European IPR Helpdesk

Fact Sheet

IPR management in software development

June 2013

Introduction

Software, or computer programs, is a complex asset. At the boundary between pure creations of the mind and technical inventions, multiple Intellectual Property Rights (IPR) can protect it. The intangible nature, diversity of uses, and the various related means available in order to create value with software also has an impact on such a complexity.

Intellectual Property (IP) is an essential tool to secure value generated by software. However, the means to create such value can vary considerably depending on the exploitation scheme chosen and the related ecosystem for which the use of software developments are intended. Business models are then

formalised in a contract which usually takes the form of licence agreements, imposing specific usage rules on third parties intending to exploit the software. Licensing therefore plays an essential role for value creation through intellectual property management.

This fact sheet aims to introduce the various IPR that can protect software, the main licensing schemes available and their respective potential business impact.

1 Intellectual property rights and software development

Computer programs are usually considered to be sequences of instructions written to perform specific tasks in relation to a machine. They can either be embedded in dedicated hardware or be completely versatile code, allowing for data treatments independently of using a specific operating system.

Developing software is essentially a creative task, relying on the coding and development abilities of a developer. However, this creative task relies on the translation of functional purposes meant to be achieved by the software.

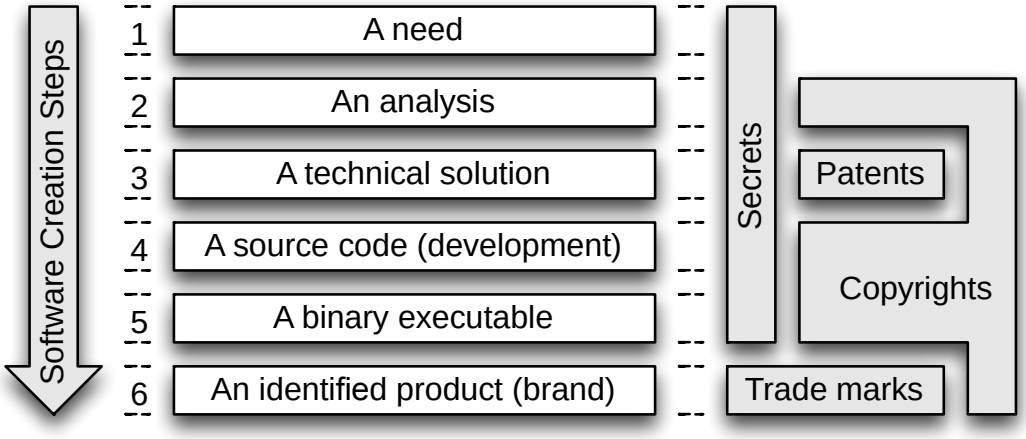

Figure 1

Indeed, a typical software development process starts with the identification of a need, which can be met by a computer (step 1). All the different aspects of the need are thoroughly examined by an analyst in a more or less formal way (step 2). This leads to the formulation of a functional answer in the form of a technical solution, which might include potential patentable elements (step 3). This solution is then translated into a creation of the mind: the source code, i.e. the human-readable version of a computer program (step 4). This source code is then converted into a machine-readable binary executable through a process known as compilation (step 5). This executable application can then be distributed in a given market and be identified by users through the use of a dedicated brand distinct from that of potential competitors (step 6).

Thus, multiple IPR are – or can be – generated in the development of software, as illustrated in Figure 1.

Types of IPR	Registration required	Software components
Copyright	No, as protection is automatic, but complementary solutions are available such as i-DEPOT or public forges[1] in order to further secure rights. In some countries registration is available and can fulfil essential purposes.	All the creative dimensions embedded in software are protected by copyright, provided they are original. Copyright is the historical and most frequently used means of protecting software. Copyrights protect the code as such, but also the user guides and the graphical elements such as icons.
Patent	Yes, under certain conditions[2]	Patents are meant to protect the functional dimension of software, by providing potential ownership of new and inventive technical effects implemented by the program.
Trade marks	It is highly advisable to seek registration.	Protects an essential aspect of software, be it of a visual or textual nature (via either a logo or a word). A trade mark is an essential protection in order to differentiate assets on a given market.
Industrial Design	Registration is generally recommended, even though unregistered designs can be protected	Protects the graphical user interfaces under certain requirements.
Database rights	No	The outputs of the software process can be protected by *sui generis* database protection.
Confidential Information (trade secrets)	No	Specific and identified information can be protected through contractual arrangements.

2 Software licensing

The intellectual property system grants a set of exclusive rights to the owner of intangible assets, such as inventions, brands or designs. This exclusivity allows the owner to exclude others from using its IP assets and, consequently, to grant third parties the rights, more or less extended, to exploit them.

[1] Public forges are internet-accessible software code repositories which are designed to facilitate collaborative developments and diffusion of developed applications. Most are free, such as SourceForge and GitHub, and the use of such solutions allows a clear registration of the date of publication and authorship, both fundamental elements in order to demonstrate ownership of copyrights.

[2] Software patentability is still a debated issue in the European Union given its exclusion as subject matter by Article 52(2)(c) and (3) EPC (European Patent Convention). However, the Enlarged Board of Appeal of the European Patent Office is inclined to its patentability as long as the claim related to a computer program defines or uses technical means (the hardware).

Licensing[3] is a fundamental means of exploiting IPR. It consists of a contract by which a licensor grants a licensee an authorisation to use an identified asset under certain conditions. Licensors can either be the owners of the IPR or act under a mandate from the actual owners.

When granting a licence, licensors are free to determine the extent of exclusive IPR granted on the assets concerned and conversely the rights reserved for themselves. This is also true for software licensing, involving either software as a whole or just a component. The following figure schematically illustrates the different possibilities of reserving such exclusive rights.

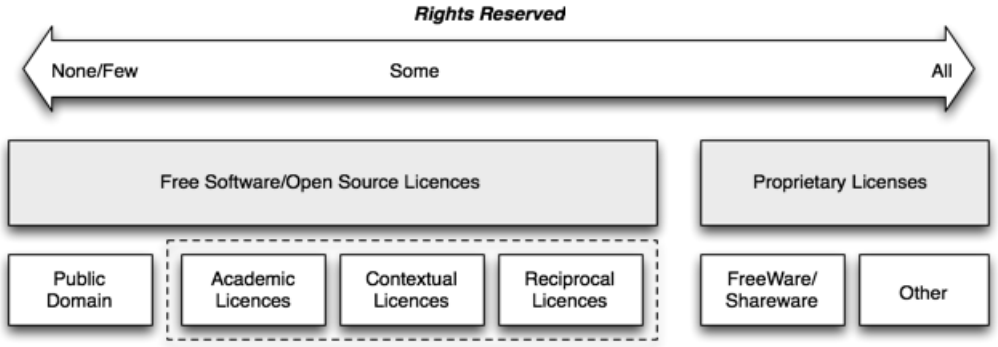

Figure 2

2.1 Free and Open Source Software licenses

Free and Open Source licences fall within the framework of "none" to "some" exclusive IPR reserved[4]. These licences deal with various IPR: they all provide for copyright protection and some for patent and/or trade mark protection as well. Clearly, such licences rely on intellectual property and are in no way opposed to such rights.

Free software licence specificities are defined by the Free Software Foundation[5]. Open Source as such is defined by the Open Source Initiative[6]. It should be noted that although both share similar objectives, their definitions differ in so far as the Free Software Foundation relies on philosophical principles such as the notion of freedom whereas the Open Source Initiative refers to a more business-oriented approach. The Free Software Foundation itself is also the editor and owner of some of the most widely used licenses such as the GNU General Public Licence[7] (GPL), but other licences also exist such as the European Public Licence[8] (EUPL). Both the Open Source Initiative and the Free Software Foundation have a well-established international reputation and any third party licence contracts

[3] Further information on licensing can be found in the dedicated fact sheet available in the European IPR Helpdesk library.
[4] See figure 2.
[5] http://www.fsf.org.
[6] http://www.opensource.org.
[7] http://www.gnu.org/licenses/gpl.html.
[8] https://joinup.ec.europa.eu/software/page/eupl/licence-eupl.

seeking their label must have their seal of approval. For the sake of simplicity, no difference will be further made between these two underlying principles, and reference will only be made to Free and Open Source Software.

2.1.1 Essential criteria for Free and Open Source Software

Free and Open Source software licences must comply with four specific unrestricted criteria on the use to be made of the software, which have to be granted by the licensor to any licensee:

Freedom to run the program

Allow the licensee to run the program, for any purpose, therefore with no restrictions.

Freedom to study and change

Allow the licensee to study how the program works and modify it according to expectations or in order to answer the needs that the software is supposed to meet, with the precondition that access to the source code is given.

Freedom to redistribute

Allow the licensee to redistribute unmodified copies, without restrictions.

Freedom to distribute

Allow the licensee to distribute modified versions (known as derivative or larger works).

Free and Open Source Software developments are usually referred to as component-based developments. The strength of such licensing models relies on sharing of existing source codes in order to improve quality while lowering investment needs without users having to reinvent the wheel. For the purposes of this fact sheet, an original component will be understood as specific to its legal owner, holding all the related IPR. However, mergers and modifications of third party codes can have strong impacts on licensing options, and necessitate an initial short introduction to core notions such as derivative and larger works.

If a pre-existing code belonging to a third party is used to develop software, the underlying licence is likely to impact the derivative creation. In fact, certain provisions in the licensing of the original component can impose conditions applying to the licensing of derivative works. For the purposes of this fact sheet,

derivative creations will include a code modification, instead of the distribution of an unmodified original component.

Larger works are also based on pre-existing components, but they differ from derivative works insofar as they are made up of the use of unmodified original software components.

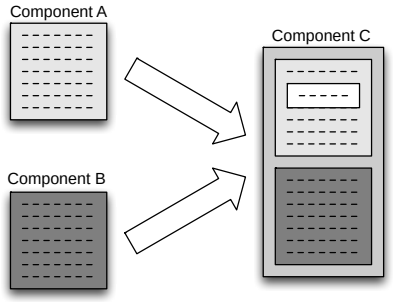

Figure 2

Figure 3 shows the aggregation of two pre-existing software components, namely component A and B, within a third, C.

Component C is made of a derived code based on a modified version of component A, merged with unmodified component B. Thereby, component C is both a derivative work of component A and a larger work of component B.

2.1.2 Categories of Free and Open Source Software licences

2.1.2.1 Academic licences

This category of licences encompasses all contracts, which allow licensees to run, modify, and distribute derivatives of the code under licence with no restrictions whatsoever. Licences over derivative works can in turn not only be granted to sub-licensees but also lead to new licensing terms, including proprietary ones.

Academic licenses are extremely open and therefore are an excellent choice for widespread dissemination of the code under licence where patrimonial restrictions or legal lock-ins on licensees might be at issue. As commonly accepted within the academic world, projects managed under such licensing rules entail proper recognition of authorship.

Examples: BSD[9], MIT[10] licences

2.1.2.2 Reciprocal licences

The reciprocal category encompasses all licences in which licensees are free to run, modify, and distribute derivatives based on the code under licence, as long as the distributed derivatives or larger works are themselves distributed under the same licence.

Reciprocal licences are usually called copyleft licences, as they strongly affect the patrimonial exclusivity granted by copyright. The major advantage of such licences is to secure a common investment as no derivative or larger works can be licensed under another licence. They allow the initial licensor to be provided

[9] http://opensource.org/licenses/BSD-3-Clause.
[10] http://opensource.org/licenses/MIT.

with the very same rights on derivatives as those first provided to licensees with the original code.

Examples: GPL v3, EUPL v1.1 licence

2.1.2.3 Contextual licences

This last category encompasses licences which include a contextual technical trigger, where reciprocal licensing obligations are likely to impact derivative or larger works. Licences of larger works, using an unmodified version of an original component under such a contextual licence, are not restricted by the original licence. However, a derivative product, for which the code of the licensed component is modified, has to be released under the same licence.

Contextual licences are compatible with multiple licensing schemes. Aggregated software based on a reciprocal licensed component is usually not bound by reciprocity of licensing obligations. However, these licences can be extremely complex from a technical standpoint in order to determine whether licensing reciprocity obligations apply.

Example: L-GPL v.2.1 licence[11]

2.2 Proprietary licences

For the purposes of this fact sheet, all licences which are not Free and Open Source licences will be called proprietary licences.

Proprietary licences encompass many different types of contracts which provide many sets of rules on how licensees can use the code under licence. Most proprietary licences require a financial contribution from the licensee in order for him to be allowed to use the software, although some proprietary licences can be different, such as freeware and shareware[12]. Proprietary licensing schemes cannot fall into categories as for the case for Free and Open Source software. In fact, proprietary licensing possibilities are diverse and are only bound by the state of technology and by the creativity of the lawyer drafting the licence clauses. The most common points in proprietary licences are the prohibition of software modifications, a strict control of software use conditions, and no access to the source code. The ultimate factor is their cost.

[11] http://www.gnu.org/licenses/lgpl.html.

[12] Freeware and shareware licensing schemes, which are considered as proprietary licensing, should not be confused with Free and Open Source software licensing frameworks. On the one hand, freeware licences imply that the product used by the end user is available free of charge, but generally these licences prohibit any code modification and focus on price. Moreover, the distribution of such software might be restricted to specific channels or editors. On the other hand, shareware licences imply that the product used by the end user might be freely used for a limited period of time or with limited functionalities. In order to get access to the complete unrestricted product, a complementary licence is required. As for the case of freeware, distribution might be restricted to specific channels or editors, and shareware licensing schemes generally prohibit any code modification.

However, some licensing schemes are fairly classical and will be illustrated below, but note that most of the following examples are mutually exclusive.

2.2.1 Licensing to a specific individual (End User Licensing)

> *"Subject to the restrictions (…), the primary user of the Computer on which the Software is installed (…) may install a second copy of the Software for his or her exclusive use on either a portable Computer or a Computer located at his or her home, provided that the Software on the portable or home Computer is not used at the same time as the Software on the primary Computer"* - Source: Adobe Photoshop CS6 EULA

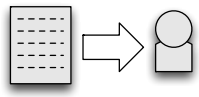

Figure 3

The software is licensed to a specific individual for his own use. The use of licensed software cannot be shared between several users, but the licensed end user may be allowed to install the software on several computers or terminals (though the use might be limited to one computer at a time).

2.2.2 Licensing to a specific hardware (Node Licensing)

> *"You may Install and Access one (…) copy of the Software on one (…) individual Computer (…)"* - Source: AutoCAD 2009 License

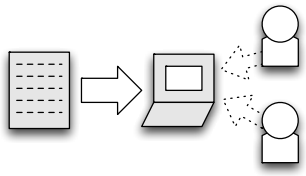

Figure 4

The software is licensed for use on a specific computer. Several individuals may use the software, as long as they share the same hardware, and do not use it at the same time. Multiple licences are required if several computers run the product.

2.2.3 Licensing to a dedicated site (Site Licensing)

> *"At any time (…) a Registered Affiliate may run for his own benefit as many copies as it chooses"* – Source: Microsoft Select Plus License Program Agreement

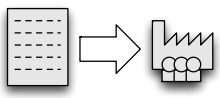

Figure 5

The software is licensed to multiple users or computers under the restriction that they are all located in a dedicated geographical space or in a dedicated company. Restrictions as to the number of users or computers may apply.

2.2.4 Floating licenses (Network Licensing)

> *"The software can be installed on as many computers as needed which are connected to a centralized server location (…) (the) number of licenses purchased determines the number of concurrent users. "* – Source: Trimble's Sketchup Licensing terms

Multiple users may install the product under licence, but only one is allowed to use the software at a given time. This may include the use of a specific technical architecture, such as a licence server, which will authorize access to the application by dedicated users. After each individual use, the server makes the licence available to other users belonging to the same network. Such a licensing solution is often found in industrial applications such as industrial design software where single licence costs are extremely high.

3 IP licensing and other exploitation strategies

Intellectual property management in the field of software requires strategic and complementary use of different types of IPR, as illustrated in the beginning of this fact sheet. Decisions about which IPR is most appropriate to protect software and its components need to be carefully considered, in particular for those requiring important investments such as patents.

Once a decision is taken about the IPR protection of software, exploitation and licensing strategies need to be addressed, which should take into account all of the related costs and market opportunities.

3.1 Why was the software created?

Software creation aimed at generating revenue through licences to individual final users implies specific intellectual property management strategies, completely different from cases where such software is developed in the framework of a scientific project with no intended exploitation strategy. However, an approach focusing only on short term R&D technical challenges might disregard medium and long term advantages of securing IPR, including not only revenue but also abilities to re-use the software in future applications.

A full understanding of an exploitation strategy meant for software is therefore a necessity. A product is developed for a market by way of exchanges, in a commercial or other framework. Whatever the choices made about software exploitation strategies, appropriate intellectual property management should be considered, as decisions relative to such exploitation strategies can evolve with time.

3.2 How was the software created?

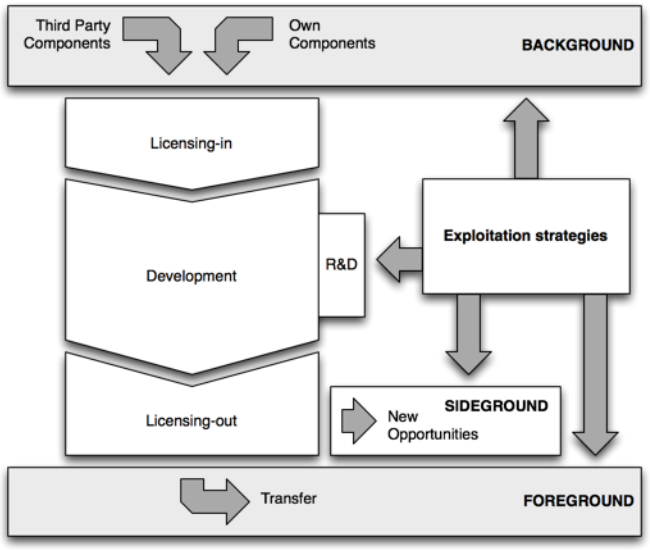

Figure 7

Software can be created by very diverse methods. As stated above, one of the most common methods relies on the re-use of pre-existing third party components.

However, component-based developments might need to overcome legal challenges, as individual licences of different third party software might not be compatible with each other. This is referred to as licensing interoperability management. Interoperability management between licences of third party creations is widespread in any component-based software development, whether based on free and open source software licences or proprietary ones. The following figure illustrates some interoperability difficulties:

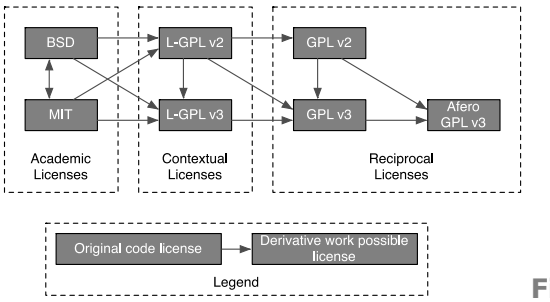

Figure 8

In Figure 8 derivative works based on academic licensed software components can be re-licensed either under the same type of licence or can be upgraded to contextual or compatible reciprocal licences. If necessary, a contextual licensed code can itself be re-licensed by re-using the same licence, upgrading the licence to a newer version remaining within the same contextual field, or by switching to compatible reciprocal licences.

This is a one-way path which requires careful selection of third party components not only for their technical abilities but also for their licensing obligations.

4 General tips

Software licensing is a technical process that has potentially strong exploitation impact, to the extent of potentially jeopardising the business or other exploitation opportunities of a project if not properly considered.

4.1 Developing multiple licensing strategies

When considering the various intellectual property assets that can be embedded in software, multiple licensing strategies can be sought, allowing third parties to use the same software under different licences, depending on their needs and capacities.

More often than not, creating a successful business model for a Free and Open Source software product should be compared to the management of a global product:

- Transforming downloads into installation: Users might download a computer program, but how many will actually install it? In order to increase this percentage, the need for a simple installation process and related utilities should not be underestimated.

- Transforming an installed base into active users: Once software is installed, what percentage of users will actively use it? In order to increase the user base, documentation, support and frequent updates can help – including activities commonly called community management.

- Transforming active users into paying customers: Referring to the number of previous active users, what percentage might need extra services for which they would have to pay? A good marketing incentive is required. Among the active users finally convinced by the business proposal, how many will finally agree to enter into a contractual relationship with a software publisher and end up paying for an extra service?

4.2 Considering the software life cycle

Another common mistake is to forget that licensing is a dynamic process and that software, like all products, has a life cycle. Decisions have to be made as to the possible geographical extension of a given industrial protection, such as a patent, which require specific obligations relative to novelty[13]. Licensing choices might also be adapted to the state of technology or the requirements of users.

4.3 Managing licence compatibilities

Furthermore, Free and Open Source software licence compatibilities must be managed. A very widespread mistake is to overlook potential licensing interoperability issues by focusing solely on technical aspects when developing the software. A good practice to avoid such risk is therefore to establish and make available to all developers a list of licences compatible with the licensing scheme suitable for the end product.

4.4 Documenting and tracking of all components

Code tracking should also be done actively, ideally in real time, and including drawing up the list of third party components used, their licences, authors, acquisition address. Good documentation recording is essential.

Third party contributions to a software development raise the issue of code ownership. Only the publisher-owner of all IPR linked to the software is entitled to make decisions about the management of the related assets. Third party contributions remain the property of their respective authors. Including them in reciprocal Free and Open Source software licensed application locks the end product into such a licensing scheme. Should the publisher wish to change his licensing scheme, it may therefore be to his advantage to require ownership transfer before implementing third party contributions.

4.5 Auditing the source code

Finally, auditing the source code before public release, not only from a code quality but also from a legal standpoint, is a common good practice. This should be done in order to verify thoroughly that all the terms of the respective licences are complied with before releasing derivative or extended software. An audit can be the best tool to avoid potential litigation.

The audit should focus on the proper use of all copyright notices and the full respect of any obligations related to re-used third party components.

[13] You need to seek registration within 6 months for a design or trade mark, and 1 year for a patent, in either another country party of the Paris Convention, or with an international authority such as the OHIM, the EPO or the WIPO, to claim your earlier filing date as a "priority date". This is important to keep the novelty requirement effective.

Useful Resources

For further information on the topic please also see:

- Open Source initiative: http://www.opensource.org
- Free Software Foundation: http://www.fsf.org

GET IN TOUCH

For comments, suggestions or further information, please contact

European IPR Helpdesk
c/o infeurope S.A.
62, rue Charles Martel
L-2134, Luxembourg

Email: service@iprhelpdesk.eu

Phone: +352 25 22 33 - 333

Fax: +352 25 22 33 – 334

ABOUT THE EUROPEAN IPR HELPDESK

The European IPR Helpdesk aims at raising awareness of Intellectual Property (IP) and Intellectual Property Rights (IPR) by providing information, direct advice and training on IP and IPR matters to current and potential participants of EU funded projects. In addition, the European IPR Helpdesk provides IP support to EU SMEs negotiating or concluding transnational partnership agreements, especially through the Enterprise Europe Network. All services provided are free of charge.

Helpline: The Helpline service answers your IP queries within three working days. Please contact us via registration on our website – www.iprhelpdesk.eu – phone or fax.

Website: On our website you can find extensive information and helpful documents on different aspects of IPR and IP management, especially with regard to specific IP questions in the context of EU funded programmes.

Newsletter and Bulletin: Keep track of the latest news on IP and read expert articles and case studies by subscribing to our email newsletter and Bulletin.

Training: We have designed a training catalogue consisting of nine different modules. If you are interested in planning a session with us, simply send us an email at training@iprhelpdesk.eu.

DISCLAIMER

This Fact Sheet has been initially developed under a previous edition of the European IPR Helpdesk (2011-2014). At that time the European IPR Helpdesk operated under a service contract with the European Commission.

From 2015 the European IPR Helpdesk operates as a project receiving funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No 641474. It is managed by the European Commission's Executive Agency for Small and Medium-sized Enterprises (EASME), with policy guidance provided by the European Commission's Internal Market, Industry, Entrepreneurship and SMEs Directorate-General.

Even though this Fact Sheet has been developed with the financial support of the EU, the positions expressed are those of the authors and do not necessarily reflect the official opinion of EASME or the European Commission. Neither EASME nor the European Commission nor any person acting on behalf of the EASME or the European Commission is responsible for the use which might be made of this information.

Although the European IPR Helpdesk endeavours to deliver a high level service, no guarantee can be given on the correctness or completeness of the content of this Fact Sheet and neither the European Commission nor the European IPR Helpdesk consortium members are responsible or may be held accountable for any loss suffered as a result of reliance upon the content of this Fact Sheet.

Our complete disclaimer is available at www.iprhelpdesk.eu.