute, copy, and recombine - too cheap to meter. It wants to be expensive because it can be immeasurably valuable to the recipient. That tension will not go away. It leads to endless wrenching debate about price, copyright, 'intellectual property', the moral rightness of casual distribution, because each round of new devices makes the tension worse, not better."

Yet tension may be creative: it may lead to constructive change. Once it seemed that the future belonged to the large monopoly corporations which had as their business model, software as a commodity - a commodity to be expensively packaged, shrink-wrapped, shipped, and retailed. The source code was the crown jewels, to be guarded, and legally fenced to preserve its value.

But in a world of ever-increasing connectedness and convergence, where software can be downloaded or services supplied across the Internet at a unit cost which is as near to free as you can get, commoditising software in the most expensive way possible - by turning a virtual good into a physical good, no longer makes as much sense as it used to. Even the value of software as a commodity (however it is distributed) makes less and less sense when one realises that most of the end-users do not want software for what it is, but for what it does.

In such a market place, the large corporations' dominant positions, so important in bolstering their control of the software market, count for less if the customers, in place of having sitting on their desktops software they don't want, acquire the services they do want over the Internet. This new paradigm of service delivery makes it irrelevant what the software is, and helps opens up competition to a wider diversity of software models.

In the old days of the big corporations' unchallenged market dominance, there arose a movement which was as much aspirational and philosophical as it was grounded in market considerations - the Free and Open Source Software movement. Like most movements, it was not homogeneous: a few of its adherents prophesied the death of copyright and fancied themselves close to the epicentre of a cause which would shake the earth and bring down the heavens, but many more of its supporters saw clearly that the idea was profoundly market-oriented, pitting against the large anti-competitive monopolies a healthier and more vigorous market where software developers and users shared the source code and worked together in freedom to run, modify, distribute and redistribute programs. The old monopolistic corporations used copyright law to close the source code down and to restrict its use, but the new model was to use copyright to do the opposite, to open the source code up and freely to share it, and so, Richard Stallman coined the epithet 'copyleft' to describe this new use of copyright law.

Free and open source software is software which is covered by a licence very different from a traditional commercial software licence, in that it grants the users - under certain conditions - the right to use, study, modify and distribute the software. For these purposes, the user is granted access to the source code and the right (and sometimes even the obligation) to distribute the source code further afield.

New ideas need time to gain general acceptance and, to begin with, few of those in the wider business community were aware of the profound changes which were taking place, and many of those who were aware of the changes misunderstood them. By and large, business people sought legal certainty and what they perceived as reliability, so clung on to nurse for fear of something worse. They regarded words like 'copyleft' with suspicion; and they looked at the preamble to GPL 2 and saw a document which looked to them more aspirational than legal in nature: a Constitution for the State of Hackerdom rather than a serious business tool.

However, some more far-sighted businesses saw Free and Open Source software much more accurately as a valuable business tool, upon the back of which entire business models might realistically be built. They were the pioneers, but it would not be long before the rest of the world began to catch up. Governments and institutions around the world began to see considerable gains in respect of strategic independence, lowered costs and increased reliability through the use of Free and Open Source software. In the commercial world, the old proprietary software business models daily seemed less and less relevant to the modern market place, and Free and Open Source software broke through as a serious player. There was a definite sense that the tide had turned.

And yet now, years later, there is still baggage. Free and Open Source software, though now a robust business proposition, evokes for many, memories of the days when it was seen as crusade. Its proponents as well as its opponents still occasionally address the subject as though they were engaging in a philosophical debate. There are also differences of emphasis, and a whiff of sectarianism amongst the proponents of the various standard licences.

It is for others to address these theological disputes. It is, rather, the purpose of this Review to take one step back and look behind the rhetoric in a rigorous and objective fashion, to probe, to analyse, to question received wisdom, and to bring proper academic discipline to the study of Free and Open Source software in its legal and wider context.

Free and Open Source Software has become a serious player. It deserves serious analysis.

## Objectives

### (a) Scope

It is with this end in view that the Editorial Committee presents this first issue of the International Free and Open Source Software Law Review (*IFOSS L. Rev.* or *IFOSSLR*), in the hope and expectation that it will provide a centre of excellence for the very best in analysis of issues facing users and advisors in the development, deployment and governance of Free and Open Source software, recognising the importance of digital rights issues to the daily professional and personal lives of many of the Review's readers and the role that open solutions might play in their resolution. The Review aims to present the perspectives of those most experienced and knowledgeable in the field and to ask how there might be attained sustainable solutions which foster the growth and development of the marketplace.

However, it is by no means the intention of the editors that the Review should exist solely for those who have an attachment, whether ideological or merely pragmatic, to Free and Open Source software. It is recognised that much of the readership will be drawn from the wider world and will include the sympathetic, the merely curious, and the actively hostile. So, the Review does not exist to preach to the converted: the readership will be much wider than that; nor, in what is intended as a publication of integrity and intellectual rigour, is there any place for preaching. This is worth stressing in light of the history of the Free Software and the Open Source Software movements.

Both of those movements largely endorse similar software licences and pursue similar goals, but the two terms have historically tended to carried different emphases. Those who use the term "Free Software" stress the rights (or freedoms) which that software provides to its users, whereas users of the term "Open Source Software" focus on the perceived benefits of peer-to-peer software development. Put differently, Free Software emphasises the long term goal, Open Source Software emphasises the means to promote the long term goal of Software Freedom.

The Review does not endorse any one licensing model, focus or emphasis, but rather seeks, in an academically rigorous and objective manner, to increase the knowledge and understanding about the legal mechanisms used by all forms of Free and Open Source Software licences. It uses the term *Free and Open Source Software* to cover both Free Software and Open Source Software.

**b) Consolidation of knowledge**

IFOSS L. Rev. aims to provide a focal point for discussion of, and research into, the legal aspects of Free and Open Source software, and in doing so to enhance the level of understanding among legal professionals and researchers of key issues facing the industry. Existing discussion *fora* serve their purposes well, and it is not the objective of this publication to replace them. Rather, it is intended that IFOSS L. Rev. should become the place where promising new approaches and insightful analysis can gain a voice – a voice which will reach all those most involved and interested in the questions at hand.

**c) Professionalism of research and discussion**

In everything that the publication does, it will strive to achieve or surpass the highest standards of academic enquiry. It will be the first publication specialising in Free and Open Source software legal issues to insist upon such standards, and in doing so the publication will raise the level of analysis of such issues and provide a stimulus for new research and analysis.

**d) Marketplace and community relevance**

A final central objective of IFOSS L. Rev. is to maintain a close degree of connectedness to the fundamental processes driving the marketplace and community. This means that it must ensure that those with expertise in realms such as software engineering, community project management and business are brought 'into the body of the kirk'.

IFOSS L. Rev. will maintain ties to these crucial stakeholders through the diverse and changing composition of its Editorial Committee and the legal network it represents. It will also devote space in each issue to examination of pertinent issues by invited non-lawyers, allowing the publication's core legal audience to increase their awareness of upcoming issues and ground their legal knowledge in practical examples.

It will also recognise that professionals, both lawyers and non-lawyers who are experienced in the field will have had an opportunity to develop informed opinions about matters of concern or contention; and for this reason, the Review will also publish an occasional series of articles under the title *Platform*, which series will give scope for the expression of those opinions, but only where such opinions are informed, perceptive and likely to advance debate and discussion in the wider community.

## Scope of coverage and methods

The core topics covered by IFOSS L. Rev. include copyright, licence implementation, licence interpretation, software patents, open standards, case law and legislation. However, the review, as its name implies, is international in scope and there are no limitations as to the jurisdictions covered.

The Editorial Committee accepts proposals and submissions on any of these, or related, topics, though it does insist on maintaining the Review's core focus on the legal aspects of Free and Open Source Software. The subject matter of submissions must reflect this.

Each issue of IFOSS L. Rev. will include a combination of full-length research articles, case law reports, legislative review articles, book reviews and editorials covering these themes. Peer-review

is applied to the appropriate sections: for full details of the submission process, please see our website at http://www.ifosslr.org .

## Operating principles and governance

The Editorial Committee of the Review is made up of members who also belong to the European Legal Network, a non-partisan professional network of Free Software legal experts. This network is facilitated by Free Software Foundation Europe (FSFE), though membership extends across a broad spectrum of interests engaging in Free and Open Source Software across four continents. FSFE exerts no editorial control over the Editorial Committee,. The composition of the Editorial Committee rotates regularly among European Legal Network members. This normally happens after the publication of each issue, and is governed by consensus decisions taken by the European Legal Network as a whole. The Review receives financial support from the NLNet Foundation.

## Overview of Volume 1, Issue 1:

This first issue of the Review contains a variety of articles which key into what is said above about the scope and ambition of the Review.

The most newsworthy case-law development in recent months in the world of FOSS licensing was the decision of the United States Federal Appeal Court in *Jacobsen v. Katzer* 535 F.3d 1373, a decision which is interesting to Free and Open Source lawyers for the Court's analysis of the fundamental nature of FOSS licensing conditions and to the wider public for its story of duplicity and dirty dealing in the world of model railway enthusiasts. In his article, *Bad Facts Make Good Law*, Lawrence Rosen magisterially unpacks the decision from the perspective of a U.S. lawyer, whilst Mark Henley looks at the decision from the other side of the pond in his stimulating article, *Jacobsen v Katzer and Kamind Associates – an English legal perspective*.

In a major contribution to working with Free and Open Source Software, Shane Martin Coughlan and Andrew Katz, in their article *Introducing the Risk Grid*, report on the work of a Special Interest Group of the European Legal Network. This Group considered possible methods to reduce or contain risk in transactions related to the supply chain in relation to commercial procurement of Free and Open Source Software. That work culminated in the creation of a Risk Grid, an invaluable tool for which both Customers and Suppliers will have cause to be grateful, regardless of their relative experience in Free andOpen Source Software and methods.

Also of great practical utility is Ywein Van den Brande's *The Fiduciary Licence Agreement: Appointing legal guardians for Free Software Projects*, which discusses the new version of the Fiduciary Licence Agreement released by the Free Software Foundation Europe. This Agreement allows Developers of Free Software Projects to assign their copyright to a single person, with the intention of (amongst other things) preserving the ability to relicense and allowing the assignee to demonstrate to the court sufficient title and interest to enforce Free and Open Source Licences.

The difficult and persistent problem of living with software patents is addressed by Christopher Wong and Jason Kreps in their article *Collaborative Approach: Peer-to-Patent and the Open Source Movement*, which gives useful guidance on how the Open Source movement, through using a process of peer review of pending patent applications, can cope with the threat of the

proliferation of non-meritorious or overly broad patents.

In a *Tech Watch* article, KDE Vice President of Legal Affairs Adriaan de Groot reviews some of the issues presently confronting community software authors: copyright consolidation, making a living from coding, and, as he puts it, 'doing legal stuff right'.

Finally, Andrew Katz, in a sceptical and entertaining book review touches on some of the themes already canvassed in this editorial, underling that theological debate on the virtues of different "movements" is more often productive of heat than light; whilst, in an equally provocative *Platform* article, *Collaboration among Counsel*, Karen Copenhaver challenges us with the question "why don't the lawyers get it?".

Consideration of this overview of the first issue should prove that one can be serious-minded and intellectually rigorous in addressing vital questions, yet at the same time be entertaining, stimulating and provocative.

And so, gentle readers, if you find yourself educated, informed, entertained, stimulated or provoked by this first issue of the International Free and Open Source Law Review, then perhaps you might find yourself moved (or even irritated) enough to write a contribution for the second issue.

In that spirit, your Editorial Committee offers this first ever issue of the Review to the world: it no longer belongs to us, but to you, all of our readers.

*Iain G. Mitchell Q.C. and the other members of the IFOSS L. Rev. Editorial Committee*

# The Fiduciary Licence Agreement: Appointing legal guardians for Free Software Projects

*Ywein van den Brande*[a]

*(a) Ywein Van den Brande is partner at De Bandt Keustermans & Van den Brande, a Brussels (Belgium) based law practice.*

**Abstract**
The new version of the FSFE Fiduciary Licence Agreement (FLA) is a short and clear document that allows developers of Free Software Projects to assign their copyright to a single person or organization. The FLA is a useful tool to ensure the legal maintainability of Free Software Projects, addressing important issues such as preserving the ability to relicense code and the need to have sufficient rights to enforce licences in court. However, its international approach might not suit the requirements of all jurisdictions. It is advised to use it in connection with local legal advice.

**Keywords**
Fiduciary Licence Agreement, license, Free Software, Open Source, trustee, copyright consolidation

## I. Introduction

On February 1, 2007, the Free Software Foundation Europe (FSFE) released a new version of its Fiduciary Licence Agreement (FLA). With this agreement, developers of Free Software Projects can assign their copyright to a single person or organization. The goal of the agreement is to ensure the legal maintainability of Free Software Projects, including important issues such as preserving the ability to re-license and the certainty of having sufficient rights to enforce licences in court[1].

Assigning or licensing copyright to a fiduciary is not a new technique. It was probably first used in the late 18th century by French play writers who sought to bargain collectively[2]. The technique is already being applied in the Free Software community, for example by the Free Software Foundation (FSF) in the United States with respect to the rights on the GNU project[3]. What is unique about the FLA is that it seeks to cover multiple jurisdictions under a single concise agreement[4].

1 See FSFE press release dated 1st February 2007 (http://mail.fsfeurope.org/pipermail/press-release/2007q1/000168.html).
2 French theatre authors saw their plays performed in Parisian bars and theatres without receiving compensation. In 1777 Beaumarchais urged these writers to manage their rights collectively (A. Berenboom (2008), *Le nouveau droit d'auteur* (Brussel, BelgiumGroupe De Boeck s.a. / Larcier) nr. 271, p. 425, ISBN 9782804414399).
3 http://www.gnu.org/licenses/why-assign.html.
4 The FLA will become available in German, French, Italian, Swedish, Serbian, Polish, Dutch, Spanish and Portuguese - see FSFE press release dated September 26, 2008 http://mail.fsfeurope.org/pipermail/press-release/2008q3/000217.html.

While FSFE should be applauded for its efforts to make this international agreement, the FLA's generic approach does not always result in a solution that is perfect in each jurisdiction. In some jurisdictions the wording of the FLA may, for example, be incompatible with the requirements for the transfer of copyrights on existing works or future works. The FLA is undoubtedly a great starting point, but will be even better for use in consultation with a local legal expert.

## II. The assignment of rights under the FLA

The European Software Directive[5] stipulates that the author of a computer program is the natural person or group of natural persons who has created the program[6]. If the computer program is created jointly by a group of natural persons, the exclusive rights are owned jointly by the authors[7]. This means that the permission of all the authors needs to be obtained to undertake any exploitation of the work.

Thanks to the specific nature of Free Software licences and the freedoms they provide, this rule of joint ownership seems to work quite well for Free Software Projects. Copyleft licences such as the GNU General Public License (GPL) are broad enough to allow collaboration between the different developers. Many Free Software Projects function without any further copyright arrangements.

But the success of Free Software projects raises questions that go beyond collaboration and software development. A simple example is the question for Free Software projects using the GNU General Public License version 2 (GPLv2) whether or not to switch to the newer GNU General Public License version 3 (GPLv3). Such decisions are difficult to take when not all copyright holders can be tracked or when they don't agree. Another example is the matter of enforcing licence compliance. This can prove to be quite difficult when the copyrights are scattered. The bundling of copyrights in a single decision taking authority is an attractive and powerful solution to such issues, and the Fiduciary Licence Agreement endeavours to achieve exactly that.

The FLA is a short and clear document that excels in its simplicity: the developer assigns his/her copyrights world-wide to a single trusted person or organization[8] that returns a broad non-exclusive licence to the developer[9]. In countries where such an assignment is not possible[10], the developer grants an exclusive licence on the software[11], comprising:

1. the right to reproduce in original or modified form;
2. the right to redistribute in original or modified form;
3. the right of making available in data networks, in particular via the Internet, as well as by providing downloads, in original or modified form;
4. the right to authorize third parties to make derivative works of the software, or to work on and commit changes or perform this conduct themselves[12].

5 Directive 91/250/EEC.
6 The legislation of the Member States may designate legal persons as the rightholder.
7 Directive 91/250/EEC, Article 2.
8 FLA §1.
9 FLA §3 (2).
10 For example, and as cited in the FLA in footnote 1, Germany, Austria, Slovenia and Hungary are countries where assignments of the copyright in a work are impossible.
11 The assignment of copyrights to the fiduciary is not always necessary to achieve a similar result. The Apache Software Foundation Individual Contributor License Agreement, e.g., opts for a broad licence instead of the assignment of copyright (http://www.apache.org/licenses/icla.txt).
12 FLA §1.

The copyrights assigned (or exclusively licensed) under the FLA also include, where possible, rights on future developments, future corrections of errors or faults and other future modifications and derivative works of the software made by the developer assigning the copyrights. Modifications that are not derived from the software but that should be regarded as independent and original software are not affected by the assignment of future rights[13]. If the assignment of rights in future works is not possible in a country, like France, for example, these rights will not be assigned[14]. If future rights are not assigned, those rights will remain fragmented with their respective authors. In these situations, a periodical assignment of copyrights would be required to accomplish a full assignment over time[15].

It is evident that the authors of the FLA had to choose a generic wording that is valid in as many jurisdictions as possible, instead of addressing each country individually. But drafting such a catch all clause is a difficult exercise that might not result in a solution that is perfect in each jurisdiction. Under the Belgian jurisdiction, for example, the assignment of rights in future works is only valid for a limited period in time[16]. A transfer of copyrights in future works that is *temporally unlimited* is not valid. To achieve the same result under Belgian law, the wording *"for the entire term of copyright protection"* would be more effective than *"temporally unlimited"* as provided for in FLA article 3 (4). Another particularity under Belgian law for the transfer of rights in future works is the obligation to stipulate the genre of the work[17]. The term genre is ambiguous and often leads to confusion. Is it meant to serve in a peer-to-peer environment, strictly on a mobile device, or merely as documentation? All these exploitation methods and uses of the same work may qualify as different genres under Belgian law and should be named in the agreement. The transfer of rights with respect to still unknown exploitation methods is void under Belgian law[18]. That is why the FLA should be used in connection with local legal advice.

Article 6(b) of the Berne Convention[19] provides that the author of a literary work has the right to claim authorship and to object to certain modifications and other derogatory actions, even after the transfer of his economic copyrights on said work. These are the so-called inalienable moral rights of the author. As the European Software Directive provides that computer programs need to be protected by copyright as literary works within the meaning of the Berne Convention[20], the question of how these moral rights affect the transfer of the copyrights to the fiduciary under the FLA arises. The FLA stipulates that it leaves the moral and/or personal rights of the author unaffected[21]. This is a good decision on the part of the authors of the FLA, as the question of how the inalienable moral rights affect software is not unique to Free Software[22].

The copyright assignment under the FLA is world-wide, temporally unlimited and quasi-unconditional. Only in case that the fiduciary violates the principles of Free Software does the

13 FLA §2.
14 FLA, footnote 2.
15 The Apache Software Foundation Individual Contributor License Agreement contains an opening to such a periodical transfer by providing that only *intentionally submitted* modifications and additions will be covered under the Individual Contributor License Agreement.
16 Belgian Copyright law of 30 June 1994, article 3, §2 (available at http://www.juridat.be/cgi_loi/loi_a.pl?language=nl&caller=list&cn=1994063035&la=n&fromtab=wet&sql=dt='wet'&tri=dd+as+rank&rech=1&numero=1).
17 Belgian Copyright law of 30 June 1994, article 3, §1.
18 Belgian Copyright law of 30 June 1994, article 3, §1.
19 Berne Convention for the Protection of Literary and Artistic Works of September 9, 1886, as last amended on September 28, 1979.
20 Directive 91/250/EEC, Article 1.1.
21 FLA §1 (2).
22 E. J. Louwers and C. E. J. Prins (2008), *International Computer Law* (San Francisco, CA, USA: Matthew Bender / LexisNexis) § 7.20, p. 7-86, ISBN 9780820513188.

copyright revert to its original owner(s)[23]. Thus, only persons and organisations held in great trust by the original copyright holders should be appointed as fiduciary. The FLA §3 (3) provides: *"In the event FSFE violates the principles of Free Software, all granted rights and licences shall automatically return to the Beneficiary and the licences granted hereunder shall be terminated and expire."* It might have been prudent to add that licences granted by the fiduciary to users that comply with the Free Software principles remain unaffected by a termination of the FLA.

## III. The fiduciary

In theory, any person or organization can offer to act as fiduciary for a Free Software project using the FLA as a form. However, such responsibility should not be undertaken lightly as there may be a question of whether the fiduciary assumes liability for the software through the FLA and the subsequent licensing of the software to third parties in its own name[24]. Besides a guarantee regarding employer's rights[25], the FLA does not contain any contractual warranty or disclaimer with respect to the assigned software. What for instance if a third party not having accepted the GPL suffers damages or if a virus has been wilfully included in the code? Even though under normal circumstances an appeal on the exclusion of liability may be upheld in court, assuming liability is not always free of risk[26]. Of course, transferring the liability to an exploitation company or non-profit organization may also be utilized as a means to mitigate the liability of the authors.

Under its Fiduciary Programme the FSFE offers to act as fiduciary to whom the copyrights on a Free Software Project can be assigned via the FLA. In fact, the FSFE is the default fiduciary in the FLA. But FSFE only accepts a limited number of projects[27]. Projects that have been selected are Bacula.org and OpenSwarm[28].

The offer of FSFE to act as a legal guardian of Free Software Projects is a good initiative that will undoubtedly leverage a Free Software Project's strength to enforce third party compliance to the licence terms. However, whether FSFE is the optimal organisation to act as a guardian, and whether the objectives and actions of FSFE will be in line with the wishes of the developers, remains to be seen. It is up to FSFE to prove that it has the integrity and trustworthiness to fulfil this role. But even though its Fiduciary Licence Policy[29] is rather short, it clearly shows the willingness of FSFE to act as a neutral legal guardian that will not interfere with the project management, direction or administration. As Georg Greve, former President and founder of FSFE explains[30]: *"For us the most important issue is not whether projects assign their copyright to FSFE or any other organisation. We just want to do our part so projects do not neglect these issues"*.

Another significant legal question that arises with FSFE acting as guardian and manager of the copyrights on Free Software projects is whether so doing would cause FSFE to qualify as a collective management society in the field of copyright with all rights and obligations this entails. Collective management societies are bodies that group authors in order to more effectively manage their copyrights.

Since there is no uniform European definition of what the function is of collective management

23 The FLA remains silent as to how such a return of rights and licences will take place. If no amicable settlement can be negotiated, the author will need to refer the matter to the court.
24 FLA § 3 (1).
25 FLA § 1 (3).
26 E. J. Louwers and C. E. J. Prins (2008) § 17.04[B], p. 17-15.
27 FSFE has published its selection guidelines on http://www.fsfeurope.org/projects/ftf/guidelines.en.html.
28 http://mailman.fsfeurope.org/pipermail/press-release/2007q1/000168.html.
29 Available at http://www.fsfeurope.org/projects/ftf/fiduciary-policy.en.html.
30 See FSFE press release dated 1st February 2007 http://mail.fsfeurope.org/pipermail/press-release/2007q1/000168.html.

societies, this remains very much an open question. On the one hand, collective management societies are often referred to as collecting societies in European directives[31]. This might suggest that only societies that collect royalties and distribute them amongst their members qualify as collecting societies, what typically is not the case for FSFE. The denomination *collecting societies* seems however too narrow to cover the real function of collective management societies[32]. Directive 93/83/EEC e.g., defines a *collecting society* much more broadly *"as any organization which manages or administers copyright or rights related to copyright as its sole purpose or as one of its main purposes"*[33]. The European Parliament resolution of January 15, 2004 on a Community framework for collective management societies in the field of copyright and neighbouring rights[34] applies the term *"collective management societies"*. Even thought this resolution does not contain a definition, it is clear that the functions of a collective management society under this resolution exceed the mere collection and distribution of royalties[35].

Moreover, the answer to the question whether FSFE would qualify as a collective management society might vary amongst the European Member States. Collective management societies are governed by national legislation, and these various laws and provisions differ widely due to the fact that every country has its own traditions and specific historical, legal, cultural and economic characteristics[36].

The question whether the FSFE would qualify as a collective rights management body or not, is not without relevance as the recognition as a collective rights management body is conditioned to various obligations that differ widely amongst the Member States. These obligations range from transparency obligations to governmental control. Should FSFE qualify as a collective management body, it is paramount for FSFE to seek recognition, as collective rights management bodies must be regularly recognized in order to represent the holders of intellectual property rights in legal actions[37].

## IV. Conclusion

Free Software communities must take their rights seriously if they expect third parties to comply with and respect their licences. This includes organizing and structuring these rights in a way that ensures the legal coherence of a project's rights in the future. The FLA is a useful tool to assist with this task and FSFE should be applauded for its efforts to draft this agreement. However, local laws often have particular requirements that may pose a risk when utilising generic international agreements. Therefore, while the FLA is undoubtedly a great starting point, it is prudent to consult with a local legal expert prior to putting it to use.

31 E.g. old Directive 92/100/EEC, article 4; Directive 93/83/EEC, article 1 and Directive 2001/84/EC, consideration 28.

32 Under Belgian law (Belgian Copyright law of 30 June 1994, article 65) societies for the management of rights are defined as all who collect or distribute rights for the account of the right holders. Where this legal definition stresses the collection and distribution, it would be wrong to reduce the role of management societies thereto (F., De Visscher and B., Michaux (2000), *Précis du droit d "auteur et des droits voisins*, (Brussels, Belgium: Bruylant) p. 399, ISBN 2802712799).

33 Directive 93/83/EEC, article 1 §4. Available at http://www.ebu.ch/CMSimages/en/leg_ref_ec_directive_copyright_satellite_cable_270993_tcm6-4289.pdf

34 OJ C 92 E, 16.4.2004, p.425. Available at: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:C:2004:092E:0425:0432:EN:PDF.

35 The resolution stresses e.g. the cultural and social aspects (nr. 13, 22 and 27).

36 Resolution, nr. 35. For a comparative study, see The Collective Management Of Rights In Europe, The Quest for Efficiency, KEA, July 2006, available at http://www.europarl.europa.eu/comparl/juri/study/rights_en.pdf.

37 Directive 2004/48/EC, article 4 (c). Available at: http://eur-lex.europa.eu/pri/en/oj/dat/2004/l_195/l_19520040602en00160025.pdf.

# Collaborative approach: Peer-to-Patent and the Open Source movement

*Christopher Wong[a], Jason Kreps[b1]*

*(a) Project Manager for Peer-to-Patent; Postgraduate Fellow at the Center for Patent Innovations at New York Law School's Institute for Information Law & Policy (b) Development Analysis Lead for Peer-to-Patent*

**Abstract**
The patenting of software has increased significantly. Regardless of any personal bias as to the existence of software patents, it is a trend that is unlikely to end anytime soon. As a result, the open source movement may be threatened by the proliferation of non-meritorious or overly broad patents. Peer-to-Patent provides a means for mitigating the limitations that may be placed upon the open source community by software patents, as the program allows the open source community to participate in the peer review of pending patent applications. Members of the open source community are knowledgeable, interested parties with a unique stake in the software patent debate and thus are capable of making a significant contribution to improving the current system.

**Keywords**
Patents, Free and Open Source Software, Peer-to-Patent, Patent proliferation

## I. Introduction

The United States Constitution gives Congress the sole power to legislate. While it continues to be an issue of contention,[2] our government has seen fit to allow the delegation of legislative power to administrative agencies in the executive branch.[3] One of the main justifications for doing so is that agencies tasked with the administration of a particular area are uniquely able to promulgate rules that are tailored to suit the needs of the field.

1 The authors would like to thank Mark H. Webbink (Executive Director of the Center for Patent Innovations at New York Law School, home to Peer-to-Patent), and Beth Simone Noveck, (Creator of the Peer-to-Patent project), for their vision and ongoing guidance and support. The authors would also like to thank the Peer-to-Patent team for their valuable insights, and without whose efforts there would be no Peer-to-Patent about which to write: Andrea Casillas (Outreach Management Lead, Peer-to-Patent; 2L New York Law School), Thomas Lemmo (Application Management Lead, Peer-to-Patent; 1L New York Law School), Joseph Merante (Application Manager, Peer-to-Patent; 2L, Student Research Fellow, Institute for Information Law & Policy, New York Law School), and Jason Deveau-Rosen and Kaydi Osowski (Student Associates, Center for Patent Innovations, New York Law School).

2 See Field v. Clark, 143 U.S. 649 (1892) ("That Congress cannot delegate legislative power to the President is a principle universally recognized as vital to the integrity and maintenance of the system of government ordained by the constitution.")

3 See J.W. Hampton, Jr. & Co. v. United States, 276 U.S. 394 (1928) ("If Congress shall lay down by legislative act an intelligible principle to which the person or body authorized to fix rates is directed to conform, such legislative action is not forbidden delegation of legislative power."); See Amalgamated Meat Cutters v. Connally, 337 F.Supp. 737 (1971).

Such is the case with the United States Patent and Trademark Office (USPTO). The USPTO is, among other things, tasked with maintaining our system of patents. It is allowed to promulgate rules, so long as the rules are procedural in nature, because it is considered to be in the best position to make those decisions. However, if the USPTO is expected to be expert in the administration of the patent system as a whole, as a justification for allowing their existence, so too should they be held to that standard in every aspect of the patent system.

As it turns out, there are a number of considerations that limit the extent to which examiners at the USPTO can be the experts needed for the proper examination of patent applications. For example, examiners are only required to hold a bachelor's degree in a accepted scientific or technical field, whereas many patent applications are filed by inventors holding advanced degrees. That is not to say that education is the sole means of attaining expertise. However, the divergence of academic degrees between examiner and applicant evidences the extent to which an examiner may not be knowledgeable about a specific application of technology.

**Examination**

When an application for a patent is filed, it is assigned to a class and an examiner in that class is tasked with comparing the claimed invention against all of the scientific and technological antecedents that touch on the claimed invention. A person is entitled to a patent unless the claimed invention is not novel[4] or would have been obvious to a person having ordinary skill in the art (PHOSITA).[5] That is, if a claimed invention is not anticipated by another invention (and therefore novel) and not an obvious extension of an existing invention or combination of inventions (and therefore non-obvious), a patent should be granted. These antecedents are known as prior art and this process of comparison is one of the most important aspects of patent examination.

Examiners at the USPTO are restricted in what resources they have available to them when searching for prior art. As a result, they do not have access to all the prior art that they need. Innovation does not always occur in an organized, documented manner. A significant amount of prior art goes un-catalogued and resides outside of the examiner's reach. This information deficit makes it difficult for any examiner, regardless of ability, to make the correct determination as to the patentability of a claimed invention.

**Implications for the Open Source Community**

The state of patenting in the software industry is controversial, to say the least. There are many arguments as to whether software constitutes patentable subject matter to begin with. This issue was most recently visited by the Court of Appeals for the Federal Circuit's (CAFC) decision in In re Bilski.[6] Many observers had hoped that the CAFC would reject as patentable subject matter business methods and software.[7] However, while the court did put forth the "machine or transformation" test as a way of limiting patentable subject matter, it made no indication that software or business methods would fail to qualify under the new test. Thus, the patenting of software remains of real concern for the open source community, as open source technology benefits from unrestricted use of prior technology.

4 See 35 U.S.C. §102.
5 See 35 U.S.C. §103.
6 See In re Bilski, 545 F.3d 943, 88 U.S.P.Q.2d 1385 (Fed. Cir. 2008).
7 To be clear, the facts in Bilski dealt specifically with the patentability of a business method. However, many groups filed amicus briefs debating the patentability of software in the hopes that the CAFC would take software implications into consideration before announcing any new test for subject matter patentability.

Open source technology increasingly permeates the mainstream. Any person that comes in contact with the Internet is likely utilizing software that contains some form of open source code. An open source product is characterized by the ability of others to use, copy, modify, and distribute the original source code, as allowed under licensing terms granting broad, royalty-free copyright permissions.[8] This form of innovation, however, runs counter to the justification for patent protection, which gives an inventor exclusive rights to the invention for a period of 20 years during which time no other person within the patent granting jurisdiction can utilize the patented technology without the express consent of the inventor. As the patenting of software continues, the implications for the open source community are obvious: the more software is patented the less software is available for further innovation. Beyond this concern of software patenting effectively handicapping downstream innovation lies another major concern that open source software may be co-opted by an entity seeking to commercialize a product that contains open source code.

The ideal solution, from the perspective of the open source community, would be the elimination of software patenting as a whole. But, given the CAFC's recent decision in In re Bilski and the present posture of the European Patent Office, this is unlikely to happen any time soon, if ever. The concerns of the open source community, however, may be allayed through participation in Peer-to-Patent.

## II. Peer-to-Patent

Peer-to-Patent was launched by New York Law School in cooperation with the USPTO; an historic initiative to open the patent examination process up to public participation. The program launched as a one year pilot on June 15, 2007, but showed enough promise after the first year to be granted a one year extension to further investigate the effects of public participation on the patent system. The program, upon consent of the inventor, posts published patent applications on the Peer-to-Patent website for 16 weeks, during which time any member of the public may review the application, discuss the application with others in the community, submit prior art relevant to the patentability of the claimed invention, denote the relevance of the prior art, and annotate prior art submitted by others to make the prior art more readily useful. Peer-to-Patent then forwards the best prior art, as rated by the community, to the USPTO for use by examiners in the actual examination of the patent application.

The pilot was implemented in Technology Center 2100 (TC 2100) of the USPTO, an art unit covering computer architecture, software, and information security. For the second year of the pilot, the program was expanded to also include patent applications pending in class 705, Business Methods and E-Commerce.

The area of software patents was an optimal test bed for piloting Peer-to-Patent, as it suffers greatly from the problems associated with the information deficit. As a result, Peer-to-Patent, in its present form, bears directly upon the open source community. The closed databases of prior art that examiners at the USPTO have access to do not contain evidence of many open source projects that would otherwise qualify as prior art were they accessible nor are examiners generally familiar with software developments that occurred prior to the aggressive patenting of software in the last 15 years. While the ultimate decision of patentability still lies with the USPTO, Peer-to-Patent has shown not only that people aware of open source projects are willing to participate, but also that it

8 See Tiller and Fontana, Brief of Amicus Curiae Red Hat, Inc., In re Bilski (2007) ("A good example of an open source project is the Linux operating system kernel, which is one of the most commercially-important open source programs and which is a core component of Red Hat's flagship product, Red Hat Enterprise Linux. The Linux kernel contains several million lines of source code. A worldwide community of hundreds of contributors, including many employees of Red Hat, collaborate via the Internet in developing and improving the Linux kernel.")

is a useful tool for getting open source documentation in front of the examiner. For example, a reviewer on the Sun Microsystems patent application *Method and Apparatus for Delivering Device Drivers*[9] submitted a paper written by Klaus Knopper, an electrical engineer in Germany, entitled *Building a self-contained auto-configuring Linux system on an iso9660 filesystem*.[10] Mr. Knopper's company, Knopper.net, focuses on customized open source solutions while Mr. Knopper himself also works with developers on a freelance basis on a variety of IT projects based on free software. It is unlikely that documentation of Mr. Knopper's open source product would have been accessible to an examiner at the USPTO.

This example also demonstrates the importance of obtaining an international field of participants, as the relevance of prior art is unaffected by national borders. To date, Peer-to-Patent has been visited by over 70,000 unique visitors in 157 countries/territories. While software patenting is more prolific in the U.S. than anywhere else, U.S. patents nonetheless affect the software industries and patenting trends in other countries. For this reason alone it is important for the international community to participate in a program that seeks to improve patent quality by way of reducing the number of non-meritorious patents that are granted.

**Collaboration**

Members of the open source community are accustomed to working from a collaboration-based approach. Developers cooperate to write code, identify and diagnose problems, and customize software to a wide array of applications. This proclivity towards working together is easily adaptable to the work needed for participating in Peer-to-Patent.

A significant problem with patents and patent applications is that they are written in language that is difficult to read for anyone who is not a patent attorney. Recognizing that the average contributor is not a trained patent attorney, Peer-to-Patent provides a discussion board for each patent application where reviewers can communicate and help each other understand the claims of patent application. Reviewers are also able to submit items for research and rate and annotate[11] the prior art references submitted by other members of the community. Thus, much like open source projects, reviewers solve problems as a group. As a collaborative project, the success of Peer-to-Patent does not require that each person that participates be an expert capable of finding, digesting, and submitting relevant prior art. The following case illustrates the collaborative approach.

One of the first patent applications to undergo review on Peer-to-Patent and receive an office action from the USPTO was a Hewlett-Packard application for *User Selectable Management Alert Format*.[12] One of the authors of this article, Christopher Wong, holds a BSBA in Information Technology. By the time Peer-to-Patent launched and the HP application was available for review, Wong was nearly 3 years removed from any software engineering or computer programming courses he had taken during his undergraduate years. He did, however, recognize the subject

9 See Saulsbury, et al., Method and apparatus for delivering device drivers, Publication #20070162625 (2006).

10 See Knopper, Building a self-contained auto-configuring Linux system on an iso9660 filesystem, available at http://www.knopper.net/knoppix-info/knoppix-als2000-paper.pdf

11 The ability to annotate a prior art submission is itself a significant improvement to the current system. In the US, a person wishing to submit a prior art reference on a patente application may only do so for a limited time after the patent application has been published, and must pay a fee. Because the submitter may not make any annotations to the prior art, they leave to chance that the prior art will get in the hands of the proper examiner and that the examiner will utilize the prior art in the intended way. See Manual of Patent Examining Procedure, 37 C.F.R §1.99, (updated July 2008) available at http://www.uspto.gov/web/offices/pac/mpep/documents/appxr_1_99.htm.

12 See Broyles and Gibbons, User selectable management alert format, Publication #20070118658 (May 24, 2007), available at http://appft1.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&p=1&u=%2Fnetahtml%2FPTO%2Fsearch-bool.html&r=1&f=G&l=50&co1=AND&d=PG01&s1=20070118658&OS=20070118658&RS=20070118658.

matter of the patent application, even if not completely understanding it. He then performed a cursory search of the Internet and found Intel's Active Management Technology (AMT) site.[13] Not knowing exactly how to use the prior art, but recognizing that it might be relevant, Wong then uploaded the product documentation URL to Peer-to-Patent as a research item, with the note, "[a]nother example of management alert format."[14]

Roughly two months later, with about 2 weeks remaining to review the patent application, Steven Pearson, a senior engineer at IBM who had been facilitating discussion on the discussion board, posted a comment for his fellow reviewers, which read in part, "I think I'll move one of the Intel AMT "research" references to the prior art list, as it is in the neighborhood even if not dead on for this application and we haven't maxed out yet."[15] Pearson then submitted a product guide for Intel's AMT[16] as prior art, filling out the submission form with the detail and clarity that Wong could not. Pearson then annotated the prior art with directions to a specific page and figure within the product guide. In total, 30 people took part in the review of the HP application and submitted 9 prior art references for use by the USPTO in determining the patentability of the claimed invention.

On February 8, 2008, the USPTO concluded its examination of the HP application and mailed its first office action on the merits. In the office action, the examiner relied upon the non-patent literature supplied by Pearson (in combination with another patent application filed prior to the HP application) in rejecting all 21 of the application's claims as being obvious. In reaching the decision, the examiner specifically referred to the same figure and section within the AMT product guide that Pearson had noted in his annotation.

Observers have expressed motive as a prominent concern in allowing the public to participate in the examination of patent applications. Peer-to-Patent emphasizes the idea that relevant prior art is relevant regardless of impetus. As mentioned above, in this case, the prior art reference cited in rejecting the HP application was the Intel AMT product guide submitted to Peer-to-Patent by an IBM software engineer. Notwithstanding the potential benefit to IBM in defeating the claims of the HP application, it might also be fair to suggest that in doing so, the reviewer aided in protecting the intellectual property of Intel.

More importantly, the above example illustrates the type of participant that Peer-to-Patent is seeking. Though successfully reviewing a patent application likely necessitates more than one examiner with more than 20 hours to spend on it, it does not require a critical mass of the most expert members of a given field. The project leadership recognizes that few participants will have the time to perform all of the functions of examination from start to finish. What is necessary, is a community of reviewers comprised of both those who have the ability to simply recognize that a prior art reference might be useful, and those with the expert skill needed to explain specifically why, and how, a prior art reference should be used. Members of the open source community likely fall into both categories, as many have significant historical knowledge of the field of software development and others have the practical expertise and know-how that can only be obtained by working intimately with such cutting-edge projects as those associated with the open source movement.

13 See Intel Active Management Technology, available at http://www.intel.com/technology/platform-technology/intel-amt/index.htm.
14 See Research, available at http://www.peertopatent.org/patent/20070118658/research/list.
15 See Discussion, available at http://www.peertopatent.org/patent/20070118658/discussion.
16 See Intel Corp., Intel Active Management Technology (Intel AMT) Quick Reference (October 1, 2005), available at http://download.intel.com/support/motherboards/desktop/sb/amt_quick_start_guide1.pdf.

**Growth**

Thus far, 56 applications that have undergone review on Peer-to-Patent have received first office actions. In 15 of these office actions, the examiner cited prior art references submitted through Peer-to-Patent as a basis for rejecting the patent application.

The success of Peer-to-Patent has not gone unnoticed. Many other national patent offices suffer from the same problems as the USPTO, namely, a significant backlog of applications, lack of time for examination, deficiency in personnel, and gaps in the accessibility of information.[17] These agencies also understand the need for taking action. In 2008, the Japan Patent Office (JPO), with guidance from the Peer-to-Patent team, launched their version of Peer-to-Patent, called Community Patent Review.[18] The Peer-to-Patent leadership hopes to launch a pilot with another national patent office later this year. The Center for Patent Innovations at NYLS, home to Peer-to-Patent, has also had preliminary discussions with the European Patent Office, the Canadian Intellectual Property Office, and IP Australia to adapt the Peer-to-Patent system for their respective patent offices.

Up until now, Peer-to-Patent has been limited in scope to U.S. patent applications pending in USPTO Technology Center 2100. If the program were to be implemented to cover all subject matter, it would need to handle a significantly greater number of patent applications, as last year the USPTO received over 450,000 patent applications.

## III. Scalability

Given the above room for growth, there is some concern as to the scalability of the program. The open source community is, therefore, an important community to engage. In order to explore the scalability of the Peer-to-Patent model of public participation we must first analyse the extent to which the Peer-to-Patent pilot has been successful, thus far, in mobilizing contributors and utilizing their collective expertise. The measure of success informs the degree to which the project can be readily enlarged to accommodate more patent applications and its implementation to other patent systems around the globe.

The following data was assembled by the Peer-to-Patent team and provides insight into those dimensions of the Peer-to-Patent interface that show promise for scalability, and those that need to be improved. Generally, the figures illustrate various trends concerning the traffic to the website and the interaction of users with the project. Based on our conclusion that Peer-to-Patent is currently driven by a rather robust and loyal base of peer reviewers, we have determined that an essential component to enhancing the project's effectiveness rests on the ability to both solicit and retain more peer reviewers.

17 For example, the Trilateral Offices (the USPTO, European Patent Office, and Japan Patent Office) receive roughly 1 million patent applications per year. See Christopher Wong, Community Service: Adapting Peer Review to the Patenting Process, I/S: A Journal of Law & Policy for the Information Society, Ohio State University Moritz College of Law/Carnegie Mellon Heinz School of Law and Public Policy, Vol. 4 Issue 1 (2008), ("Today 5,500 U.S. patent examiners labor independently, under a backlog approaching 1 million applications, with no more than eighteen to twenty hours to review each application. The number of patent applications filed per year has grown steadily from 250,000 in 2000 to over 400,000 in 2006. If no action is taken, the backlog is projected to reach 1.4 million applications by 2012. These numbers are in stark contrast to the European Patent Office ("EPO") whose 3,500 examiners received 208,000 patent applications in 2006 while working under a backlog one-third that of the PTO. Though informative of the problem, these numbers are not conclusive. The third of the "Trilateral Offices," the Japanese Patent Office ("JPO"), works under similar (if not greater) pressure than the PTO, receiving 400,000+ patent applications annually while maintaining a backlog of about 750,000. However, the JPO only employs 1,358 patent examiners, roughly one-third of the PTO.")

18 The JPO is currently in the process of analysing the results of their first year pilot which concluded in January 2009.

Data was collected from five different time intervals, spanning the life of the pilot, from June 15, 2007, to March 20, 2009. These intervals include data collected after one month, one year, one year and six months, total time, and a focus interval ("focus group") consisting of data collected in the three months of December 9, 2008, to March 20, 2009. This specific focus interval was implemented in order to track any changes in traffic patterns emerging during a time period in which the Peer-to-Patent project had received valuable publicity by way of USPTO solicitation, and various media outlets, included articles about Peer-to-Patent, such as BusinessWeek and Ars Technica. The most informative visitor trends, and the ones we will focus on in this article, were established within the following categories: visitor loyalty, depth of visit, and traffic sources.

**Visitor loyalty**

Since its launch, Peer-to-Patent has cultivated a committed peer reviewer base that uses the site regularly and thoroughly. For instance, the number of individual users who have visited the website 9-50 times has increased from 816 (4.2% of visitors) in the first month to 6347 (6.6%) in total. This shows that there is a significant proportion of reviewers who return to the website and have some interest in keeping up to date as to site activity. Those who visit the site 9-50 times are likely spending their time monitoring the posting of new patent applications, discussing patent applications, and submitting prior art.

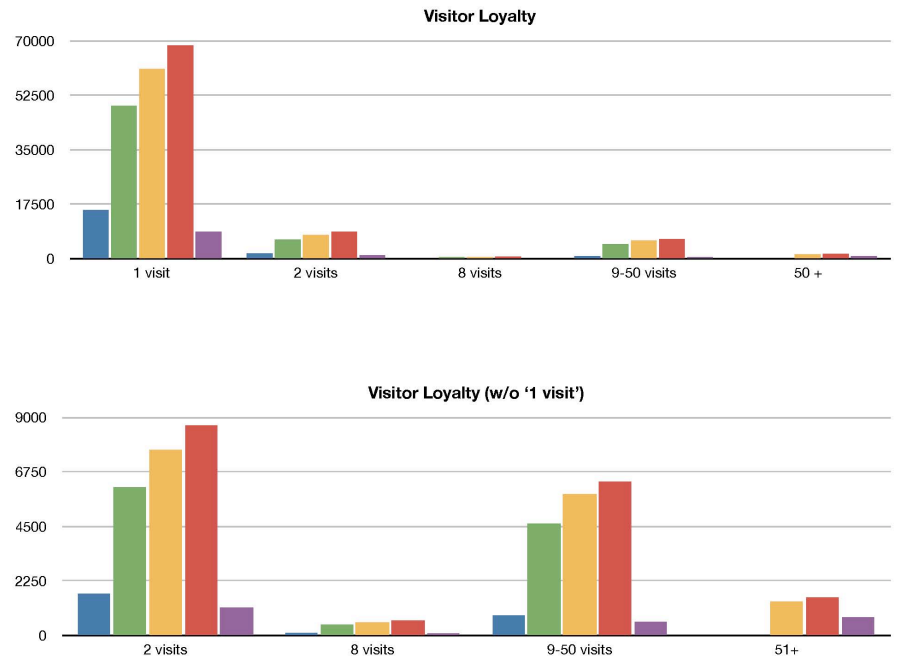


*Fig. 1: Visitor Loyalty by Number of Visits.*

During the first year of the project, not a single visitor used the site more than 50 times. Within the six month period directly following the one year mark, 1393 users had reached 50+ visits. By March 20, 2009, 1562 individual visitors had used the website 50+ times, or 1.6% of all visitors. Of this 1562, the focus group consisted of 739 users. In other words, 6% of all visitors during the last three months of data collection had used the site more than 50 times. These figures provide an illustration of the current success the interface is having in stimulating the self-selection of participants, retaining these participants, and developing an overall "human database" of interested citizen-experts.

**Traffic**

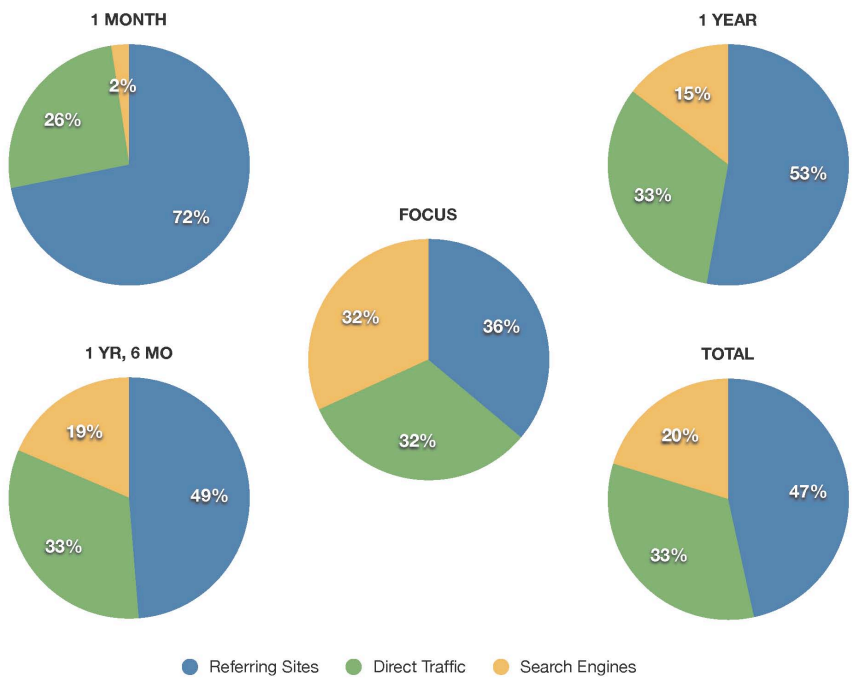


*Fig. 2: Peer-to-Patent Traffic by Source.*

Direct traffic represents visitors arriving from bookmarks and URL inputs, so it follows that users accessing the site through these means are interacting with Peer-to-Patent on a consistent and frequent basis. The percentage of direct traffic was the most stationary of the three sources throughout the testing period, accounting for 26% of all traffic to the website during the first month and 33% of all traffic during the focus period. The stability of this percentage is another demonstration of the project's ability to solicit and maintain a committed collection of peer reviewers.

Search engine traffic represents the visitors arriving at the website by way of a search engine results page. Visitors approaching a search engine likely have a previously formed interest in

online community participation in software and technology development, which they then act upon by searching for Internet content. However, another theory is that these visitors are using a search engine as a means to casually and intermittently check-in with the ongoing Peer-to-Patent project, looking for software applicable to their field. This kind of traffic is opposite to typing in a URL or adding a bookmark, the means through which Peer-to-Patent's dedicated peer reviewer base is likely to access the site. Whether visitors are acting on already formed interests or just checking-in, both are methods by which the open source community is likely interacting with Peer-to-Patent. The percentage of search engine traffic showed the most marked increase of the traffic sources, accounting for only 2% of all traffic to the website during the first month and 32% of all traffic during the focus period. This suggests a growing interest in Peer-to-Patent among those already involved or interested in software development and community participation, but not necessarily involved with the patenting of software (i.e. not part of the group that would be accessing the site directly and frequently). We interpret this as bearing great potential for the incorporation of more participation from the open source community, as collaboration-based participation is the bedrock of the open source movement and Peer-to-Patent alike.

Traffic from referring sites represents visits originating from clicking a link on a website, not Peer-to-Patent. During the first month, 72% of all traffic to the website could be attributed to referring sites. This number can be explained by the initial influence of Peer-to-Patent's progenitor website, which included information and links for the then developing Peer-to-Patent project.[19] During the focus interval, the percentage of visitors arriving from referring sites dropped to 36% of all visitors, a number that shows a continued presence of Peer-to-Patent in articles and links residing on other websites, with a diminished reliance on referrals from the Peer-to-Patent information site.

When inspected as a whole, a revealing trend emerges. As the project evolved, the ratio of the traffic sources became balanced, culminating in a roughly equal distribution among the three sources. The parity in traffic sources indicates two things. First, in terms of scalability, Peer-to-Patent is displaying the ability to be readily enlarged in all directions. Second, in terms of effectiveness, Peer-to-Patent has been successful in increasing the diversity of our visitor pool, in such a way that the site is receiving traffic through various channels of interest. This diversity of community should be highly regarded with respect to collaboration-based projects.

**Areas for Improvement**

While the numbers indicate a dedicated core of peer reviewers, one area in need of improvement is visitor depth: the number of pages on the site accessed by a user in a single visit. After the first month of the project, 44.4% of all visits yielded only a single page. In other words, 44.4% of visitors left without any interaction. The percentage continued to rise to 51.75%, 57.8%, for one year and the focus group, respectively. On average, 53.5% of visitors ended their visit after viewing one page. The notion that more than 50% of visitors are abandoning the website without a single click suggests a huge market for potential peer reviewers. In response, the Peer-to-Patent development team is working to release a new landing page specifically designed to increase site activity beyond the front page.

19 This site, referred to as the DoTank site, has continued in existence, serving as the information center for all things Peer-to-Patent. See The Peer-to-Patent Project: Community Patent Review, available at http://dotank.nyls.edu/communitypatent.

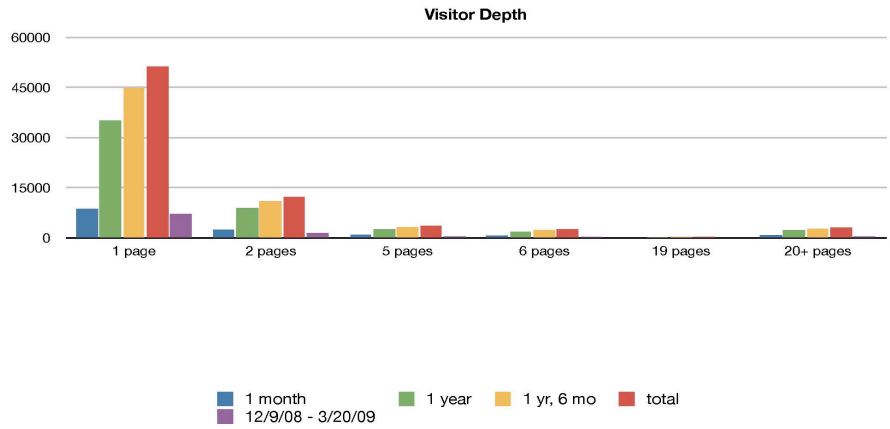


*Fig. 3: Visitor activity by page depth*

Within the focus group, the bounce rate is the highest among those visitors entering the website from a non-patent related referring site. For instance, the top referring site within the focus group was huffingtonpost.com. During this time, Peer-to-Patent received 456 visits originating from huffingtonpost.com, 95% of which were from first time visitors to Peer-to-Patent. The bounce rate for these visitors was 66%, while those visitors who did not bounce only interacted with the site for an average of 2 pages. Similarly troublesome were the visitors from wired.com. Following a story on Peer-to-Patent, wired.com sent 91 visitors, 98% of which were first-time visitors. The bounce rate for this group was 87%, with the other 13% interacting at an average 1.2 pages. Compare these two 'passing interest' pools of visitors to that of the 151 visitors coming from a link on uspto.gov. Of these 151 visitors, only 61% were new and only 27% bounced. The other 73% interacted with the website at an average of 8.6 pages.

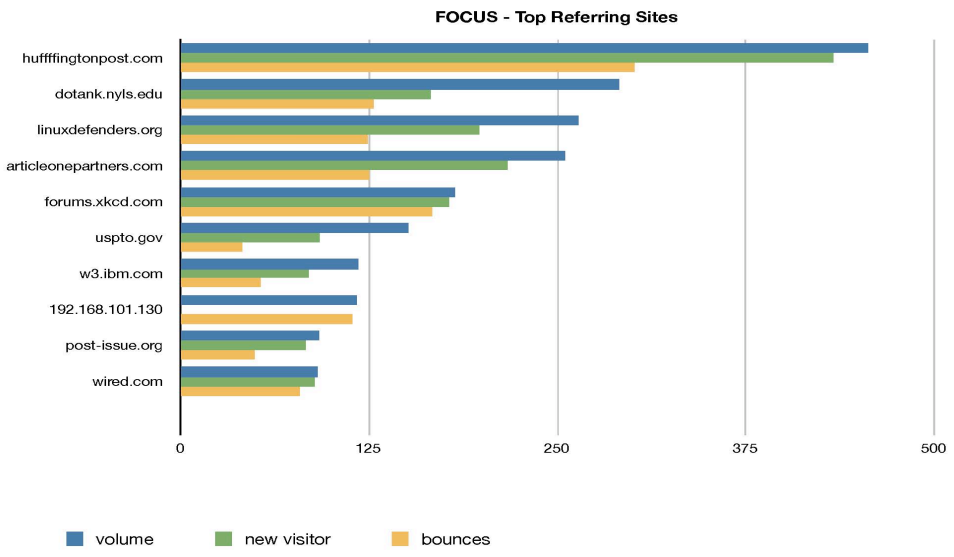


*Fig. 4: Page depth by Referring Site*

The obvious conclusion here is that those who end up on Peer-to-Patent are more likely to remain on the site and interact with the site if they are coming to the website with a purpose. Increasing participation and reducing the rate of attrition depends in large part upon the project leadership's ability to demonstrate to different communities exactly why they should care about the goals of Peer-to-Patent. In order to do so, a community needs to understand how patenting affects their industry. The open source community, more so than many other communities of practice, understands the implications of software patenting on the open source movement. As such, it is a population that can readily contribute to the cause of increasing transparency through collaboration and a population that Peer-to-Patent needs to engage.

## IV. Conclusion

Software patents are far more prolific in the US than any other country. However, while other national patent offices are more sceptical than the USPTO when it comes to recognizing computer software as patentable subject matter, software patents are still not unheard of.[20] Regardless of where software patents issue, they have an effect upon the software industry as a whole. As a

20 While Article 52(2)(c) of the European Patent Convention states that computer programs are explicitly ineligible for patents, the Board of Appeals of the European Patent Office has determined that when incorporated into a machine or a process that is itself patentable, the resulting system or process of operating a computer can be protected by patent. See European Patent Convention, art. 52, Dec. 13, 1970, E.P.C. 1973; see also IPR Helpdesk, CIP Programme, DG Enter. and Indus. of the European Comm'n, Patentability of Computer Programs (2005), http://www.ipr-helpdesk.org/documentos/docsPublicacion/html_xml/8_patentabilityComputerPrograms%5B0000001159_00%5D.html. For the Board of Appeals of the European Patent Office decision, see Case T-0928/03–3.5.01, Konami Co., Ltd., E.P.O. (June 2, 2006), available at http://legal.european-patent-office.org/dg3/pdf/t030928eu1.pdf.

result, all those who have an interest in the software industry also have an interest in maintaining the integrity of that industry.

One way of doing so is to participate in a process that assuages the concerns of litigation over open source products by ensuring that if software patents are to be issued (as it appears is the case), they will be issued only to those inventions that truly demonstrate innovation in the field of computer software. The open source movement depends upon the examiner's ability to defeat non-meritorious or overly broad claims within pending software patent application. By helping to provide patent offices with relevant prior art that would otherwise go unnoticed, the open source community can assist in safeguarding the very foundation upon which it is built. Peer-to-Patent is the path for doing so.

## About the authors

***Christopher Wong*** *is the Project Manager for Peer-to-Patent, Postgraduate Fellow at the Center for Patent Innovations at New York Law School's Institute for Information Law & Policy, and was recently named a Visiting Fellow at Yale Law School's Information Society Project. Mr. Wong earned his J.D. from New York Law School and holds a B.S.B.A. in Information Technology from the University of Denver. He can be reached at chris.wong@peertopatent.org.*

***Jason Kreps*** *is the Development Analysis Lead for Peer-to-Patent. Mr. Kreps is currently a first-year law student at New York Law School where he is a Student Associate at the Institute for Information Law & Policy's Center for Patent Innovations. Mr. Kreps holds a B.S. in Neuroscience from Allegheny College, where he was an Alden Scholar. He can be reached at jason.kreps@peertopatent.org.*

# Bad Facts Make Good Law: The Jacobsen Case and Open Source

*Lawrence Rosen*[1]

**Abstract**
For the first time, a major U.S. appeals court has held that an open source license is enforceable through preliminary injunctions. The court also found that exacting conditions in the form of compliance with open source requirements for disclosure and explanation of changes is entitled to recognition as consideration for a contractual license.

**Keywords**
Open source licenses, contracts, enforcement, infringement, consideration.

For many years the open source community has been eagerly awaiting a U.S. lawsuit that would bless the open source licensing model. In what in retrospect may seem like a leap of faith, millions of software programmers around the world published their works expecting that their open source licenses, including the GPL, would be honored and enforced in court. That fundamental assumption was never effectively tested in court, until the Jacobsen v. Katzer case [Page references in this article are to 535 F.3d 1373; 208 U.S.App. LEXIS 17161, Aug. 13, 2008.]

This decision finally explains how U.S. courts should analyze open source and open content licenses. The bottom line for us is that *copyright* law provides the remedies but contract law provides the analytical tools.

This is a strange case, based upon facts that did not seem favorable to Jacobsen ("J" in this article when I first heard of him several years ago. For one, the license actually used by J was the first version of the Artistic License, a document of which one could generously say that the non-lawyers who drafted it years ago took *artistic license* with legal style and method. It is not a very good license. Anyone looking for the formalities and magic words of modern proprietary or open source software licenses would be disappointed with the old Artistic License. It is not the license of choice on which we would want to hang the entire open source copyright licensing model in federal court.

The *Jacobsen* case found its way to the Court of Appeals for the Federal Circuit (CAFC), arguably the most important court short of the U.S. Supreme Court for intellectual property matters. That itself is a story that may interest lawyers who enjoy civil litigation, and may serve as an object lesson for plaintiffs and defendants alike. This case ended up in the CAFC because of statutory and factual reasons completely unrelated to the license itself. The court in which one litigates can present opportunities—and risks—that the parties to that lawsuit may not intend. Lawyers who enjoy civil litigation practice

1  This paper was prepared for presentation at the PLI Open Source Software 2008 program in New York (11/18/2008) and San Francisco (12/10/2008, also webcast). It is also published at the author's website, www.rosenlaw.com.

may appreciate the story that unfolds in this partly historical paper.

The most interesting part of the story took place during the recent few months in which, under the leadership of attorneys at Creative Commons, various open source organizations cooperated to write an amicus brief that focused the CAFC on the critical issues that we wanted the court to answer firmly for us.

We got just that. The *Jacobsen* decision finally settled that open source licensors can enforce their licenses effectively in U.S. courts. Defendants can't avoid copyright law by relying on contract law principles that are inapplicable in the open source and open content world.

As is true for many decisions that reach the highest appellate courts, this case has important implications far beyond open source. Creative Commons lawyers were involved in this case because *Jacobsen* would also settle the enforceability of their licenses, under which hundreds of millions of musical, literary, and artistic works are made available to the world on generous copyright terms. The authors of those works now have a valuable remedy—the preliminary injunction—to enforce their copyrights in U.S. Courts.

## *Jacobsen's* Complicated Civil Litigation Path

The plaintiff in this case, J, is a physicist who, as a hobby, developed software for controlling model trains. He licensed it to the world under the first version of the Artistic License. This license authorizes anyone to copy, modify and distribute the software on condition that they place certain notices in those works as attribution to its original author. Licensees are also required to identify that they have changed the work, to preserve the original author's reputation.

A small open source community developed around that model train software. Among those who took that software was the defendant, Katzer ("K" in this paper). According to the complaint, K modified the model train software, placed his own name on it, and distributed it to his customers. K didn't place the required attribution notices in his copies.

As this case progressed, K never actually denied that those copies were made and distributed, but asdescribed below, this case never reached the factual resolution stage. To the date this article is written, K has never formally answered the complaint, so we must assume that the facts as portrayed in J's complaint are true.

Before a lawsuit was filed, during the early stages of this open source dispute about attribution requirements of the Artistic License, K took the unusual step of retaliating by sending a letter to J accusing him of infringing a patent. The legal basis for that accusation was dubious at best; companies around the world throw such letters into the virtual trash can every day. But J (and his wife) didn't know how to react; their life savings were at risk.

J subsequently reacted by filing a lawsuit for *declaratory relief that he had not infringed* K's patent. Filing a declaratory relief action is sometimes an appropriate reaction to a false accusation of patent infringement.

A litigation strategy involving patent claims can sometimes be dangerous. In the U.S., patent infringement claims are always litigated in District Court, and all appeals are automatically routed to the CAFC. The Court of Appeals for the Federal Circuit is a careful and sophisticated appeals court that often deals with complex technical issues. Patent lawsuits require a high level of diligence before they are filed so they don't waste the court's time.

Both a potential defendant in a claim for declaratory relief for patent infringement (before he writes the accusatory letter) and the plaintiff seeking that relief (before he files the declaratory relief lawsuit because of that accusatory letter) ought first to analyze the patent claims carefully. It soon turned out that K's patent infringement allegations were bogus. The plaintiff, J, and his wife need not have been so afraid. But by then the litigation path was fixed: Even without those mooted patent claims being relevant any longer and until a final resolution of the case, federal law requires that appeals be heard by the CAFC (or ultimately the U.S. Supreme Court). This automatically became an "important case."

A second characteristic of this litigation was the large catalog of additional tort claims asserted in J's complaint for declaratory relief. Some of those additional causes of action, of course, were the fundamental copyright claims that later became the focus of the CAFC decision. But the litigation was also burdened by other tort causes of action relating to trademark, unfair business practices, and breach of contract. Some claims involved state law, some federal law. Many of these non-patent claims involved complicated choices of law, with different issues of available remedies and different penalties, including attorney's fees. This case was for months mired in preliminary motions, all involving the non-patent claims. Dispute resolution attempts failed.

In such situations, it is not unexpected that the District Court would try to resolve this case as if it were a contractual dispute: If the Artistic License demands that licensees place attribution notices in their copies but they don't, determine the dollar value of those notices and assess damages. The District Court's decision to treat this as a simple contractual complaint would, in many situations, make it more likely that the parties would settle the dispute themselves for money. Such negotiated resolutions are the predilections of most courts because they encourage mediation or arbitration rather than expensive motion practice and trials. But these parties didn't settle, for various reasons unrelated to the software itself or its license.

All those matters were destined to be resolved in federal court because the declaratory relief patent claim carried those other claims along by supplemental jurisdiction. Federal court, and in particular the CAFC, is not usually the correct venue for such disputes, and so the CAFC used the law of the parties' local jurisdiction, California and the Ninth Circuit, in deciding this case.[2]

## Open Source Frames the Important Issue

This case deals with hobby software, given away by J under an open source license, subsequently used by K for minor commercial purposes, albeit without honoring an attribution provision in the license. The dollar value of a settlement on contract law terms would be small. At the scale of the vast software world the value of notices in J's and K's model train software is *de minimis*; such a claim cannot justify the cost of federal litigation.

More practically, for *contractual* disputes in the U.S., there is no automatic provision in federal court for attorney's fees. Clients take on such cases sometimes at great personal financial risk that they will end up with nothing but some small penalty assessed against a bad faith licensee, and a large bill for attorney's fees. (Of course, a contract can expressly provide for attorney's fees to the victor, but the Artistic License contains no such provision.)

*Copyright* law, on the other hand, permits a preliminary injunction as a remedy for infringement, and the law provides for attorney's fees in certain circumstances. Among the many remedies sought in J's complaint, injunctive relief for copyright infringement was perhaps the most important. Such an order from the court could stop K's business cold, no matter how large or small. And then K might also be

2 Jacobsen v. Katzer, 535 F.3d 1373, 1377-1378 (Aug. 13, 2008).

liable to pay attorney's fees.

It was understandable that K fought against that preliminary injunction in court. It was understandable that J demanded it.

A preliminary injunction usually requires at least some evidence of likely harm if the court doesn't take action and, in this particular case, there was no such ready factual evidence before the court. No matter, the District Court said, that a copyright infringement claim is appropriately and often resolved by a preliminary injunction, because that is irrelevant for contract claims. The District Court took a specific path toward resolving this dispute by treating the Artistic License as a contract, and thereupon applied contract law to refuse a preliminary injunction.

This became the fundamental issue of the case on appeal: Was it proper for the District Court to view this as a contract dispute rather than a copyright infringement dispute when considering the remedy of preliminary injunction?

That's also what caught the attention of theopen source and open content organizations when J appealed the District Court's refusal to order a preliminary injunction. That became the single issue upon which these groups had a shared interest: Unless a breach of a copyright license can be stopped outright by a preliminary injunction, there is often no value in a contract lawsuit, especially for works published for free public use and with small commercial value.

Only under copyright law—which generally permits preliminary injunctions as way of dealing with copyright infringement, and where attorney's fees can be assessed against bad faith defendants—is there some hope of enforcing the conditions that many authors demand for their works.

## The CAFC Analyzes the Important Issue

The CAFC boiled the *Jacobsen* case down to its fundamental issue: "The heart of the argument on appeal concerns whether the terms of the Artistic License are conditions of, or merely covenants to, the copyright license.[3]

Whether a provision is a condition or a covenant affects the way courts have previously dealt with copyright licenses. A "mere covenant" (particularly an "independent covenant"), when breached, is to be remedied under contract law. A "condition," on the other hand, is a limitation on the scope of the copyright license itself, and courts should treat its breach as copyright infringement.[4]

Then specifically analyzing the Artistic License, the CAFC said that J's requirement to publish attribution notices was a "condition" of the license. K's failure to do so was a use outside the scope of the license and therefore infringing.[5] The remedy for such infringement is found in copyright law. For copyright infringement, the court said, "where a copyright holder has shown likelihood of success on the merits of a copyright infringement claim," the Ninth Circuit has held that irreparable harm is presumed.[6]

The *Jacobsen* case was remanded "for further proceedings consistent with this opinion."[7]

3 Id. at 1380.
4 Id., and cases cited therein including Sun v. Microsoft, 188 F.3d 1115 (90th Cir. 1999).
5 Id. at 1381.
6 Id. at 1378.
7 Id. at 1383.

The people who wrote the Artistic License, and those who wrote the GPL, and those who wrote many other open source licenses, lucked out on the *Jacobsen* case. Many of us license authors didn't know the legal difference between a "covenant" and a "condition" when our licenses were written (and many attorneys still don't). Fortunately, the Artistic License states on its face that the document creates conditions, and so the CAFC found conditions in the Artistic License when we needed them most. If the court had found covenants in that license, open source might have lost this case.

This presumably does not mean that the new *magic word* for open source and open content licenses is "condition." The CAFC relied on a long-ago California Supreme Court decision to the effect that a condition can be found by "attributing the usual and ordinary signification to the language of the parties."[8] Thus the CAFC determined that the Artistic License, when it also uses the phrase "provided that," "denotes a condition" under California contract law.[9]

This does, however, raise an interesting question: Of the current approved open source and Creative Commons licenses, which of them clearly distinguish their *conditions* from their *covenants*, and under which state's *contract* law do we analyze that question?

## Other Important Lessons from the *Jacobsen* Case

The CAFC also resolved for us certain other concerns that arise when licensors try to enforce their open source copyright licenses. These strengthen our ability to set conditions for the use of our copyrighted works and to enforce those terms against those who don't meet those conditions.

At oral argument in the CAFC, the court asked the attorneys for both parties whether it is important that this software, like much other open source software, is licensed more for its reputational and cooperative value in the "community" rather than for financial reward. This was an important enough issue that it warranted a supplemental brief by the Creative Commons attorneys. The CAFC decision reflected the arguments in that brief.

The Artistic License was "clear" that it intended to "accomplish the objectives of the open source licensing collaboration, including economic benefit."[10] But it is not only an economic benefit that is at stake in the *Jacobsen* case.

> "The choice to exact consideration in the form of compliance with the open source requirements of disclosure and explanation of changes, rather than as a dollar-denominated fee, is entitled to no less legal recognition."[11]

The court's decision is very helpful for reassuring millions of copyright holders who engage in open source and Creative Commons licensing that they "have the right to control the modification and distribution of [their] copyrighted material."11[12]

The CAFC decision also requires district courts to enforce open source *license restrictions* (another term like "condition" that has ambiguous meaning here) through the remedy of injunctive relief. Otherwise, "those types of license restrictions might well be rendered meaningless."[13]

8 Id. at 1381, citing Diepenbrock v. Luiz, 159 Cal. 716 (1911)
9 Id
10 Id
11 Id. at 1382.
12 Id. at 1381.
13 Id. at 1382.

Even though injunctive relief is available, the plaintiff (J) is still required to prove that the conditions of the Artistic License were actually violated. The *Jacobsen* case was remanded for further action in District Court, not finally determined by the CAFC.[14]

As of the date this article is written, litigation continues. But we can take great comfort from this case already.[15]

## About the author

***Lawrence Rosen*** *is both an attorney and a computer specialist. He is founding partner of Rosenlaw & Einschlag, a technology law firm with offices in Los Altos Hills and Ukiah, California, that specializes in intellectual property protection, licensing and business transactions for technology companies.*
*In addition to this law practice, Larry also served for many years as general counsel and secretary of the non-profit Open Source Initiative (OSI). He currently advises many open source companies and non-profit open source projects including Apache Software Foundation and the Python Software Foundation. In 2005-2006 he was a Lecturer in Law at Stanford Law School.*
*Larry currently serves on the board of directors of International Characters, and on the advisory boards of SpikeSource, Black Duck Software, and JasperSoft.*



14 Id. at 1383.

15 I want to acknowledge the diligent efforts of Jacobsen's attorney, Victoria Hall (Law Offices of Victoria Hall), who persevered in this case despite my skepticism and won important additional allies to her client's cause. The legal heroes were two attorneys from Creative Commons, Anthony Falzone and Christopher Ridder, who under the leadership of Prof. Lawrence Lessig (Stanford Law School) submitted the amicus brief to the CAFC. Also providing legal support during litigation and appeal were Mark Radcliffe, David Gross and Steve Chiari (DLA Piper and counsel to Open Source Initiative (OSI)); Karen Copenhaver (Choate Hall and counsel to the Linux Foundation); Allison Randal (PERL Foundation, author of the newer and better version of the Artistic License); Roberta Cairney (Law Offices of Roberta Cairney and counsel to the PERL Foundation); and Scott Peterson (HP and a member of the OSI Legal Advisory Counsel). I was honored to work with them on this case. I apologize that this list is incomplete.

# Introducing the Risk Grid

*by Andrew Katz[a] and Shane Coughlan.[b]*

*(a) Moorcrofts LLP*
*(b) Opendawn Consulting*

**Abstract**
A Special Interest Group of the European Legal Network discussed issues around the commercial procurement of Free/Open Source Software, and methods to reduce or contain risk in transactions related to the supply chain. The initial focus of this group was on creating generic contractual language for use by Customers when establishing a relationship with a Supplier. However, it quickly became apparent that for a true solution far more than generic contractual language was required. There needed to be a guidance document to contextualise the scope of potential issues and to describe the potential remedies available for both Customer and Supplier regardless of their relative experience in Free/Open Source Software. To do this the members of the Special Interest Group created the Risk Grid, a table designed to describe the different ways in which publicly available code could be infringed, with rows to separate out each instance, and with example wording to help in drafting procurement contracts for software projects which make use of Free/Open Source Software components.

**Keywords**
Law; information technology; Free and Open Source Software; Risk management, contract law,

## Introducing The Risk Grid

In 2008 the European Legal Network founded a Special Interest Group to discuss the commercial procurement of Free/Open Source Software. The group initially focused on producing generic contractual language to minimise risk when sourcing code or products potentially involving such code. This was envisioned as text to be used by companies relatively knowledgeable in the field to regulate their relationships with suppliers. The implicit assumption – borne of market experience – was that suppliers might not adhere to Free/Open Source Software licence terms without explicit prompting from their customers.

While such an approach had its merits, it quickly became apparent that far more than generic contractual language was required. There needed to be a guidance document to contextualise the scope of potential issues and to describe the potential remedies available, and this document needed to be accessible to both the Customer and the Supplier irrespective of their relative experience in dealing with Free/Open Source Software. Andrew Katz, Malcolm Bain and Amanda Brock from the Special Interest Group began work on such a document, and created an overarching Risk Grid as a proposed solution to the problem.

The Risk Grid is envisioned as a document that describes various actions or risks involving software, and allocates each risk to the Customer and/or Supplier as appropriate. It is designed to de-

scribe the different ways in which publicly available code could be infringed, use table rows to separate out each instance, and provide example wording to help in drafting procurement contracts for software projects which make use of free/open source components. The assumption underlying the Risk Grid is that the best contracts are those which:

1. Explicitly identify the risks which arise from the transaction, and allocate those risks appropriately to the parties;

2. Recognise that risk can always be priced. However, the party generally best placed to bear the risk is the one which has control/knowledge of the circumstances giving rise to the risk and is therefore likely to be able to price it most appropriately. Indemnities are one appropriate mechanism for allocating risk;

3. Try to manage the expectations of the parties;

4. Are easy to understand.

The Special Interest Group focused on the areas which typically generate the most controversy in contract negotiation, and tried to encapsulate the arguments typically used by the supplier and the customer in relation to each area. Rather than being prescriptive, the Risk Grid suggests sample wording with various options to outline approaches that may be taken by either party. It is acknowledged that there cannot be a universal contract that will adequately provide for the requirements of a market where at one extreme there are software distributions being provided gratis and at the other there are bespoke management systems being built for specialised uses.

The Risk Grid is intended to assist with negotiation, and its utility extends beyond Free/Open Source Software. The document can equally assist in contextualising proprietary software transactions, or those where a mixture of Free/Open Source Software and proprietary software are involved. However, it is a work in progress and will by necessity be expanded and refined by those making use of it. Parties in different market segments and in various national jurisdictions may require quite substantial additions to ensure relevance. For example, in some jurisdictions, there is an emerging market providing insurance for IP infringement, and this has ramifications for the language used to off-set risk in purchasing contracts.

We are also working on skeleton wording for a precedent purchasing agreement, with definitions and terminology consistent with the Risk Grid, and enabling the chosen sample wording to be inserted in appropriate places. (The Risk Grid uses a number of terms which are capitalised - such as “Publicly Available Code”. Their meaning should be clear from the context, but the intention is that these terms will be defined in the precedent purchasing agreement).

Suggestions for improvement to the Risk Grid are welcome. The primary maintainer is Andrew Katz, and he can be contacted at <Andrew.Katz@moorcrofts.com> to discuss potential additions, alternations or ancillary guidance in applying the document.

## The Risk Grid

Text of final risk grid is attached at the end of this paper. Current version is 8. It is licensed under the conditions set forth at the end of this paper.

## Appendices

The Risk Grid is intended to be a largely self-contained reference document, but it will also be usable in conjunction with the planned precedent purchasing agreement. However, to assist with contextualising the transaction between Customer and Supplier it makes references to three appendices. These would have to be created by the Customer and/or Supplier to meet their requirements and annexed to the final contract between the parties. There is an overview of the intended content of each appendix below. These appendices will also be referred to in the planned precedent purchasing agreement.

### Appendix [1]

This appendix should list the locations from which any and all Publicly Available Code incorporated in the Software has been acquired.

### Appendix [2]

This appendix should list any and all guidelines to be followed to accurately document the source of each acquisition of Publicly Available Code incorporated in the Software.

### Appendix [3]

This appendix should list any and all licences regarded by the Purchaser as acceptable for the purposes of this transaction and/or the contractual relationship directly related to this transaction.

| Issue | Commentary | Who is best placed to bear risk? | Best mechanism to tackle risk | Sample Wording | Supplier's Arguments | Customer's arguments | Comments |
|---|---|---|---|---|---|---|---|
| Supplier-created code infringes copyright | The risk of detection of infringement is easier for [F/OSS] (as the code is more readily available for comparison purposes, especially if the code is GPL and re-distributed, but the ability of the customer to mitigate its loss is greater, as it automatically has access to the source code, to enable it to re-engineer infringing code itself if the Supplier will not or cannot do so. | Supplier | Indemnity/warranty from Supplier. Supplier has right to rewrite infringing code. Version control system (VCS) shared repository and allowing audit rights | The Supplier warrants that it has title to all Supplier-Created Code and that its delivery [assignment/licence] to the Customer and use in accordance with this Agreement does not infringe the [copyright] of any third party. | No good ones! | Supplier is in control of code creation, and should therefore be liable for third party infringements. Supplier should use a common source code repository, to which Customer may be given access. | |
| Publicly-available code (i.e. code acquired from third parties under a [F/OSS] licence, and incorporated into the software) infringes third party copyright. | One risk is that the publicly available code selected is inherently infringing (i.e. there is a provenance issue), or alternatively, the component is available under a [F/OSS] licence, but not the one attached to it. | Varies from project to project. If the Customer specifies use of a specific component, then it should be liable for claims in relation to that component. If the Supplier selects the components, there is a stronger argument that the Supplier should bear some of the risk, or at least take care in the selection process. | Warranty or indemnity from the Supplier, to encourage Supplier to take care in source selection. A list of agreed sources of code may give the Customer comfort (even if this is by no means conclusive), and may encourage the Supplier to take fewer risks in terms of provenance. Further, if code is obtained from recognised locations, it is more likely to be heavily reused, and therefore there is arguably safety in numbers (i.e. it's been used lots of times before and there hasn't been a claim yet), and also the likelihood that if it is found to be infringing, the community will generate a non-infringing alternative | The Supplier warrants that each component of Publicly Available Code incorporated in the Software has been acquired solely from the locations listed in Appendix [1] and that the source of each such acquisition shall be accurately documented [as set out in Appendix [2]]. | Each Customer has a different appetite for risk. Requiring the Customer to document how it regards the risk of accessing code from different locations, gives the Supplier more information on which to base an accurate price for the job. Alternatively, Supplier may want to give the Customer the option of a cheaper price by doing "quick and dirty" development by scraping code from anywhere, without provenance checking, providing that the Customer takes the risk. In any case, this clause as drafted could prove unduly restrictive for the Supplier. There are vast amounts of quality code available from "grey" sites. Also, is "reasonable skill and care" capable of consistent interpretation given the state of the art? Koders.com contains plenty of roll-your own licences, for example. Also, just because something is on sourceforge.net does not mean that it is necessarily of any better provenance than elsewhere. | Supplier is contracting to supply IPR, and should bear all the risk. How Supplier intends to source IPR should not be Customer's issue. In any event, where the Supplier is actively choosing the code to use, provenance checking should be a selection criterion. | Infringement can occur either because the infringing code is not available under any [F/OSS] licence (e.g. it is derived from proprietary code), or because it is not available under the licence supposedly attached to it (e.g. it is available under the GPL, but appears to be available under the BSD). |
| | Another risk is that the Customer may specify the use of specific [F/OSS] components, and in using these components faces a similar issue as above, though with a different context for allocating potential liability. | If the Supplier selects the components, there is a stronger argument that the Supplier should bear some of the risk, or at least take care in the selection process. If the Customer performs this selection, the opposite is true. | Customer takes all risks relating to the nominated code. | The Customer acknowledges, notwithstanding any other provision of this Agreement, that the Supplier shall not be responsible for any claim, cost or expense howsoever arising from the Supplier's incorporation, use of, modification of, linking to the Customer's Specified Components [and the Customer shall indemnify the Supplier for any cost, claim or expense arising therefrom]. | The Supplier's choice of component is restricted, and therefore it should not be held liable for such use. | | |

| Issue | Commentary | Who is best placed to bear risk? | Best mechanism to tackle risk | Sample Wording | Supplier's Arguments | Customer's arguments | Comments |
|---|---|---|---|---|---|---|---|
| | It is possible to explicitly address the risk of publicly available code not being available under the licence apparently attached to it, and instead actually falling under a different licence and potentially incompatible licence. | This is similar to the provenance issue, in that the Customer's use/modification/distribution of the Software may infringe third party rights, but in this case, infringement may depend on the Customer's intended out-licence or intended use of the Software. This wording contains an option which limits the Supplier's obligations to checking that the components' attached licences are on an approved list, but not that they are compatible with any intended use. | Warranty relating to the licences attached to publicly-available code components. Optional exclusion of liability for licence incompatibility (Customer takes risk of incompatibility). | [The Supplier warrants that[, so far as it is aware,] each component of Publicly Available Code incorporated in the Software is available under one of the licences specified in Appendix [3] and has documented the provenance of each such component [as set out in Appendix [1]][ The Supplier does not warrant that use, modification or distribution by the Customer of the Software will not infringe the rights of any third party, and no provision of this Agreement or implied term shall be construed as such a warranty]. | The Supplier does not want to be responsible for ensuring licence compatibility, as the Customer will be much better placed to determine what its intended use is. Therefore, it's more practical for the Customer to specify a list of compatible licences, than having the Supplier do compatibility checks. | The Customer selects code | |
| | Sweeper up warranty designed to ensure that code-selection for copyrights is within the ambit of the Supplier's services. | Supplier | Warranty that skill and care has been taken in component selection, so far as third party copyrights are concerned | [The Supplier warrants that it has taken reasonable skill and care in selecting publicly available components having regard to the non-infringement of third party copyrights [the Customer's Specified Use and the Customer's Specified Out-Licence], and has documented the provenance and licences applicable to such components [as set out in Appendix [1] and [2] [with reference to Appendix [3] where applicable]].] | This warranty is too vague, at least without qualification as to whether the licences which are attached to the components are compatible with the Customer's Specified Use or (preferably) the Customer's Specified Out-Licence. | The Supplier needs to be put under a practical obligation to make copyright compatibility/awareness part of its selection criteria. | |
| | Publicly available code is incompatible with the Customer's Specified Use or Specified Out-Licence. By requiring the Customer to specify in this way, expectations are managed, and minds are focused | Supplier | | The Supplier warrants that [so far as it is aware, but without having made any specific enquiry] the development of the Software, its delivery to the Customer and the Customer's modification, distribution and use of the Software within the Specified Use [or relicensing to third parties within the Specified Out-Licence] shall not infringe the licences set out in Appendix [3]. | This warranty places the onus on the Supplier (at least without the awareness qualification) to ensure compatibility, which can include a legal analysis of different licences, which may be outside the scope of the ability of the Supplier, or the scope of the services intended to be provided. | The Customer has taken time to specify either the licences to be used, or the Specified Use, and it is up to the Supplier to ensure that the Software complies with this requirement. | |
| Infringement by misuse of third party code by the Customer. | | Customer | | [The Customer is responsible for ensuring that its own subsequent use, modification and re-distribution of the software [outside the Specified Use] is in accordance with [the licences set out in Appendix [3]]. | The Supplier is developing for the Customer. Therefore the Supplier is not to be concerned about out-licensing, outside the scope of the specified use. This is the Customer's issue. Any future or different uses would be subject to a future or different agreement. | The Customer may want to distribute in the future, and may want to out-license to customers etc. Also, passing around the group, or to the acquirer of the business may be "distribution" and therefore should be covered. | |

| Issue | Commentary | Who is best placed to bear risk? | Best mechanism to tackle risk | Sample Wording | Supplier's Arguments | Customer's arguments | Comments |
|---|---|---|---|---|---|---|---|
| Infringement of copyright in bought-in proprietary code | | Supplier (through contractual relationship with provider of the proprietary code) (unless use of that component is nominated by the Customer – see above) | Indemnity/warranty from Supplier - but can Supplier obtain a back to back indemnity from the provider of that code? | The Supplier [confirms that the licences under which the third party components of the Software are available [are contained within the list set out in Appendix [3] as amended from time to time by agreement between the parties]][will not be breached by the Customer's Specified Use][,permit the Customer to out-license the Software under the Specified Out-License] and that so far as it is aware [but not having made specific enquiry] the development of the Software and its delivery to the Customer do not infringe such licences. [The Customer is responsible for ensuring that its own subsequent use, modification and re-distribution of the software [outside the Specified Use] is in accordance with such licences.][The Supplier agrees to provide reasonable assistance to the Customer in passing the benefit of any warranties associated with such third party [proprietary] components to the Customer subject to the Customer's continued compliance with the licences applicable to such code. | Supplier to use reasonable skill and care in selecting code, but should not be liable for third party infringement. Similar to the supply of third party hardware. May offer to pass on any third party warranties available. May also be subject to the Customer complying with terms passed through by the Supplier. | Supplier is contracting to supply IPR, and should bear all the risk. How Supplier intends to source IPR should not be Customer's issue. | |
| Infringement of patent in Supplier Created Code | | Where Supplier has choice of implementation: Supplier. Where implementation is dictated by Customer's requirements: Customer | Right to change implementation, if implementation is determined by Supplier. Otherwise, risk is on Customer. May be possible to negotiate risk sharing. May be possible to get insurance? Audit rights? | The Supplier warrants that [so far as the Supplier is aware [not having made any enquiry]] the use by the Customer of the Software for its Specified Use [within [jurisdictions]] will not infringe any right which any third party may hold under any valid patent. | It is not economically feasible to undertake a patent clearance prior to implementation. If the implementation is dictated by the Customer's requirements, this should not affect liability. | Supplier is contracting to supply IPR, and should bear all the risk. How Supplier intends to source IPR should not be Customer's issue. | |
| Infringement of patent in publicly available code | | Where Supplier has choice of implementation: Supplier. Where implementation is dictated by Customer's requirements: Customer | Where implementation is dictated by Customer: Customer to bear risk. Otherwise, negotiated on a case by case basis. | <none> | It is not economically feasible to undertake a patent clearance prior to implementation. If the implementation is dictated by the Customer's requirements, this should not affect liability. If supplier has to accept some liability for patent infringement, Again there is the potential to insure against this in the UK at a high price and the additional costs of this would be passed through to the Customer. | Supplier is contracting to supply IPR, and should bear all the risk. How Supplier intends to source IPR should not be Customer's issue. | |
| Infringement of patent in bought-in proprietary code | | Where Supplier has choice of implementation: Supplier. Where implementation is dictated by Customer's requirements: Customer | Where implementation is dictated by Customer: Customer to bear risk. Otherwise, negotiated on a case by case basis. Can Supplier obtain a back to back indemnity from the proprietary Supplier? | <none> | Supplier to use reasonable skill and care in selecting code, but should not be liable for third party infringement. Similar to the supply of third party hardware. May offer to pass on any third party warranties available, or to assist and again this may be subject to a pass through of third party restrictions. | Supplier is contracting to supply IPR, and should bear all the risk. How Supplier intends to source IPR should not be Customer's issue. | |
| Trade secrets | | Supplier | | The Supplier warrants that, to the best of the Supplier's knowledge [but not having made any specific enquiry], its delivery [assignment/licence] to the Customer and use in accordance with this Agreement does not breach any obligations of confidentiality to a third party. | | | |
| Trademarks | | Customer | | For the avoidance of doubt nothing in this Agreement [except for clause []] is intended to grant any licence over any trade mark of the Supplier or its licensors. The Customer shall comply with the terms of the licences governing all third-party components comprised in the Software, which may include terms relating to trade marks. | The Customer may wish to use the Supplier's trade mark if the code is distributed (or accessed remotely). The parties may rely on trade mark law to tackle this, or incorporate an explicit licence permitting the use of the trade mark in relation to the Supplier's code only if it is not modified in any way. | | |

| Issue | Commentary | Who is best placed to bear risk? | Best mechanism to tackle risk | Sample Wording | Supplier's Arguments | Customer's arguments | Comments |
|---|---|---|---|---|---|---|---|
| General Indemnity Wording | | | | The Supplier will indemnify and hold the Customer harmless on demand against any claim or loss arising as a consequence of a breach of any of the [above warranties – warranties set out in this clause]. | | | |
| Implied terms, pre-contractual representations | | | | Except as expressly set out in this Agreement, the Supplier makes no representations or warranties in respect of or in connection with the Software or its use. All other representations, warranties, conditions or other terms which might have effect between the parties or be implied or incorporated into this Agreement or any collateral contract, whether by virtue of statute, common law or otherwise, are hereby excluded to the maximum extent permitted by law, including, without limitation, implied conditions, warranties or other terms as to satisfactory quality, merchantability, fitness for purpose or the use of reasonable skill and care. | | | |
| Conduct of Claim | | | | The Customer shall notify the Supplier promptly ("a Claim Notice") should it receive any claim that any portion of the code delivered under this Agreement infringes the rights of any third party, or where it otherwise has reason to believe that it does so. The Supplier's obligation to indemnify the Customer under [clause [ ]] in connection with a claim against the Customer by a third party is subject to: (a) the Customer promptly serving a Claim Notice; (b) the Customer not making any admission as to liability or compromising or agreeing to any settlement of any such claim without the prior written consent of the Supplier[, which consent shall not be unreasonably withheld or delayed]; (c) at the Supplier's written request and at its own expense, the Supplier having the conduct of and the right to settle all negotiations and litigation arising from such claim; and (d) at the Supplier's request and expense, the Customer giving the Supplier all reasonable assistance in connection with such negotiations and litigation. [The Customer shall take all reasonable steps to mitigate its loss arising from any default of the Supplier] | | | |
| Access to CVS repository | | | | The Supplier undertakes that it will [during the Term] allow the Customer [read-only] access to the [CVS Repository]. | | | |
| Replace or Re-write | | | | The Supplier may at any time replace any part of the code ("the Original Portion") delivered under this Agreement where it reasonably believes that such code infringes the rights of any third party or where a claim of such infringement has been made, provided that such replacement code materially complies with the Specification. The Supplier shall cease to be liable to the Customer for any claim relating to the Original Portion to the extent that it arises after delivery of the Replacement Code, except where such claims apply to items already created or manufactured and currently being deployed to market. | | | |
| Licence of Collective Work | The Software is likely to consist of a number of components, and the list of components itself will amount to a collective work. Although in many jurisdictions, the collective work will be implied, in some jurisdictions, e.g. Spain, it may need to be explicitly granted. Note also that the GPL may not be an appropriate licence for a collective work – FDL, or creative commons may be more appropriate as they do not introduce source code complications. | | | The Supplier acknowledges that the combination of the components within the Software constitutes a collective work. The Supplier hereby grants a non-exclusive licence to such collective work to the Licensee [consistent with the rest of this Agreement] [consistent with the Specified Use] | | | |

| Issue | Commentary | Who is best placed to bear risk? | Best mechanism to tackle risk | Sample Wording | Supplier's Arguments | Customer's arguments | Comments |
|---|---|---|---|---|---|---|---|
| Limitations and exclusions of liability | | | | The Supplier's liability under or in connection with this Agreement (whether in contract, tort (including negligence) or otherwise) is limited as follows: (a) the Supplier will have no liability for any loss of profits, loss of business, loss of goodwill, loss of anticipated savings, loss of or corruption to data or for any indirect or consequential loss or damage; and (b) the maximum aggregate amount of any such liability which is not excluded by (a) shall be [ ]. Nothing in this Agreement shall limit the Supplier's liability for death or personal injury or arising as a result of fraud. | On a risk and reward basis the Supplier will wish to limit to the fees for the specific project. | | |
| Status of Supplier | This needs to be considered carefully in the context of each licence. Generally, the Supplier will want to be providing services to the Customer, rather than deliverables. This has issues for distribution, acquired rights directive, liability. | | | The Supplier is [an independent contractor][acts as Agent for the Customer in developing the Software] | | | |
| Failure of software to meet specification: Supplier created | Note that the source is automatically available. No need for escrow. More natural to have documentation available. | Supplier | Warranty from Supplier + ability to re-write non-performing code | To the extent that any Supplier-Created Code fails to meet the Specification, the Supplier shall during the Warranty Period [replace such Supplier Created Code with code that is compliant][insert SLA] | Offer SLA? Maintenance agreement. Warranty period. Source is automatically available | Warranty that Software will perform to spec. | |
| Failure of software to meet specification: publicly available | | Supplier, generally | Warranty (negotiated) from Supplier + ability to re-write non-performing code | To the extent that any Publicly-Available Code fails to meet the Specification, the Supplier shall during the Warranty Period [replace such Publicly Available Code with code that is compliant][insert SLA] | The Supplier should not be responsible for the performance of third party code. | The Customer should not be concerned about how the Supplier opts to select code. Further, for Publicly-Available Code, the Supplier has access to the source, and can therefore treat that code as simply a more-rapidly-developed version of its own code. There is therefore no reason why it cannot give a warranty. | |
| Failure of software to meet specification: proprietary | | Original supplier - can supplier pass on warranties etc? | Back to back warranty from supplier, or mechanism to enable customer to benefit from original suppliers' warranties (agency, third party beneficiary, collateral warranty) | The Supplier shall take reasonable steps to assist the Customer with the enforcement of any warranties applicable to proprietary code, but shall [except to the extent that no reasonable supplier could have specified the use of such code] not otherwise be liable for any failure of any third party code to reach Specification. | Industry standard to use third party code. Depends on type of code (OS/database engine/DLL/Embedded component) | Software should perform to spec. | |

# Jacobsen v Katzer and Kamind Associates – an English legal perspective

*Mark Henley*[a]

*(a) Director at Wragge & Co. LLP.*

**Abstract**

In this article Mark Henley examines, from an English lawyer's perspective, the decision of the US Court of Appeals for the Federal Circuit in the case of Jacobsen v Katzer and Kamind Associates. He also considers some of the implications of the Court's findings for the Free and Open Source Software communities.

**Keywords**

Contract law, Copyright law, Licensing, Jacobsen v Katzer and Kamind Associates, Jacobsen, Ninth Circuit, Free and Open Source Software, Free Software, Open Source Software, England and Wales

In The Canterville Ghost, Oscar Wilde wrote: "We have really everything in common with America nowadays except, of course, language." It is quite possible that the analytical techniques of English and American lawyers were not uppermost in his mind when he composed that famous line but, on the evidence of *Jacobsen v Katzer and Kamind Associates*[1], it remains surprisingly accurate; there appears to be a distinct difference between how lawyers on each side of the Pond would approach the interpretation of the same, relatively simple, FOSS licence.

Court judgements which look at FOSS licences in detail are extremely rare and there is a complete absence of case law for England & Wales. When cases arise they pique the interest of the international FOSS community, even if the decisions do not directly establish a binding precedent. Being set, furthermore, in the glamorous world of the model railway enthusiast, the Jacobsen case was always destined to receive widespread attention.

In England, similar to the Ninth Circuit, interim injunctive relief is commonly granted in copyright infringement cases and is relatively unusual in plain contract cases. The outcome of the *Jacobsen* appeal did not surprise English lawyers. Interim injunctive relief would most likely have been available here in similar circumstances to prevent a licensee from using FOSS without observing the terms of the relevant copyright licence.

However the confusion begins when the fundamental issue for the Court of Appeals is whether Prof. Jacobsen's claim should have been brought as a contract dispute or for copyright infringement. For English lawyers, the option of bringing a claim as a contract dispute will only be available where all of the requirements for formation of a contract are met.

Under English and US legal systems a licence may or may not be contractual; if non-contractual, it is known as a "bare licence." A simple example of a bare licence would be a cinema owner's

1 For the background to this case and a commentary from a US legal perspective, please see Lawrence Rosen's IFOSS L. Rev. article "Bad Facts Make Good Law: The Jacobsen Case and Open Source" at http://www.ifosslr.org/index.php/ifosslr/article/view/5

permission for you to enter the foyer of the cinema to enquire about the start time of the new John Grisham movie and to stand in line for the box office. There is no exchange of promises at that point, no consideration given. If you go on to buy a ticket, then at the moment of purchase you will be upgraded to a contractual licence permitting you to enter the auditorium and watch your legal thriller. The licence becomes an ancillary provision of the contract for watching the movie.
An English court would not assume that a copyright licence was contractual. It would seek initially to establish the existence of a contract by looking for the required elements of offer, acceptance, consideration and an intention to create legal relations. The differences between a contractual and a bare licence are, for the following reasons, too significant for this preliminary test to be overlooked.

First, the terms of a contractual licence may be enforced against both licensor and licensee. In contrast, under a bare licence, the licensee cannot bring a claim against the licensor.

Second, when interpreting a contract, a court will look beyond the literal meaning of the words and seek to ascertain the objective intentions of both parties. It may also imply or disallow terms in certain circumstances to give effect to public policy.

Third, a contractual licence may be terminated only in accordance with the terms of the contract under which it was acquired. A bare licence may be revoked at the licensor's will or perhaps on giving the licensee reasonable notice (although estoppel arguments may assist a licensee who has relied on the licence to his or her detriment).

Fourth, the governing law may differ between a bare and a contractual licence. In the UK, the governing law for a contract dispute is determined by the Rome Convention on the Law Applicable to Contractual Obligations[2]. However, for a non-contractual dispute it will be determined by the Rome II Regulation[3] or another statute of Private International Law.

Finally, if a licence is a contract then it is possible that the remedy of specific performance might be granted by a court in the event that its terms are broken. Specific performance is an order that someone will do what he or she has promised to do and, for the courts of England & Wales, it is a discretionary remedy that will only be granted if damages are not an adequate remedy. For a FOSS licence like the GNU GPL (the most popular licence for FOSS projects at the time of writing), specific performance of the obligation to disclose source code would be an extremely powerful remedy for the licensor.

Returning to the *Jacobsen* case, the Court of Appeals applied the principle that whether breach of licence is actionable as copyright infringement or breach of contract turns on whether the provision breached is a condition of the licence, or a mere covenant.[4] In doing so the judges appeared to presume that the Artistic License was a contract; how else could a breach of one of its covenants amount to a breach of contract?[5] They even acknowledged that consideration, a critical

2 CONVENTION ON THE LAW APPLICABLE TO CONTRACTUAL OBLIGATIONS opened for signature in Rome on 19 June 1980 (80/934/EEC), http://www.rome-convention.org/instruments/i_conv_orig_en.htm

3 REGULATION (EC) No 864/2007 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 11 July 2007 on the law applicable to non-contractual obligations, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2007:199:0040:0049:EN:PDF

4 For an explanation of the distinction drawn by the Court between conditions and covenants see Lawrence Rosen's article "Bad Facts Make Good Law: The *Jacobsen* Case and Open Source", ibid.

5 One theory might be that the Court of Appeals has adopted contract law language and contract law analytical tools to resolve a dispute that they believe is not, in fact, contractual. That seems unlikely given that the Court cites the principle in *Graham v James*, 144 F.3d 229 at 236-37, ("whether breach of license is actionable as copyright infringement or breach of contract turns on whether provision breached is condition of the license, or mere covenant") as sitting at the heart of the argument. The Court appeared to believe that breaches of the Artistic License could be actual breaches of contract, not that they merely ought to be treated **like** breaches of contract. It is also difficult to see

component for contract formation, may be present, finding that "The choice to exact consideration in the form of compliance with the open source requirements of disclosure and explanation of changes, rather than as a dollar-denominated fee, is entitled to no less legal recognition."

Some commentators[6] have similarly sought to characterise the GPL as a contract. On one view the GPL starts out as a "unilateral contract" – an offer made to the world by the author to use his/her software in compliance with certain conditions – where the normal requirement for communication of acceptance is waived by the licensor. That offer can be accepted by conduct and a standard bilateral contract will result. Section 0 of GPL v2 states that "The act of running the Program is not restricted…" but at section 5 provides that "by modifying or distributing the Program (or any work based upon the Program), you indicate your acceptance of this License to do so…."[7] This wording appears consistent with the idea that the GPL starts out as a bare licence for the purpose of running the Program, but is converted into a bilateral contract if modification or distribution takes place.

However, some of the most influential lawyers in the FOSS community take the view that the GPL is not a contract at all. Eben Moglen, former GC of the Free Software Foundation has observed[8], "The GPL is a very simple form of copyright license, as compared to other current standards in the software industry, because it involves no contractual obligations." Pamela Jones of Groklaw,[9] Daniel Ravicher of the Software Freedom Law Center[10] and Lawrence Rosen, former GC of the Open Source Initiative[11], appear to take similar positions. Such an approach makes good sense if you are encouraging uptake of a one-to-many free licensing model. A bare licence will insulate FOSS developers from the claims that could potentially be brought by thousands of dissatisfied licensees and from the unpredictable consequences that can sometimes ensue when judges attempt to give effect to the intentions of two or more parties.

Given that the GPL is even more consistent with the language of contractual rights and obligations than the Artistic License, it is reasonable to suppose that the courts of the Ninth Circuit will also characterise the GPL and perhaps other FOSS licences as contracts. Much will, of course, depend on the cases that follow *Jacobsen*, and perhaps that decision will be distinguished on the basis of the particular wording of the Artistic License.

If *Jacobsen* were heard before the courts of England & Wales, there would be every likelihood that no contract would be found and the Artistic License would be considered a bare licence. Breach of its terms might take the licensee outside the scope of the licence or alternatively might entitle the licensor to revoke the licence even without providing reasonable notice. Either way, interim injunctive relief might be available to prevent further "unauthorised" use by the licensee pending trial.

If, on the other hand, the Artistic License was found to be a contract, the courts of England & Wales would probably take a similar approach to that in *Jacobsen*, considering whether there had been breach of a condition or of a less critical term. If a condition was breached then that might

why, as a matter of principle, contract law analytical tools, focused as they are on a notional meeting of minds, should be applied to determine the scope of a unilateral bare licence.

6 Jason B. Wacha, Taking the Case: Is the GPL Enforceable?, 21 Santa Clara Computer & High Tech. L.J. 451 at 456 (2005); Robert W. Gomulkiewicz, De-bugging Open Source Software Licensing, 64 U. Pitt. L. Rev. 75 at 83 (2002).

7 A similar provision appears in section 9 of GPL v3.

8 http://www.gnu.org/press/mysql-affidavit.html

9 http://www.groklaw.net/article.php?story=20031214210634851

10 http://radio.weblogs.com/0120124/2003/07/23.html

11 Lawrence Rosen, Open Source Licensing: Software Freedom and Intellectual Property Law (2005) at pp. 65, 138 and 139. Note, however, that page 140 and http://www.rosenlaw.com/lj20.htm clarify that Rosen sees the GPL as an exceptional case and believes most other FOSS licences to be contractual.

take the licensee's use of the FOSS outside the scope of the licence entirely. Alternatively, breach of a condition might entitle the licensor to consider the contract repudiated by the licensee and, if that repudiation was accepted by the licensor, the contract would terminate and the copyright licence along with it. In any event, as with a bare licence, use of the software in breach of the licence conditions might still constitute copyright infringement and interim injunctive relief again might be available to prevent use from continuing, pending trial.

So the *Jacobsen* case presents a win for the FOSS community in that interim injunctive relief is, in principle, available to stop licensees from disregarding the terms of FOSS licences. However the sting in the tail – that the Artistic Licence is a contract and that the courts of the Ninth Circuit may be expected to treat the GPL and even relatively permissive open source licences the same way – could turn out to have a chilling effect on the FOSS movement in the longer term if, as discussed above, the result is that FOSS developers become exposed to claims brought by dissatisfied licensees.

## About the author

***Mark Henley*** *is a Director at Wragge & Co. LLP, a major UK law firm providing a full range of legal services to businesses worldwide. He can be reached at mark_henley@wragge.com.*

# Book Review: 'Open Source Technology and Policy' by Fadi P. Deek and James A.M. McHugh

*Andrew Katz*[a],

*(a) Moorcrofts LLP*

**Abstract**

Andrew Katz casts a critical eye over Fadi P. Deek and James A.M. McHugh's three-part study of open source principles and practice, assessing the relevance of the volume's research, and the merits of its balanced approach to introducing new software development models.

**Keywords**

Law; information technology; technology policy; public policy; Free and Open Source Software

Richard Dawkins believes that god may exist. He just believes that the probability of his existence is vanishingly small. He also believes it possible that something called epigenetic inheritance (that parents can pass on traits to their offspring by mechanisms other than genetic transmission – DNA) exists, although the overall effect of epigenetics is very small. To a small-minded critic, it would be easy to look at these two positions, and decide that Dawkins is neither an atheist, nor does he stand by the central thesis of *The Selfish Gene*: in other words, that the two ideas which are the central core of his standing as a leading public intellectual, and which he would regard as his life's work, are flawed.

But Dawkins is a scientist, and scientists do not deal in certainties. Good scientists are also rigorously, and intellectually honest: if they have doubt in any aspect of their theory, the area of doubt becomes the focus of their endeavours, and they will publicise their doubts to other scientists. That is how trust is developed in the scientific community, and how it should develop in any context which relies on the power of enquiry and reason. And I assume that lawyers, and all readers of the International Free and Open Source Software Law Review, are firmly on the side of reason.

In contrast, in the sphere of fundamentalist religion (and politics, for that matter), doubts are actively suppressed: a single doubt becomes a crack which the holders of the sacred truths fear can be pried open. This leads to a certain mindset: the avoidance of certain difficult questions, schisms within certain belief/dogma systems, and ultimately the labelling of particularly recalcitrant individuals as heretics.

The parallels between religion and free/open source software are not difficult to see. Richard Stallman even encourages them, and casts himself as "Saint IGNUcius", replete with appropriately flowing beard and locks (and sometimes a halo). It's obligatory for any mention of him (including this one) to be accompanied by the word "evangelical". This world even has its seemingly inevitable catholic/protestant, sunni/shi'ite schism: free/open source software. (Although just to clarify, this is an analogy: Stallman firmly regards himself as an atheist: see

http://www.stallman.org/extra/personal.html).

And like religions, there are the sacred texts. We can argue about the relative importance of these: to the Free Software fanboys, GPL2 is the King James Version, GPL3 is the increasingly-accepted New English Bible. Open sourcers tend to carry around a well-thumbed copy of Eric Raymond's *"The Cathedral and the Bazaar"*.

The quasi-religious nature of the movement means that the free software movement (and to a lesser extent, the open sourcers) have articles of faith that are not permitted to be questioned. Novitiates will find themselves dissuaded from asking the following questions: "How does it make money?", "How can anything that's free (i.e. gratis) be any good?", "Who can I sue if it all goes wrong?", "How on earth can a bunch of self-selecting volunteers with an ad-hoc management structure, no training, and without any vetting for appropriate qualifications write software which effectively competes with the world's most successful, and can safely be used to run hospitals, transport systems, nuclear power stations and the world's financial markets" [on the last of those: don't blame the software]. It's regarded as naïve to ask these questions, as if everyone should already know the answers, when in reality this attitude disguises a fear of the questions themselves.

Not surprisingly, a failure by the community to address these issues directly leads to the conclusion by outsiders that the FOSS movement is based more on sentiment than reason: and in certain cases, this conclusion would seem fully justified. It is compounded by the one article of faith which all adherents universally cling to (i.e. the identity of Satan) which doesn't do much to dispel the "stick it to the man" Abbie Hoffman/hippie/School of Rock ethos which is still perceived to pervade much of the movement. (By the way, if you've ever wondered what it would feel like to utter the word "Voldemort" at Hogwarts, try visiting the offices of a major Linux distribution and saying the "M" word).

When I started reading Deek and McHugh's *Open Source Technology and Policy* I was underwhelmed by how low key it was. Most literature relating to FOSS (and its related areas) has been polemical: read any essay by Richard Stallman or Eric Raymond's *The Cathedral and the Bazaar* or anything by Larry Lessig and you will find it exhilarating, occasionally frustrating but ultimately intellectually challenging.

However, as I read further, it became increasingly clear that this book, with its careful and sober assessment of the past, present and potential future for free and open source software, directly answers the forbidden questions head-on. It does not seek to obscure the failings of free and open source software, and is, in the end, a more persuasive advocate of FOSS, and the development model which created so much of it, than many of the more standard texts.

It's the mark of the maturity of free/open source world that a book like this can exist. The book is a comprehensive snapshot of the state of the world of FOSS, divided into three sections: the techy section (Open Source – Internet Applications, Platforms and Technologies); the socioeconomic section (Social, Psychological, Legal and Economic Aspects of Open Source); and a context section (Free Software: the Movement, the Public Sector and the Future).

The style can be a little dry at times, but its academic tone is balanced by an equally academic and invaluable bibliography at the end of each chapter. There are also a couple of places where generalizations are stated as fact (and had me making notes: "no! not correct" ) and then over the course of the following paragraphs, the more complex facts behind the generalization are revealed. For this reason, although as a whole the book is pretty accurate, and Deek and McHugh clearly

know their stuff, it's dangerous to quote a paragraph out of context without having read the whole section. The insertion of the word "generally" here or there would have been helpful, as would the wholesale deletion of exclamation marks (which appear from time to time in inappropriate places, like Dr. Hibbert's chuckle in *The Simpsons*).

Readers need to be aware that, inevitably, there have been significant changes in a number of areas since the book was published. It seems that the final research was undertaken in January 2007, so sections in relation to the SCO case, for example, have now been superseded by events. This is not a criticism: merely an issue that readers should be aware of.

Deek and McHugh are clearly proponents of freedom and openness, but are balanced in their approach. This has the welcome effect of making the book a much less inflammatory introduction for readers locked in the proprietary mindset.

The case for freedom and openness is made so much more strongly when the answer to the question "Does the Open Source Software Model Work?", is "Yes, frequently it works extremely well, but sometimes it doesn't work at all", rather than "Of course: always. Why are you even bothering to answer the question?".

This book, therefore, is recommended as a comprehensive introduction to the free software/open source ecosystem, and, strangely, by failing to be strident and shrill, it manages in many ways to be a much better advocate than many more polemical texts. By an honest awareness of their subject's shortcomings, the authors, like Richard Dawkins, advance their cause more effectively.

# Tech Watch

*Adriaan de Groot*[a]

*(a) Vice President of Legal Affairs, KDE e.V.*

IFOSS L. Rev's Tech Watch section aims to allow technical experts and organisational leaders in Free and Open Source Software to introduce and explain topical issues with important legal aspects. These discussions may form the basis of subsequent detailed examination by IFOSS L. Rev. contributors.

This issue, KDE e.V. Vice President of Legal Affairs Adriaan de Groot reviews some of the issues presently confronting community software authors: copyright consolidation, making a living from coding, and 'doing legal stuff right'.

**Keywords**
Law; information technology; Free and Open Source Software; Bacula.org; Fiduciary Licence Agreement; Copyright; Copyright consolidation

Insofar as there is a "typical free software person," one might think of a software developer, the kind that sits up late writing code licensed as free software. Of course, here I mean free as in freedom to copy, modify, and distribute, although the real driving force for a developer may be the itch to create something useful or beautiful. At times like that, a developer's thoughts about legal issues are far, far away.

However, the real world intrudes in the end. Software needs to be used to be useful, and most developers want to distribute their software to as many users as possible. Distribution (or even assisting distribution, as in the Pirate Bay[1] case) means interacting with the rest of the world. One's private creative work is to be published with all the attendant details. Unlike many creative professionals, free software developers, especially the hobbyists, do not have agents to help with the legal issues and mechanics of distribution.

This journal is about the legal issues that attend free software development, publication and distribution in the broadest sense. It is a legal journal, largely from lawyers' perspectives. This column looks at what has been (or what should have been) keeping free software developers busy on the oft-neglected legal side of their work.

1 http://www.piratebay.org

The biggest news on the copyright, patent and trademark front in the month of April 2009 is perhaps the initial verdict in the case of the Pirate Bay.[2] While it is of considerable importance to a large population of file-sharing enthusiasts, the impact on software developers and free software projects seems to be limited. One might argue about contributory infringement and how the development of software applications for file transfer can interact with the law. For the most part, though, free software developers are interested in working within the law, since it is that law -- and copyright in particular -- that makes free software licenses possible and necessary.

A developer does not exist in a vacuum, though. Assuming the software is interesting and useful to others as well, a community grows around that software. The community is more than only the developer -- there are users, writers, artists, system administrators and possibly even community managers (an odd job title, but one that flourishes in the free software world). "How do you eat?" is a common question, and securing the livelihood of all the members of a free software community is something that may require sound legal footing.

Two debates wandered around the free software world this month on topics directly related to legal matters. Both crossed the path of Tarus Balog, the lead developer of OpenNMS, a network monitoring suite released under the GNU GPL. OpenNMS is free software developed by a fairly large community, centrally managed and maintained, and supports a business. It is therefore a crossroads of forces around free software. The discussions spilled out across many different projects, especially after a discussion at the Open Source Business Conference[3] on community and licensing.

One debate is on the principles of licensing: what constitutes open source software? Where do paid-for services or code fit in to the free software world? Who controls the code? Where do the rights to the code reside? (These topics are culled from all of the blog entries that spawned around the original discussion; you might want to start at Aaron Seigo's blog post on copyright conslidation[4] or at Matthew Aslett's contribution[5], neither of which are by Tarus but which begin by examining the fringes of the argument).

The nature of discussion on the Internet between different communities is one of reaction-to-blog-entries, it seems. While communications within a single community are handled by mailing lists and forums, communication across community boundaries is mostly via public journals. As such it can be difficult to follow the discussions, and many red herrings can show up. At the tail end of the discussion (the first of the two blogs mentioned above), Aaron Seigo points out why fiduciary license agreements (FLA) can be good and useful, both to the community, organisations, individuals and corporations. The value of an FLA for a software project -- for the community, the organisation, individuals and corporations -- comes from the clarity it provides for long-term stability of the project. An FLA can be used to express the relationship between community contributors and the organisation or business that manages and maintains the project. That same FLA brings risks, though, since consolidating copyright in one place also provides a single point of failure.

2 Four found guilty in landmark Pirate Bay case, http://edition.cnn.com/2009/TECH/04/17/sweden.piracy.jail/index.html
3 Open Source Business Conference Retrospective, http://www.opensource.org/node/416
4 http://aseigo.blogspot.com/2009/04/on-copyright-assignment.html
5 http://blogs.the451group.com/opensource/2009/02/02/define-open-source-vendor/

The Free Software Foundation Europe and its Freedom Task Force have produced a model FLA[6] for use by Free Software projects. An early adopter was the Bacula project.[7] The KDE project, which produces the K Desktop Environment software (11 years, 6 million lines of code, 600 developers) has recently adopted its own version of that FLA.[8] Aaron's blog post reminds us why the KDE project recommends an FLA, but does not require one. Current efforts related to improving and extending the FLA include translation (on the FSFE's part) and standardised wording (for developers to fill in on their forms).

For hobbyist developers, there is little interest in "doing it right" from a legal perspective. However, for long-term viability of free software projects, getting it right is something that needs to happen -- at least once. There is a great need for education, outreach and materials to enable free software projects to do the right thing by the law. Given the right tools, projects can engage with legal advisors to secure their own long-term viability.

## About the author

***Adriaan de Groot*** *is a researcher in software quality and verification; his free software efforts are confined to the KDE project, where he has the title Vice-President of Legal Affairs. His byline for developers is "I talk to lawyers so you don't have to." In this column, he writes for lawyers about what is going on in the trenches of free software development.*

6 http://www.fsfe.org/projects/ftf/FLA.en.pdf
7 http://www.bacula.org/fr/dev-manual/Fiduciary_License_Agreement.html
8 http://ev.kde.org/resources/FLA-generic.pdf

# Platform: Collaboration among counsel

*by Karen F. Copenhaver.*[a]

*(a) Partner, Choate Hall & Stewart LLP*

**Abstract**

How do we advance the development of sound legal constructs for FOSS without waiting for the results of litigation? Through the same sorts of collaborative development models that brought us open source software. But, because there are legitimate concerns that cause lawyers to hesitate before jumping in to a discussion thread, the development of private places for conversations among lawyers is a positive development that will result in benefits for the entire community.

**Keywords**

Law; information technology; Free and Open Source Software; Lawyers; Ethics; FOSSBazaar; FUD; Patents; Copyrights; GPL; derivative work; intellectual property; compliance; jurisdictions

## Celebrating the Formation of a Community of Lawyers for the Advancement of Understanding of Free and Open Source Licensing and Business Models

It is unfortunate that so many people are first introduced to the concept of open source software and open, collaborative development models through the lens of licence agreements and legal concerns. The benefits of community development of key industry infrastructure loom so much larger and are so much more important than what should be a side bar discussion about how we build the right legal models to support these efforts. Nevertheless, lawyers have been given a very prominent place in the conversation and as a result, for many free and open source developers, lawyer jokes are no joke. There is a special place reserved in their hearts for their most profound disappointments that is dedicated to lawyers.

## Why are open source licences so challenging for lawyers?

Why don't lawyers get it? There is actually a long list of reasons why open source software licences present challenges for lawyers. And when we consider these challenges, it becomes clear why the lawyer's network promoted by FSFE has been so warmly embraced by so many lawyers around the globe.

### Intellectual Property Law

All matters relating to intellectual property are challenging because there is a long history of underlying public policy and evolving statutes, case law and industry practices that must be considered to put any IP issue into context. It is not uncommon for current industry practices to be

analysed using legal authorities that are decades if not centuries old – sometimes with surprising results. New applications of intellectual property rights are always subject to debate until a rich body of case law or other rulings is developed that provides both clarity and finality on issues of first impression. Of course, free and open source licences are often called "copyleft" licences precisely because these licences use copyright concepts for revolutionary purposes which challenge the efficacy of these historical precedents. Lawyers who litigate for their living love the use of "old wine in new bottles" as fodder for disputes that have good arguments on both sides and that have sufficient importance to justify the cost of litigating the claims all the way through to the issuance of a judge's decision on the merits. But lawyers who are charged with giving timely advice to decision makers are less enthusiastic about the benefits of future court room dramas.

**Edge Cases**

There are very few areas of the law where the boundaries are absolutely clear and firmly established. The law develops through incremental decisions which over time fix and move the boundary time and time again. With open source, however, we have spent many years in grueling arguments over imponderable questions on the meaning of edge cases even though there are many clear cases in the centre that would be more useful teaching tools. Although such lively debates are interesting, they have their downside. This constant bickering contributes to a common impression that these licences are too dangerous and too complex for commercial adoption.

Why do lawyers do this? Well, for one, it is in our code of ethics to push the limits of the law as far as it can go in order to make the best argument for our client. Thus lawyers naturally push the edges of the envelope. Second, it is the way we are taught in law school. Law students are grilled with questions until they reach the limit where the proposition they are arguing is no longer true or no longer works. This is the way lawyers make points and undermine opposing arguments in an adversarial process. Lawyers naturally spend a lot of time arguing edge cases. And, third, the edge cases are intellectually challenging and fun to think about!

But there are other reasons to dwell on the edges cases that are more specific to open source. Because copyleft open source licences often want to extend the reciprocal obligations as far as intellectual property rights can go, the licences drive the analysis to the edge cases. And questions about open source licences are often posed in a context in which the possible reach of the licence is <u>the</u> question – even if no one has ever asserted a claim that the licence should extend that far. The question prior to adoption is often: "How far under any possible interpretation of the licence could someone argue that the licence's obligations extend?" Even if the answer to 99% of the possible realistic scenarios is the subject of a fairly clear consensus, the analysis is still going to focus on the 1% which remains unclear until we have more litigation and more rulings that provide a better basis for analysis.

**The Law Relating to the Software Industry**

It is not just the complexity of intellectual property law that lends such uncertainty to the analysis of open source licences. There are many other legal issues relating to the non-open source software industry that are unsettled. Two decades after the industry adopted "shrinkwrap" licensing practices, courts were only beginning to address actual trade practices. The questions about whether an open source licence should be analysed as a contract or as a licence arise because of the lack of certainty around the requirements of contract formation that has plagued the industry for decades (and continues to be argued in many jurisdictions today.)

**Legal Analysis versus Community Consensus**

In a FOSSBazaar meeting we discussed the two step process through which most companies move toward embracing the use of open source software within their organizations. Many companies cross the first threshold with a rude awakening. One day they think their operations are open source free. They have little or no understanding of the business benefits of open source and are happy to be in a position to be able to avoid the entire discussion. The next day they realize that they are using open source, lots of it. With this discovery they enter into a period in which open source software is purely a problem to manage. During this phase, the focus is on the role of lawyers as the front-line of defence and risk avoidance. And these lawyers tend to focus on a strict legal analysis of the relevant documents based on how a judge in a court of law would interpret the licence permissions and obligations.

Many companies go on to cross a second threshold. This step is more like an "a ha!" moment than a rude awakening. Management stops seeing open source as a necessary evil and starts to embrace it as an opportunity to be exploited to their advantage. These companies move from prohibiting employee access to open source projects to encouraging active participation by their employees in projects on which the company relies. A company that has entered this phase moves the focal point of the compliance process out of the lawyer's office and into the product development process or data center. Lawyer's representing clients that have entered this phase are more focused on how the communities in which they are invested interpret the licences. These lawyers are tasked with discerning the community consensus and that community replaces the judge and jury as the most important arbiter. These lawyers discuss the licence obligations in terms of that community consensus – or what one of the most experienced lawyers in this are refers to as the "folk wisdom" – rather than in terms of a strict legal analysis.

But when lawyers posit the community consensus as legal analysis there is violent pushback, and for good reason. Lawyers who are considering how these issues will be decided by courts, have legitimate concerns that a community consensus as to the meaning of, for example, "derivative work" as used in a licence such as the GPL, might be given weight by a judge in a consideration of copyright law. Lawyers and developers operating on the basis of the technical community's assessment of where the sharing obligations should be triggered for practical reasons have to remember that an expansive interpretation of copyright law relating to software gives rights to proprietary software vendors as well as open source developers. It is important to draw a distinction between legal analysis and a practical, technical approach to appropriate rules in support of collaborative development.[1]

Not surprisingly, lawyers working in these different contexts find it very difficult to reach a consensus as the basis for their analysis is completely different. Unfortunately, the inevitable fireworks sometimes adds to the impression that these are dangerous documents indeed.

**Litigators versus Pragmatists; In-House versus Outside Counsel**

In-house attorneys play a very important role in any discussion of open source licences. Even if in-house counsel have a wide variety of perspectives based on where their employers are in the spectrum of open source adoption, they share a certain immediacy of purpose. They make

1 An example of a separation of the legal analysis from the technically based community consensus is the definition of Corresponding Source Code which is used in GPL 3.0. Using Corresponding Source Code to define the sharing obligation separates the technical discussion (what does a developer need in order to avail him or herself of the freedom to develop a derivative) from the use of the copyright term "derivative work" as the boundary of the sharing obligation.

decisions and set a course of action based on what will happen in the short term, long before litigation might provide a final determination of the parties' respective rights and obligations. Legal analysis of the licence is only one consideration. A company can be technically right, and still lose. In-house counsel are often providing advice and counsel to their employer client in terms of the impact of negative publicity on employees, shareholders, customers, partners and other important stakeholders. In-house counsel are also considering issues in terms of internal policies and processes that drive issue identification and resolution, consistency in interpretation and risk assessment, and must always keep in mind the preservation of systems of internal controls.

Outside counsel are often engaged to play a role similar to that of an in-house counsel, but they are more often expected to predict how a situation would play out if fully litigated. The issues are framed in terms of legal procedure and the issuance of a final ruling by a judge sitting in a court room reviewing case law and statutes as the basis for his or her decision. The same presenting issue can look very different through this lens.

Not surprisingly, lawyers in these different roles sometimes find it very difficult to see eye to eye. And also not surprisingly, the public discourse is more often focused on the litigation scenario. It does not require the discussion of internal private matters and has the potential for more drama. It also is a better way for lawyers to drum up business.

**Lawyers versus Engineers**

While all of these complexities and varying perspectives mean it is not surprising that the legal community has trouble speaking with a single voice, developers are not sympathetic. Although developers who participate in open source communities often appear to enjoy jousting with lawyers on points of copyright law, they continue to be frustrated by the legal communities' inability to provide a definitive answer to even the most basic questions. Development is delayed while lawyers talk. Development is disrupted with remediation tasks for lack of a definitive legal ruling. There are distinctions drawn where there are no meaningful functional differences, and situations are considered analogous where there are critical dissimilarities. Developers can describe the same operation or function to the lawyers in many different ways and get different answers. The lawyers sometimes appear completely at sea – desperate to leave an opening for every possible adverse argument to be made over and over again.

Developers are engineers and everything in their being drives them toward decision making and closure. Developers are familiar with the community consensus and are confident that common sense will prevail. They are far more comfortable looking to project committers like Linus Torvalds for guidance than to the legal community. They are comfortable with rolling interpretations of licences that reflect programming and industry developments. They assume those interpretations will be backward compatible with past interpretations, enabling everyone to continue to build on the work that has already been completed. They do not stop and wait for certainty.

Although the lawyers spend time discussing whether the licence is a contract, the developers see the licence as much more akin to a constitution than a contract. A constitution is a living document which comes to embody an evolving understanding. You cannot truly understand a constitution without a study of the historical contexts in which the precepts have been tested and new understandings forged. Developers view open source licences in much the same way.

Lawyers, on the other hand, who are focused on how the document might be interpreted by a judge

in a court of law, must look at the licence as a legal instrument. Lawyers must follow the rules of contract construction and look first for answers within the four corners of the document. The words, all of the words, must be given meaning. Interpretations that conflict with the actual words of the licence must be rejected. Lawyers must divide the document into those parts that are limitations on intellectual property and those that are contractual covenants. To the extent there is ambiguity in the document, lawyers must first identify the parties to the agreement and look for statements made by those parties and which might be considered parol evidence of the party's intent. Statements by others who are not a party to the agreement are relevant only if they can be shown to have been adopted by one or both of the parties or to evidence a community consensus as to the meaning of the document that is sufficiently universal that anyone using the document would be presumed to have intended to adopt that interpretation as trade usage or industry custom.

## Progress through Collaboration

So how do we move beyond the cacophony of voices and perspectives and end the FUD (Fear, Uncertainty and Doubt)? Through collaboration, of course. Education by and for attorneys is the key to supporting adoption of open source. Unfortunately, lawyers are closed source by nature. Why? Why don't lawyers participate in "licensediscuss" and other open discussions regarding licence interpretation and copyright law? Developers often assume it is either ignorance or stinginess that keeps us away, but there is a long list of legitimate concerns that make lawyers hesitate before jumping in to a discussion thread.

### Ethical Limitations

Lawyers have special obligations to their clients. For example, they cannot take on work that conflicts with the client's interests without the client's permission. In order to fulfil this obligation, lawyers have to know exactly who their clients are at all times.

Of course, the rules are much more complex than this, but, in essence, a client relationship is formed when an individual believes that an attorney is providing legal advice to them. In other words, client relationships are determined by the reasonable expectations of the client and lawyers have an obligation to manage the formation of client relationships so that individuals are not misled. This obligation is very difficult to fulfil in the midst of a discussion through email chat.

### Context

One of the lessons that I learned my very first day practising law is that practising law is 99% facts and 1% law. Lawyers give advice based on a very specific set of facts and they test those facts through questioning and review of documentation to make sure that the facts that they have been given are accurate. A slightly different set of facts might result in completely different recommendations from the same lawyer.

A statement made by a lawyer on an email list that is based on a very specific question can be taken out of context and applied to a different set of facts which would have elicited a very different response from the lawyer. More often, the email discussions are based on a very limited statement of facts. It often happens that in a lengthy discussion it emerges that contributors to the discussion have made very different assumptions based on the sketchy information originally provided. When more details are provided, a very different discussion ensues.

**Cutting and Pasting**

Similarly, the lawyer does not want his or her statements quoted in a different context to which they may not be applicable or accurate.

**Timing**

As we have said, the law is still developing. Statements made today will be rendered obsolete by a case handed down tomorrow. And yet the posted comments will continue to be available unless the lawyer takes affirmative action to have them deleted or altered. Even with affirmative steps to remove or update the statements on the archive, the comments will survive in every forwarded copy.

**Tone and Balance**

Lawyers who want to see legal developments which are supportive of open source projects and licences hesitate to participate in discussions that focus on risk and the arguments that might be made in support of alternative positions. Although a good lawyer always fully considers the arguments that can be made for or against any point, even articulating the "dark side" in print to balance a discussion may permanently link the lawyer to that point of view. On the other hand, only stating the preferred analysis is not a complete consideration of an issue.

**Liability**

Individuals who rely on a lawyer's advice, even those who lurk on websites and have never identified themselves on a discussion list, may take action based on the statements made by the lawyer and may assert a claim against the lawyer if statements on which they relied cause damage.

For all of these reasons, despite the acknowledged benefits of "open," providing lawyers with a private space, limited to lawyers, in which to discuss these issues, to learn from each other and to arrive at common understandings, is absolutely essential to developing the necessary legal ecosystem around free and open source software. The FSFE[2], Open Bar and the Linux Foundation have all established and fostered such communities. The benefits of these legal communities will be seen in public communications that are more thoughtful and in less disruption and more confidence for the user community. We need a collaborative development process from which best practices can emerge. We need dialogue to form an articulated consensus. We need thoughtful explanation to understand why we have reached different conclusions and how those differences might be bridged.[3]

Although some might despair that this process has taken a long time and continues to move more slowly than anyone would like, there are clear benefits to moving slowly. However frustrated lawyers have been with the dearth of clarifying case law interpreting open source licences, the absence of litigation has been good for open source adoption. There is the obvious implication that

2 Through its legal arm, the Freedom Task Force

3 The most difficult challenge with respect to achieving clarity in the interpretation of open source licences is the impact of local laws. For fairness, the licences should have a similar interpretation across the globe. Actors in one jurisdiction should not benefit from a more advantageous interpretation. Laws in a single jurisdiction should not interfere with the reasonable expectations of those who contribute to development projects under a licence that accurately expresses their goals for the project under commonly held community understandings. But the same process as happens within a single jurisdiction can to some extent also occur across jurisdictions. As a common consensus emerges, and as open source developed infrastructure provides the backbone for local industries, courts will struggle within permissible boundaries to preserve expectations and to avoid disruptions that could be damaging to local economic activity.

the absence of litigation is evidence that companies should not have undue fear of open source adoption. But there is a much more important benefit that has come from keeping these matters out of the courts – the decisions, if and when they do come, will benefit from the long community conversation and the judges will reach different conclusions than they probably would have had the litigation commenced early in the development of open source licensing models. The recent *Jacobsen* case in the U.S. is an example of the kind of balanced decision making that is possible when not only the issues in the particular case are ripe for resolution but the case can be presented through *amicus* briefs in the context of its impact on a set of established industry expectations.

## Conclusion

I was asked by a member of the audience at a conference once what I thought about all of the discussion on the "licensediscuss" email list for the Open Source Initiative that began "IANAL but" (I Am Not A Lawyer). The questioner clearly anticipated a lecture on how dangerous and potentially misleading these discussions of legal issues by non-layers can be. But my immediate and heartfelt reaction was to be grateful for the non-lawyers who had shown us the way. I responded, "If the lawyers had been all over this from the very beginning, we would have killed it." No one should confuse the "licensediscuss" posts with legal advice, but we should not ignore them either. These are the discussions out of which a community consensus has begun to form and through which many people, lawyers included, began to understand the value of collaborative development and the community fundamentals that are essential to its successful implementation. Let the conversations continue! And let the lawyers heed the call to collaborate in the development of legal models to support collaborative development and free and open source software.

## About the author

***Karen Copenhaver****, a partner at Choate, Hall & Stewart LLP and counsels business clients in the drafting and negotiation of strategic alliances,technology transfer and licensing of intellectual property, particularly in the areas of patent licensing, software licensing and open source business models. She is engaged by the Linux Foundation to serve as Director of Intellectual Property Strategy, is listed in Chambers USA,Legal 500, Massachusetts Super Lawyers, and has been chosen by Intellectual Asset Management (IAM) magazine as one of the world's top IP strategists, and is featured in the June 2009 "IAM 250 - A Guide to the World's Leading IP Strategists." Karen is only the 5th lawyer ever to receive Mass High Tech's prestigious "Mass High Tech All-Stars Award," which honours the thought leaders and innovators throughout theNew England technology sector.*

There are many reasons why The International Free and Open Source Software Law Review will be warmly received in legal circles, in the free and open source community and in the wider software industry. For many years the focus in the legal community has been on raising questions about free and open source software licenses and development models. With this journal we have turned the page and begun to focus on the answers. It is rewarding to see lawyers adopting collaborative models to share knowledge and work product, arrive at common understandings, and further the development of the necessary legal ecosystem around free and open source software. It is also significant that this international journal reflects the global community that has formed around these models. My congratulations and thanks go to the editorial board that worked so well together to create this valuable resource.

*Karen Copenhaver,*
*Partner Choate Hall & Stewart LLP*
*and counsel to the Linux Foundation*

The intersection of open source and the law yields unique opportunities for new insights to be drawn and leadership to be exhibited as we move inexorably toward an era of unrivaled collaborative development. It is only fitting that this journal focus on Legal issues in open source that relate to patents and beyond as we need a forum to discuss and debate critical legal issues so that the full benefit of open source as a modality of invention and innovation can be realized.

*Keith Bergelt*
*Chief executive officer of Open Invention Network (OIN)*

Thanks to our Sponsors

IFOSS L. Rev. is published by its Editorial Committee. The Committee gratefully acknowledges the sponsorship of its many supporters and donors, including:

nlnet FOUNDATION

mozilla FOUNDATION