Problems in Open Source Licensing[1]

By Jeremy Malcolm [2]

Introduction

It is all too easy to assume that open source software licences[3] are valid and enforceable. Many of them have been written by lawyers, and even some of those that haven't, seem to have stood the test of time pretty well. But the truth is that to assume that a software licence is not only valid, but valid throughout the world, and safe from being revoked, is not merely optimistic but certainly wrong.

In this paper I intend to point out some of the little-known problems associated with the use of open source software licences, at least in the context of the Australian legal system. However it is not all bad news, as I will also discuss some of the strengths of the common open source software licenses. I will conclude with some suggestions on how we can ensure that the open source licenses we use will hold water in the event that they are legally challenged.

What isn't a software licence?

A software licence is not necessarily a contract. It can be, but that requires a couple of preconditions to be satisfied.[4] One of those preconditions is the existence of consideration on both sides. Consideration is a legal concept that simply means a quid-pro-quo, or something of value given by each party in exchange for what the other party provides. In the case of open source software, there usually isn't anything provided by the licensee of the software (that is, the person who uses it) back to the licensor (usually, the person who wrote it). As a consequence of this lack of consideration there is no contract between the licensee and licensor.

There may be other preconditions of the existence of a contract that are also missing. One is that the contract must be accepted by the licensee, and that acceptance must be manifested in some way. This doesn't necessarily mean signing on the dotted line, but it it does require that you have been given an adequate opportunity to accept or reject the terms of the contract, and that you have made a conscious and visible decision to accept.

We believe for example that clicking on an "I agree" icon can be sufficient to indicate assent to a contract; these are known as "click-wrap" agreements, and at least in one case in America the courts have upheld this form of agreement.[5] On the other hand

[1] This is a paper to be presented at Australia's national Linux conference, Linux.Conf.Au in January 2003.

[2] Jeremy Malcolm is an Information Technology lawyer with a successful niche practice in Internet-related law, and is involved at board level in a number of relevant organisations such as the Society of Linux Professionals of WA, the Internet Society of Australia, the Western Australian Internet Association, the Australian Public Access Network Association (as Secretary), the WA Society for Computers and the Law (as President) and previously Electronic Frontiers Australia. He has been since 1998 the Manager of Terminus Network Services which specialises in the use of open source software in networked environments and in the development of online systems; he has also established the http://www.linuxconsultants.com.au portal for Linux consultants, and is a developer of the Debian operating system.

[3] For the purposes of this paper, I will take an open source software licence to be one that satisfies the Open Source Definition (http://www.opensource.org/docs/definition.php).

[4] Actually, five: legal capacity, intent to enter contractual relations, an offer, an acceptance of that offer, and consideration to support it.

[5] *Hotmail Corp v Van Money Pie, Inc* (1998) 47 U.S.P.Q. 2D (BNA) 1020, 1025 (N D Cal).

simply receiving a set of licence conditions in the documentation of software isn't necessarily enough to create a contract between the licensor and licensee. Later, I will be discussing an Australian case which suggests that sufficiently prominent notices in the documentation are enough to indicate the licensee's agreement to the software terms, but it is fair to say that there are relevant decisions going both ways[6] and that the law on that question therefore remains unsettled.

What is a software licence?

Even if we don't have a contract, either due to lack of consideration or lack of acceptance, we do still have copyright. Copyright automatically exists in computer software, and it lasts for 50 years[7] (or in the USA 70 years[8]) after the death of the author. It gives us the exclusive right to control who makes copies of our software and who creates modified versions of it. If we wanted to, we could use copyright to stop anyone from receiving a copy of our software at all.

As it happens, we are much more generous than that, and we want to encourage people to copy and modify our software. But we are allowed to put conditions on our generosity. These are the same kind of conditions that the owner of land can impose on the right of visitors to their land, for example "if you enter this shop we can check your bags", or "persons entering this building site do so at their own risk". These are not contractual conditions, they are licence conditions.

A licence does not grant you a legal right to the property that is being licensed to you. All it does is to make something lawful that would otherwise be unlawful.[9] Therefore, unlike in the case of a contract, the conditions that are placed on a licence can't give the licensor any more rights or powers than he already has. All they can do is place limits or conditions on the licensee's entitlement to exercise the rights that copyright law grants exclusively to the licensor.

The Free Software Foundation's lawyer, Eben Moglen, has described the distinction in this way:

> [M]ost proprietary software companies want more power than copyright alone gives them. These companies say their software is "licensed" to consumers, but the license contains obligations that copyright law knows nothing about. Software you're not allowed to understand, for example, often requires you to agree not to decompile it. Copyright law doesn't prohibit decompilation, the prohibition is just a contract term you agree to as a condition of getting the software when you buy the product under shrink wrap in a store, or accept a "clickwrap license" on line. Copyright is just leverage for taking even more away from users.[10]

So what kind of licence conditions can we legally impose on the use of open source software, in the absence of a contract? In general, we can impose conditions that restrict the right to copy the software, because this is one of the exclusive rights that copyright grants to the licensor. We can also impose conditions that restrict the licensee's ability to modify the software. But we probably cannot restrict their right to

[6] Particularly in the United States: compare *Step-Saver Data Systems, Inc v Wyse Technology* (1991) 939 F 2d 91 (3d Cir) holding "shrink-wrap" software licences unenforceable, with *ProCD, Inc v Zeidenberg* (1996) 86 F 3d 1447 (7th Cir), holding them enforceable.

[7] Copyright Act 1968 (Cwlth) s.33.

[8] 17 USC 302.

[9] *Thomas v Sorrell* (1673) Vaugh 330.

[10] Moglen, E. "Free Software Matters: Enforcing the GPL, 1" Linux User, August 2001.

run the software,[11] nor require them to destroy their copy of the software if the licence is revoked, or require them to allow the licensor into their premises to perform a software audit, because these are not rights that the licensor possesses under copyright law.

It is not quite as simple as that though, because a clever licence agreement can be drafted to make almost any condition into a restriction on the the right to copy ("by copying this software for your own use you must allow us to monitor your computer for license infringements"). At what point does an obligation of the licensee cease to be enforceable as a licence condition, and become an undertaking which requires a contract in order to be enforceable? The answer, such as it is, is that it depends on what a court of law is willing to enforce.

Because there is no enforceable contract, if the licence conditions are breached, the licensor's action against the licensee is simply an action for breach of copyright. A court hearing such an action is entitled to refuse to enforce a licence condition (or to award damages for its breach) if it goes too far outside the scope of copyright law and into the realm of private ordering. The only guide that we have is that in order to be enforceable, a licence condition must be "reasonable".[12] Given the dearth of authority on what amount to "reasonable" software licence conditions, the enforceability of a given condition can only be determined on a case-by-case basis.

In light of these principles, let us briefly have a look at some of the common open source licences to see what sort of conditions they impose and to make an educated assessment about whether they would be enforceable as licence conditions, rather than as contractual conditions.

Common licences

GNU General Public License

The preamble to the GNU General Public License[13] (GPL)[14] makes much of the fact that "licenses for most software are designed to take away your freedom" whereas the GPL "is intended to guarantee your freedom to share and change free software". It is true that a licensor under the GPL does gives up many of its exclusive rights to control the copying and modification of the software, in that it allows the licensee:

- To distribute verbatim copies;[15]
- To modify copies or parts of them;[16] and
- To copy and distribute modified copies of the program.[17]

[11] Section 47B(2) of the Copyright Act 1968, added in 2000, appears to have that effect. However the provision is merely a qualification to s.47B(1) which has been held to grant no new substantive rights to the copyright owner (see *Kabushiki Kaisha Sony Computer Entertainment v Stevens* [2002] FCA 906). The correct interpretation of s.47B(2) is therefore that it simply preserves the effect of licences that restrict the right to run software and that are independently enforceable, for example contractually.

[12] See *Robinson v Balmain New Ferry Co. Ltd* [1910] AC 295, in which the right to enter a jetty was conditioned on the payment of one penny before being allowed to exit.

[13] I will give "licence" its British English spelling except when referring by name to a licence that uses the American spelling in its title.

[14] http://www.fsf.org/licenses/gpl.html.

[15] Clause 1.

[16] Clause 2.

[17] Clause 3.

However, the GPL also imposes certain significant limits on the rights of copying and distribution of the software that it grants, namely:

- a copy of the licence must be distributed with the software;[18]
- interactive programs must display a notice that describes the licensing terms;[19]
- a copy of the source code must be included or otherwise made available to the licensee;[20] and
- distribution of the software is not permitted except on the terms that the GPL provides.[21]

The GPL also regulates modification (also known as the creation of "derivative works") of the software; another exclusive right of the copyright owner:

- any modifications must be documented;[22] and
- derivative works must be licensed under the GPL.[23]

Are these conditions enforceable? Although an authoritative answer cannot be given until the GPL is tested in court, all indications are that these are exactly the kind of conditions that can be successfully attached to a non-contractual software licence, since they only affect the distribution and modification of the software, which are within the right of the copyright owner to control. They do not purport to impose any conditions on a person who simply acquires the software for their own use, which many other software licences including shareware licences do try to do.

The most controversial clause of the GPL is probably clause 2(b) which gives it its "viral" quality, in that it requires any works derived from the software also to be distributed under the GPL.[24] Legally however there is nothing exceptional about that condition, given that most software licences prohibit derivative works from being produced at all.

Perhaps more questionable legally are the restrictions on the user's rights to sue for breach of implied warranties[25] and tortious claims[26] (for example, claims for damages arising out of loss of data caused by the programmer's negligence). These have nothing to do with the licensor's exclusive rights under copyright law, and therefore in the absence of a binding contract these conditions are unlikely to be enforceable as terms of the licence.

However, this does not mean that they are completely ineffectual. The law has a saying that there can be no injury done to someone who consents,[27] and that is the basis for the effect of these clauses. Whilst such disclaimers are not always effective,[28] the licensor of proprietary software[29] is in no better position than his or her

[18] Clause 1.

[19] Clause 2(c).

[20] Clause 3.

[21] Clause 4.

[22] Clause 2(a).

[23] Clause 2(b).

[24] A licence imposing such a requirement on derivative works is also known as a "copyleft" licence.

[25] Clause 11: such terms are implied into contracts by division 2 of the Trade Practices Act 1974 (Cwlth) and by the various State Sale of Goods Acts and Fair Trading Acts, however these are not applicable in the case of a non-contractual licence.

[26] Clause 12.

[27] More commonly, *volenti non fit injuria* : *Insurance Commissioner v Joyce* (1948) 77 CLR 39.

[28] See eg. *Council of the City of Sydney v West* (1965) 114 CLR 481.

[29] I use the term "proprietary" in its colloquial sense, rather than to imply that open source software is

counterpart in the open source community. The proprietary software licensor may in fact be in a more difficult position by reason that it is reasonable for the licensee to have higher expectations; they expect to get what they are paying for.

Overall, then, the provisions of the GPL receive a qualified tick of approval under Australian law. Because the GPL does not attempt to interfere with matters unconnected with the licensor's exclusive rights under copyright law, its provisions are likely to be just as enforceable as the provisions of a contractual licence for proprietary software.

GNU Lesser General Public License

I will spend a much shorter time in dealing with the other open source licences. The Lesser (formerly Library) General Public License (LGPL)[30] can be distinguished from the GPL in two relevant respects. Firstly, it specifies that it is only to be used for software libraries, although in fact it is commonly used for other software also.[31] Second, it allows the software to be linked with proprietary code, which the GPL does not. Essentially, the LGPL can be regarded as a non-viral form of the GPL that is intended for software libraries only.

These two distinctions from the GPL don't have any impact upon the enforceability of the licence. That is not to say that there aren't some traps in using the LGPL; the distinction between a derivative work which has to be distributed under the LGPL, and a work that just uses the library which doesn't have to, can be slippery. Another tricky issue is exactly what you can distribute together and what you can't.

However, to go any further into those issues would take me beyond the scope of this paper. Suffice it to say, the LGPL should be just as enforceable as the GPL is.

The BSD License

The BSD licence[32] is the shortest and simplest of the common licences, at only 225 words compared to the GPL's 2968 and the LGPL's 4380. Its simplicity makes it the more likely to be enforceable as a whole than either of those other licences. It allows the software to be used and modified freely, subject only to the retention of attribution of the work of previous contributors.

The main effective difference between the BSD licence and the GPL is that derivative works are not required to be licensed freely, but can be relicensed under a proprietary licence. Unlike the LGPL, this doesn't just apply to programs that are linked with the original code.

The original BSD licence[33] also contained a clause which required the code to contain an acknowledgment sentence in any advertising materials, whereas the revised BSD licence that was released in 1999 does not. Some other simple open source licences in a similar vein to the BSD licence include the Apache licence[34] (which also contained an advertising clause in its original version[35]), and the MIT X11 licence.[36]

any less proprietary (that is, a form of property) than contractually licensed software.

[30] http://www.fsf.org/licenses/lgpl.html.

[31] For example, OpenOffice.org - http://www.openoffice.org.

[32] http://www.xfree86.org/3.3.6/COPYRIGHT2.html#5.

[33] http://www.xfree86.org/3.3.6/COPYRIGHT2.html#6.

[34] http://www.apache.org/LICENSE-1.1.

[35] http://www.apache.org/LICENSE-1.0.

[36] http://www.x.org/Downloads_terms.htm.

The Mozilla Public License

The Mozilla Public License (MPL)[37] is a much more comprehensive contractual-style licence document, which like the GPL requires derivative works to be freely licensed (although not necessarily under the MPL[38]). Its cousin the Netscape Public License (NPL)[39] is based on the MPL but grants the original software creator, namely Netscape in this instance, rights to use the software together with any derivative works in its own proprietary software.[40]

The MPL contains some very good clauses dealing with the possible existence of third party intellectual property claims or patent rights over contributed code,[41] and clauses detailing how the contributions of developers are to be documented.[42] There is also a clause[43] which states that if your licence to use the software is terminated because you have breached its terms, that does not affect any sublicences that you may have granted to other people.[44]

Once again, we have encountered nothing that takes these licences outside the scope of what can be legally enforced in the absence of a contract.

The Artistic License

Software licensed under the Artistic License[45] may be freely distributed and modified, provided that any modifications are documented and in some cases[46] made freely available. Like the MPL but unlike the GPL, the Artistic License does not require derivative works to be licensed under the same terms as the original.

The Artistic License in its most widely-used version suffers from loose wording, however version 2.0 of the licence, which will cover Perl version 6 when released, addresses these concerns. Even in its earlier form, it is unlikely that an Australian court would refuse to give effect to any part of this licence.

So having reviewed a selection of the best-known open source software licences, it appears that the prognosis is good. None of them contain any provisions that I have identified as being unenforceable under Australian law. For such a revolutionary mode of licensing as open source is, it is gratifying and perhaps a little surprising that it appears to be so compatible with traditional copyright law.

In fact the further we stray away from the Open Source Definition[47], the more problems with enforceability we will find. The Microsoft Shared Source family of licences are a classic example, in that they attempt to restrict the commercial use of the software by the licensee. As noted earlier, copyright law has nothing to say about

[37] http://www.mozilla.org/MPL/MPL-1.1.html.

[38] Clause 6.3.

[39] http://www.mozilla.org/NPL/NPL-1.1.html.

[40] Clause V.

[41] Clauses 2.1, 2.2, 3.4.

[42] Clauses 3.5, 3.6.

[43] Clause 8.1.

[44] The GPL attacks this problem in a different way, by asserting that the "sublicence" is made not between the distributor and the user to whom it is distributed, but rather between the original copyright owner and the end user: clause 6.

[45] http://www.perl.com/language/misc/Artistic.html.

[46] But not others. The software can be incorporated into a proprietary product so long as it is hidden from the end user, and modified binaries can be distributed without source so long as they are accompanied by the unmodified files (or instructions on where to obtain them) and documentation of the modifications.

[47] http://www.opensource.org/docs/definition.php.

the use of software, only such acts as its copying and modification.[48] Consequently, unless you have a contract with Microsoft for the provision of shared source software, the parts of that licence that purport to prohibit you from making commercial use of the software are almost certainly unenforceable.

Why contracts are better than licences

Relicensing

So far, so good; each of the open source licences we have examined appears to be legally sound. But what happens if the licensor of an open source software program wishes to change the conditions under which it is licensed? It is in this circumstance that we encounter the biggest deficiency of licence conditions as against contractual conditions. Because the licensee hasn't given any consideration in exchange for the software, the licence can be revoked by the licensor at any time simply by giving notice to the licensee.[49]

In the context of software licensing, this means that there is nothing that can be done to stop the licensor from changing the licence conditions, including makinq them non-free or withdrawing the software altogether. It doesn't matter if an open source licence claims to be irrevocable. Because the licence hasn't been paid for, it isn't.

The result of this is that in a worst case scenario, a copyright owner could release a program under an open source licence, distribute it to 10,000 users and then change the licensing conditions to require payment of an annual license fee. If after a reasonable time any licensees are still using the software in breach of the license conditions, they are liable to be sued for copyright infringement.[50]

This is one of the best kept secrets of the open source movement, and unsurprisingly it is not consistent with the public position of the Free Software Foundation (FSF).[51] Richard M Stallman (RMS) has posed and answered this Frequently Asked Question on the FSF's Web site:

> Can the developer of a program who distributed it under the GPL later license it to another party for exclusive use?
>
> No, because the public already has the right to use the program under the GPL, and this right cannot be withdrawn.[52]

There isn't any legal authority cited in support of that proposition. In fact, it seems to contradict clause 9 of the GPL which explicitly acknowledges that GPL software can be relicensed when a new version of the GPL is released. So although RMS may be correctly stating the law in some States of America,[53] he is without doubt incorrect as to the position in Australia.

It gets worse

As if that isn't bad enough, there are a few scary, but less obvious corollaries of the

[48] See also Liberman, M. "Comment, Overreaching Provisions in Software License Agreements", (1995) 1 Rich J L & Tech 4, http://www.richmond.edu/jolt/v1i1/liberman.html.

[49] *Wood v Leadbitter* (1845) 13 M & W 838.

[50] *Computermate Products (Aust) Pty Ltd v Ozi-Soft Pty Ltd* (1988) 20 FCR 46 at 49.

[51] Or others, see eg. http://www.openoffice.org/FAQs/mostfaqs.html#3.

[52] http://www.gnu.org/licenses/gpl-faq.html

[53] The effect of the Uniform Computer Information Transactions Act (UCITA) will be to give the force of a contract to a (possibly non-contractual) shrinkwrap or clickwrap software licence. However this Act is not in force in most States.

revocability of open source licences:

1. It seems to have been presumed by many people that if software is relicensed then that only applies to future versions of the software; it can't retrospectively effect versions that were released under the original license. This is a false assumption. If for example the original developer of SSH wanted to relicense all existing versions of the software developed by him, including the GPL version that OpenSSH is based on, that could legally be done. We are fortunate that it hasn't been done in any high-profile project so far. The prospect that it could happen to something like the Linux kernel doesn't bear thinking about.
2. Even if we would trust the author of software not to revoke its open source licence, all bets are off if the author assigns the copyright to someone else (that is, sells it, gives it away, or passes it by will). In a case from two years ago that many of you will remember, two young programmers, Matthew Skala and Eddie Jansson, released a program called cphack which was designed to decrypt the database of blocked URLs embedded in Mattel's proprietary Web-filtering software, CyberPatrol. Mattel induced the programmers to assign their copyright in the software over to it. However Wired magazine reported the following day that cphack had already been released under the GPL, with two possible outcomes: either that the GPL prevailed over the assignment of copyright to Mattel, or that that licence could be revoked by Mattel.[54] It turned out that this was a false issue because the software actually wasn't released under the GPL at all. But if it had been, Mattel could certainly have revoked the licence,[55] unless the assignment of copyright left the original authors with a residual interest in the software that could continue to support the licence.
3. In the case of a fork in the development of software, both branches of the fork inherit the licence conditions of the root package. This does not necessarily mean that the forked packages cannot be relicensed if the root package allows that, but what it does mean is that those sub-licenses remain subject to the continuance of the head licence. As soon as they become inconsistent with the head licence, they will be invalid to the extent of the inconsistency. The legal principle that lies behind this is that you cannot give what you do not have.[56] In other words if you do not have an irrevocable licence yourself, you cannot grant an irrevocable licence to anyone else. The most you can do is to grant a licence that is subject to being withdrawn or modified at any time.

Fear, Uncertainty and Doubt?

Do we really need to worry about the scenarios I have described above? Yes unfortunately we do, but there are three mitigating factors that might give us some comfort.

Multiple licensors

Often there is more than one copyright holder for an open source program. This can either take the form of joint ownership of the copyright (where several authors share copyright over the work as a whole), or a set of individual ownerships of copyright

[54] McCullagh, D. "Wired News: Mattel Ruling Confuses Hackers", http://www.wired.com/news/business/0,1367,35258,00.html.

[55] In fact, it would arguably have been revoked automatically when the copyright was transferred, unless the GPL licence had been signed by the licensor: 71 USC 205(e). This is not the case in Australia: Copyright Act 1968 (Cwlth) s.196(4).

[56] More commonly, *nemo dat quod non habet* : *Cundy v Lindsay* (1878) 3 AC 459.

over particular files, patches or forked derivative works. Taking first the case of joint ownership, it would be necessary for all of the copyright owners to agree to withdraw an open source licence. This is not so much of a legal limitation on the relicensing of open source software as a practical one, but it does provide an extra measure of protection for users in comparison to the position of a single copyright owner who can unilaterally relicence the software.

On the other hand in the case where there are multiple separate copyrights contained in software, the news is not so good. In that case, any individual copyright owner can change the licensing conditions of what they have contributed, which may result either in the package as a whole becoming non-free, or in that contributed code having to be excised from the software. This is similar to what happened when the University of California Berkeley was required to excise the AT&T code from the 4.4BSD distribution.[57]

Estoppel

There may be scope to argue that a software author is precluded from changing the licence conditions on software if the licensor misled the licensee into relying on the continuance of the existing terms in the knowledge that the licensee would be detrimentally affected by that reliance if the terms were changed.

The argument would be based on a legal doctrine called estoppel, which can prohibit you from exercisinq your legal rights - in this case, the right to change the licence conditions - if you have allowed someone else to rely on the fact that you wouldn't exercise those rights. In light of the rhetoric that is often bandied about about to the effect of "Once GPL, always GPL", circumstances could well exist in which this argument would be quite persuasive.

The question is, can estoppel restrict a licensor from changing their licence conditions to the user community as a whole, or only one particular licensee? Probably only the latter, since it is generally impossible to demonstrate reliance, which is one of the elements of estoppel, except on a case-by-case basis. [58] This reduces the usefulness of the estoppel argument to combat the withdrawal of open source licensing of a package that is in widespread use. At best, one open source licensee's victory will cause the licensor to reconsider enforcing the new license terms against other former licensees.

Another limitation of the estoppel argument is that a court would not necessarily compensate a licensee who had been misled into relying on the continuity of an open source licence by requiring the licensor to reinstate that license; it would be equally open for the court simply to order the licensor to pay damages. Those damages might be negligible unless the licensee has significant business interests riding on the use of the package.

Consideration revisited

Perhaps we can argue that there is some consideration on both sides of an open source licence agreement after all, and that the agreement is therefore enforceable as a contract. This would allow the agreement to be made irrevocable.

Take the example of the GPL, for example. The licensee of GPL software does appear to be giving something in exchange for the use of the software. Firstly, the licensee

[57] See McKusick, M. "20 Years of Berkeley Unix: From AT&T-Owned to Freely Redistributable", http://www.oreillynet.com/pub/a/network/2000/03/17/bsd.html?page=1.

[58] Furthermore, although it may be possible for a class action to be instituted, this is likely only to be feasible for the licensees who are resident in the jurisdiction where the action is commenced.

makes a number of promises that I have described earlier, for example, to make the source code available at no charge when distributing the software, and to document any modifications. Second, the licensee is required to accept all risks involved in the use of the software. Surely these are worth something?

Unfortunately as far as the law is concerned, they are not. The GPL explicitly states, "Activities other than copying, distribution and modification are not covered by this License; they are outside its scope."[59] Therefore there is no requirement on licensees to distribute the software at all, and if they don't, they are not agreeing to anything and not providing any consideration. The law takes the position that consideration isn't acceptable if the licensee has a choice about whether or not to provide it.[60]

As for the licensee's acceptance of liability for faults in the software, this doesn't amount to sufficient consideration either, because there may not be any faults, or if there are, the licensee might not suffer any loss from them, or none that can be blamed on the licensor. It is only if the licensee actually has an existing right to sue the licensor, or at least honestly believes that he does, that an agreement not to sue can amount to good consideration.[61]

The law in Australia

Apart from the general principles that I have outlined, there is no law in Australia specifically concerning free software licences. The closest that we have is a case on shareware licences; *Trumpet Software Pty Ltd v OzEmail Pty Ltd* .[62] In this Federal Court case, the author of Trumpet Winsock sued what was then Australia's largest commercial Internet Service Provider, OzEmail, for breaching the shareware licence of Trumpet Winsock by distributing the program to its users after its licence to do so had been revoked, and in any case, in breach of what would have been the terms of its licence to do so.[63] The breach would have been committed by OzEmail's modification of the software's configuration files without documenting those modifications, and its failure to distribute some of the documentation. This type of breach is especially relevant for present purposes, because it would be a breach of the GPL also.

The dispute came about when OzEmail wished to distribute copies of the unregistered version of Trumpet Winsock on the front cover of Australian Personal Computer magazine. The managing director of OzEmail contacted the author Peter Tattam and requested his permission to include the program on the cover disk, but permission was refused on the ground that the version that OzEmail sought to distribute did not come with a timelock to disable it after 30 days. Tattam did give permission to distribute a time-locked version of the software, but as the time-locked version was not available by press time, OzEmail distributed the unlocked version instead. OzEmail took the view that regardless of Tattam's specific refusal of permission for it to distribute the unlocked version, the shareware licence itself prevailed over what Tattam had said, and it allowed the distribution to go ahead.

The Court found that the opposite was the case. Justice Heerey said:

> Determinative of this case in my opinion is the proposition that, prior to the

[59] Clause 0.

[60] *British Empire Films Pty Ltd v Oxford Theatres Pty Ltd* [1943] VLR 163.

[61] *Butler v Fairclough* (1917) 23 CLR 78.

[62] (1996) 34 IPR 481.

[63] There were in fact no express terms of the licence that dealt with the copying of the program by a distributor, so the court was required to imply terms into the licence, which it did on the same basis as if the licence had been a contract.

> distributions complained of, Mr Tattam expressly told OzEmail that he objected to OzEmail using Trumpet Winsock 2.0B and thus revoked any licence OzEmail might have had.[64]

It was found in the alternative that if OzEmail's licence to distribute the software had not been revoked, OzEmail had breached the terms of the licence by making the modifications to the software that it did.

So what does this tell us about open source software licences? The case contains good and bad implications. On the positive side, it tells us that the licence terms of a non-contractual software licence agreement can be enforced, including terms which prohibit the software from being distributed with modifications or deletions that the licensor has not decided to permit.

However, the case also confirms that a software licence that is granted without consideration can be revoked by the licensor at any time so long as notice of the revocation is conveyed to the licensee; this is not such good news for the open source software community.

Strategies

The ideal solution to the problems that have been identified would be the introduction of legislative reforms to provide a regime for the enforcement of open source licences by both licensors and licensees. Such an enforcement mechanism would allow the licensor to prevent licensees from infringing the terms of the licence, even if a commercial loss by the licensor could not be shown. But it would also allow the licensee to enforce a term of an open source licence that was intended to make it irrevocable, even if the licensee had provided no consideration for the licence.

In the absence of such legislative reforms, there is a simple step that can be taken towards improving certainty for open source licensees and licensors. First, copyright of open source software should be vested in a person or body who can be trusted not to withdraw or detrimentally alter its open source licence. This will require copyright of the source code in the original package, and the source code of any modifications, to be assigned to the trusted person or body by a written document signed by the licensor.

The Free Software Foundation encourages contributors to its projects either to assign copyright of their contributions to the Foundation, or to disclaim copyright in those contributions,[65] thereby placing them in the public domain.[66] Ironically, unlike a license, an assignment of copyright is enforceable without consideration, so long as it is made in writing.[67]

For software that is not GPL-licensed, the FSF does not accept assignments of copyright. For such software, it is best for the assignment of copyright to be made to the lead developer, as for example in the case of contributions to the OpenOffice.org project which are required to be jointly[68] assigned to Sun Microsystems.[69]

In addition to providing certainty as to the licensing terms of a software project, the

[64] *Ibid*. 499.

[65] For contributors who are employed or are students, the FSF also provides a template disclaimer of copyright for the employer or educational institution to sign.

[66] GNU Maintenance Instructions, http://www.gnu.org/prep/maintain_7.html#IDX2.

[67] Copyright Act 1968 (Cwlth) s.196(3), and in the United States 17 USC 201.

[68] In other words, the original copyright owner retains the copyright in joint ownership with Sun.

[69] http://www.openoffice.org/licenses/jca.pdf

other advantage of assigning copyright to a trusted body is that it can then assume responsibility for enforcement of the licence in court. If you are working on a project that derives code from a variety of sources all of which are separately licensed, it would be necessary for all of the licensors to join together in any court action, and in most cases this will be impractical.

Conclusion

It would be redundant to observe that open source licensing is here to stay, regardless of its recognition by the law. The legislative reforms that I have proposed would provide a greater level of certainty to both users and developers of open source software, but I am not naïve enough to suppose that there is much prospect of them being passed here, let alone also in the United States and the European Community.

We must therefore make the best of what we have. What we have is a development model that is capable of producing the best software in the world, and a community that is full of brilliant, enthusiastic and altruistic developers. We owe it to them to do whatever we can to ensure their work remains free. Giving careful consideration to the form of licence that you select for a software project, and to who the copyright owner should be, is a good first step to take, while we all wait for the law to catch up.