Release 0.5.2

A Practical Guide for Developers, Managers, Companies, and OS Experts

Open Source License Compendium

How to to Use Open Source Software in a Regular Manner

K. Reincke, J. Dobson, G. Sharpe, M. Kern*

24th August 2012

* This text is licensed under the Creative Commons Attribution-ShareAlike 3.0 Germany License (http://creativecommons.org/licenses/by-sa/3.0/de/): Feel free 'to share (to copy, distribute and transmit)' or 'to remix (to adapt)' it, if you '[...] distribute the resulting work under the same or similar license to this one' and if you respect how 'you must attribute the work in the manner specified by the author(s) [...]':
In an internet based reuse please mention the initial authors in a suitable manner, name their sponsor *Deutsche Telekom AG* and link it to `http://www.telekom.com`. In a paper-like reuse please insert a short hint to `http://www.telekom.com`, to the initial authors, and to their sponsor *Deutsche Telekom AG* into your preface. For normal quotations please use the scientific standard to cite.

[derived from myCsrf (= 'mind your Scholar Research Framework') ©K. Reincke CC BY 3.0 http://mycsrf.fodina.de/)]

The Open Source Community is a swarm: it is stronger than a set of some accidentally selected experts. We are proud to get its' help. Gladly we thank (in alphabetical order):

Eitan Adler,
Thomas Quiehl,
Peter Schichl,
Helene Tamer,
and all the
other...

Contents

Table 0.1: History of the Open Source License Compendium

2012-08-25	0.5.0	Expanded Break Through Release ▷ MIT license fulfilling to-do lists(Chapter ?.?) ▷ Apache license fulfilling to-do lists(Chapter ?.?) ▷ using integrated eclipse spell checking methods
2012-07-06	0.4.0	Break Through Release ▷ Open Source Use Case definition and taxonomy (Chapter 6) ▷ Open Source Use Case based general finder(Chapter 7) ▷ corresponding BSD specific mini finder(Chapter 8.2) ▷ BSD license fulfilling to-do lists(Chapter 8.2)
2012-03-22	0.2.1	▷ frameworke published as first community edition
2012-01-31	0.1.8	▷ renamed existing introduction as prolegomena ▷ inserted a shorter top-down written introduction ▷ inserted an OSLiC disclaimer
2012-01-21	0.1.7	▷ oscCopiedButNotRead.bib expanded by many verified bibliographic data ▷ list of periodicals and shortcuts added
2011-12-29	0.1.6	▷ many bibliographic data added
2011-10-17	0.1.5	▷ bibliographic data updated
2011-09-29	0.1.4	▷ Document history integrated ▷ Some Typos erased
2011-09-28	0.1.3	▷ Review of english teacher integrated
2011-09-19	0.1.2	▷ First comments of english teacher integrated
2011-09-15	0.1.1	▷ Improvements of John integrated
2011-09-12	0.1.0	▷ Introduction completed: purpose and methods

Disclaimer

This book shall be thoroughly developed - together with the Open Source Community. Finally it shall deliver reliable information, following the rule, that the swarm knows more than the single fish.

But nevertheless it can't offer more than the opinion(s) of its authors and contributors. It's only one voice of chorus discussing the topic of Open Source Licenses. For protecting the authors and contributors from charges and claims of idemnification we adopt the lightly modified GPL3 disclaimer:

THERE IS NO WARRANTY FOR THE OSLiC, TO THE EXTENT PERMITTED BY APPLICABLE LAW. THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE TEXT "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE OSLiC IS WITH YOU. SHOULD THE OSLiC PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE OSLiC AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

1 Introduction

This chapter shortly describes what the OSLiC is, how it should be used, and how it can be read. It shall be written as top-down explanation.

[TDB ...]

2 Prolegomena

This chapter describes why we need an OSLiC and how its content and form has been derivated from the need. It's written as bottom-up explanation and shall deliver deeper insights.

2.1 Why

Do we need another book about Open Source? Do *you* need another book about Open Source Software? Let us address this question from the viewpoint of what we already know, what we instinctively believe and what we may have heard. For example you may presume one or more of the following statements is correct. Or you may even have experienced similar perceptions from your peers or managers. Or you have been told they describe 'Open Source':

- The Open Source Definition offers rules to use Open Source Software.
- Modified Open Source Software must be published.
- Modified Open Source Software must be given back to the community.
- All generations of Open Source Software will remain open for ever.
- Software can either be Open Source Software or proprietary software.
- The opposite of Open Source Software is commercial software.
- Open Source Software prohibits to earn money.
- Modifications of Open Source Software must be marked explicitly.
- Modifiers of Open Source Software must identify themselves.
- When distributing an Open Source binary it's enough point to a download page to obtain the source code.
- The aim of Open Source Software is to improve the world ethically.
- Open Source Software is viral and infectious.

Do these conceptions sound familiar to you? Unfortunately, whatever we might believe or wish for, these concepts are incorrect. Naturally we will discuss this issue later on. For the moment let us assume they are indeed incorrect[1].

[1] For those who want directly verify our argumentation, we have generated a condensed summary of the arguments and citations. You can find this summary in our appendices.

So, again: Do *we* need another book about Open Source Software? *We*, that is - in this case and at least initially - the large German company *Deutsche Telekom AG*. Arguing from the perspective of a large company requires not only identifying the common misconceptions, but catering for the unique needs of a large Enterprise. And indeed the very size of the company brings its own problems.

Large companies use more Open Source Software in more varied contexts than small companies. There is an important question that every company should ask: *'Are we sure that we respect all those requirements of Open Source Software we have to respect?'*. But large companies can not answer this question as easily as small companies: the large number of diverse Open Source deployments in different contexts mean that case by case governance, a model that may work in small concerns, is far from appropriate for our needs. This leads to wasting both time and money. Further, the chances of success are small: training at least one employee in each software team as an Open Source Software License expert is unrealistic in terms of cost-efficiency and reliability.

Nevertheless even large companies want to and try to fulfill the rules of Open Source Software thoroughly - especially *Deutsche Telekom AG*. When this company realized that the question *Are we sure that we respect all those rules of Open Source Software correctly which we have to respect* could be problematic, it directly asked some of its' employees who were known as Open Source enthusiasts - to establish a service and a process for answering this question.

So, it is no surprising that we, the initial authors of this *Open Source License Compendium*, were asked by our employer *Deutsche Telekom AG*. Naturally we were proud to work on an Open Source topic officially. But while we were doing our job we had to ask ourselves if *we* perhaps needed another book on Open Source. Our answer was *Yes, we do!* Let us shortly explain, why:

Firstly, we already knew that there exist supporting software. These meta-programs take the code of any other application and try to list those Open Source Software being 'covered' by that application[2]. But we had also already realised that this supporting software did not always match the way we thought the problem should be solved. Secondly we recognized fairly quickly that we need a reliable guide. We personally were asked to give the *ok* for projects of our company. We could not answer such requests on the base of *'Oh yes, I read this in the* Heise-Ticker *a few days ago'* - even if the *Heise-Ticker* had described the situation completely correctly. We ourselves had to be reliable than this. Naturally we already knew a great deal about Open Source Software. Even so, our knowledge was not as systematic as necessary. We looked for an Open Source Compendium which adequately described what a project or product development team had to do to fulfill the criteria of its Open Source Licenses. We wanted to

[2] As general examples let us mention Palamida (`http://www.palamida.com/`) and BlackDuck (`http://www.blackducksoftware.com/`).

use that compendium to the basis of our recommendations.

We were very thorough but we did not find what we were looking for. Our 'little' bibliography attest our seriousness. What we found was a lot of information releating to individual issues spread over many sources. We did not find answers for our question even in the specific literature. Let us describe three little steps to increase the understanding of the issue:

Without Open Source Licenses there is no Open Source movement. Nevertheless in dealing with Open Source Licenses, this is sometimes neglected. Take the *Apache Web Server* as an example: No doubt, it's one of the most important pieces of Open Source Software[3] with a specific license[4]. Moreover: the success of the Open Source movement in the commercial world depends directly on the decision of IBM to replace its corresponding own component in the *IBM WebSphere Application Server* with the free *Apache Web Server*[5]. Meanwhile many companies use the *Apache Web Server* to act as a web provider. Currently the *Apache http server* - as it has to be named correctly - is used more than twice as much as all the other http server software together[6]. Hence many business models depends on the Apache License. Another aspect is that even the famous *Apache Cookbook*, which explains the installation, the configuration and the maintaining of an Apache Web Server in details[7], does not mention anything about the license which allows for installation, configuration and maintenance. Neither the index

[3] To prove that the *Apache* is really a piece of Open Source Software one must execute a set of steps: Firstly you have to note, that *Apache* is something like a meta project, covered by the *Apache Software Foundation*, also known as *ASF* (cf. `http://www.apache.org/`, wp.). Therefor you can not directly jump into the *Apache License*. First of all you have to visit the project site (cf. `http://httpd.apache.org/`, wp.) even if at the end its' license link leads you back to the general *Apache License sub site* (cf. `http://www.apache.org/licenses/`, wp.) which announces, that 'all software produced by The Apache Software Foundation or any of its projects or subjects is licensed according to the terms of the documents listed below'. Only now you can use the offered link for switching to the *Apache License*, Version 2.0, if you want to check your rights and duties. But that is difficult. There does not exist any simple list what you have to do for fulfilling the license. Even the faq (cf. `http://httpd.apache.org/docs/2.2/faq/`, wp.) - meanwhile being moved to a wiki - only says that the server '[...] comes with an unrestrictive license' and that you are allowed to put the code on a CD (cf. `http://wiki.apache.org/httpd/FAQ`, wp.). Hence, from the viewpoint of the ASF the license itself shall answer all questions. [Reference download for all urls: 2011-08-31]

[4] cf. *Apache Software Foundation*: Apache License, Version 2.0; 2004 ⟨URL: `http://www.apache.org/licenses/LICENSE-2.0`⟩ – reference download: 2011-08-31, wp..

[5] cf. *Moody, Glyn*: Die Software-Rebellen. Die Erfolgsstory von Linus Torvalds und Linux; transl. from the American [edition, 2000] by Annemarie Pumpering; Landsberg am Lech: verlag moderne industrie, 2001, ISBN 3–478–38730–2, pp. 287ff.

[6] cf. *Netcraft*: August 2011 Web Server Survey; 2011 ⟨URL: `http://news.netcraft.com/archives/2011/08/05/august-2011-web-server-survey-3.html`⟩ – reference download: 2011-08-31, wp.

[7] cf. *Coar, Ken* a. *Rich Bowen*: Apache Kochbuch; deutsche Übersetzung v. Jochen Wiedmann; Beijing [...]: O'Reilly, 2004, ISBN 3–89721–371–0, et passim.

lists the word 'license'[8], nor the chapters 'Installation'[9] or the chapter 'Miscellaneous'[10] mentions the license question in a serious way. There's only one short hint as to the advantage of Open Source Software, i.e. that everybody is allowed to install it[11]. Can you be sure that you are allowed to do what you are doing on the base of such a phrase?

Naturally, the *Apache Cookbook* is not a book for lawyers, it's a book for administrators and developers, They do not want to get bogged down by legalities, they want to set up an Apache Web Server as fast as possible and get down to work. Indeed, the Apache Cookbook offers a good support. But not only as a company you have to ask yourself whether you are really allowed to do what you are doing. Can you find the answer in the *Apache Cookbook*? No. Can you find it in the license itself? Yes, but it is difficult[12]. So again: Can you find your answer in another book, which is *Amazon's* current top recommendation for the request *'apache server'*[13]? Not really: Sascha Kersken's Apache 2.2 Handbook offers a license chapter, but only two pages long[14]. Moreover the rights and duties are condensed into just 5 bullet points which taken together do not explain when the software and the license has to be handed over to a customer and when you are allowed to hide your improvements[15].

This brings us to the question of what prevents us from using something like a *'general license cookbook'* which explains all the necessary details and which offers quick access to the relevant points:

Of course we also browsed the internet. At least for German speaking people there is an excellent site concerning the topic *Open Source Licenses.* offered by *iffross*, which, loosely translated, means an *Institute for Legal Aspects of the Free and Open Source Software*[16], founded in 2000 as a private institute to tracke the phenomenon 'free software' from the viewpoint of (German) lawyers[17]. Besides many other aspects this site offers a very well and thoroughly elaborated FAQ[18]

[8] cf. *Coar a. Bowen*: Apache Kochbuch, 2004, pp. 245ff, esp. p. 250.

[9] cf. id., l.c., pp. 1ff.

[10] cf. id., l.c., pp. 219ff.

[11] cf. id., l.c., pp. 1: ' ... einer der Vorzüge von Open Source Software besteht darin, dass jedermann die Erlaubnis zur Erzeugung eines eigenen Installationskits hat '.

[12] And do we really want our developers and maintainers to read the original licenses? Do we really want them to discover that they also have to check the licenses of the used modules?

[13] Tested on `http://www.amazon.de/` at 2011-08-31.

[14] cf. *Kersken, Sasche*: Apache 2.2. Das umfassende Handbuch; 3rd, refreshed a. expanded edition; Bonn: Galileo Press, 2009, ISBN 978–8362–1325–7, pp. 111f.

[15] cf. id., l.c., p. 112.

[16] originally: 'Institut für Rechtsfragen der Freien und Open Source Software'. Main entry point for its' site is the URL `http://www.ifross.org/`.

[17] cf. *ifross*: Ziele, Aufgaben, Geschichte; 2011 ⟨URL: `http://www.ifross.org/node/16`⟩ – reference download: 2011-09-05, wp.

[18] cf. *ifross*: FAQ; 2011 ⟨URL: `http://www.ifross.org/faq-haeufig-gestellte-fragen`⟩ – reference download: 2011-09-05, wp.

and a large list of Open Source Licenses and other related licenses: moreover, evidently it is classifying the Open Source Licenses in those 'without copyleft-effect' (BSD), in those with 'strict copyleft-effect' (GPL)) and in those with 'restricted copyleft-effect' (LGPL)[19].

However, even this excellent site does not fulfill our needs. It does not offer those context specific to-do lists which companies, developers or project managers can use to ensure their Open Source Software is used in a regular manner.

We therefore evaluated that standard book which is listed in the most legal bibliographies[20]: the book of Jaeger and Metzger which concerns - loosely translated - *the judicial framework requirement for Open Source Software*[21]. Even the most earliest edition of this book already had a clear structure in its' chapter 'copyright': For each license mentioned (or at least for each license cluster) it offered a subchapter for the rights and a subchapter for the duties[22] of the software user[23]. Many other important aspects of the topic *Open Source* are discussed, too[24].

But we needed more than this. Despite the quality of the book we were certain that we could not hand over this book to our programmers with the recommendation *check your touched licenses and follow the instructions of the relevant subchapters...*. This book did not contain simply checkable to-do lists, either in the first edition[25] and in the second edition[26] or in the recently published third edition[27]. So, how can a company or a developer or a project manager be sure of fulfilling the requirements of the Open Source Licenses sufficiently if he/she does not have a verified list telling him *'do this' and 'in case of that, do that', and 'then do this'*? Why should he himself implicitly become an Open Source Licenses expert which has to extract the necessary steps out of the literature?

While we were searching for an existing Open Source compendium we found an article with the title 'Compendium for the Publication of Open Source Soft-

[19] cf. *ifross*: Lizenz-Center; 2011 ⟨URL: `http://www.ifross.org/lizenz-center`⟩ – reference download: 2011-09-05, wp.

[20] at least in that German judicial literature dealing with Open Source

[21] cf. *Jaeger, Till* a. *Axel Metzger*: Open Source Software. Rechtliche Rahmenbedingungen der Freien Software; 1st edition. München: Verlag C.H. Beck, 2002, ISBN 3406484026, pp. V - It can not be any surprise that both authors, Mr. Jaeger and Mr. Metzger are members of ifross (cf. `http://www.ifross.org/personen/`, wp.).

[22] cf. id., l.c., pp. 30ff.

[23] For getting a good survey of the structure and the line of thought see the contents cf. id., l.c., pp. VIIIf.

[24] pars pro toto: have a look at the chapter concerning the liability: cf. id., l.c., pp. 137ff.

[25] cf. id., l.c., pp. VIff.

[26] cf. *Jaeger, Till* a. *Axel Metzger*: Open Source Software. Rechtliche Rahmenbedingungen der Freien Software; 2nd edition. München: Verlag C.H. Beck, 2006, ISBN 3406538037, pp. VIIff.

[27] cf. *Jaeger, Till* a. *Axel Metzger*: Open Source Software. Rechtliche Rahmenbedingungen der Freien Software; 3rd edition. München: Verlag C.H. Beck, 2011, pp. VIIIff. Naturally we use this latest edition for adopting or discussing systematical aspects.

ware'[28]. It aims to be a 'pragmatic guidebook' and an 'assistance' for 'publishing software under the conditions of an Open Source License'[29]. Moreover, at the end of this article its' authors formulate ambitiously that their 'guide' should be carried out, section by section - for getting a legally water tight process of publishing Open Source software[30].

The authors of this article describe something close to what we were looking for. Indeed, the article lists important aspects which have to be taken in consideration if you want to deal Open Source Software correctly: It announces that no obligation exists to publish code either if you embed GPL code into your proprietary code or if you modify the GPL code. It is only if you hand over your binary to other persons that you have to distribute the code too, but only to them and not to the general public[31]. Additionally the articles explains exactly that software - at least in Germany - can only be acknowledged as Open Source Software by transferring the rights to use - the 'Nutzungsrechte' - to other people, while the copyright itself - the 'Urheberpersönlichkeitsrecht' - is not transferable and belongs to the author[32]. Moreover, besides other aspects the articles discusses briefly and deeply the problem of the No-Warranty-Clauses which are not valid in Germany and which will therefore automatically be replaced by the liability rules for a donation[33]. And last but not least this article actually summarizes the idea of Copyleft and the differences between LGPL and GPL[34].

However some gaps remain. The article does not analyze in which cases a University or a company perhaps *must* publish its' developments based upon Open Source Software. It does not discern between different licenses and conditions. It also does not discuss what Universities or companies, which (re-)use and/or distribute Open Source Software (internally), must do to fulfill the touched Open Source Licenses. And finally this article does not offer the step by step list as promised.

We did, however, feel supported by this article, in two ways. Firstly it was a well written summary of some main problems. Secondly it stated the necessity to have a compendium for being able to establish a legally 'water-tight' process of

[28] approximately translated

[29] cf. *Bretschneider, Ulrich*, *Rainer Glaschick*, a. *Gernot Gräfe*: Ratgeber für die Veröffentlichung von Open-Source-Software durch eine Hochschule; In *Asche* et al.: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 166f (originally: ein 'pragmatischer Ratgeber' zur 'Veröffentlichung einer Software unter den Rahmenbedingungen einer Open-Source-Lizenz').

[30] cf. id., l.c., pp. 186 (originally: ein 'Ratgeber', der es erlaubt ' (...) die zu berücksichtigende Aspekte (strukturiert abzuarbeiten) (...) ' und einen 'rechtlich nicht angreifbaren Veröffentlichungsprozess' zu ermöglichen).

[31] cf. id., l.c., pp. 170 and 181.

[32] cf. id., l.c., p. 173.

[33] cf. id., l.c., p. 177.

[34] cf. id., l.c., p. 181.

publishing Open Source software[35]. We seemed to be justified in our assumptions. But the Open Source Compendium we were looking for had to be more practical, more processable, more distinguishing and more elaborated.

So again: Did we need a new book about Open Source Software? We had looked for a reliable integrated Open Source Compendium. But we found separate pieces of information and - as we know today - some rumors. Our answer was clear: naturally we did not need a new general book about Open Source. But what was lacking was a description what responsible developers, project managers or product developers require to fulfill Open Source Licenses. We needed an *Open Source License Compendium*.

At the best such an *Open Source License Compendium* would contain a set of simply to process *'For-Fulfilling-The-License-To-Do-Lists'*. Additionally it should offer an intuitively user-friendly search option for these lists. In any case, it should share developers and project managers the effort of having to become Open Source License experts. For the other users, it should also clearly explain why one has to do this and not that. Hence a reliable *Open Source License Compendium* should not only list what one has to do, but should offer both, thoroughly verified reliable details and clearly condensed guidance.

Although we did not find such an Open Source Compendium we were familiar with the spirit of the Open Source Community. Hence we followed one of its' most simple rules: *'what you miss you must develop ion your own'*. Some principles should help us to achieve our targets:

To-do lists as the core, discussions around them : Our work should be split into two parts. As it core we wanted to offer a set of To-Do-Lists. Each of these lists should be relevant to one specific Open Source License and should be clustered by the Open Source specific use cases. Around this all those aspects of Open Source Software which influence the interpretation of the licenses and the rules core should be precisely characterized. Nevertheless, the users should be able to skip details and go directly to the section they require.

Quotations with thoroughly specified sources : Even if our users should not be obliged to read every part of the compendium they should not be required to believe us without question. We wanted to be revisable. Because our sources and our conclusions should be easily verifiable, we decided to use the academic citations and list bibliographic data extensively on the basis that our task should be to collect information, not to invent new 'facts'.

Not the internet alone, also books and articles : We wanted to go back to the originals even if the internet was full of more or less modified copies.

[35] cf. *Bretschneider, Glaschick, a. Gräfe*: Ratgeber für die Veröffentlichung von Open-Source-Software durch eine Hochschule, 2008a, p. 186.

We wished to get reliable facts and descriptions. Therefore we decided to evaluate not only the internet but also scientific sources - for example - offered by university libraries.

Not clearing out the forest land, but cutting out a swathe : Even if we had to deal with licenses and their legal aspects we did not want to get lost in detailed discussions. It should not be our task to find out whether a specific kind of handling would still be legal or already forbidden. We did not want to fight against the licenses. We did not want to stretch their ambit or to test their boundary. We wished to accept Open Source Licenses as they are: rules written from developers for developers. And even if some parts of these licenses would not be valid with respect to a legal system[36]), we wanted to take them as our guideline - at least while they do not violate more general laws[37]. We simply wanted to *find one proven way* to cross the maybe slightly unsure forest of Open Source Licenses. Even if indeed some clauses of the licenses finally were not enforceable against us we wanted to respect them 'voluntarily'. We wanted to deliver a set of rules which support users and remove the possibility of becoming involved in license disputes with Open Source developers or the Free Software Foundation.

Take the text seriously : On the other side we wanted to take our license texts as they were. If they lacked anything[38], we would interpret the open issues in the spirit of the Open Source idea. But where the text was clear and

[36] And indeed for example for the GPL one can argue in this way: Even if you take the GPL as a contract of the type 'donation' respectively 'Schenkung', it is presented in the form of AGBs respectively 'Allgemeine Geschäftsbedingungen' and must therefore follow the general AGB rules.'Regrettably' in Germany these general AGB rules do not allow to exclude each type of warranty. If we follow Oberhem, §11 and §12 of the GPL must be invalid in Germany because of these general AGB rules. Moreover, for Oberhem even §5 - the important clause of the GPL by which you can only get the right to use and to distribute GPL software if you respect the rules of the GPL - seems also to be invalid respectively 'unwirksam'. But the good message is that the GPL as whole is not invalid even if it contains invalid clauses. *Oberhem, Carolina*: Vertrags- und Haftungsfragen beim Vertrieb von Open Source Software; Dissertation; Hamburg: Verlag Dr. Kovač, 2008 (= Recht der Neuen Medien, [Vol./No.] 50), ISBN 978–3–8300–4075–0, pp. 128, 133ff, 150ff, esp. 146, 159.

[37] what they clearly do not do!

[38] The systematical underdetermination of licenses is a problem being also known in the Open Source respectively Free Software movement. Following the biography of RMS his main judicial counselor Moglen has stated, that 'there is uncertainty in every legal process (...) ' and that it seemed to be silly to try '(...) to take out all the bugs (...)'. Nevertheless - so Moglen resp. Williams - the goal of Richard Stallman was 'the complete opposite': He tried '(...) to remove uncertainty which is inherently impossible'. But - and that's the nub of this analysis - Moglen had to follow Stallmann because of RMS character. And he had to summarize their work so, that '(...) the resulting elegance (of the GPL; KR.), the resulting simplicity (of the GPL; KR.) in design almost achieves what it has to achieve'. Hence we are asked to take the license texts themselves seriously. cf. *Williams, Sam*: Free as in Freedom. Richard Stallman's Crusade for Free Software; Beijing [... etc.]: O'Reilly, 2002, ISBN 0–596–00287–4, pp. 177f.

definite we wanted to take its propositions as a definite decision - even if that meaning stood against well known Open Source 'facts'.

Trust the swarm : We did not want to use our own research alone as a basis. We knew that the swarm is ever stronger than a set of some randomly selected experts. Therefore we decided to publish our text as a still unfinished work, starting with an early release 0.2. And then we wanted to invite the community to complete the compendium together with us. We would elaborate our Open Source Compendium as a set of LaTeX- and BibTeX files which could be developed and managed in GIT or any other version control system. And finally we would publish our text under a Creative Commons Attribution-Share Alike German 3.0 license[39] , to allow other people to correct us, to help us or even to take our results for their own purposes.

And so we did. Here is the result. Feel free to use it - according to our licensing.

2.2 What

Now we can briefly explain how you can use this compendium:

Open Source: Idea and Concepts :- Here you will find background information to help you interpret Open Source Licenses in the sense of the *Free Software*

[39] This text is licensed under the Creative Commons Attribution-ShareAlike 3.0 Germany License (http://creativecommons.org/licenses/by-sa/3.0/de/): Feel free 'to share (to copy, distribute and transmit)' or 'to remix (to adapt)' it, if you '[...] distribute the resulting work under the same or similar license to this one' and if you respect how 'you must attribute the work in the manner specified by the author(s) [...]'):
In an internet based reuse please mention the initial authors in a suitable manner, name their sponsor *Deutsche Telekom AG* and link it to `http://www.telekom.com`. In a paper-like reuse please insert a short hint to `http://www.telekom.com`, to the initial authors, and to their sponsor *Deutsche Telekom AG* into your preface. For normal quotations please use the scientific standard to cite.
[derived from myCsrf (= 'mind your Scholar Research Framework') ©K. Reincke CC BY 3.0 http://mycsrf.fodina.de/)]

movement[40], the *Open Source Software movement*[41] or the GNU-Project[42]. If you are familiar with the evolution of the Open Source Initiative, with the character of the Open Source Definition as a set of necessary but insufficient criteria and if you know about the history and meaning of free software as older and broader concept then you can ignore this chapter.

The Same Idea, Different Licenses :- In this chapter we discuss different ways to cluster Open Source Licenses. Finally we present our own taxonomy based on the labels 'protecting the developer', 'protecting the licensed code' and 'protecting the on-top-developments'. If you are familiar with the methods of grouping different Open Source Licenses and particular if you know that you can not authorize your doings on the base of descriptions of such license groups than it's enough, in order to understand our line of thought, to briefly note our taxonomy and its wording.

The Problem of Derivated Works :- This chapter is important. In the spirit of software developer we try to explain which kinds of programming evoke a derivated work and which not. Our to-do lists will refer to this analysis.

The Problem of Combining Different Licenses :- You should not ignore this chapter. We will explain why and how combining software of different licenses is not as dangerous as it's often told. The results of this chapter influence the structure of our to-do lists.

Open Source Software and Money :- Here we will shortly discuss ways in which money is no problem. If you already know that it is only prohibited to require payment for the act of licensing a piece of Open Source Software to second or third parties and if you already know that this is only forbidden by some licenses, and not by all, than you can postpone the reading of this chapter.

The Problem of Implicitly Freeing Patents :- Here we will illuminate some aspects of software patents and how the are handled by some Open Source

[40] At least at this place you are perhaps expecting that we use the logograms FLOSS, F/OSS, F/LOSS, or whatever. As you will read later on the word *Free* is ambiguous and has strained the use of the concept *Free Software*. Later on we will also talk about the invention of the concept *Open Source* designed as a 'replacement' and acting as a 'splitter'. The mentioned logograms are introduced to re-establish or - at least - to underline the common history and the common center of 'both' movements, whereby the word *Libre* shall resolve the ambiguity of the word *Free*. For a first survey cf. *wikipedia (en)*: Free and open source software; n.l., 2011 ⟨URL: `http://en.wikipedia.org/wiki/Free_and_open_source_software`⟩ – reference download: 2011-09-08, wp..

[41] For another brief and informative introduction cf. *Fogel, Karl*: Producing Open Source Software; How to Run a Successful Free Software Project; Beijing, Cambridge, Köln [...]: O'Reilly, 2006, ISBN 978–0–596–00759–1, pp. 231ff esp. p. 232f..

[42] We ourselves will stay with the concept *Open Source* because the OSD specifies the scope of our analysis. But we do it with a deep obeisance to Stallmann and the FSF - even if we know that this will not protect us from the thunderbolt of RMS.

Licenses. You should know what licenses implicitly do with your patents. But it's not our intention to write a software patent compendium.

Open Source: Use Cases as Principle of Classification :- This is an important chapter. We explain our categories 'Use as it is', 'Modify the Code', 'With Redistribution', 'Without Redistribution', 'Isolated Initial Development', 'On-Top-Development': we develop and discuss our taxonomy with respect to the side effects of 'combining different licenses' and 'generating derivated works'. This taxonomy will determine the following chapters.

Open Source Licenses: Find Your Specific To-do Lists :- This is a kind of summary which joins the relevant aspects and elaborates the 'finder for your to-do lists'. This is that chapter which you probably will reuse multiply, even if you do not want to read any of our explanations.

Open Source License Fulfillment: Classified To-do Lists :- This chapter offers all classified to-do lists. The structure of its' subchapters will match the structure of our finder and the structure of our taxonomy.

Open Source Licenses and Their Legal Environments :- Here we discuss that using Open Source Software in a regular manner is not only a question of the licenses themselves but of the kind of the surrounding legal system.

Appendices: Some Widespread Open Source Myths :- Here we make good on our promise to explain why all the propositions mentioned at the beginning of this chapter are wrong. You might read this chapter as a special introduction or a reminder epilogue whenever you want to do.

A final remark: We have already characterized the tone of our footnotes. Let us now briefly explain a little peculiarity of our bibliography:

Modern times have also changed the humanities. Formerly a book or an article must be printed for being ripe to be quoted. Our statements relied on static, readily prepared works. Nowadays even university libraries sometimes offer those books and articles as PDF files which are printed in the original. As a scholar, now you must rely on the equality of the printed version and the PDF file - at least with respect to the page numbers and the appearance. You can not verify the equivalence - at least to a certain degree.

Moreover: in case of such 'e-books' and 'e-articles' the libraries often do not offer the pdf files themselves but links to the download pages of the publisher. Formerly as a scholar you could trust that your readers would be able to retrieve the quoted work if they want to verify your citations. It's one task of our libraries to hold available our scientific sources. But now they do not buy any longer the books, but the right to download files over the university net. In this case these PDF files are not stored on the serves of the university library. By using the link provided by the publisher each student or each reader downloads his own file -

case by case. Therefore - as a scholar - you now have to trust that the publisher, who provides the link, will not change that pdf file that you have cited.

But it gets even worse: While it might be that publishers modify their work secretly (even it is not very likely that they do it), it's a definite feature of the web that its' pages are frequently changed. Hence we must ask ourselves: Can we seriously argue on the basis of statements and documents which might disappear? Can we quote such possibly volatile sources? The problem is: we must do it, especially if we write about an internet topic - and even if we want to write a really reliable compendium.

So, what can we do? Firstly we must confide in our readers, that they either will retrieve our sources or - if they can not find them - that they believe that we really have found and read what we have written and quoted. Secondly we store all these e-wares[43] we read[44]. And thirdly we should lay open to our readers the different levels of reliableness of our sources. Therefore we use the following markers in our bibliographic data[45]:

- Print / Copy:- The source is printed and we saw either the printed work really or we get an official copy by our library. Hence you should also be able to get the work in a library, at least in those we used (UB Frankfurt or ULB Darmstadt).
- BibWeb/[PDF/...] :- The source might be printed, but we read only the electronic version (PDF or other type of format), offered by and over the net of our university libraries (UB Frankfurt or ULB Darmstadt).
- FreeWeb/[PDF/...] :- We read the electronic version offered by the free web. In this case we add the url[46] and the date when we downloaded / saw the text.

[43] Take this little word as (new) generalization of 'e-book', 'e-article', 'e-paper' and so on.

[44] But because of the copyright we ourselves are naturally not allowed to offer a download link for them or to send a copy of it to those who want to verify our quotes.

[45] And another hint: Nowadays sometimes even scientific libraries doesn't offer exact 'e-copies' of the original. In some cases one can get only html-versions of articles which formerly were printed as part of journals. In these case the scholar has to use sources which lost their original page-numbers. The same can happen to articles of proceedings etc. which are now only offered as autonomous pdf files with an internal paging. If we quote such kind of articles we try to specify the number of the quoted article in the original row of articles, added - if possible - by an internal page number. But naturally we also try to follow the bibliographic data delivered by that organization which distributes these kind of copies.

[46] Please note: Long urls often destroy the pleasing appearance of a text because it's difficult to wrap the lines acceptably. Hence we wished to make it easier for LaTeX to do this job. Therefor we sometime split the urls and inserted blanks. So you have to erase all blanks if you want to verify our urls.

3 Open Source: Idea and Concepts

In this chapter we discuss the meaning of Open Source *and the common features of all Open Source Software. We scan historical and conceptual aspects. It's the chapter of background knowledge. At the end you will know that software is only Open Source Software if you as customer gets the right to use, to modify, and to redistribute the code without any limitations, but with some obligations. And these obligations implicitly refer to different historical contexts which might influence the understanding of your licenses.*

[TDB ...]

3.1 Open Source, OSI, and OSD

Here we describe the meaning of Open Source. At the end of this chapter you should know that Open Source Software is defined by a set of necessary criteria which together determine the common basic features of Open Source Licenses. Additionally you will have understood that the opposite of Open Source Software can not only by defined ex negativo. But you should also know that these features can differently be implemented. Therefore the OSD can not be read as a set of rules describing what we have to do if we want to fulfill the Open Source Licenses. You should know that you have to go back to the license itself.

...

3.2 Open Source and its history: some hints

Here we present main lines of the Open Source genesis: The start with the bundling of hardware and software in the beginning on the one side and the monopole of AT&T and the free distribution of unix in the universities on the other side - which together established the free hacker culture. We will shortly describe the increase of the value of software evoked by the IBM unbundling strategy and the antitrust suit against AT&T which let become the software an object of a specific which had to be protected and which destroyed the free exchange within the early hacker community. Naturally we will illuminate the answer of RMS, the GNU project, the founding of the FSF and the GPL. Then we will highlight the introduction of the concept Open Source invented for dissolving the troubles to talk about Free Software with managers of companies. We will hint to the Linux kernel as an unwelcome completion of the GNU system. Finally we will outline the convergency of business and Open Source, not only by Netscape/Mozilla, IBM, Apache, Redhat, SUN/OpenOffice but also by IBM/eclipse, Sun/Java and so on. And naturally we will highlight the meaning of 'the Cathedral and the Bazar', which had not been

written to contrast the working style of the Open Source Development and the proprietary 'in company' development by for example Microsoft, but for discern the working and leading style of RMS and Linus.

...

3.3 Free Software versus Open Source: some hints

Here we will illuminate the differences between the ideas of the Free Software Movement and the Open Source Movement.

...

4 The Same Idea, Different Models of License

In this chapter we describe different license models which fulfill the common idea. We discusss existing types of grouping single Open Source Licenses. We highlight the limits of building such clusters for being able to analyse prototypic licenses. But finally for learning the field we ourselves cluster these license models in three groups, the Minimal Prescribing Licensemodels for protecting the developer like MIT, BSD or Apache, the Reflexive Prescribing Licensemodels for protecting the licensed code like LGPL, EPL? or EUPL and the 'viral' Overlapping Prescribing Licensemodels for protecting the on-top-developments. At the end you will know the main obligations in generally.

[TDB ...]

Be careful: Linux Magazine (04/05 2011) uses another taxonomy which seems not to be as elaborated as this taxonomy. This must be discussed[...] ...

4.1 Minimal Prescribing Licensemodels: Protecting the developer

Here we discuss licenses which only try to protect the developer, like the licenses BSD, MIT *and* Apache.

...

4.2 Reflexive Prescribing Licensemodels: Protecting the code

Here we discuss licenses which try to protect the developer and to protect the licensed code, like the licenses EPL?, EUPL? *and* LGPL

...

4.3 Overlapping Prescribing Licensemodels: Protecting the on-top-developments

Here we discuss licenses which try to protect the developer, which try to protect the licensed code, and which try to protect the on-top-developments, also known as derivated works. You may expect the licenses GPL-2, GPL-3, *and* AGPL...

...

4.3.1 Excursion: GPL is not the Evil

Here we simply want to underline that licenses protecting the on-top-developments does not 'infect' other proprietary software automatically. It's not an aspect of the code of the license, but of the acts of the combining developers or managers. It's their decision if they have to fulfill the GPL.

...

4.3.2 Excursion: The Peculiarity of GPL-3

In the same sense we want to explain how the GPL-3 only upholdes former ideas by transposing them to new technical trends.

...

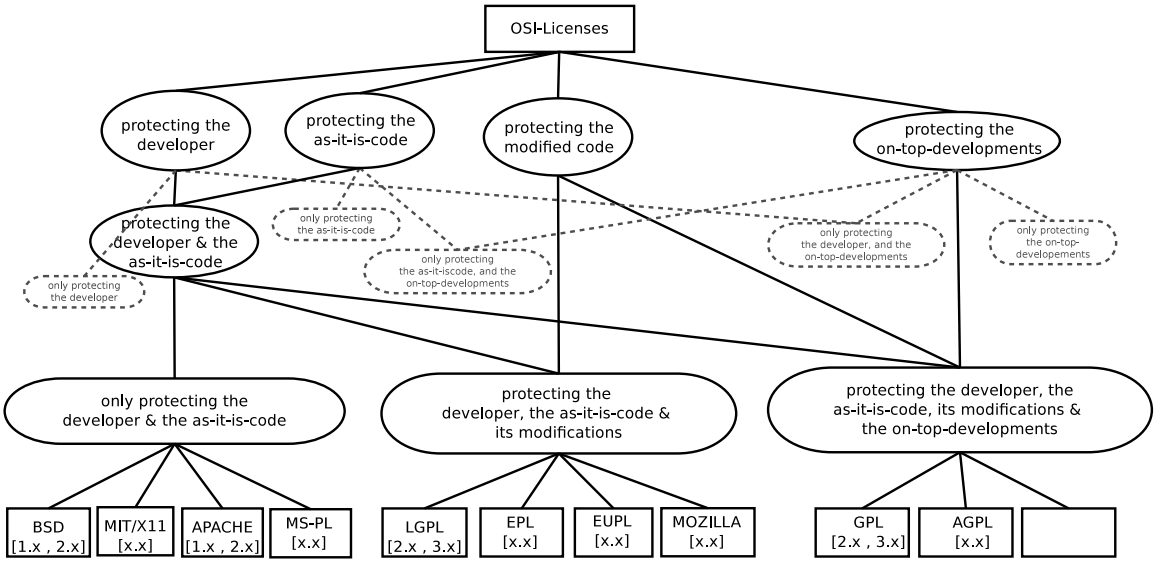
4.4 Other Models

...Do they exist? ...

4.5 Summary: Our Taxonomy for the analyzed Licenses

One pictures and the hint, that the criteria can be implemented differently by different licenses and that one therefore has to analyze the single licenses nevertheless.

...



5 Open Source: Important Minor Points

This chapter we shortly discusses some minor but although important issues.

[TDB . . .]

5.1 Excursion: The Problem of Derivated Works

We will shortly discuss existing attempts to define the derivated works of technical aspects, like dynamical or statical linking or not. We will prove that linking can not deliver a definite criteria: 1) modules are only unzipped libraries. 2) you can distribute software as modules added by a script, which statically(sic!) links all modules before executing the program. 3) The criteria of pipe-communication is good, but not sufficient. 4) All these attempts do not match the constituting features of script languages. Therefore we will follow Moglen(?) and will argue from the viewpoint of a developer: it's only a question of a function, method or anything else which calls (jumps into) a piece of code which has been licensed by a license protecting on-top-developments and you have a derivated work.

. . .

5.2 Excursion: The Problem of Combining Different Licensemodels

Here we discuss the often neglected or only loosely touched problem of combining differently licensed software. We will hint to the Exclusion-List of the Free software foundation; we will hint to the eclipse / GPL-plugin problem; we will mention the recent discussion whether the kernel requires to license the complete Android as GPL; and finally we will discuss the just now published, short analysis of Jaeger and Metzger presenting a combining matrix which seems to fall into their lap. We ourselves will argue that question can simply be answered: only if you embed two libraries which both are licensed by an on on-top-development protecting license and if these both license require the licensing of the derivated work by different licenses then you have a problem. In all other case which we will describe an list there is no problem.

. . .

5.3 Excursion: Open Source Software and Money

Here we will shortly discuss ways in which money and Open Source is no problem.

...

5.4 Excursion: The Problem of Implicitly Freeing Patents

Here we analyze the implicitly freeing of patents by licensing code as Open Source Software.

...

6 Open Source Use Cases: Concept and Taxonomy

This chapter establishes our concept of Open Source Use Cases *as a classification system for to-do lists. The conditions of a specific license, in the context of a particular* Open Source Use Case*, will be fulfilled by following the corresponding to-do list. Additionally this chapter introduces a taxonomy for these* Open Source Use Cases*. Later on, this taxonomy will organize the* Open Source Use Case Finder*.*

After all these introductory remarks, we can summarize our idea. We know that the right to use Open Source Software depends on the tasks required by the Open Source Licenses. As opposed to commercial licenses, you can not buy the right to use a piece of Open Source Software using money. It is embedded into the Open Source Definition that the right to use the software may not be sold. The OSD states firstly that an Open Source License may '[...] not restrict any party from selling or giving away the software as a component of (any) aggregate software distribution', and adds secondly in the same context that an Open Source License '[...] shall not require a royalty or other fee for such sale'[47].

However, it would be wrong to conclude that you are automatically allowed to use Open Source Software without any service in return: generally you have to do something to gain the right to use the software. In other words: Open Source Software is covered by the idea of 'paying by doing'. As such, Open Source Licenses describe specific circumstances under which the user must execute some tasks in order to be compliant with the licenses. So, if we want to offer to-do lists for fulfilling license conditions, we must consider these tasks and circumstances.

In practice, such circumstances are not linear and simple. They contain combinations of (sometimes context sensitive) conditions, which can be grouped together into classes of tokens. Such a class of tokens might denote a feature of the software itself - such as being an application or a library. Or it can refer to the circumstances of using the software, such as 'using the software only for yourself' or 'distributing the software also to third parties'.

At the end, we want to determine a set of specific OSUCs - the *Open Source Use Cases*. And we want to deliver for each of these OSUCs and for each of the considered Open Source Licenses one list of actions, which fulfills the license in that context[48].

[47] cf. *Open Source Initiative*: The Open Source Definition; 2012 [Year n.st] ⟨URL: `http://www.opensource.org/docs/osd`⟩ – reference download: 2012-06-21, wp. §1.

[48] Fortunately, sometimes one task list fulfills the conditions of more than one use case - a welcome reduction of complexity

Such an *Open Source Use Case* shall be considered as a set of tokens describing the circumstances of a specific usage. Hence, to begin, we must specify the relevant classes of tokens, before we can determine the valid combinations of these tokens - our *Open Source Use Cases*. Finally, based on the tokens, we generate a taxonomy in form of a tree. This tree will become the base of the *Open Source Use Case Finder* which will be offered by the next chapter, and which leads you to your specific OSUC by evaluating just a few questions and answers.

There are only a handful of tokens which are relevant to the circumstances of Open Source Software Licenses:

- The **type of the Open Source Software**: On one hand there are code snippets, modules, libraries and plugins, and on the other hand, autonomous applications, programs and servers. We'll coin the word 'snimolis' for the first set, and 'proapses' for the second. This is necessary, as we are not only talking about libraries and applications in the everyday sense, but rather in the broadest sense[49]. More specifically, we will ask you, whether the Open Source Software, you want to use, is an includable code snippet, a linkable module or library, or a loadable plugin, or whether it is an autonomous application or server which can be executed or processed. In the first case, the answer should be 'it's a snimoli', in the second 'it's a proapse'.
- The **state of the of the Open Source Software**: It might be used, as one has got it. Or it can be modified, before being used. More specifically, we will ask you, whether you want to leave the Open Source Software as you have got it, or whether you want to modify it before using and/or distributing it to 3rd parties. In the first case, the answer should be 'unmodified', in the second 'modified'.
- The **usage context of the of the Open Source Software**: On the hand you might use the received Open Source Software as a readily prepared application. On the other hand you might embed the received Open Source into a larger application as one of its' components. More specifically, we will ask you, whether you are you using the Open Source Software as an autonomous piece of software, or whether you are using it as an embedded part of a larger, more complex piece of software. In the first case, the answer should be 'independent', in the second 'embedded'.
- The **recipient of the of the Open Source Software**: Sometimes you

[49] Of course, our newly introduced concepts 'snimoli' and 'proapse' are not absolutely one of the most elegant words. So, initially we tried to talk about 'applications' and 'libraries', although in our context these words should denote more, than they traditionally do. But we couldn't minimize the irritations of our interlocutors. Too often we had to amend that we were not only talking about applications and libraries in the strict sense of the words. Finally we decided to find our own words - and to stay open for better proposals ;-)

might wish to use the received Open Source Software only for yourself. In other cases you might intend to hand over the software (also) to other people. More specifically, we will ask you, whether you are going to use the Open Source Software only for yourself, or whether you plan to (re)distribute it (also) to third parties. In the first case, the answer should be '4yourself', in the second '4others'.

- The **mode of combination**: In this case, we will ask you, whether you are going to combine or to embed the Open Source Software with other software components by linking them statically, by linking them dynamically, or by textually including (parts of) the Open Source Software into your larger product. In the first case, the answer should be 'statically linked', in the second 'dynamically linked', in the third 'textually included'

From a more programmatic point-of-view, we can summarize these tokens as follows:

- `type::snimoli` *or* `type::proapse`
- `state::unmodified` *or* `state::modified`
- `context::independent` *or* `context::embedded`
- `recipient::4yourself` *or* `ecipient::4others`
- `mode::statically-linked` *or* `mode::dynamically-linked` *or* `mode::textually-included`

We already defined an Open Source Use Case as a combination of these tokens. If we simply combine all these tokens of all these classes with all the tokens of the other classes[50], we get 2*2*2*2*3 = 48 sets of tokens - or 48 *Open Source Use Cases*. Fortunately, some of the generated sets are invalid from an empirical or logical view, and some of these sets are context sensitive:

1. It makes no sense to ask you whether you are going to combine the received software with other software components by linking them statically or dynamically, or by including it textually into a larger unit, if you already have answered, that the received Open Source Software is a *proapse* or that it shall be used *independently*: A readily prepared application or server can't be linked to another application or server, which also contains a `main`-function. And using a *proapse* or *snimoli independently* includes that it is used *not in combination* with other units, simply because they are tokens of the same class.
2. If you already have specified that the used Open Source Software is a *proapse* - hence an autonomous program, an application, or a server -, then your answer includes, that the software is used independently and is not embedded

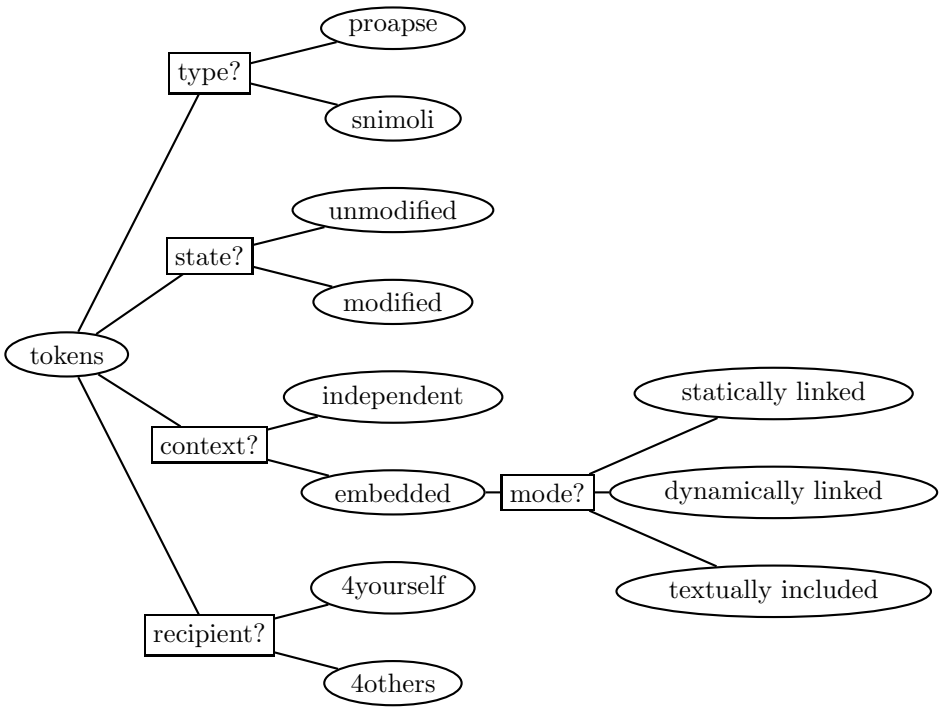
[50] in the sense of the cross product TYPE × STATE × CONTEXT × RECIPIENT × MODE

with other components into a larger unit - simply because of the nature of all *proapses*. But if you have specified that the used Open Source Software is a *snimoli* - hence a snippet of code, a module, a plugin, or a library -, then it can indeed be used as an embedded component of a constructed larger application or server, or it can be used independently in case you 'only' re-distribute it to 3rd parties.

3. If you already have specified that the used Open Source Software is a *snimoli* - hence a snippet of code, a module, a plugin, or a library -, and that this *snimoli* shall be used only by yourself (not distributed to other 3rd. parties), then your answer must also imply that this *snimoli* is used in combination, as an embedded part of a larger unit. It makes no sense to 'try' to use a library autonomously, without using it as component of another application. In this case, it would simply sit on the disk and would do nothing more than occupying space.

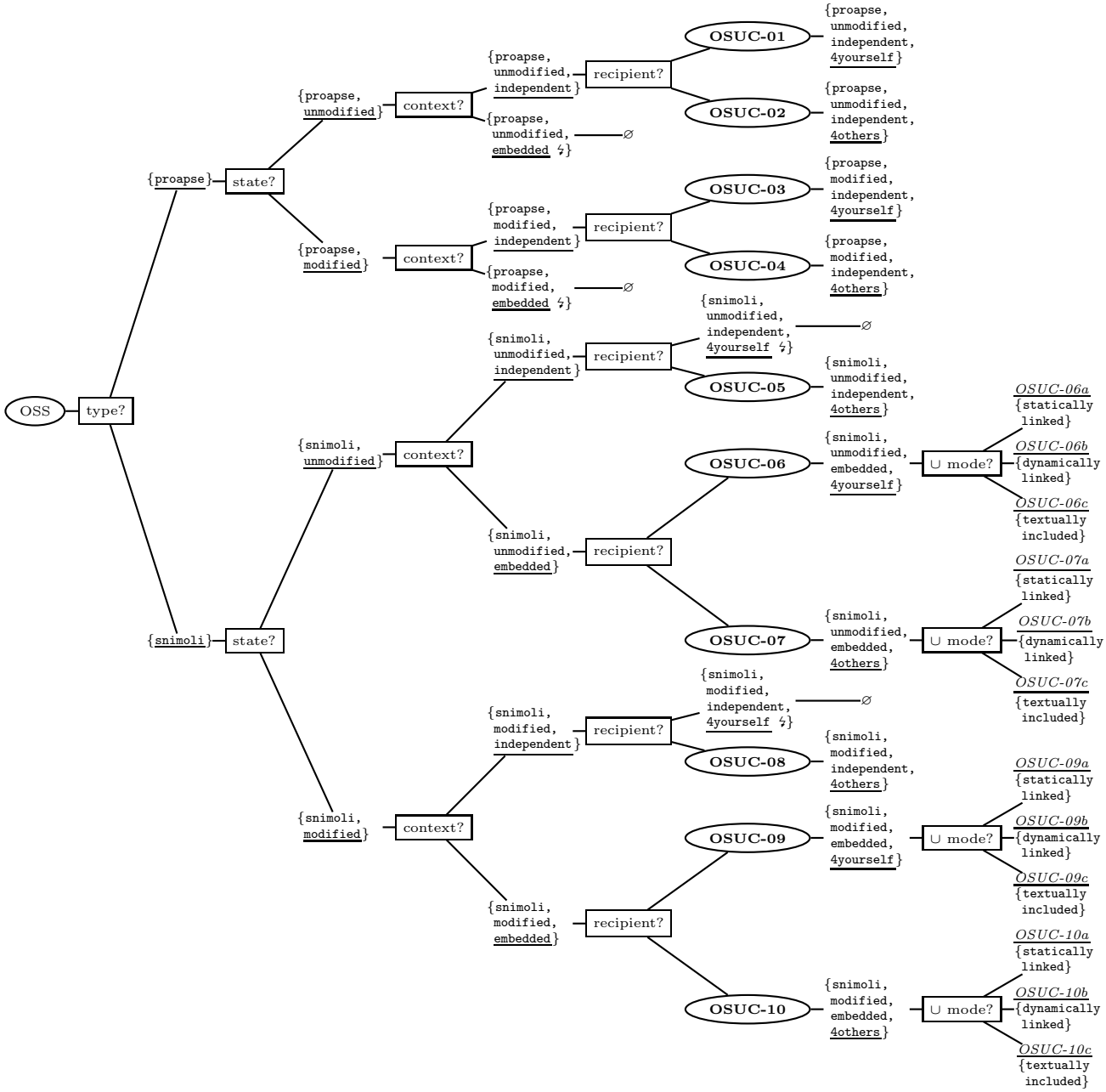
Does this sound complex? We thought so too. We spent much time explaining these constraints to ourselves, and only when we had transposed all the combinations and rules into a tree, did the situation become clearer. The following diagrams shall summarize this way of clarification:

6.1 The OS Use Case Dimensions: Classes and Tokens



6.2 The OS Use Case Taxonomy

This is one tree, 'collecting' the tokens and offering the *Open Source Use Cases* as their leafs[51]:



[51] Each of the invalid use cases (= sets of tokens) [for details s. p. 28] is marked by an ↯ and leads to an empty set (= ∅): A proapse can not be embedded with another software unit, also containing a main-function. Using a software library only for yourself and independent (not in combination with larger software unit), is like having an unused heap of bytes on your disc.

7 Open Source Use Cases: Find the License Fulfilling To-do Lists

This chapter offers the Open Source Use Case Finder*: Firstly, it presents a form to gather the specifying information. Secondly it offers a tree, which can easily be transversed by the gathered information. And finally it contains the list of* Open Source Use Cases*. Each leaf of the tree leads to one* Open Source Use Case *which itself then refers to the specific license fulfilling to-do lists.*

7.1 A standard form for gathering the relevant information

Class	Questions	Answers
Type	*Is the Open Source Software you want to use a software library in the broadest sense (an includable code snippet, a linkable module or library, or a loadable plugin) [=snimoli], or is it an autonomous program, application or server which can be executed or processed [=proapse]?*	☐ proapse ☐ snimoli
State	*Do you want to leave your Open Source Software as you have got it, or do you want to modify it before using and/or distributing it to 3rd parties?*	☐ unmodified ☐ modified
Context	*Are you using your Open Source Software as an autonomous piece of software [=independent], or are you using it as an embedded part or component of a larger, more complex piece of software [=embedded]?*	☐ independent ☐ embedded
Recipient	*Are you are going to use the received Open Source Software only for yourself [=4yourself], or do you plan to (re)distribute it (also) to third parties [=4others]?*	☐ 4yourself ☐ 4others
Mode	*Are you going to combine the received Open Source Software with other software components by linking all together statically, by linking them dynamically, or by textually including (parts of) the Open Source Software into your larger unit?*	☐ statically linked ☐ dynamically linked ☐ textually included

As discussed earlier, there are of course some invalid combinations[52]. Here are

[52] type::proapse excludes state::embedded; recipient::4yourself excludes the combination with state::independent and type::snimoli; any value of class 'mode' implies state::embedded [for details see page 28]. If you have gathered one of these invalid combinations, please check the

some extra explanations about each class:

Type: A piece of (Open Source) Software shall be viewed as a program, an application, or a server, if you can start its binary form with your normal program launcher, or (in case of an text file which still must be interpreted by an interpreter like php, perl, bash etc.) if you can start an interpreter taking the file as one of its' arguments.

State: You modify Open Source Software if you expand, reduce or modify at least one of the received software files, and - in case of dealing with binary object code - if you (re)compile and (re)link the modified software to a new binary file. If you only modify configuration files, you do not modify the Open Source Software.

Context: You use Open Source Software embedded into a larger unit, if one of your files of the larger unit contains a verbatim or modified copy (i.e. a snippet) of the received Open Source Software, or if the larger unit contains an include statement referring to a file of the received Open Source Software, or if your development environment contains a compiler or linker directive referring to the received Open Source Software.

Recipient You use the received Open Source Software only for yourself, if you as person do not pass it to other persons, or if you as a member of a specific development group pass it only to the other members of your development group. But if you store Open Source Software on any device such as a mobile phone, an USB stick, etc. or if you attach it to any transport medium like email etc. and if you then sell, give away, or simply send this device or transport medium to anyone (other than a direct member of your development group), then you indeed handover the Open Source Software to third parties[53].

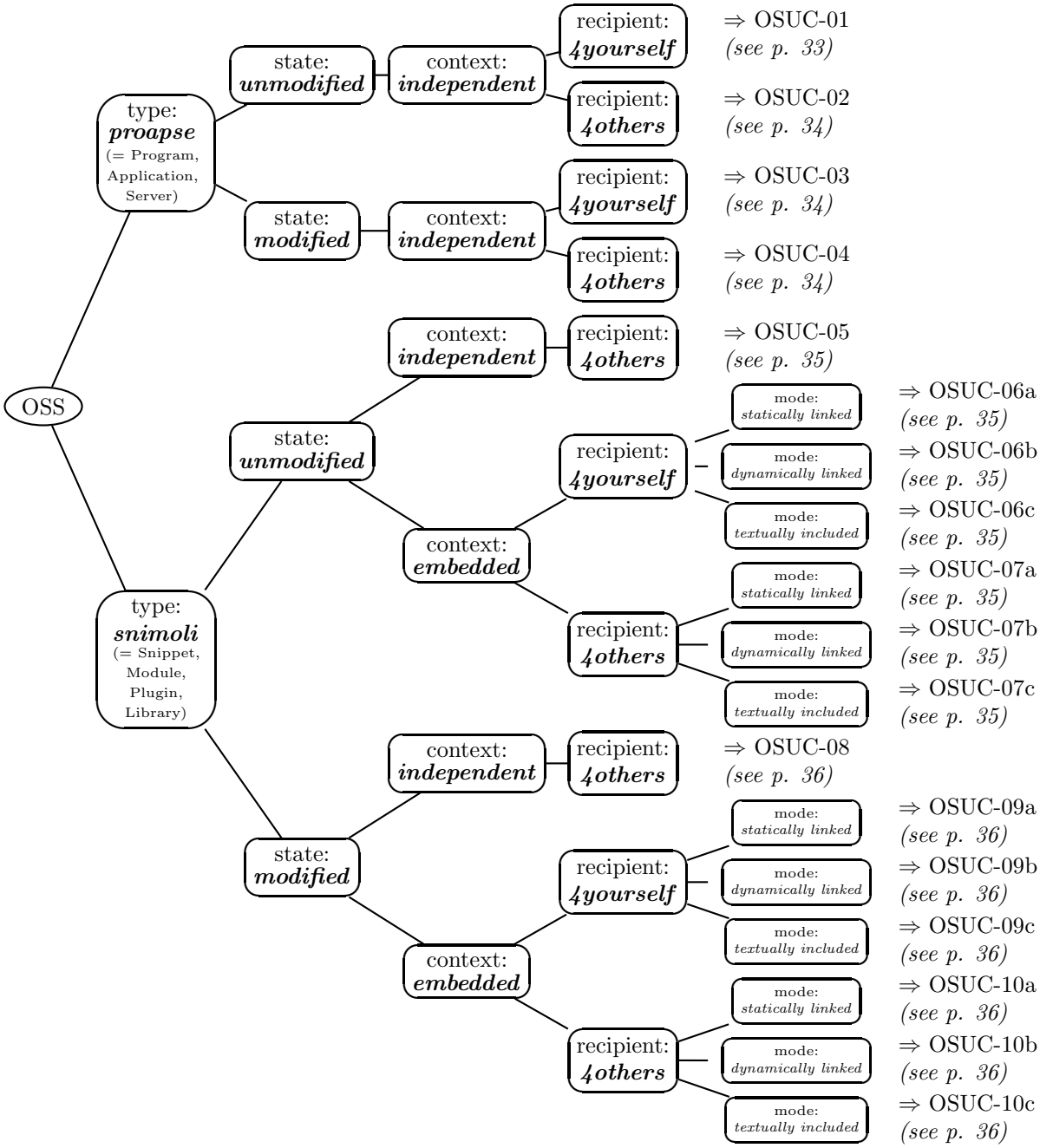
Mode This follows the standard terminology of software development. this question.

7.2 The taxonomic Open Source Use Case Finder

Now, after having gathered the necessary information, determine your specific Open Source Use Case by traversing the following tree and its corresponding branches:

corresponding explanations

[53] Please remember that - at least in Germany - there are opinions that even handing over software to another legal entity or department of the same company is also a kind of distribution. It is always safest to take the broadest possible meaning of distributing or handing over.



7.3 The Open Source Use Cases and their links to the to-do lists

OSUC-01: Only for yourself, you are using an unmodified Open Source program, application, or server - just as you received it. You are not going to combine it with other components in the sense of software development (= *proapse,*

unmodified, independent, 4yourself). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 38 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL 2.x**
- p. 46 for license **MIT**

OSUC-02: Just as you received it, you are going to distribute an unmodified Open Source program, application, or server to 3rd parties. In this act of distribution, you do not combine this program, application, or server with other software components in the sense of software development (= *proapse, unmodified, independent, 4others*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 39 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

OSUC-03: Only for yourself, you are modifying a received Open Source program, application, or server, before you are using it. But you do not combine it with other components in the sense of software development (= *proapse, modified, independent, 4yourself*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 38 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

OSUC-04: You are going to modify a received Open Source program, application, or server, before you distribute it to 3rd parties. But you do not combine this modified program, application, or server with other software components in the sense of software development (= *proapse, modified, independent, 4others*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**

- p. 40 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

OSUC-05: Just as you received it, you are going to distribute an unmodified Open Source library, code snippet, module, or plugin to 3rd parties. In this act of distribution, you do not combine this library, code snippet, module, or plugin with other software components in the sense of software development (= *snimoli, unmodified, independent, 4others*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 39 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

OSUC-06: Only for yourself and just as you received it, you are going to combine an unmodified Open Source library, code snippet, module, or plugin into a larger software unit as one of its parts. (= *snimoli, unmodified, embedded, 4yourself*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 38 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

OSUC-07: Just as you received it and before you will distribute it to 3rd parties together with the larger software unit, you combine an unmodified Open Source library, code snippet, module, or plugin into a larger software unit in the sense of software development (= *snimoli, unmodified, embedded, 4others*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 39 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2

- p. 46 for license **MIT**

OSUC-08: Before you will distribute it, you are going to modify an Open Source library, code snippet, module, or plugin to 3rd parties, but you do not combine it with other software components in the sense of software development (= *snimoli, modified, independent, 4others*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 41 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

OSUC-09: Only for yourself, you are going to modify an Open Source library, code snippet, module, or plugin, before you will combine it - in the sense of software development - into a larger software unit as one of its parts . (= *snimoli, modified, embedded, 4yourself*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 38 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

OSUC-10: Before you will distribute it to 3rd parties, you are going to modify an Open Source library, code snippet, module, or plugin, and to combine it with other software components in the sense of software development (= *snimoli, modified, independent, 4others*). To see the *specific, license fulfilling to-do lists* jump to the following pages:

- p. 37 for license **Apache 2.0**
- p. 42 for license **BSD**
- p. 46 for license **ECL**
- p. 46 for license **GPL**, release 2
- p. 46 for license **MIT**

8 Open Source License Fulfillment: Classified To-do Lists for ...

With respect to the defined Open Source Use Cases, this chapter lists for the Open Source Licenses what one has to do for fulfilling a specific license. You should be able to jump into the license specific chapter and find all relevant information - though without proving details.

8.1 Apache Licensed Software ...

8.1.1 Apache specific mini finder

8.1.2 Apache specific use case 1

(covers OSUC-X - OSUC-Z)

8.1.3 Apache specific use case n

(covers OSUC-x - OSUC-z)

8.2 BSD Licensed Software ...

8.2.1 The BSD specific mini finder

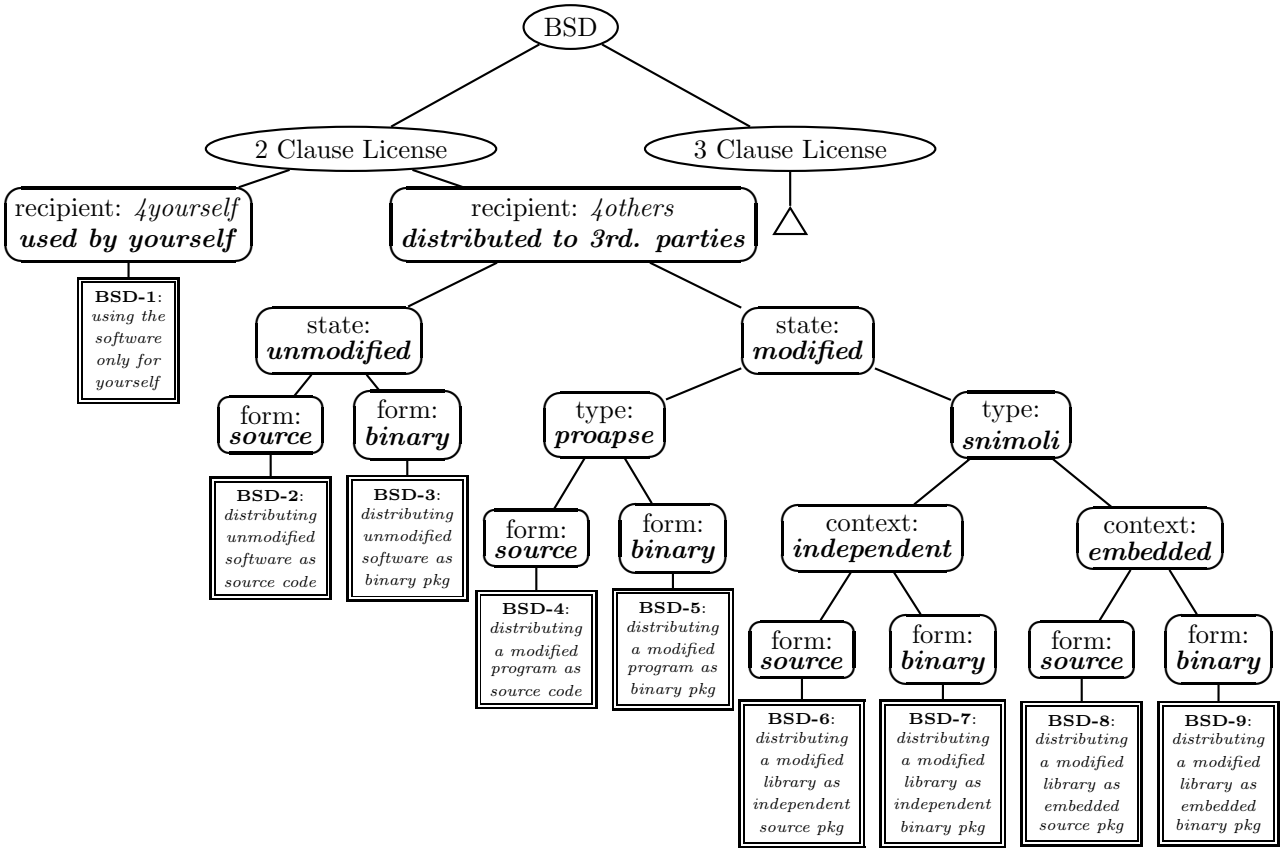
As an approved Open Source license, the BSD license exists in two versions[54]. The latest, most modern release is the *BSD 2-Clause license*[55], the elder release

[54] Following the Open Source Initiative, initially a not approved BSD license contained a fourth clause also known as advertising clause which '(...) officially was rescinded by the Director of the Office of Technology Licensing of the University of California on July 22nd, 1999'. Cf. *Open Source Initiative*: The BSD 3-Clause License; 2012 [Year n.st.] ⟨URL: `http://www.opensource.org/licenses/BSD-3-Clause`⟩ – reference download: 2012-07-04, wp. Because of the cancellation you can simply act according the *BSD 3-Clause license* if you have to fulfill the eldest BSD license.

[55] cf. *Open Source Initiative*: The BSD 2-Clause License; 2012 [Year n.st.] ⟨URL: `http://www.opensource.org/licenses/BSD-2-Clause`⟩ – reference download: 2012-07-03, wp.

is the *BSD 3-Clause license*[56]. Nevertheless, the little differences between the two versions have to be respected.

All BSD Open Source Licenses explicitly focuses 'only' on the (re-)distribution *Open Source Use Cases*, which we have specified by our token *4others*. Conditions for the use cases specified by the token *4yourself* can be derived[57]. Additionally the BSD license considers the form of the distribution, eg. whether the work is distributed as a (set of) source code file(s) or as a (set of) the binary file(s). Use the following tree to find the BSD license fulfilling to-do lists.



8.2.2 Software licensed by the BSD 2-Clause License

8.2.2.1 BSD-1: using the software only for yourself

means that you are going to use a received BSD software only for yourself and that you do not handover it to any 3rd. party in any sense.

[56] cf. *Open Source Initiative*: The BSD 3-Clause License, 2012, wp.

[57] For details of the *Open Source Use Cases tokens* see p. 27. For Details of the *Open Source Use Cases* based on these token see p. 30

covers OSUC-01, OSUC-03, OSUC-06, and OSUC-09[58]

requires the tasks in order to fulfill the conditions of the BSD license:

- You are allowed to use any kind of BSD software in any sense and in any context without any obligations if you do not handover the software to 3rd parties.

8.2.2.2 BSD-2: Passing the unmodified software as a source code package

means that you are going distribute an unmodified version of the received BSD software to 3rd. parties in form of a set of source code files or an integrated source code package[59]

covers OSUC-02, OSUC-05, OSUC-07[60]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that the licensing elements - eg. the BSD license text, the specific copyright notice of the original author(s), and the BSD disclaimer - are retained in your package in the form you have got them.
- [**voluntary:**] Let the documentation of your distribution and/or your additional material also contain the original copyright notice, the BSD conditions, and the BSD disclaimer.

8.2.2.3 BSD-3: Passing the unmodified software as a binary package

means that you are going distribute an unmodified version of the BSD received software to 3rd. parties in form of a set of binary files or an integrated binary package[61]

covers OSUC-02, OSUC-05, OSUC-07[62]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that your distribution contains the original copyright notice, the BSD license, and the BSD disclaimer in the form you have got them. If you compile the binary file on the base of the

[58] For details see pp. 33 - 36

[59] In this case it doesn't matter whether you distribute a program, an application, a server, a snippet, a module, a library, or a plugin as an independent or an embedded unit

[60] For details see pp. 34 - 35

[61] In this case it doesn't matter whether you distribute a program, an application, a server, a snippet, a module, a library, or a plugin as an independent or an embedded unit

[62] For details see pp. 34 - 35

source code package and if this compilation does not also generate and integrate the licensing files, then create the copyright notice, the BSD conditions, and the BSD disclaimer according to the form of the source code package and insert these files into your distribution manually.

- [**mandatory:**] Ensure, that the documentation of your distribution and/or your additional material also contain the author specific copyright notice, the BSD conditions, and the BSD disclaimer.

General remark for all binary distributions: *Even if you are distributing a BSD binary package on a medium, which doesn't allow the user, to see package files directly - some mobile devices don't give their users the full access to all stored elements - you have to fulfill the mandatory requirements. So, sometimes, it is necessary, to let simply the BSD license and the copyright notice become an invisible part of the binary package and to deploy the files together with the package. The point is: you have fulfilled the license.*

8.2.2.4 BSD-4: Passing a modified program as a source code package

means that you are going distribute a modified version of the received BSD program, application, or server (proapse) to 3rd. parties in form of a set of source code files or an integrated source code package.

covers OSUC-04[63]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that the licensing elements - e.g. the BSD license text, the specific copyright notice of the original author(s), and the BSD disclaimer - are retained in your package in the form you have got them.
- [**voluntary:**] Let the documentation of your distribution and/or your additional material also contain the original copyright notice, the BSD conditions, and the BSD disclaimer.
- [**voluntary:**] It is a good practice of the Open Source community, to let the copyright notice which is shown by the running program also state that the program is licensed under the BSD license. Because you are already modifying the program, you can also add such a hint, if the presented original copyright notice lacks such a statement.

[63] For details see pp. 34

8.2.2.5 BSD-5: Passing a modified program as a binary package

means that you are going distribute a modified version of the received BSD program, application, or server (proapse) to 3rd. parties in form of a set of binary files or an integrated binary package.

covers OSUC-04[64]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that your distribution contains the original copyright notice, the BSD license, and the BSD disclaimer in the form you have got them. If you compile the binary file on the base of the source code package and if this compilation does not also generate and integrate the licensing files, then create the copyright notice, the BSD conditions, and the BSD disclaimer according to the form of the source code package and insert these files into your distribution manually[65].
- [**mandatory:**] Ensure, that the documentation of your distribution and/or your additional material also contain the author specific copyright notice, the BSD conditions, and the BSD disclaimer.
- [**voluntary:**] It is a good practice of the Open Source community, to let the copyright notice which is shown by the running program also state that the program is licensed under the BSD license. Because you are already modifying the program, you can also add such a hint, if the presented original copyright notice lacks such a statement.

8.2.2.6 BSD-6: Passing a modified library as independent source code package

means that you are going distribute a modified version of the received BSD code snippet, module, library, or plugin (snimoli) to 3rd. parties in form of a set of source code files or an integrated source code package, but without embedding it into another larger software unit.

covers OSUC-08[66]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that the licensing elements - e.g. the BSD license text, the specific copyright notice of the original author(s), and the BSD disclaimer - are retained in your package in the form you have got them.

[64] For details see pp. 34
[65] see also our 'Mobile Device Hint' on p. 40
[66] For details see pp. 36

- [**voluntary:**] Let the documentation of your distribution and/or your additional material also contain the original copyright notice, the BSD conditions, and the BSD disclaimer.

8.2.2.7 BSD-7: Passing a modified library as independent binary package

means that you are going distribute a modified version of the received BSD code snippet, module, library, or plugin (snimoli) to 3rd. parties in form of a set of binary files or an integrated binary package, but without embedding it into another larger software unit.

covers OSUC-08[67]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that your distribution contains the original copyright notice, the BSD license, and the BSD disclaimer in the form you have got them. If you compile the binary file on the base of the source code package and if this compilation does not also generate and integrate the licensing files, then create the copyright notice, the BSD conditions, and the BSD disclaimer according to the form of the source code package and insert these files into your distribution manually[68].
- [**mandatory:**] Ensure, that the documentation of your distribution and/or your additional material also contain the author specific copyright notice, the BSD conditions, and the BSD disclaimer.

8.2.2.8 BSD-8: Passing a modified library as an embedded source code package

means that you are going distribute a modified version of the received BSD code snippet, module, library, or plugin (snimoli) to 3rd. parties in form of a set of source code files or an integrated source code package together with another larger software unit, which contains this code snippet, module, library, or plugin as an embedded component.

covers OSUC-10[69]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that the licensing elements - e.g. the BSD license text, the specific copyright notice of the original author(s), and

[67] For details see pp. 36
[68] see also our 'Mobile Device Hint' on p. 40
[69] For details see pp. 36

the BSD disclaimer - are retained in your package in the form you have got them.

- [**voluntary:**] Let the documentation of your distribution and/or your additional material also contain the original copyright notice, the BSD conditions, and the BSD disclaimer.
- [**voluntary:**] It is a good practice of the Open Source community, to let the copyright notice which is shown by the running program also state that it contains components licensed under the BSD license. Because you are embedding this snimoli into a larger software unit, you are developing this larger unit. Hence, you can also expand the copyright notice of this larger unit by such a hint to its BSD components.

8.2.2.9 BSD-9: Passing a modified library as an embedded binary package

means that you are going distribute a modified version of the received BSD code snippet, module, library, or plugin to 3rd. parties in form of a set of binary files or an integrated binary package together with another larger software unit, which contains this code snippet, module, library, or plugin as an embedded component.

covers OSUC-10[70]

requires the tasks in order to fulfill the license conditions

- [**mandatory:**] Ensure, that your distribution contains the original copyright notice, the BSD license, and the BSD disclaimer in the form you have got them. If you compile the binary file on the base of the source code package and if this compilation does not also generate and integrate the licensing files, then create the copyright notice, the BSD conditions, and the BSD disclaimer according to the form of the source code package and insert these files into your distribution manually[71].
- [**mandatory:**] Ensure, that the documentation of your distribution and/or your additional material also contain the author specific copyright notice, the BSD conditions, and the BSD disclaimer.
- [**voluntary:**] It is a good practice of the Open Source community, to let the copyright notice which is shown by the running program also state that it contains components licensed under the BSD license. Because you are embedding this snimoli into a larger software unit, you are developing this larger unit. Hence, you can also expand the copyright notice of this larger unit by such a hint to its BSD components.

[70] For details see pp. 36
[71] see also our 'Mobile Device Hint' on p. 40

8.2.3 Software licensed by the BSD 3-Clause License

Compared with the *BSD 2-Clause license*, the *BSD 3-Clause license* only contains one additional 3rd. clause, the rest is the same. So, for acting according the *BSD 3-Clause license*, do that, what you would have to do for fulfilling the 2 clause license. And additionally do not use the name of licensing organization or the names of the licensing distributors to promote your own work.

8.2.4 Discussions and Explanations

The *BSD 2-Clause license* has a simple textual structure: In the beginning, it generally '(permits) [the] redistribution and [the] use in source and binary forms, with or without modification, [...]', if one fulfills the two rules of the license[72]. The first rule concerns the (re)distribution in form of source code, the second the (re)distribution of binary packages. Here are some explanations why we translated the rules into which sets of executable tasks:

- For the 'redistribution of source code' the license requires, that the package must ' [...] retain the above copyright notice, this list of conditions and the following disclaimer'[73]. Hence, you are not allowed, to modify any of the copyright notes, which are already embedded in the received (source) files. And from a logical point of view, there must exist an explicit or implicit assertion, that the software is licensed under the *BSD 2-Clause license*[74]. This is often implemented by simply adding a copy of the license into the package. Hence, you are furthermore not allowed to modify these files or corresponding text snippets. For our purposes, we translated the bans into the following executable task:

  > *Ensure, that the licensing elements - eg. the BSD license text, the specific copyright notice of the original author(s), and the BSD disclaimer - are retained in your package in the form you have got them.*

- For the redistribution in form of binary files, the license requires, that the licensing elements must be '[...] (reproduced) in the documentation

[72] cf. *Open Source Initiative*: The BSD 2-Clause License, 2012, wp.
[73] cf. id., ibid.
[74] The BSD license requires, that a re-distributed software package must contain the (package specific) copyright notice, the (license specific) conditions and the BSD disclaimer. (cf. id., l.c., wp.) You might ask what you should do, if these elements are missed in the package you got. If so, the package you got had not been licensed adequately. Hence, you do not know reliably whether you have received it under a BSD license. In other words: If you have received a BSD licensed software package, it must contain sufficient license fulfilling elements, or it is not a BSD licensed software.

and/or other materials provided with the distribution'[75]. Hence, this is not required as necessary condition for the (re)distribution as source code package . But nevertheless, even for a distribution in form of source code, it is often possible, to fulfill this rule too - eg., if you offer an own download site for source code packages. In such cases, it is a sign of respect, to mention the licensing not only inside of the packages, but also in the text of your site. Because of that, we added the following voluntary task for all BSD Open Source Use Cases, which deal with the redistribution in form of source code'

> *Let the documentation of your distribution and/or your additional material also contain the original copyright notice, the BSD conditions, and the BSD disclaimer.*

- Naturally, because the reproduction of the licensing elements 'in the documentation and/or other materials provided with the distribution' is explicitly required for the 'redistribution in binary form'[76], we had to rewrite the facultative task for a distribution in form of source code as a mandatory task for all BSD Open Source Use Cases, which deal with the redistribution in binary form':

  > *Ensure, that the documentation of your distribution and/or your additional material also contain the author specific copyright notice, the BSD conditions, and the BSD disclaimer.*

- In case of (re)distributing the program in form of binary files, it is sometimes not enough, to pass the licensing elements as one has got them. If you compile the binary package from the source code, it is not necessarily true, that the licensing elements are also automatically generated and embedded into the 'binary package'. But nevertheless, you have to add the copyright notice, the conditions and the disclaimer to this package for acting according to the BSD license. Therefore we chose the following form of an executable, license fulfilling task for all binary oriented distributions:

  > *Ensure, that your distribution contains the original copyright notice, the BSD license, and the BSD disclaimer in the form you have got them. If you compile the binary file on the base of the source code package and if this compilation does not also generate and integrate the licensing files, then create the copyright notice, the BSD conditions, and the BSD disclaimer according to the form of the source code package and insert these files into your distribution manually.*

[75] cf. *Open Source Initiative*: The BSD 2-Clause License, 2012, wp.
[76] cf. id., ibid.

- Finally, we wished to insert a hint to the general (Open Source) tradition, to mention the used Open Source Software and their licenses as a remark of the 'copyright widget' of an application. This is not required by the BSD license. But it is a general, good tradition. Naturally, because of the freedom, to use and modify Open Source Software, and to redistribute a modified version of it, you are also allowed to insert such references, even if they are missed. Therefore we added a third voluntary, license tradition fulfilling task for all relevant Open Source Use Cases.

8.3 Eclipse Licensed Software in the usage context of ...

8.3.1 ECL specific mini finder

8.3.2 ECL specific use case 1

(covers OSUC-X - OSUC-Z)

8.3.3 ECL specific use case n

(covers OSUC-x - OSUC-z)

8.4 GPL V2 Licensed Software in the usage context of ...

8.4.1 GPL-2 specific mini finder

8.4.2 GPL-2 specific use case 1

(covers OSUC-X - OSUC-Z)

8.4.3 GPL-2 specific use case n

(covers OSUC-x - OSUC-z)

8.5 MIT Licensed Software in the usage context of ...

8.5.1 MIT specific mini finder

8.5.2 MIT specific use case 1

(covers OSUC-X - OSUC-Z)

8.5.3 MIT specific use case n

(covers OSUC-x - OSUC-z)

9 Open Source Licenses and Their Legal Environments

In this chapter we analyze why to know a license alone is not enough. At the end you will know that Open Source Licenses are embedded into the legal environment of a state. And you will know in which sense the German legal environment predetermines your readings of Open Source Licenses.

[TDB . . .]

10 Conclusion

This chapter shortly describes what the OSLiC is, how it should be used, and how it can be read. It shall be written as top-down explanation.

[TDB . . .]

11 Appendices

This chapter shortly describes what the OSLiC is, how it should be used, and how it can be read. It shall be written as top-down explanation.

[TDB ...]

11.1 Some Widespread Open Source Myths

From the viewpoint of an internet student we have to consider that the web offers a mass of rumors concerning the nature of Open Source Software (Licenses). Here are some of the myths[77]. we met:

Open Source tries to improve the world ethically :- no, there's a clear ban to exclude persons, groups, purposes

Changed Open Source Software must be re-published :- no, in a double sense! There are OS licenses which a allow the proprietarization of the modified code. And even the LGPL and the GPL, which clearly try to prevent the proprietarization, do not require generally that a modified code must be (re-)published. Only if you give your modfied (L)GPL licensed application as binary to anyboday then you have to handover the modified code too.

Modified Open Source Software must be given back to the whole community :- No. Again, there are OS licenses which a allow the proprietarization of the modified code. And even the LGPL and the GPL - which clearly require, that you also publish the modified code, if you give the modified binary to anybody - do not require that you distribute your modification around the world. LGPL and GPL clearly say that you have to hand over the code to those persons which you give the binary. And if you only give your improvement only one person or a group of person, then you must handover your the code only to that persons or only to all members of that group.

[77] At least one time even a scientific legally discussing book is talking about the 'Myth around Open Source Licenses' - although only as part of the title: cf *Guibault, Lucie* a. *Ot van Daalen*: Unravelling the Myth around Open Source Licenses. An Anaysis from A Dutch and European Law Perspective; The Hague: T. M. C. Asser Press, 2006 (= IT & Law, [Vol./No.] 8), ISBN 978–90–6704–214–7, pp. 1ff, especially 209ff.

Published Open Source Software is open for ever :- No. The Copyright holder ever holds the Copyright. The can change the licence of next release of its software.

Software can either be Open Source Software or proprietary software :- The Copyright holders can ever distribute the code under other conditions, een additionally. That's not a question of the licence, but of the Copyright.

The opposite of Open Source Software is commercial Software :- No. Firstly you are also allowed to use the Open Source software in any commercial purpose. There's only one point which is excluded in OSS: you are not allowed to ask for a licence fee if you distribute 'Open Source Software'. Secondly there are many other forms like Freeware, Public Domain Software or anything else which is neither Open Source Software nor Commercial Software. It's senseless to take the question of money as mark for distinguish Open Source Software and its opposite. Moreover: Proprietary Software as opposite of Open Source Software should be defined ex negativo: all kind of software, which does not fit the OSD is proprietary.

Open Source Software prohibits to earn money :- No, you are allowed to invent each business model you want. There's only one exception: you are not allowed to ask for a licence fee if you distribute 'Open Source Software [Achtung: sollte eigentlich nur für GPL gelten]. Historically this mistake might be evoked by Debian: The GNU project missed its kernel while the Linux kernel was already distributed as part of collections which also include GNU software. Then, in 1983? Ian Murdock was supported by RMS and its FSF to build a really free distribution (Debian) containg GNU software and the Linux kernel. But Ian Murdock states also, that debian does not want to earn money. (clear details)

Modifications of Open Source Software must be marked :- No. This is not a defining postulation of the OSD. The OSD allows licenses to require the mark of modifacations. But it does not require from all licenses to rquire the mark modifications for being an Open Source License.

Modifications of Open Source Software must be marked by your personal data :- No, it's only required to mark modifications so that a reader could distinguish the modifications from the original code. It's required for saving the integrity of the original author. And therefor it's not required as a constitutiv criteria by the OSD. It might be, that a license additionally requires your name. But's not featue of Open Source Software in general. And at least the licenses discussed by us do not require to insert your name.

The Open Source Definition determines the conditions to use Open Source Software :- No. The Open Source Definition determines which licenses are Open Source Licenses, nothing more. The OSD is a set of necessary conditions

to be an Open Source License. It determines the freedom and the responsibilities of a user as a set of more or less abstract rules. But it does not constitute a set of sufficient tasks which a user has to do for fulfilling any Open Source License. Open Source Licenses may differ by instatiating the OSD criteria. So, if you want to know what you have to do to fulfill a license you have to go back to the real license of that software you are using.

Periodicals, Shortcuts, and Overlapping Abbreviations

BISE	Business & Information Systems Engineering [ISSN: 1867-0202]
	Berkeley Technology Law Journal [ISSN:]
BWV	Berliner Wissenschafts-Verlag GmbH
	Cultural Anthropology [ISSN: 1548-1360]
CiHB	Computers in Human Behavior [ISSN: 0747-5632]
CotACM	Communications of the ACM [ISSN: 0001-0782]
CR	Computer und Recht. Zeitschrift für die Praxis des Rechts der Informationstechnologien
CRi	Computer Law Review international [ISSN: 1610-7608]
	Computers & Education [ISSN: 0360-1315]
	Cutter IT Journal [ISSN: 1048-5600]
DDT	Drug Discovery Today [ISSN: 1359-6446]
DSS	Decision Support Systems [ISSN: 0167-9236]
	Ethics and Information Technology [ISSN: 1388-1957]
E.C.L.R.	European Competition Law Review [ISSN:]
EER	European Economic Review [ISSN: 0014-2921]
	Information & Management [ISSN: 0378-7206]
ibid.	ibidem = latin for 'at the same place'
ICC	Industrial and Corporate Change [ISSN: 0960-6491]
id.	idem = latin for 'the same', be it a man, woman or a group...
IEaP	Information Economics and Policy [ISSN: 0167-6245]
	IEEE Software [ISSN: 0740-7459]
ifross	Institut für Rechtsfragen der Freien und Open Source Software
	International Information and Library Review [ISSN: 1057-2317]
	International Journal of Medical Informatics [ISSN: 1386-5056]
	interactions[ISSN: 1072-5520]
ISJ	Information Systems Journal [ISSN: 1365-2575]
ITRB	Der IT-Rechtsberater [ISSN: 1617-1527]
JAIS	Journal of the Association for Information Systems [ISSN: 1536-9323]
JCSC	Journal of Computing Sciences in [Small] Colleges [ISSN: 1937-4771]
JISE	Journal of Information Science and Engineering [ISSN: 1016-2364]
JLEO	Journal of Law, Economics, & Organization [ISSN: 1465-7341]
JMIR	Journal of Medical Information Research [ISSN: 1438-8871]
	Journal of Academic Librarianship [ISSN: 0099-1333]
	Journal of Comparative Economics [ISSN: 0147-5967]
	Journal of Systems and Software [ISSN: 0164-1212]
JSIS	Journal of Strategic Information Systems [ISSN: 0963-8687]
l.c.	loco citato = latin for 'in the place cited'
LJ	Linux Journal [ISSN: 1075-3583]

.....	netWorker [ISSN: 1091-3556]
np.	no page numbering
n.st.	not stated
.....	Organization Science [ISSN: 1047-7039]
.....	Queue [ISSN: 1542-7730]
.....	R&D Management [ISSN: 1467-9310]
RP	Research Policy [ISSN: 0048-7333]
SIGCSE Bulletin ...	SIGCSE Bulletin [ISSN: 0097-8418]
SIGCAS	ACM SIGCAS Computers and Society [ISSN: 0095-2737]
SIGMIS Database ..	ACM SIGMIS - The Data Base for Advances in Information Systems [ISSN: 0095-0033]
SIGSOFT SEN	SIGSOFT Software Engineering Notes [ISSN: 0163-5948]
.....	Stanford Law Review [ISSN: 00389765]
.....	Software Qualilty Journal [ISSN: 0963-9314]
STHV	Science, Technology & Human Values [ISSN: 0162-2439]
ToIT	Transaction on Internet Technology [ISSN: 1533-5399]
ToSEM	Transactions on Software Engineering Methodology [ISSN: 1049-331X]
Ubiquity	Ubiquity - The ACM IT Magazine and Forum [ISSN: 1530-2180]
.....	University of Chicago Law Review [ISSN:]
.....	University of Illinois Law Review [ISSN:]
.....	University of Pittsburgh Law Review [ISSN:]
wp.	webpage / webdocument without any internal (page)numbering
ZGE / IPJ	Zeitschrift für geistiges Eigentum [ISSN: 1867-237x]

Bibliography

Apache Software Foundation: Apache License, Version 2.0; 2004, FreeWeb/Html ⟨URL: `http://www.apache.org/licenses/LICENSE-2.0`⟩ – reference download: 2011-08-31
The orignal text of the apache license. Specifies the 'Does' and the 'Don'ts'.

Arlt, *Brinkel*, a. *Volkmann; Spindler, Gerald, editor*: 'BSD' - und 'Mozilla'-artige Lizenzen; In *Spindler*: Rechtsfragen bei Open Source Software, 2004, pp. 317–372, Print
This chapter of the book describes the thoughts of the BSD and the Mozilla licenses. The Mozilla licenses are taken as Copyleft licenses which differ from the GPL.

Asche, Michael et al., editors: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen; (= POWeR / Patent Offensive Westfalen Ruhr, [Vol./No.] 3) Münster, New York, München [... etc.]: Waxmann, 2008, Print, ISBN 978–3–8309–1845–5
This collection focuses on patents and Open Source Software as a challenge for innovative universities. It offers articles analyzing different aspects of this topic.

Babcock, Charles: Big Test For Open Source GPL; in: Informationweek, 17 December (2006), p. np., Copy
The very short article refers to the famous case 'Jacobsen v. Katzer' and states, that it only covered the Artitistic License. For the GPL and under the same interpretation the Free Software Foundation would get the right 'to stop Cisco from using the code' - a big test.

Behlendorf, Brian: How Open Source Can Still Save the World; conference contribution; In *Boldyreff et al.*: Open Source Ecosystems, 2009, p. 2, BibWeb/PDF
Keynote. Affirms that the style of cooperation being offered by Open Source Software supports also the management of general crises.

Berry, David M.: Copy, Rip, Burn; The Politics of Copyleft and Open Source; London: Pluto Press, 2008, Print, ISBN 978–0–7453–2414–2
This sociological book contains at least some remarks on the open source history - focusing on the internal 'battles' ('from software to open source?') and its discources ('the contestation of code').

Boldyreff, Cornelia et al., editors: Open Source Ecosystems: Diverse Communities Interaction; 5th IFIP WG 2.13 International Conference on Open Source Systems, OSS 2009 Skövde, Sweden, June 3-6, 2009; Berlin, Heidelberg and New York: Springer, 2009, BibWeb/PDF, ISBN 978–3–642–02031–5
Proceedings. Relevant articles are itemized explicitly.

Booth, David R.: Peer Production and Software. What Mozilla Has To Teach Government; Cambridge (Massachusetts) and London (England): MIT Press, 2010 (= The John D. and Catherine T. MacArthur Foundation Reports on Digital Media and Learning), Print, ISBN 978–0–262–51461–3
The book describes some aspects of the collaboration in the Mozilla project and its objective Firefox.

Bretschneider, Ulrich, *Rainer Glaschick*, a. *Gernot Gräfe*: Ratgeber für die Veröffentlichung von Open-Source-Software durch eine Hochschule; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 167–188, Print
This article summarizes basic aspects of an Open Source publication succesfully. It addresses the (German) difference between copyright ('Urheberecht') and transferring the right to use

('Nutzungsrecht'). It mentions the problem of liability. It discusses the choice of an Open Source License with respect to the indented purpose and many things more. Nevertheless the article can't be taken as the sought-after 'Open Source Compendium': it doesn't analyze in which cases a University perhaps must publish its' developments or what it must do for fulfilling the licenses of internally (re-)used and distributed Open Source Software.

Brügge, Bernd et al.: Open-Source Software. Eine ökonomische und technische Analyse; Berlin and Heidelberg: Springer, 2004, Print, ISBN 3–540–20366–4

This book wants to explain central factors of OSS: it lists examples, clusters OS licenses, describes the development process and analyzes the OSS work of companies. But nevertheless it does not describe in a real sense what companies have to do for fulfilling the licenses. For the authors these obligations seem to be implicitely clear.

Buchtala, Rouven: Determinanten der Open Source Software-Lizenzwahl. Eine spieltheoretische Analyse; Frankfurt am Main, Berlin, Bern [... etc.]: Peter Lang, 2007 (= Informationsmanagement und strategische Unternehmensführung), [Vol./No.] 12), Print, ISBN 978–3–631–57114–9

The book tries to detect why specific OS-licenses are chosen. Game theoretic aspects shall help to answer the question. Thereto OS-licenses are classified as 'permissive', 'restrictive' and 'highly restrictive' licenses. This differentiation highlights some necessary constraints for respecting a license. But the names of these categories are generated with respect to a company which wants to be able to (re)use and (re)sell OS code with minimal restrictions. The intention of the OS licensors to defend the licensed freedom for revocations is not covered by these names.

Bygott, David: David Bygott's Gnu Book. A light-hearted look at the Gnu, or Wildebeest; firstly published 1998; Southerton, Harare: Robert Woollacott, 1992, Print, ISBN 0–7974–1082–1

Just a little witty comic book playing with the pronounciation of 'GNU' and 'nu' or 'new': it illustrates the 'Revegnue' or the 'Sgnuker'.

Christ, Fabian a. *Stefan Sauer*: OSS - Open-Source-Stacks; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 133–154, Print

OSS is a favored abbreviation: It doesn't stand only for 'Open Source Software' but for 'Open Source Stacks' or even for 'Operation Support Systems'.

Coar, Ken a. *Rich Bowen*: Apache Kochbuch; deutsche Übersetzung v. Jochen Wiedmann; Beijing [...]: O'Reilly, 2004, Print, ISBN 3–89721–371–0

Significantly the 'Apache License' which has enabled the wide use of this http server is not discussed in this book. It doesn't ask wether you are allowed to use the software or under which conditions you may use which module, although these answers are necessary to do really what the book describes.

De Nicolò, Christopher: Open Source Software - Rechtliche Aspekte nach deutschem und italienischem Recht. Eine rechtsvergleichende Studie. Dissertation; Regensburg: Universität Regensburg, 2010, Print

Tries to describe the differences between the Italien and the German handling of Open Source Software: With respect to the copyright the OS licenses offer a nearly equal level of proctection. But the type of the contract seems to be differently classified: In Germany Copyleft licenses shall be seen as tradeoff contracts, all other OS licenses as a donation. In Italy contracts of donation are a problem. In return in both countries the licenses are equivalent with respect to the liability.

Djordjevic, Valle et al., editors: Urheberrecht im Alltag. Kopieren, bearbeiten, selber machen; Bonn: Bundeszentrale für politische Bildung, 2008, Print, ISBN 978–3–89331–812–4

German written collection of articles which describe aspects of the copy right in the internet.

It is not focused on 'Open Source' but on the copy of electronical goods in general. Nevertheless the book mentions the topics '(Open Source) software', 'free culture' and 'creative commons (licences)'.

Drossou, Olga, *Stefan Krempl*, a. *Andreas Poltermann*: Der Kampf um die Innovationsfreiheit: Der Big Bang des Wissens und seine Sprengkraft. Plädoyer für einen offenen Umgang mit Wissen im Interesse der Innovationskraft von Wirtschaft und Gesellschaft; Editorial; In *Drossou, Krempl, a. Poltmann*: Die wunderbare Wissensvermehrung, 2006, pp. 1–10, Print

This editorial states that collaboratively supported knowledge must be free if it shall evoke an augmentation of knowledge. Example for the new method is the Open Source collaboration which has already changed the style of cooperating of writers (GNU Free Documentation License, Creative Commons-Initiative). Patents are opposed to that method. But the editorial states that innovation needs a free interplay of forces and an exchange of knowledge as bride as possible.

Drossou, Olga, *Stefan Krempl*, a. *Andreas Poltmann, editors*: Die wunderbare Wissensvermehrung. Wie Open Innovation unsere Welt revolutioniert; (= Telepolis) Hannover: Heise, 2006, Print, ISBN 3–936931–38–0

A collection of articles which highlight the necessity of a free exchange for innovation emphaticly.

Eckl, Julian: Die politische Ökonomie der Wissenschaftsgesellschaft. Geistige Eigentumsrechte und die Frage des Zugangs zu Ideen; Marburg: Tectum Verlag, 2004, Print, ISBN 3–8288–8735–X

This book wants to show that for ever the concept of intellectual properties has been discussed controversially and that Open Source therefore challanges the current 'state of the art'-position legitimately. It contains a short but meaningful summary of the Open Source history.

Euler, Ellen: Creative Commons: Mehr Innovation durch die Öffnung des Urheberechts? In *Drossou, Krempl, a. Poltmann*: Die wunderbare Wissensvermehrung, 2006, pp. 147–158, Print

The article states that the German 'Urheberrecht' was developed to protect the common interest in new ideas: it should solve the free rider problem. Only the author has the right to allow or forbid the distribution of his works. And this right stimulates to create / describe new ideas. But this right is limited by the also true interests of the community. Therefore in Germany private citizens are ever allowed to cite a work or to make private copies of the work. But in our digital world DRM etc. disable this common right and undermine the 'Urheberrecht'. Creative Commons gives the possibility to grant rights back to the author.

Feig, Michael: Einführung in GNU; München and Wien: Carl Hanser Verlag, 1996 (= Unix easy), Print, ISBN 3–446–18311–6

This elder little book concerns the GNU project as a collection of free tools which together shall constitute a free Unix. The GNU license and philosophy are very briefly discussed. But the GNU tools - mostly running in a shell at this time - are described in a deeper sense. Hence in a way this book is something like a compendium for the elder GNU applications.

Fogel, Karl: Producing Open Source Software; How to Run a Successful Free Software Project; Beijing, Cambridge, Köln [...]: O'Reilly, 2006, Print, ISBN 978–0–596–00759–1

Well known english written standard work. Focuses on the process of developing Open Source Software. Licences are discussed in a small chapter. Describes very clearly the concepts of 'Free Software', 'Open Source Software', 'FLOSS' and so on. Analyzes also the topic 'License Compatibility'.

Geese, Elmar: Innovation und freie Software; In *Drossou, Krempl, a. Poltmann*: Die wunderbare Wissensvermehrung, 2006, pp. 77–84, Print

The article states that we are living in a time of monopolizing knowledge. It is established by

patents, drm and laws. By supporting the privatization of knowlegde politics undermine their aim to support innovation. Positive counterexamples for a free culture are the maintenance of Staroffice/OpenOffice or Mozilla.

Gehring, Robert A. a. *Bernd Lutterbeck, editors*: Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell; Berlin: Lehmanns Media, 2004 ⟨URL: `http://www.opensourcejahrbuch.de/download/jb2004/OpenSourceJahrbuch2004.pdf`⟩ – reference download: 2011-08-29, Print & FreeWeb/PDF, ISBN 3–936427–78–X

First volume of the famous German written row 'Open Source Annual'. Offers articles on heterogeneous aspects of Open Source Software.

Gilroy, Bernard Michael a. *Tobias Volpert*: Die Funktionen eines Patentsystems und ihre Bedeutung für Unternehmensausgründungen aus Hochschulen; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 21–39, Print

This article outlines the two social functions of a patent system: Firstly it shall evoke innovations by granting an exclusive right to use the invented results. Secondly it shall inform the scientific society by requiring the publication of new data. Finally the article argues that publishing information also evokes innovations because it's the base of succedding research.

Grassmuck, Volker: Freie Software. Zwischen Privat- und Gemeineigentum; Themen und Materialien; Bonn: Bundeszentrale für politische Bildung, 2002, Print, ISBN 3–89331–432–6

German standard work. The book was written in that time when Open Source seemed to be strange. It still offers a reasonable descriptions although the set of licences has grown in the mean time.

Gräfe, Gernot: Open-Source-Software und Open-Source-Portale - Potentiale für die Softwareentwicklung in Hochschulen und den Ergebnistransfer in die Praxis; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 55–72, Print

The article lists some opportunities of university specific open source software portals, as the support of the software development, a better reuse of the development results or a better transfer of the theoretical results into the praxis.

Guibault, Lucie a. *Ot van Daalen*: Unravelling the Myth around Open Source Licenses. An Anaysis from A Dutch and European Law Perspective; The Hague: T. M. C. Asser Press, 2006 (= IT & Law, [Vol./No.] 8), Print, ISBN 978–90–6704–214–7

The book describes the Open Source idea from the viewpoint of the Dutch legal system. It ends in five recommendations for enforcing the clarity of OS licenses and their usage. Except for the title the myth of Open Source is not explicitly mentioned or discussed.

Heap, Nicholas: OSI-Referenzmodell ohne Geheimnis; translated by G & U, Flensburg; Hannover: Heise, 1994, Print, ISBN 3–88229–045–5

In the most cases 'OSI' refers the 'Open Systems Interconnection Model', not the 'Open Source Initiative'. Here a German written book explaining the elder meaning of OSI as a model of network layers.

Hofmann, Susanne, *Sven Pfeiffer*, a. *Urs Walter*: Open Source School. Neue Synergien zwischen Schule und Kiez in Gropiusstadt. Architektur als sozialer Katalysator; Berlin, 2010, BibWeb/PDF ⟨URL: `http://opus.kobv.de/tuberlin/volltexte/2010/2841/pdf/9783798322738_content.pdf`⟩ – reference download: 2011-07-30

Although using a seducing title the pdf file doesn't offer anything concerning 'Open Source Software': topic is a special kind of managing architecture of buildings.

ifross: FAQ; 2011, FreeWeb/HTML ⟨URL: `http://www.ifross.org/faq-haeufig-gestellte-fragen`⟩ – reference download: 2011-09-05

This page presents quintessences concerning the topic 'Open Source' - in form of a FAQ list

ifross: Lizenz-Center; 2011, FreeWeb/HTML ⟨URL: `http://www.ifross.org/lizenz-center`⟩ – reference download: 2011-09-05

This page lists many more (Open Source) Licenses than the OSI itself. It's classifying the Open Source Licenses in those without copyleft-effect, in those with strict copyleft-effect and in those with restricted copyleft-effect - one of the most sophisticated and elaborated considerations!

ifross: Ziele, Aufgaben, Geschichte; 2011, FreeWeb/HTML ⟨URL: `http://www.ifross.org/node/16`⟩ – reference download: 2011-09-05

This page describes the targets and the history of ifross, the 'Institut für Rechtsfragen der Freien und Open Source Software'. It was founded in 2000 as a private institute which shall accompany the phenomenon 'free software' from the viewpoint of (German) lawyers.

ifross et al.: Die GPL kommentiert und erklärt; Beijing, Cambridge, Farnham [etc ..]: O'Reilly, 2005, Print, ISBN 3–89721–389–3

This book explains the legal implications and meaning of each GPL section - including the meaning of 'derivative work'

Imhorst, Christian: Die Anarchie der Hacker. Richard Stallman und die Freie-Software-Bewegung; Marburg: Tectum Verlag, 2004, Print, ISBN 3–8288–8769–4

This book tries - a little willfully - to tell the story of Open Source as a consequence of the American kind of anarchy: basically it mentions the two branches of this history, the personal computer on the one hand (lateron allegedly monopolized by Microsoft) and the university tradition of free collaboration and exchange.

Jaeger, Till a. *Axel Metzger*: Open Source Software. Rechtliche Rahmenbedingungen der Freien Software; 1st edition. München: Verlag C.H. Beck, 2002, Print, ISBN 3406484026

This is the first edition of the most important German standard work concerning Open Source Software Licenses. It deals with many important topics like liability, patents, or branding, And with respect to each of the mentioned licenses (or license clusters) it discusses already the rights and the duties of the software user - but regrettably not in form of a processable todo-list and not with distinguishing the different use cases of the software development process. But nevertheless: it's a very important groundwork!

Jaeger, Till a. *Axel Metzger*: Open Source Software. Rechtliche Rahmenbedingungen der Freien Software; 2nd edition. München: Verlag C.H. Beck, 2006, Print, ISBN 3406538037

This is the second strongly expanded edition of the most important German standard work concerning Open Source Software Licenses. Like ifross it classifies licenses as those with 'strict copyleft clauses', those with 'restricted copyleft-clauses' and those without any 'copyleft-clauses'. Beside other aspects it now discusses how to achieve acceptance of an Open Source licensing and analyzes the AGB side effects. And again it discusses the rights and the duties of the software user, at least with respect to the most import Open Source Licenses - but again regrettably not in form of a processable todo-list and not with distinguishing the different use cases of the software development process. But naturally: it remains a very important groundwork!

Jaeger, Till a. *Axel Metzger*: Open Source Software. Rechtliche Rahmenbedingungen der Freien Software; 3rd edition. München: Verlag C.H. Beck, 2011, Print

This is the third current edition of the most important German standard work concerning Open Source Software Licenses. It retains the 'established structure of the chapters', adds new licenses and discusses - beside many other aspects - the compatibility respectively combinability of different licenses. Hence this work again specifies the rights and the duties of the software user, at least with respect to the most important licenses - but again regrettably not in form of a processable todo-list and not with distinguishing the different use cases of the software development process. But it retains the most important groundwork!

Jansson, Kurt, *Patrick Danowski*, a. *Jakob Voss*: Wikipedia: Kreative Anarchie für den freien

Informations- und Wissensaustausch; In *Drossou, Krempl, a. Poltmann*: Die wunderbare Wissensvermehrung, 2006, pp. 159–167, Print

The article describes shortly the character of 'wikipedia', its' link to the encyclopedists around Diderot and the parallelism of Wikipedia and the Open Source development process,

Kersken, Sasche: Apache 2.2. Das umfassende Handbuch; 3rd, refreshed a. expanded edition; Bonn: Galileo Press, 2009, Print, ISBN 978–8362–1325–7

An Apache compendium with more than 900 pages offering 2 pages concerning the history and the content of the Apache license.

Koglin, Olaf: Opensourcerecht. Die urheber- und schuldrechtlichen Beziehungen zwischen Lizenzgeber und Lizenznehmer bei Open Source Software am Beispiel der General Public License (GPL); Frankfurt am Main: Peter Lang, 2007 (= Schriften zum Wirtschafts- und Medienrecht, Steuerrecht und Zivilprozeßrecht, [Vol./No.] 31), Print, ISBN 978–3–631–56308–3

Thoroughly this book analyzes the validity of GPL with respect to the general German right, the German copyright ('Urheberrecht') and the German law of contract. Additionally it discusses all paragraphs of the GPL and their concret meaning as part of the German right. Other Open Source licenses are not outlined in the same manner.currently it is used more than twice as much as all the others together.

Kreutzer, Till: Software und Spiele kopieren[:] Das Lizenzmodell entscheidet; In *Djordjevic et al.*: Urheberrecht im Alltag, 2008, pp. 29–33, Print

Hints to the German 'Urheberrecht', which predetermines, that in Germany a back-up copy of software can't be forbidden by any licence and that all other types of copying software must explicitly be allowed by the connected licence.

Kreutzer, Till: Software veröffentlichen[:] Wem gehören die Rechte? In *Djordjevic et al.*: Urheberrecht im Alltag, 2008, pp. 163–167, Print

Specifies that 'Urheberrecht' and 'Verwertungsbefugnis' are initially linked in Germany. The right to use ('Verwertungsbefugnis') can be assigned to other, even generally: In Germany employed developers assign their right to determine the software use to their employers by signing the contract. But: the German 'Urheberrecht' doesn't include a protection of the embedded ideas.

Kreutzer, Till: Softwarelizenzen - Beispiele[:] Und welche Lizenz nehme ich jetzt? In *Djordjevic et al.*: Urheberrecht im Alltag, 2008, pp. 176–179, Print

Offers a case based analysis concerning the interests of the developer: Open Source licences can be used if he hasn't any further interests or if he wants to get back improvements of other developer or if he wants to support his name; in other cases he should use a proprietary licence. The different Open Source licences are not analyzed in a deeper sense.

Kugler, Petra: Coordinating Innovation: Evidence from Open Source Development; Dissertation; St. Gallen: University of. St. Gallen, 2005, Print

This work focuses on the question, how informal organizations work, how they establish their hierarchies etc. Open Source communities are taken as examples. Roughly spoken their structures are volatily established on the base of domain and/or project knowledge. The book refers to Open Source licenses only and shortly as a method to found the free exchange of code. How to fulfill the license exactly is out of scope.

Käs, Simone: Rethinking industry practice. The emergence of openness in the embedded component industry; München: Pro BUSINESS, 2008, Print, ISBN 978–3–86805–256–5

This book analyzes the change to open software developement by interviewing 'embeddedlinux'-companies. The result is that openness is required by the customers and that practicing openness evokes a process of learning. The book offers a short but tellingly survey of OSS and their licenses, although it doesn't offer a systematical review of the obligations established by the different licenses.

Laroque, Chrstoph, *Andre Döring*, a. *Thorsten Timm*: 'Give or Let Buy': Kritische Überlegungen eines Software-Ingeneurs zur Veröffentlichung von Software als Open-Source-Projekte; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 155–166, Print

This article wants to address some criteria for and against the publication of source code, explicitly in an 'at least dimly' form. The article has acchieved its' goals. License challenges are not really mentioned.

Maaß, Christian: Zur Bedeutung des Urheber- und Patanterechts in der quelloffenen Softwareentwicklung; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 41–54, Print

The article mentions that the European copryright protects the 'text', the patent the idea behind the text. In Europe it's possible to get a patent 'for' a piece of software only if the main claim concerns a technical invention. Nevertheless this disturbs the idea of Open Source Software. Finally the article hints to the patent pool established by the OSDL.

McAllister, Neil: Licence to Profit [. Hybrid Open Source Licensing]; in: New Architect and Web Techniques (www.newarchitectmag.com), 8 (2003), p. np., Copy

A very short articel. But it describes how mysql organizes its' dual licensing: they let the external contributors transfer their copyright to MySql AB. By this step the company receives also 'the right to license the (modifed) product under any license (they) wish'.

Moody, Glyn: Die Software-Rebellen. Die Erfolgsstory von Linus Torvalds und Linux; transl. from the American [edition, 2000] by Annemarie Pumpering; Landsberg am Lech: verlag moderne industrie, 2001, Print, ISBN 3–478–38730–2

If your friend only wants to read one book on the topic 'Open Source', then give him this! It tells the story - of nearly all aspects.

Mundhenke, Jens: Wettbewerbswirkungen von Open-Source-Software und offenen Standards auf Softwaremärkten; Berlin, Heidelberg, and New York: Springer, 2007 (= Kiel Studies, [Vol./No.] 338), Print, ISBN 978–540–71415–6

This book asks why and how Open Source Software successfully works as part of the economic markets. One answer shall be that OSS increments the competitive pressure by offering an alternative. Fort analyzing the topic the book follows the 'natural' structure of handling Open Source: it explains the concept, classifies the licenses as Copyleft, Noncopyleft, Public Domain and proprietary, and gives a summary of the OSS history.

Netcraft: August 2011 Web Server Survey; 2011, FreeWeb/Html ⟨URL: `http://news.netcraft.com/archives/2011/08/05/august-2011-web-server-survey-3.html`⟩ – reference download: 2011-08-31

Monthly offered statistic counting sites and their delivering web servers: Following this sheet the apache http server is the most often used software since 1996. And currently it is used more than twice as much as all the others together.

Oberhem, Carolina: Vertrags- und Haftungsfragen beim Vertrieb von Open Source Software; Dissertation; Hamburg: Verlag Dr. Kovač, 2008 (= Recht der Neuen Medien, [Vol./No.] 50), Print, ISBN 978–3–8300–4075–0

The book analyzes the liability of authors and distributors of GPL licensed software. This is necessary because in Germany the NO-Warranty clauses of the GPL are not applicable. They are substituted by the rules of 'gifts'. In this sense private authors and distributors are only liable in case of acting deliberately ["vorsätzlich"] or strongly negligently ["grob fahrlässig"]. Additionally commercial distributors must thouroughly check whether a program might damage a customer. But commercial distributors have the full liability for their additionally offered specific services. Beside the 'AGB' character and the typus of GPL as a contract the book refers the idea of Open Source seriously.

Open Source Initiative: The BSD 2-Clause License; 2012 [Year n.st.], FreeWeb/HTML ⟨URL: `http://www.opensource.org/licenses/BSD-2-Clause`⟩ – reference download: 2012-07-03
The latest current BSD license, known as 2 clause BSD license.

Open Source Initiative: The BSD 3-Clause License; 2012 [Year n.st.], FreeWeb/HTML ⟨URL: `http://www.opensource.org/licenses/BSD-3-Clause`⟩ – reference download: 2012-07-04
The elder BSD license, known as 3 clause BSD license.

Open Source Initiative: The Open Source Definition; 2012 [Year n.st], FreeWeb ⟨URL: `http://www.opensource.org/docs/osd`⟩ – reference download: 2012-06-21
This page lists the ten rules which together specify under which conditions a specific license can be classified as Open Source license. It presents the Open Source Definition. So, a piece of software is only then Open Source Software, if its license has been approved as an Open Source License by the Open Source Initiative.

Osterloh, Margit, *Sandra Rota*, a. *Roger Lüthi*: 'Collective Invention' als neues Innovationsmodell; In *Drossou, Krempl, a. Poltmann*: Die wunderbare Wissensvermehrung, 2006, pp. 65–76, Print
The article shows that 'collective invention' can already be discovered in the 19th century. 'Collective invention' denotes the free exchange and development of ideas. In the past it took place in the volatile not productive phase of thinking and stopped when the main architecture / design had been invented. OS development doesn't follow this pattern: its' free exchange is preserved because of the low costs of participation and the OS licenses which demand the 'Giving-Back' of inventions.

Peters, Stormy: Open Source Is Changing the Way Work Gets Done; conference contribution; In *Boldyreff et al.*: Open Source Ecosystems, 2009, p. 1, BibWeb/PDF
Keynote. Affirms that the new methods of collaboration and communication, being used by Open Source development, will also change the cooparation of individuals and companies.

Phillips, Douglas E.: The Software License Unveiled. How Legislation by License Controls Software Access; Oxford, New York, Auckland [etc. ...]: Oxford University Press, 2009, ISBN 978–0–19–534187–4
This book contrasts the proprietary licenses and their Free and/or Open Source alternatives. The conclusion is daring: Proprietary software is described as being focused on the usability (and disregarding the rights of the user), Open Source Software as being focused on the developer (and neglecting the usability).

Piller, Frank T.: User Innovation: der Kunde kann's besser; In *Drossou, Krempl, a. Poltmann*: Die wunderbare Wissensvermehrung, 2006, pp. 85–97, Print
The article describes that the development of new sails for kite-surfing is supported by the end users: Similar to the OS development style they get CAD models freely, let their sails be sewed, and give their improvings as a set of bug reports back to the developer.

Renner, Thomas et al.: Open Source Software. Einsatzpotentiale und Wirtschaftlichkeit; eine Studien der Fraunhofer Gesellschaft; Stuttgart: Fraunhofer IRB Verlag, 2005, Print, ISBN 3–8167–7008–8
The study discusses potentials of OSS and offers a method to compute its' cost effectiveness. It presents a good survey of the underlying concepts and a list of OS applications. For client mashines the computation of the cost effectiveness refers only to the migration from MS-Office to Open-Office. Unfortunately the list of advantages and disadvantages based on interviews is a little inconsistent. Remarkably is also that the study doesn't focus on the act of fulfilling an OS license although it highlights, that OSS is not free of license conditions.

Rose, Marshall T.: The Open Book, A Practical Perspective on OSI; Englewood Cliffs NJ: Prentice Hall, 1990, Print, ISBN 0–13–643016–3
In the most cases 'OSI' refers the 'Open Systems Interconnection Model', not the 'Open

Source Initiative'. Here an English written book explaining the elder meaning of OSI as a model of network layers.

Sebald, Gerd: Offene Wissensökonomie. Analysen zur Wissenssoziologie der Free/Open Source Softwareentwicklung; Dissertation; Wiesbaden: VS Verlag für Sozialwissenschaften, 2008, Print a. BibWeb/PDF, ISBN 978–3–531–15705–4

From the practical viewpoint a withdrawn socialogical book: tries to discuss the conditions for the possibility that an open knowledge economy establishes itself in a captalist environment. OS seems to be reduced to GPL. License questions are discussed on only 3 pages. But at least the book outlines the evolution of the idea 'Free Software'. And it highlights that the forerunner of the GPL - the emacs license - had still required to publish all changings - even the most private improvements - whereas later on the GPL gave up this condition.

Seel, Bernd a. *Miriam Kraft*: Einführung in das Prinzip Open Source; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 9–19, Print

This article is an introduction into the topic of the other articles of the collection. For that purpose it offers a really lean summary of the history and the ideas of Open Source. Because of the shortage the article seems to mislead sometimes.

Spielkamp, Mathias: Creative Commons[:] Andere Zeiten, andere Lizenzen; In *Djordjevic et al.*: Urheberrecht im Alltag, 2008, pp. 219–221, Print

Very short disquisition on the usage of Creative Commons Licences

Spielkamp, Mathias: Lessigletters-Remix[:] Die Creative-Commons-Initiative; In *Djordjevic et al.*: Urheberrecht im Alltag, 2008, pp. 223–230, Print

Describes the history and the targets of the Creative Commons Licence movement: the principle of reciprocity - for software established by the FSF/GPL - is transferred to the world of texts.

Spindler, Gerald; Spindler, Gerald, editor: Rechtsfragen bei Open Source Software; Köln: Verlag Dr. Otto Schmidt KG, 2004, Print, ISBN 3–504–56080–0

This book is based upon a legal opinion. Most of the chapters are written by Spindler himself and deal with the Open Source License types, the German Copyright ('Urheberrecht'), the German contract right ('Vertragsrecht') and the German liability law ('Haftungsrecht'). At the end it contains an excursus dealing with BSD and Mozilla licenses. In all cases the chapters deeply discuss different aspects of the topic.

Splittgerber, Andrea; Schröder, Georg F., editor: Lizenzen und Open Source rechtlicht einwandfrei nutzen. Eine klare Darstellung der Lizenzierung, Nutzungsrechtseinräumung und deren Auswirkung auf Vertragsgestaltung; Kissing: Weka Media, 2005, Print, ISBN 3–8245–1286–3

The book describes the classical licenses detailedly. The Open Source part concerns only the GPL, on only a very few pages.

Stallman, Richard M.: Can You Trust Your Computer? [originally written in 2002]; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 115–117, Print

The article argues that 'DRM' - realized by a 'trusted computing' campaign - is technically established by a key based digital encryption of your data. This encryption is automatically incorporated into your computer (software). But because the keys are kept secret from the owner of the machine one can say that the programs using the hidden keys are controlling the owner, not the owner the computer. Therefore RMS uses the terms 'Digitial Restriction Management' and 'Treacherous Computing'.

Stallman, Richard M.: The Danger of Software Patents; transcript of a speech given at University of Cambridge, London on the 25th of March 2002; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 95–111, Print

Stallman emphasizes that a patent cover ideas and the use of ideas while copyrights only

cover the details of expression of a work. Then he discussed approaches to react on a patent which tries to influence the development.

Stallman, Richard M.: Free Software Definition; originally written in 1996; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 41–43, Print

This article has defined the four constitutive features of free software: the freedom to run a program without any restrictions, the freedom to study how it works, the freedom to redistribute copies of the work and the freedom to improve it and to release the improvements. Additionally the article helps to interpret these features: The accessability to the sourcecode for example is a necessary condition, not a value itself. Or there might be free software which is not copyleft software.

Stallman, Richard M.; Gay, Joshua, editor: Free Software, Free Society: Selected Essays of Richard M. Stallman; [with an] Introduction by Lawrence Lessig; Boston, MA USA: GNU Press, 2002, Print, ISBN 1–882114–98–1

A collection of those important articles by which RMS has established the philosophy of GNU.

Stallman, Richard M.: Free Software: Freedom and Cooperation; transcript of a speech given at New York University on 29 May 2001; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 155–186, Print

This transcript summarizes the main ideas of Free Software and their genesis: it tells the history of RMS.

Stallman, Richard M.: Free Software Needs Free Documentation; originally written in 2000; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 67–68, Print

The article argues that Free Software needs a good free documentation for being practically usable: the lack of good free manuals undermine the freedom to be able to use free software freely.

Stallman, Richard M.: The GNU Manifesto; originally written in 1984; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 31–39, Print

One of the earliest documents delineating the idea of the GNU project and GNU software. The larger part of the article discusses the question that free software offers advantages for all users, even in the context of a commercial use.

Stallman, Richard M.: The GNU Project; originally published in 'Open Sources: Voices from the Open Source Revolution, O'Reilly, 1999'; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 15–30, Print

This article summarizes aspects of the history, the targets, and the philosophy of GNU.

Stallman, Richard M.: The Right to Read; originally written in 1997; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 73–77, Print

From an anticipated viewpoint of the year 2096 Stallman analyzes a dilemma of a student who wants to help his girl friend. But he isn't allowed to help her because in this case it would be necessary to allow her to read his e-books stored on his computer. And that's forbidden and obstructed by a cooperation of the publisher and the computer companies.

Stallman, Richard M.: Selling Free Software; originally written in 1996; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 63–65, Print

This article opposes the myth that you should not charge money for distributing copies of free software. It underlines that - with one exception - the GPL has no requirements about how much you can charge for distributing a copy of free software: Following the GPL you can charge nothing, a penny, a dollar, or a billion dollars.

Stallman, Richard M.: What is Copyleft? originally written in 1996; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 89–90, Print

This article describes Copyleft as a method for making a programm free software and requiring all modified and extended versions of the program to be free software as well: firstly one

states that the code is copyrighted, than - as copyright holder - one allows to use, modify, and redistribute the sourcecode under the condition, that this permission is not deleted.

Stallman, Richard M.: What's in a Name? originally written in 2000; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 51–53, Print

The articles emphasizes to talk about 'GNU/Linux' instead of 'Linux'. The simple name 'Linux' facilitates to combine free and proprietary software so that in the end the GNU campaign for freedom might have been failed. In the opposite the name 'GNU/Linux' preserves and transports the idea of the freedom to use free software in a specific manner, even to those who don't automatically agree with that philosophy.

Stallman, Richard M.: Why 'Free Software' is Better than 'Open Software'; originally written in 1998; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 55–60, Print

This article argues that the concept 'Open Source' was designed not to raise that users deserve freedom. And it underlines that the GNU project sticks to the term 'free software' for spreading the idea of freedom without any fear.

Stallman, Richard M.: Why Software Should Not Have Owners; originally written in 1994; In *Stallman*: Free Software, Free Society: Selected Essays, 2002, pp. 45–49, Print

This article repeats the statement that the society needs programs that people can read, fix, adapt, and improve. Instead of this software owners deliver black boxes that can't be studied. Additionally the article exemplifies some important methods to earn money with free software.

Steinbring, Marc a. *Thorsten Hampel*: Connecting Babbling Bazaars - Der Open-Source-Gedanke im Wandel zum offenen Service; In *Asche et al.*: Open Source. Kommerzialisierungsmöglichkeiten und Chancen für die Zusammenarbeit von Hochschulen und Unternehmen, 2008, pp. 73–97, Print

The article addresses the 'symbiosis' of open structures and the commercial use. 'Open Source' seems to be taken as a spin doctor for the free WEB-2.0.

Suchomski, Bernd: Proprietäres Patentrecht beim Einsatz von Open Source Software. Eine rechtliche Analyse aus unternehmerischer Sicht; Bonn: Tgramedia, 2011 (= Medien Internet und Recht, [Vol./No.] 3), Print, ISBN 978–3–941192–03–4

This book analyzes the meaning of 'gaining' and 'losing' a patent based on OSS: Firstly (in Germany) patents can be registers on the base of OSS. But mostly it's difficult to prevent their registration with a reference to an already existing OS software. Secondly in general a company distributes the right to use its' patents with distributing the OSS: In case of Non-Copyleft software the right is granted implicitly by granting the right to use the software. In case of copyleft software the patent must also be given free because of the obligation to publish the changings. And in case of OSS patent clauses all this is done explicitly.

Taubert, Niels C.: Produktive Anarchie? Netzwerke freier Softwareentwicklung; Bielefeld: transcript, 2006 (= Science Studies), Print, ISBN 3–89942–418–2

This sociological book analyzes, in which sense the idea of 'Open Source' or 'Free Software' determines also the style of a cooperating elaboration. Roughly spoken this kind of collaboration must rather be characterized by argueing and making compromises than being determined by external decisions.

Van den Brande, Ywain, *Shane Coughlan*, a. *Till Jaeger, editors*: The International Free and Open Source Software Law Book; Munich (Germany): Open Source Press, 2011, Print, ISBN 978–3–941841–49–9

Beside a very short historical introduction this book contains legal descriptions of the Open Source context and interpretation in different European countries.

van Wendel de Joode, R., *J. A. de Bruijn*, a. *M. J. G. van Eeten*: Protecting the Virtual Commons. Self-Organizing Open Source and Free Software Communities and Innovative

Intellectual Property Regimes; The Hague: T.M.C. Asser Press, 2003 (= Information Technology & Law Series, [Vol./No.] 3), Print, ISBN 90–6704–159–9
Beside other questions this report wants to answer the question why the communities have created so many different Free / Open Source Software licenses. The answer is perhaps a little thin: the most part of the licenses are derived from BSD or GPL 'to protect the fundamental principles of the communities'.

Viesel, Edward: Freiheit statt Freibier. Geschichte und Praxis der freien digitalen Welt - mit einer Einführung in Linux; Münster: Unrast-Verlag, 2006, Print, ISBN 3–897771–450–7
One half of the book introduces into GNU/Linux, nearly a quarty into DRM, formats, and linked collaborative projects. And the first quarter summarizes the history of Open Source and some ideas of OS licenses.

Widmer, Mike J.: Open Source Software - Urheberrechtliche Aspekte freier Software; Dissertation; Bern: Stämpfli Verlag, 2003, Print
The book explains concepts of Open Source movement and its history competently. It also describes 'Non Copyleft Licenses', 'Copyleft Licenses' and other forms like proprietary or commercial licenses etc. Then it analyzes systematical connections between Open Source Software and the (Europaen/Swiss) 'Urheberrecht' (Copyright). And finally it analyzes - clause by clause - the links from the GPL into the 'Urheberrecht' and vice versa.

wikipedia (en): Free and open source software; n.l., 2011, FreeWeb/HTML (German Version unter http://de.wikipedia.org/wiki/FLOSS) ⟨URL: `http://en.wikipedia.org/wiki/Free_and_open_source_software`⟩ – reference download: 2011-09-08
Illuminates the attempt to (re-)amalgamate the split Free Software movement and Open Source movement at least on the level of meta-concepts by simply linking the majuskels of Free, Libre, Open, Source, Software as hybrid concepts. 'Libre' is introduced to resolve the ambiguity of 'free' in the sense of 'freedom'.

wikipedia (en): MIT License; n.l, 2011, FreeWeb/HTML ⟨URL: `http://en.wikipedia.org/wiki/MIT_License`⟩ – reference download: 2011-09-20
The article explains clearly that there doesn't exist one single MIT-License: one must distinguish between the simpler MIT-expat-License and the more complex MIT-X11-License. But basically the two license denote the same license class. They protect the developer, not the code.

Williams, Sam: Free as in Freedom. Richard Stallman's Crusade for Free Software; Beijing [... etc.]: O'Reilly, 2002, Print, ISBN 0–596–00287–4
This is the first(?) biography of RMS. It contains so many interviews and background information that it might be read as primary source, not only for the history of RMS, but also for the history of the Free Software movement and its internal counterpart, the Open Source movement.