Copyright, Contract and FOSS

Luke McDonagh

Introduction

Over the course of this chapter three crucial aspects of the law's relationship with FOSS licenses are reviewed. Firstly, a comparison of the licenses themselves is outlined with particular regard to copyright provisions. In this respect, it is noted that while there is a great diversity of FOSS licenses, the licenses broadly fall into one of three categories – 'no copyleft', 'weak copyleft' and 'strong copyleft'. Secondly, the debate over enforcement is discussed, focusing on the question of whether these licenses typically operate as 'bare licenses' or whether they are in fact 'contracts'. This is an important issue because different legal consequences flow with regard to each category. Moreover, this is an issue which is difficult to resolve given the fact that FOSS typically operates online, across national boundaries, while different legal rules apply in various national jurisdictions. Thirdly, the compatibility of the most significant FOSS licenses is examined.[1]

[1] C. Thorne, 'Open Source Software – UK perspective,' Reynolds Porter Chamberlain LLP Presentation (2010); accessible at http://www.scl.org/bin_1/6.%20Clive%20Thorne.pdf - According to Thorne, the major GPL licenses together account for 65% of the total OSS license world. – GPL v 2 accounts for 50%, LGPL v 2 accounts for 10%, while GPL v 3 accounts for 5%. See also J. Lovejoy, 'Understanding the Three Most Common Open Source Licenses' Open Logic; accessible via download from http://www.openlogic.com/resources-library/webinar-understanding-the-most-common-oss-licenses/ – According to Lovejoy, by percentage of projects, 70% use GPL, while 7.6% use Apache and 6.7% use LGPL.

1.Comparative Analysis of Key Licenses – 'No Copyleft' v 'Weak Copyleft' v 'Strong Copyleft'

Original works of software are protected under copyright laws.[2] Several different copyrights can arise in this context. For an entirely new original work of authorship, the authors are the first owners of the copyright.[3] Furthermore, for a later work consisting of new modifications to existing code, a separate copyright will arise with respect to this new original material, which in the US is commonly referred to as a 'derivative work'.[4] There is also the possibility for a copyright in an aggregated 'compilation'.[5]

As noted above, there is copyright in works of software. However, the concept of 'copyleft' must also be briefly explained. This concept has been described as 'a general method for making a program (or other work) free, and requiring all modified and extended versions of the program to be free as well'.[6] In this sense, 'free' does not mean 'free of all restrictions' or 'free of copyright'. In fact, the key concept at the heart of 'copyleft' is that the person who creates the software has the right as the copyright-owner to license the work as he or she sees fit. In this regard, the collaborative nature of FOSS operates so that initial authorship is 'the first link in the chain'.[7] Every new creator/collaborator produces new original modifications to the code, and then licenses these new modifications onwards down the chain. As described in detail below, what is crucial about the different FOSS licenses is that some of the copyleft licenses, including the 'weak copyleft' and 'strong strong' licenses, require that this derivative material must be licensed under the same license as the first work in the chain. The 'no copyleft' licenses on the other hand typically allow modifications to be issued under any other license.

[2] 17 U.S.C. 101. CDPA 1988 s 3(1b). Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs.
[3] On the originality standard, in the US see *Feist Publication Inc. v Rural Telephone Service Inc*. (1991) 499 US 340, 345; for the UK see *University of London Press Ltd. v University Tutorial Press Ltd.* [1916] 2 Ch 601 and *Newspaper Licensing Agency v Meltwater* [2010] EWHC 3099 (Ch); for the EU see *Infopaq International A/S v Danske Dagblades Forening* (C-5/08) [2009] ECR I-6569 (ECJ (4th Chamber)); [2009] ECDR 16 259.
[4] 17 U.S.C. ss 101 and 103. For the UK see *Ibcos Computers Ltd. v Barclays Mercantile Highland Finance* [1994] FSR 275.
[5] CDPA 1988 S 3(1) (a). See also 17 U.S.C. 101.
[6] http://www.gnu.org/copyleft/
[7] L. Rosen, *Open Source Licensing – Software Freedom and Intellectual Property Law* (Upper Saddle River, NJ: Prentice Hall PTR, 2004), 28.

One final thing must be outlined here – the difference between static and dynamic linking. This distinction is of key concern as different legal conditions may apply to each category. The crucial difference between the two is how and when the linking takes place. 'Static linking' typically involves combining components 'through compilation, copying them into the target application and producing a merged object file that is a stand-alone executable'; on the other hand, 'dynamic linking' typically involves the use of components 'at the time the application is loaded (load time) or during execution (run time)'.[8] In particular, when a piece of software is 'linked' to another piece of software, the resulting software may or may not be described as a 'derivative work' (or to use a non-US description, a work which requires the authorisation of the copyright holder). The FSF generally takes an expansive view, arguing that even dynamic linking can create a 'derivative work'.[9] Nonetheless, the most common view, and the view taken in this chapter, is that if the linking is static, it is likely that a 'derivative work' is produced; however, if the linking is dynamic then it is likely that no 'derivative work' is produced, and therefore the licensor has no copyright interest which would enable him or her to place conditions on use.[10]

1.1 Outlining the key terms of FOSS licenses – 'Distribution' and 'Derivative Works'

Generally all FOSS licenses allow the user to make private use of the software. It is when the user seeks to re-distribute the software, or distribute a 'modified' version of the software, that the differences between the various FOSS licenses become clear. A number of key terms recur in many FOSS licenses and understanding these terms is integral to comprehending the difference between the licenses. In this regard it is particularly necessary to discuss the much debated FOSS

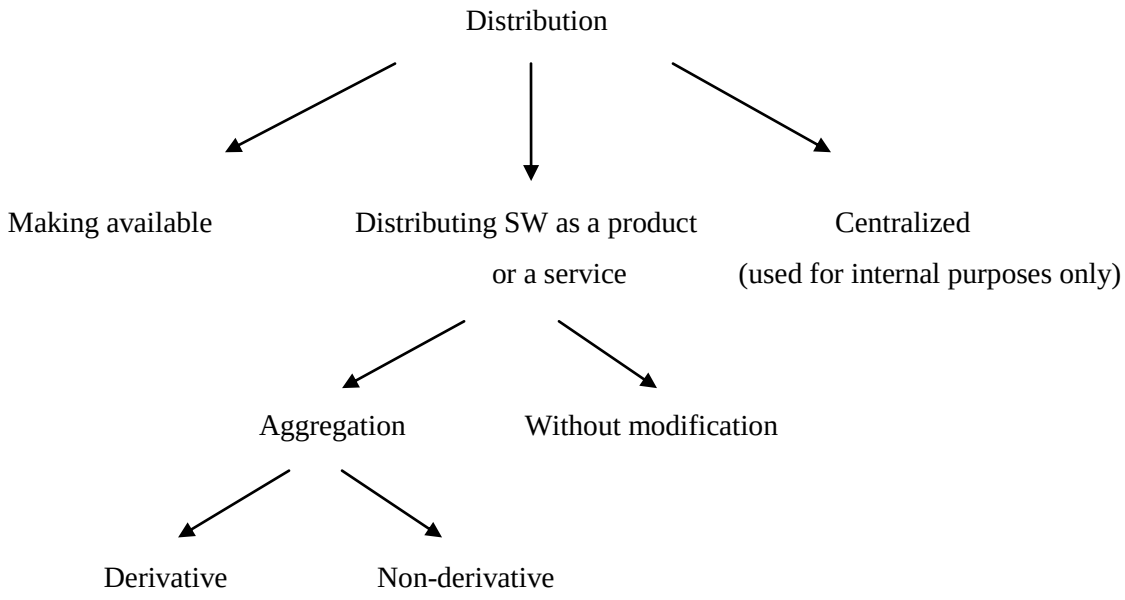
[8] See discussion of differences between static and dynamic linking at http://joinup.ec.europa.eu/software/page/eupl/eupl-compatible-open-source-licences#section-3

[9] See FSF's discussion of dynamic linking here - http://www.gnu.org/licenses/lgpl-java.html

[10] M. L. Stoltz, 'The Penguin Paradox: How the Scope of Derivative Works in Copyright Affects the Effectiveness of GPL,' *Boston University Law Review* 85 (2005), 1439, 1451. See also discussion of 'derivative works' in this context at http://joinup.ec.europa.eu/software/page/eupl/eupl-compatible-open-source-licences#section-3 – See also comments of J. Leonard, 'A Guide to Using Open Source Software': "In the case of static linking, the various portions of software are linked prior to compiling. If static linking used with open source and proprietary software, then, arguably, the open source has been modified and all of the source code that was linked to the open source software would need to be disclosed upon distribution. By contrast, however, in the case of dynamic linking, it might be argued that since modified software is really only created at the time the program is actually run and dynamically linked to others software. Thus, there may be no distribution since the modified software may only be created on an end-user's machine." - http://c.ymcdn.com/sites/www.coloradotechnology.org/resource/collection/335E7C59-E3F5-428F-9FAB-F318428051F4/A_Guide_to_Using_Open_Source_Software-FW_J._Leonard.pdf

concept of 'distribution'. Figure 1 illustrates the various types of 'distribution' that could occur in the context of FOSS. These terms are described here as including notions of 'making available', 'centralized' distribution, 'distribution of non-derivative works' and 'distribution of derivative works'.

Figure 1 – Possible FOSS Distribution channels



In this context, the idea of 'making available'[11] refers to the use of FOSS code in the making available of a product or facility. Google[12] and Facebook[13] are notable examples of this. Although FOSS code is used to facilitate the search engine, the FOSS code itself is not distributed. Within this type of use FOSS code may be interacting with proprietary code (which may be held as a trade secret and which is not released). For this reason many FOSS licenses do not seek to bind the licensee by placing restrictions on this type of use.[14]

[11] This category is self-standing for the purpose of this chapter and it bears no resemblance to any other known category of 'making available' as sometimes defined under copyright law.
[12] J. Corbet, 'How Google Uses Linux': http://lwn.net/Articles/357658/
[13] S. Campbell, 'How Does Facebook Work?': http://www.makeuseof.com/tag/facebook-work-nuts-bolts-technology-explained/
[14] One exception is the Affero GPL v 3 license, which was specifically designed in order to capture this kind of activity - http://www.gnu.org/licenses/agpl-3.0.html

The notion of 'centralized' distribution, on the other hand, covers internal distribution within a company or use on an intranet. In general, this type of use or distribution is not specifically addressed by a FOSS license's provisions. Moreover, it is strongly arguable that since there is no distribution to another natural or legal person, this type of use is commonly within the scope of a FOSS license, unless it is specifically forbidden.[15]

The idea of 'distribution of non-derivative works' includes distribution of original works and aggregations,[16] such as via dynamic linking, which do not create derivative works. The idea of 'distribution of derivative works' includes distribution of modified works and aggregations, such as via static linking, which do result in derivative works. In this regard, as outlined above, understanding the difference between static and dynamic linking is also crucial.

Furthermore, in this context the interpretation of the definition of 'derivative work' in court will be of great significance, as will an understanding of other common terms found in many FOSS licenses such as 'modified work' and 'contribution'. For instance, from the copyright perspective it is important to note that the term 'derivative work' does not have a universal meaning. Typically it is has a meaning under US law but it is a contestable notion in other jurisdictions such as those in Europe.[17] Some licensors attempt to clarify what they mean by 'derivative works' in the text of the license. For example, Linus Torvalds puts forward the notion that user-space programs i.e. non-kernel applications running on the Linux kernel, do not creative 'derivative works'.[18] Nevertheless, the ultimately authority for deciding this question remains the court.

---

[15] It is notable that the the Affero GPL v 3 license, which was designed to capture some forms of network/web use e.g. 'making available', specifically allows this 'centralized' use – See AGPL Frequently Asked Questions: http://www.affero.org/oagf.html. See also comments at Free Software Foundation Question and Answer section: http://www.gnu.org/licenses/gpl-faq.html#WhyPropagateAndConvey, http://www.gnu.org/licenses/gpl-faq.html#ConveyVsDistribute and http://www.gnu.org/licenses/gpl-faq.html#NoDistributionRequirements

[16] The FSF uses a slightly narrower use of the term 'aggregation' than the one in this chapter. The FSF clarifies that in its view 'mere aggregation' means distributing discrete non-derivative works on the same storage medium (e.g. on a same compact disc) and that this will not trigger the copyleft requirement - see http://www.gnu.org/licenses/gpl-faq.html#MereAggregation. This chapter makes use of the term 'aggregation' where relevant to cover distribution of both derivative and non-derivative works. Therefore, it can be said that this chapter refers to "aggregation" in a general sense instead of the FSF's term 'mere aggregation'.

[17] T. Jaeger, 'Enforcement of the GNU GPL in Germany and Europe,' *JIPITEC* 1 (2010), 34.

[18] See license text at http://www.kernel.org/pub/linux/kernel/COPYING, noting the disclaimer at the top of GPL.

As described below, there are three main categories of license – 'no copyleft', 'weak copyleft' and 'strong copyleft', all of which outline different permissions and restrictions with regard to 'distribution', and in particular distribution of 'derivative works'. As noted above, the contestability of some of the terms commonly found in FOSS licenses adds uncertainty to the meaning of license terms. As a result, a certain amount of reasoned speculation is inevitable. The license comparison analysis below must be read with this in mind.

1.2 Licenses featuring No Copyleft provisions – Apache 2.0, BSD and MIT

'No copyleft' licenses are licenses with limited or virtually non-existent 'copyleft' provisions. Typically software released under 'no copyleft' licenses can be used in nearly all distribution models, including proprietary and closed software models. These licenses tend to impose minimal or no restrictions on use and distribution e.g. affixation of notices, requiring specific trademark permissions etc.

The most prominent and popular 'no copyleft' license is **Apache 2.0**. It is written by the Apache Software Foundation.[19] With regard to licensing FOSS works under Apache, copying and linking are permitted under the license. [20] Regarding the key issue of 'distribution', distribution of the original version of the work is expressly allowed with minimal restrictions.[21] Similarly distribution of the 'Work' with modifications is allowed under the same terms.[22] The minimal restrictions include requiring a permission notice (license text) to appear in all copies of the source code, necessitating the provision of a copy of the license (s. 4.1), requiring the retention of all notices (s. 4.3-4), and requiring the giving of a notice of modifications (s. 4.2).[23]

[19] http://opensource.org/licenses/Apache-2.0
[20] The license states that '... each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.'
[21] The license states 'You may reproduce and distribute copies of the Work or Derivative Works thereof, in any medium, with or without modifications, and in Source or Object form …'
[22] 'You may reproduce and distribute copies of the Work or Derivative Works thereof, in any medium, with or without modifications, and in Source or Object form …' - Cu, PN, AM, CL-Apache License – only has limited effect. The 'Key' to these abbreviations is found in Annex I of this chapter.
[23] Different rules apply in s 5 to works submitted as a contribution to the Apache Software Foundation - Copyleft clause applies 'unless you explicitly state otherwise' and only to 'any Contribution intentionally submitted for inclusion in the Work by you to the Licensor…'

Apache 2.0 also provides a definition of 'Derivative Work'. It explicitly excludes 'works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof'. Under this definition, linked works are excluded from the notion of 'derivative work' - any work being linked to such a 'derivative work' would most probably not be deemed by a court to be part of such 'derivative work'. The reason for this is that both works would remain separable even if they are linked statically.

The license also gives a definition of 'Contribution' and 'Contributor'.[24] There are three additional requirements within the terms of Apache 2.0, two of which have caused some controversy with regard to compatibility with other licenses, as discussed in part 3 of this chapter. The first and least controversial requirement is a trade mark permission clause (s 6).[25] There is also a patent retaliation (termination) clause (s. 3), as well as an indemnity clause which operates in the case additional support is offered by a licensee (s. 9).

Given the permissive nature of Apache 2.0 it can be said to be generally suitable for all types of 'distribution' discussed above – 'making available', 'centralized' distribution, distribution of 'non-derivative works' and distribution of 'derivative works'.[26] Provided that the minimal requirements of the license are met, any person is able to modify the source code and release, commercially or non-commercially, a free/open or proprietary/closed version of Apache-licensed software.

The **BSD licenses** are a series of permissive, 'no copyleft' licenses authored by UC-Berkley.[27] The most common BSD licenses are the 3-clause 'modified' BSD license and the 2-clause 'simplified' BSD license. The primary difference between the two main BSD licenses is that the

[24] Under the license it is defined as '…any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, 'submitted' means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."' The license states "'Contributor' shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.'

[25] See further discussion of trade marks chapter [] of this book.

[26] However, given the patent retaliation provision, the use of Apache 2.0 may not suit a distributor or business that seeks to enforce software patents in the manner described in the retaliation provision.

[27] 2-clause (simplified) BSD license - http://opensource.org/licenses/BSD-2-Clause and 3-clause (modified) BSD license - http://opensource.org/licenses/BSD-3-Clause

'simplified' 2-clause version omits the non-endorsement clause found in the 'modified' 3-clause version. Copying and linking are permitted under the licenses - dynamic linking to the work is possible with no restrictions while static linking to the work falls within the scope of the permission to copy i.e. redistribution and use, both in source and binary forms, and with or without modification, are broadly permitted.[28] Distribution of the 'Work' is generally allowed with or without modifications.[29]

BSD licenses are very short at mere 2 or 3 clauses. This potentially leaves a lot open to interpretation. For example, no definition of 'derivative work' is given. However, the permissive nature of both licenses is undeniable. The minimal requirements of the licenses consist of requiring the affixation of both a copyright notice and a related liability disclaimer (applies to source code and binary code). Moreover, it is stated in the license that the name of the copyright holder and/or of the organization which created the license may not be used in advertising without prior permission. Like Apache 2.0, BSD licenses are generally suitable for all types of 'distribution' discussed above – 'making available', 'centralized' distribution, distribution of 'non-derivative works' and distribution of 'derivative works'. As above, provided that the minimal requirements of the license are met, any person is able to modify the source code and release, commercially or non-commercially, a free/open or proprietary/closed version of Apache-licensed software.

The **MIT license** is a permissive, 'no copyleft' license developed by the Massachusetts Institute of Technology.[30] Copying and linking are permitted under the license.[31] Dynamic linking to the Work is possible with no restrictions. Static linking to the Work falls within the scope of the permission to copy.[32]

[28] For the purposes of clarity and space, abbreviations are used here to give detail regarding the terms of the licenses. The key for these abbreviations is included as an annex at the end of the chapter. For this license the relevant abbreviations are C+, PN+.

[29] 'Redistribution and use in source and binary forms, with or without modification, are permitted…' - modified BSD license (C+, NA, PN+) and simplified BSD license (C+, PN+).

[30] http://opensource.org/licenses/MIT

[31] 'Permission is hereby granted … to deal in the Software without restrictions, including without limitations the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so…' - C++, PN++.

[32] The scope is described herein: If a modified Work links to another work statically such work might be deemed to form part of the modified Work and it would have to be treated accordingly. However, the fact that a modified Work links to another work dynamically does not mean that such other work forms part of the modified Work - C++, PN++.

The minimalist requirements of this license consist of an affixation of copyright notice requirement and requiring the use of a contract and tort disclaimer. Distribution of the 'Work' with or without modifications is allowed.[33] Like the BSD licenses, MIT is very short. Nonetheless, its permissive nature is clear. As a 'no copyleft' license it is generally suitable for all types of 'distribution' discussed above – 'making available', 'centralized' distribution, distribution of 'non-derivative works' and distribution of 'derivative works'. As above, provided that the minimal requirements of the license are met, any person is able to modify the source code and release, commercially or non-commercially, a free/open or proprietary/closed version of Apache-licensed software.

In conclusion, it must be noted that the 'no copyleft' licenses examined above are sometimes described as 'permissive' or copyfree.[34] Regarding the notion of a 'permissive' or 'copyfree' license, for the purpose of this chapter the key concept at the heart of these licenses is 'no copyleft' i.e. the lack of copyleft provisions restricting how the software can be redistributed. It is this that makes the licenses 'permissive' or 'free'. For this reason, and for the purpose of clarity, the term 'copyleft' is used. It is also important to reiterate that software released under a 'no copyleft' license can not only be used 'permissively' or 'freely' by the general computer programmer/user - companies can also make use of the code and incorporate it under stricter 'weak copyleft' or 'strong copyleft' licenses, as described below.

1.3 Licenses featuring Weak Copyleft provisions – MPL and LGPL

Licenses with 'weak copyleft' provisions can be easily utilized in some distribution models but not in others. For example, the provisions in these licenses usually require that derivative works must be issued under the particular license in question. However, non-derivative and/or 'linked' works may be distributed under another license - something which envisages commercial use in 'proprietary' software models. Therefore the source code for 'linked' software can remain closed even if this software is linked with open source code (which must itself remain open). Typically, there are other requirements with regard to trademarks, the use and availability of source code provisions, etc. Examples of these types of licenses discussed below are the Mozilla Public License 1.1 and the GNU Library or 'Lesser' General Public License v 2.1.

[33] 'Permission is hereby granted … to deal in the Software without restrictions, including without limitations the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so...' - C++, PN++.
[34] http://copyfree.org/standard/

The **Mozilla Public License 1.1 (MPL)** is a weak copyleft license written by the Mozilla Foundation.[35] Copying, display, performance and use are explicitly permitted.[36] As discussed further below with regard to compatibility, sub-licensing is also permitted.[37] The definition of the 'Covered Code' includes both the 'Original Code' and its 'Modifications'. Distribution of the 'work' without modifications is allowed. However, there are different conditions that apply to distribution in object code[38] and in source code.[39] Each Contributor must inform Recipients about any third party IPRs applicable to the software by including such information in a text file named 'LEGAL'.[40] If the recipient of the license cannot comply with all of its terms due to statute, regulation or judicial order, he or she can still use the work provided he or she complies with the terms to the maximum extent possible and provides the reasons why the LEGAL file cannot be complied with.

Distribution of the 'work' with modifications is allowed, with restrictions. As above there are different conditions that apply to distribution in object code[41] and in source code,[42] and each 'Contributor' must inform 'Recipients' about any third party IPRs by including this information in the 'LEGAL' file.[43] With regard to 'modifications' (the point which potentially relates to 'derivative works'), the license permits the recipient to create 'Larger Works'. The notion of a 'Larger Work' is defined as 'a work which combines Covered Code or portions thereof with code not governed by the terms'. This explicitly provides that in such case it is only the 'Covered Code' or portions thereof that must be subject to MPL, not the other parts of such 'Larger Work'. This permission encompasses both the situation where another program links to the 'Covered

[35] http://opensource.org/licenses/MPL-1.1
[36] C, PND.
[37] The license notes 'The Initial Developer (and each Contributor) hereby grants You a world-wide, royalty-free, non-exclusive license … to use, reproduce, ..., display, perform, sublicense ... the Original Code (or portions thereof) with or without Modifications and/or as part of a Larger Work (the Modifications created by such Contributor (or portions thereof) ... with other Modifications, as Covered Code and/or as part of a Larger Work'.
[38] ASC+ (source code must be made available for redistribution under conditions described hereafter).
[39] C, PN, CL - Mozilla PL.
[40] 'The Initial Developer (and each Contributor) hereby grants You a world-wide, royalty-free, non-exclusive license … to … sublicense and distribute the Original Code (or portions thereof) (the Modifications created by such Contributor (or portions thereof) ...'
[41] AM, ASC+ (source code must be licensed under conditions described hereafter).
[42] C, PN, AM, CL - Mozilla PL.
[43] 'The Initial Developer (and each Contributor) hereby grants You a world-wide, royalty-free, non-exclusive license … to … modify, ... sublicense and distribute the Original Code (or portions thereof) with or without Modifications and/or as part of a Larger Work (the Modifications created by such Contributor (or portions thereof) ... with other Modifications, as Covered Code and/or as part of a Larger Work'.

Code' (either statically or dynamically) as well as the circumstances where modified 'Covered Code' links to another program. In both cases the restrictions prescribed by the license apply only to the 'Covered Code' and not to other parts of the 'Larger Work', which may be licensed under different terms. This gives the MPL its 'weak copyleft' character. The meanings of 'Contributor' and 'Contributor Version' are defined in s 1.1 and s 1.2 – 'Contributor' is said to mean 'each entity that creates or contributes to the creation of 'Modifications' while 'Contributor Version' means 'the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor'.

The requirements of the license include the fact that source code must be published for a period of one year, or six months, from the time the executable version was made available. Furthermore, there must be affixation of a copyright notice. The license therefore requires the inclusion of the source code for any MPL aspects, for a limited time. As noted below with regard to compatibility, MPL envisages the use of other licenses, such as GPL. Overall it can be said that with respect to distribution, MPL is suitable for 'making available' and 'centralized' (though no specific provisions are given on these issues). The 'distribution of non-derivative works' allowable with the condition that the MPL-licensed code must be left open and accessible for a specified time (there are different requirements for object and source). The major limitation comes in the context of 'modifications'. MPL can also be used for 'distribution of derivative works', but the MPL requires that the derivative content must be licensed under MPL (though not the entire larger work).[44] As noted above, the fact that derivative content must be licensed under MPL, but in the case of any other type of interaction between the MPL-code and other code (such as static or dynamic linking) another license may be utilized, typifies the 'weak copyleft' category of license.

The **GNU Library or 'Lesser' General Public License v 2.1 (LGPL)** is a 'weak copyleft' license authored by the Free Software Foundation.[45] Copying is explicitly permitted.[46] Distribution of the unmodified 'Work' is allowed, but there are different conditions that apply to distribution in object code[47] and in source code.[48] Distribution of the 'Work' featuring

[44] In this regard the MPL code must be left open and accessible, with acknowledgement of the different requirements for object and source.
[45] http://opensource.org/licenses/lgpl-2.1.php
[46] C, Pnus.
[47] ASC+ (complete source code must be made available for redistribution under conditions described hereafter)

modifications is also allowed, with different conditions that apply to distribution in object[49] and in source code.[50] Permission to modify the 'Library' and to copy and distribute its 'modified' versions is subject to two additional conditions. Firstly, the 'modified' work itself must be a software library. Secondly, it is noted that 'if a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility ... then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful'.[51] These requirements apply to the modified work as a whole, including separable parts that are not derived from the 'Library'. However, it is crucial to note that mere aggregation of another work with the 'Library' on a volume of a storage or distribution medium does not bring the other work under the scope of LGPL. Therefore, works based on the 'Library' can be placed in a single library with other library facilities not covered by LGPL and be distributed within such a combined library provided that access to the works based on the 'Library', and used in such combined library, is granted under LGPL. Works that do not contain the 'Library', or any portion thereof, but are designed to work with the Library, by being compiled or linked with it are called 'works that use the Library'. These works have a specific regime described under the 'Linking' section of the LGPL.

Works which merely engage in 'dynamic linking' with the 'Library' most probably fall outside of the scope of the LGPL (despite the contrary intentions of its drafters).[52] As noted above, dynamically linked works are unlikely to create a copyright interest as a 'derivative work' which would enable the licensor to place conditions on the use of these works.

Works statically linking to the Library fall out of the scope of the LGPL until they are compiled with the 'Library'. 'Executables' created by linking a work that uses the 'Library' with the 'Library' are treated under the LGPL as derivatives of the 'Library'. These 'derivative works' can be said to fall under the LGPL and therefore they have a special regime different from

[48] 'You may copy and distribute verbatim copies of the Library´s source code as you receive it, in any medium…' - C, PNus, PN? or CL-GPL v2 or any later version of GPL.
[49] ASC+ (complete source code must be made available for redistribution under conditions described hereafter).
[50] 'You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work…' - C, PNus, PN?, AM, CL-LGPL or GPLv2 or any later version of GPL.
[51] LGPL section 2(d).
[52] See section 5 LGPL appears designed to apply to all linking - http://www.gnu.org/licenses/lgpl-2.1.html

modifications of the 'Library' itself.[53] The combining or linking of a work that uses the 'Library' with the 'Library' itself to produce a work containing parts of the Library is permitted as well is distribution of the resulting work under any terms, provided that the terms permit modification of the work for the customer´s own use and allow reverse engineering for debugging such modifications.[54]

In light of the above, it is clear that LGPL is a 'weak copyleft' license. Overall it can be said that the LGPL is suitable for 'making available' and 'centralised' distribution (including use as part of a 'Library' since works can be used for personal, internal purposes subject to conditions).[55] The license is also suitable for the distribution of non-derivative works, including software that merely links to the LGPL library and which is not considered as a 'derivative work'. However, all LGPL code must be left open and accessible. The LGPL is suitable for the distribution of derivative works, but crucially the LGPL requires that derivative content must be licensed under LGPL/GPL v 2 (see below).

1.4 Licenses featuring Strong Copyleft provisions – GPL v 2 and GPL v 3

Licenses with 'strong copyleft' provisions can only be used restrictively. These licenses typically maintain that 'derivative works' cannot be distributed under any other license and that full source code must be provided. Moreover, these licenses typically try to catch as much software 'material' within the remit of the license by taking as wide a definition of 'derivative work' as possible. For example, a 'strong copyleft' license will typically seek to prevent all linked works, whether linking statically or dynamically, from being issued under another license.[56]

The GNU General Public License (GPL) v 2 is a strong copyleft license written by the Free Software Foundation in 1991.[57] Copying is explicitly permitted.[58] The license grants the licensee the right to copy, distribute and modify the open source software on the crucial condition the

[53] This is the case except in relation to object files that use only numerical parameters, data structure layouts, small macros and small inline functions (up to ten lines in length). Such object files are unrestricted by the LGPL.
[54] The following conditions relate only to the Library itself: C?, Ca, PN?, ASC+CL-LGPL or GPL v 2 or any later version of GPL or using suitable shared library mechanism.
[55] Conditions include affixation of copyright notice and disclaimer, modified work must be a software library and be licensed to third parties with no charge, source code to be attached to any distributed works
[56] GNU Affero allows distribution (subject to conditions) on a web but not full distribution as a product or service - http://www.gnu.org/licenses/agpl-3.0.html
[57] http://opensource.org/licenses/GPL-2.0
[58] C, Pnus.

software is again distributed under the conditions of GPL v 2. The requirements of the license include making reference to GPL v 2, including the GPL license text, providing the source code and making reference to the disclaimer of warranty. There is a clear provision which states that failure to follow the license terms results in the revocation of the license, though third parties are deemed to be unaffected.

Distribution of the unmodified 'work' is allowed, though there are different conditions that apply to distribution in object code[59] and in source code.[60] The obligation to grant access to the source code in case of distribution of executable copies covers 'complete source code' defined as 'all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable'.[61]

Distribution of the 'work' with modifications is allowed, with different conditions that apply to distribution in object code[62] and in source code, but these derivative works must also be licensed under GPL v 2.[63] The obligation to grant access to the source code in case of distribution of executable copies covers 'complete source code'.[64]

It is not exactly clear what is deemed to be part of the modified or 'derivative' work and what is not. A work based on the Program is defined as 'any derivative work under copyright law: that is to say, a work containing the Program or a portion of it'. Regarding aggregation, it is explicitly stated in GPL v 2 that 'mere aggregation of another work not based on the Program with the Program ... on a volume of a storage or distribution medium does not bring the other work under the scope of this License'. GPL v 2 further states that when sections of the new work that can be reasonably considered independent and separate works in themselves are distributed as part of a

[59] ASC+ (complete source code must be made available for redistribution under conditions described hereafter).
[60] 'You may copy and distribute verbatim copies of the Program´s source code as you receive it, in any medium…' - C, PNus, PN?
[61] It is explicitly stated that 'the source code distributed need not include anything that is normally distributed ... with the major components (compiler, kernel, and so on) of the operating system on which the executable normally runs, unless this component itself accompanies the executable'.
[62] ASC+ (complete source code must be made available for redistribution under conditions described hereafter).
[63] 'You may modify your copy or copies of the Program´s source code or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work ...' - C, Ca, PNus, PN?, AM, CL-GPL.
[64] 'Complete source code' is defined as 'all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable' of the modified work as a whole'.

whole which is a work based on the Program, the distribution of the whole must be on the terms of GPL v 2. Regarding the drafters' intention, GPL v 2 explicitly mentions that its 'intent is to exercise the right to control the distribution of derivative or collective works based on the Program' and the second part of the definition of 'a work based on the Program' also suggests that collective works that include the Program or a portion thereof are deemed to be 'works based on the Program'.[65]

Despite the drafter's intentions, under copyright law where the new work, including the modification of the Program, could not be defined as a whole to be a derivative of the Program, it would instead be seen as a 'collective work'.[66] Regarding this collective work, it would typically be composed of 'a work based on the Program' and other separate works. These other separate works would not have to be licensed under GPL v 2, despite the intention of the drafters.[67]

With respect to linking, it was noted above that due to the fact that GPL v 2's restrictions apply to 'derivative works', anything which is outside the definition of a 'derivative' work will not be affected by the license's restrictions. According to the license if a modified Work links to other program (statically or dynamically) such program is deemed to form part of the modified Work and it must be treated accordingly. Static linking to the Work falls within the scope of the permission to copy.[68] However, if the resulting work is distributed the license states it must be treated as a modified Work (a work based on the Program). In this regard, the drafters of the GPL license firmly stand on the view that dynamic linking to the Work makes the program linking to the Work 'a work based on the Program' as well. However, this is most probably not true. It is strongly arguable that the mere act of dynamic linking does not constitute use of the Work in any way. Furthermore, if the new program linking to such Work is not distributed together with the Work it arguably cannot be caught under the GPL. Statically linked code on the other hand is more likely to be found to be 'derivative' and in this respect the GPL-derived code would have to be left open and accessible (acknowledging the different requirements for object and source code).

---

[65] See section 0 of http://www.gnu.org/licenses/gpl-2.0.html
[66] See discussion of 'Derivative work' in chapter one and above, supra n 10.
[67] See section 0 of http://www.gnu.org/licenses/gpl-2.0.html - It is also notable that even under the FSF's expansive view, not all collective works including the Program would be deemed to be 'works based on the Program' - only those that form one functional application would be. For example, according to the FSF if one internal module of MS Word was a program licensed under GPL v 2, the whole of MS Word would have to be licensed under GPL v 2, but not the complete MS Office package.
[68] C, Pnus.

Ultimately, it can be said that GPL v 2 is a 'strong copyleft' license. It is generally suitable for 'making available' and 'centralized' distribution (though no specific provisions on these issues). It is also suitable for 'distribution of non-derivative works', but any GPL v 2 code must be open and accessible (and there are different requirements for object and source). The license is suitable for 'distribution of derivative works', but GPL v 2 requires that derivative content must be licensed under the same terms. As noted above, the confusion in the license concerning the definition of a derivative work leaves the court with some room for interpretation, particularly regarding dynamically linked works. The attempt of the drafters to catch all varieties of linking within the GPL license is probably not successful.

GNU General Public License (GPL) v 3 is a strong copyleft license authored by the Free Software Foundation.[69] It is explicitly permitted to run the unmodified Program and to make, run and propagate works that are not conveyed, without restrictions. Under GPL v 3 works can be distributed subject to conditions: affixation of copyright notice and disclaimer, modifications must be licensed back to the public under the same terms, source code must be attached to any distributed works, respect for the anti-tivoization clause must be given.[70]

Distribution of the unmodified 'Work' is allowed.[71] In this regard, if the 'Work' is distributed in object code the 'Corresponding Source' must be provided together with the 'Work' by one of the ways described in the license.[72] If the 'Work' in object code is distributed in, with or specifically for use in a 'User Product', and the right of possession of the product is transferred to the recipient and anybody retains the ability to install modified object code on the 'User Product', the 'Corresponding Source' must be accompanied by the Installation Information.[73] As Asay has stated, this essentially requires that for any such user or consumer products 'any encryption keys

[69] http://opensource.org/licenses/GPL-3.0
[70] It also allows use with Affero GNU v 3 licensed-works.
[71] 'You may convey verbatim copies of the Program´s source code as you receive it, in any medium … You may convey a covered work in object code…' - C?, PN?, ASC.
[72] 'Corresponding Source' means "all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However it does not include the work´s System Libraries or general purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work.'
[73] The requirements concern all information, authorization keys and methods required to install and execute modified versions of the Work in such a 'User Product' – this also known as the 'anti-tivoization' clause.

or other information necessary to operate modified GPLv3'ed software on such products (i.e., the Installation Information) must be provided as part of the Corresponding Source'.[74]

Distribution of the 'Work' with modifications is permitted subject to the license restrictions.[75] Dates of alterations must be provided. If the Work is distributed in object code the Corresponding Source must be provided together with the Work by one of the ways described in the License.[76] There is also a requirement that seeks to bypass the requirements of the Digital Millennium Copyright Act 1998 (DMCA)[77] - if anybody conveys a covered work, it is stated that he or she waives 'any legal power to forbid circumvention of TPMs to the extent such circumvention is effected by exercising rights under this License'. Regarding the DMCA provision, GPL v 3 provides that no covered work shall be deemed part of an effective technological measure under any applicable law. However, the effectiveness of this provision is to some extent uncertain. For instance, it is very difficult to foresee in advance how a court or jurisdiction may interpret the notion of 'applicable law'. It is also possible that the program licensed under GPL v 3 may be used as part of TPMs protecting access to works, for example via online access, without being distributed together with such works, and therefore without the necessity to license that program to recipients of such works.

Regarding derivative content, sub-licensing is not allowed at all under GPL v 3. Nonetheless, with regard to aggregations, the license explicitly states that inclusion of a covered work into an aggregate[78] does not cause GPL v 3 to apply to the other parts of the aggregate. With respect to the issue of linking, although it is clear that the GPL v 3 drafters intended to catch linking within

[74] C. D. Asay, 'The General Public License version 3.0: Making or Breaking the FOSS Movement,' 14 *Mich. Telecomm. Tech. L. Rev*. 14 (2008), 265, 275.
[75] 'You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code … You may convey a covered work in object code …' - C?, Ca, PN?, AM, ASC, CL-GPL v 3.
[76] Corresponding Source refers to '…all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However it does not include the work´s System Libraries or general purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work.'
[77] Digital Millennium Copyright Act 1998 (DMCA) Pub. L. No. 105-304, 112 Stat. 2860 (Oct. 28, 1998).
[78] Defined as 'a compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium', but only 'if the compilation and it´s resulting copyright are not used to limit the access or legal rights of the compilation´s users beyond what the individual works permit'. If the Work in object code is distributed in, with or specifically for use in a User Product, and the right of possession of the product is transferred to the recipient, and anybody retains the ability to install modified object code on the 'User Product', the 'Corresponding Source' must be accompanied by the 'Installation Information'

the meaning of 'modification' as defined in the license,[79] it is doubtful whether this is really the case. It is strongly arguable that if a covered work dynamically or statically links to another program, the source code of such program must be provided as part of the 'Corresponding Source', unless it is a 'System Library' (and unless it is a 'Major Component' e.g. kernel or window system of the specific operating system). It is more unclear whether the program linked to by a modified work forms part of such modified work and must be therefore licensed under GPL v 3. The drafters of the license believe the answer is yes.[80] However, given the fact that some kind of derivative work' must have been created in order for the license to be binding, the answer is most probably no in relation to dynamically linked programs, and it may also be no even in relation to statically linked programs, depending on the nature of the modified work, as noted below.

The drafters believe that if another program links to a covered work such program must be licensed under GPL v 3 'because the program as it is actually run includes the library'.[81] However this reasoning seems to be flawed. The license does not impose any restrictions on running the covered work, only on conveying modified works, and furthermore the program dynamically linking to a covered work does not convey nor modify such work in any way. Static linking to a covered program would arguably result in a modified work that would have to be conveyed under GPL v 3 but it would depend on the nature of the resulting program i.e. whether such program could be defined as an aggregate, as defined above. Surprisingly the license does not mention specifically the situations in which covered works are 'linked to' by other programs.

Overall, GPL v 3 is generally suitable for 'making available' (though no specific provisions are provided on this issue).[82] With regard to 'centralized' distribution it appears that the GPL v3 does attempt to tackle this issue by introducing two new terms: 'propagation' and 'conveying'.[83] Under this view, mere 'propagation' that does not amount to 'conveying' of software licensed under GPL v 3 will not trigger the copyleft requirement. This appears to allow 'centralized' distribution.

GPL v 3 is also suitable for 'distribution of non-derivative works', though any GPL v 3 code must be open and accessible (there are different requirements for object and source). It is also possible

[79] See GNU FAQ - http://www.gnu.org/licenses/gpl-faq.html
[80] See GNU FAQ - http://www.gnu.org/licenses/gpl-faq.html
[81] See GNU FAQ - http://www.gnu.org/licenses/gpl-faq.html
[82] As noted supra n 13 and 14, there is a separate license, Affero GPL v 3, which does specifically attempt to capture the 'making available' of FOSS.
[83] See 'Definitions' section of GPL v 3 - http://www.gnu.org/licenses/gpl.html

to use the license for 'distribution of derivative works' but according to the license terms, distribution of statically or dynamically linked works must be under the terms of the GPL (noting that GPL code and any linked code must remain open and accessible). However, despite the contrary intention of the drafters of GPL v 3, it is likely that only derivative content is bound under GPL. In other words, dynamically-linked programs are probably not affected by this provision since they unlikely to be considered as modified or 'derivative works'.

2. FOSS Licenses – 'Contracts' or 'Bare Licenses'?

There has been a tremendous amount of academic debate concerning whether FOSS 'licenses' are in fact 'bare licenses' in the legal sense or whether they are in fact 'contracts'. In this respect there is some overlap between the different categories. Generally it is the case that a license can be a contract but it does not necessarily have to be one. A license is said to be analogous to giving permission – a licensor gives a licensee permission to do something which otherwise the licensee would not be able to do.[84] Over the course of this sub-section, the requirements for 'contracts' and 'bare licenses' are assessed, along with analysis of the consequences of finding that a FOSS license operates in either category.

2.1 Assessing the Requirements of a Contract in the FOSS context

The question of whether FOSS licenses are valid contracts has been much debated. For instance, on one hand Gomulkiewicz has argued that GPL licenses fulfil the legal requirements of a contract under the US model code Uniform Computer Information Transaction Act (UCITA).[85] However, the UCITA has been accused by Richard Stallman of being more suitable for use in the context of 'proprietary' software than FOSS and this is one reason the FSF has not accepted the idea that the GPL is a contract.[86]

---

[84] CDPA 1988 s 16. 17 U.S.C. ss 106.

[85] R. W. Gomulkiewicz, 'How Copyleft Uses License Rights to Succeed in the Open Source Software Revolution and the Implications for Article 2B,' *Houston Law Review* 36 (1999), 179. UCITA s 102 (a) (41) refers to a license as a contract defined as 'a contract that authorises access to, or use, distribution, performance, modification, or reproduction of, informational rights, but expressly limits the access or uses authorized or expressly grants fewer than all rights in the information, whether or not the transferee has title to a licensed copy.'

[86] R. Stallman, 'Why We Must Fight UCITA,' *Linux Today* (2000); accessible at http://www.linuxtoday.com/developer/2000020600105NWLF and http://www.gnu.org/philosophy/ucita.html

On this point Moglen has argued:

> "A contract… is an exchange of obligations, either of promises for promises or of promises of future performance for present performance or payment. The idea that 'licenses' to use patents or copyrights must be contracts is an artefact of twentieth-century practice, in which licensors offered an exchange of promises with users: 'We will give you a copy of our copyrighted work,' in essence, 'if you pay us and promise to enter into certain obligations concerning the work.' With respect to software, those obligations by users include promises not to decompile or reverse-engineer the software, and not to transfer the software."[87]

In order to evaluate whether FOSS licenses can properly be considered as 'contracts', the first thing which must be discussed are the requirements for the formation of a valid contract under the Anglo-American common law legal system, which typically occur via 'offer', 'acceptance' and 'consideration'.[88] Generally, contractual terms must be sufficiently drawn to the attention of the contracting party[89] and be within the 'reasonable expectations of the parties'.[90]

In this respect, finding the 'offer' in the FOSS context is relatively straightforward. Zhu has defined an offer in the FOSS context as 'a licensor's manifested willingness to give users permission to access, use, modify or redistribute a piece of FOSS and these permissions are usually accompanied by some restrictions pursuant to Free Software Definition and Open Source Definition'.[91] Meanwhile Rosen has noted that posting the offer to an accessible FOSS repository/website demonstrates a willingness to offer.[92]

---

[87] Moglen quote reported by P. Jones, 'The GPL Is a License, not a Contract' (2003); accessible at http://lwn.net/Articles/61292/

[88] J. Beatson, *Anson's Law of Contract* (Oxford: OUP, 2002, 28th Ed.), 1-37 and 88-89.

[89] *Thornton v Shoe Lane Parking* [1971] 2 QB 163

[90] *Equitable Life v Hyman* [2000] UKHL 39; [2002] 1 AC 408.

[91] C. Zhu, 'Authoring collaborative projects: a study of intellectual property and free and open source software (FOSS) licensing schemes from a relational contract perspective,' PhD thesis, The London School of Economics and Political Science (LSE), 148; accessible at http://etheses.lse.ac.uk/294/. See also American Law Institute, s 24, Restatement (Second) of Contracts, which defines an offer as the 'manifestation of willingness to enter into a bargain so made as to justify another person in understanding that his assent to that bargain is invited and will conclude it'; accessible at http://www.lexinter.net/LOTWVers4/restatement_(second)_of_contracts.htm

[92] In this sense 'all prospective licensees will be able to retrieve the software under the terms of the license'. L. Rosen, *Open Source Licensing – Software Freedom and Intellectual Property Law* (Upper Saddle River, NJ: Prentice Hall PTR, 2004), 60.

With respect to 'acceptance' the situation is more complex. It is generally the case that it must correspond exactly with the terms of the offer. In this vein, it must be 'absolute' and it must leave no doubt 'as to the fact of acceptance, or as to the coincidence of terms of the acceptance with those of the offer'.[93] In the software area, there is some case law regarding acceptance via 'shrinkwrap', 'clickwrap' and 'browsewrap'. The idea of 'shrinkwrap' concerns the consumer tearing off the shrinkable clear plastic on the software box. Once this has been done, the consumer is said to have assented to the terms of the license. In the US this type of acceptance was found to be valid in *ProCD v Zeidenberg*.[94] Nonehteless, this type of acceptance is controversial – Lemley has stated that the conduct supposedly showing evidence of a shrinkwrap contract 'is hardly unambiguous evidence of assent'.[95] In any event, most FOSS licenses are not 'accepted' via 'shrinkwrap' licenses because FOSS software often does not come in a box package, but is instead downloaded online. The 'clickwrap' and 'browsewrap' are more relevant to FOSS.

Zhu has remarked that 'clickwrap' licenses 'require affirmative actions from licensees to manifest their acceptance'.[96] Typically, in the context of this kind of license the user clicks a button to say 'Yes, I accept the license terms'. Kim has stated that since the user has notice of the terms and the ability to engage with them prior to acceptance, these types of licenses are generally less controversial than 'shrinkwrap' licenses.[97] Under a 'browsewrap' license it is assumed that because a user has installed the software the user is effectively a 'licensee' i.e. the user has agreed to the terms of the license (which can usually be viewed or 'browsed' on a webpage). The key element appears to be that there must be prominent notice of the license terms.[98] In this respect, GPL v 3 section 5 requires that 'prominent notices' must be given by licensors/downstream distributor, something which potentially means that GPL code does not require the 'clickwrap'

[93] J. Beatson, *Anson's Law of Contract* (Oxford: OUP, 2002, 28th Ed.), 37.
[94] *ProCD v Zeidenberg* 86 F.3d 1447 (7th Cir. 1996). See also F. Easterbrook, 'Copyright and Contract,' *Houston Law Review* 42 (4) (2005), 953.
[95] M. Lemley, 'Terms of Use,' *Minnesota Law Review* 91 (2006), 459, 468.
[96] C. Zhu, 'Authoring collaborative projects: a study of intellectual property and free and open source software (FOSS) licensing schemes from a relational contract perspective,' PhD thesis, The London School of Economics and Political Science (LSE), 153; accessible at http://etheses.lse.ac.uk/294/.
[97] N. S. Kim, 'Clicking and Cringing,' *Oregon Law Review* 86 (2007), 797, 842-843. Zhu has also noted that some FOSS software licenses also attempt to manifest acceptance via mere use of the software, as is the case with the Google Chrome Browser – C. Zhu, 'Authoring collaborative projects: a study of intellectual property and free and open source software (FOSS) licensing schemes from a relational contract perspective,' PhD thesis, The London School of Economics and Political Science (LSE), 153; accessible at http://etheses.lse.ac.uk/294/.
[98] See *Ticketmaster Corp. v Tickets.com, Inc* 2003 U.S. Dist. LEXIS 6483 (C.D. Cal. 2003) and *Specht v Netscape Communications Corp.* 306 F.3d 17 (2d Cir. 2002).

license. The OSI also stipulates that 'non-clickwrap' licenses are acceptable.[99] Nevertheless, from the point of view of 'acceptance' the 'browsewrap' licenses are more difficult to comprehend and courts may be less willing to enforce them.[100]

'Consideration' is the final major requirement for the formation of a valid contract. In *Currie v Misa*, Lush J. stated that 'a valuable consideration, in the sense of the law, may consist in some right, interest, profit, or benefit accruing to the one party, or some forbearance, detriment, loss, or responsibility given, suffered, or undertaken by the other'.[101] Treitel has noted that 'an act, forbearance or promise' amounts to consideration only if the court recognises that it has some economic value, even if that value cannot be precisely quantified.[102] Chen-Wishart has remarked that consideration must of the 'right kind' under the law, and 'non-monetary performance' of 'doubtful economic value' is difficult to assess in this context.[103] In this vein Zhu has stated with regard to FOSS 'volunteer licensees' their contributions are mostly non-monetary performances (e.g. reporting bugs or testing submitted patches etc.) therefore it is not always clear whether these performances can have the right 'economic values' to qualify as consideration'.[104] It also must be noted that the UK and US positions on consideration are not identical, something which adds another layer of complexity to these issues.[105]

Nonetheless, Wacha has taken the view that in the context of FOSS licenses, including GPL, there is valid consideration in the form of 'reciprocal promises' undertaken between the licensor and licensee(s), which require the licensees to do a number of things e.g. to post requisite notices, to distribute the code under the same terms etc., in return for making use of the FOSS software.[106] On the other hand, Kumar has argued that the adherence to the requirements of the FOSS license

[99] OSI Open Source Definition Criterion 10, Rationale - http://opensource.org/osd-annotated
[100] It has been noted that browsewrap licenses are more frequently enforced against businesses, rather than individuals – M. Lemley, 'Terms of Use,' *Minnesota Law Review* 91 (2006), 459 at 476.
[101] *Currie v Misa* (1975) L.R. 10 Ex. 153 at 162.
[102] G. Treitel, *The Law of Contract* (London: Sweet and Maxwell, 2003, 11th Ed.), 83.
[103] M. Chen-Wishart, Contract Law (Oxford: OUP, 2008, 2nd Ed.), 134
[104] C. Zhu, 'Authoring collaborative projects: a study of intellectual property and free and open source software (FOSS) licensing schemes from a relational contract perspective,' PhD thesis, The London School of Economics and Political Science (LSE), 158; accessible at http://etheses.lse.ac.uk/294/.
[105] For an examination of the English notion of 'consideration' see G. Treitel, The Law of Contract (London: Sweet and Maxwell, 2003, 11th Ed.), 67 and 83. For an examination of the US notion of 'consideration', which consists of a performance or promise which is bargained for and given in exchange for promise, see American Law Institute, *Restatement (Second) of the Law of Contracts* (1981) s. 71; accessible at http://www.lexinter.net/LOTWVers4/restatement_(second)_of_contracts.htm
[106] J. B. Wacha, 'Taking the Case: Is the GPL Enforceable,' *Santa Clara Computer and High Technology Law Journal* 21 (2005), 451, 474.

cannot be 'consideration' because this adherence does not 'directly benefit the licensor'.[107] Within this dichotomy, Zhu has pointed out that there are generally two types of FOSS licensee – one type of licensee is a mere consumer of the software, but the second type of licensee makes efforts to improve the FOSS software and often passes these changes on the FOSS community.[108] The assessment of consideration may depend on the court's view of the licensee's use of the software.

2.2. Assessing the Requirements of a 'Bare License' in the FOSS context

A license does not have to be a contract - it may be unilateral and not require mutual assent. Such a license is known as a 'bare license'. As a legal concept it has its roots in Land Law in common law systems.[109] Moglen has remarked:

> "The word 'license' has, and has had for hundreds of years, a specific technical meaning in the law of property. A license is a unilateral permission to use someone else's property. The traditional example given in the first year law school Property course is an invitation to come to dinner at my house. If, when you cross my threshold, I sue you for trespass, you plead my 'license,' that is, my unilateral permission to enter on and use my property."[110]

This is a more straightforward concept than a contract, as the requirements of 'offer', 'acceptance' and 'consideration' are not present. As noted above, a bare license is a unilateral permission to use the work in a manner which would otherwise infringe.[111]

The FSF claims that GPL is a unilateral bare license, arguing that the GPL licensees are not required to 'accept' the license:[112]

---

[107] S. Kumar, 'Enforcing the GNU GPL,' *University of Illinois Journal of Law, Technology and Policy* 1 (2006), 1, 19-21.
[108] C. Zhu, 'Authoring collaborative projects: a study of intellectual property and free and open source software (FOSS) licensing schemes from a relational contract perspective,' PhD thesis, The London School of Economics and Political Science (LSE), 159; accessible at http://etheses.lse.ac.uk/294/.
[109] I. J. Dawson and R. A. Pearce, *Licenses Relating to the Occupation or Use of Land* (London: Butterworths, 1979), 1. See also *Thomas v Sorrell* (1673) Vaugh 330 at 351.
[110] Moglen quote reported by P. Jones, 'The GPL Is a License, not a Contract' (2003); accessible at http://lwn.net/Articles/61292/
[111] L. Rosen, *Open Source Licensing – Software Freedom and Intellectual Property Law* (Upper Saddle River, NJ: Prentice Hall PTR, 2004), 65-66.
[112] E. Moglen and R. Stallman, 'Transcript of Opening session of first international GPLv3 conference,

"The GPL, however, is a true copyright license: a unilateral permission, in which no obligations are reciprocally required by the licensor."[113]

Regarding the issue of FOSS Licenses as bare licenses, the following passage from GPL v 2 is of note:

> "…[you] are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program..., you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it."[114]

This passage is intended to lay down the concept that since exclusive rights cannot be exercised without the permission of the copyright owner, a licensee must either follow the terms of the license or not exercise the rights i.e. any other action amounts to copyright infringement. A similar provision can be found in GPL v-3, section 9, which explicitly states that acceptance is not required for the license to operate.[115] No 'acceptance' of a 'bare license' is therefore required. As noted below, if a FOSS license is found to be a 'bare license' rather than a 'contract' this will have a number of legal consequences.

2.3 What are the Consequences of a FOSS license being held to be either a 'Contract' or 'Bare License'?

There are a number of consequences which arise from holding that a FOSS license is either a contract or a bare license.[116] Regarding enforceability, if FOSS licenses are held to be contracts

---

January 16th 2006' (2006); http://www.ifso.ie/documents/gplv3-launch-2006-01-16.html

[113] E. Moglen, 'Free Software Matters: Enforcing the GPL,' (2001); accessible at http://emoglen.law.columbia.edu/publications/lu-12.pdf

[114] GPL v 2 section 5.

[115] GPL v 3 section 9 states 'You are not required to accept this License in order to receive or run a copy of the Program' and '…nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.'

[116] M. Henley, '*Jacobsen v Katzer and Kamind Associates* – an English legal perspective,' *IFOSS L. Rev.* 1(1) (2009), 41.

the doctrine of privity of contract (which would not apply in the case of a bare license) has significant implications. The traditional understanding of this doctrine refers to the 'contractual relationship' which exists between the parties to a contract.[117] The relationship allows them to take legal action against each other in the case that one party is dissatisfied with the enforcement of the contract. However, the nature of the contractual relationship is that it binds only the parties to it – as a general rule, a contract cannot confer rights or impose obligations arising under it on any person except the parties to it. For instance, in the FOSS context a third party who made use of the FOSS might not be bound by the contract. Nonetheless, it must be noted that in recent years the doctrine has been relaxed both in the US[118] and the UK which alleviates this concern to some extent.[119] If, on the other hand, the FOSS license is held to be a 'bare license' then it would not enforceable against the licensor, with the exception of 'estoppel', as discussed further below.

Henley has further argued that with a contract courts are more willing to look beyond mere terminology and they often try to give effect to the intentions of the parties. The interpretation of terms is of great importance. In the absence of contract law, 'bare licenses' would be merely regulated by intellectual property law, which typically says very little on interpretation of license terms. There is therefore some uncertainty about terms that are not consistent with consumer protection laws e.g. warranty disclaimers as limitations of liability. One major reason for this claim by FSF is that the FSF seeks to avoid the diversity of contract law in preference for the more uniform application of copyright law under Berne. In particular it is notable that in the US copyright is largely a matter within the jurisdiction of the federal courts, whereas contract is much more within the various individual states' jurisdictions.[120] However, the interpretation of copyright law in various national jurisdictions is not as unitary as the FSF claims, particularly with regard to the concept of 'derivative work'. Furthermore, the different interpretations of contract law concepts in various national jurisdictions are not as divergent as the FSF claims. The question of whether a FOSS license is a 'contract' or a 'bare license' is also significant because

[117] The Law Commission, *Privity of Contract: Contracts for the Benefit of Third Parties* (31 July 1996); accessible at http://lawcommission.justice.gov.uk/docs/lc242_privity_of_contract_for_the_benefit_of_third_parties.pdf

[118] See *Lawrence v Fox* 20 N.Y. 268 and *Burr v Beers*, 24 N. Y. 178, which refer to an 'Intent to Benefit Test' which suggests that a third party ought to be able to enforce a contract if the parties intended to benefit such a party.

[119] Contracts (Rights of Third Parties) Act 1999. See also New Zealand legislation - The Contracts (Privity) Act 1982.

[120] See for example the recent case of *MDY Industries, LLC v Blizzard Entertainment, Inc and Vivendi Games, Inc.*, 629 F.3d 928 (9th Cir. 2010) and the discussion on point by R. W. Gomulkiewicz, 'Enforcement of Open Source Software Licenses: The MDY Trio's Inconvenient Complications,' *Yale J.L. & Tech* 14 (2011), 106.

there is the possibility that national legislation will include specific provisions regulating contracts which will not automatically apply to 'bare licenses'.[121]

In addition, Henley has stated that unlike a contract a bare license can be interpreted solely at the licensor's will –it is revocable.[122] In this regard GPL v 2 states:

"…all rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met…"

However, unless this term is part of a contract, this statement is merely a 'promise'. As such it may be revoked by the licensor at will. In this regard, Zhu has argued that the equitable doctrine of estoppel could empower a licensee to stop the full revocation of the license from taking place.[123] Promissory estoppel would work in this context where there is 'detrimental reliance' on the part of the licensee.[124]

It is also notable that the applicable governing law may differ in each case. For instance, under UK law 'the governing law for a contract dispute is determined by the Rome Convention on the Law Applicable to Contractual Obligations'.[125] In non-contractual obligations it will be decided 'by the Rome II Regulation or another statute of Private International Law'.[126] In fact it is logical that different laws apply to contractual disputes and disputes over 'bare licenses'. The FSF seeks to avoid the diversity of contract law in preference for what it sees as the more uniform application of copyright law under Berne. In particular it is notable that in the US copyright is largely a matter within the jurisdiction of the federal courts, whereas contract is more within the

[121] See examples in the UK - Sale of Goods Act 1979, applying to 'contracts of sale of goods', Supply of Goods and Services Act 1982, applying to 'contracts', and the Unfair Contract Terms Act 1977, which also governs 'contracts'.
[122] *Microsystems Software, Inc. v Scandinavia Online* AB, 98 F. Supp. 2d 74 (D. Mass., 2000), aff'd, 226F. 3d 35(1st Cir., 2000). See also D. McGowan, 'Legal Implications of Open-source Software,' *University of Illinois Law Review* (2001), 241, footnote 283 at 302.
[123] Denning MR in *Moorgate Mercantile Co. Ltd. v Twitchings* [1976] 1 QB 225, CA, at 241. See also *Central London Property Trust Ltd. v High Trees House Ltd*. [1947] KB 130.
[124] E. Cooke, *The Modern Law of Estoppel* (Oxford: OUP, 2000), 105. See C. Patterson, 'Copyright Misuse and Modified Copyleft: New Solutions to the Challenges of Internet Standardization,' *Michigan Law Review* 98 (2000), 1351.
[125] M. Henley, '*Jacobsen v Katzer and Kamind Associates* – an English legal perspective,' *IFOSS L. Rev.* 1(1) (2009), 41, 41-44.
[126] M. Henley, '*Jacobsen v Katzer and Kamind Associates* – an English legal perspective,' *IFOSS L. Rev.* 1(1) (2009), 41, 41-44.

various individual states' jurisdictions.[127] However, as noted above the extent to which copyright is interpreted in a uniform fashion, while contract law is interpreted in a diverse manner, tends to be overstated by the FSF.

As assessed below, there are also different 'remedies' which are applicable in each case e.g. specific performance can be ordered in the case of a contract. In the context of breach of contract, the unique doctrines of 'part performance' and 'specific performance' are potentially available.[128] Generally, in the arena of contract, damages are the usual remedy but are limited to those within the 'contemplation of the parties'.[129] Moreover, damages are the usual remedy for breach of contract in many European jurisdictions. In France, the amount of damages is allocated depending on the importance of the breach and its consequences.[130] In Germany, in a case of a breach of contract, or a breach of any obligation set forth under the contract, the main remedies/methods of compensation are damages or termination of the contract or a mixture of both.[131]

Finding a mechanism for calculating appropriate damages in the case of FOSS is not straightforward.[132] Regarding attorney fees and legal costs, in the US these are usually only recoverable if expressly provided for within the contract. In the UK and most European systems, such as the German one, there is a loser pays costs system, whereby the loser pays not only his own costs but the costs, or the majority of the cast, of the winner as well.[133]

---

[127] C. Zhu, 'Authoring collaborative projects: a study of intellectual property and free and open source software (FOSS) licensing schemes from a relational contract perspective,' PhD thesis, The London School of Economics and Political Science (LSE), 161-163; accessible at http://etheses.lse.ac.uk/294/

[128] Where a claimant partly performs their contractual obligations under the expectation the defendant will perform its obligations, a court may order specific performance – An order for specific performance is an equitable remedy to compel actual performance of contractual obligations.

[129] *Hadley v Baxendale* [1854] EWHC J70.

[130] Article 1142 of Civil Code *(Code Civil)*: Any obligation to do or not to do resolves itself into damages, in case of non-performance on the part of the debtor – accessible at http://www.legifrance.gouv.fr/affichCodeArticle.do?cidTexte=LEGITEXT000006070721&idArticle=LEGIARTI000006436337&dateTexte=20130222 - See also article 1142 + 1589 of Civil Code *(Code Civil)*.

[131] S. 280 et seqq. of Civil Law Code (*Bürgerliches Gesetzbuch*) – accessible at http://www.gesetze-im-internet.de/englisch_bgb/englisch_bgb.html#p0828. The termination of a contract depends on the sort of contract and how the contract was breached, S. 346 et.seqq. of Civil Law Code (*Bürgerliches Gesetzbuch*).

[132] Typically, there are two ways to calculate expectation loss (a) a 'cost of cure' analysis (ii) a 'difference in value' analysis, neither of which seem to suit the FOSS context.

[133] M. Gryphon, 'Assessing the Effects of A "Loser Pays" Rule on the American Legal System: An Economic Analysis and Proposal for Reform,' *Rutgers Journal of Law & Public Policy* 8 (2011), 567.

Furthermore, if FOSS Licenses are bare licenses a licensor cannot restrict activities that do not amount to copyright infringement - such as dynamic linking.[134] This is important because in the context of copyright injunctions can only be granted to prevent infringing distribution. In common law jurisdictions injunctions are generally available as a remedy in copyright law, but it is typically difficult, though not impossible, to get an injunction for breach of contract.[135] In European jurisdictions there may also be a difference in remedies available with respect to contract and copyright. To take the example of Spain, in a case of copyright infringement the available remedies are cessation, damages and the granting of an injunction.[136] In a case of breach of contract, the norm is to either enforce performance, or claim termination, of the relevant obligation, and to claim damages.[137]

In the UK with respect to damages, both tortious damages, in the form of reasonable compensation, and statutory damages are potentially available in the copyright context.[138] Other potential remedies include an account of profits and' delivery up'.[139] Moreover, criminal remedies often exist for commercial scale copyright infringement, while these generally do not exist with respect to breach of contract.

One other thing is of significance - with respect to remedies in the field of copyright, only copyright owners have the ability to enforce these rights in court. This could prove to be problematic in the FOSS context. As noted above, FOSS software typically involves numerous contributors, who are not sole copyright owners of every right which exists in the work, but instead own only a part of the copyright. It may prove to be difficult both to identify and to distinguish between the owner and the distributor without imposing large costs on potential users and developers.

---

[134] For example, 'strong copyleft' licenses e.g. GPL v 2 and GPL v 3, include clauses restricting dynamic linking and 'derivative works'.

[135] See for instance *Warner Brothers v Nelson* [1937] 1 KB 209 and *Page One Records v Britton* [1968] 1 WLR 157.

[136] Articles 138-141, Consolidated text of the Law on Intellectual Property, regularizing, clarifying and harmonizing the Applicable Statutory Provisions (approved by Royal Legislative Decree No. 1/1996 of April 12, 1996, and last amended by Royal Decree No. 20/2011 of December 30, 2011) - http://www.wipo.int/wipolex/en/details.jsp?id=11050

[137] Article 1124 of the Civil Code *(Código Civil)* - http://www.wipo.int/wipolex/en/text.jsp?file_id=221319

[138] CDPA 1988 s 97(2).

[139] Directive 2004/48/EC of the European Parliament and of the Council of 29 April 2004 on the enforcement of intellectual property rights; accessible at http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32004L0048R(01):EN:HTML.

2.4 Legal Enforceability of FOSS Licenses – Enforcement in the US and Europe

It is likely that the enforceability of FOSS licenses, including the question of whether a FOSS license is a 'bare license' or a 'contract', will be jurisdiction-dependent. Regarding the US jurisdiction, the case of *Jacobsen v Katzer* is of significance to enforcement of FOSS licenses.[140]

The *Jacobsen* case hinged upon the meaning of a term of the 'Artistic license'.[141] Jacobsen had devised software for controlling model trains and released the software under Artistic license. A key obligation of the license is when distributing the work to include attribution notices as well as identification of any modifications. Katzer, the defendant, had failed to provide attribution or give identification of modifications. This key term fell to be considered by the courts.

Henley has remarked that the question turned on whether the provision breached was 'a condition of the license, or a mere covenant'.[142] In other words, the case hinged upon whether the crucial term of the Artistic License would be interpreted by the court as amounting to either a 'condition' of the contract or as a mere 'covenant'. In the District Court the court stated that the license included both contractual covenants and copyright conditions:

> "[t]he condition that the user insert[s] a prominent notice of attribution does not limit the scope of the license."

Thus, according to the District Court, violation of the Artistic License's terms constituted a breach of contract, rather than copyright infringement. This decision affected the type of relief available to Jacobsen. Typically, injunctive relief, which is available in the context of copyright infringement, is unlikely to be available for breach of contract.

---

[140] *Jacobsen v Katzer*, 2007 WL 2358628 (N.D. Cal. 2007); Pre-Jacobsen US cases include - *Progress Software Corporation v MySQL* AB, 195 F. Supp. 2d. 328, 239 (D. Mass 2001); Complaint, *Drew Technologies, Inc. v Society of Auto Engineers, Inc*., No. 2:03-CV-74535 (DT), 2003 WL 238562505; , *Planetary Motion, Inc. v Techplosion, Inc*. 261 F.3d 1188 (11th Cir. 2001); Computer Associates International v Quest Software, Inc., 333 F. Supp. 2d. 688, 697-98. (N.D. III. 2004). See also the various cases taken by the Software Freedom Law Center assessed in detail by H. Meeker, 'Open Source and the Age of Enforcement,' *Hastings Science and Technology Journal* 4(2) (2012), 267, 271-274.

[141] http://opensource.org/licenses/artistic-license

[142] M. Henley, '*Jacobsen v Katzer and Kamind Associates* – an English legal perspective,' *IFOSS L. Rev.* 1(1) (2009), 41, 41-44.

Nevertheless, the Court of Appeals for the Ninth Circuit vacated the District Court's ruling.[143] The court found that Katzer's obligations did amount to 'conditions' limiting the scope of the license – these were not independent contractual 'covenants'. With regard to the key contract question of 'consideration', the court stated:

> "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."

The court therefore found that the license was of a 'hybrid' nature, which did include enforceable copyright 'conditions'. Katzer's actions had gone beyond the scope of the license by failing to comply with these 'conditions'. Therefore an action for copyright infringement could legitimately be brought by the licensor and the appropriate remedies sought.[144]

The *Jacobsen* case was undoubtedly an important one for the enforceability of FOSS licenses.[145] By finding that the term was a contractually enforceable 'condition' of the contract, the court confirmed that such licenses can be legally binding and enforceable in the US jurisdiction. A significant post-*Jacobsen* case, although it does not directly concern FOSS licenses, is *MDY v Blizzard*.[146] In this case, the Ninth Circuit stated that a term prohibiting the use of 'bots' was a 'covenant', not a 'condition'. Crucially, in making this decision it cited relevant state law, whereas in the *Jacobsen* decision the court did not defer to state law.[147] Gomulkiewicz has remarked that the effect of this case may lead to 'inconvenient complications' arising in the future with regard to FOSS licenses.[148] In particular, he has argued that the method of delineating between contractual covenants and license conditions laid down in *MDY* may make it more

---

[143] *Jacobsen v Katzer*, 535 F.3d 1373, (Fed. Cir. 2008).
[144] The case eventually reached a court settlement on 19 February 2010. Katzer agreed to pay Jacobsen $100,000. Katzer also accepted a permanent injunction against him copying or modifying the relevant software; settlement reported at http://yro.slashdot.org/story/10/02/19/1614216/jacobsen-v-katzer-settled-victory-for-foss.
[145] C. Zhu, '"Copyleft" Reconsidered: Why Software Licensing Jurisprudence Needs Insights from Relational Contract Theory,' *Social & Legal Studies* [forthcoming 2013, draft copy on file with author] has nonetheless argued that the *Jacobsen* ruling does not radically deviate from the still dominant neo-classical software licensing jurisprudence since the *ProCD* ruling.
[146] *MDY Industries, LLC v Blizzard Entertainment, Inc and Vivendi Games, Inc*., 629 F.3d 928 (9th Cir. 2010).
[147] H. Meeker, 'Open Source and the Age of Enforcement,' *Hastings Science and Technology Journal* 4(2) (2012), 267, 286.
[148] R. W. Gomulkiewicz, 'Enforcement of Open Source Software Licenses: The MDY Trio's Inconvenient Complications,' *Yale J.L. & Tech* 14 (2011), 106.

difficult for open source licensors from obtaining injunctive relief. Nonetheless, the overall picture for FOSS enforceability in the US jurisdiction is a positive one.

Indeed, in light of the above analysis of *Jacobsen* it is worth considering whether the major licenses explored over the course of this chapter, Apache 2.0, BSD, MIT, MPL, LGPL, GPL v 2 and GPL v 3, are likely to be interpreted as 'contracts' or 'bare licenses' under *Jacobsen*. Menon has argued that the terms of GPL v 2 are likely to be interpreted as 'conditions' because they use the appropriate language, including conditional phrases such as 'provided that'.[149] This would expose the user to copyright liability. The same can be said with respect to other GNU licenses such as LGPL and GPL v 3, which also use 'conditional' terminology.[150] Apache 2.0 uses traditional contractual language, subjecting use to 'terms and conditions'. As such, its provisions can largely be considered to be 'conditions', while the same can be said for the BSD and MIT licenses.[151] Moreover, while MPL uses terms more traditionally associated with 'covenants' rather than conditions, such as 'must do' and 'curing the breach', features a termination clause which is undoubtedly 'conditional'.[152]

Nevertheless, Goss has remarked that since FOSS licenses depend on contract law for enforcement, this may present challenges for courts, particularly since contractual issues may vary from jurisdiction to jurisdiction. Thus, even though a FOSS license was enforceable in the US jurisdiction, in another jurisdiction the interpretation of the law may well be different.

As yet, there is no UK case concerning the validity of FOSS licenses. Despite this, it has been argued that if a case similar to *Jacobsen v Katzer* was to come before the courts in the UK jurisdiction, a different conclusion would be reached on whether a 'contract' exists between the licensor and licensee.[153] For instance, Shemtov has argued that in line with *Currie v Misa*[154] UK

[149] Y. Menon, 'Jacobsen Revisited: Conditions, Covenants and the Future of Open-Source Software Licenses,' *Washington Journal of Law, Technology and the Arts* 6(4) (2011), 311, 336-338.
[150] Y. Menon, 'Jacobsen Revisited: Conditions, Covenants and the Future of Open-Source Software Licenses,' *Washington Journal of Law, Technology and the Arts* 6(4) (2011), 311, 338-340. However, Menon has further noted that GPL v 3 also includes other terms, such as the DRM clause, which may not be considered to be 'conditions' by a court.
[151] Y. Menon, 'Jacobsen Revisited: Conditions, Covenants and the Future of Open-Source Software Licenses,' *Washington Journal of Law, Technology and the Arts* 6(4) (2011), 311, 343-347.
[152] Y. Menon, 'Jacobsen Revisited: Conditions, Covenants and the Future of Open-Source Software Licenses,' *Washington Journal of Law, Technology and the Arts* 6(4) (2011), 311, 347-349.
[153] M. Henley, '*Jacobsen v Katzer and Kamind Associates* – an English legal perspective,' *IFOSS L. Rev.* 1(1) (2009), 41, 41-44.
[154] (1975) L.R. 10 Ex. 153.

courts may not find a binding contract to have been formed due to the lack of 'consideration'.[155] On this point Henley has remarked that the courts of England and Wales would likely consider the Artistic License as a bare license rather than a contract.[156] Moreover, with regard to the condition/covenant distinction, which was of crucial importance under US law in *Jacobsen v Katzer*, it must be stated that many European jurisdictions including France, Germany, and Italy do not feature this same condition/covenant distinction.[157] Nonetheless, a number of courts in civil law jurisdictions such as Germany and France have accepted that such licenses are legally valid.[158] For instance, the German case of *Welte v Sitecom Deutschland GmbH* is of significance due to the fact that the Munich District Court held that failing to comply with GPL license terms could constitute both a breach of contract and copyright infringement.[159] This is in line with *Jacobsen v Katzer*. In France, the case of *EDU v AFPA*, also known as the 'Paris GPL case', is of significance because the court seemed to accept that a violation of GPL terms could bring copyright infringement considerations into play.[160]

The fact that civil law jurisdictions have so far found FOSS licenses to be valid and enforceable ought to come as no surprise. Civil law jurisdictions typically consider license agreements, including FOSS licenses, to be enforceable contracts because 'consideration' is generally not a

[155] N. Shemtov, 'FOSS License: Bare License or Contract'; presentation accessible at http://web.ua.es/es/contratos-id/documentos/itipupdate2011/shemtov.pdf
[156] M. Henley, '*Jacobsen v Katzer and Kamind Associates* – an English legal perspective,' *IFOSS L. Rev.* 1(1) (2009), 41, 41-44.
[157] Regarding the relevant French legal terms 'obligation', 'conditions precedent' and 'conditions subsequent', see Articles 1168, 1181 and 1182 of the French Civil Code. In Germany, parties to an agreement may draft a contract based on s. 311 and S.241 Civil Law Code, while 'conditions' are regulated by virtue of s. 158 et. seqq Geman Civil Law Code. Under Italian contract law the relevant definitions of 'conditions' are set forth in s. 1353 of Italian Civil Code.
[158] German cases include *Welte v Sitecom Deutschland GmbH* District Court of Munich, 19 May 2004, case 21 O 6123/04; *Welte v Skype Technologies S.A.* District Court of Munich, 12 July 2007, case 7 O 5245/07. The major French case is *EDU 4 v AFPA*, Cour d'Appel de Paris, Pole 5, Chambre 10, no: 294. Many European cases have been taken by Harald Welte, founder of http://www.gpl-violations.org/. Welte has a case pending against Iliad, a French telecom company, over the failure to disclose source code with regard to 'Freebox' - a DSL technology which extensively uses GPL-licensed software; details accessible at http://gpl-violations.org/links.html.
[159] T. Jaeger, 'Enforcement of the GNU GPL in Germany and Europe,' *JIPITEC* 1 (2010), 34. It is also noted in this case that GPL licensors can rely on Directive 2004/48/EC of the European Parliament and of the Council of 29 April 2004 on the
enforcement of intellectual property rights; accessible at http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2004:195:0016:0025:en:PDF
[160] M. von Willebrand, 'Case law report: A look at EDU 4 v. AFPA, also known as the "Paris GPL case",' *IFOSSLR* 1(2) (2009), 123.

formal requirement of contract formation.[161] Moreover, it is highly unlikely that a FOSS license could be considered as akin to a 'bare license' – as noted above this is a concept which has its roots in Land Law in common law jurisdictions.

Ultimately, Shemtov has remarked that FOSS licenses in civil law jurisdictions appear to have a 'dual nature'; where relevant either, or both, contract law and copyright law may provide remedies.[162] It seems that the terms of FOSS licenses are valid and enforceable conditions. Nonetheless, case law in all jurisdictions is still in its infancy, which means the above assessment of the enforceability of FOSS licenses must be greeted with a degree of caution.

3. Examining License Compatibility

It has been argued that open source licenses are now overly diverse, and that this diversity could lead to legal complications.[163] For instance, the proliferation of different licenses that are potentially available to programmers may make it difficult for later users/contributors/distributors to comprehend which uses are acceptable and legal.[164] For this reason, the Open Source Initiative has tried to curb the enactment of new licenses, and some older, or poorly designed, licenses have effectively been 'retired' from use. However, it has been argued that these efforts have largely failed to prevent the negative aspects of proliferation from taking place.[165] Furthermore, the majority of software programs are released via one of the popular licenses examined over the course of this chapter, which to some extent mitigates some of the proliferation issues. On this point, it has been argued that the proliferation of licenses represents both 'helpful diversity' and 'hopeless confusion'.[166] In other words, it is unfortunate that confusion often results from license proliferation, but there may be no other way to satisfy the diverse licensing needs of the software programmers.

---

[161] A. Guadamuz-Gonzalez, 'The License/Contract Dichotomy in Open Licenses: A Comparative Analysis,' *University of La Verne Law Review* 30 (2) (2009), 296, 302-304. See also H. MacQueen and J. M. Thomson, *Contract Law in Scotland* (London: Tottel Publishing, 2007, 2nd ed.), 54-56.
[162] N. Shemtov, 'FOSS License: Bare License or Contract'; presentation accessible at http://web.ua.es/es/contratos-id/documentos/itipupdate2011/shemtov.pdf
[163] L. Guibault and O. van Daalen, *Unravelling the Myth around Open Source Licenses: An Analysis from a Dutch and European Law Perspective* (The Netherlands: T.M.C. Asser Press, 2006), 4.
[164] R. W. Gomulkiewicz, 'Open Source License Proliferation: Helpful Diversity or Hopeless Confusion?,' *Washington University Journal of Law and Policy* 30 (2009), 261, 261-263.
[165] R. W. Gomulkiewicz, 'Open Source License Proliferation: Helpful Diversity or Hopeless Confusion?,' *Washington University Journal of Law and Policy* 30 (2009), 261, 291.
[166] R. W. Gomulkiewicz, 'Open Source License Proliferation: Helpful Diversity or Hopeless Confusion?,' *Washington University Journal of Law and Policy* 30 (2009), 261, 291.

The assessment undertaken in this sub-section discusses compatibility in terms of 'one way compatibility' and 'two way compatibility'. One way compatibility refers to the fact that when two licenses are compared, material which works under License A can be taken, modified and licensed under License B, but material under License B cannot be taken, modified and incorporated under License A. In other words, the licenses are compatible in one direction only. Typically License A is a 'no copyleft' license and License B is a 'weak copyleft' or a 'strong copyleft' license. Two-way compatibility implies some degree of reflexive/reciprocal compatibility, such as via licensing or via linked works.

There is also an important jurisdictional concern which arises here. As noted above, the term 'derivative work' has a meaning under US law[167], but it is a contestable concept in other jurisdictions such as those in Europe.[168] The jurisdictional interpretation of the boundaries of a 'derivative work' or a work featuring modifications will be of great significance. Indeed, the question of whether dynamic and/or static linking creates a 'derivative work' could be answered differently in the various jurisdictions of the US and Europe. Furthermore, as described above, the notion of 'distribution' may not have a uniform meaning. The notions of distribution discussed here are in line with those discussed in sub-section 1 of this chapter – 'making available', 'centralized' distribution, 'distribution of non-derivative works' and 'distribution of derivative works'.

3.1 Compatibility between the 'No Copyleft' licenses - Apache 2.0, BSD and MIT

There are few complications which can arise with regard to compatibility between the permissive, 'no copyleft' licenses. Copying and linking are broadly permitted by all three licenses examined here, with only minimal requirements. As noted above, the most popular 'no copyleft' license is Apache 2.0. It is broadly two-way compatible with BSD. In other words, Apache and BSD material can be incorporated under either license. With respect to two-way compatibility between Apache 2.0 and MIT, Apache and MIT materials can be incorporated under either license. Similarly, BSD is two-way compatible with MIT - BSD and MIT materials can be incorporated under either license.

[167] See chapter [] of this text.
[168] T. Jaeger, 'Enforcement of the GNU GPL in Germany and Europe,' *JIPITEC* 1 (2010), 34.

3.2 Compatibility between the 'Weak Copyleft' licenses - MPL and LGPL

With regard to compatibility, the weak copyleft licenses tend to have limited two-way compatibility. A weak copyleft license typically stipulates that derivative content must be licensed under that same license. However, these restrictions tend to not be applied in the context of 'linked' works. Therefore, these licenses generally allow and encourage linking.

In line with this, MPL has two-way limited compatibility with LGPL. In this respect, there is no compatibility regarding derivative works - these must be licensed under either MPL or LGPL so no compatibility it possible. However, linking is permitted by LGPL and MPL which means that in the context of 'non-derivative works' the licenses are compatible. Furthermore, clause 13 of MPL concerns choices made by the 'initial developer', whereby the latter can designate portions of the covered code as 'multi licensed'. It appears to bet the case that if you follow 'Exhibit A' in the MPL (as referred to by clause 13) you may enable third parties to utilize parts of the code that you released under MPL under other licenses of your choice. Therefore clause 13 allows alternative use of GPL in limited circumstances.

3.3 Compatibility between the 'Strong Copyleft' Licenses - GPL v 2 and GPL v 3

There are compatibility problems between the strong copyleft licenses.[169] Under the strict terms of these licenses, there is little that can be done with the material that will not apparently cause the copyleft clauses to come into effect. In this respect, GPL v 2 is generally thought not to be compatible with GPL v 3.[170] Nonetheless, v 2 envisages use via the terms of later versions of the GPL. Therefore, there is a possibility that v 2 code may be used under GPL v 3 i.e. it is possible to license the content under v 3.

3.4 Compatibility between the 'No Copyleft' licenses and the 'Weak Copyleft' licenses

With regard to compatibility between the 'no copyleft' licenses and the 'weak copyleft' licenses there is one-way compatibility between the licenses. In this sense, derivative 'no copyleft' material can be incorporated under the 'weak copyleft' license, but not vice versa, though there are

[169] http://www.gnu.org/licenses/license-list.html#GPLCompatibleLicenses
[170] See GNU FAQ - http://www.gnu.org/licenses/gpl-faq.html

typically some exceptions allowing limited two-way compatibility between the licenses for 'linked' works, as discussed below.

Apache 2.0 is one-way compatible with MPL. MPL requires that derivative content be licensed under MPL. In other words, Apache-derived material can be incorporated under MPL but not vice versa. However, as detailed above, under MPL clause 13 if the programmer follows 'Exhibit A' in the MPL (as referred to by clause 13) he or she may enable third parties to utilize parts of the code that were released under MPL under other licenses of your choice. Therefore MPL clause 13 allows alternative use of Apache 2.0 in limited circumstances, and therefore there is some two-way compatibility between the licenses. As with MPL, Apache 2.0 is generally one-way compatible with LGPL in that LGPL requires derivative content to be licensed under MPL.[171] Therefore, Apache-derived works can be incorporated under LGPL, but not vice versa. As with MPL, there is also the possibility of two-way compatibility via linking - software that links to LGPL library not considered a derivative work (clause 5).

BSD is one-way compatible with MPL (MPL requires that derived content be licensed under MPL). BSD material can be incorporated under MPL but not vice versa (unless specifically indicated under MPL clause 13). Similarly, BSD is one-way compatibile with LGPL - BSD material can be incorporated under a LGPL but not vice versa. LGPL requires that content be licensed under GPL or LGPL (clause 2).

MIT is one-way compatible with MPL. MIT-derived works can be incorporated under MPL, not vice versa, unless specifically indicated under MPL clause 13. Derivative works must be licensed under MPL. MIT is one-way compatible with LGPL. MIT-derived works can be incorporated under LGPL, not vice versa.

3.5 Compatibility between the 'No Copyleft' licenses and the 'Strong Copyleft' licenses

There is clear one-way compatibility between the no copyleft licenses and the strong copyleft licenses. The terms of the strict strong copyleft licenses mean that no copyleft content can be integrated under a strong copyleft license, but this does not work the other way around because the strong copyleft license requirements do not allow this. Moreover, as outlined above, the

[171] Derivative content must be licensed under GNU GPL or GNU LGPL - clause 2.

'strong copyleft' licenses typically try to catch any 'linked' material within the terms of the license.

Regarding GPL v 2, Apache 2.0 has one-way compatibility with the license. In other words, Apache material can be incorporated under GPL v 2 but not vice versa. Linking is broadly permitted by Apache (clause 2) but GPL v 2 requires that derivative works must be licensed under GPL. However, since GPL v 2 refers to 'derivative works' it is likely that anything which is outside the definition of a 'derivative work' will not be affected. In addition, with regard to the contract/bare license debate discussed above, in a jurisdiction where a FOSS license is considered to be a bare license rather than a contract, it may not be possible legally to impose these types of obligations on downstream users.

The FSF claimed that GPL v 2 was not compatible with Apache because the patent retaliation clause and the indemnity clause are seen as 'further restrictions'.[172] Nonetheless, Lovejoy has noted that this interpretation was not accepted by the Apache Software Foundation, which claimed that the terms are in line with GPL v 2.[173] Furthermore, given the 'no copyleft' nature of Apache it is possible that a court would take this permissive nature into account when determining compatibility – it seems unlikely that the court would take a restrictive view of the Apache 2.0 requirements.

BSD licenses have one-way compatibility with GPL v 2. As above, BSD material can be incorporated under GPL v 2 but not vice versa. Copying and linking are broadly permitted by BSD but GPL requires that derivative works be licensed under GPL. However since GPL v 2 refers to 'derivative works' anything which is outside the law's definition of a 'derivative work' will likely not be affected. As noted above, in a jurisdiction where such a license is considered to be a bare license rather than a contract, it is not possible legally to impose these types of obligations on downstream users. Further to this, MIT also has one-way compatibility with GPL v 2. MIT-derived works can be incorporated under GPL, not vice versa. As detailed above, any derivative GPL v 2 content must be GNU GPL licensed (clause 2).

[172] http://www.gnu.org/licenses/license-list.html#GPLIncompatibleLicenses.
[173] J. Lovejoy, 'Understanding the Three Most Common Open Source Licenses' Open Logic; accessible via download from http://www.openlogic.com/resources-library/webinar-understanding-the-most-common-oss-licenses/

Regarding GPL v 3, Apache 2.0 has one-way compatibility with the license. Apache material can be incorporated under GPL v 3 but not vice versa. Copying and linking are broadly permitted by Apache (clause 2) but GPL v 3 requires that 'derivative' content must be licensed under GPL v 3 (clause 5). Similarly, BSD is one-way compatible with GPL v 3 i.e. BSD material can be incorporated under GPL v 3 but not vice versa. Linking is permitted by BSD but GPL v 3 requires that content must be licensed under GPL v 3 (clause 5).

Finally, MIT has one-way compatibility with GPL v 3 i.e. MIT material can be incorporated under GPL v 3, but not vice versa. Copying and linking are broadly permitted by MIT, but GPL v 3 requires that content must be licensed under GPL v 3 (clause 5).

3.6 Compatibility between the 'Weak Copyleft' licenses and the 'Strong Copyleft' licenses

Generally 'weak copyleft' licenses have one-way limited compatibility with 'strong copyleft' licenses in the arena of non-derivative 'linked' content, rather than 'derivative' content.

MPL has one-way limited compatibility with GPL v 2. Under clause 13 of MPL, a GPL can be used alongside MPL code if specifically indicated under MPL clause 13. However, GPL does not have a similar provision. Copying and linking are permitted by both licenses but both licenses have restrictive clauses with regard to modified or 'derivative' works – such works must be licensed under MPL or GPL. Moreover, since GPL v 2 refers to 'derivative works' it is likely that anything which is outside the definition of a 'derivative work' will not be affected. Furthermore, it is notable that Clause 13 MPL is first and foremost about choices made by the 'initial developer', where the latter can designate portions of the covered code as 'multi licensed'. It seems that if a person follows 'Exhibit A' in the MPL (as referred to by clause 13) that person may enable third parties to utilize parts of the code that released under MPL under other licenses. Therefore clause 13 allows alternative use of GPL in limited circumstances. As noted above, there is no compatibility for derivative works because both require derivative works to be licensed under the respective license. Similarly, LGPL v 2 has one-way compatibility with GPL v 2. LGPL-derived material can be licensed under GPL. Software that links to LGPL library is not considered to be a derivative work (clause 5). However, a derivative work under LGPL must be GPL licensed. There is also some two-way limited compatibility with regard to GPL works linking to the LGPL library.

MPL has one-way limited compatibility with GPL v 3. As noted above, under MPL in certain limited circumstances, GPL can be utilized (clause 13). It seems that if you follow 'Exhibit A' in the MPL (as referred to by clause 13) you may enable third parties to utilize parts of the code that you released under MPL under other licenses of your choice. Therefore clause 13 allows alternative use of GPL v 3 in limited circumstances. Copying and linking are permitted by both licenses but derivative works must be licensed under either MPL or GPL. As above, since GPL v 3 refers to 'derivative works' anything which is outside the definition of a 'derivative' work will not be affected. Furthermore, Clause 13 MPL is primarily concerned with choices made by the 'initial developer', where the latter can designate portions of the covered code as 'multi licensed'. There is, however, no compatibility for derivative works. LGPL has two-way compatibility via linking with GPL v 3. Copying and linking are allowed by both LGPL and GPL v 3. Derivative content must be GNU LGPL/GPL v 3 licensed (clause 2). Moreover, LGPL clause 5 states that software that links to the library is not considered a derivative work, so this also must be borne in mind.

Conclusion

This chapter has outlined and compared the terms of the various types of FOSS licenses – 'no copyleft', 'weak copyleft' and 'strong copyleft'. The possible effects of the contract/bare license debate have also been explored, along with the relevant compatibility issues. Ultimately, it is clear that while there are challenges to the legality of FOSS licenses, these challenges are not insurmountable. The diversity of licenses does create legal complexities with regard to compatibility, but given the diverse nature of FOSS programmers, the proliferation of different FOSS licenses seems inevitable.

On this point, one element in particular warrants further reflection – so far there is a relative paucity of FOSS case law concerning the issues discussed above. Given the widespread adoption of FOSS it is surprising that there are no cases on enforcement in the UK, and few in other jurisdictions such as the US, France and Germany. This in itself implies that open source programmers and users, even commercial ones, are not getting tied up in costly and time-consuming legal actions. There may be a number of reasons for the lack of cases. For example, it may be that disputes do arise, but they are largely of a minor nature and can be easily rectified before the formality of a court hearing. It may also be the case that many licenses are breached

but these breaches simply go unnoticed by FOSS licensors. The underlying DIY ethos of FOSS may also have a role to play. Nonetheless, it stands to reason that given the diversity and complexity outlined above 'hard cases' will inevitably come up in the future, and these in turn may alter the FOSS legal landscape. In particular, given the fact that FOSS thrives online in a global environment, if jurisdictional legal differences kick in over the next few years, this may have a detrimental effect on the continued 'viral' spread of FOSS globally.

Overall, there is much work yet to be done to bring clarity to the crucial enforceability and compatibility issues outlined above. All the parties involved, the FOSS developers, individual FOSS users and businesses which make use of FOSS, need guidance as to the legal ramifications of their actions. The recent EU 'Joinup' initiative, which maps license compatibility issues, is one such helpful guide; this chapter provides another.[174]

Annex I – Key of License Abbreviations used in Footnotes

A acknowledgment must be included in any redistribution

Ca if the program is interactive and such announcements are customary for similar kind of programs, copyright notice must be displayed or printed at each time the program commences operation or at the request of the user, depending on custom practice related to the kind of programs in question.

C? copyright notice must be included (but no explicit obligation to include permission notice in the source code, object code or documentation)

Cu copyright notice must remain unchanged as included in the original package (but no explicit obligation to include copyright notices in the source code, object code or documentation)

Cus copyright notice must retained in the source code as included in the original source code

Cs copyright notice must appear in all copies of the source code (but not necessarily in the documentation)

C copyright notice must appear in all copies of the source code and in the documentation (but no obligation to provide any supporting documentation with the binary code)

C+ copyright notice must appear in all copies of the source code and in the documentation that must be provided with the binary code (but not necessarily in the binary code)

Cc+ copyright notice must appear in all copies of the code, but no obligation to include it in the documentation

[174] http://joinup.ec.europa.eu/software/page/license_compatibility_and_interoperability%20

C++ copyright notice must appear in all copies and in supporting documentation

PN? permission notice (license text) must be included (but no explicit obligation to include permission notice in the source code, object code or documentation)

Pnu permission notice (license text) must remain unchanged as included in the original package (but no explicit obligation to include copyright notices in the source code, object code or documentation)

PNus permission notice must retained in the source code as included in the original source code

PN permission notice (license text) must appear in all copies of the source code

PND permission notice (license text) must appear in supporting documentation

PN+ permission notice (license text) must appear in all copies of the source code and in the documentation (but not necessarily in the binary code)

PNc+ permission notice (license text) must appear in all copies of the code, but no obligation to include it in the documentation

PN++ permission notice (license text) must appear in all copies and in supporting documentation

NA name of the copyright holder and/or of the organization which created the license may not be used in advertising without prior permission

NC name of the original program cannot be changed

NP name of the original program cannot be used in connection with any derived programs

AM altered versions must be plainly marked as such

ASC access to the source code must be provided to each recipient of the Work

AWP access to the modified Work must be provided to the public

CL copyleft clause, conditions are described. If CL appears with the name(s) of specific licenses it means that modifications must be licensed exclusively under this license or those licenses. Mere obligation to include original permission notice or license text is NOT considered as copyleft clause for purpose of this definition.

F cannot be sold, must be distributed for free